Advances in Robot Manipulators

Advances in Robot Manipulators

Edited by
Ernest Hall

In-Tech
intechweb.org

Published by In-Teh

In-Teh
Olajnica 19/2, 32000 Vukovar, Croatia

First published April 2010
Printed in India

Technical Editor: Sonja Mujacic
Cover designed by Dino Smrekar

Advances in Robot Manipulators,
Edited by Ernest Hall

p. cm.
ISBN 978-953-307-070-4

Preface

The purpose of this volume in Advances in Robot Manipulators is to encourage and inspire the continual invention of robot manipulators for science and the good of humanity. The concepts of artificial intelligence combined with the engineering and technology of feedback control, have great potential for new, useful and exciting machines. The concept of eclecticism for the design, development, simulation and implementation of a real time controller for an intelligent, vision guided robots is now being explored. The dream of an eclectic perceptual, creative controller that can select its own tasks and perform autonomous operations with reliability and dependability is starting to evolve. We have not yet reached this stage but a careful study of the contents will start one on the exciting journey that could lead to many inventions and successful solutions.

Editor:

Ernest Hall

Contents

A Biomimetic steering robot for Minimally invasive surgery application

G. Chen*
Unilever R&D, Port Sunlight
United Kingdom

M.T. Pham, T. Maalej, H. Fourati,
R. Moreau and S. Sesmat
Laboratoire Ampère, UMR CNRS 5005,
INSA-Lyon, Université de Lyon, F-69621
France

Abstract
Minimally Invasive Surgery represents the future of many types of medical interventions such as keyhole neurosurgey or transluminal endoscopic surgery. These procedures involve insertion of surgical instruments such as needles and endoscopes into human body through small incision/ body cavity for biopsy and drug delivery. However, nearly all surgical instruments for these procedures are inserted manually and there is a long learning curve for surgeons to use them properly. Many research efforts have been made to design active instruments (endoscope, needles) to improve this procedure during last decades. New robot mechanisms have been designed and used to improve the dexterity of current endoscope. Usually these robots are flexible and can pass the constrained space for fine manipulations. In recent years, a continuum robotic mechanism has been investigated and designed for medical surgery. Those robots are characterized by the fact that their mechanical components do not have rigid links and discrete joints in contrast with traditional robot manipulators. The design of these robots is inspired by movements of animals' parts such as tongues, elephant trunks and tentacles. The unusual compliance and redundant degrees of freedom of these robots provide strong potential to achieve delicate tasks successfully even in cluttered and unstructured environments. This chapter will present a complete application of a continuum robot for Minimally Invasive Surgery of colonoscopy. This system is composed of a micro-robotic tip, a set of position sensors and a real-time control system for guiding the exploration of colon. Details will be described on the modeling of the used pneumatic actuators, the design of the mechanical component, the kinematic model analysis and the control strategy for automatically guiding the progression of the device inside the human colon. Experimental results will be presented to check the performances of the whole system within a transparent tube.

* Corresponding author.
gang.chen@unilever.com

1. Introduction

Robotics has increasingly become accepted in the past 20 years as a viable solution to many applications in surgery, particularly in the field of Minimally Invasive Surgery (MIS)Taylor & Stoianovici (2003). Minimally Invasive Surgery represents the future of many types of medical interventions such as keyhole neurosurgery or transluminal endoscopic surgery. These procedures involve insertion of surgical instruments such as needles and endoscopes into human body through small incision/ body cavity for biopsy and drug delivery. However, nearly all surgical instruments for these procedures are inserted manually and they are lack of dexterity in small constrained spaces. As a consequence, there is a long learning curve for surgeons to use them properly and thus risks for patients. Many research efforts have been made to improve the functionalities of current instruments by designing active instruments (endoscope, needles) using robotic mechanisms during the last decades, such as snake robot for throat surgery Simaan et al. (2004) or active cannula Webster et al. (2009). Studies are currently underway to evaluate the value of these new devices. Usually these robots are micro size and very flexible so that they can pass the constrained space for fine manipulations. Furthermore, how to steer these robots into targets safely during the insertion usually needs additional sensors, such as MRI imaging and US imaging, and path planning algorithms are also needed to be developed for the intervention.

Colonoscopy is a typical MIS procedure that needs the insertion of long endoscope inside the human colon for diagnostics and therapy of the lower gastrointestinal tract including the colon. The difficulty of the insertion of colonoscope into the human colon and the pain of the intervention brought to the patient hinders the diagnostics of colon cancer massively. This chapter will present a novel steerable robot and guidance control strategy for colonoscopy interventions which reduces the challenge associated with reaching the target.

1.1 Colonoscopy

Today, colon cancer is an increasing medical concern in the world, where the second frequent malignant tumor is found in industrialized countries Dario et al. (1999). There are several different solutions to detect this kind of cancer, but only colonoscopy can not only make diagnostics, but make therapy. Colonoscopy is a procedure which is characterized by insertion of endoscopes into the human colon for inspection of the lower gastrointestinal tract including the colon in order to stop or to slow the progression of the illness. The anatomy of the colon is showed in Fig. 1.

The instrument used for diagnostics and operation of the human colon is called endoscope (also colonoscope) which is about 1.5cm in diameter and from 1.6 to 2 meters in length. Colonoscopy is one of the most technically demanding endoscopic examinations and tends to be very unpopular with patients because of many sharp bends and constrained workspace. The main reason lies in the characteristics of current colonoscopes, which are quite rigid and require the doctor to perform difficult manoeuvres for long insertion with minimal damage of the colon wall Fukuda et al. (1994); Sturges (1993).

1.2 State of the art: Robotic colonoscopy

Since the human colon is a tortuous "tube" with several sharp bends, the insertion of the colonoscope requires the doctor to exert forces and rotations at shaft outside of the patient, thus causing discomfort to the patient. The complexity of the procedure for doctors and the discomfort experienced by the patient of current colonoscopies lead many researchers to choose the automated colonoscopy method. In Phee et al. (1998), the authors proposed the

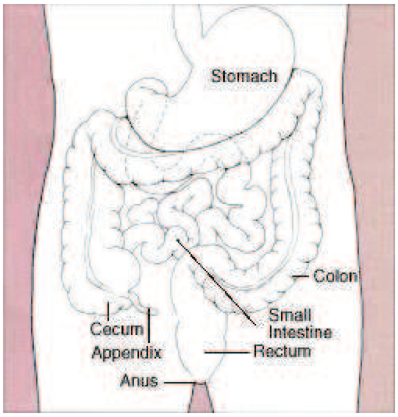


Fig. 1. The anatomy of the colon

concept of automated colonoscopy (also called robotic colonoscopy) from two aspects: locomotion and steering of the distal end, which are the two main actions during a colonoscopy. In order to facilitate the operation of colonoscopy, some studies on the robotic colonoscopy have been carried out from these two aspects. Most current research on autonomous colonoscopies have been focused on the self-propelled robots which utilize various locomotion mechanisms Dario et al. (1997); Ikuta et al. (1988); Kassim et al. (2003); Kumar et al. (2000); Menciassi et al. (2002); Slatkin & Burdick (1995). Among them, inchworm-like locomotion attracted much more attention Dario et al. (1997); Kumar et al. (2000); Menciassi et al. (2002); Slatkin & Burdick (1995). However, most of the current inchworm-based robotic systems Dario et al. (1997); Kumar et al. (2000); Menciassi et al. (2002); Slatkin & Burdick (1995) showed low efficiency of locomotion for exploring the colon because of the structure of the colon wall: slippery and different diameters at each section.Another aspect work that could improve the performance of current colonoscopies is to design an autonomous steering robot for guidance inside the colon during the colonoscopy. Fukuda et al. (1994) proposed Shape Memory Alloy (SMA) based bending devices, called as Micro-Active Catheter (MAC), with two degrees of freedom. With three MACs connected together in series, an angle of bend of nearly 80° is possible. In Menciassi et al. (2002), a bendable tip has been also designed and fabricated by using a silicone bellows with a length of 30mm. It contains three small SMA springs with a 120° layout. This device allows a 90^o bending in three directions. These flexible steering tips are the only parts of the whole self-propelling robots, however those works did not focus on how to control this special robot to endow it with a capability for autonomous guidance Kim et al. (2006); Kumar et al. (2000); Menciassi et al. (2002); Piers et al. (2003). Since 2001, there is another method to perform colon diagnostics: capsule endoscopy (n.d.a;n). With a camera, a light source, a transmitter and power supply integrated into a capsule, the patient can swallow and repel it through natural peristalsis without any pain. Despite capsule endoscopy advantages, it does not allow to perform the diagnostics more thoroughly and actively. Recently, different active locomotion mechanisms have been investigated and designed to address this problem, such as clamping mechanism Menciassi et al. (2005), SMA-based Gorini et al. (2006);

Kim et al. (2005), magnet-based Wang & Meng (2008) locomotion and biomimetic geckoGlass et al. (2008) .

1.3 An approach to steering robot for colonoscopy

The objective of our work in this chapter is original from all the works from other laboratories, which is to design a robot with high dexterity capable of guiding the progression with minimal hurt to the colon wall. Our approach emphasizes a robotic tip with a novel design mounted on the end of the traditional colonoscope or similar instruments. The whole system for semi-autonomous colonoscopy will be presented in this chapter. It is composed of a micro-tip, which is based on a continuum robot mechanism, a proximity multi-sensor system and high level real-time control system for guidance control of this robot. The schema of the whole system, called Colobot, is shown in Fig. 2. Section 2 briefly presents the Colobot and its proximity sensor system. Then section 3 will present model analysis of Colobot system and the validation of kinematic model in section 4. In section 5 guidance control strategy is presented and control architecture and implementation is then described. Finally, experimental results in a colon-like tube will be presented to verify the performance of this semi-autonomous system.

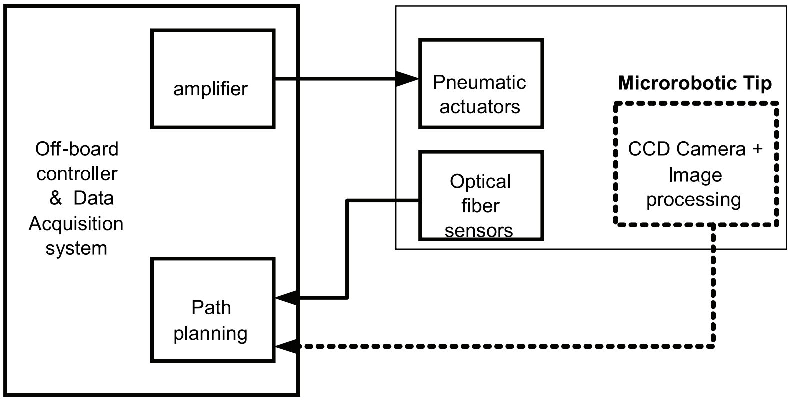


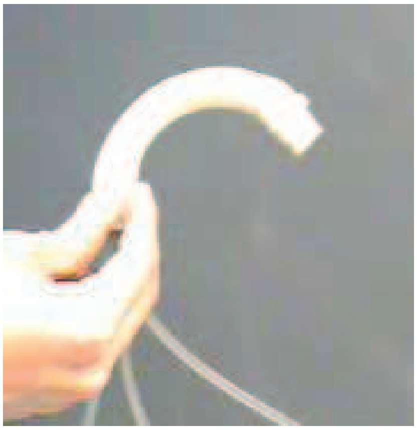
Fig. 2. The scheme of the whole system

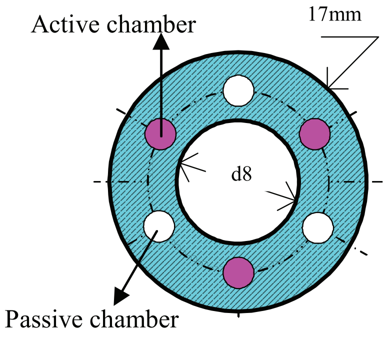
2. Micro-robotic tip: Colobot

Biologically-inspired continuum robots Robinson & Davies (1999) have attracted much interest from robotics researchers during the last decades to improve the capability of manipulation in constrained space. These kinds of systems are characterized by the fact that their mechanical components do not have rigid links and discrete joints in contrast with traditional industry robots. The design of these robots are inspired by movements of animals' parts such as tongues, elephant trunks and tentacles etc. The unusual compliance and redundant degrees of freedom of these robots provide strong potential to achieve delicate tasks successfully even in cluttered and/or unstructured environments such as undersea operations Lane et al. (1999), urban search and rescue, wasted materials handling Immega & Antonelli (1995), Minimally Invasive Surgery Bailly & Amirat (2005); Dario et al. (1997); Piers et al. (2003); Simaan et al. (2004).The Colobot Chen et al. (2006) designed for our work, is a small-scaled continuum robot. Due to the size requirement of the robot, there are challenges on how to miniaturize sensor system integrated into the small-scale robot to implement automatic guidance of progression inside the human colon. This section will present the detailed design of the Colobot and its fibre-optic proximity sensor system.

2.1 Colobot

The difference between our robotic tip and other existing continuum robots is the size. Our design is inspired by pioneer work Suzumori et al. (1992) on a flexible micro-actuator (FMA) based on silicone rubber. Fig. 3(a) shows our design of the Colobot. The robotic tip has 3


(a) Colobot



(b) Cross section of Colobot

Fig. 3. Colobot and its cross section

DOF (Degree of Freedom), which is a unique unit with 3 active pneumatic chambers regularly disposed at 120 degrees apart. These three chambers are used for actuation; three other chambers shown in Fig. 3(b) are designed to optimize the mechanical structure in order to reduce the radial expansion of active chambers under pressure. The outer diameter of the tip is 17 mm that is lesser than the average diameter of the colon. The diameter of the inner hole is 8mm, which is used in order to place the camera or other lighting tools. The weight of the prototype is 20 grams. The internal pressure of each chamber is independently controlled by using pneumatic jet-pipe servovalves. The promising result obtained from the preliminary experiment showed that this tip could bend up to 120° and the resonance frequency is 20 Hz.

2.2 Modeling and experimental characterization of pneumatic servovalves

During an electro-pneumatic control, the follow up of the power transfer from the source to the actuator is achieved through one or several openings with varying cross-section called restrictions: this monitoring organ is the servovalve Sesmat (1996). The Colobot device is provided by three jet pipe micro-servovalves Atchley 200PN *Atchley Controls, Jet Pipe catalogue* (n.d.), which allow the desired modulation of air inside the different active chambers in Fig. 3(b). In this component, a motor is connected to an oscillating nozzle, which deflects the gas stream to one of the two cylinder chambers (Fig. 4(a)). A voltage/current amplifier allows to control the servovalves by the voltage Atchley (1982). A first pneumatic output of this component is directly connected to one of the robot chambers, and a second output is left unconnected. A sensor pressure (UCC model PDT010131) (Fig. 4(b)) is used to measure the pressure in each of the three Colobot robot chambers. The measured pressure, comprised between 0 and 10 bars, was used to determine the servovalve control voltage.

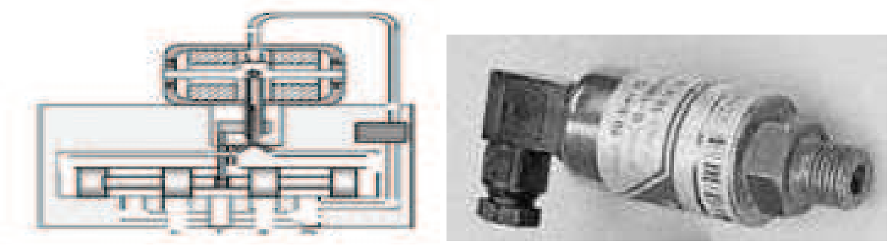

(a) Atchley servovalve 200PN (b) Pressure sensor

Fig. 4. Atchley servovalve and pressure sensor

As the three servo valves used for the COLOBOT actuator are identical, a random servovalve was chosen for the mass flow and pressure characterization. The pressure gain curve is the relationship between the pressure and the current control when the mass flow rate is null. It is performed by means of the pneumatic test bench shown in Fig. 5. A manometer was placed downstream of the servovalve close by the utilization orifice in order to measure the pressures. Fig. 6 shows the pressure measurements P_n and P_p carried out for an increasing and a decreasing input current. It appears that the behavior of the servovalve is quite symmetric but with a hysteresis cycle. Arrival in stop frame couple creates pressure saturation at -18 mA, respectively +18 mA, for the negative current, respectively for the positive current. In the Fig. 5, we substitute the manometer on the test bench for a static mass flow-meter to plot the mass flow rate gain curve (mass flow rate with respect to the input current). This curve presented in Fig. 7 shows a non linear hysteresis.
Because of the specific size of Colobot's chambers, the experimental mass flow rate inside the chamber is very small, the current input and the pressure variations are small enough to neglect the hysteresis and consider linear characteristics for Fig. 6 and Fig. 7.

2.3 Optical Fibre proximity sensors

The purpose of this robotic system is to guide the insertion of the colonoscope through the colon. So it is necessary to integrate the sensors to detect the position of the tip inside the colon. Due to the specific operation environments and the small space constraint, two important criteria must be taken into account to choose the distance sensors:

- the flexibility and size of the colonoscope,
- the cleanliness of the colon wall.

Tests have been performed using ultrasound and magnetic sensors as well as optical fibre. We decided to use optical fibre because of its flexibility, small size, high resolution, and the possibility of reflecting light off the porcine intestinal wall [16].This optical fibre system consists of one emission fibre and a group of four reception fibres (Fig. 8(a)). The light is emitted from a

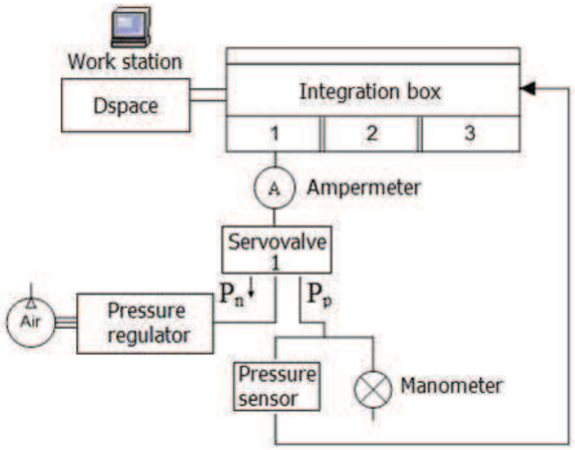


Fig. 5. Pressure gain pneumatic characterization bench

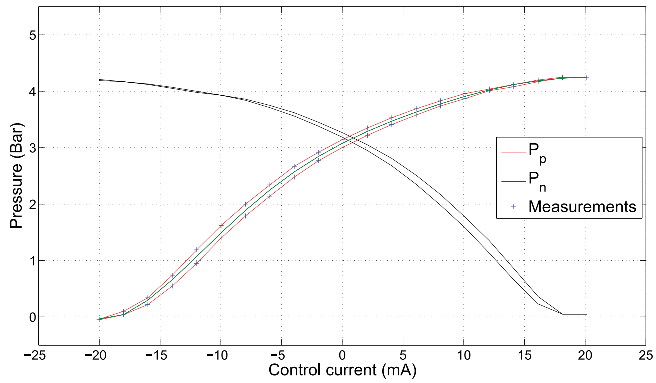


Fig. 6. Pressure gain characterization

cold light source and conveyed by transmission fibres. After reflection on an unspecified body in front of the emission fibre, the reception fibres surrounding the emission fibre detect the reflected light. The amount of reflected light detected is a function of the distance between the sensor and the body. Fig. 8(b) shows the output voltage determined by the distance between the sensor and the porcine intestinal wall. This curve shows that the sensor's resolution is sufficient for detecting the intestinal wall up to 8 mm. Fig. 9 shows the Colobot integrated three fibre optic proximity sensors. The first optical fibre is placed in front of the first pneumatic chamber and the other two in front of their individual pneumatic chambers.

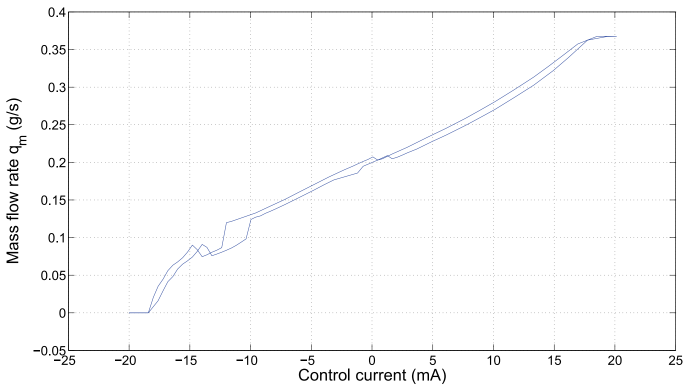
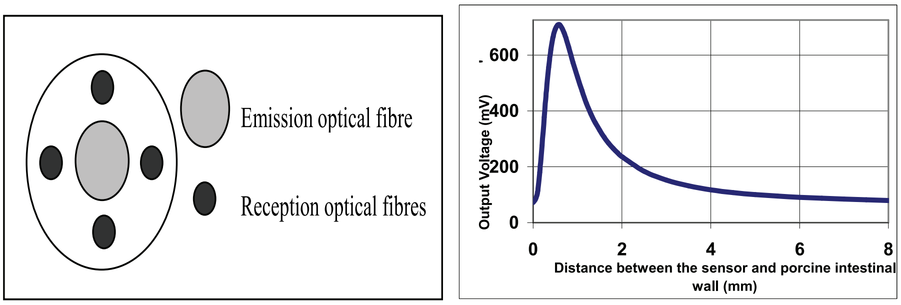


Fig. 7. Mass flow gain characterization



(a) Cross section of the optical fibre proximity sensors (b) Characteristic of the optical fibre sensors

Fig. 8. Proximity sensors and its characterization

3. Kinematic modeling the tip and the proximity sensor system

This section will deal with the kinematic modeling of the robotic tip and the model of the optical fibre sensors.

3.1 Kinematic analysis of the robotic tip

Fig. 10 shows the robot shape parameters and the corresponding frames. The deformation shape of ColoBot is characterized by three parameters as done in our previous prototype EDORA Chen et al. (2005). It is worth to note that Bailly & Amirat (2005); Jones & Walker (2006); Lane et al. (1999); Ohno & Hirose (2001); Simaan et al. (2004); Suzumori et al. (1992) used almost the same set of parameters for the modeling:

- L is the length of the virtual center line of the robotic tip
- α is the bending angle in the bending plane

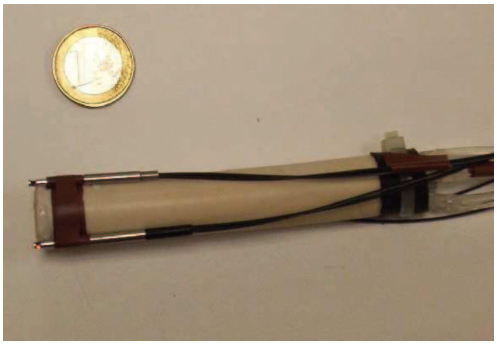


Fig. 9. Prototype integrated with optical fibre proximity sensors

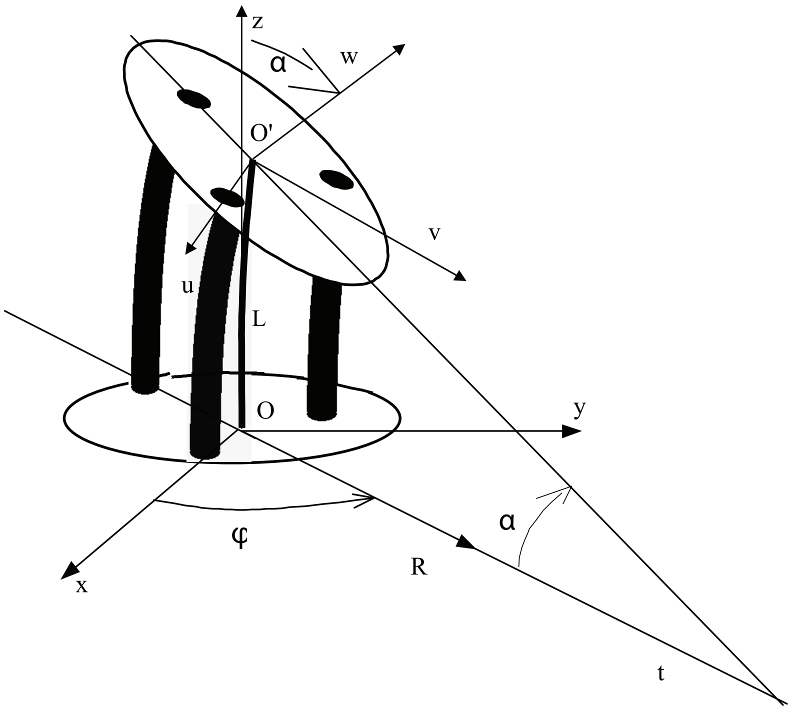


Fig. 10. Kinematic parameters of Colobot

- ϕ is the orientation of the bending plane

The frame R_u (O-xyz) is fixed at the base of the actuator. The X-axis is the one that passed by the center of the bottom end and the center of the chamber 1. The XY-plane defines the plane of the bottom of the actuator, and the z-axis is orthogonal to this plane. The frame R_s (u, v, w) is attached to the top end of the manipulator. So the bending angle α is defined as the angle between the o-z axis and o-w axis. The orientation angle ϕ is defined as the angle between the o-x axis and o-t axis, where o-t axis is the project of o-w axis on the plane x-o-y. Given the assumption that the shape at the bending moment is an arc of a circle, the geometry-based kinematic model Chen et al. (2005) relating the robot shape parameters to the actuator inputs (chamber length) is expressed as follows:

$$\begin{cases} L = \frac{1}{3}\sum_{i=1}^{3} L_i \\ \phi = \text{atan2}\frac{\sqrt{3}(L_2 - L_3)}{L_3 + L_2 - 2L_1} \\ \alpha = \frac{2\sqrt{\lambda_L}}{3r} \end{cases} \tag{1}$$

where $\lambda_L = {L_1}^2 + {L_2}^2 + {L_3}^2 - L_1L_2 - L_2L_3 - L_3L_1$ and r is the radius of the Cobobot and direct kinematic equations with respect to the input pressures are represented by:

$$\begin{cases} L = L_0 + \frac{1}{3}\sum_{i=1}^{3} f_i(P_i) \\ \phi = \text{atan2}\left(\frac{\sqrt{3}(f_2(P_2) - f_3(P_3))}{f_3(P_3) + f_2(P_2) - 2f_1(P_1)}\right) \\ \alpha = \frac{\sqrt{\lambda_p}}{h} \end{cases} \tag{2}$$

where:

$$\lambda_p = f_1(P_1)^2 + f_2(P_2)^2 + f_3(P_3)^2 - f_1(P_1)f_2(P_2) - f_2(P_2)f_3(P_3) - f_3(P_3)f_1(P_1)$$

The function $f_i(P_i)$ $(i = 1, 2, 3)$ shows the relationship relating the stretch length of the chamber to the pressure variation of the silicone-based actuator as described as:

$$\Delta L_i = f_i(P_i) \tag{3}$$

Where ΔL_i $(i = 1, 2, 3)$ is the stretch length of each chamber with corresponding pressure and f_i $(i = 1, 2, 3)$ is a nonlinear function of P_i. The corresponding results can be written as:

$$\begin{cases} \text{if} \quad P_{1min} < P_1 < P_{1max} \\ \Delta L_1 = 37(P_1 - P_{1min})^3 - 54(P_1 - P_{1min})^2 \\ \qquad -9.5(P_1 - P_{1min}) \\ \text{if} \quad P_{2min} < P_2 < P_{2max} \\ \Delta L_2 = -9(P_2 - P_{2min})^3 - 18(P_2 - P_{2min})^2 \\ \qquad -11(P_2 - P_{2min}) \\ \text{if} \quad P_{3min} < P_3 < P_{3max} \\ \Delta L_3 = 0.8(P_3 - P_{3min})^3 - 8.9(P_3 - P_{3min})^2 \\ \qquad -34(P_3 - P_{3min}) \end{cases} \tag{4}$$

where P_{imin} $(i = 1, 2, 3)$ is the threshold of the working point of each chamber and their values equal: $P_{1min} = 0.7\ bar,\ P_{2min} = 0.8\ bar,\ P_{3min} = 0.8\ bar$ and P_{imax} $(i = 1, 2, 3)$ is the maximum pressure that can be applied into each chamber. The detailed deduction of these equations can be found in Chen et al. (2005). The Cartesian coordinates (x, y, z) of the distal end of Colobot in the task space related to the robot bending parameters is obtained through a cylindrical coordinate transformation:

$$\begin{cases} x = \frac{L}{\alpha}(1-\cos\alpha)\cos\phi \\ y = \frac{L}{\alpha}(1-\cos\alpha)\sin\phi \\ z = \frac{L}{\alpha}\sin\alpha \end{cases} \tag{5}$$

And the state-space form of this model is given by:

$$X = f(Q_p) \tag{6}$$

where $X = (\alpha, \phi, L)^T, Q_p = (P_1, P_2, P_3)^T$.

3.2 Modeling and calibration of optical fibre sensors

For the preliminary test of our system, a transparent tube will be used which will be detailed in section 6. So the distance model of the optical fibre sensors with respect to this tube needs to obtained before performing the test. Experimental methods are used to obtain the model of each sensor. The voltage (u_i in volts) with respect to the distance (d_i in mm) between the sensor and the tube wall is measured. Fig. 11 shows the measurements and the approximation model of the third sensor. The model of each sensor is obtained as follows:

$$u_1 = \frac{-40}{3.2d_1^2 + 3} \tag{7}$$

$$u_2 = \frac{-50}{1.6d_2^2 + 2.2} \tag{8}$$

$$u_3 = \frac{-38}{1.7d_3^2 + 2.3} \tag{9}$$

4. Validation of the kinematic model

Since the kinematics of Colobot has been described as the relationship between the deflected shape and the lengths of the three chambers (three pressures of each chamber), the validation of the kinematic model needs to have a sensor to measure the deflected shape, *i.e.* the bending angle, the arc length and the orientation angle. This section first presents sensor choice and its experimental setup for determining these system parameters, and presents the validation of the static kinematic model.

4.1 The sensor choice and experimental setup

For most continuum style robots, the determination of the manipulator shape is a big problem because of the dimension and the inability to mount measurement device for the joint angles. Although there are several technologies that could solve this problem for large size robots Ohno & Hirose (2001), they are difficult to implement on a micro-robot. Since a Cartesian frame has been analyzed with relation to the deflected shape parameters, an indirect

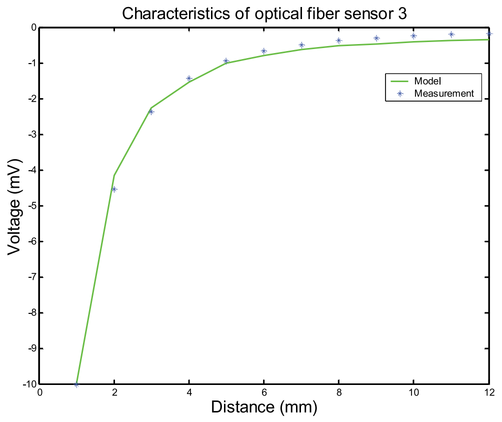


Fig. 11. modeling of optical fiber sensor

method is used to validate the kinematic model with the 3D position measurement. For this purpose, an electromagnetic miniBIRD sensor is used for the experimental validation.
MiniBIRD is a six degree-of-freedom (position and orientation) measuring device from Ascension Technology Corporation (n.d.c). It consists of one or more Ascension Bird electronic units, a transmitter and one or more sensors (Fig. 12). It offers full functionality of other magnetic trackers, with miniaturized sensors as small as 5mm wide. For data acquisition, the

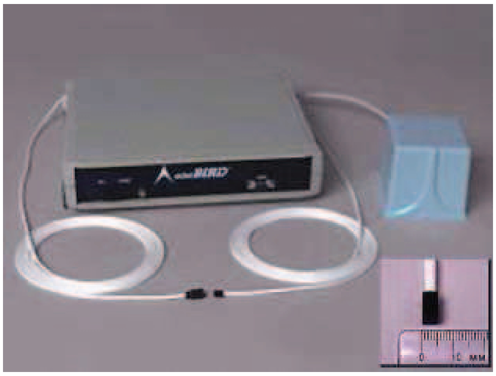


Fig. 12. MiniBIRD 6 DOF magnetic sensor

bottom of Colobot is bounded to a fixture and the sensor is placed on the top of Colobot, shown in Fig. 12. The transmitter is placed at a stationary position. Thus the position and orientation of top-end of Colobot are directly measured from the sensor receiver with relation to the transmitter, and then the position of top-end of the manipulator with relation to the bottom of the manipulator is calculated indirectly through reference transformation.

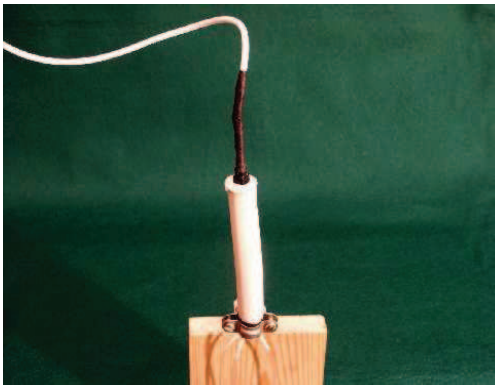


Fig. 13. Measurement configuration

4.2 Validation of the static model

Using the sensor configuration, an open-loop experiment was carried out to validate the static model of the bending angle and the orientation angle (Eq. 2). As for the validation of bending angle, one orientation of Colobot movement is used for validation. The bending angle is directly measured from the miniBIRD sensor and compared with theoretical results from actual pressure obtained from the proportional valves. As shown in Fig. 14, the bending angle concerning the chamber length and the chamber pressure respectively has almost the same characteristics compared with the actual measurements.

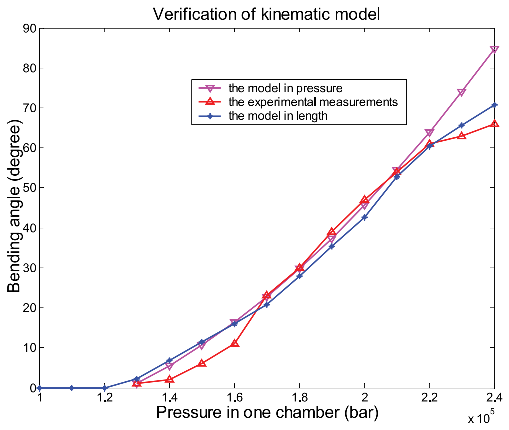


Fig. 14. Comparisons of the bending angle with relation to the chamber length and chamber pressure

To check the orientation angle, the position in the XY frame coordinate of the top-end of Colobot are measured for the six principal manipulator directions. Firstly, expected pressure combinations were used for Colobot to follow the six principal orientation angles ($0°, 60°, 120°, 180°, 240°, 300°$) while the bending angle varied from $0°$ to the maximum. Then the measured positions of the top-end of Colobot were plotted relative to the original position of Colobot without deformation. This experimental protocol leads to Fig. 15. This figure highlights that the six orientation angles are in accordance with the theoretical values except for high pressures in the chambers.

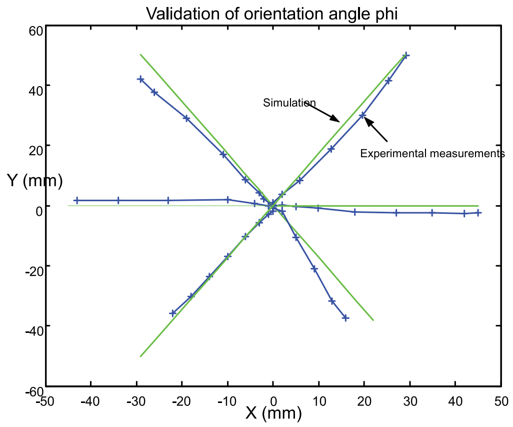


Fig. 15. Comparison of the orientation angle: measurement and simulation

4.3 Verification of the coupling between each chamber

Section 4.2 validated the bending angle and orientation angle separately in static. However, most of the time the motion of the device results from the pressure differentials between each chamber, this is to say, the interaction of each chamber. So it is necessary to check this mutual interaction between each chamber. To achieve this goal, sinus reference signals of pressure with $120°$ delay are applied to each servovalve. They are employed to make Colobot turn around its vertical axis with a constant velocity (see the experimental setup Fig. 13) to see the mutual interaction of each chamber. By using miniBIRD, the endpoint coordinates of Colobot can be obtained in XOY plane. Thus the comparison between these coordinates and those obtained from the simulation of the kinematic model (Eq. 5) allows us to check if there are interactions between chambers on the elongation of the prototype.
Two comparisons are then proposed in Figures 16 and 17. For the first case, three sinus signals of pressure with amplitude of 0.4 bar and an offset of 0.9 bar are applied in the chambers of the prototype. The path of the Colobot's endpoint is a form of triangle (Fig. 16) because these actuators of Colobot work across the threshold of their dead zones. For the latter case, three sinus signals of pressure with amplitude of 0.4 bar and an offset of 1.2 bar are applied in the chamber of Colobot. In this case, Colobot works in the working zone and the endpoint path of Colobot lead to a circular shape (Fig. 17). The lines in the outer layer are the simulation result from the kinematic model relating XY coordinates to the corresponding pressure of each chamber (Eq. 4). Since the characteristics of deformation under pressure is performed each chamber by each chamber independently (Eq. 4), the difference between the results

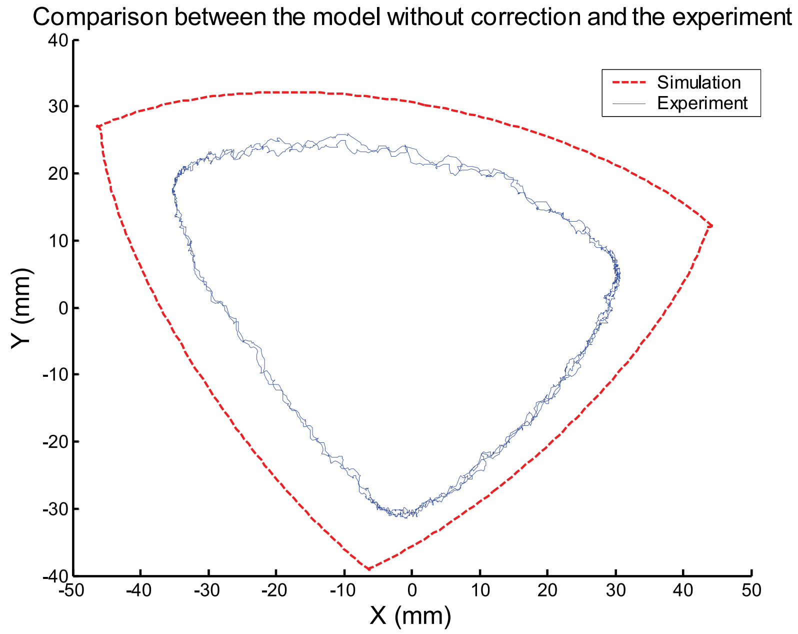


Fig. 16. Simulation et experimental results of the movement of the Colobot's tip (across dead zone)

of simulation and the experimental results showed in Figure 16 and Figure 17 suggests that interactions exist among each chamber. These interactions are taken into account in section 4.4.

4.4 Estimation of a correction parameter

In this section, new parameters are chosen to represent the interactions between each chamber. Thus, six stiffness parameters are introduced to describe the coupling effect of stretching of one chamber to that of other two chambers. Let denotes k_{ij} the mutual stiffness that determines the effect of P_i (i=1,2,3) on the length of the chamber j (j = 1,2,3) (where i does not equal j). The coefficients are obtained by minimizing the difference between the operational coordinates (X_s, Y_s) measured by miniBIRD and the operational coordinates (X_m, Y_m) obtained by simulation of the kinematic model (Fig. 18).

A classical non-linear optimization based on the Levenberg-Marquardt algorithm is proceeded to adjust the unknown parameters k_{ij}. The cost criterion chosen is:

$$J(k_{ij}) = \left\| \sqrt{(X_s)^2 + (Y_s)^2} - \sqrt{k_{ij}(X_m)^2 + k_{ij}(Y_m)^2)} \right\| \tag{10}$$

The numerical results roughly lead to the same coefficient $k = 0.3$ for the unknown parameters k_{ij}. Thus the new expression of the kinematic model is given by:

$$\begin{cases} \Delta L_1 = f_1(P_1) + 0.3(f_2(P_2) + f_3(P_3)) \\ \Delta L_2 = f_2(P_2) + 0.3(f_1(P_1) + f_3(P_3)) \\ \Delta L_3 = f_3(P_3) + 0.3(f_1(P_1) + f_2(P_2)) \end{cases} \tag{11}$$

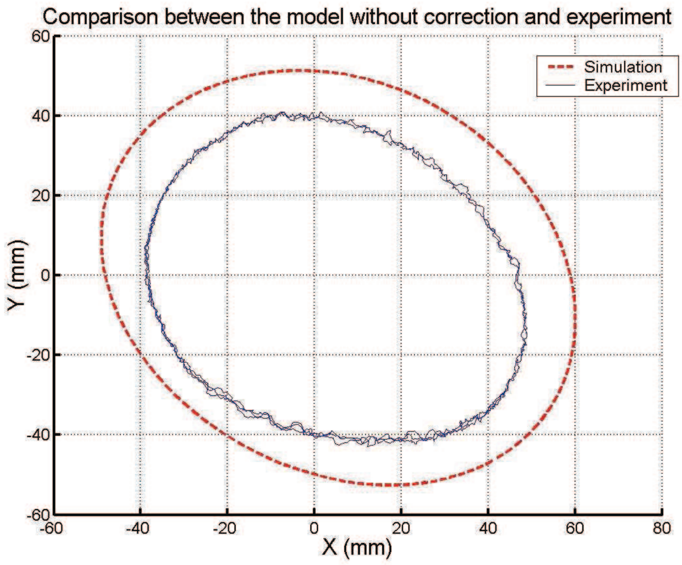


Fig. 17. Simulation and experimental results of the movement of the endpoint of Colobot

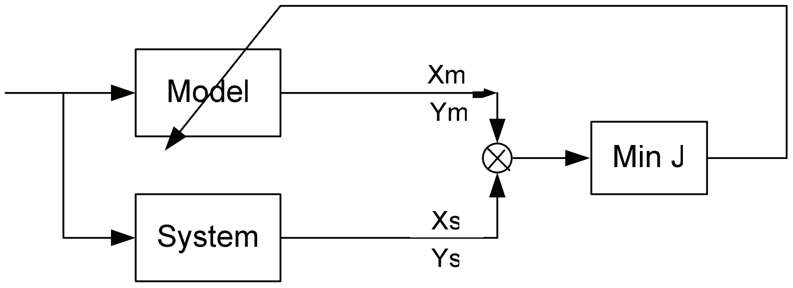


Fig. 18. Optimization model

It is noteworthy that the elongation expression ΔL_1 (respectively ΔL_2, ΔL_3) in (Eq. 11) is equivalent to (Eq. 3) when the relative pressures P_2 and P_3 (respectively P_1, P_3 and P_1, P_2) are equal to zero.
To check this new kinematic model a cross validation is made with three other experiments. Three sinus input pressures with amplitude from 0.1 bar to 0.3 bar are applied into three chambers of Colobot. The improved kinematic model with the correction coefficient k is used to a straightforward comparison with the sets of data. Results shown in Fig. 19 and Fig. 20 are testimony to the behavior of the proposed model in these two cases.

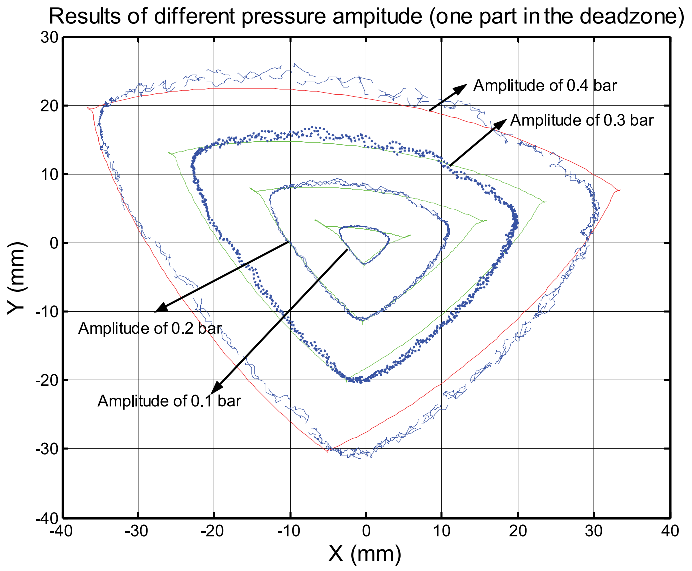


Fig. 19. Verification of corrected model with different pressure inputs (across dead zone)

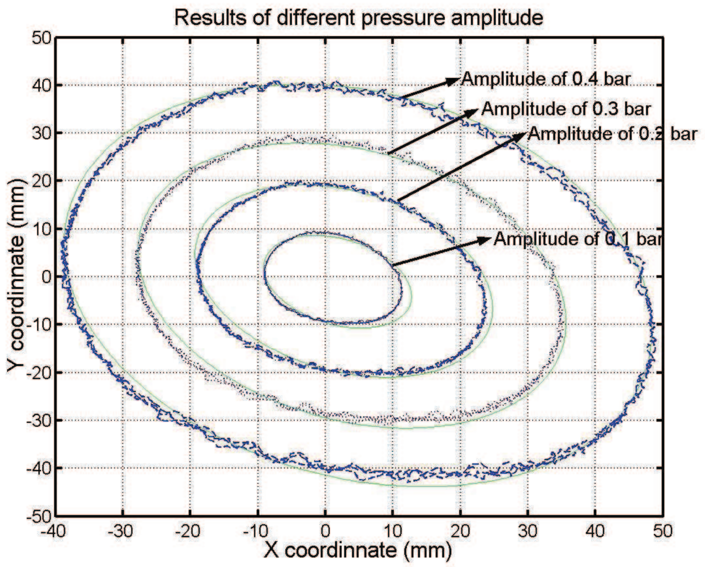


Fig. 20. Validation with different pressure inputs

5. Guidance control strategy based on proximity multi-sensor system

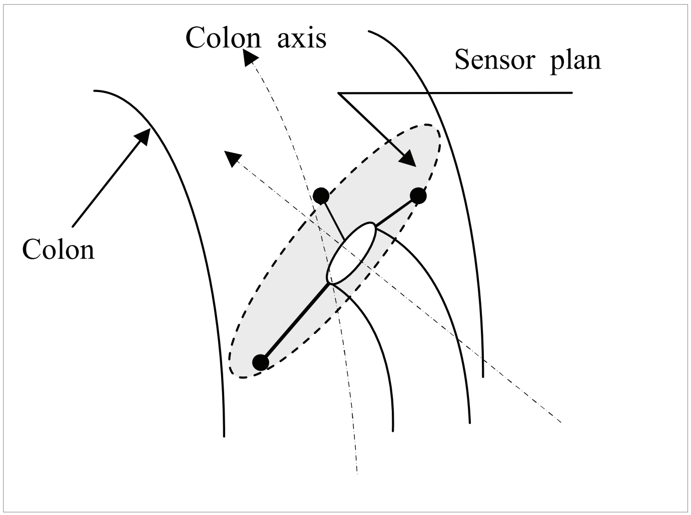


Fig. 21. Position of Colobot inside the colon

5.1 Guidance control strategy

The objective of sensor-based guidance strategy is to calculate the safe position of the distal-end of Colobot compared to the colon wall in real-time based on the measurements of three distance sensors for guidance inside the colon. For the sake of simplicity but without loss of generality, it is assumed that a colon is a cylindrical tube and its cross section is an ellipse at the sensor plane. Fig. 21 illustrates the sensor plane, the distal end of ColoBot and the colon axis. With these assumptions, the safe position will be the central axis of the colon. To approximate the colon axis, a method based on a circumscribed circle is proposed. Since three points D_1, D_2 and D_3 (Fig. 22) of sensor measurements form a triangle, the center of the circumscribed circle of this triangle is chosen as the safe position. This approach iterates as following:

- Three sensor measurements are collected.
- Position P_n in the frame R_s (Fig. 22) is evaluated with these three measurements.
- If P_n is a safe position, then it's necessary to go back to the first step for the next period; otherwise, next the safe position P_{n+1} described in the Frame R_u (Fig. 22) is calculated through the circumscribed method and is provided to the kinematic control for execution.

For more details about the guidance control strategy, please find the reference Chen et al. (2008).

5.2 Guidance control architecture

The control of Colobot is organized in three hierarchical levels, as shown in Fig. 23. The first level consists of local pressure control of each Colobot's chamber through three servovalves.

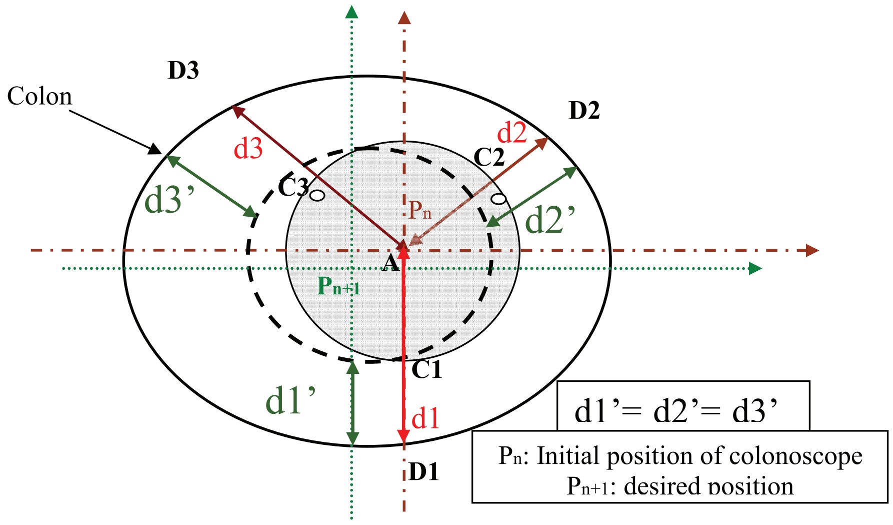


Fig. 22. Computation of the safe position

Three independent PI controllers are used to implement the closed-loop pressure control of the chamber. The position and orientation of Colobot are controlled at level 2 using an instantaneous inverse Jacobian method. This section will describe the implementation detail. Level 3 is the sensor-based planning for automatic navigation described in section 5.1.

5.3 Formulation of task space control of Colobot

After determining the desired trajectory from sensor-based planning, the kinematic control of Colobot will be described in this section. It should be noted that two variables are used to represent the position of Colobot inside the colon. However, the Colobot has 3 degrees of freedom. So this manipulator becomes redundant for the chosen task. The velocity kinematic equations are rewritten as following:

$$\begin{aligned} X &= f(Q_p) \\ \dot{X} &= \frac{\partial X}{\partial(\alpha,\phi,L)} \frac{\partial(\alpha,\phi,L)}{\partial Q_L} \frac{\partial Q_L}{\partial Q_P} \dot{Q}_p \qquad \text{or} \\ \dot{X} &= J_s J_l J_p \dot{Q}_p = J\dot{Q}_p \end{aligned} \tag{12}$$

where $X = (x,y)^T, Q_L = (L_1, L_2, L_3)^T, Q_p = (P_1, P_2, P_3)^T$ and $J = J_s J_l J_p$ is the Jacobian matrix with relation to the three levels of pressure in the chambers.

5.4 Resolution of the inverse kinematics with redundancy

In the case of a redundant manipulator with respect to a given task, the inverse kinematic problem admits infinite solutions. This suggests that redundancy can be conveniently exploited to meet additional constraints on the kinematic control problem in order to obtain greater manipulability in terms of the manipulator configurations and interaction with the environment. A viable solution method is to formulate the problem as a constrained linear

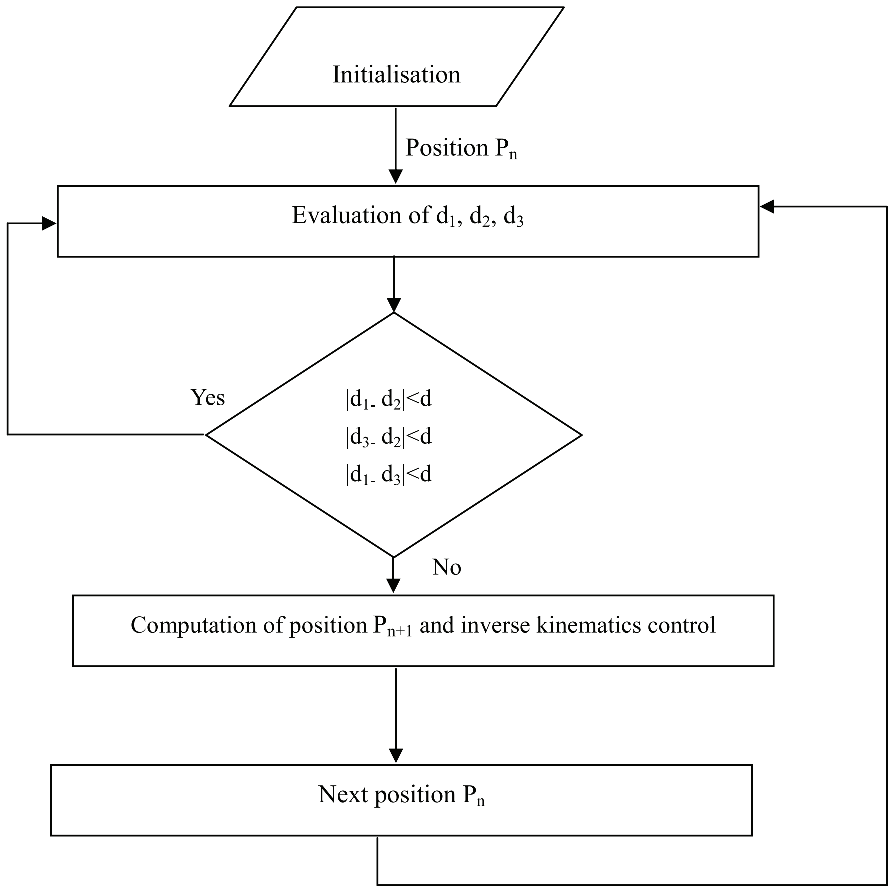


Fig. 23. sensor-based planning and guidance control procedure

optimization problem. Work on resolved-rate control Whitney (1969) proposed to use the Moore-Penrose pseudo inversion of the Jacobian matrix as:

$$\dot{Q}_p = J^+ \dot{X} = (J^T (JJ^T)^{-1}) \dot{X} \tag{13}$$

In our case, however, there is a mechanical limit range for the elongation of each chamber and the corresponding pressure applied into the chamber of the Colobot. The objective function is constructed to be included in the inverse Jacobian algorithms as the second criteria also called the null-space method Hollerbach & Suh (1986); Nakamura (1991).

$$\dot{Q}_p = J^+ \dot{X} + \mu [I - J^+ J] \dot{g} \tag{14}$$

where I is the identity matrix, μ is constant and g is a second criterion for the optimization of the solution. This objective function evaluates the pressure difference between the applied pressure in the chamber and the average pressure applied in the chamber. So the cost function $w(P)$ is expressed as follows:

$$w(P) = \frac{1}{3}\sum_{i=1}^{3}\left(\frac{P_i - P_{iave}}{P_{imax} - P_{imin}}\right)^2 \tag{15}$$

We can then minimize $w(p)$ by choosing:

$$\dot{g} = \text{grad}\left(w(P)\right) = \left\{ \frac{\partial w}{\partial P_1}\frac{\partial w}{\partial P_2}\frac{\partial w}{\partial P_3} \right\} \tag{16}$$

6. Experimental results

This section will present the implementation of the whole system and experimental results of automatic guidance capability of this system in a colon-like tube.

6.1 Hardware implementation

Fig. 24 shows the low-level control system of ColoBot. The pressurized air comes through the compressor (1) and the general pressure is adjusted thanks to the device (2). The pressure in the chambers are controlled by three Jet-pipe servovalves (3a, 3b and 3c). Three pressure sensors (4a, 4b and 4c) are connected between the servovalves and the Colobot (5) for the pressure feedback control. Suitable drivers and amplifiers in the rack (6) were designed to amplify control signals applied to the actuator. A real-time controller is implemented through a DSpace board and coupled with the Real-Time Workshop of Simulink. The Simulink block diagram designed for path planning and kinematics algorithms are expressed with Simulink block diagram which will be compiled as real-time executable under the DSP Processor of the DSpace board. The system runs at 500 Hz for a real-time control loop.

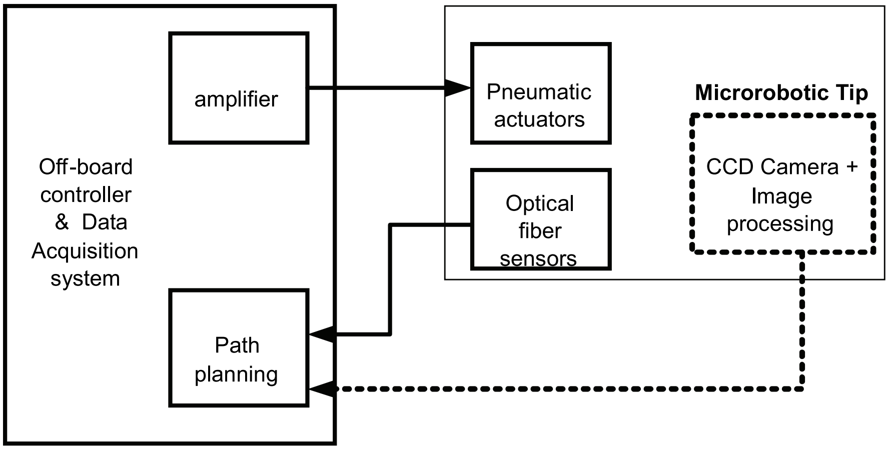


Fig. 24. The implementation of the whole system

6.2 Experimental results in a colon-like tube

A more realistic experiment to test the performance of this semi-autonomous colonoscopy system is to use a colon-like transparent tube to see if Colobot can cross the tube with minimal contact with the tube wall. The diameter of the tube is 26 mm and its length is 50 cm (Fig. 25). For this guidance experiment, the calibration of the optical fibres was adapted to the transparent tube. It is highly probable that results for the distance sensors in a porcine intestine will be similar to those obtained in the human bowel. However, the locomotion of the system is manually operated. The evolution of the measurements of three optical fibres are represented in the left Fig. 26(a). During the entire movement, the distances are never less than 0.8 mm. This demonstrates that the colonoscope tip is moving through the tube without touching it. For a better representation and visualization, Fig. 26(b) shows the extreme positions of the top-end of Colobot as it progresses (with a velocity of 4 cm/s). The position of the Colobot at the centre of the tube is represented by the smallest circle. The larger circle represents the tube wall and the line shows the extreme positions of Colobot. This experiment demonstrates that Colobot has the capability to guide the exploration of the tube with a sensor-based steering control method.

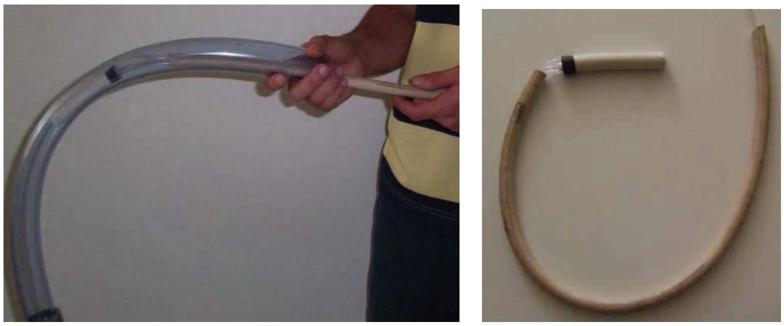
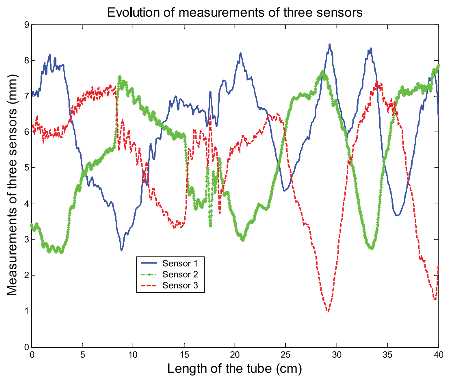


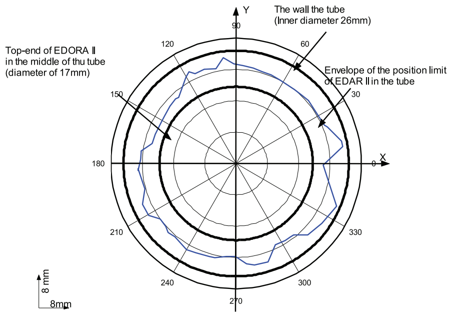
Fig. 25. Guidance control test in a colon-like tube

7. CONCLUSIONS AND FUTURE WORKS

This paper presented a complete robotic system for semi-autonomous colonoscopy. It is composed of a microtip, a proximity multi-sensor system and high level real-time control system for guidance control of this robot. This system was focused on its guidance ability of endoscope inside the human colon with the fiber optic proximity sensors. Colobot is a continuum robot made of silicone rubber. It has three DoF with its outer diameter of 17mm and the weight of 20 gram. The pneumatic actuators of ColoBot are independently driven through three servovalves. The kinematic model of this soft robot was developed based on the geometric deformation and validated its correction. A method using a circumscribed circle is utilized to calculate the safe reference position and orientation of the Colobot. While kinematic-based orientation control used these reference paths to adjust the position of Colobot inside the colon to achieve guidance. Experimental results of guidance control with a transparent tube verified the effectivity of kinematic control and guidance control strategy. In the near future, the proposed method will be tested in a vitro environment.



(a) Evolution of three measurements



(b) Extreme position projected into the tube plane

Fig. 26. Guidance control result analysis

8. References

(n.d.a). *Technical report*.
URL: *http://www.rfnorika.com*

(n.d.b). *Technical report*.
URL: *http://www.givenimaging.com*

(n.d.c). *Technical report*.
URL: *http://www.ascension-tech.com/*

Atchley, R. (1982). A more reliable electrohydraulic servovalve, *Robot VI Conference*, Detroit, USA.

Atchley Controls, Jet Pipe catalogue (n.d.).

Bailly, Y. & Amirat, Y. (2005). Modeling and control of a hybrid continuum active catheter for aortic aneurysm treatment, *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, pp. 924–929.

Chen, G., Pham, M. & Redace, T. (2008). Sensor-based guidance control of a continuum robot for a semi-autonomous colonoscopy, *Robotics and autonomous systems* **57**(6): 712–722.

Chen, G., Pham, M. T. & Redarce, T. (2006). Development and kinematic analysis of a silicone-rubber bending tip for colonoscopy, *IEEE/RSJ Intemational Conference on Intelligent Robots and Systems*, Beijing, China, pp. 168–173.

Chen, G., Pham, M. T., Redarce, T., Prelle, C. & Lamarque, F. (2005). Design and control of an actuator for colonoscopy, *6th International Workshop on Research and Education in Mechatronic*, Annecy, France, pp. 109–114.

Dario, P., Carrozza, M. & Pietrabissa, A. (1999). Development and in vitro tests of a miniature robotic system for computer-assisted colonoscopy, *Jounal of Computer Aided Surgery*, **4**: 4–14.

Dario, P., Paggetti, C., Troisfontaine, N., Papa, E., Ciucci, T., Carrozza, M. & Marcacci, M. (1997). A miniature steerable end-effector for application in an integrated system for computer-assisted arthroscopy, *IEEE International Conference on Robotics and Automation*, Albuquerque, USA, pp. 1573–1579.

Fukuda, T., Guo, S., Kosuge, K., Arai, F., Negoro, M. & Nakabayashi, K. (1994). Micro active catheter system with multi degrees of freedom, *Proceedings of the International Conference on Robotics and Automation*, San Diego, USA, pp. 2290–2295.

Glass, P., Cheung, E. & Sitti, M. (2008). A legged anchoring mechanism for capsule endoscopes using micropatterned adhesives, *IEEE Transactions on Biomedical Engineering* **55**(12): 2759–2767.

Gorini, M., Menciassia, A., Pernorio, G., G., S. & Dario, P. (2006). A novel sma-based actuator for a legged endoscopic capsule, *IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomimetics*.

Hollerbach, J. & Suh, K. (1986). Redundancy resolution of manipulator through torque optimization, *A.I.Memo 882*, Massachussett Institute of Technology.

Ikuta, K., Tsukamoto, M. & Hirose, S. (1988). Shape memory alloy servo actuator system with electric resistance feedback and application for active endoscope, *IEEE International Conference on Robotics and Automation*, Hitachi City, Japan, pp. 427–430.

Immega, G. & Antonelli, K. (1995). The KSI tentacle manipulator, *IEEE International Conference on Robotics and Automation*, Nagoya, Japan, pp. 3149 –3154.

Jones, B. & Walker, I. D. (2006). Kinematics for multi-section continuum robots, *IEEE Transactions on Robotics* **22**(1): 43 –55.

Kassim, I., Ng, W., Feng, G. & Phee, S. (2003). Review of locomotion techniques for robotic colonoscopy, *Proceedings of the International Conference on Robotics and Automation*, Taipei, Taiwan, pp. 1086–1091.

Kim, B., Lim, H. Y., Par, J. H. & Park, J. (2006). Inchworm-like colonoscopic robot with hollow body and steering device, *JSME International Journal Series C* **49**(1): 205–212.

Kim, B., sunghak, L., Heong, P. J. & Jong-oh, P. (2005). Design and fabrication of a locomotive mechanism for capsule-type endos using shape-memory alloys (sma), *IEEE/ASME Transactions on Mechatronics* **10**(1): 77–86.

Kumar, S., Kassim, I. & Asari, V. (2000). Design of a vision- guided microrobotic colonoscopy system, *Advanced robotics* **14**(2): 87–114.

Lane, D., David, J., Robinson, G., OB́rien, D., Sneddon, J., Seaton, E. & A., E. (1999). The amadeus dextrous subsea hand: Design, modeling, and sensor processing, *IEEE Journal of Oceanic engineering* **24**(1): 96–111.

Menciassi, A., J.H., P., Lee, S., Gorini, S., Dario, P. & Park, J. (2002). Robotic solutions and mechanisms for a semi-autonomous endoscope, *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Lausane, Switzerland, pp. 1379–1384.

Menciassi, A., Moglia, A., Gorini, S., Pernorio, G., Stefnini, C. & Dario, P. (2005). Shape memory alloy clamping devices of a capsule for monitoring tasks in the gastrointestinal tract, *Journal of Micromechanics and Microengineering* **15**(11): 2045–2055.

Nakamura, Y. (1991). *Advanced robotics, Redundancy and Optimization*, Addison-Wesley.

Ohno, H. & Hirose, S. (2001). Design of slim slime robot and its gait of locomotion, *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Hawaii, USA, pp. 707–715.

Phee, S., Ng, W. S., Chen, I.-M., Seow-choen, F. & Davies, B. L. (1998). Automation of colonoscopy, part one: Locomotion and steering aspects in automation of colonoscopy, *IEEE Engineering in Medecine and Biology Magazine* **17**(3): 81–89.

Piers, J., Reynaerts, D., Van Brussel, H., De Gersem, G. & Tang, H. T. (2003). Design of an advanced tool guiding system for robotic surgery, *Proceedings of the International Conference on Robotics and Automation*, Taipei, Taiwan, pp. 2651–2656.

Robinson, G. & Davies, J. (1999). Continuum robots - a state of the art, *IEEE International Conference on Robotics and Automation*, Detroit Michigan, USA, pp. 2849–2853.

Sesmat, S. (1996). *Modélisation, Simulation et Commande dúne Servovalve Electropneumatique (in French)*, PhD thesis, INSA de Lyon.

Simaan, N., Taylor, R. & Flint, P. (2004). A dexterous system for laryngeal surgery- multi-backbone bending snake-like slaves for teleoperated dexterous surgical tool manipulation, *IEEE International Conference on Robotics and Automation*, New Orleans, USA, pp. 351–357.

Slatkin, A. B. & Burdick, J. (1995). The development of a robot endoscope, *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Pittsburgh, USA, pp. 3315–3320.

Sturges, R. H. (1993). A flexible, tendon-controlled device for endoscopy, *The International Journal of Robotics Research* **12**(2): 121–131.

Suzumori, K., Iikura, S. & Tannaka, H. (1992). Applying a flexible-micro-actuator robotic mechanisms, *IEEE control systems* **12**(1): 21–27.

Taylor, R. H. & Stoianovici, D. (2003). Medical robotics in computer-integrated surgery, *IEEE Transaction on Robotics and Automation* **19**(5): 765–781.

Wang, X. & Meng, M. Q.-H. (2008). IEEE/RSJ international conference on intelligent robots and systems, Nice, France, pp. 1198–1203.

Webster, R. J. I., Romano, J. M. & Cowan, N. J. (2009). Mechanics of precurved-tube continuum robots, *IEEE Transactions on Robotics* **25**(1): 67 – 78.

Whitney, D. (1969). Resolved motion rate control of manipulators and human prostheses, *IEEE Transaction on Man-Machine systems* **10**(2): 47–53.

A Regressor-free Adaptive Control for Flexible-joint Robots based on Function Approximation Technique

Ming-Chih Chien and An-Chyau Huang
Name of the University (Company)
Country

Abstract
An adaptive controller is presented in this paper to control an n-link flexible-joint manipulator with time-varying uncertainties. The function approximation technique (FAT) is utilized to represent time-varying uncertainties in some finite combinations of orthogonal basis. The tedious computation of the regressor matrix needed in traditional adaptive control is avoided in the new design, and the controller does not require the variation bounds of time-varying uncertainties needed in traditional robust control. In addition, the joint acceleration is not needed in the controller realization. Via the Lyapunov-like stability theory, adaptive update laws are derived to give convergence of the output tracking error. Moreover, the upper bounds of tracking errors in the transient state are also derived. A 2 DOF planar manipulator with flexible joints is used in the computer simulation to verify the effectiveness of the proposed controller.

Keywords: Adaptive control; Flexible-joint robot; FAT

1. INTRODUCTION

In practical applications, most controllers for robot manipulators equipped with harmonic devices are based on rigid-body dynamics formulation. To achieve high precision tracking performance, the joint flexibility should be carefully considered.1 However, the modeling of flexible-joint robots is far more complex than that of rigid-joint robots. Besides, the mathematical model of the robot inevitably contains model inaccuracies such as parametric

M. C. Chien is with the Mechanical and Systems Research Laboratories, Industrial Technology Research Institute, No. 195, Sec. 4, Chung-Hsing Rd., Chutung, Hsinchu, 310, Taiwan, R.O.C. (Tel: +886-3-591-8630/Fax:+886-3-5913607 ,E-mail: D9203401@mail.ntust.edu.tw).
A. C. Huang is with the Department of Mechanical Engineering, National Taiwan University of Science and Technology. No. 43, Keelung Rd., Sec. 4, Taipei, Taiwan, ROC. (Tel:+886-2-27376490, Fax: +886-2-37376460, E-mail: achuang@mail.ntust.edu.tw).

uncertainties, and unmodeled dynamics. Since these inaccuracies may degrade the performance of the closed-loop system, any practical design should consider their effects. Under the problems of joint flexibility and model inaccuracies, several strategies based on adaptive control or robust control for flexible-joint robots had been proposed.

Spong[2,3] proposed an adaptive controller for flexible-joint robots by using the singular perturbation formulation. Chen and Fu[4] presented a two-stage adaptive control scheme for a single-link robot based on a simplified dynamic model. Khorasani[5] designed an adaptive controller using the concept of integral manifolds for n-link flexible-joint robots. Without using the velocity measurements, Lim et. al.[6] proposed an adaptive integrator backstepping scheme for rigid-link flexible-joint robots. Dixon et. al.[7] designed an adaptive partial state feedback controller by using a nonlinear link velocity filter. Yim[8] suggested an output feedback adaptive controller based on the backstepping design. Kozlowski and Sauer[9,10] suggested an adaptive controller under the assumption of bounded disturbances to have semiglobal convergence. Tian and Goldenberg[11] proposed a robust adaptive controller with joint torque feedback. Jain and Khorrami[12] suggested a robust adaptive control for a class of flexible-joint robots that are transformable to a special strict feedback form. However, like most adaptive control strategies, the uncertainties should be linearly parameterizable into regressor form[13]. Availability of the regressor matrix is crucial to the derivation of adaptive controllers for robot manipulators. This is because traditional adaptive control strategies have a common assumption that the uncertain parameters should be constant or slowly time varying. Therefore, the robot dynamics is linearly parameterized into known regressor matrix and an unknown vector with constant parameters. In general, derivation of the regressor matrix for a given robot is tedious. Once it is obtained, we may find that, for most robots, elements in the unknown vector are simple combinations of system parameters such as link mass, link length and moment of inertia, and these are sometimes relatively easy to measure.[13]

Huang and Chen[14] proposed an adaptive backstepping-like controller based on FAT[15-28] for single-link flexible-joint robots with mismatched uncertainties. Similar to most backstepping designs, the derivation is too complex to robots with more joints. In this paper, we would like to propose a FAT based adaptive controller for n-link flexible-joint robots. The tedious computation of the regressor matrix is avoided in the new design. Moreover, the novel controller does not require the variation bounds of time-varying uncertainties needed in traditional robust control. In addition, the control strategy does not need to feedback joint acceleration. Convergence of the output error and the boundedness of all signals are proved using Lyapunov-like direct method with consideration of the effect of the approximation error.

This paper is organized as follows: in section 2, we derive the proposed adaptive controller in detail; section 3 presents simulation results of a 2-D flexible-joint robot using the proposed controller; finally, some conclusions are given in section 4.

2. MAIN RESULTS

The dynamics of an n-rigid link flexible-joint robot can be described by[29]

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{K}(\boldsymbol{\theta} - \mathbf{q}) \tag{1}$$

$$\mathbf{J}\ddot{\boldsymbol{\theta}} + \mathbf{B}\dot{\boldsymbol{\theta}} + \mathbf{K}(\boldsymbol{\theta} - \mathbf{q}) = \mathbf{u} \tag{2}$$

where $\mathbf{q} \in \Re^n$ is the vector of link angles, $\boldsymbol{\theta} \in \Re^n$ is the vector of actuator angles, $\mathbf{u} \in \Re^n$ is the vector of actuator input torques, $\mathbf{D}(\mathbf{q})$ is the $n \times n$ inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ is an n-vector of centrifugal and Coriolis forces, and $\mathbf{g}(\mathbf{q})$ is the gravity vector. $\mathbf{J}$, $\mathbf{B}$ and $\mathbf{K}$ are $n \times n$ constant diagonal matrices of actuator inertias, damping and joint stiffness, respectively. Here, we would like to consider the case when the precise forms of $\mathbf{D}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ and $\mathbf{g}(\mathbf{q})$ are not available and their variation bounds are not given. This implies that traditional adaptive control and robust control cannot be applicable. In the following, we would like to use the function approximation technique to design an adaptive controller for the flexible-joint robot. Moreover, it is well-known that derivation of the regressor matrix for the adaptive control of high DOF rigid robot is generally tedious. For the flexible-joint robot in (1) and (2), its dynamics is much more complex than that of its rigid-joint counterpart. Therefore, the computation of the regressor matrix becomes extremely difficult. Different form the conventional adaptive control schemes for robot manipulators, the proposed FAT-based adaptive controller does not need the computation of the regressor matrix. This largely simplifies the implementation of the control loop.
Define $\boldsymbol{\tau} = \mathbf{K}(\boldsymbol{\theta} - \mathbf{q})$ to be the vector of transmission torques, so (1) and (2) becomes[11]

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \tag{3}$$

$$\mathbf{J}_t\ddot{\boldsymbol{\tau}} + \mathbf{B}_t\dot{\boldsymbol{\tau}} + \boldsymbol{\tau} = \mathbf{u} - \overline{\mathbf{q}}(\dot{\mathbf{q}}, \ddot{\mathbf{q}}) \tag{4}$$

where $\mathbf{J}_t = \mathbf{J}\mathbf{K}^{-1}$, $\mathbf{B}_t = \mathbf{B}\mathbf{K}^{-1}$ and $\overline{\mathbf{q}}(\dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \mathbf{J}\ddot{\mathbf{q}} + \mathbf{B}\dot{\mathbf{q}}$. Define signal vector $\mathbf{s} = \dot{\mathbf{e}} + \boldsymbol{\Lambda}\mathbf{e}$ and $\mathbf{v} = \dot{\mathbf{q}}_d - \boldsymbol{\Lambda}\mathbf{e}$, where $\mathbf{q}_d \in \Re^n$ is the vector of desired states, $\mathbf{e} = \mathbf{q} - \mathbf{q}_d$ is the state error, and $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ with $\lambda_i > 0$ for all i=1, … n. Rewrite (3) in the form

$$\mathbf{D}\dot{\mathbf{s}} + \mathbf{C}\mathbf{s} + \mathbf{g} + \mathbf{D}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} = \boldsymbol{\tau} \tag{5}$$

A. Controller Design for Known Robot
Suppose $\mathbf{D}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ and $\mathbf{g}(\mathbf{q})$ are known, and we may design a proper control law such that $\boldsymbol{\tau}$ follows the trajectory below

$$\boldsymbol{\tau} = \mathbf{g} + \mathbf{D}\dot{\mathbf{v}} + \mathbf{C}\mathbf{v} - \mathbf{K}_d\mathbf{s} \tag{6}$$

where $\mathbf{K}_d$ is a positive definite matrix. Substituting (6) into (5), the closed loop dynamics becomes $\mathbf{D}\dot{\mathbf{s}} + \mathbf{C}\mathbf{s} + \mathbf{K}_d\mathbf{s} = \mathbf{0}$. Define a Lyapunov function candidate as $V = \frac{1}{2}\mathbf{s}^T\mathbf{D}\mathbf{s}$. Its time derivative along the trajectory of the closed loop dynamics can be computed as $\dot{V} = -\mathbf{s}^T\mathbf{K}_d\mathbf{s} + \mathbf{s}^T(\dot{\mathbf{D}} - 2\mathbf{C})\mathbf{s}$. Since $\dot{\mathbf{D}} - 2\mathbf{C}$ can be proved to be skew-symmetric, the above equation becomes $\dot{V} = -\mathbf{s}^T\mathbf{K}_d\mathbf{s} \le 0$. It is easy to prove that $\mathbf{s}$ is uniformly bounded and square integrable, and $\dot{\mathbf{s}}$ is also uniformly bounded. Hence, $\mathbf{s} \to \mathbf{0}$ as $t \to \infty$, or we may say $\mathbf{e} \to \mathbf{0}$ as $t \to \infty$. To make the actual $\boldsymbol{\tau}$ converge to the perfect $\boldsymbol{\tau}$ in (6), let us consider the reference model

$$\mathbf{J}_r\ddot{\boldsymbol{\tau}}_r + \mathbf{B}_r\dot{\boldsymbol{\tau}}_r + \mathbf{K}_r\boldsymbol{\tau}_r = \mathbf{K}_r\boldsymbol{\tau}_d + \mathbf{B}_r\dot{\boldsymbol{\tau}}_d + \mathbf{J}_r\ddot{\boldsymbol{\tau}}_d \tag{7}$$

where $\boldsymbol{\tau}_r \in \Re^n$ is the state vector of the reference model and $\boldsymbol{\tau}_d \in \Re^n$ is the desired states. Matrices $\mathbf{J}_r \in \Re^{n\times n}$, $\mathbf{B}_r \in \Re^{n\times n}$ and $\mathbf{K}_r \in \Re^{n\times n}$ are selected such that $\boldsymbol{\tau}_r \to \boldsymbol{\tau}_d$ exponentially. Define $\overline{\boldsymbol{\tau}}_d(\dot{\boldsymbol{\tau}}_d, \ddot{\boldsymbol{\tau}}_d) = \mathbf{K}_r^{-1}(\mathbf{B}_r\dot{\boldsymbol{\tau}}_d + \mathbf{J}_r\ddot{\boldsymbol{\tau}}_d)$, we may rewrite (4) and (7) in the state space form as

$$\dot{\mathbf{x}}_p = \mathbf{A}_p\mathbf{x}_p + \mathbf{B}_p\mathbf{u} - \mathbf{B}_p\overline{\mathbf{q}} \tag{8}$$

$$\dot{\mathbf{x}}_m = \mathbf{A}_m\mathbf{x}_m + \mathbf{B}_m(\boldsymbol{\tau}_d + \overline{\boldsymbol{\tau}}_d) \tag{9}$$

where $\mathbf{x}_p = [\boldsymbol{\tau} \quad \dot{\boldsymbol{\tau}}]^T \in \Re^{2n}$ and $\mathbf{x}_m = [\boldsymbol{\tau}_r \quad \dot{\boldsymbol{\tau}}_r]^T \in \Re^{2n}$ are augmented state vectors. $\mathbf{A}_p = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{n\times n} \\ -\mathbf{J}_t^{-1} & -\mathbf{J}_t^{-1}\mathbf{B}_t \end{bmatrix} \in \Re^{2n\times 2n}$ and $\mathbf{A}_m = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{n\times n} \\ -\mathbf{J}_r^{-1}\mathbf{K}_r & -\mathbf{J}_r^{-1}\mathbf{B}_r \end{bmatrix} \in \Re^{2n\times 2n}$ are augmented system matrices. $\mathbf{B}_p = \begin{bmatrix} \mathbf{0} \\ \mathbf{J}_t^{-1} \end{bmatrix} \in \Re^{2n\times n}$ and $\mathbf{B}_m = \begin{bmatrix} \mathbf{0} \\ \mathbf{J}_r^{-1}\mathbf{K}_r \end{bmatrix} \in \Re^{2n\times n}$ are augmented input gain matrices, and the pair $(\mathbf{A}_m, \mathbf{B}_m)$ is controllable. Since all system parameters are assumed to be available at the present stage, we may select a controller in the form[30]

$$\mathbf{u} = \Theta\mathbf{x}_p + \Phi\boldsymbol{\tau}_d + \mathbf{h}(\overline{\boldsymbol{\tau}}_d, \overline{\mathbf{q}}) \tag{10}$$

where $\Theta \in \Re^{n\times 2n}$ and $\Phi \in \Re^{n\times n}$ satisfy $\mathbf{A}_p + \mathbf{B}_p\Theta = \mathbf{A}_m$ and $\mathbf{B}_p\Phi = \mathbf{B}_m$, respectively, and $\mathbf{h}(\overline{\boldsymbol{\tau}}_d, \overline{\mathbf{q}}) = \Phi\overline{\boldsymbol{\tau}}_d + \overline{\mathbf{q}}$. Substituting (10) into (8) and after some rearrangements, we may have the system dynamics

$$\dot{\mathbf{x}}_p = \mathbf{A}_m\mathbf{x}_p + \mathbf{B}_m(\boldsymbol{\tau}_d + \overline{\boldsymbol{\tau}}_d) \tag{11}$$

Define $\mathbf{e}_m = \mathbf{x}_p - \mathbf{x}_m$ and we may have the error dynamics directly from (9) and (11)

$$\dot{\mathbf{e}}_m = \mathbf{A}_m\mathbf{e}_m \tag{12}$$

Let $\mathbf{e}_\tau = \boldsymbol{\tau} - \boldsymbol{\tau}_r$ be the output vector of the error dynamics (12) as

$$\mathbf{e}_\tau = \mathbf{C}_m\mathbf{e}_m \tag{13}$$

where $\mathbf{C}_m \in \Re^{n\times 2n}$ is the augmented output matrix such that the pair $(\mathbf{A}_m, \mathbf{C}_m)$ is observable and the transfer function $\mathbf{C}_m(s\mathbf{I} - \mathbf{A}_m)^{-1}\mathbf{B}_m$ is strictly positive real. Since $\mathbf{A}_m$ is stable, (12) implies $\mathbf{e}_m \to 0$ as $t \to \infty$. This further gives $\boldsymbol{\tau} \to \boldsymbol{\tau}_d$ as $t \to \infty$.

B. Controller Design for Uncertain Robot

Suppose $\mathbf{D}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ and **g**(**q**) are not available, and $\ddot{\mathbf{q}}$ is not easy to measure, we would like to design a desired transmission torque $\boldsymbol{\tau}_d$ so that a proper controller **u** can be constructed to have $\boldsymbol{\tau} \to \boldsymbol{\tau}_d$.

Instead of (6), let us design a desired transmission torque $\boldsymbol{\tau}_d$ as

$$\boldsymbol{\tau}_d = \hat{\mathbf{g}} + \hat{\mathbf{D}}\dot{\mathbf{v}} + \hat{\mathbf{C}}\mathbf{v} - \mathbf{K}_d\mathbf{s} \tag{14}$$

where $\mathbf{K}_d > \frac{1}{4}\mathbf{I}_{n\times n}$, and $\hat{\mathbf{D}}$, $\hat{\mathbf{C}}$ and $\hat{\mathbf{g}}$ are estimates of $\mathbf{D}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ and **g**(**q**), respectively. Substituting (14) into (5), we may have the closed loop dynamics

$$\mathbf{D}\dot{\mathbf{s}} + \mathbf{C}\mathbf{s} + \mathbf{K}_d\mathbf{s} = (\boldsymbol{\tau} - \boldsymbol{\tau}_\mathbf{d}) + (\hat{\mathbf{D}} - \mathbf{D})\dot{\mathbf{v}} + (\hat{\mathbf{C}} - \mathbf{C})\mathbf{v} + (\hat{\mathbf{g}} - \mathbf{g}) \tag{15}$$

If a proper controller **u** and update laws for $\hat{\mathbf{D}}$, $\hat{\mathbf{C}}$ and $\hat{\mathbf{g}}$ can be designed, we may have $\boldsymbol{\tau} \to \boldsymbol{\tau}_d$, $\hat{\mathbf{D}} \to \mathbf{D}$, $\hat{\mathbf{C}} \to \mathbf{C}$ and $\hat{\mathbf{g}} \to \mathbf{g}$ so that (15) can give desired performance. Let us consider the control law

$$\mathbf{u} = \Theta\mathbf{x}_p + \Phi\boldsymbol{\tau}_d + \hat{\mathbf{h}} \tag{16}$$

where $\hat{\mathbf{h}}$ is an estimate of $\mathbf{h}$. Substituting (16) into (8), we may have the system dynamics

$$\dot{\mathbf{x}}_p = \mathbf{A}_m\mathbf{x}_p + \mathbf{B}_m(\boldsymbol{\tau}_d + \overline{\boldsymbol{\tau}}_d) + \mathbf{B}_p(\hat{\mathbf{h}} - \mathbf{h}) \tag{17}$$

Together with (9), we may have the error dynamics

$$\dot{\mathbf{e}}_m = \mathbf{A}_m\mathbf{e}_m + \mathbf{B}_p(\hat{\mathbf{h}} - \mathbf{h}) \tag{18}$$

$$\mathbf{e}_\tau = \mathbf{C}_m\mathbf{e}_m \tag{19}$$

If we may design an appropriate update law such that $\hat{\mathbf{h}} \to \mathbf{h}$, then (18) implies $\mathbf{e}_m \to 0$ as $t \to \infty$. This further implies $\boldsymbol{\tau} \to \boldsymbol{\tau}_d$ as $t \to \infty$. Since **D**, **C**, **g** and **h** are functions of time, traditional adaptive controllers are not directly applicable. To design the update laws, let us apply the function approximation representation[15-21]

$$\begin{aligned} \mathbf{D} &= \mathbf{W}_{\mathbf{D}}^T\mathbf{Z}_{\mathbf{D}} + \boldsymbol{\varepsilon}_{\mathbf{D}}, \qquad & \mathbf{C} &= \mathbf{W}_{\mathbf{C}}^T\mathbf{Z}_{\mathbf{C}} + \boldsymbol{\varepsilon}_{\mathbf{C}}, \\ \mathbf{g} &= \mathbf{W}_{\mathbf{g}}^T\mathbf{Z}_{\mathbf{g}} + \boldsymbol{\varepsilon}_{\mathbf{g}}, \qquad & \mathbf{h} &= \mathbf{W}_{\mathbf{h}}^T\mathbf{Z}_{\mathbf{h}} + \boldsymbol{\varepsilon}_{\mathbf{h}} \end{aligned} \tag{20a}$$

where $\mathbf{W}_{\mathbf{D}} \in \Re^{n^2\beta_D \times n}$, $\mathbf{W}_{\mathbf{C}} \in \Re^{n^2\beta_C \times n}$, $\mathbf{W}_{\mathbf{g}} \in \Re^{n\beta_g \times n}$, and $\mathbf{W}_{\mathbf{h}} \in \Re^{n\beta_h \times n}$ are weighting matrices, $\mathbf{Z}_{\mathbf{D}} \in \Re^{n^2\beta_D \times n}$, $\mathbf{Z}_{\mathbf{C}} \in \Re^{n^2\beta_C \times n}$, $\mathbf{Z}_{\mathbf{g}} \in \Re^{n\beta_g \times 1}$, and $\mathbf{Z}_{\mathbf{h}} \in \Re^{n\beta_h \times 1}$ are matrices of basis functions, and $\boldsymbol{\varepsilon}_{(\cdot)}$ are approximation error matrices. The number $\beta_{(\cdot)}$ represents the number of basis functions used. Using the same set of basis functions, the corresponding estimates can also be represented as

$$\begin{aligned} \hat{\mathbf{D}} &= \hat{\mathbf{W}}_{\mathbf{D}}^T\mathbf{Z}_{\mathbf{D}}, \qquad & \hat{\mathbf{C}} &= \hat{\mathbf{W}}_{\mathbf{C}}^T\mathbf{Z}_{\mathbf{C}}, \\ \hat{\mathbf{g}} &= \hat{\mathbf{W}}_{\mathbf{g}}^T\mathbf{Z}_{\mathbf{g}}, \qquad & \hat{\mathbf{h}} &= \hat{\mathbf{W}}_{\mathbf{h}}^T\mathbf{Z}_{\mathbf{h}} \end{aligned} \tag{20b}$$

Define $\tilde{\mathbf{W}}_{(\cdot)} = \mathbf{W}_{(\cdot)} - \hat{\mathbf{W}}_{(\cdot)}$, then equation (15) and (18) becomes

$$\mathbf{D}\dot{\mathbf{s}} + \mathbf{C}\mathbf{s} + \mathbf{K}_d\mathbf{s} = (\boldsymbol{\tau} - \boldsymbol{\tau}_d) - \tilde{\mathbf{W}}_{\mathbf{D}}^T\mathbf{Z}_{\mathbf{D}}\dot{\mathbf{v}} - \tilde{\mathbf{W}}_{\mathbf{C}}^T\mathbf{Z}_{\mathbf{C}}\mathbf{v} - \tilde{\mathbf{W}}_{\mathbf{g}}^T\mathbf{Z}_{\mathbf{g}} + \boldsymbol{\varepsilon}_1 \tag{21}$$

$$\dot{\mathbf{e}}_m = \mathbf{A}_m \mathbf{e}_m - \mathbf{B}_p \tilde{\mathbf{W}}_{\mathbf{h}}^T \mathbf{Z}_{\mathbf{h}} + \mathbf{B}_p \boldsymbol{\varepsilon}_2 \tag{22}$$

where $\boldsymbol{\varepsilon}_1 = \boldsymbol{\varepsilon}_1(\boldsymbol{\varepsilon}_{\mathbf{D}}, \boldsymbol{\varepsilon}_{\mathbf{C}}, \boldsymbol{\varepsilon}_{\mathbf{g}}, \mathbf{s}, \ddot{\mathbf{q}}_d)$ and $\boldsymbol{\varepsilon}_2 = \boldsymbol{\varepsilon}_2(\boldsymbol{\varepsilon}_h, \mathbf{e}_m)$ are lumped approximation errors. Since $\mathbf{W}_{(\cdot)}$ are constant vectors, their update laws can be easily found by proper selection of the Lyapunov-like function. Let us consider a candidate

$$\begin{aligned} V(\mathbf{s}, \mathbf{e}_m, \tilde{\mathbf{W}}_{\mathbf{D}}, \tilde{\mathbf{W}}_{\mathbf{C}}, \tilde{\mathbf{W}}_{\mathbf{g}}, \tilde{\mathbf{W}}_{\mathbf{h}}) &= \frac{1}{2}\mathbf{s}^T \mathbf{D}\mathbf{s} + \mathbf{e}_m^T \mathbf{P}_t \mathbf{e}_m \\ &+ \frac{1}{2} Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T \mathbf{Q}_{\mathbf{D}} \tilde{\mathbf{W}}_{\mathbf{D}} + \tilde{\mathbf{W}}_{\mathbf{C}}^T \mathbf{Q}_{\mathbf{C}} \tilde{\mathbf{W}}_{\mathbf{C}} + \tilde{\mathbf{W}}_{\mathbf{g}}^T \mathbf{Q}_{\mathbf{g}} \tilde{\mathbf{W}}_{\mathbf{g}} + \tilde{\mathbf{W}}_{\mathbf{h}}^T \mathbf{Q}_{\mathbf{h}} \tilde{\mathbf{W}}_{\mathbf{h}}) \end{aligned} \tag{23}$$

where $\mathbf{P}_t = \mathbf{P}_t^T \in \Re^{2n \times 2n}$ is a positive definite matrix satisfying the Lyapunov equation $\mathbf{A}_m^T \mathbf{P}_t + \mathbf{P}_t \mathbf{A}_m = -\mathbf{C}_m^T \mathbf{C}_m$. The matrices $\mathbf{Q}_{\mathbf{D}} \in \Re^{n^2\beta_{\mathbf{D}} \times n^2\beta_{\mathbf{D}}}$, $\mathbf{Q}_{\mathbf{C}} \in \Re^{n^2\beta_C \times n^2\beta_C}$, $\mathbf{Q}_{\mathbf{g}} \in \Re^{n\beta_g \times n\beta_g}$ and $\mathbf{Q}_{\mathbf{h}} \in \Re^{n\beta_h \times n\beta_h}$ are positive definite. The notation *Tr*(.) denotes the trace operation of matrices. The time derivative of *V* along the trajectory of (21) and (22) can be computed as

$$\begin{aligned} \dot{V} &= \mathbf{s}^T \mathbf{D}\dot{\mathbf{s}} + \frac{1}{2}\mathbf{s}^T \dot{\mathbf{D}}\mathbf{s} + \dot{\mathbf{e}}_m^T \mathbf{P}_t \mathbf{e}_m + \mathbf{e}_m^T \mathbf{P}_t \dot{\mathbf{e}}_m \\ &\quad - Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T \mathbf{Q}_{\mathbf{D}} \dot{\hat{\mathbf{W}}}_{\mathbf{D}} + \tilde{\mathbf{W}}_{\mathbf{C}}^T \mathbf{Q}_{\mathbf{C}} \dot{\hat{\mathbf{W}}}_{\mathbf{C}} + \tilde{\mathbf{W}}_{\mathbf{g}}^T \mathbf{Q}_{\mathbf{g}} \dot{\hat{\mathbf{W}}}_{\mathbf{g}} + \tilde{\mathbf{W}}_{\mathbf{h}}^T \mathbf{Q}_{\mathbf{h}} \dot{\hat{\mathbf{W}}}_{\mathbf{h}}) \\ &= -\mathbf{s}^T \mathbf{K}_d \mathbf{s} + \mathbf{s}^T \mathbf{e}_\tau - \mathbf{e}_\tau^T \mathbf{e}_\tau + \mathbf{s}^T \boldsymbol{\varepsilon}_1 + \mathbf{e}_m^T \mathbf{P}_t \mathbf{B}_p \boldsymbol{\varepsilon}_2 \\ &\quad - Tr[\tilde{\mathbf{W}}_{\mathbf{D}}^T (\mathbf{Z}_{\mathbf{D}} \dot{\mathbf{v}} \mathbf{s}^T + \mathbf{Q}_{\mathbf{D}} \dot{\hat{\mathbf{W}}}_{\mathbf{D}}) + \tilde{\mathbf{W}}_{\mathbf{C}}^T (\mathbf{Z}_{\mathbf{C}} \mathbf{v} \mathbf{s}^T + \mathbf{Q}_{\mathbf{C}} \dot{\hat{\mathbf{W}}}_{\mathbf{C}})] \\ &\quad - Tr[\tilde{\mathbf{W}}_{\mathbf{g}}^T (\mathbf{Z}_{\mathbf{g}} \mathbf{s}^T + \mathbf{Q}_{\mathbf{g}} \dot{\hat{\mathbf{W}}}_{\mathbf{g}}) + \tilde{\mathbf{W}}_{\mathbf{h}}^T (\mathbf{Z}_{\mathbf{h}} \mathbf{e}_m^T \mathbf{P}_t \mathbf{B}_p + \mathbf{Q}_{\mathbf{h}} \dot{\hat{\mathbf{W}}}_{\mathbf{h}})] \end{aligned} \tag{24}$$

According to the Kalman-Yakubovic Lemma, we have $\mathbf{e}_m^T \mathbf{P}_t \mathbf{B}_p = \mathbf{e}_\tau^T$ by picking $\mathbf{B}_m = \mathbf{B}_p$ [31]. According to (24), the update laws can be selected as

$$\begin{aligned} &\dot{\hat{\mathbf{W}}}_{\mathbf{D}} = -\mathbf{Q}_{\mathbf{D}}^{-1} \mathbf{Z}_{\mathbf{D}} \dot{\mathbf{v}} \mathbf{s}^T - \sigma_{\mathbf{D}} \hat{\mathbf{W}}_{\mathbf{D}}, \quad \dot{\hat{\mathbf{W}}}_{\mathbf{C}} = -\mathbf{Q}_{\mathbf{C}}^{-1} \mathbf{Z}_{\mathbf{C}} \mathbf{v} \mathbf{s}^T - \sigma_{\mathbf{C}} \hat{\mathbf{W}}_{\mathbf{C}}, \\ &\dot{\hat{\mathbf{W}}}_{\mathbf{g}} = -\mathbf{Q}_{\mathbf{g}}^{-1} \mathbf{Z}_{\mathbf{g}} \mathbf{s}^T - \sigma_{\mathbf{g}} \hat{\mathbf{W}}_{\mathbf{g}}, \qquad \dot{\hat{\mathbf{W}}}_{\mathbf{h}} = -\mathbf{Q}_{\mathbf{h}}^{-1} \mathbf{Z}_{\mathbf{h}} \mathbf{e}_\tau^T - \sigma_{\mathbf{h}} \hat{\mathbf{W}}_{\mathbf{h}} \end{aligned} \tag{25}$$

where $\sigma_{(\cdot)}$ are positive numbers. Then (24) becomes

$$\dot{V} = -[\mathbf{s}^T \quad \mathbf{e}_\tau^T]\mathbf{Q}\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix} + [\mathbf{s}^T \quad \mathbf{e}_\tau^T]\begin{bmatrix}\boldsymbol{\varepsilon}_1\\ \boldsymbol{\varepsilon}_2\end{bmatrix} + \sigma_{\mathbf{D}} Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\hat{\mathbf{W}}_{\mathbf{D}})$$
$$+\sigma_{\mathbf{C}} Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\hat{\mathbf{W}}_{\mathbf{C}}) + \sigma_{\mathbf{g}} Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\hat{\mathbf{W}}_{\mathbf{g}}) + \sigma_{\mathbf{h}} Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\hat{\mathbf{W}}_{\mathbf{h}}) \tag{26}$$

where $\mathbf{Q} = \begin{bmatrix}\mathbf{K}_d & -\frac{1}{2}\mathbf{I}_{n\times n}\\ -\frac{1}{2}\mathbf{I}_{n\times n} & \mathbf{I}_{n\times n}\end{bmatrix}$ is positive definite due to proper selections of $\mathbf{K}_d$ and $\mathbf{K}_c$. Owing to the existence of $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{\varepsilon}_2$ the definiteness of $\dot{V}$ cannot be determined. According to **Appendix** *Lemma A.1* 、*Lemma A.4* and *Lemma A.7*, the right hand side of (26) can be divided into two parts to derive following inequalities

$$-[\mathbf{s}^T \quad \mathbf{e}_\tau^T]\mathbf{Q}\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix} + [\mathbf{s}^T \quad \mathbf{e}_\tau^T]\begin{bmatrix}\boldsymbol{\varepsilon}_1\\ \boldsymbol{\varepsilon}_2\end{bmatrix} \le -\frac{1}{2}\left(\lambda_{\min}(\mathbf{Q})\left\|\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix}\right\|^2 - \frac{1}{\lambda_{\min}(\mathbf{Q})}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1\\ \boldsymbol{\varepsilon}_2\end{bmatrix}\right\|^2\right) \tag{27a}$$

$$Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\hat{\mathbf{W}}_{\mathbf{D}}) \le \frac{1}{2}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}}) \tag{27b}$$

$$Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\hat{\mathbf{W}}_{\mathbf{C}}) \le \frac{1}{2}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}}) \tag{27c}$$

$$Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\hat{\mathbf{W}}_{\mathbf{g}}) \le \frac{1}{2}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}}) \tag{27d}$$

$$Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\hat{\mathbf{W}}_{\mathbf{h}}) \le \frac{1}{2}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}}) \tag{27e}$$

According to (23), we hav

$$V = \frac{1}{2}[\mathbf{s}^T\mathbf{D}\mathbf{s} + \mathbf{e}_m^T\mathbf{P}_t\mathbf{e}_m + Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\mathbf{Q}_{\mathbf{D}}\tilde{\mathbf{W}}_{\mathbf{D}} + \tilde{\mathbf{W}}_{\mathbf{C}}^T\mathbf{Q}_{\mathbf{C}}\tilde{\mathbf{W}}_{\mathbf{C}} + \tilde{\mathbf{W}}_{\mathbf{g}}^T\mathbf{Q}_{\mathbf{g}}\tilde{\mathbf{W}}_{\mathbf{g}} + \tilde{\mathbf{W}}_{\mathbf{h}}^T\mathbf{Q}_{\mathbf{h}}\tilde{\mathbf{W}}_{\mathbf{h}})]$$
$$\le \frac{1}{2}\left[\lambda_{\max}(\mathbf{A})\left\|\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix}\right\|^2 + \lambda_{\max}(\mathbf{Q}_{\mathbf{D}})Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}}) + \lambda_{\max}(\mathbf{Q}_{\mathbf{C}})Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}})\right. \tag{28}$$
$$+\lambda_{\max}(\mathbf{Q}_{\mathbf{g}})Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}}) + \lambda_{\max}(\mathbf{Q}_{\mathbf{h}})Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}})]$$

where $\mathbf{A} = \begin{bmatrix}\mathbf{D} & \mathbf{0}\\ \mathbf{0} & 2\mathbf{C}_m^T\mathbf{P}_t\mathbf{C}_m\end{bmatrix}$. With (27) and (28), (26) can be further written as

$$\begin{aligned}\dot{V} \le -\alpha V + \frac{1}{2}\Bigg\{&\left[\alpha\lambda_{\max}(\mathbf{A}) - \lambda_{\min}(\mathbf{Q})\right]\left\|\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix}\right\|^2 + [\alpha\lambda_{\max}(\mathbf{Q_D}) - \sigma_{\mathbf{D}}]Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}})\\ &+ [\alpha\lambda_{\max}(\mathbf{Q_C}) - \sigma_{\mathbf{C}}]Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}}) + [\alpha\lambda_{\max}(\mathbf{Q_g}) - \sigma_{\mathbf{g}}]Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}})\\ &+ [\alpha\lambda_{\max}(\mathbf{Q_h}) - \sigma_{\mathbf{h}}]Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}}) + \frac{1}{\lambda_{\min}(\mathbf{Q})}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1\\ \boldsymbol{\varepsilon}_2\end{bmatrix}\right\|^2\\ &+ \sigma_{\mathbf{D}}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}}) + \sigma_{\mathbf{C}}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) + \sigma_{\mathbf{g}}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) + \sigma_{\mathbf{h}}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}})\Bigg\}\end{aligned} \tag{29}$$

Although **D** and **L** are unknown, we know that $\exists \overline{D}\text{ and }\underline{D}$ s.t. $\underline{D} \le \|\mathbf{D}\| \le \overline{D}$ 、 $\exists \overline{L}\text{ and }\underline{L}$ s.t. $\underline{L} \le \|\mathbf{L}\| \le \overline{L}$, $\exists \overline{\eta}_A, \underline{\eta}_A > 0$ s.t. $\lambda_{\max}(\mathbf{A}) \le \overline{\eta}_A$ and $\lambda_{\min}(\mathbf{A}) \ge \underline{\eta}_A$ no 32.

Picking $\alpha \le \min\left\{\frac{\lambda_{\min}(\mathbf{Q})}{\overline{\eta}_A}, \frac{\sigma_{\mathbf{D}}}{\lambda_{\max}(\mathbf{Q_D})}, \frac{\sigma_{\mathbf{C}}}{\lambda_{\max}(\mathbf{Q_C})}, \frac{\sigma_{\mathbf{g}}}{\lambda_{\max}(\mathbf{Q_g})}, \frac{\sigma_{\mathbf{h}}}{\lambda_{\max}(\mathbf{Q_h})}\right\}$, then we have

$$\begin{aligned}\dot{V} \le -\alpha V + \frac{1}{2\lambda_{\min}(\mathbf{Q})}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1\\ \boldsymbol{\varepsilon}_2\end{bmatrix}\right\|^2 &+ \frac{1}{2}[\sigma_{\mathbf{D}}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}})\\ &+ \sigma_{\mathbf{C}}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) + \sigma_{\mathbf{g}}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) + \sigma_{\mathbf{h}}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}})]\end{aligned} \tag{30}$$

Hence, $\dot{V} < 0$ whenever

$$\begin{aligned}(\mathbf{s}, \mathbf{e}_\tau, \tilde{\mathbf{W}}_{\mathbf{D}}, \tilde{\mathbf{W}}_{\mathbf{C}}, \tilde{\mathbf{W}}_{\mathbf{g}}, \tilde{\mathbf{W}}_{\mathbf{h}}) \in \{(\mathbf{s}, \mathbf{e}_\tau, \tilde{\mathbf{W}}_{\mathbf{D}}, \tilde{\mathbf{W}}_{\mathbf{C}}, \tilde{\mathbf{W}}_{\mathbf{g}}, \tilde{\mathbf{W}}_{\mathbf{h}}) \,|\, V >\\ \frac{1}{2\alpha}\left[\frac{1}{\lambda_{\min}(\mathbf{Q})}\sup_{\tau \ge t_0}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1(\tau)\\ \boldsymbol{\varepsilon}_2(\tau)\end{bmatrix}\right\|^2 + \sigma_{\mathbf{D}}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}})\right.\\ \left. + \sigma_{\mathbf{C}}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) + \sigma_{\mathbf{g}}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) + \sigma_{\mathbf{h}}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}})\right]\}\end{aligned}$$

This further concludes that **s**, $\mathbf{e}_\tau$, $\mathbf{e}_i$, $\tilde{\mathbf{W}}_{\mathbf{D}}$, $\tilde{\mathbf{W}}_{\mathbf{C}}$, $\tilde{\mathbf{W}}_{\mathbf{g}}$, and $\tilde{\mathbf{W}}_{\mathbf{h}}$ are uniformly ultimately bounded(*u.u.b.*). The implementation of the desired transmission torque (14), control input (16) and update law (25) does not need to calculate the regressor matrix which is required in most adaptive designs for robot manipulators. The convergence of the parameters, however, can be proved to depend on the persistent excitation condition of the input.

The above derivation only demonstrates the boundedness of the closed loop system, but in practical applications the transient performance is also of great importance. For further development, we may apply the comparison lemma[32] to (30) to have the upper bound for V as

$$V(t) \le e^{-\alpha(t-t_0)}V(t_0) + \frac{1}{2\alpha}\left[\frac{1}{\lambda_{\min}(\mathbf{Q})}\sup_{t_0<\tau<t}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1(\tau)\\ \boldsymbol{\varepsilon}_2(\tau)\end{bmatrix}\right\|^2 + \sigma_{\mathbf{D}}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}}) \right. \\ \left. + \sigma_{\mathbf{C}}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) + \sigma_{\mathbf{g}}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) + \sigma_{\mathbf{h}}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}})\right] \tag{31}$$

From (23), we obtain

$$V \ge \frac{1}{2}\left[\lambda_{\min}(\mathbf{A})\left\|\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix}\right\|^2 + \lambda_{\min}(\mathbf{Q}_{\mathbf{D}})Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}}) + \lambda_{\min}(\mathbf{Q}_{\mathbf{C}})Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}}) \right. \\ \left. + \lambda_{\min}(\mathbf{Q}_{\mathbf{g}})Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}}) + \lambda_{\min}(\mathbf{Q}_{\mathbf{h}})Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}})\right] \tag{32}$$

Thus, the bound of $\left\|\begin{bmatrix}\mathbf{s}^T & \mathbf{e}_\tau^T\end{bmatrix}^T\right\|^2$ for $t \ge t_0$ can be derived from (31) and (32) as

$$\begin{aligned}\left\|\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix}\right\|^2 &\le \frac{1}{\underline{\eta}_A}[V - \lambda_{\min}(\mathbf{Q}_{\mathbf{D}})Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}}) - \lambda_{\min}(\mathbf{Q}_{\mathbf{C}})Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}}) \\ &\quad - \lambda_{\min}(\mathbf{Q}_{\mathbf{g}})Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}}) - \lambda_{\min}(\mathbf{Q}_{\mathbf{h}})Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}}) \\ &\le \frac{1}{\underline{\eta}_A}\left\{2e^{-\alpha(t-t_0)}V(t_0) + \frac{1}{\alpha}\left[\frac{1}{\lambda_{\min}(\mathbf{Q})}\sup_{t_0<\tau<t}\left\|\begin{bmatrix}\boldsymbol{\varepsilon}_1(\tau)\\ \boldsymbol{\varepsilon}_2(\tau)\end{bmatrix}\right\|^2 + \sigma_{\mathbf{D}}Tr(\mathbf{W}_{\mathbf{D}}^T\mathbf{W}_{\mathbf{D}})\right.\right. \\ &\quad \left. + \sigma_{\mathbf{C}}Tr(\mathbf{W}_{\mathbf{C}}^T\mathbf{W}_{\mathbf{C}}) + \sigma_{\mathbf{g}}Tr(\mathbf{W}_{\mathbf{g}}^T\mathbf{W}_{\mathbf{g}}) + \sigma_{\mathbf{h}}Tr(\mathbf{W}_{\mathbf{h}}^T\mathbf{W}_{\mathbf{h}})\right] \\ &\quad - \lambda_{\min}(\mathbf{Q}_{\mathbf{D}})Tr(\tilde{\mathbf{W}}_{\mathbf{D}}^T\tilde{\mathbf{W}}_{\mathbf{D}}) - \lambda_{\min}(\mathbf{Q}_{\mathbf{C}})Tr(\tilde{\mathbf{W}}_{\mathbf{C}}^T\tilde{\mathbf{W}}_{\mathbf{C}}) \\ &\quad \left. - \lambda_{\min}(\mathbf{Q}_{\mathbf{g}})Tr(\tilde{\mathbf{W}}_{\mathbf{g}}^T\tilde{\mathbf{W}}_{\mathbf{g}}) - \lambda_{\min}(\mathbf{Q}_{\mathbf{h}})Tr(\tilde{\mathbf{W}}_{\mathbf{h}}^T\tilde{\mathbf{W}}_{\mathbf{h}})\right\}\end{aligned} \tag{33}$$

From the derivations above, we can conclude that the proposed design is able to give bounded tracking with guaranteed transient performance. The following theorem is a summary of the above results.

Theorem 1: Consider the n-rigid link flexible-joint robot (1) and (2) with unknown parameters **D**, **C**, and **g** then desired transmission torque (14), control input (16) and update law (25) ensure that

(i) error signals **s**, $\mathbf{e}_\tau$, $\widetilde{\mathbf{W}}_{\mathbf{D}}$, $\widetilde{\mathbf{W}}_{\mathbf{C}}$, $\widetilde{\mathbf{W}}_{\mathbf{g}}$, and $\widetilde{\mathbf{W}}_{\mathbf{h}}$ are *u.u.b.*

(ii) the bound of the tracking error vectors for $t \geq t_0$ can be derived as the form of (33), if the Lyapunov-like function candidates are chosen as (23).

Remark 1: The term with $\sigma_{(\cdot)}$ in (25) is to modify the update law to robust the closed-loop system for the effect of the approximation error[17]. Suppose a sufficient number of basis functions $\beta_{(\cdot)}$ is selected so that the approximation error can be neglected then we may have $\sigma_{(\cdot)} = 0$, and (26) becomes

$$\dot{V} = -[\mathbf{s}^T \quad \mathbf{e}_\tau^T]\mathbf{Q}\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix} \leq 0 \tag{34}$$

It is easy to prove that **s** and $\mathbf{e}_\tau$ are also square integrable. From (21) and (22), $\dot{\mathbf{s}}$ and $\dot{\mathbf{e}}_\tau$ are bounded; as a result, asymptotic convergence of **s** and $\mathbf{e}_\tau$ can easily be shown by Barbalat's lemma. This further implies that $\boldsymbol{\tau} \to \boldsymbol{\tau}_d$ and $\mathbf{q} \to \mathbf{q}_d$ even though **D**, **C**, and **g** are all unknown.

Remark 2: Suppose ε_1 and ε_2 cannot be ignored but their variation bounds are available[16,17] i.e. there exists positive constants δ_1 and δ_2 such that $\|\boldsymbol{\varepsilon}_1\| \leq \delta_1$, and $\|\boldsymbol{\varepsilon}_2\| \leq \delta_2$. To cover the effect of these bounded approximation errors, the desired transmission torque (14) and the control input (16) are modified to be

$$\boldsymbol{\tau}_d = \hat{\mathbf{D}}\dot{\mathbf{v}} + \hat{\mathbf{C}}\mathbf{v} + \hat{\mathbf{g}} - \mathbf{K}_d\mathbf{s} + \boldsymbol{\tau}_{robust1} \tag{35}$$

$$\mathbf{u} = \boldsymbol{\Theta}\mathbf{x}_p + \boldsymbol{\Phi}\boldsymbol{\tau}_{td} + \hat{\mathbf{h}} + \boldsymbol{\tau}_{robust2} \tag{36}$$

where $\boldsymbol{\tau}_{robust1}$ and $\boldsymbol{\tau}_{robust2}$ are robust terms to be designed. Let us consider the Lyapunov-like function candidate (23) and the update law (25) again. The time derivative of V can be computed as

$$\dot{V} = -\left[\mathbf{s}^T \quad \mathbf{e}_\tau^T\right]\mathbf{Q}\begin{bmatrix}\mathbf{s}\\ \mathbf{e}_\tau\end{bmatrix} + \delta_1\|\mathbf{s}\| + \delta_2\|\mathbf{e}_\tau\| + \mathbf{s}^T\boldsymbol{\tau}_{robust1} + \mathbf{e}_\tau^T\boldsymbol{\tau}_{robust2} \tag{37}$$

By picking $\boldsymbol{\tau}_{robust1} = -\delta_1[\mathrm{sgn}(s_1) \quad \cdots \quad \mathrm{sgn}(s_n)]^T$, where s_k, k=1,…,n is the k-th element of **s**, and $\boldsymbol{\tau}_{robust2} = -\delta_2[\mathrm{sgn}(e_{\tau_1}) \quad \cdots \quad \mathrm{sgn}(e_{\tau_n})]^T$ where e_{τ_k}, k=1,…,2n is the k-th element of $\mathbf{e}_\tau$, we may have $\dot{V} \leq 0$, and asymptotic convergence of the state error can be concluded by Barbalat's lemma.

3. SIMULATION STUDY

Consider a planar robot (Fig.1) with two rigid links and two flexible joints represented by the differential equation (1), and (2). The quantities m_i, l_i, l_{ci} and I_i are mass, length, gravity center distance and inertia of link i, respectively. Actual values of link parameters in the simulation are[18] m_1=0.5kg, m_2=0.5kg, l_1=l_2=0.75m, l_{c1}=l_{c2}=0.375m, I_1=0.09375kg-m^2, and I_2=0.046975kg-m^2. The actuator inertias, damping, and joint stiffness are $\mathbf{J} = diag(0.02, 0.01)(kg \cdot m^2)$, $\mathbf{B} = diag(5,4)(Nm \cdot \sec/rad)$ and $\mathbf{K} = diag(100,100)(Nm/rad)$ respectively. We would like the end-point to track a 0.2m-radius circle centered at (0.8 m, 1.0 m) in 10 seconds. To have more challenge, we pick the initial condition of the link angles and the motor angles as $\mathbf{q} = [-0.184 \quad 1.94 \quad 0 \quad 0]^T$ and $\boldsymbol{\theta} = [-0.184 \quad 1.94 \quad 0 \quad 0]^T$ that are significantly away from the desired trajectory. The initial value of the reference model state vector is $\boldsymbol{\tau}_r = [0.39 \quad -0.72 \quad 0 \quad 0]^T$ which is the same as the initial value of the desired reference input $\boldsymbol{\tau}_d$. The controller gains are selected as $\mathbf{K}_d = diag(0.1, 0.1)$ and $\boldsymbol{\Lambda} = diag(5,5)$. Each element of **D**, **C**, **g** and **h** is approximated by the first 41 terms of the Fourier series. The simulation results are shown in Fig. 2 to 8. Fig. 2 shows the tracking performance of the end-point and the desired trajectory in the Cartesian space. It is observed that the end-point trajectory converges nicely to the desired trajectory, although the initial position error is quite large. Fig. 3 is the joint space tracking performance. It shows that the transient response vanishes very quickly. Fig. 4 is the actuator inputs in N-m. Fig. 5 to 8 are the performance of function approximation for **D**, **C**, **g** and **h** respectively. Since the reference input does not satisfy the persistent excitation condition, some estimates do not converge to their actual values but remain bounded as desired. It is worth to note that in designing the controller we do not need much knowledge for the system. All we have to do is to pick some controller parameters and some initial weighting matrices.

4. CONCULSIONS

In this paper, we have proposed a FAT-based adaptive controller for a flexible joint robot containing time-varying uncertainties. The new design is free from regressor calculation and knowledge of bounds of uncertainties.

Feedback of the joint acceleration is also avoided. The function approximation technique is used to deal with time-varying uncertainties. Using the Lyapunov like analysis, rigorous proof of the closed loop stability has been investigated with consideration of the approximation error. Computer simulation results justify its feasibility of giving satisfactory tracking performance on a 2-D flexible-joint robot although we do not know much knowledge about the system model.

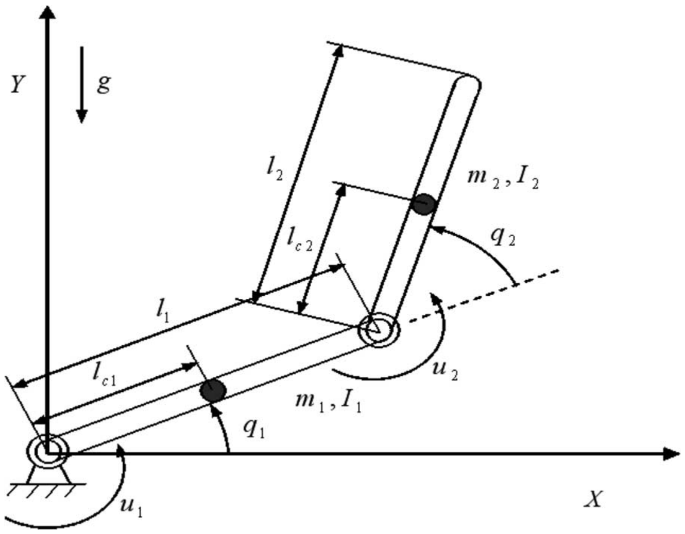


Fig. 1. 2-DOF planar robot

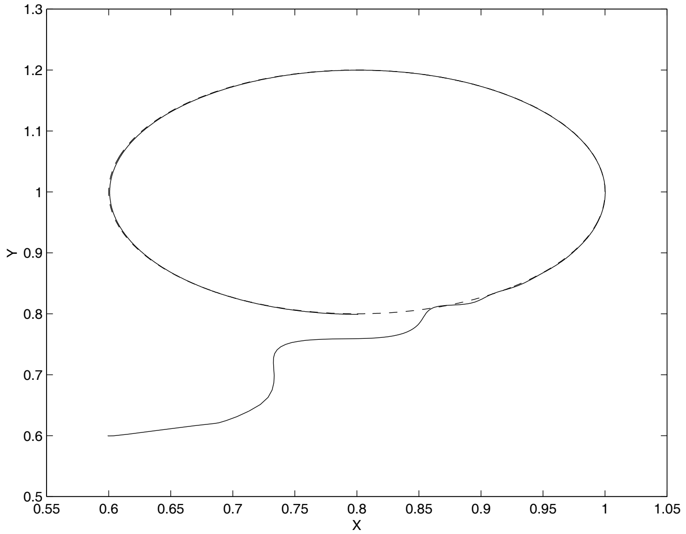


Fig. 2. Tracking performance of end-point in the Cartesian space (— actual; --- desired). Initial position of end-point is at the point (0.6*m*, 0.6*m*). After some transient, the tracking error is very small, although we do not know precise dynamics of the robot.

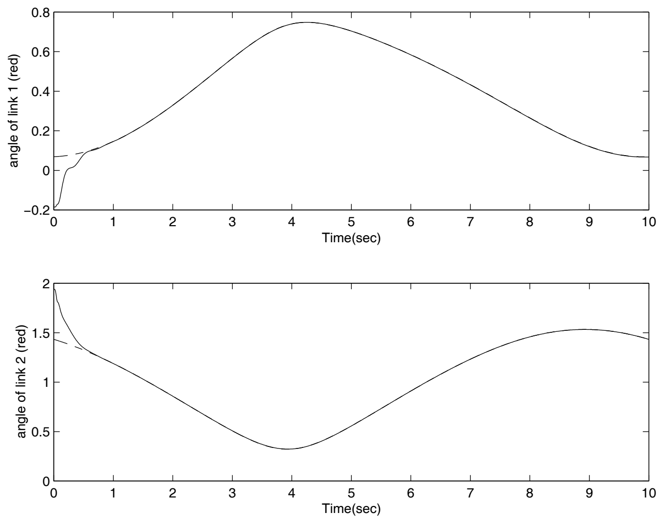


Fig. 3. The joint space tracking performance(— actual; --- desired). The real trajectory converges very quickly.

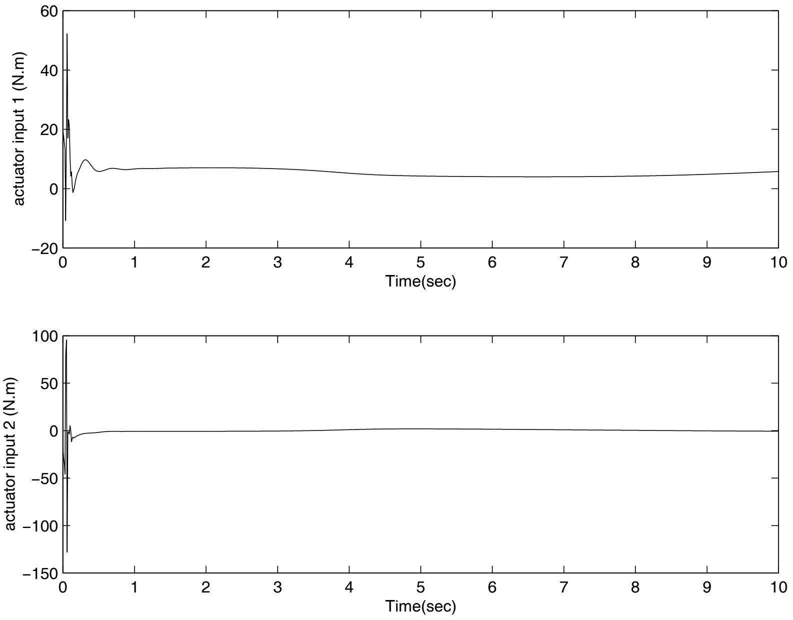


Fig. 4. Actuator input.

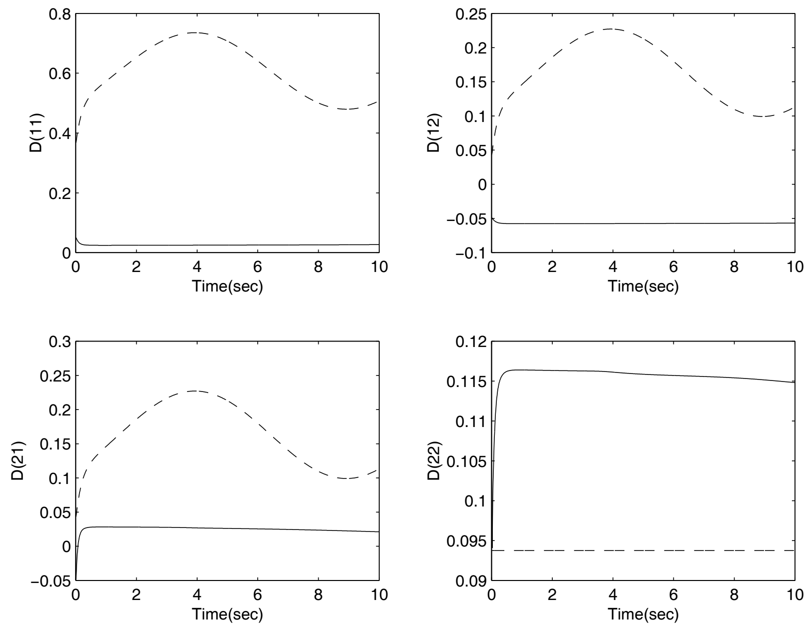


Fig. 5. Approximation of D matrix(— estimate; --- real). Although the estimated values do not converge to the true values, they are bounded and small.

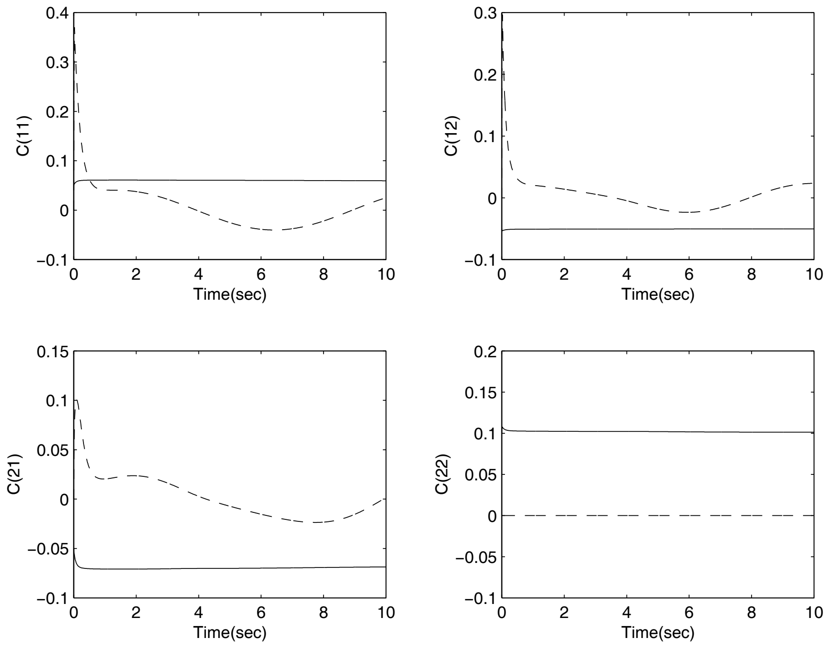


Fig. 6. Approximation of **C** matrix(— estimate; --- real).

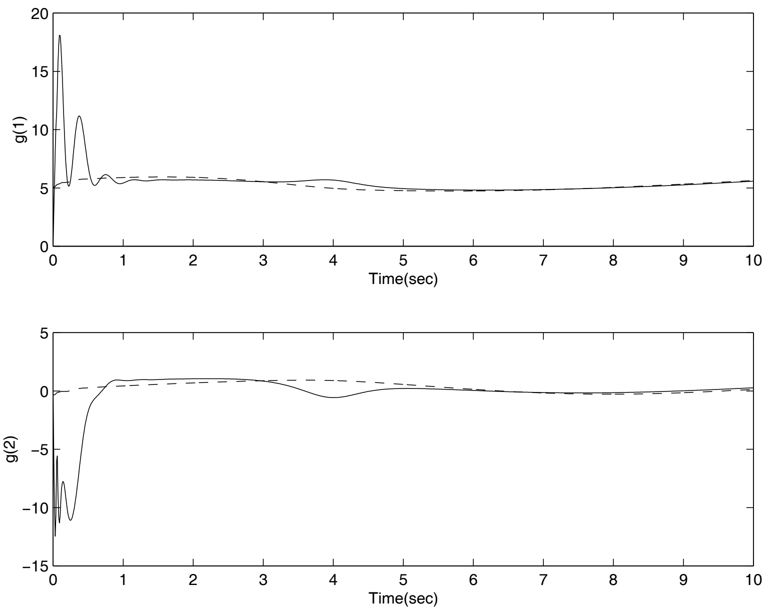


Fig. 7. Approximation of vector **g**(— estimate; --- real).

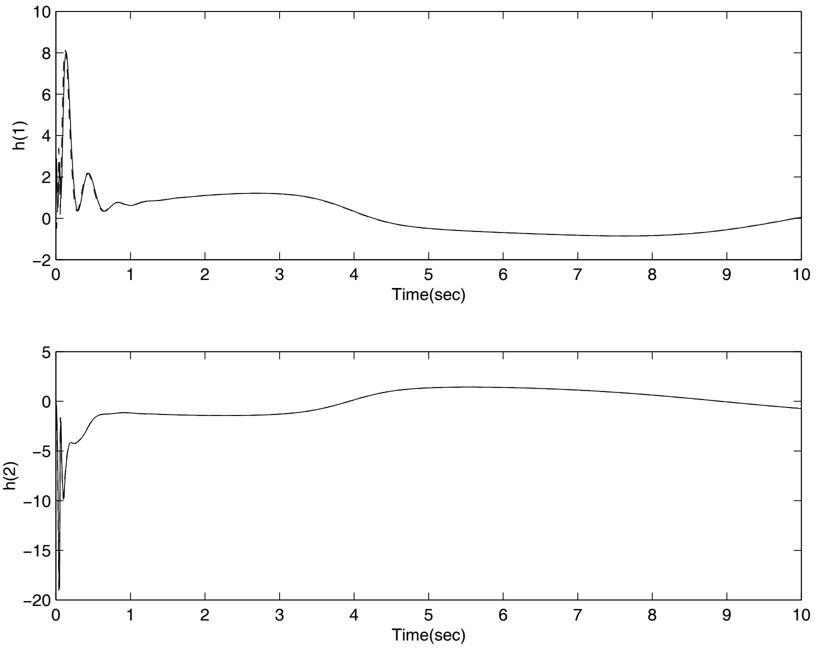


Fig. 8. Approximation of vector **h**(— estimate; --- real).

ACKNOWLEDGMENT

This work was supported in part by the Ministry of Economic Affairs of Taiwan, R.O.C., under Grant 8353CA1300.

5. REFERENCES

1. L. M. Sweet and M. C. Good, Redefinition of the robot motion control problem: effects of plant dynamics, drive system constraints, and user requirements, in *Proc. IEEI Int. Conf. of Decision and Control*, Las Vegas, pp. 724-730 (1984).
2. M. W. Spong, Adaptive control of flexible-joint manipulators, *Systems and Control Letters*, vol. 13, pp.15-21 (1989).
3. M. W. Spong, Adaptive control of flexible-joint manipulators: Comments on two papers, *Automatica*, vol.31, no. 4, pp.585-590 (1995).
4. K. P. Chen and L. C. Fu, Nonlinear adaptive motion control for a manipulator with flexible-joints, In *Proc. IEEE Intl. Conf. on Robotics and Automation*, Phoenix, Az., pp. 1201-1206 (1989).
5. K. Khorasani, Adaptive control of flexible-joint robots, *IEEE Trans. on robotics and Automation*, Vol. 8, No. 2, pp. 250-267, April (1992).
6. S. Y. Lim, D. M. Dawson, J. Hu, and M. S. de Queiroz, An adaptive link position tracking controller for rigid-link flexible-joint robots without velocity measurements, *IEEE Trans. on System, man, and Cybernetics-Part B: Cybernetics*, vol. 27, No. 3 (1997).
7. W. E. Dixon, E. Zergeroglu, D. M. Dawson and M. W. Hannan, Global adaptive partial state feedback tracking control of rigid-link flexible-joint robots, in *Proc. IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics*, Atlanta, GA, pp. 281-286 (1999).
8. W. Yim, Adaptive control of a flexible-joint manipulator, In *Proc. IEEE Int. Conf. on Robotics and Automation*, pp.3441-3446 (2001).
9. K. Kozlowski and P. Sauer, On adaptive control of flexible-joint manipulators: theory and experiments, in *Proc. IEEE Intl. Sym. on Industrial Electronics*, Bled, Slovenia, vol. 3, pp. 1153 -1158 (1999).
10. K. Kozlowski and P. Sauer, The stability of the adaptive tracking controller of rigid and flexible-joint robots, in *Proc. on the First Workshop on Robot Motion and Control*, Kiekrz, Poland, pp. 85-93 (1999).
11. L. Tian and A. A. Goldenberg, Robust adaptive control of flexible joint robots with joint torque feedback, In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pp.1229-1234 (1995).
12. S. Jain and F. Khorrami, Robust adaptive control of flexible-joint manipulators, *Automatica*, Vol. 34, No. 5, pp.609-615 (1998).
13. R. Ortega and M. W. Spong, Adaptive motion control of rigid robots: a tutorial, *Automatica*, Vol. 25, pp. 509-519 (1989).
14. A. C. Huang and Y. C. Chen, Adaptive sliding control for single-link flexible-joint robot with mismatched uncertainties, *IEEE Trans. on Control Systems Technology*, Vol. 12, Issue 5, pp. 770-775 (2004).
15. S. S. Ge, C. C. Hang, T. H. Lee and T. Zang, *Stable Adaptive Neural Network Control*, (Boston: Kluwer Academic, 2001).

16. A. C. Huang and Y. S. Kuo, Sliding control of nonlinear systems containing time-varying uncertainties with unknown bounds, *Int. J. of Control*, Vol.74, No.3, pp.252-264 (2001).
17. J. T. Spooner, M. Maggiore, R. Ordonez and K. M. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems – Neural and Fuzzy Approximator Techniques*, NY: John Wiley & Sons (2002).
18. M. C. Chien and A. C. Huang, Adaptive impedance control of robot manipulators based on function approximation technique, *Robotica*, Vol. 22, pp. 395-403 (2004).
19. A. C. Huang and Y. C. Chen, Adaptive multiple-surface sliding control for non-autonomous systems with mismatched uncertainties, *Automatica*, Vol. 11, No. 40, pp. 1939-1945 (2004).
20. P. C. Chen and A. C. Huang, Adaptive multiple-surface sliding control of hydraulic active suspension systems based on the function approximation technique, *J. of Vibration and Control*, Vol. 11, No. 5, pp. 685-706 (2005).
21. P. C. Chen and A. C. Huang, Adaptive sliding control of non-autonomous active suspension systems with time-vary loadings, *J. of Sound and Vibration*, Vol. 282, No.3-5, pp.1119-1135, (2005).
22. A. C. Huang, S. C. Wu, and W. F. Ting, "An FAT-based Adaptive Controller for Robot Manipulators without Regressor Matrix: Theory and Experiments," *Robotica*, vol. 24, pp. 205-210, (2006).
23. A. C. Huang and K. K. Liao, "FAT-based Adaptive Sliding Control for Flexible Arms, Theory and Experiments," *Journal of Sound and Vibration*, vol. 298, issue 1-2, pp. 194-205, (2006).
24. M. C. Chien and A. C. Huang, "Adaptive control of flexible-joint electrically-driven robot with time-varying uncertainties," *IEEE Trans. Industrial Electronics*, vol.54, no.2, pp.1032-1038, (2007).
25. M. C. Chien and A. C. Huang, "Adaptive Control of Electrically-driven Robot without Computation of Regressor Matrix," *J. Chinese Institute of Engineers*, vol.30, no.5, pp.855-862, (2007).
26. M. C. Chien and A. C. Huang, "An Adaptive Controller Design for Flexible-Joint Electrically-Driven Robots with Consideration of Time-Varying Uncertainties," *Chapter 5 in the book Frontiers in Adaptive Control*, I-Tech Education and Publishing, Vienna, Austria, (2009).
27. M. C. Chien and A. C. Huang, "FAT-based Adaptive Visual Servoing for Robots with Time Varying Uncertainties," in *Proc. IEEE Int. Conf. Robot. Automa.*, pp.3700-3705, (2009).
28. A. C. Huang and M. C. Chien, "Design of a Regressor-free Adaptive Impedance Controller for Flexible-joint Electrically-driven Robots," in *Proc. IEEE Int. Conf. Industrial Electronics and Applications* , pp.17-22, (2009).
29. M. W. Spong, Modeling and control of elastic joint robots, *Tran. of the ASME*, vol.109, pp.310-319 (1987).
30. K. S. Narendra and A. M. Annaswamy, *Stable Adaptive System*, Prentice Hall, (1989).
31. J-J. E. Slotine and W. Li, *Applied Nonlinear Control*, Prentice Hall, NJ, (1991).
32. Khalil, H.K., *Nonlinear Systems*, Prentice-Hall, New Jersey, (2002).

APPENDIX

Lemma A.1:

Let $\mathbf{s} \in \Re^n$, $\boldsymbol{\varepsilon} \in \Re^n$ and $\mathbf{K}$ is the $n \times n$ positive definite matrix. Then,

$$-\mathbf{s}^T\mathbf{K}\mathbf{s} + \mathbf{s}^T\boldsymbol{\varepsilon} \le \frac{1}{2}[\lambda_{\min}(\mathbf{K})\|\mathbf{s}\|^2 - \frac{\|\boldsymbol{\varepsilon}\|^2}{\lambda_{\min}(\mathbf{K})}]. \tag{A.1}$$

Proof:

$$\begin{aligned}
-\mathbf{s}^T\mathbf{K}\mathbf{s} + \mathbf{s}^T\boldsymbol{\varepsilon} &\le [-\lambda_{\min}(\mathbf{K})\|\mathbf{s}\| + \|\boldsymbol{\varepsilon}\|]\|\mathbf{s}\| \\
&= -\frac{1}{2}[\sqrt{\lambda_{\min}(\mathbf{K})}\|\mathbf{s}\| - \frac{\|\boldsymbol{\varepsilon}\|}{\sqrt{\lambda_{\min}(\mathbf{K})}}]^2 \\
&\quad -\frac{1}{2}[\lambda_{\min}(\mathbf{K})\|\mathbf{s}\|^2 - \frac{\|\boldsymbol{\varepsilon}\|^2}{\lambda_{\min}(\mathbf{K})}] \\
&\le -\frac{1}{2}[\lambda_{\min}(\mathbf{K})\|\mathbf{s}\|^2 - \frac{\|\boldsymbol{\varepsilon}\|^2}{\lambda_{\min}(\mathbf{K})}]
\end{aligned}$$

Q.E.D.

Lemma A.2:

Let $\mathbf{w}_i^T = [w_{i1} \quad w_{i2} \quad \cdots \quad w_{in}] \in \Re^{1\times n}$, *i*=1,…,*m* and $\mathbf{W}$ is a block diagonal matrix defined as $\mathbf{W} = diag\{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_m\} \in \Re^{mn\times m}$. Then,

$$Tr(\mathbf{W}^T\mathbf{W}) = \sum_{i=1}^{m}\|\mathbf{w}_i\|^2 \tag{A.2}$$

The notation *Tr*(.) denotes the trace operation.

Proof: The proof is straightforward as below:

$$\mathbf{W}^T\mathbf{W} = \begin{bmatrix} w_{11} & \cdots & w_{1n} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & w_{21} & \cdots & w_{2n} & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & w_{m1} & \cdots & w_{mn} \end{bmatrix} \begin{bmatrix} w_{11} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ w_{1n} & 0 & \cdots & 0 \\ 0 & w_{21} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & w_{2n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{m1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{mn} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{w}_1^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{w}_2^T & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{w}_m^T \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{w}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{w}_m \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{w}_1^T\mathbf{w}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{w}_2^T\mathbf{w}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{w}_m^T\mathbf{w}_m \end{bmatrix}$$

$$= \begin{bmatrix} \|\mathbf{w}_1\|^2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \|\mathbf{w}_2\|^2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \|\mathbf{w}_m\|^2 \end{bmatrix}$$

The last equality holds because by definition $\mathbf{w}_i^T\mathbf{w}_i = w_{i1}^2 + w_{i2}^2 + \ldots + w_{im}^2 = \|\mathbf{w}_i\|^2$.

Therefore, we have $$Tr = (\mathbf{W}^T\mathbf{W}) = \sum_{i=1}^{m} \|\mathbf{w}_i\|^2 .$$

Q.E.D.

Lemma A.3:

Suppose $\mathbf{w}_i^T = [w_{i1} \quad w_{i2} \quad \cdots \quad w_{in}] \in \Re^{1\times n}$ and $\mathbf{v}_i^T = [v_{i1} \quad v_{i2} \quad \cdots \quad v_{in}] \in \Re^{1\times n}$, *i*=1,…,*m*. Let **W** and **V** be block diagonal matrices that are defined as $\mathbf{W} = diag\{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_m\} \in \Re^{mn\times m}$ and $\mathbf{V} = diag\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m\} \in \Re^{mn\times m}$, respectively. Then,

$$Tr(\mathbf{V}^T\mathbf{W}) \le \sum_{i=1}^{m} \|\mathbf{v}_i\|\|\mathbf{w}_i\| \tag{A.3}$$

Proof: The proof is also straightforward:

$$\mathbf{V}^T\mathbf{W} = \begin{bmatrix} \mathbf{v}_1^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{v}_2^T & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{v}_m^T \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{w}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{w}_m \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{v}_1^T\mathbf{w}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{v}_2^T\mathbf{w}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{v}_m^T\mathbf{w}_m \end{bmatrix}$$

Hence,

$$\begin{aligned} Tr(\mathbf{V}^T\mathbf{W}) &= \mathbf{v}_1^T\mathbf{w}_1 + \mathbf{v}_2^T\mathbf{w}_2 + \ldots + \mathbf{v}_m^T\mathbf{w}_m \\ &\le \|\mathbf{v}_1\|\|\mathbf{w}_1\| + \|\mathbf{v}_2\|\|\mathbf{w}_2\| + \ldots + \|\mathbf{v}_m\|\|\mathbf{w}_m\| \\ &= \sum_{i=1}^{m} \|\mathbf{v}_i\|\|\mathbf{w}_i\| \end{aligned}$$

Q.E.D.

Lemma A.4:

Let $\mathbf{W}$ be defined as in *Lemma A.2*, and $\widetilde{\mathbf{W}}$ is a matrix defined as $\widetilde{\mathbf{W}} = \mathbf{W} - \hat{\mathbf{W}}$, where $\hat{\mathbf{W}}$ is a matrix with proper dimension. Then

$$Tr(\widetilde{\mathbf{W}}^T\hat{\mathbf{W}}) \le \frac{1}{2}Tr(\mathbf{W}^T\mathbf{W}) - \frac{1}{2}Tr(\widetilde{\mathbf{W}}^T\widetilde{\mathbf{W}}). \tag{A.4}$$

Proof:

$$
\begin{aligned}
Tr(\tilde{\mathbf{W}}^T\hat{\mathbf{W}}) &= Tr(\tilde{\mathbf{W}}^T\mathbf{W}) - Tr(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}) \\
&\leq \sum_{i=1}^{m}(\|\tilde{\mathbf{w}}_i\|\|\mathbf{w}_i\| - \|\tilde{\mathbf{w}}_i\|^2) \qquad \text{(by } \mathit{Lemma\ A}.2 \text{ and } \mathit{A}.3\text{)} \\
&= \frac{1}{2}\sum_{i=1}^{m}[\|\mathbf{w}_i\|^2 - \|\tilde{\mathbf{w}}_i\|^2 - (\|\tilde{\mathbf{w}}_i\| - \|\mathbf{w}_i\|)^2] \\
&\leq \frac{1}{2}\sum_{i=1}^{m}(\|\mathbf{w}_i\|^2 - \|\tilde{\mathbf{w}}_i\|^2) \\
&= \frac{1}{2}Tr(\mathbf{W}^T\mathbf{W}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}) \quad \text{(by } \mathit{Lemma\ A}.2\text{)}
\end{aligned}
$$

Q.E.D.

In the above lemmas, we consider properties of a block diagonal matrix. In the following, we would like to extend the analysis to a class of more general matrices.

Lemma A.5:

Let $\mathbf{W}$ be a matrix in the form $\mathbf{W}^T = [\mathbf{W}_1^T \quad \mathbf{W}_2^T \quad \cdots \quad \mathbf{W}_p^T] \in \Re^{pmn\times m}$ where $\mathbf{W}_i = diag\{\mathbf{w}_{i1}, \mathbf{w}_{i2}, \cdots, \mathbf{w}_{im}\} \in \Re^{mn\times m}$, *i*=1,…,*p*, are block diagonal matrices with the entries of vectors $\mathbf{w}_{ij}^T = [w_{ij1} \quad w_{ij2} \quad \cdots \quad w_{ijn}] \in \Re^{1\times n}$, *j*=1,…,*m*. Then, we may have

$$
Tr(\mathbf{W}^T\mathbf{W}) = \sum_{i=1}^{p}\sum_{j=1}^{m}\|\mathbf{w}_{ij}\|^2 . \tag{A.5}
$$

Proof:

$$
\begin{aligned}
\mathbf{W}^T\mathbf{W} &= [\mathbf{W}_1^T \quad \cdots \quad \mathbf{W}_p^T]\begin{bmatrix}\mathbf{W}_1 \\ \vdots \\ \mathbf{W}_p\end{bmatrix} \\
&= \mathbf{W}_1^T\mathbf{W}_1 + \cdots + \mathbf{W}_p^T\mathbf{W}_p
\end{aligned}
$$

Hence, we may calculate the trace as

$$
\begin{aligned}
Tr(\mathbf{W}^T\mathbf{W}) &= Tr(\mathbf{W}_1^T\mathbf{W}_1) + \cdots + Tr(\mathbf{W}_p^T\mathbf{W}_p) \\
&= \sum_{j=1}^{m}\|\mathbf{w}_{1j}\|^2 + \cdots + \sum_{j=1}^{m}\|\mathbf{w}_{pj}\|^2 \qquad \text{(by } \mathit{Lemma\ A}.1\text{)} \\
&= \sum_{i=1}^{p}\sum_{j=1}^{m}\|\mathbf{w}_{ij}\|^2
\end{aligned}
$$

Q.E.D.

Lemma A.6:

Let **V** and **W** be matrices defined in *Lemma A.5*, Then,

$$Tr(\mathbf{V}^T\mathbf{W}) \le \sum_{i=1}^{p}\sum_{j=1}^{m}\left\|\mathbf{v}_{ij}\right\|\left\|\mathbf{w}_{ij}\right\|. \tag{A.6}$$

Proof:

$$\begin{aligned}
Tr(\mathbf{V}^T\mathbf{W}) &= Tr(\mathbf{V}_1^T\mathbf{W}_1) + \cdots + Tr(\mathbf{V}_p^T\mathbf{W}_p) \\
&\le \sum_{j=1}^{m}\left\|\mathbf{v}_{1j}\right\|\left\|\mathbf{w}_{1j}\right\| + \cdots + \sum_{j=1}^{m}\left\|\mathbf{v}_{pj}\right\|\left\|\mathbf{w}_{pj}\right\| \qquad \text{(by } \textit{Lemma A.}3\text{)} \\
&= \sum_{i=1}^{p}\sum_{j=1}^{m}\left\|\mathbf{v}_{ij}\right\|\left\|\mathbf{w}_{ij}\right\|
\end{aligned}$$

Q.E.D.

Lemma A.7:

Let **W** be defined as in *Lemma A.5*, and $\tilde{\mathbf{W}}$ is a matrix defined as $\tilde{\mathbf{W}} = \mathbf{W} - \hat{\mathbf{W}}$, where $\hat{\mathbf{W}}$ is a matrix with proper dimension. Then

$$Tr(\tilde{\mathbf{W}}^T\hat{\mathbf{W}}) \le \frac{1}{2}Tr(\mathbf{W}^T\mathbf{W}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}). \tag{A.7}$$

Proof:

$$\begin{aligned}
Tr(\tilde{\mathbf{W}}^T\hat{\mathbf{W}}) &= Tr(\tilde{\mathbf{W}}^T\mathbf{W}) - Tr(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}) \\
&\le \sum_{i=1}^{p}\sum_{j=1}^{m}(\left\|\tilde{\mathbf{w}}_{ij}\right\|\left\|\mathbf{w}_{ij}\right\| - \left\|\tilde{\mathbf{w}}_{ij}\right\|^2) \qquad \text{(by } \textit{Lemma A.}5 \text{ and } \textit{A.}6\text{)} \\
&= \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{m}[\left\|\mathbf{w}_{ij}\right\|^2 - \left\|\tilde{\mathbf{w}}_{ij}\right\|^2 - (\left\|\tilde{\mathbf{w}}_{ij}\right\| - \left\|\mathbf{w}_{ij}\right\|)^2] \\
&\le \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{m}(\left\|\mathbf{w}_{ij}\right\|^2 - \left\|\tilde{\mathbf{w}}_{ij}\right\|^2) \\
&= \frac{1}{2}Tr(\mathbf{W}^T\mathbf{W}) - \frac{1}{2}Tr(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}) \quad \text{(by } \textit{Lemma A.}5\text{)}
\end{aligned}$$

Q.E.D

A 9-DoF Wheelchair-Mounted Robotic Arm System: Design, Control, Brain-Computer Interfacing, and Testing

Redwan Alqasemi and Rajiv Dubey
University of South Florida
Tampa, Florida
USA

1. Introduction

A wheelchair-mounted robotic arm (WMRA) system was designed and built to meet the needs of mobility-impaired persons with limitations of upper extremities, and to exceed the capabilities of current devices of this type. The control of this 9-DoF system expands on the conventional control methods and combines the 7-DoF robotic arm control with the 2-DoF power wheelchair control. The 3-degrees of redundancy are optimized to effectively perform activities of daily living (ADLs) and combine wheelchair mobility and arm manipulation to overcome singularities, joint limits and some workspace limitations. The control system is designed for teleoperated or autonomous coordinated Cartesian control, and it offers expandability for future research. Several interchangeable user interfaces were implemented in the design, including a Brain Computer Interface (BCI). That BCI system was modified and integrated to the control of the WMRA system for users who are totally paralyzed or "locked-in" and cannot use conventional augmentative technologies, all of which require some measure of muscle control. Testing and data collection were performed on human subjects, and the design, various optimized control methods and test results are presented in this paper.
According to the 2006 US Census Bureau report (US Census Bureau, 2002), about 51.2 million Americans (18.1 percent of the US population) had some level of disability and 32.5 million of them (11.5 percent) had a severe disability. About 10.7 million Americans older than 6 years of age needed personal assistance with one or more activities of daily living (ADL). This work focuses on people who have limited or no upper extremity mobility due to spinal cord injury or dysfunction, or genetic predispositions, or people who are "locked-in" (e.g., by end-stage amyotrophic lateral sclerosis, brainstem stroke, or severe polyneuropathy). Robotic aides used in these applications may vary from advanced limb orthosis to robotic arms (Reswick, 1990).
A wheelchair mounted robotic arm can enhance the manipulation capabilities of individuals with disabilities that are using power wheelchairs, and reduce dependence on human aides.

Unfortunately, most WMRAs have had limited commercial success due to poor usability and low payload. It is often difficult to accomplish many of the Activities of Daily Living (ADL) tasks with the two commercial WMRAs currently on the market due to its physical and control limitations. Furthermore, the lack of the integration of the robotic arm controller with the wheelchair controller leads to an increased mental load on the user. This project attempts to surpass available commercial WMRA devices by offering an intelligent system that combines the mobility of the wheelchair and the manipulation of a newly designed arm in an effort to improve performance, usability, control and reduce mental burden on the user while maintaining cost competitiveness.

It is desired to fulfill the need of such integrated systems to be used for many ADL tasks such as opening a spring-loaded door autonomously and going through it, interactively exchange objects with a companion on the move, and avoiding singularities in a small working environment, such as an office, where wheelchair motion can be slightly utilized to maneuver objects while avoiding singularities (similar to a person sitting on an office chair and handling surrounding objects by moving his/her arm while slightly moving the chair to get closer to an object that is otherwise unreachable). These tasks can be performed without the need to switch between the wheelchair controller and the robotic arm controller. The implementation of the combined control will still give the user the option to control the robotic arm alone or the wheelchair alone.

In addition, a Brain-Computer Interface (BCI), which does not require any muscular activity for device manipulation, is also a feasible control mechanism for the totally paralyzed users. For a “locked-in” user, the BCI control system introduces a wide range of self-performed ADL tasks that are otherwise impossible to perform independently.

2. Background

There are several designs of workstation-based robotic arm systems, but WMRAs combine the idea of workstation and mobile-base robots to mount a manipulator arm onto a power wheelchair. One of the most important design considerations of where to mount a robotic arm in a power wheelchair is the safety of the operator (Yanco, 1998). There have been several attempts in the past to create commercially viable wheelchair mounted robotic arms, including the two currently available WMRAs: Manus and Raptor.

The Manus manipulator, manufactured by Exact Dynamics, available since the early 1990s (Eftring & Boschian, 1999), can be programmed in a manner comparable to industrial robotic manipulators. A picture of Manus mounted onto a wheelchair is shown in Figure 1. It is a 6 DoF arm, with servomotors all housed in a cylindrical base. Besides the fact that it is controlled independent of the wheelchair control, the current controller allows for Cartesian control, but when it comes close to a singularity, it stops and waits for the user to move it in a different direction. This kind of control increases the cognitive load on the user.

Another production WMRA is the Raptor, manufactured by Applied Resources (Mahoney, 2001), as shown in Figure 2, which mounts to the right side of the wheelchair. This manipulator has four degrees of freedom plus a planar gripper. The user directly controls the arm with either a joystick or a keypad controller. Because the Raptor does not have encoders, the manipulator cannot be controlled in Cartesian coordinates. This compromise was done to minimize the cost, but it decreases the usability of the arm.

Various redundancy resolution methods were developed in the past to optimize a solution based on certain criteria function. Weighted least norm solution method was used by Chan et al (Chan & Dubey, 1995) to penalize the motion of some joints over others. This method can be used in the case of WMRAs to make the wheelchair motion as a secondary motion when needed. The combination of mobility and manipulation has been studied by researchers in the form of a mobile platform that carries a robotic arm. Chung, et al (Chung & Velinsky, 1999) resolved the kinematic redundancy by decomposing the mobile manipulator into two different subsystems, the mobile platform and the manipulator. Each one of these subsystems is controlled independently with an interaction algorithm between the two controllers.

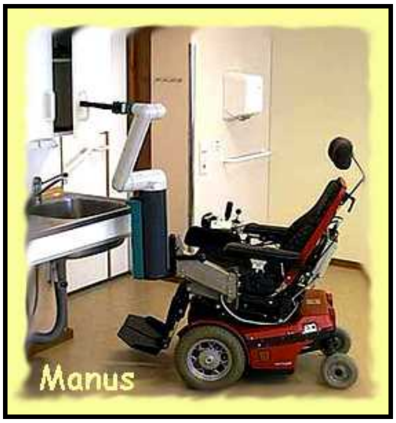


Fig. 1. Manus arm.

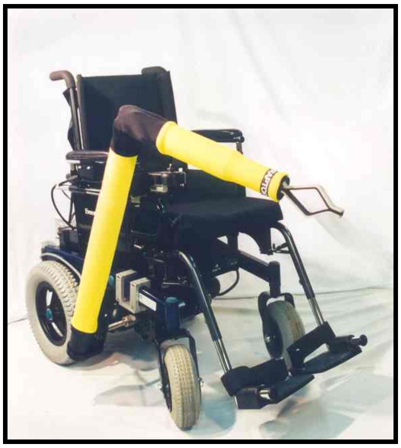


Fig. 2. Raptor arm.

Mirosaw (Galicki, 2005) used external penalty functions to enforce the holonomic manipulability and collision avoidance. His results showed continuous velocities near obstacles. An extension to different redundancy resolution schemes has been proposed by

Luca (Luca et al., 2006) to include the representation of mobile platforms in the Jacobian. His simulation showed consistent results.
BCI systems have been used in the past to control peripheral devices such as TVs, cell phones, and computers among others (Farwell & Donchin, 1988). In the case of BCI-controlled computers, the control can be extended to operate rehabilitation devices, prosthetic limbs, and robotic hands. However, using a brain computer interface to control a robotic system is relatively new and primarily at a research stage. In this work, conventional user interfaces, such as a spaceball, a joystick, a keypad and a touch screen, were used to control the WMRA system. A Brain-Computer Interface (BCI) was modified along this line of work and implemented as one of the modular interchangeable user interfaces for the WMRA to be controlled and used by the locked-in patients who are paralyzed from the neck down. This allows the user to communicate action choices to the robot using the BCI.
Farwell and Donchin (Farwell & Donchin, 1988) developed a BCI system that utilized the P300 component of the Event Related Brain Potential (ERP) to allow a locked-in patient to "type" text into a computer without using any neuromuscular function.
D. Valbuena et. al. and T. Lüth et. al. (Valbuena et al., 2007; Lüth et al., 2007) used a Steady-state Visual Evoked Potentials (SSVEP) system to control a semi-autonomous robot to provide a user 1½ hours of independence. The BCI transforms high level orders from the user into low level commands for the robot. The user selects tasks, such as pouring a liquid into a glass, from a menu to control the robotic arm.
The main objective of this work is to develop and optimize a control system that combines the manipulation of the newly designed 7-DoF robotic arm and the mobility of a modified 2-DoF wheelchair in a single control algorithm. Redundancy resolution is to be optimally solved to avoid singularities and joint limits as well as to allow larger wheelchair or manipulator motion depending on the proximity to the goal. The controller is capable of moving autonomously or using teleoperation. A Brain-Computer Interface (BCI) system is to be modified and adapted to the modular control algorithm to allow locked-in users to communicate with the WMRA system and command it to perform a pre-selected list of commands.

3. Motion Control of the 9-DoF WMRA System

3.1 Wheelchair Motion

The differential drive used in power wheelchairs represents a 2-DoF system that moves in plane (Papadopoulos & Poulakakis, 2000). When wheelchair motion control is not combined with the mobile manipulator control, it is always desired to align the wheelchair in certain direction to position the arm in the desired position and orientation to perform ADLs. In this case, non-holonomic constraints on mobile platforms restrict the system's ability to control all 3-DoFs in the workspace. Trajectory planning is required to compensate for the lost DoF in that plane. Suppose that the wheelchair is commanded to move the arm's base reference frame from "T0" position to "T1" position, where "T" is the homogeneous transformation matrix of the corresponding configuration, the motion can be divided into three sub-motions that can be planned in six steps to realize the X-direction motion, Y-direction motion and the Z-direction orientation. The following six steps can be programmed to execute these three sub-motions: First, from the initial point of the arm base at "T0", find the corresponding wheelchair's frame transformation matrix at that pose.

Second, from the destination point of the arm base at "T1", find the corresponding wheelchair's frame transformation matrix at that pose. Third, draw a virtual line between the two new frame transformations of the wheelchair's frame, and find the angle of that line using the transformation resultant between the two. Fourth, command the wheelchair to rotate to the angle of the new line with no translation. Fifth, command the wheelchair to move in a straight line from the initial position to the final position of the wheelchair, ignoring the orientation. Sixth, command the wheelchair to rotate from the angle of the new line to that of the final position. Figure 3 shows the trajectory planning resulting in three sub-motions when given the initial and final position and orientation:

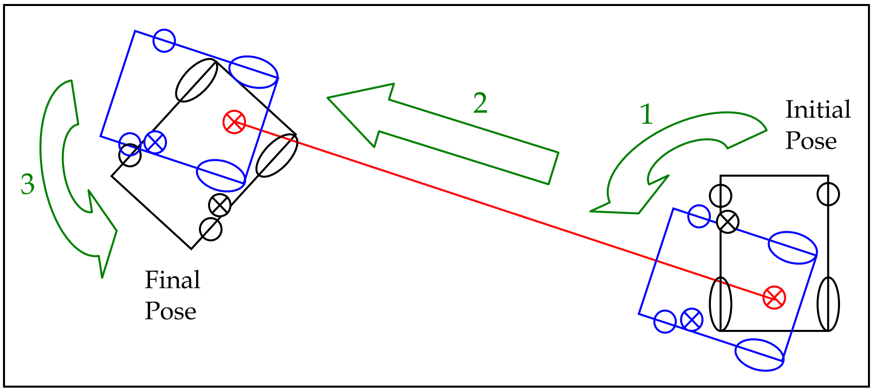


Fig. 3. Trajectory planning for planar motion.

When the wheelchair control is combined with the manipulator control, the above procedure is not necessary since the total system will be redundant. Assuming that the manipulator is mounted on the wheelchair with "L2" and "L3" offset distances from the center of the differential drive across the x and y coordinates respectively (as shown in Figure 4), the mapping of the wheels' velocities to the manipulator's end effector velocities along its coordinates is defined by:

$$\dot{q}_c = J_c \cdot J_W \cdot V_c \tag{1}$$

where:

$$\dot{q}_c = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\alpha} & \dot{\beta} & \dot{\phi} \end{bmatrix}^T, \quad V_c = \begin{bmatrix} \dot{\theta}_l \\ \dot{\theta}_r \end{bmatrix},$$

$$J_c = \begin{bmatrix} [I]_{2x2} & \begin{bmatrix} -(P_{xg} \cdot S\phi + P_{yg} \cdot C\phi) \\ P_{xg} \cdot C\phi - P_{yg} \cdot S\phi \end{bmatrix} \\ [0]_{2x2} & [0]_{3x1} \\ [0]_{2x2} & 1 \end{bmatrix}_{6x3}, \text{ and}$$

$$J_W = \frac{L_5}{2}\begin{bmatrix} c\phi_c + \frac{2}{L_1}(L_2 s\phi_c + L_3 c\phi_c) & c\phi_c - \frac{2}{L_1}(L_2 s\phi_c + L_3 c\phi_c) \\ s\phi_c - \frac{2}{L_1}(L_2 c\phi_c - L_3 s\phi_c) & s\phi_c + \frac{2}{L_1}(L_2 c\phi_c - L_3 s\phi_c) \\ \frac{-2}{L_1} & \frac{2}{L_1} \end{bmatrix}_{3x2}$$

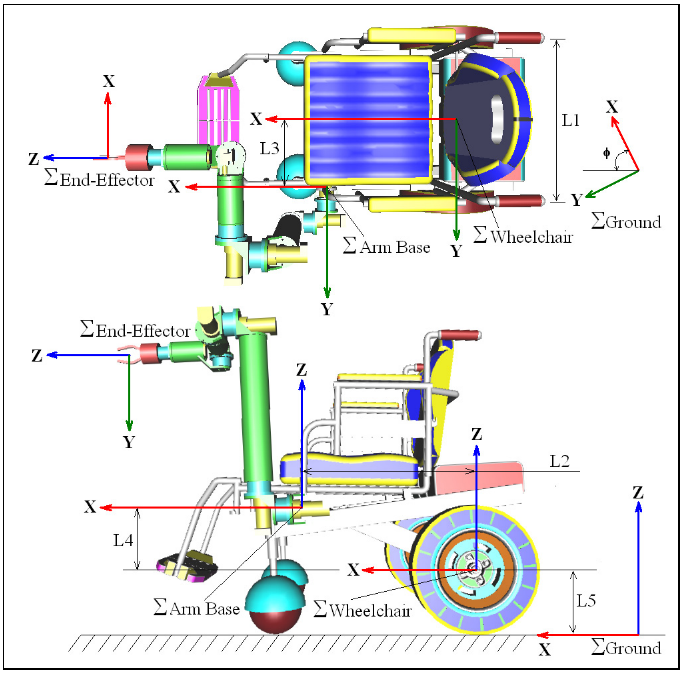


Fig. 4. Coordinate frames and dimensions of interest.

where "P_{xg}" and "P_{yg}" are the x-y coordinates of the end-effector based on the arm base frame, "Φ" is the angle of the arm base frame, which is the same as the angle of the wheelchair based on the ground frame, "L_5" is the wheels' radius, and "L_1" is the distance between the two driving wheels. The above Jacobian can be used to control the wheelchair with the Jacobian of the arm after combining them together.

3.2 The 7-DoF Arm Motion

From the D-H parameters specified in an earlier publication (Alqasemi et al., 2005), the 6x7 Jacobian that relates the joint rates to the Cartesian speeds of the end effector based on the base frame is generated according to Craig's notation (Craig, 2003)

$$\dot{r} = J_A \cdot V_A \tag{2}$$

where:

$\dot{r} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\alpha} & \dot{\beta} & \dot{\gamma} \end{bmatrix}^T$ is the task vector, and

$V_A = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 & \dot{\theta}_6 & \dot{\theta}_7 \end{bmatrix}^T$ is the joint rate vector.

Numerical solutions are implemented using the Jacobian to follow the user directional motion commands or to follow the desired trajectory. Manipulability measure (Yoshikawa, 1990) is used as a factor to measure how far is the current configuration from singularity. This measure is defined as:

$$w = \sqrt{\det(J_A * J_A^T)} \tag{3}$$

Redundancy is resolved in the program structure using Pseudo Inverse of the Jacobian (Yoshikawa, 1990), and singularity is avoided by maximizing the manipulability measure. Since this method carries a guaranteed valid solution only at a singular configuration and not around it, the results carried high joint velocities when singularity is approached. We then decided to use S-R Inverse of the Jacobian (Nakamura, 1991) to give a better approximation around singularities, and use the optimization for different subtasks. S-R Inverse of the Jacobian is used to carry out the inverse kinematics as follows:

$$J_A^* = J_A^T * (J_A * J_A^T + k * I_6)^{-1} \tag{4}$$

where "I_6" is a 6x6 identity matrix, and "k" is a scale factor. It has been known that this method reduces the joint velocities near singularities, but compromises the accuracy of the solution by increasing the joint velocities error. Choosing the scale factor "k" is critical to minimizing the error. Since the point in using this factor is to give approximate solution near and at singularities, an adaptive scale factor is updated at every time step to put the proper factor as needed:

$$k = \left\langle \begin{matrix} k_0 * (1 - \frac{w}{w_0})^2 & for \quad w < w_0 \\ 0 & for \quad w \geq w_0 \end{matrix} \right. , \tag{5}$$

where "w_0" is the manipulability measure at the start of the boundary chosen when singularity is approached, and "k_0" is the scale factor at singularity. It was found that the optimum values of "w_0" and "k_0" for our system are $20\text{x}10^{-3}$ and $0.35\text{x}10^{-3}$ respectively. Now that the singularity is taken care of using the S-R Inverse of the Jacobian, we can use the joint redundancy to optimize for a secondary task as follows:

$$V_d = J_A^* * \dot{r}_d + (I_7 - J_A^* * J_A) * f \tag{6}$$

where "f" is a 7x1 vector representing the secondary task. That task can either be the desired trajectory in the case of pre-set task execution, or it can be a criterion function that represents the potential energy to be minimized.

3.3 The 9-DoF WMRA System Motion

Combining the two subsystems together by means of Jacobian augmentation (Luca et al., 2006) can give the flexibility of using conventional control and optimization methods without compromising the total system coordinated control. In the case of combined control, let the task vector be:

$$r = f(q_c, q_A \tag{7}$$

Differentiating (7) with respect to time gives:

$$\dot{r} = \frac{\partial f}{\partial q_c} V_c + \frac{\partial f}{\partial q_A} V_A \;= J_c J_W V_c + J_A V_A = \begin{bmatrix} J_c J_W & J_A \end{bmatrix} \begin{bmatrix} V_c \\ V_A \end{bmatrix} \;\text{ or, } \dot{r} = J \cdot V \tag{8}$$

Solving (8) in conventional methods is now possible. Choosing the Projected Gradient method, gives:

$$\begin{bmatrix} V_W \\ V_A \end{bmatrix} = J^* \dot{r} + \left(I - J^* J\right) V_0 \tag{9}$$

where $V_0 = a \nabla_q H(q)$ for conventional arms, and "H(q)" is the optimization criteria y = H(q).

The existence of the mobile platform means that "Vo" may not exist for non holonomic constraint such as that of the wheelchair. To go around this limitation (Luca et al., 2006) proposed the following: Differentiate the optimization criteria function "H" with respect to time as follows:

$$\dot{y} = \dot{H}(q) = \frac{\partial H}{\partial q_c} V_c + \frac{\partial H}{\partial q_A} V_A \;= \nabla_q^T H \begin{bmatrix} J_W & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} V_c \\ V_A \end{bmatrix} \tag{10}$$

In this case, the value of "VH" that improves the objective function is:

$$V_H = \pm a \begin{bmatrix} J_W^T & 0 \\ 0 & I \end{bmatrix} \nabla_q H(q) \equiv V_0 \tag{11}$$

and that velocity vector can be used for optimization. This gives a good representation of the arm joints' velocities and the wheels' velocities of the wheelchair.

Weighted Least Norm solution can also be used as proposed by (Chan & Dubey, 1995). In order to put a motion preference of one joint rather than the other (such as the wheelchair wheels and the arm joints), a weighted norm of the joint velocity vector can be defined as:

$$|V|_W = \sqrt{V^T W V} \tag{12}$$

where "W" is a 9X9 symmetric and positive definite weighting matrix, and for simplicity, it can be a diagonal matrix that represent the motion preference of each joint of the system. For the purpose of analysis, the following transformations are introduced:

$$J_W = J\,W^{-1/2} \text{ and } V_W = W^{-1/2}\,V \tag{13}$$

From (12) and (13), it can be shown that the weighted least norm solution is:

$$V_W = W^{-1} J^T \left(J\,W^{-1} J^T\right)^{-1} \dot{r} \tag{14}$$

The above method has been used in simulation of the 9-DoF WMRA system with the nine state variables "Vd" that represent the seven joint velocities of the arm and the two wheels' velocities of the power wheelchair. It was found that the latter two state variables are of limited use since they tend to unnecessarily rotate the wheelchair back and forth during a long forward motion due to their equal weights. Changing the weights of these two variables will only result in a preference of one's motion over the other.
Two new state variables were introduced out of the wheels' velocities, which represent the angular speed of the wheelchair when both wheels run at equal but opposite velocities and the forward speed of the wheelchair when both wheels run at equal velocities as follows:

$$\dot{\phi} = \frac{2 l_5 \dot{\theta}_r}{l_1}, \quad \textit{and} \quad \dot{X} = l_5 \dot{\theta}_r \tag{15}$$

The combination of the above two variables would be sufficient to describe any forward and rotational motion of the wheelchair. Having these two state variables in vector "V" instead of the wheels' velocities give a greater advantage in controlling the preferred rotation or translation of the wheelchair. The wheelchair's Jacobian in (1) was changed for the new state variables before augmenting it to the arm's Jacobian, and the results were much better in terms of valuable control. The new state variables are:

$$V = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 & \dot{\theta}_6 & \dot{\theta}_7 & \dot{X} & \dot{\phi} \end{bmatrix}^T \tag{16}$$

Care must be taken when implementing the above change in the controller algorithm since one of the state variables (the forward motion of the wheelchair) carry linear velocity units, which are different from the rest of the state variables, which carry angular velocity units.

3.4 Criterion Function for Joint Limit Avoidance

The criterion function used for optimization can be defined based on the physical joint limits of the WMRA system, and minimizing such a function results in limiting the joint motion to its limit. A mathematical representation of joint limits in robotic manipulators was proposed (Chan & Dubey, 1995) as follows:

$$H(q) = \sum_{i=1}^{7} \frac{1}{4} \cdot \frac{(q_{i,\max} - q_{i,\min})^2}{(q_{i,\max} - q_{i,current}) \cdot (q_{i,current} - q_{i,\min})} \tag{17}$$

where "q_i" is the angle of joint "i". This criterion function becomes "1" when the current joint angle is in the middle of its range, and it becomes "infinity" when the joint reaches either of its limits. Using this optimization function in (14) can be accomplished through the weight matrix used for optimization. Rather than choosing arbitrary weight values for each individual joint based on the user preference only, an additional value can be added to represent the optimization criterion function as follows:

$$W = \begin{bmatrix} w_{1,u} + \left|\frac{\partial H(q)}{\partial q_1}\right| & 0 & \cdots & 0 \\ 0 & w_{2,u} + \left|\frac{\partial H(q)}{\partial q_2}\right| & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{9,u} + \left|\frac{\partial H(q)}{\partial q_9}\right| \end{bmatrix} \tag{18}$$

where "$w_{i,u}$" is the user-defined weight preference to joint "i", and the second term in each element is the gradient projection of the criterion function defined as:

$$\frac{\partial H(q)}{\partial q_i} = \frac{(q_{i,\max} - q_{i,\min})^2 \cdot (2 \cdot q_{i,current} - q_{i,\max} - q_{i,\min})}{4 \cdot (q_{i,\max} - q_{i,current})^2 \cdot (q_{i,current} - q_{i,\min})^2} \tag{19}$$

When any particular joint is in the middle of the joint range, (19) becomes zero for that joint, and the only weight left is the user defined weight. On the other hand, when any particular joint is at its limit, (19) becomes "infinity", which means that the joint will carry an infinite weight that makes it impossible to move any further. When the user prefers to move the robotic arm with minimal wheelchair motion, heavy weight can be assigned to the two wheelchair state variables. When any of the robotic arm joints gets close to its limit and its weight approaches infinity, the wheelchair's weight will be much less than that of the joint, and hence it will be more free to move than the joint that is close to its limit.

It is important to note two different deficiencies that may lead to unintended operation or "joint lock" when using this method. The first deficiency is that the joint is penalized with higher weight whether it is approaching its limit or getting away from it. This may cause the robotic arm to use the null space inefficiently by preferring to move a joint with heavy weight going towards its limit rather than moving a joint with heavier weight that is moving away from its limit. This problem was eliminated using the first two conditions in (20) on the criterion function to identify the direction of approach (towards or away from) the joint limit. The second deficiency is that the precise joint limit that takes the weight to "infinity" may never be reached, instead, the numerical solution with its relatively coarse step size may jump from a joint value close to the joint limit before it is reached to a joint value close to the joint limit after it is reached. This will result in a heavy weight that will slowly get lower as the joint gets away from the set limit towards its actual limit. If the previous two conditions were applied alone, the result could be a dangerous motion that gives the weight as "$w_{i,u}$" only since the joint is getting away from its limit from inside that limit. This can either break the joint or lock it when it reaches its actual physical limit with the hard stop. To overcome this deficiency, the last two conditions in (20) were imposed on the criterion function to identify whether the joint is within its limit, or the limit is exceeded.

Imposing the above four conditions (see figure 5), on the weight matrix to perform on the optimization criterion gave the control mechanism much better results in terms of joint limit avoidance and user-preferred motion of each individual variable in the joint space The following condition statement summarizes the conditions imposed in the control code:

$$w_i = w_{i,u} + \begin{cases} \left|\dfrac{\partial H(q)}{\partial q_i}\right| & when \quad q_{\min} \le q_i \le q_{\max} \quad \& \quad \Delta\left|\dfrac{\partial H(q)}{\partial q_i}\right| \ge 0 \\ 0 & when \quad q_{\min} \le q_i \le q_{\max} \quad \& \quad \Delta\left|\dfrac{\partial H(q)}{\partial q_i}\right| \le 0 \\ \infty & when \quad q_{\min} \ge q_i \ge q_{\max} \quad \& \quad \Delta\left|\dfrac{\partial H(q)}{\partial q_i}\right| \le 0 \\ 0 & when \quad q_{\min} \ge q_i \ge q_{\max} \quad \& \quad \Delta\left|\dfrac{\partial H(q)}{\partial q_i}\right| \ge 0 \end{cases} \tag{20}$$

Two different trajectory generation functions were implemented in the control system when autonomous operation option was chosen: linear trajectory and polynomial trajectory with parabolic blending (Craig, 2003). The end point can either be given by the user, or can be obtained from an on-board laser pointer, as shown in Figure 6. Three different control reference frames were programmed so that the user can choose the most suited based on the task at hand: The ground frame for autonomous operation with pre-set tasks; the wheelchair's frame for wheelchair motion in the most part; and the end-effector's frame for teleoperation using the end-effector. The option of controlling the arm alone, the wheelchair alone, or the combined wheelchair and arm together was also programmed for the user's convenience.

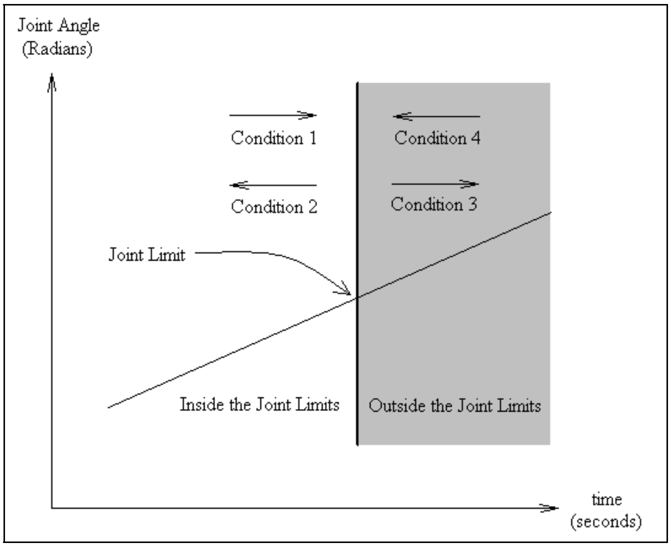
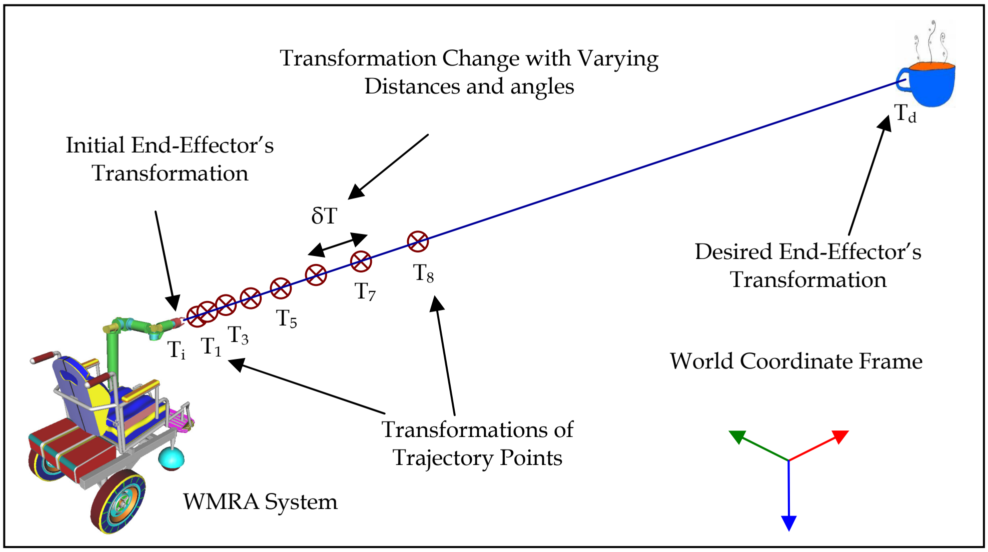


Fig. 5. Four joint limit boundary conditions.


Transformation Change with Varying Distances and angles
Initial End-Effector's Transformation
δT
T_d
Desired End-Effector's Transformation
T_8
T_7
T_5
T_3
T_i
T_1
World Coordinate Frame
Transformations of Trajectory Points
WMRA System
Fig. 6. Polynomial trajectory from the gripper to the target.

4. The 9-DoF WMRA System

4.1 Hardware Design of the Arm

An entirely new WMRA was developed, designed and built. The goal was to produce an arm that has better manipulability, greater payload, and easier control than current designs. As found in previous research (Edwards et al., 2006), side mounting is preferable overall because it provides the best balance between manipulability and unobtrusiveness. This mounting location allows the arm to be stowed by folding it back and then wrapping the forearm behind the seat. This helps avoid the stigma that these devices can bring. It virtually disappears when not in use, especially when the arm is painted to match the chair. However, care must be taken to prevent widening of the power chair. Our arm only increases chair width by 7.5cm.

This manipulator is intended for use in Activities of Daily Living (ADL), and for job tasks of a typical office environment. As such, it is important that the arm be strong enough to move objects that are common in these environments. Approximately 4 kg mass is set as the upper limit for a typical around-the-house object that must be manipulated. This was set as the baseline payload for the arm at full horizontal reach at rest. Then, an extra margin of 2 kg was added to allow for a choice of end-effector capable of this load. Reconfigurable arm lengths allow greater leverage on the engineering input, as a single basic design may be adapted to numerous applications. This is only practical with electric drive and actuator placement directly at each joint.

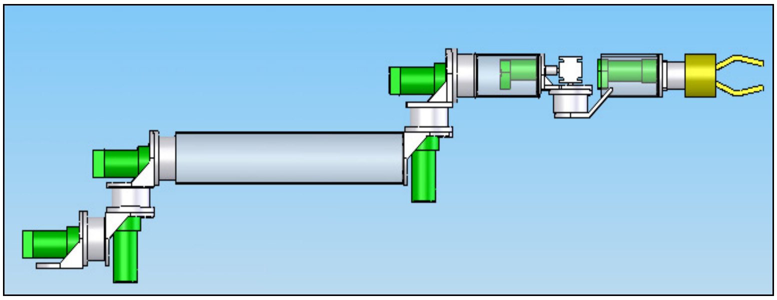


Fig. 7. Complete SolidWorks model of the arm.

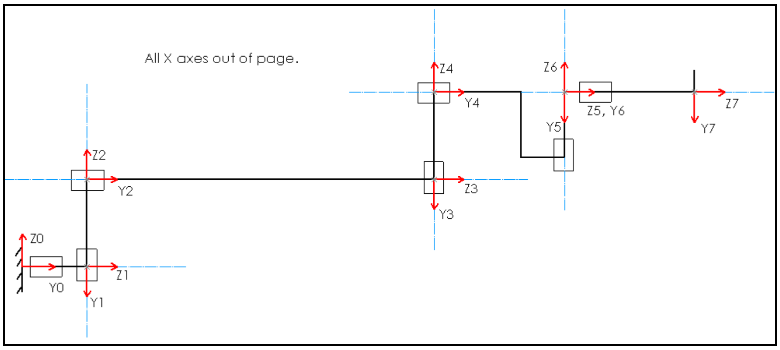


Fig. 8. Kinematic diagram with link frame assignments.

In the power wheelchair industry, a 24-volt lead-acid battery pack is standard, and is the natural choice for the power supply of a WMRA with minimum power consumption. To have a widespread adoption of these devices, reasonable cost is important. The target was to be in the mid-range of commercially available systems in terms of cost. Extra degrees of freedom help improve manipulability. This is evidenced by the considerable increase going from Raptor's 4 DoF to the 6 DoF of MANUS. Our new design incorporates 7 joints, allowing full pose control even in difficult regions of the workspace, such as reaching around the wheelchair, or up to a high shelf.
The arm is a 7-DoF design, using 7 revolute joints. It is anthropomorphic, with joints 1, 2 and 3 acting as a shoulder, joint 4 as an elbow, and joints 5, 6 and 7 as a wrist as shown in Figure 7. The 3 DoF shoulder allows the elbow to be positioned anywhere along a spherical surface, whereas with the Raptor arm, elbow movement is limited to a circle. Throughout the arm, adjacent joint axes are oriented at 90 degrees as shown in Figure 8. This helps to meet two goals: mechanical design simplicity and kinematic simplicity with low computational cost. All adjacent joint axes intersect, also simplifying the kinematics. The basic arrangement for each joint is a high-reduction gearhead, a motor with encoder and spur-gear reduction, and a bracket that holds these two parts and attaches to the two neighbouring links.

4.2 Hardware Design of the Controller

As shown in Fig 9, PIC-SERVO SC controllers (C1 through C7) that support the DC servo actuators (J1 through J7) were chosen. At 5cm x 7.5cm, this unit has a microprocessor that

drives the built-in amplifier with a PWM signal, handles PID position / velocity control, communicates with RS-485, and can be daisy-chained with up to 32 units. It also reads encoders, limit switches, an 8 bit analogue input, and supports coordinated motion control. Data for the entire arm is interfaced to the main computer using a single serial link. The PIC-Servo SC controllers use RS-485, and a hardware converter interfaces this with the RS-232 or a USB port on the host PC. A timer has been utilized to cut the arm's power off after a preset time to minimize power consumption while not in use. An emergency stop button is placed to cut the power off the motors and leave the logic power on so that the system can be diagnosed without rebooting.
The current host PC is an IBM laptop, running Windows XP. However, the communications protocol is simple and open, and could be adapted to virtually any hardware/software platform with an RS-232 or USB port. Currently, the tested user interfaces are the keyboard and a SpaceBall controller.

4.3 Wheelchair Modification

Figure 10 shows the WMRA system installed on the modified wheelchair "Action Ranger X Storm Series". The wheelchair has been modified by adding an incremental encoder on each one of the wheels. The controller module of the wheelchair has also been modified using TTL compatible signal conditioner and a DA converter so that the signal going to the wheels can be controlled using the same PIC-Servo SC controllers used in the arm. The only difference is that the output from this control board used for the wheelchair is the PWM signal rather than the amplified analogue signal. Two more PIC-Servo SC controllers were added to the control system shown in Figure 9 to control the wheelchair.

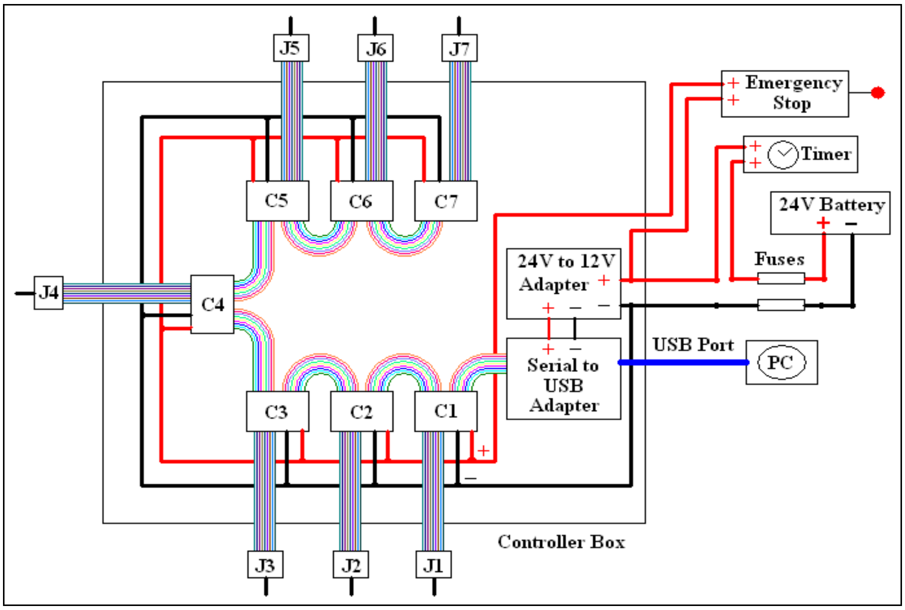


Fig. 9. Control system circuitry of the arm.

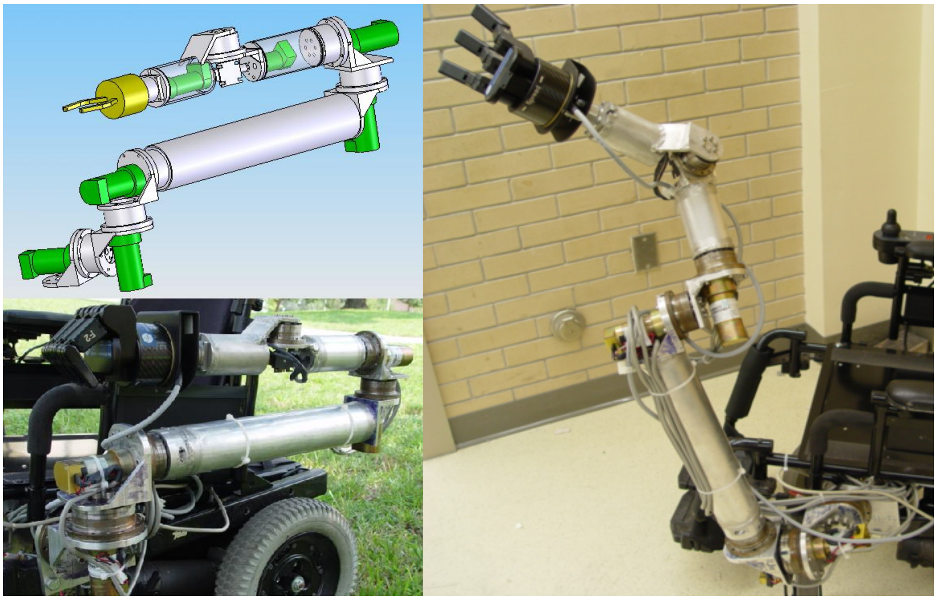


Fig. 10. WMRA SolidWorks models and the built device.

4.4 Hardware Design of an Ergonomic Gripper

A new robotic gripper was designed and constructed (Alqasemi et al., 2007) for Activities of Daily Living (ADL) to be used with the new WMRA. As shown if Figure 11, two aspects of the new gripper made it unique; one is the design of the paddles, and the other is the design of the actuation mechanism that produces parallel motion for effective gripping. The paddles of the gripper were designed to grasp a wide variety of objects with different shapes and sizes that are used in every day life as shown in Figure 12. The driving mechanism was designed to be simple, light, effective, safe, self content, and independent of the robotic arm attached to it.

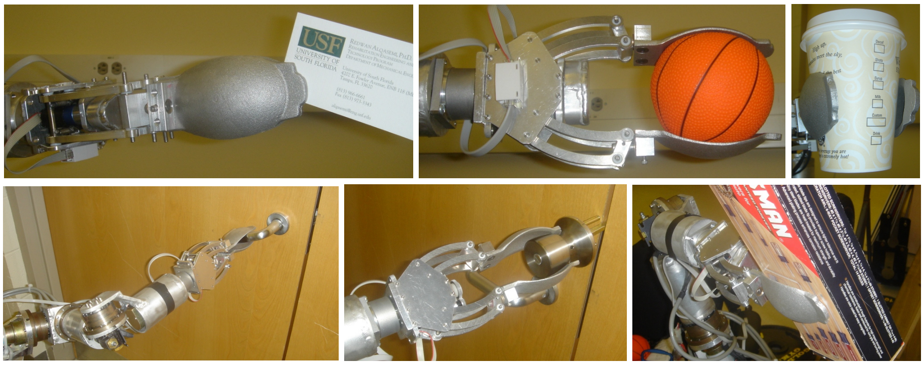


Fig. 12. Using the new gripper in typical ADL tasks.

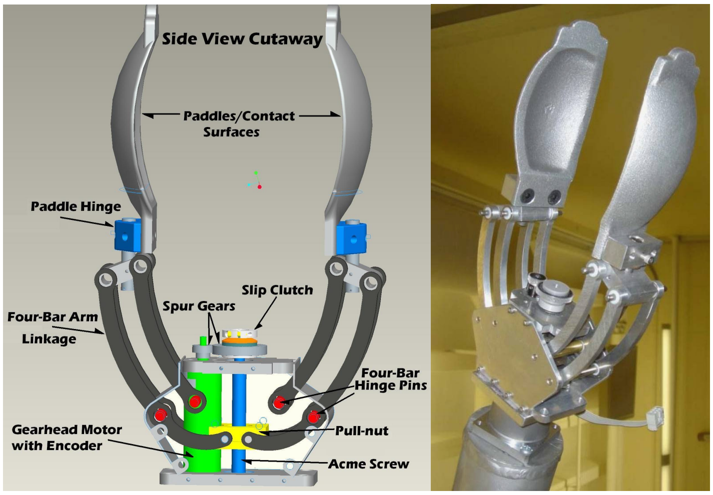


Fig. 11. The designed ergonomic gripper.

4.5 Simulation of the WMRA System in Virtual Reality

The control methods that combined the manipulation and mobility of the newly developed WMRA were tested in simulation before applying them to the actual WMRA system. This step is very important for debugging and inspecting the methods before applying them into the actual arm so that no harm to the physical system is done in case of unexpected errors. In the control software, several options were made available to include the modularity, reconfigurability and flexibility requirements for this WMRA system.
The control system was implemented in simulation using Matlab 2008 with Virtual Reality toolbox installed on a PC running Windows XP. Modules of small programs were generated for different operations and different user interfaces, and a main program that uses the modules was developed to simulate the WMRA system using different control parameters and user interfaces, including a Graphical User Interface (GUI). Figure 13 shows a sample of the Virtual Reality simulation.

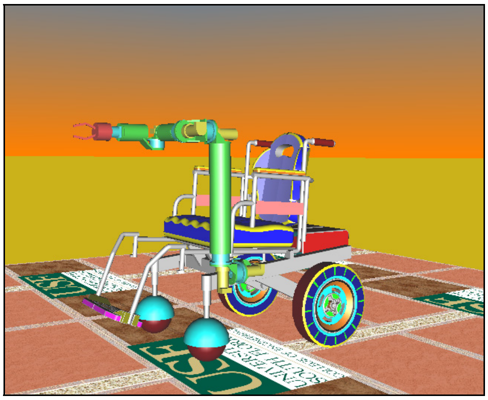


Fig. 13. Sample of the virtual reality simulation at the initial position.

5. The Integration of the Brain-Computer Interface

In the program structure, flexibility was one of the objectives in the design of user interfaces so that a wider range of these interfaces can be used based on the user's abilities and preference. A six-axis, twelve-way SpaceBall that is capable of moving in the six Cartesian coordinates was implemented. A computer keyboard and a mouse was also used with the on-screen graphical user interface. A touch screen with a choice of robotic actions was programmed as one of the integrated interfaces to the system. Figure 14 shows some of the user interface device options used in the system.
Another user interface used in this work is the Brain-Computer Interface (BCI). Over the past two decades, a variety of studies have evaluated the possibility that brain signals recorded from the scalp or from within the brain could provide new augmentative technology that does not require muscle control (Schalk et al., 2004).

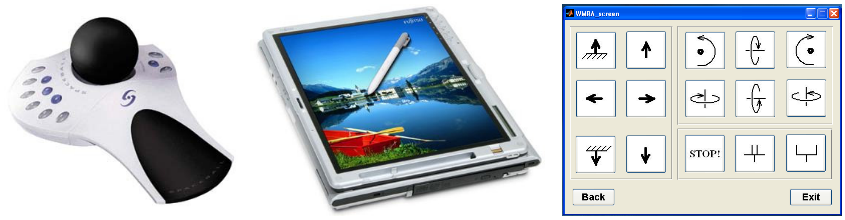


Fig. 14. User interfaces, left to right: SpaceBall, touch screen and GUI.

These BCI systems measure specific features of brain activity and translate them into device control signals as shown in Figure 15. This brain activity can be elicited using a framework called the "oddball paradigm". Studies have shown that when subjects are assigned a task of categorizing an item into 2 possible categories, and one of the two categories occurs infrequently, those events in that rare category will elicit an event-related brain potential (ERP) with latency of about 300 ms, labeled the P300 (Farwell & Donchin, 1988). The P300 is a neural evoked potential component of the EEG, or electroencephalogram (Sutton et al., 1965). It is supposed to follow unexpected sensory stimuli that provide useful information to the subjects according to his/her task. For easier, non-invasive use of this neuro-imaging technology, the user wears a head cap fitted with several electrodes to measure the P300 EEG signals from the activities of the brain.
In this work, the subjects viewed a 5x3 matrix whose rows and columns intensify randomly. Each of the 15 symbols in the matrix corresponds to a specific direction or task command as shown in Figures 15&16. The subject focuses attention on the desired cell carrying his/her desired task or direction of motion. Every time the user counts one more view of the symbol, the P300 EEG signal is recorded, and the corresponding row or column that was shown at that moment was also recorded. In about 15 seconds, the BCI gives the selected row and column of the matrix on the screen as only the row and column containing target cell elicit a P300. The system then translates the chosen character to a corresponding command, which translates into a Cartesian velocity in the proper direction and executes the algorithm to move the arm. As the detection of P300 requires signal averaging, number of trials is required by the system to correctly determine user's intention. The speed of the system thus depends on the number of sequences of flashes required to achieve a given level of accuracy.

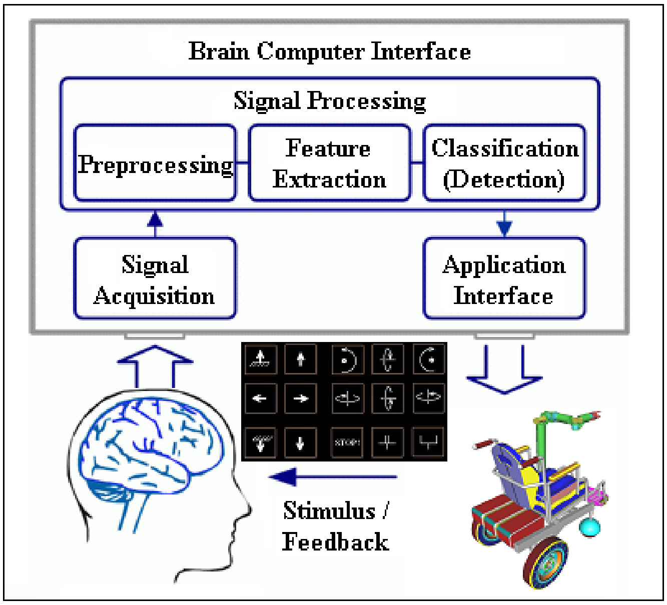


Fig. 15. Basic design and operation of the BCI system.

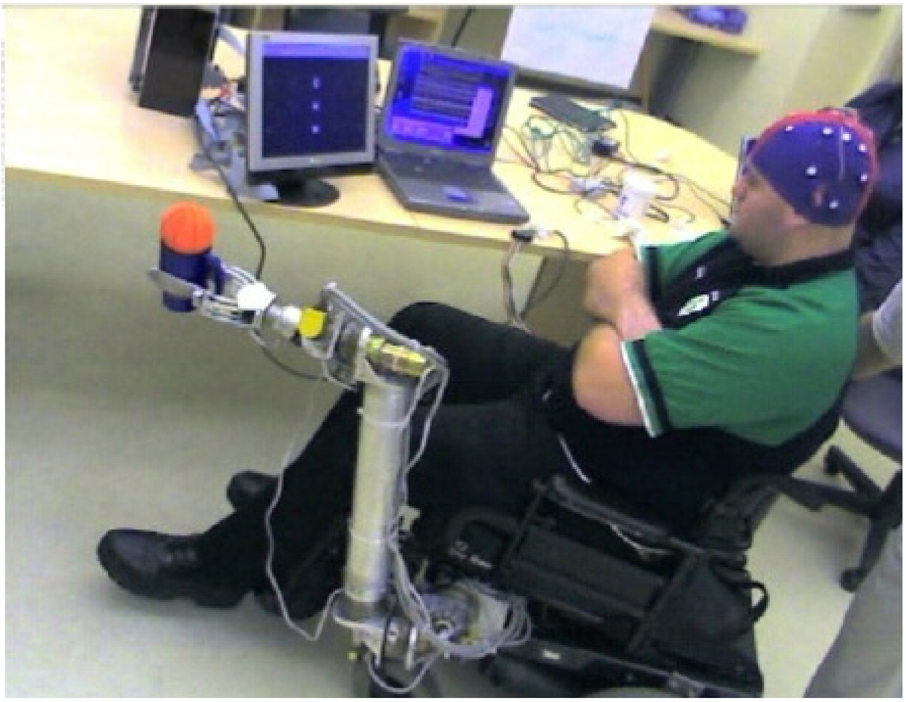


Fig. 16. WMRA user with BCI electrode cap.

Six able-bodied young adults were able to control the WMRA system movements using the BCI. Participants sat on the wheelchair, wearing a 16-channel electrode cap (Electro Cap International, Inc.) and attended to different symbols on the screen. Every 50 ms a row or a column intensified for 75 ms. For a 5x3 matrix, each sequence of flashes contained 8 intensifications (5 columns and 3 rows) and lasted for 1000 ms. The BCI was also trained to be optimized for each particular human subject, and it showed high accuracy of the selected choice (ranging from 92% to 100 %). These gains were recorded and used for the actual test. We tested the accuracy of character selection as a function of number of sequences of flashes. During the testing phase, a successful control with high accuracy of the motion response was apparent.

6. Experimental Results

The control method used in this work to combine mobility and manipulation in redundant mobile robots was tested using the developed Matlab program that can simulate the WMRA motion and control the physical WMRA system with various user interfaces. The WMRA was commanded to go in an autonomous mode from its default initial position shown in Figure 17 to a defined point in space for the end effector. Several methods were tested in this simulation. In this paper, we will limit our findings to the Weighted Least Norm solution control, and we will discuss the system response using different control methods in an extreme case where the arm was commanded to go to an out-of-reach position.

6.1 Simulation Results Using Different Weights

Three different values were tried for the diagonal elements of the weight matrix "W_d" to implement the control system and to verify its effectiveness in damping the wheelchair motion or the arm motion. Figures 18 through 20 show the final poses of the WMRA system after the end-effector reached the desired destination for the five cases studied.

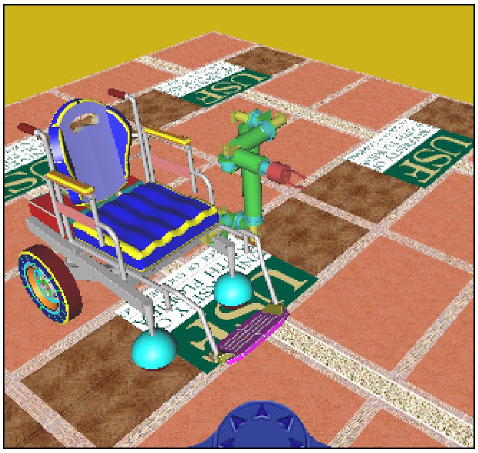


Fig. 17. Initial WMRA pose.

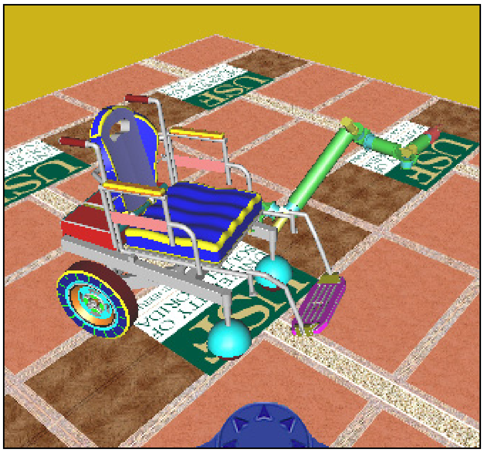


Fig. 18. Destination pose - *Wd* = [1, 1, 1, 1, 1, 1, 1, 1, 1].

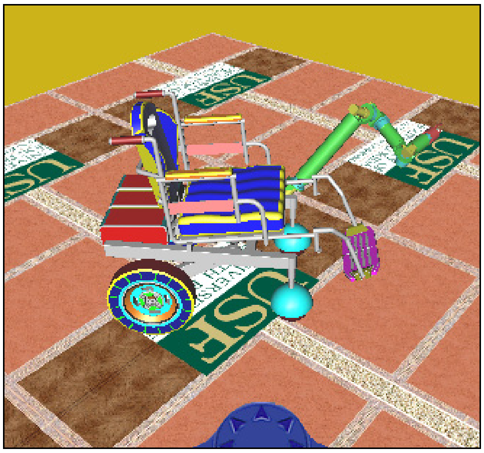


Fig. 19. Destination pose - *Wd* = [10, 10, 10, 10, 10, 10, 10, 1, 1].

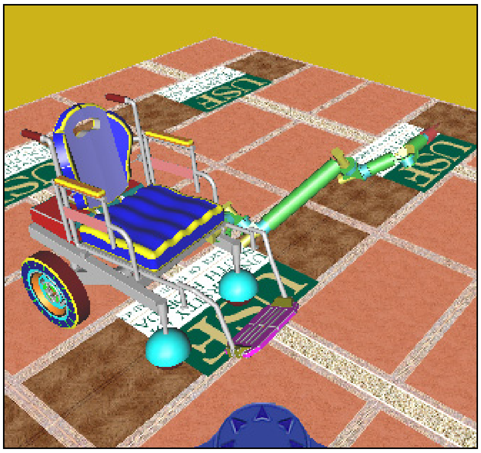


Fig. 20. Destination pose - *Wd* = [1, 1, 1, 1, 1, 1, 1, 100, 100].

The weight matrix of the first case carried in its diagonal elements "1" for each of the arm's seven joints, and "1" for wheelchair's position and orientation variables, which means that the wheelchair's two variables and the arm's joints carry the same tendency for motion, as shown in Figure 18. In this case, the weight matrix is useless since it is identity. The weight matrix of the second case carried in its diagonal elements "10" for each of the arm's seven joints, and "1" for wheelchair's position and orientation variables, which means that the wheelchair's two variables are 10 times more likely to move than the arm's joints, and that is apparent in the results shown in Figure 19. In the third case, "W_d" carried weights of "1" for the arm's seven joints, and a weight of "100" for the wheelchair's position and orientation. This means that the arm's joints are 100 times more likely to move than the wheelchair's two variables, and Figure 20 shows how the wheelchair's motion was minimal.
The simulation program was designed to give different useful values and plots throughout the simulation process for observation and diagnosis of any potential problems that might occur during the task execution. In the first case, when all 9 variables carried the same weights, an apparent motion in the arm and the wheelchair alike occurred. In the second case, when the arm carried a heavy weight in the weight matrix, it was clear that the seven arm joints had minimal motion that was necessary for the destination to be reached. That end-effector destination was impossible to reach by using the two wheelchair's variables only. The third case shows an easy arm motion and very minimal wheelchair motion that was necessary to avoid singularity.
An important property of this optimization method was apparent during simulation, which was the minimization of singularity. As the arm was moving to the destination and both wheels were moving backwards, the wheels reversed their motion in the middle of the simulation period when the arm started approaching singularity. Figures 21 through 23 show the manipulability index of both arm only and the combined WMRA system. It is important to note here that these values were multiplied by (10^{-9}) to get the normalized manipulability measure. It is clear that the manipulability is much higher for the WMRA

system than that of the arm only due to the fact that the WMRA system carries two more degrees of freedom.

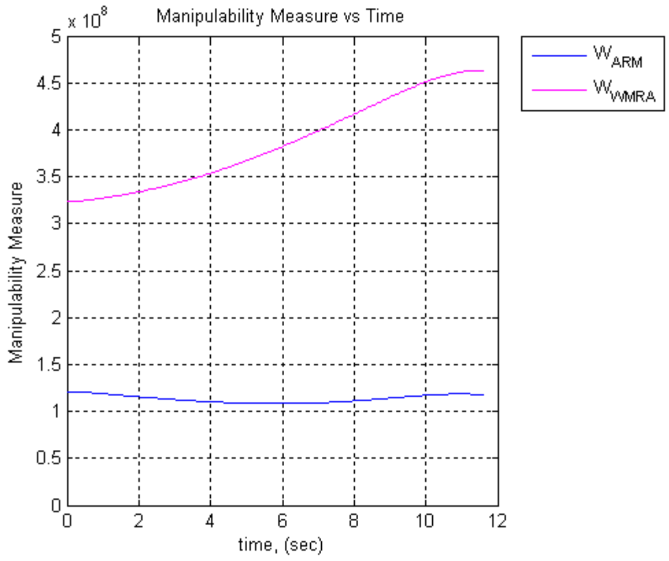


Fig. 21. Manipulability index - *Wd* = [1, 1, 1, 1, 1, 1, 1, 1, 1].

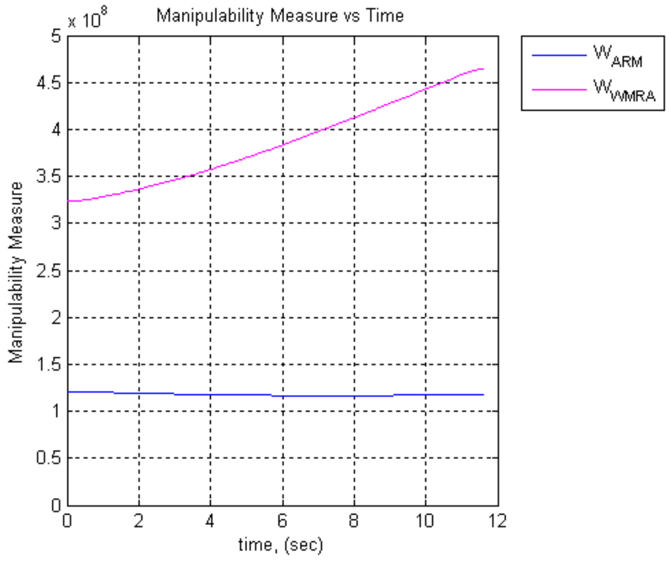


Fig. 22. Manipulability index - *Wd* = [10, 10, 10, 10, 10, 10, 10, 1, 1].

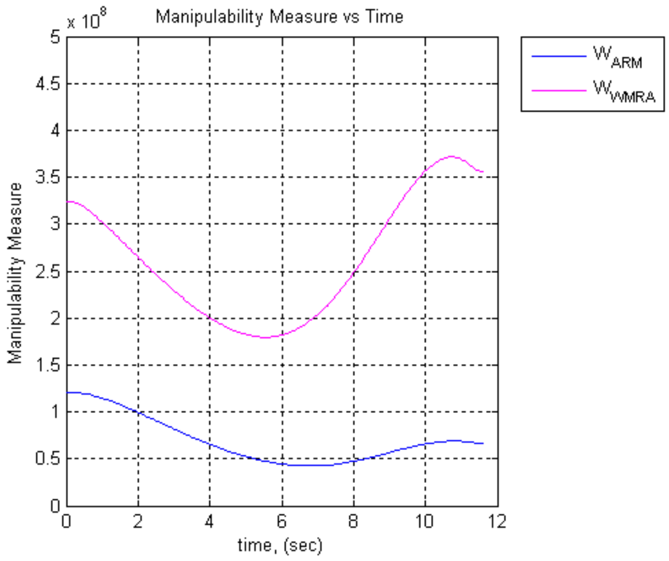


Fig. 23. Manipulability index - *Wd* = [1, 1, 1, 1, 1, 1, 1, 100, 100].

In all three cases, the manipulability measure was maximized based on the weight matrix. Figure 21 shows an improvement trend of the WMRA's manipulability index over the arm's manipulability index towards the end of simulation. Figure 22 shows the manipulability of the arm as nearly constant compared to that in Figure 23 because of the minimal motion of the arm. Figure 23 shows how the wheelchair started moving rapidly later in the simulation (see figure 20) as the arm approached singularity, even though the weight of the wheelchair motion was heavy. This helped in improving the WMRA system's manipulability.

6.2 Simulation Results in an Extreme Case

To test the difference in the system response when using different methods, an extreme case was tested, where the WMRA system is commanded to reach a point that is physically unreachable. The end-effector was commanded to move horizontally and vertically upwards to a height of 1.3 meters from the ground, which is physically unreachable, and the WMRA system will reach singularity. The response of the system can avoid that singularity depending on the method used. Singularity, joint limits and preferred joint-space weights were the three factors we focused on in this part of the simulation. Eight control cases simulated were as follows:

(a) Case I: Pseudo inverse solution (PI): In this case, the system was unstable, the joints went out of bounds, and the user had no weight assignment choice.
(b) Case II: Pseudo inverse solution with the gradient projection term for joint limit avoidance (PI-JL): In this case, the system was unstable, the joints stayed in bounds, and the user had no weight assignment choice.
(c) Case III: Weighted Pseudo inverse solution (WPI): In this case, the system was unstable, the joints went out of bounds, and the user had weight assignment choices.

(d) Case IV: Weighted Pseudo inverse solution with joint limit avoidance (WPI-JL): In this case, the system was unstable, the joints stayed in bounds, and the user had weight assignment choices.
(e) Case V: S-R inverse solution (SRI): In this case, the system was stable, the joints went out of bounds, and the user had no weight assignment choice.
(f) Case VI: S-R inverse solution with the gradient projection term for joint limit avoidance (SRI-JL): In this case, the system was unstable, the joints stayed in bounds, and the user had no weight assignment choice.
(g) Case VII: Weighted S-R inverse solution (WSRI): In this case, the system was stable, the joints went out of bounds, and the user had weight assignment choices.
(h) Case VIII: Weighted S-R inverse solution with joint limit avoidance (WSRI-JL): In this case, the system was stable, the joints stayed in bounds, and the user had weight assignment choices.

In the first case, Pseudo inverse was used in the inverse Kinematics without integrating the weight matrix or the gradient projection term for joint limit avoidance. Figure 24 shows how this conventional method led to the singularity of both the arm and the WMRA system. The user's preference of weight was not addressed, and the joint limits were discarded. In the last case, the developed method that uses weighted S-R inverse and integrates the gradient projection term for joint limit avoidance was used in the inverse kinematics. Figure 25 shows the best performance of all tested methods since it fulfilled all the important control requirements. This last method avoided singularities while keeping the joint limits within bounds and satisfying the user-specified weights as much as possible. The desired trajectory was followed until the arm reached its maximum reach perpendicular to the ground. Then it started pointing towards the current desired trajectory point, which minimizes the position errors. Note that the arm reaches the minimum allowed manipulability index, but when combined with the wheelchair, that index stays farther from singularity.

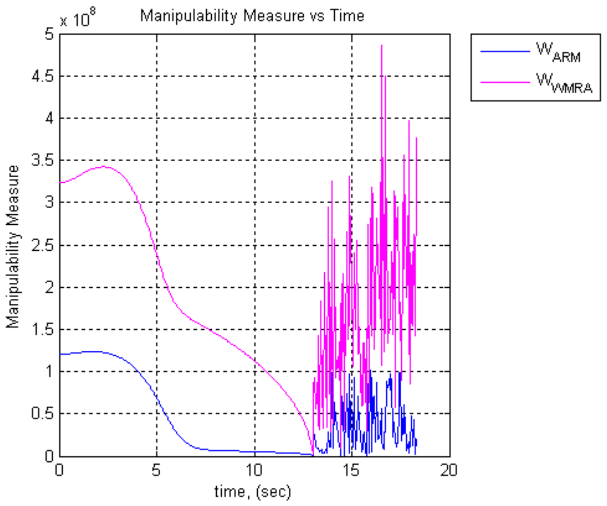


Fig. 24. Manipulability index – using only Pseudo inverse in an extreme case.

It is important to mention that changing the weights of each of the state variables gives motion priority to these variables, but may lead to singularity if heavy weights are given to certain variables when they are necessary for particular motions. For example, when the seven joints of the arm were given a weight of "1000" and the task required rapid motion of the arm, singularity occurred since the joints were nearly stationary. Changing these weights dynamically in the control loop depending on the task in hand leads to a better performance. This subject will be explored and published in a later publication.

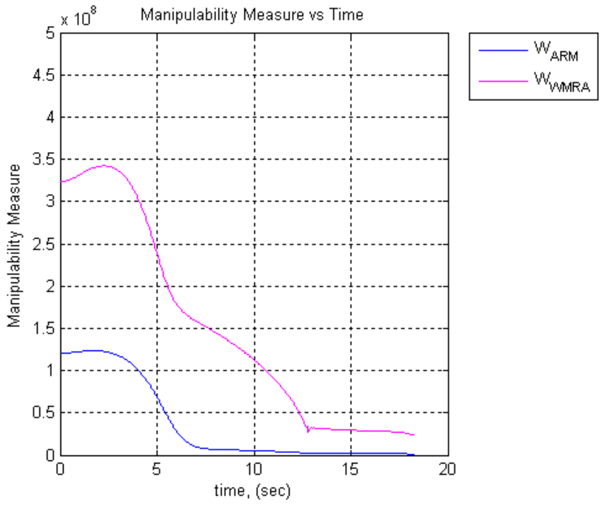


Fig. 25. Manipulability index – using weighted S-R inverse with the gradient projection term for joint limit avoidance in an extreme case.

6.3 Clinical Testing on Human Subjects

In the teleoperation mode of the testing, several user interfaces were tested. Figure 29 shows the WMRA system with the Barrette hand installed and a video camera used by a person affected by Guillain-Barre Syndrome. In her case, she was able to use both the computer interface and the touch-screen interface. Other user interfaces were tested, but in this paper, we will discuss the BCI user interface results. When asked, participants informed the tester that they preferred the 4 and 6 sequences of flashes over the longer sequences. The common explanation was that it was easier to stay focused for shorter periods of time. Figure 30 shows accuracy data obtained when participants spelled 50 characters of each set of sequences (12, 10, 8, 6, 4, and 2). As the number of sequences of flashes decrease, the speed of the BCI system increases as the maximum number of characters read per unit of time increases. This compromise affects the accuracy of the selected characters. Figure 31 shows the mean percentages correct for each of the sequences. The percentages are presented as number of maximum characters per minute.

The results call for the evaluation of the speed accuracy trade-off in an online mode rather than in an offline analysis to account for the users' ability to attend to a character over time. Few potential problems were noticed as follows: Every full scan of a single user input takes about 15 second, and that might cause a delay in the response of the WMRA system to change direction on time as the human user wishes. This 15-second delay may cause problems in case the operator needs to stop the WMRA system for a dangerous situation such as approaching stairs, or if the user made the wrong selection and needed to return back to his original choice.

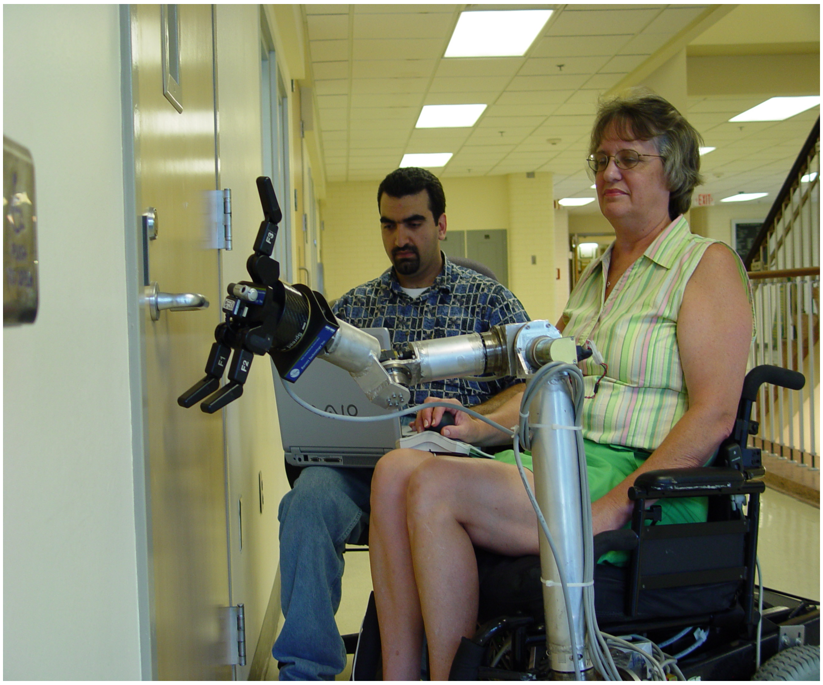


Fig. 29. A person with Guillain-Barre Syndrome driving the WMRA system.

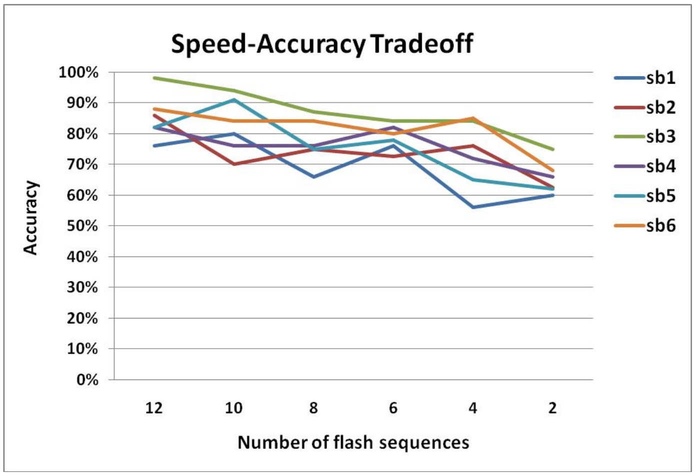


Fig. 30. Accuracy data (% correct) for 6 human subjects.

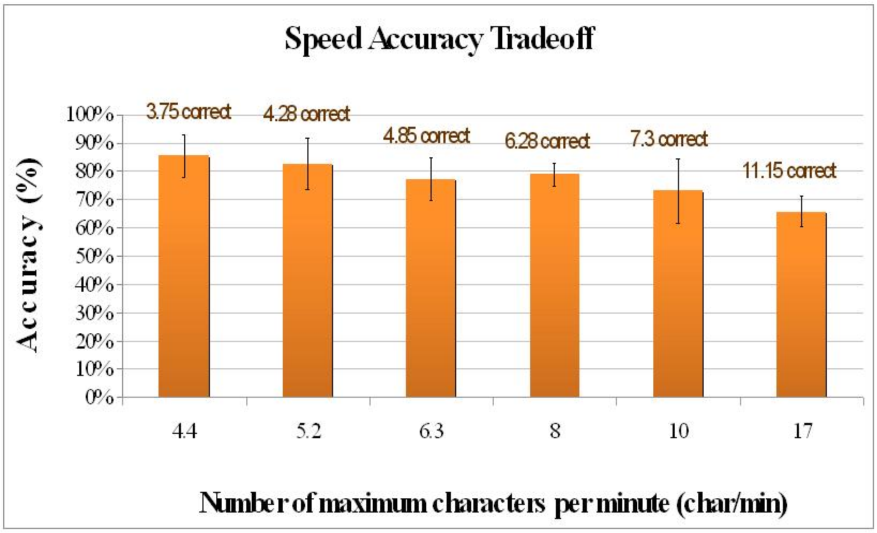


Fig. 31. Accuracy data (% correct) for each of the flash sequences.

It is also noted that after an extended period of time in using the BCI system, fatigue starts to appear on the user due to his concentration on the screen when counting the appearances of his chosen symbol. This tiredness on the user's side can be a potential problem.
Furthermore, when the user needs to constantly look at the screen and concentrate on the chosen symbol, This will distract him from looking at where the WMRA is going, and that poses some danger on the user. Despite the above noted problems, a successful interface with a good potential for a novel application was developed. Further refinement of the BCI interface is needed to minimize potential risks.

7. Conclusions and Recommendations

A wheelchair-mounted robotic arm (WMRA) was designed and built to meet the needs of mobility-impaired persons, and to exceed the capabilities of current devices of this type. Combining the wheelchair control and the arm control through the augmentation of the Jacobian to include representations of both resulted in a control system that effectively and simultaneously controls both devices at once. The control system was designed for coordinated Cartesian control with singularity robustness and task-optimized combined mobility and manipulation. Weighted Least Norm solution was implemented to prioritize the motion between different arm joints and the wheelchair.
Modularity in both the hardware and software levels allowed multiple input devices to be used to control the system, including the Brain-Computer Interface (BCI). The ability to communicate a chosen character from the BCI to the controller of the WMRA was presented, and the user was able to control the motion of WMRA system by focusing attention on a specific character on the screen. Further testing of different types of displays (e.g. commands, picture of objects, and a menu display with objects, tasks and locations) is planned to facilitate communication, mobility and manipulation for people with severe

disabilities. Testing of the control system was conducted in Virtual Reality environment as well as using the actual hardware developed earlier. The results were presented and discussed.

The authors would like to thank and acknowledge Dr. Emanuel Donchin, Dr. Yael Arbel, Dr. Kathryn De Laurentis, and Dr. Eduardo Veras for their efforts in testing the WMRA with the BCI system. This effort is supported by the National Science Foundation.

8. References

Alqasemi, R.; Mahler, S.; Dubey, R. (2007). “Design and construction of a robotic gripper for activities of daily living for people with disabilities,” Proceedings of the 2007 ICORR, Noordwijk, the Netherlands, June 13–15.

Alqasemi, R.M.; McCaffrey, E.J.; Edwards, K.D. and Dubey, R.V. (2005). “Analysis, evaluation and development of wheelchair-mounted robotic arms,” Proceedings of the 2005 ICORR, Chicago, IL, USA.

Chan, T.F.; Dubey, R.V. (1995). “A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators,” IEEE Robotics and Automation Transactions (R&A Transactions 1995). Vol. 11, Issue 2, pp. 286-292.

Chung, J.; Velinsky, S. (1999). “Robust interaction control of a mobile manipulator - dynamic model based coordination,” Journal of Intelligent and Robotic Systems, Vol. 26, No. 1, pp. 47-63.

Craig, J. (2003). “Introduction to robotics mechanics and control,” Third edition, Addison-Wesley Publishing, ISBN 0201543613.

Edwards, K.; Alqasemi, R.; Dubey, R. (2006). “Design, construction and testing of a wheelchair-mounted robotic arm,” Proceedings of the 2006 ICRA, Orlando, FL, USA.

Eftring, H.; Boschian, K. (1999). “Technical results from manus user trials,” Proceedings of the 1999 ICORR, 136-141.

Farwell, L.; Donchin, E. (1988). “Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials,” Electroencephalography and Clinical Neurophysiology, 70, 510–523.

Galicki, M. (2005). “Control-based solution to inverse kinematics for mobile manipulators using penalty functions,” 2005 Journal of Intelligent and Robotic Systems, Vol. 42, No. 3, pp. 213-238.

Luca, A.; Oriolo, G.; Giordano, P. (2006). “Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators,” Proceedings of the 2006 ICRA, pp. 1867-1873.

Lüth, T.; Ojdaniæ, D.; Friman, O.; Prenzel, O.; and Gräser, A. (2007). “Low level control in a semi-autonomous rehabilitation robotic system via a Brain-Computer Interface,” Proceedings of the 2007 ICORR, Noordwijk, The Netherlands.

Mahoney, R. M. (2001). “The Raptor wheelchair robot system”, Integration of Assistive Technology in the Information Age. pp. 135-141, IOS, Netherlands.

Nakamura, Y. (1991). “Advanced robotics: redundancy and optimisation,” Addison- Wesley Publishing, ISBN 0201151987.

Papadopoulos, E.; Poulakakis, J. (2000). “Planning and model-based control for mobile manipulators,” Proceedings of the 2001 IROS.

Reswick, J.B. (1990). “The moon over dubrovnik - a tale of worldwide impact on persons with disabilities,” Advances in External Control of Human Extremities.

Schalk, G.; McFarland, D.; Hinterberger, T.; Birbaumer, N.; and Wolpaw, J. (2004). “BCI2000: A general-purpose brain-computer interface (BCI) system,” IEEE Transactions on Biomedical Engineering, V. 51, N. 6, pp. 1034-1043.

Sutton, S.; Braren, M.; Zublin, J. and John, E. (1965). “Evoked potential correlates of stimulus uncertainty,” Science, V. 150, pp. 1187–1188.

US Census Bureau, “Americans with disabilities: 2002,” Census Brief, May 2006, http://www.census.gov/prod/2006pubs/p70-107.pdf

Valbuena, D.; Cyriacks, M.; Friman, O.; Volosyak, I.; and Gräser, A. (2007). “Brain-computer interface for high-level control of rehabilitation robotic systems,” Proceedings of the 2007 ICORR, Noordwijk, The Netherlands.

Yanco, Holly (1998). “Integrating robotic research: a survey of robotic wheelchair development,” AAAI Spring Symposium on Integrating Robotic Research, Stanford, California.

Yoshikawa, T. (1990). “Foundations of robotics: analysis and control,” MIT Press, ISBN 0262240289.

Advanced Techniques of Industrial Robot Programming

Frank Shaopeng Cheng
Central Michigan University
United States

1. Introduction

Industrial robots are reprogrammable, multifunctional manipulators designed to move parts, materials, and devices through computer controlled motions. A robot application program is a set of instructions that cause the robot system to move the robot's end-of-arm-tooling (or end-effector) to robot points for performing the desired robot tasks. Creating accurate robot points for an industrial robot application is an important programming task. It requires a robot programmer to have the knowledge of the robot's reference frames, positions, software operations, and the actual programming language. In the conventional "lead-through" method, the robot programmer uses the robot teach pendant to position the robot joints and end-effector via the actual workpiece and record the satisfied robot pose as a robot point. Although the programmer's visual observations can make the taught robot points accurate, the required teaching task has to be conducted with the real robot online and the taught points can be inaccurate if the positions of the robot's end-effector and workpiece are slightly changed in the robot operations. Other approaches have been utilized to reduce or eliminate these limitations associated with the online robot programming. This includes generating or recovering robot points through user-defined robot frames, external measuring systems, and robot simulation software (Cheng, 2003; Connolly, 2006; Pulkkinen1 et al., 2008; Zhang et al., 2006).
Position variations of the robot's end-effector and workpiece in the robot operations are usually the reason for inaccuracy of the robot points in a robot application program. To avoid re-teaching all the robot points, the robot programmer needs to identify these position variations and modify the robot points accordingly. The commonly applied techniques include setting up the robot frames and measuring their positional offsets through the robot system, an external robot calibration system (Cheng, 2007), or an integrated robot vision system (Cheng, 2009; Connolly, 2007). However, the applications of these measuring and programming techniques require the robot programmer to conduct the integrated design tasks that involve setting up the functions and collecting the measurements in the measuring systems. Misunderstanding these concepts or overlooking these steps in the design technique will cause the task of modifying the robot points to be ineffective.
Robot production downtime is another concern with online robot programming. Today's robot simulation software provides the robot programmer with the functions of creating

virtual robot points and programming virtual robot motions in an interactive and virtual 3D design environment (Cheng, 2003; Connolly, 2006). By the time a robot simulation design is completed, the simulation robot program is able to move the virtual robot and end-effector to all desired virtual robot points for performing the specified operations to the virtual workpiece without collisions in the simulated workcell. However, because of the inevitable dimensional differences of the components between the real robot workcell and the simulated robot workcell, the virtual robot points created in the simulated workcell must be adjusted relative to the actual position of the components in the real robot workcell before they can be downloaded to the real robot system. This task involves the techniques of calibrating the position coordinates of the simulation Device models with respect to the user-defined real robot points.

In this chapter, advanced techniques used in creating industrial robot points are discussed with the applications of the FANUC robot system, Delmia IGRIP robot simulation software, and Dynalog DynaCal robot calibration system. In Section 2, the operation and programming of an industrial robot system are described. This includes the concepts of robot's frames, positions, kinematics, motion segments, and motion instructions. The procedures for teaching robot frames and robot points online with the real robot system are introduced. Programming techniques for maintaining the accuracy of the exiting robot points are also discussed. Section 3 introduces the setup and integration of a two dimensional (2D) vision system for performing vision-guided robot operations. This includes establishing integrated measuring functions in both robot and vision systems and modifying existing robot points through vision measurements for vision-identified workpieces. Section 4 discusses the robot simulation and offline programming techniques. This includes the concepts and procedures related to creating virtual robot points and enhancing their accuracy for a real robot system. Section 5 explores the techniques for transferring industrial robot points between two identical robot systems and the methods for enhancing the accuracy of the transferred robot points through robot system calibration. A summary is then presented in Section 6.

2. Creating Robot Points Online with Robot

The static positions of an industrial robot are represented by Cartesian reference frames and frame transformations. Among them, the robot base frame R(x, y, z) is a fixed one and the robot's default tool-center-point frame Def_TCP (n, o, a), located at the robot's wrist faceplate, is a moving one. The position of frame Def_TCP relative to frame R is defined as the robot point $P[n]^{R}_{Def\_TCP}$ and is mathematically determined by the 4 × 4 homogeneous transformation matrix in Eq. (1)

$$P[n]^{R}_{Def\_TCP} = {}^{R}T_{Def\_TCP} = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where the coordinates of vector p = (p_x, p_y, p_z) represent the location of frame Def_TCP and the coordinates of three unit directional vectors n, o, and a represent the orientation of frame

Def_TCP. The inverse of ${}^{R}T_{Def\_TCP}$ or $P[n]^{R}_{Def\_TCP}$ denoted as $({}^{R}T_{Def\_TCP})^{-1}$ or $(P[n]^{R}_{Def\_TCP})^{-1}$ represents the position of frame R to frame Def_TCP, which is equal to frame transformation ${}^{Def\_TCP}T_{R}$. Generally, the definition of a frame transformation matrix or its inverse described above can be applied for measuring the relative position between any two frames in the robot system (Niku, 2001). The orientation coordinates of frame Def_TCP in Eq. (1) can be determined by Eq. (2)

$$\begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix} = Rot(z,\theta_z)Rot(y,\theta_y)Rot(x,\theta_x)$$
$$= \begin{bmatrix} \cos\theta_z\cos\theta_y & \cos\theta_z\sin\theta_y\sin\theta_x - \sin\theta_z\cos\theta_x & \cos\theta_z\sin\theta_y\cos\theta_x + \sin\theta_z\sin\theta_x \\ \sin\theta_z\cos\theta_y & \sin\theta_z\sin\theta_y\sin\theta_x + \cos\theta_z\cos\theta_x & \sin\theta_z\sin\theta_y\cos\theta_x - \cos\theta_z\sin\theta_x \\ -\sin\theta_y & \cos\theta_y\sin\theta_x & \cos\theta_y\cos\theta_x \end{bmatrix}, \quad (2)$$

where transformations Rot(x, θ_x), Rot(y, θ_y), and Rot(z, θ_z) are pure rotations of frame Def_TCP about the x-, y-, and z-axes of frame R with the angles of θ_x (yaw), θ_y (pitch), and θ_z (roll), respectively. Thus, a robot point $P[n]^{R}_{Def\_TCP}$ can also be represented by Cartesian coordinates in Eq. (3)

$$P[n]^{R}_{Def\_TCP} = (x, y, z, w, p, r) \cdot \quad (3)$$

It is obvious that the robot's joint movements are to change the position of frame Def_TCP. For an n-joint robot, the geometric motion relationship between the Cartesian coordinates of a robot point $P[n]^{R}_{Def\_TCP}$ in frame R (i.e. the robot world space) and the proper displacements of its joint variables q = (q_1, q_2, ..q_n) in robot joint frames (i.e. the robot joint space) is mathematically modeled as the robot's kinematics equations in Eq. (4)

$$\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_{11}(q,r) & f_{12}(q,r) & f_{13}(q,r) & f_{14}(q,r) \\ f_{21}(q,r) & f_{22}(q,r) & f_{23}(q,r) & f_{24}(q,r) \\ f_{31}(q,r) & f_{32}(q,r) & f_{33}(q,r) & f_{34}(q,r) \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

where $f_{ij}(q, r)$ (for i = 1, 2, 3 and j = 1, 2, 3, 4) is a function of joint variables q and joint parametersr.

Specifically, the robot forward kinematics equations will enable the robot system to determine where a $P[n]^{R}_{Def\_TCP}$ will be if the displacements of all joint variables q=(q_1, q_2, ..q_n) are known. The robot inverse kinematics equations will enable the robot system to calculate what displacement of each joint variable q_k (for k = 1 ,..., n) must be if a $P[n]^{R}_{Def\_TCP}$ is specified. If the inverse kinematics solutions for a given $P[n]^{R}_{Def\_TCP}$ are infinite, the robot system defines the point as a robot "singularity" and cannot move frame Def_TCP to it.

In robot programming, the robot programmer creates a robot point $P[n]^{R}_{Def\_TCP}$ by first declaring it in a robot program and then defining its coordinates in the robot system. The conventional method is through recording a particular robot pose with the robot teach pendent (Rehg, 2003). Under the teaching mode, the robot programmer jogs the robot's joints for poisoning the robot's end-effector relative to the workpiece. As joint k moves, the serial pulse coder of the joint measures the joint displacement q_k relative to the "zero" position of the joint frame. The robot system substitutes all measured values of $q = (q_1, q_2, ..q_n)$ into the robot forward kinematics equations to determine the corresponding Cartesian coordinates of frame Def_TCP in Eq. (1) and Eq. (3). After the robot programmer records a $P[n]^{R}_{Def\_TCP}$ with the teach pendant, its Cartesian coordinates and the corresponding joint values are saved in the robot system. The robot programmer may use the "Representation" softkey on the teach pendant to automatically convert and display the joint values and Cartesian coordinates of a taught robot point $P[n]^{R}_{Def\_TCP}$. It is important to notice that Cartesian coordinates in Eq. (3) is the standard representation of a $P[n]^{R}_{Def\_TCP}$ in the industrial robot system, and its joint representation always uniquely defines the position of frame Def_TCP (i.e. the robot pose) in frame R.

In robot programming, the robot programmer defines a motion segment of frame Def_TCP by using two taught robot points in a robot motion instruction. During the execution of a motion instruction, the robot system utilizes the trajectory planning method called "linear segment with parabolic blends" to control the joint motion and implement the actual trajectory of frame Def_TCP through one of the two user-specified motion types. The "joint" motion type allows the robot system to start and end the motion of all robot joints at the same time resulting in an unpredictable, but repeatable trajectory for frame Def_TCP. The "Cartesian" motion type allows the robot system to move frame Def_TCP along a user-specified Cartesian path such as a straight line or a circular arc in frame R during the motion segment, which is implemented in three steps. First, the robot system interpolates a number of intermediate points along the specified Cartesian path in the motion segment. Then, the proper joint values for each interpolated robot point are calculated by the robot inverse kinematics equations. Finally, the "joint" motion type is applied to move the robot joints between two consecutive interpolated robot points.

Different robot languages provide the robot systems with motion instructions in different format. The motion instruction of FANUC Teach Pendant Programming (TPP) language (Fanuc, 2007) allows the robot programmer to define a motion segment in one statement that includes the robot point P[n], motion type, speed, motion termination type, and associated motion options. Table 1 shows two motion instructions used in a FANUC TP program.

FANUC TPP Instruction	Description
1. J P[1] 50% FINE	Moves the TCP frame to robot point P[1] with "Joint" motion type (J) and at 50% of the default joint maximum speed, and stops exactly at P[1] with a "Fine" motion termination.
2. L P[2] 100 mm/sec FINE	Utilizes "Linear" motion type (L) to move TCP frame along a straight line from P[1] to P[2] with a TCP speed of 100 mm/sec and a "Fine" motion termination type.

Table 1. Motion instructions of FANUC TPP language

2.1 Design of Robot User Tool Frame

In the industrial robot system, the robot programmer can define a robot user tool frame UT[k](x, y, z) relative to frame Def_TCP for representing the actual tool-tip point of the robot's end-effector. Usually, the UT[k] origin represents the tool-tip point and the z-axis represents the tool axis. A UT[k] plays an important role in robot programming as it not only defines the actual tool-tip point but also addresses its variations. Thus, every end-effector used in a robot application must be defined as a UT[k] and saved in robot system variable UTOOL[k]. Practically, the robot programmer may directly define and select a UT[k] within a robot program or from the robot teach pendant. Table 2 shows the UT[k] frame selection instructions of FANUC TPP language. When the coordinates of a UT[k] is set to zero, it represents frame Def_TCP. The robot system uses the current active UT[k] to record a robot point $P[n]^{R}_{UT[k]}$ as shown in Eq. (5) and cannot move the robot to any robot point $P[m]^{R}_{UT[g]}$ that is taught with a UT[g] different from UT[k] (i.e. $g \neq k$).

$$P[n]^{R}_{UT[k]} = {}^{R}T_{UT[k]} \tag{5}$$

It is obvious that a robot point $P[n]^{R}_{Def\_TCP}$ in Eq. (1) or Eq. (3) can be taught with different UT[k], thus, represented in different Cartesian coordinates in the robot system as shown in Eq. (6)

$$P[n]^{R}_{UT[k]} = P[n]^{R}_{Def\_TCP} \times {}^{Def\_TCP}T_{UT[k]} \ . \tag{6}$$

FANUC TPP Instruction	Description
1. UTOOL_NUM=1	Set UT[1] frame to be the current active UT.

Table 2. UT[k] frame selection instructions of FANUC TPP language

To define a UT[k] for an actual tool-tip point $P_{T\text{-}Ref}$ whose coordinates (x, y, z, w, p, r) in frame D_{ef_TCP} is unknown, the robot programmer must follow the UT Frame Setup procedure provided by the robot system and teach six robot points $P[n]^{R}_{Def\_TCP}$ (for n = 1, 2, … 6) with respect to $P_{T\text{-}Ref}$ and a reference point $P_{S\text{-}Ref}$ on a tool reachable surface. The "three-point" method as shown in Eq. (7) and Eq. (8) utilizes the first three taught robot points in the UT Frame Setup procedure to determine the UT[k] origin. Suppose that the coordinates of vector ${}^{Def\_TCP}p = [p_n, p_o, p_a]^T$ represent point $P_{T\text{-}Ref}$ in frame Def_TCP. Then, it can be determined in Eq. (7)

$${}^{Def\_TCP}p = (T_1)^{-1} \times {}^{R}p, \tag{7}$$

where the coordinates of vector ${}^{R}p = [p_x, p_y, p_z]^T$ represents point $P_{T\text{-}Ref}$ in frame R and T_1 represents the first taught robot point $P[1]^{R}_{Def\_TCP}$ when point $P_{T\text{-}Ref}$ touches point $P_{S\text{-}Ref}$. The coordinates of vector ${}^{R}p = [p_x, p_y, p_z]^T$ also represents point $P_{S\text{-}Ref}$ in frame R and can be solved by the three linear equations in Eq. (8)

$$(I - T_2 T_3^{-1}) \times {}^{R}p = 0 \ , \tag{8}$$

where transformations T$_2$ and T$_3$ represent the other two taught robot points $P[2]^{R}_{Def\_TCP}$ and $P[3]^{R}_{Def\_TCP}$ in the UT Frame Setup procedure respectively when point P$_{T\text{-}Ref}$ is at point P$_{S\text{-}Ref}$. To ensure the UT[k] accuracy, these three robot points must be taught with point P$_{T\text{-}Ref}$ touching point P$_{S\text{-}Ref}$ from three different approach statuses. Practically, $P[2]^{R}_{Def\_TCP}$ (or $P[3]^{R}_{Def\_TCP}$) can be taught by first rotating frame Def_TCP about its x-axis (or y-axis) for at least 90 degrees (or 60 degrees) when the tool is at $P[1]^{R}_{Def\_TCP}$, and then moving point P$_{T\text{-}Ref}$ back to point P$_{S\text{-}Ref}$. A UT[k] taught with the "three-point" method has the same orientation of frame Def_TCP.

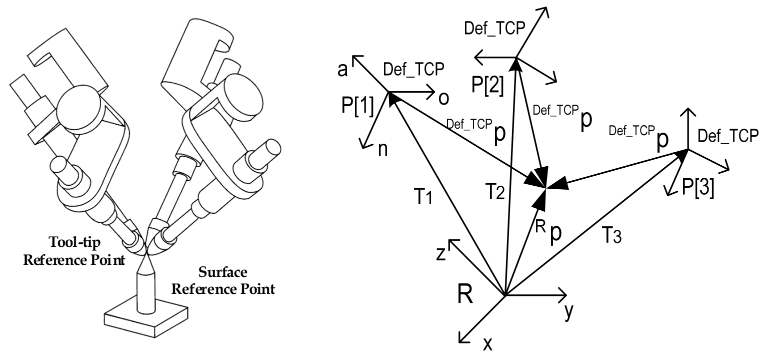


Fig. 1. The three-point method in teaching a UT[k]

If the UT[k] orientation needs to be defined differently from frame Def_TCP, the robot programmer must use the "six-point" method and teach additional three robot points required in UT Frame Setup procedure. These three points define the orient origin point, the positive x-direction, and the positive z-direction of the UT[k], respectively. The method of using such three non-collinear robot points for determining the orientation of a robot frame is to be discussed in section 2.2.

Due to the tool change or damage in robot operations the actual tool-tip point of a robot's end-effector can be varied from its taught UT[k], which causes the inaccuracy of existing robot points relative to the workpiece. To aviod re-teaching all robot points, the robot programmer needs to teach a new UT[k]' for the changed tool-tip point and shift all existing robot points through offset $^{Def\_TCP}T_{Def\_TCP'}$ as shown in Fig. 2. Assume that transformation $^{Def\_TCP}T_{UT[k]}$ represents the position of the original tool-tip point and remains unchanged when frame UT[k] changes into new UT[k]' as shown in Eq. (9)

$$^{Def\_TCP}T_{UT[k]} = {}^{Def\_TCP'}T_{UT[k]'} \ , \tag{9}$$

where frame Def_TCP' represents the position of frame Def_TCP after frame UT[k] moves to UT[k]′. In this case, the pre-taught robot point $P[n]^{R}_{UT[k]}$ can be shifted into the corresponding robot point $P[n]^{R}_{UT[k]'}$ through Eq. (10)

$$^{Def\_TCP}T_{UT[k]'} = {}^{Def\_TCP}T_{Def\_TCP'} \times {}^{Def\_TCP'}T_{UT[k]'} \,. \tag{10}$$

The industrial robot system usually implements Eq. (9) and Eq. (10) as both a system utility function and a program instruction. As a system utility function, the offset $^{Def\_TCP}T_{Def\_TCP'}$ changes the position of frame Def_TCP in the robot system so that the robot programmer is able to change the current UT[k] of a taught P[n] into a different UT[k]′ while remaining the same Cartesian coordinates of P[n] in frame R. As a program instruction, $^{Def\_TCP}T_{Def\_TCP'}$ shifts the pre-taught robot point $P[n]^{R}_{UT[k]}$ into the corresponding point $P[n]'^{R}_{UT[k]'}$ without changing the position of frame Def_TCP. Table 3 shows the UT[k] offset instruction of FANUC TPP language for Eq. (10).

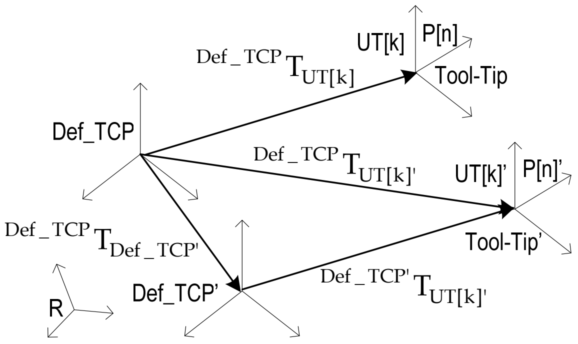


Fig. 2. Shifting a robot point through the offset of frame Def_TCP

TP Instructions	Description
1. Tool_Offset Conditions PR[x], UTOOL[k],	Offset value $^{Def\_TCP}T_{Def\_TCP'}$ is stored in a user-specified position register PR[x].
2. J P[n] 100% Fine Tool_Offset	The "Offset" option in motion instruction shifts the existing robot point $P[n]^{R}_{UT[k]}$ into corresponding point $P[n]'^{R}_{UT[k]'}$.

Table 3. UT[k] offset instruction of FANUC TPP language

2.2 Design of Robot User Frame

In the industrial robot system, the robot programmer is able to establish a robot user frame UF[i](x, y, z) relative to frame R and save it in robot system variable UFRAME[i]. A defined UF[i] can be selected within a robot program or from the robot teach pendant. The robot system uses the current active UF[i] to record robot point $P[n]^{UF[i]}_{UT[k]}$ as shown in Eq. (11) and cannot move the robot to any robot point $P[m]_{UT[k]}^{UF[j]}$ that is taught with a UF[j] different from UF[i] (i.e. $j \neq i$).

$$P[n]_{UT[k]}^{UF[i]} = {}^{UF[i]}T_{UT[k]} \, . \tag{11}$$

It is obvious that a robot point $P[n]_{Def\_TCP}^{R}$ in Eq. (1) or Eq. (3) can be taught with different UT [k] and UF[i], thus, represented in different Cartesian coordinates in the robot system as shown in Eq. (12)

$$P[n]_{UT[k]}^{UF[i]} = ({}^{R}T_{UF[i]})^{-1} \times P[n]_{Def\_TCP}^{R} \times {}^{Def\_TCP}T_{UT[k]} \, . \tag{12}$$

However, the joint representation of a $P[n]_{Def\_TCP}^{R}$ uniquely defines the robot pose.

The robot programmer can directly define a UF[i] with a known robot position measured in frame R. Table 4 shows the UF[i] setup instructions of FANUC TPP language.

FANUC TPP Instructions	Description
1. UFRAME[i]=PR[x]	Assign the value of a robot position register PR[x] to UF[i]
2. UFRAME[i]=LPOS	Assign the current coordinates of frame Def_TCP to UF[i]
3. UFRAME_NUM= i	Set UF[i] to be active in the robot system

Table 4. UF[i] setup instructions of FANUC TPP language

However, to define a UF[i] at a position whose coordinates (x, y, z, w, p, r) in frame R is unknown, the robot programmer needs to follow the UF Setup procedure provided by the robot system and teach four specially defined points $P[n]_{UT[k]}^{R}$ (for n = 1, 2, … 4) where UT[k] represents the tool-tip point of a pointer. In this method as shown in Fig. 3, the location coordinates (x, y, z) of P[4] (i.e. the system-origin point) defines the actual UF[i] origin. The robot system defines the x-, y- and z-axes of frame UF[i] through three mutually perpendicular unit vectors a, b, and c as shown in Eq. (13)

$$\vec{c} = \vec{a} \times \vec{b} \, , \tag{13}$$

where the coordinates of vectors a and b are determined by the location coordinates (x, y, z) of robot points P[1] (i.e. the positive x-direction point), P[2] (i.e. the positive y-direction point), and P[3] (i.e. the system orient-origin point) in R frame as shown in Fig. 3.
With a taught UF[i], the robot programmer is able to teach a group of robot points relative to it and shift the taught points through its offset value. Fig. 4 shows the method for shifting a taught robot point $P[n]_{UT[k]}^{UF[i]}$ with the offset of UF[i].

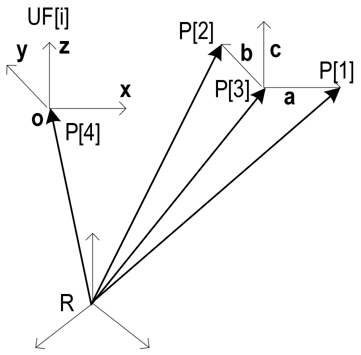


Fig. 3. The four-point method in teaching a UF[i]

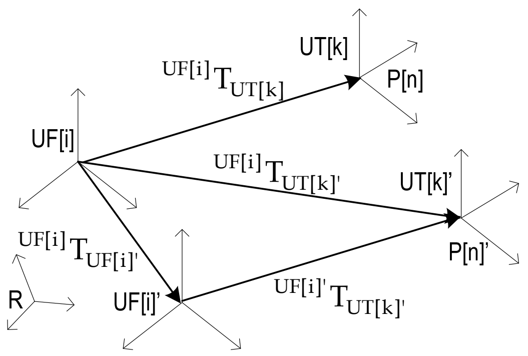


Fig. 4. Shifting a robot point through the offset of UF[i]

Assume that transformation $^{UF[i]}T_{UT[k]}$ represents a taught robot point P[n] and remains unchanged when P[n] shifts to P[n]′ as shown in Eq. (14)

$$^{UF[i]}T_{UT[k]} = {}^{UF[i]'}T_{UT[k]'} \quad \text{or} \quad P[n]^{UF[i]}_{UT[k]} = P[n]'^{UF[i]'}_{UT[k]'}, \tag{14}$$

where frame UF[i]′ represents the position of frame UF[i] after P[n] becomes P[n]′. Also, assume that transformation $^{UF[i]}T_{UF[i]'}$ represents the position change of UF[i]′ relative to UF[i], thus, transformation $^{UF[i]}T_{UT[k]}$ (or robot point $P[n]^{UF[i]}_{UT[k]}$) can be converted (or shifted) to $^{UF[i]}T_{UT[k]'}$ (or $P[n]'^{UF[i]}_{UT[k]'}$) as shown in Eq. (15)

$$^{UF[i]}T_{UT[k]'} = {}^{UF[i]}T_{UF[i]'} \times {}^{UF[i]'}T_{UT[k]'} \quad \text{or} \quad P[n]'^{UF[i]}_{UT[k]'} = {}^{UF[i]}T_{UF[i]'} \times P[n]^{UF[i]}_{UT[k]}. \tag{15}$$

Usually, the industrial robot system implements Eq. (14) and Eq. (15) as both a system utility function and a program instruction. As a system utility function, offset ${}^{UF[i]}T_{UF[i]'}$ changes the current UF[i] of a taught robot point P[n] into a different UF[i]′ without changing its Cartesian coordinates in frame R. As a program instruction, ${}^{UF[i]}T_{UF[i]'}$ shifts a taught robot point $P[n]^{UF[i]}_{UT[k]}$ into the corresponding point $P[n]'^{UF[i]}_{UT[k]'}$ without changing its original UF[i]. Table 5 shows the UF[i] offset instruction of FANUC TPP language for Eq. (15).

FANUC TPP Instructions	Description
3. Offset Conditions PR[x], UFRAME(i),	Offset value ${}^{UF[i]}T_{UF[i]'}$ is stored in a user-specified position register PR[x].
4. J P[n] 100% Fine Offset	The "Offset" option in motion instruction shifts the existing robot point $P[n]^{UF[i]}_{UT[k]}$ into corresponding point $P[n]'^{UF[i]}_{UT[k]'}$.

Table 5. UF[i] offset instruction of FANUC TPP language

A robot point $P[n]^{UF[i]}_{UT[k]}$ can also be shifted by the offset value stored in a robot position register PR[x]. In the industrial robot system, a PR[x] functions to hold the robot position data such as a robot point P[n], the current value of frame Def_TCP (LPOS), or the value of a user-defined robot frame. Different robot languages provide different instructions for manipulating PR[x]. When a PR[x] is taught in a motion instruction, its Cartesian coordinates are defined relative to the current active UT[k] and UF[i] in the robot system. Unlike a taught robot point $P[n]^{UF[i]}_{UT[k]}$ whose UT[k] and UF[i] cannot be changed in a robot program, the UT[k] and UF[i] of a taught PR[x] are always the current active ones in the robot program. This feature allows the robot programmer to use the Cartesian coordinates of a PR[x] as the offset of the current active UF[i] (i.e. ${}^{UF[i]}T_{UF[i]'}$) in the robot program for shifting the robot points as discussed above.

3. Creating Robot Points through Robot Vision System

Within the robot workspace the position of an object frame Obj[n] can be measured relative to a robot UF[i] through sensing systems such as a machine vision system. Methods for integrating vision systems into industrial robot systems have been developed for many years (Connolly, 2008; Nguyen, 2000). The utilized technology includes image processing, system calibration, and reference frame transformations (Golnabi & Asadpour, 2007; Motta et al., 2001). To use the vision measurement in the robot system, the robot programmer must establish a vision frame Vis[i](x, y, z) in the vision system and a robot $UF[i]_{cal}$(x, y, z) in the robot system, and make the two frames exactly coincident. Under this condition, a vision measurement represents a robot point as shown in Eq. (16)

$$ {}^{Vis[i]}T_{Obj[n]} = {}^{UF[i]cal}T_{Obj[n]} = P[n]^{UF[i]cal}_{UT[k]} . \tag{16} $$

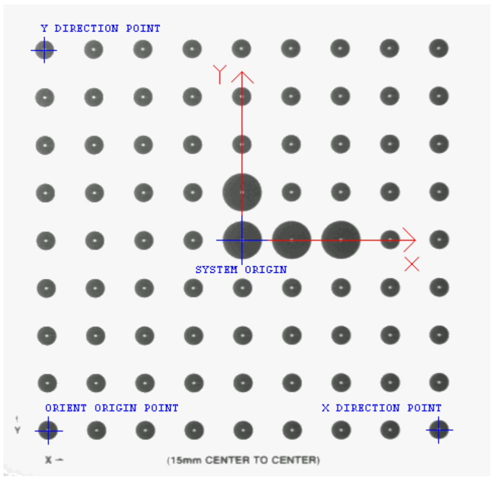
3.1 Vision System Setup

A two-dimensional (2D) robot vision system is able to use the 2D view image taken from a single camera to identify a user-specified object and measure its position coordinates (x, y, roll) for the robot system. The process of vision camera calibration establishes the vision frame Vis[i] (x, y) and the position value (x, y) of a pixel in frame Vis[i]. The robot programmer starts the vision calibration by adjusting both the position and focus of the camera for a completely view of a special grid sheet as shown in Figure 5a**.** The final camera position for the grid view is the "camera-calibration position" $P[n]_{cal}$. During the vision calibration, the vision software uses the images of the large circles to define the x- and y-axes of frame Vis[i] and the small circles to define the pixel value. The process also establishes the camera view plane that is parallel to the grid sheet as shown in Figure 5b. The functions of a geometric locator provided by the vision system allow the robot programmer to define the user-specified searching window, object pattern, and reference frame Obj of the object pattern. After the vision calibration, the vision system is able to identify an object that matches the trained object pattern appeared on the camera view picture and measure position coordinates (x, y, roll) of the object at position Obj[n] as transformation ${}^{Vis[i]}T_{Obj[n]}$.

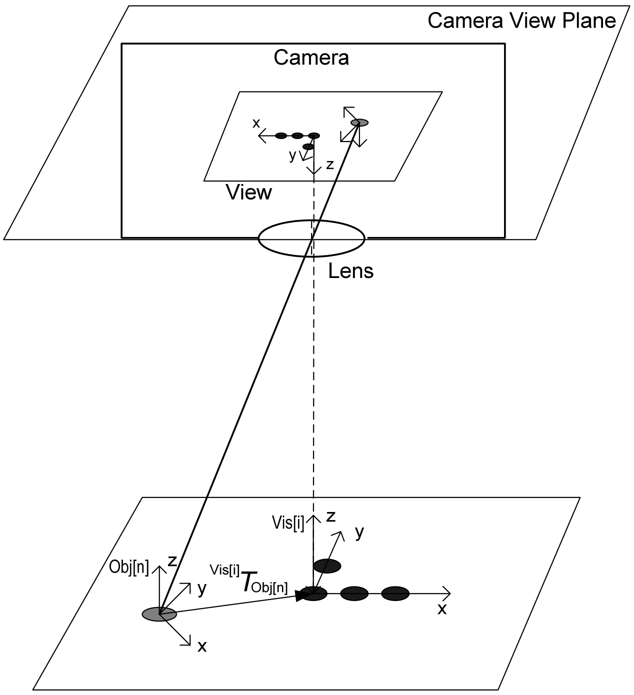
3.2 Integration of Vision "Eye" and Robot "Hand"

To establish a robot user frame $UF[i]_{cal}$ and make it coincident with frame Vis[i], the robot programmer must follow the robot UF Setup procedure and teach four points from the same grid sheet this is at the same position in the vision calibration. The four points are the system origin point, the X and Y direction points, and the orient origin point of the grid sheet as shown in Fig. 5a.

In a "fixed-camera" vision application, the camera must be mounted at the camera-calibration position $P[n]_{cal}$ that is fixed with respect to the robot R frame. Because frame Vis[i] is coincident with frame $UF[i]_{cal}$ when the camera is at $P[n]_{cal}$, the vision measurement ${}^{Vis}T_{Obj[n]}$=(x, y, roll) to a vision-identified object at position Obj[n] actually represents the same coordinates of the object in $UF[i]_{cal}$ as shown in Eq. (16). With additional values of z, pitch, and yaw that can be either specified by the robot programmer or measured by a laser sensor in a 3D vision system, ${}^{Vis[i]}T_{Obj[n]}$ can be used as a robot point $P[n]^{UF[i]_{Cal}}_{UT[k]}$ in the robot program. However, after reaching to vision-defined point $P[n]^{UF[i]_{Cal}}_{UT[k]}$,the robot system cannot perform the robot motions with the robot points that are taught via the same vision-identified object located at a different position Obj[m] (i.e. $m \neq n$).



(a) Camera calibration grid sheet



(b) Vision measurement

Fig. 5. Vision system setup

To reuse all pre-taught robot points in the robot program for the vision-identified object at a different position, the robot programmer must set up the vision system so that it can determine the position offset of frame $UF[i]_{cal}$ (i.e. ${}^{UF[i]cal}T_{UF[i]'cal}$) with two vision measurements ${}^{Vis}T_{Obj[n]}$ and ${}^{Vis}T_{Obj[m]}$ as shown in Fig. 6 and Eq. (17)

$$
{}^{UF[i]cal}T_{UF[i]'cal} = {}^{Vis[i]}T_{Obj[m]} \times ({}^{Vis[i]}T_{Obj[n]})^{-1},
$$

and (17)

$$
{}^{UF[i]cal}T_{Obj[n]} = {}^{Vis[i]}T_{Obj[n]} = {}^{Vis[i]'}T_{OBJ[m]} = {}^{UF[i]'cal}T_{Obj[m]},
$$

where frames Vis[i]′ and $UF[i]'_{cal}$ represent the positions of frames Vis[i] and $UF[i]_{cal}$ after object position Obj[n] changes to Obj[m]. Usually, the vision system obtains ${}^{Vis[i]}T_{Obj[n]}$ during the vision setup and acquires ${}^{Vis[i]}T_{Obj[m]}$ when the camera takes the actual view picture for the object.

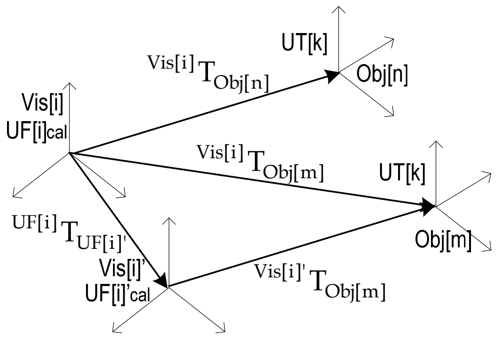


Fig. 6. Determining the offset of frame $UF[i]_{cal}$ through two vision measurements

In a "mobile-camera" vision application, the camera can be attached to the robot's wrist faceplate and moved by the robot on the camera view plane. In this case, frames $UF[i]_{cal}$ and Vis[i] are not coincident each other when camera view position $P[m]_{vie}$ is not at $P[n]_{cal}$. Thus, vision measurement ${}^{Vis[i]}T_{Obj[m]}$ obtained at $P[m]_{vie}$ cannot be used for determining ${}^{UF[i]cal}T_{UF[i]'cal}$ in Eq. (17) directly. However, it is noticed that frame Vis[i] is fixed in frame Def_TCP and its position coordinates can be determined in Eq. (18) as shown in Fig. 7

$$
{}^{Def\_TCP}T_{Vis[i]} = ({}^{R}T_{Def\_TCP})^{-1} \times {}^{R}T_{UF[i]_{cal}}, \tag{18}
$$

where transformations ${}^{R}T_{UF[i]cal}$ and ${}^{R}T_{Def\_TCP}$ are uploaded from the robot system when the robot-mounted camera is at $P[n]_{cal}$ during the vision setup. With vision-determined ${}^{Def\_TCP}T_{Vis[i]}$, vision measurement ${}^{Vis[i]}T_{Obj[m]}$ can be transformed into ${}^{UF[i]}T_{Obj[m]}$ for the robot system in Eq. (19) if frame Def_TCP is used as frame $UF[i]_{cal}$ (i.e. $UF[i]_{cal}$ = Def_TCP) in the robot program as shown in Fig. 7.

$$
{}^{UF[i]cal}T_{Obj[m]} = {}^{Def\_TCP}T_{Obj[m]} = {}^{Def\_TCP}T_{Vis[i]} \times {}^{Vis[i]}T_{Obj[m]}\ . \tag{19}
$$

By substituting Eq. (19) into Eq. (17), frame offset ${}^{UF[i]cal}T_{UF[i]'cal}$ can be determined in Eq. (20)

$$
\begin{aligned}
&{}^{UF[i]cal}T_{UF[i]'cal} = {}^{Def\_TCP}T_{Vis[i]} \times {}^{Vis[i]}T_{Obj[m]} \times ({}^{Vis[i]}T_{Obj[n]})^{-1} \\
\text{and} \qquad & \qquad\qquad\qquad (20) \\
&{}^{UF[i]cal}T_{Obj[n]} = {}^{Vis[i]}T_{Obj[n]} = {}^{Vis[i]'}T_{Obj[m]} = {}^{UF[i]'cal}T_{Obj[m]},
\end{aligned}
$$

where frames Vis[i]′ and UF[i]′$_{cal}$ represent positions of frames Vis[i] and UF[i]$_{cal}$ after object position Obj[n] changes to Obj[m].

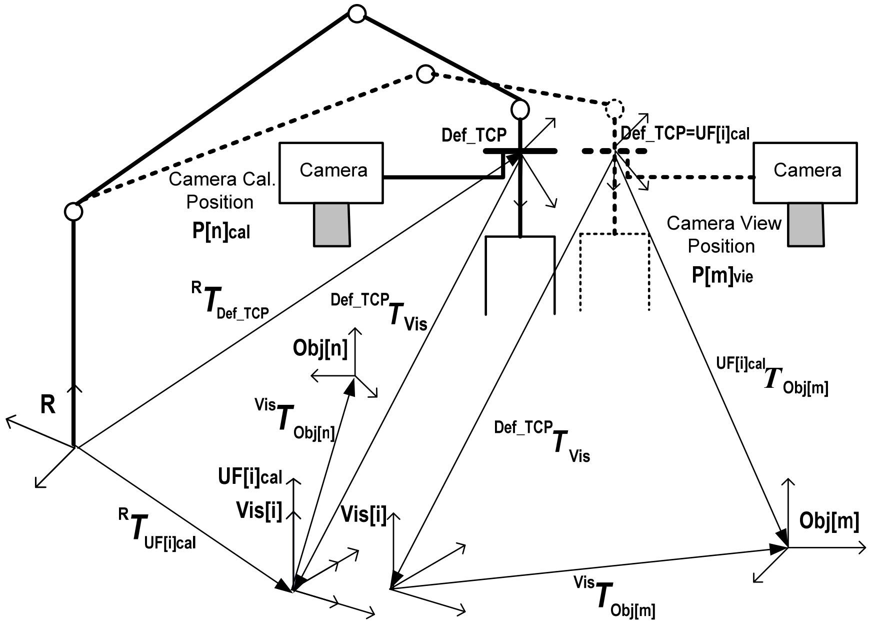


Fig. 7. Frame transformations in mobile-camera application

With vision-determined ${}^{UF[i]cal}T_{UF[i]'cal}$ in Eq. (17) (for fixed-camera) or Eq. (20) (for mobilecamera), the robot programmer is able to apply Eq. (15) for shifting all pre-taught robot points $P[n]^{UF[i]cal}_{UT[k]}$ into $P[n]'^{UF[i]cal}_{UT[k]}$ for the vision-identified object at position Obj[m] as shown in Eq. (21)

$$
\begin{aligned}
&P[n]'^{UF[i]cal}_{UT[k]} = {}^{UF[i]cal}T_{UF[i]'cal} \times P[n]^{UF[i]'cal}_{UT[k]} \\
\text{and} \qquad & \qquad\qquad\qquad (21) \\
&P[n]'^{UF[i]'cal}_{UT[k]} = P[n]^{UF[i]cal}_{UT[k]}.
\end{aligned}
$$

Table 6 shows the FANUC TP program used in a fixed-camera FANUC vision application. The program calculates "vision offset" ${}^{UF[i]cal}T_{UF[i]'cal}$ in Eq. (17), sends it to user-specified robot position register PR[x], and transforms robot point $P[n]^{UF[i]cal}_{UT[k]}$ in Eq. (21).

FANUC TP Program	Description
1: R[1] = 0;	Clear robot register R[1] which is used as the indicator for vision "Snap & Find" operation.
2: VisLOC Snap & Find ('2d Single', 2);	Acquire ${}^{Vis}T_{OBJ[m]}$ from snapshot view picture '2d single', find vision-measured offset ${}^{UF[i]cal}T_{UF[i]'cal}$, and send it to robot position register PR[1].
3: WAIT R[1] <> 0;	Wait until the VisLOC vision system sets R[1] to '1' for a successful vision "Snap & Find" operation.
4: IF R[1] <> 1, JMP LBL[99]	Jump out of the program if the vision system cannot set R[1] as '1'.
5: OFFSET CONDITION PR[1], UFRAME[i]$_{cal}$;	Apply ${}^{UF[i]cal}T_{UF[i]'cal}$ as Offset Condition.
6: J P[n] 50% FINE OFFSET;	Transforms robot point $P[n]^{UF[i]cal}_{UT[k]}$ by ${}^{UF[i]cal}T_{UF[i]'cal}$.

Table 6. FANUC TP program used in a fixed-camera FANUC vision application

4. Creating Robot Points through Robot Simulation System

With the today's robot simulation technology a robot programmer may also utilize the robot simulation software to program the motions and actions of a real robot offline in a virtual and interactive 3D design environment. Among many robot simulation software packages, the DELMIA Interactive Graphics Robot Instruction Program (IGRIP) provides the robot programmers with the most comprehensive and generic simulation functions, industrial robot models, CAD data translators, and robot program translators (Cheng, 2003; Connolly, 2006) .

In IGRIP, a simulation design starts with building the 3D device models (or Device) based on the geometry, joints, kinematics of the corresponding real devices such as a robot and its peripheral equipment. The base frame B[i](x, y, z) of a retrieved Device defines its position in the simulation workcell (or Workcell). With all required Devices in the Workcell, the robot programmer is able to create virtual robot points called tag points and program the desired motions and actions of the robot Device and end-effector Device in robot simulation language. Executing the Device simulation programs allows the robot programmer to verify the performance of the robot Device in the Workcell. After the tag points are adjusted relative to the position of the corresponding robot in the real robot workcell through conducting the simulation calibration, the simulation robot program can be downloaded to the real robot controller for execution. Comparing to the conventional online robot programming, the true robot offline programming provides several advantages in terms of the improved robot workcell performance and reduced robot downtime.

4.1 Creation of Virtual Robot Points

A tag point Tag[n] is created as a Cartesian frame and attached to the base frame B[i] of a user-selected Device in the Workcell. Mathematically, the Tag[n] position is measured in frame B[i] as frame transformation $^{B[i]}T_{Tag[n]}$ and can be manipulated through functions of Selection, Translation, Rotation, and Snap. During robot simulation, the motion instruction in the robot simulation program is able to move frame Def_TCP (or UT[k]) of the robot Device to coincide a Tag[n] only if it is within the robot's workspace and not a robot's singularity. The procedures for creating and manipulating tag points in IGRIP are:

Step 1. Create a tag path and attach it to frame B[i] of a selected Device.

Step 2. Create tag points Tag[n] (for n = 1, 2, … m) one at a time in the created path.

Step 3. Manipulate a Tag[n] in the Workcell. Besides manipulation functions of selection, translation, and/or rotation, the "snap" function allows the programmer to place a Tag[n] to the vertex, edge, frame, curve, and surface of any Device in the Workcell. Constraints and options can also be set up for a specific snap function. For example, if the "center" option is chosen, a Tag[n] will be snapped on the "center" of the geometric entities such as line, edge, polygon, etc. If a Tag[n] is required to snap on "surface," the parameter "approach axis" must be set up to determine which axis of Tag[n] will be aligned with the surface normal vector.

4.2 Accuracy Enhancement of Virtual Robot Points

It is obvious that inevitable differences exist between the real robot wokcell and the simulated robot Workcell because of the manufacturing tolerance and dimension variation of the corresponding components. Therefore, it is not feasible to directly download tag point Tag[n] to the actual robot controller for execution. Instead, the robot programmer must apply the simulation calibration functions to adjust the tag points with respect to a number of robot points uploaded from the real robot workcell. The two commonly used calibration methods are calibrating frame UT[k] of a robot Device and calibrating frame B[i] of a Device that attaches Tag[n]. The underlying principles of these methods are the same with the design of robot UT and UF frames as introduced in section 2.1 and 2.2. For example, assume that the UT[k]' of the robot end-effector Device is not exactly the same with the UT[k] of the actual robot end-effector prior to UT[k] calibration. To determine and use the actual UT[k] in the simulation Workcell, the programmer needs to teach three non-collinear robot points through UT Frame Setup procedure in the real robot system and upload them into the simulation Workcell so that the simulation system is able to calculate the origin of UT[k] with the "three-point" method as described in Eq. (5) and Eq. (6) in section 2.1. With the calibrated UT[k] and the assumption that the robot Device is exactly the same as the real robot, the UT[k] position relative to the R frame ($^{R}T_{UT[k]}$) of a robot Device in the simulation Workcell is exactly the same as the corresponding one in the real robot workcell. Also, prior to frame B[i] calibration, the Tag[n] position relative to frame R of a robot Device ($^{R}T_{Tag[n]}$) may not be the same as the corresponding one in the real robot workcell. In this case, the Device that attaches Tag[n] serves as a "fixture" Device. Thus, the programmer may define a robot UF[i] frame by teaching (or create) three or six robot points (or tag points) on the features of the real "fixture" device (or "fixture" Device) in the real workcell (or the simulation Workcell). Coinciding the created UF tag points in the simulation Workcell with the corresponding uploaded real robot points results in calibrating the position of frame B[i] of the "fixture" Device and the Tag[n] attached to it.

5. Transferring Robot Points to Identical Robots

In industrial robot applications, there are often the cases in which the robot programmer must be able to quickly and reliably change the existing robot points in the robot program so that they can be accurate to the slight changes of components in the existing or identical robot workcell. Different methods have been developed for measuring the dimensional difference of the similar components in the robot workcell and using it to convert the robot points in the existing robot programs. For example, as introduced in section 2.1 and 2.2, the robot programmer can measure the positional variations of two similar tool-tip points and workpieces in the real robot workcell through the offsets of UT[k] and UF[i], and compensate the pre-taught robot points with either the robot system utility function or the robot program instruction. However, if the dimensional difference exists between two identical robots, an external calibration system must be used for identifying the robots' difference so that the taught robot points P[n] for one robot system can be transferred to the identical one. The process is called the robot calibration, which consists of four steps (Cheng, 2007; Motta et al, 2001). The first step is to teach specially defined robot points P[n]. The second step is to "physically" measure the taught P[n] with an appropriate external measurement device such as laser interferometry, stereo vision, or mechanical "string pull" devices, etc. The third step is to calculate the relevant actual parameters of the robot frames through a specific mathematical solution.
The Dynalog DynaCal system is a complete robot calibration system that is able to identify the parameters of robot joint frames, UT[k], and UF[i] in two "identical" robot workcells, and compensate the existing robot points so that they can be download to the identical robot system for execution. Among its hardware components, the DynaCal measurement device defines its own measurement frame through a precise base adaptor mounted at an alignment point. It uses a high resolution, low inertia optical encoder to constantly measure the extension of the cable that is connected to the tool-tip point of the robot's end-effector through a DynaCal TCP adaptor, and sends the encoder measurements to the Window-based DynaCal software for the identification of the robot parameters.
Prior to the robot calibration, the robot programmer needs to conduct the calibration experiment in which a developed robot calibration program moves the robot Def_TCP frame to a set of taught robot calibration points. Depending on the required accuracy, at least 30 calibration points are required. It is also important to select robot calibration points that are able to move each robot joint as much as possible in order to "excite" its calibration parameters. The dimensional difference of the robot joint parameters is then determined through a specific mathematical solution such as the standard non-linear least squares optimization. Theoretically, the existing robot kinematics model can be modified with the identified robot parameters. However, due to the difficulties in directly modifying the kinematic parameters of an actual robot controller, the external calibration system compensates the corresponding joint values of all robot points in the existing robot program by solving the robot's inverse kinematics equations with the identified robot joint parameters.

In DynaCal UT[k] calibration, the programmer needs to specify at least three non-collinear measurement points on the robot end-effector and input their locations relative to the desired tool-tip point in the DynaCal system during the DynaCal robot calibration. However, when only the UT[k] origin needs to be calibrated, one measurement point on the end-effector suffices and choosing the measurement point at the desired tool-tip point further simplifies the process because its location relative to the desired tool-tip point is then simply zero. In DynaCal UF[i] calibration, the programmer needs to mount the DynaCal measurement device at three (or four) non-collinear alignment points on a fixture during the DynaCal robot calibration. The position of each alignment point relative to the robot R frame is measured through the DynaCal cable and the TCP adaptor at the calibrated UT[k]. The DynaCal software uses the measurements to determine the transformation between the UF[i]$_{Fix}$ on the fixture and the robot R frame, denoted as ${}^{R}T_{UF[i]Fix}$. With the identified values of frames UT[k] and UF[i]$_{Fix}$ in the original robot workcell and the values of UT[k]′ and UF[i]′$_{Fix}$ in the "identical" robot workcell, offsets UF and UT can be determined and the robot points P[n] used in the original robot cell can be converted into the corresponding ones for the "identical" robot cell with the methods as introduced in sections 2.1 and 2.2.

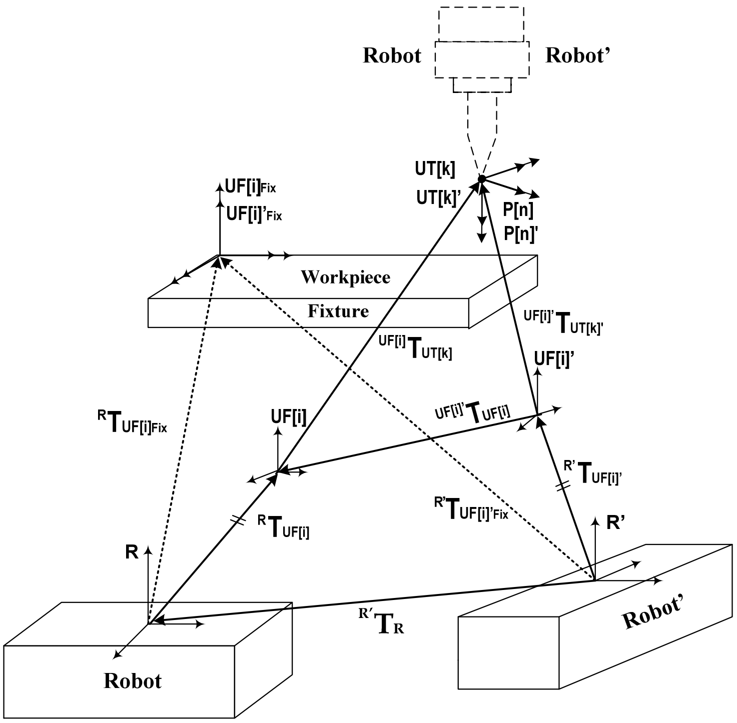


Fig. 8. Determining the offset of UF[i] in two identical robot workcells through robot calibration system

The following frame transformation equations show the method for determining the robot offset ${}^{UF[i]'}T_{UF[i]}$ in two identical robot workcells through calibrated values of UF[i]$_{Fix}$ and

$UF[i]'_{Fix}$ as shown in Fig. 8. Given that the coincidence of $UF[i]_{Fix}$ and $UF[i]'_{Fix}$ represents a commonly used calibration fixture in two "identical" robot workcells, the transformation between two robot base frames R′ and R can be calculated in Eq. (22)

$$^{R'}T_{R} = {}^{R'}T_{UF[i]'_{Fix}} \times ({}^{R}T_{UF[i]_{Fix}})^{-1}. \tag{22}$$

It is also possible to make transformation $^{R'}T_{UF[i]'}$ equal to transformation $^{R}T_{UF[i]}$ as shown in Eq. (23)

$$^{R'}T_{UF[i]'} = {}^{R}T_{UF[i]}, \tag{23}$$

where frames UF[i] and UF[i]′ are used for recording robot points P[n] and P[n]′ in the two "identical" robot workcells, respectively. With Eq. (22) and Eq. (23), robot offset $^{UF[i]'}T_{UF[i]}$ can be calculated in Eq. (24)

$$^{UF[i]'}T_{UF[i]} = ({}^{R'}T_{UF[i]'})^{-1} \times {}^{R'}T_{R} \times {}^{R}T_{UF[i]}. \tag{24}$$

6. Conclusion

Creating accurate robot points is an important task in robot programming. This chapter discussed the advanced techniques used in creating robot points for improving robot operation flexibility and reducing robot production downtime. The theory of robotics shows that an industrial robot system represents a robot point in both Cartesian coordinates and proper joint values. The concepts and procedures of designing accurate robot user tool frame UT[k] and robot user frame UF[i] are essential in teaching robot points. Depending on the selected UT[k] and UF[i], the Cartesian coordinates of a robot point may be different, but the joint values of a robot point always uniquely define the robot pose. Through teaching robot frames UT[k] and UF[i] and measuring their offsets, the robot programmer is able to shift the originally taught robot points for dealing with the position variations of the robot's end-effector and the workpiece. The similar method has also been successfully applied in the robot vision system, the robot simulation, and the robot calibration system. In an integrated robot vision system, the vision frame Vis[i] serves the role of frame UF[i]. The vision measurements to the vision-identified object obtained in either fixed-camera or mobile-camera applications are used for determining the offset of UF[i] for the robot system. In robot simulation, the virtual robot points created in the simulation robot workcell must be adjusted relative to the position of the robot in the real robot workcell. This task can be done by attaching the created virtual robot points to the base frame B[i] of the simulation device that serves the same role of UF[i]. With the uploaded real robot points, the virtual robot points can be adjusted with respect to the determined true frame B[i]. In a robot calibration system, the measuring device establishes frame UF[i] on a common fixture for the workpiece, and the measurement of UF[i] in the identical robot workcell are used to determine the offset of UF[i].

7. References

Cheng, F. S. (2009). Programming Vision-Guided Industrial Robot Operations, *Journal of Engineering Technology*, Vol. 26, No. 1, Spring 2009, pp. 10-15.

Cheng, F. S. (2007). The Method of Recovering TCP Positions in Industrial Robot Production Programs, *Proceedings of 2007 IEEE International Conference on Mechatronics and Automation*, August 2007, pp. 805-810.

Cheng, S. F. (2003). The Simulation Approach for Designing Robotic Workcells, *Journal of Engineering Technology*, Vol. 20, No. 2, Fall 2003, pp. 42-48.

Connolly, C. (2008). Artificial Intelligence and Robotic Hand-Eye Coordination, *Industrial Robot:An International Journal,* Vol. 35, No. 6, 2008, pp. 496-503.

Connolly, C. (2007). A New Integrated Robot Vision System from FANUC Robotics, *Industrial Robot: An International Journal,* Vol. 34, No. 2, 2007, pp. 103-106.

Connolly, C. (2006). Delmia Robot Modeling Software Aids Nuclear and Other Industries, *Industrial Robot:An International Journal,* Vol. 33, No. 4, 2008, pp. 259-264.

Fanuc Robotics (2007). Teaching Pendant Programming, *R-30iA Mate LR HandlingTool Software Documentation*, Fanuc Robotics America, Inc.

Golnabi, H. & Asadpour, A. (2007). Design and application of industrial machine vision systems, *Robotics and Computer-Integrated Manufacturing,* 23, pp. 630–637.

Motta, J.T.; de Carvalhob, G. C. & McMaster, R.S. (2001). Robot calibration using a 3D vision-based measurement system with a single camera, *Robotics and Computer Integrated Manufacturing,* 17, 2001, pp. 487–497

Nguyen, M. C. (2000), Vision-Based Intelligent Robots, In SPIE: Input/Output and Imaging Technologies II, Vol. 4080, 2000, pp. 41-47.

Niku, S. B. (2001). Robot Kinematics, *Introduction to Robotics: Analysis, Systems, Applications*, pp. 29-67, Prentice Hall. ISBN 0130613096, New Jersey, USA.

Pulkkinen1, T.; Heikkilä1, T.; Sallinen1, M.; Kivikunnas1, S. & Salmi, T. (2008). 2D CAD based robot programming for processing metal profiles in short series, *Proceedings of Manufacturing, International Conference on Control, Automation and Systems 2008*, Oct. 14-17, 2008 in COEX, Seoul, Korea, pp. 157-160.

Rehg, J. A. (2003). Path Control, *Introduction to Robotics in CIM Systems*, 5th Ed. pp. 102-108Prentice Hall, ISBN 0130602434, New Jersey, USA.

Zhang, H.; Chen, H. & Xi, N. (2006). Automated robot programming based on sensor fusion, *Industrial Robot: An International Journal,* Vol. 33, No. 6, 2006, pp. 451-459.

An Open-architecture Robot Controller applied to Interaction Tasks

A. Oliveira[1], E. De Pieri[2] and U. Moreno[2]
[1]*Mechanical Engineering Department*
University of Caxias do Sul (UCS)
[2]*Department of Automation and Systems*
Federal University of Santa Catarina (UFSC)
[1-2]*Brazil*

1. Introduction

Many current robotic applications are limited by the industry state of art of the manipulators control algorithms. The inclusion of force and vision feedbacks, the possibility of cooperation between two or more manipulators, the control of robots with irregular topology will certainly enlarge the industrial robotics applications. The development of control algorithms to this end brings the necessity of using open-architecture controllers.
Generally the robotic controllers are developed for position control, without accomplishing integrally the requirements of tasks in which interactions with the environment occur. Therefore, this is currently one of the main research areas in robotics, e.g., in (Abele et al., 2007) is presented the identification of characteristics to an industrial robot to execute machining applications. To consider this interaction the robot controller has to give priority to the force control time response, because in the instant of end-effectors contact with the surface, several forces act on the system. Depending on the speeds and the accelerations involved in the process, damages or errors can occur. To avoid these effects, compliances are inserted in tool or in surface of operation.
A new reference model for a control system functional architecture applied to open-architecture robot controllers is presented. Where, this model is applied for integrally development of a five-layer based open-architecture robotic controller for interaction tasks, which uses parallel and distributed processing techniques, avoiding the necessity of compliance in system, allowing a real-time processing of the application and the total control of information. This architecture provides flexibility, the knowledge of all the control structures and allows the user to modify all controller layers. The used controller conception aims to fulfill with the following requirements: high capacity of processing, low cost, connectivity with other systems, availability for the remote access, easiness of maintenance, flexibility in the implementation, integration with a personal computer and programming in high level.

This chapter is organized as follows. Section 2 overviews the most relevant categories, definitions and requirements of robot controllers. Section 3 details the reference model for open-architecture controller development. Section 4 describes the robot retrofitting for interactions tasks. Section 5 presents and discusses the experimental setup. Finally, Section 6 concludes the chapter and outlines future research and development directions.

2. Open-architecture Robot Controllers

Various open control architectures for industrial robots have already been developed by robot and control manufacturers as well as in research labs. In (Lippiello et al., 2007) is presented an open architecture for sensory feedback control of a dual-arm industrial robotic cell for cooperation tasks. In (Macchelli & Melchiorri, 2007) is presented a real-time control system based on RTAI-Linux operating system and developed for coupling of an advanced end-effector. (Hong et al., 2001) develop a system of robot open control based on a reference model OSACA. (Bona et al., 2001) propose a real-time architecture for robot control system development based in real-time operating system for embedded systems, RTOS. In (Donald & Dunlop, 2001) present a retrofitting of a path control system for a hydraulic robot based on a FPGA executing the embedded operating system RTSS. The inexistence of a standard methodology for architecture controller project difficult the development of high-openness degree control system.

Most of the existing robot control open architectures are based on a standard PC hardware and a standard operating system, because I/O boards and communication boards for robots have a higher cost in relation to the similar boards for PCs. Another reason is the lack of standardization of robot peripherals, with each manufacturer developing its own protocols and interfaces, forcing the users to buy all the components of a single manufacturer (Lages et al., 2003). Additionally, a PC based controller can be integrated more easily with many commercially available add-on peripherals such as mass storage devices, Ethernet card and other I/O devices. So, the facility to integrate other functionalities is a strong reason to use a standard PC hardware in robot control open architectures.

Another reason is that the robot programming languages are, at low level, more similar to the Assembly languages than to the modern high level languages and this may difficult implementations (Lages et al., 2003). In a PC based controller standard software development tools (e.g., Visual C++, Visual Basic or Delphi) can be used.

2.1 Definitions

The definition of open system, according to Technical Committee of Open Systems of IEEE is “An open system provides capabilities that enable properly implemented applications to run on a variety of platforms from multiple vendors, interoperate with other system applications and present a consistent styler of interaction with the user”. A open-architecture control system has the capacity to operate with the best components of different manufacturers. What makes possible the easy integration of new system functionalities.

From user point of view, the “openness” of the systems consists in capabilities to integrate, extend and reuse software modules in control systems (Lutz & Sperling, 1997). In (Pritschow & Altintas, 2001) and (Nacsa, 2001) the “degree of openness” of a system is defined by some criteria, as:

- ***Extendibility***: A variable number of modules can be executed simultaneously in a same platform, without causing conflicts, i.e., this characteristic depends mainly on the operating system, that should accomplish a multi-task processing, and also of modules coupling level, that should allow those operations.
- ***Interoperability***: The modules work together efficiently and they can interchange data in a defined way through logical and physical communication buses.
- ***Portability***: The modules can be executed in different platforms without modifications, maintaining their functionalities, i.e., they should conform software and hardware standards to maintain the system compatibility with other platforms.
- ***Scalability***: Depending on the user requirements, the module functionalities and performance and size of the hardware, software and firmware can adapt for the system optimization.

Those characteristics define the "degree of openness" of a system, how more extended and refined, major will be the level of openness. For open-architecture controllers, one more characteristic should be considered, the modularity.

- ***Modularity***: The system is divided in specialized subsystems, called modules, that can be substituted without significant modifications in system. This characteristic consists of logical and physical system decomposition in small functional units.

2.2 Categories

The controllers are characterized by the freedom of access information or simply for "degree of openness". Usually, the control of several system modules (e.g., unit power and low level control) is proprietary and cannot be modified by user, other levels are considered open (e.g., communication interface and high-level control), i.e., they are based on hardware and software standards with specifications of open interface.

In (Pritschow & Altintas, 2001), (Lutz & Sperling, 1997) and (Ford, 1994), the "degree of openness" of a system is defined in agreement with access concept to controller layers, like this, they can be classified in three categories:

- ***Proprietary***: That system modality allows the access just to application layer, being therefore, a closed system. In those systems is extremely difficult or impossible the integration of external modules.
- ***Hybrid or Restricted***: That category makes available the access to application layer and a controlled access to operating system module. The operating system has a fixed topology, however, allows small changes in control system modules (e.g., gains and parameters).
- ***Open***: Open-architecture systems allow integral access of application layers and operating system modules, supplying a homogeneous vision of the system, allowing the manipulation and modification of all modules that compose the system. Its offers interchangeability, scalability, portability and interoperability.

2.3 Requirements

One of main requirements for a system to be characterized with open-architecture is the necessity of the control functionalities be subdivided in small functional units with a solid relationship among the subsystems. Consequently, the modularity becomes fundamental for a control system to have an open-architecture (Pritschow & Altintas, 2001).

The determination of module complexity should consider factors as the "degree of openness" wanted and integration cost. Small modules supply a high-level openness, but they increase the complexity and integration costs. A low modularity can drive for a high demand of resources and to deteriorate the system performance, not allowing real-time data articulation (Nacsa, 2001).
The system structuring through a modular interaction requests a detailed group of relationship methods, composed by Application Programming Interfaces (i.e., these are a group of routines and software standards for extern access of their functionalities). In open control systems these interfaces need to be standardized (Pritschow & Altintas, 2001).
The modular platforms encapsulate the operation system specific methods absorbing the hardware, operating system and communication characteristics. What promotes a high level data exchange, this abstraction requests a data mediation module, called middleware. These data concatenation and adaptation points increase the portability and interoperability of distributed applications in heterogeneous environments.

3. The Reference Model for Open-Architecture Robot Controllers

The reference model for a control system functional architecture presented in (Sciavicco & Siciliano, 2000) has a priority focus in the control structure, little exploring the other levels of robot controllers.
This work proposes a new reference model for a control system functional architecture applied to open-architecture robot controller. The model is based on model of (Sciavicco & Siciliano, 2000), however, it expands the approach for all controller levels, adapts their layers in agreement with the standard ISO 7498-1 and considers the definition, categories, requirements and tendencies for open-architecture controllers. The structure of the proposed reference model is represented in Fig. 1, where the five hierarchical levels are illustrated. To proceed, those layers will be described individually.

3.1 Task Layer
The task layer is responsible for industrial robot control tasks grouped in three categories: trajectory planning, supervisory system and control law. Those operations are processed in the central equipment of the system, usually a personal computer (PC). In remote control operations, the operations can be divided in two software modules with relationship client-server. The trajectory planning and supervisory system will be processed with smaller time requirements in client, while the control structure will be processed in real time of application in server.

3.2 Integration Layer
The adopted functional architecture hierarchical structure, together with its articulation into different modules, suggests a hardware implementation that exploits distributed computational resources interconnected by means of suitable communication channels. At the integration layer, the information adaptation is accomplished (i.e., concatenation and organization) incoming from several processors that compose the distributed system.

These operations supply to superior layer a heterogeneity vision of the system to sharing resources. In this level, peripherals with high-level of abstraction (e.g., exteroceptive sensors) are also appropriate in this level.

3.3 Communication Layer

At the communication layer, the interconnection of information among the system processors is accomplished, usually using high-speed data transmission buses. The network topology is indifferent, however, it is important the use of redundant ways for connection among all intermediate points and the net central knot through the main bus and the embedded systems interconnection by an alternative communication bus. Every system interconnection accomplished in this layer is based in International Standard ISO/IEC 7498-1.

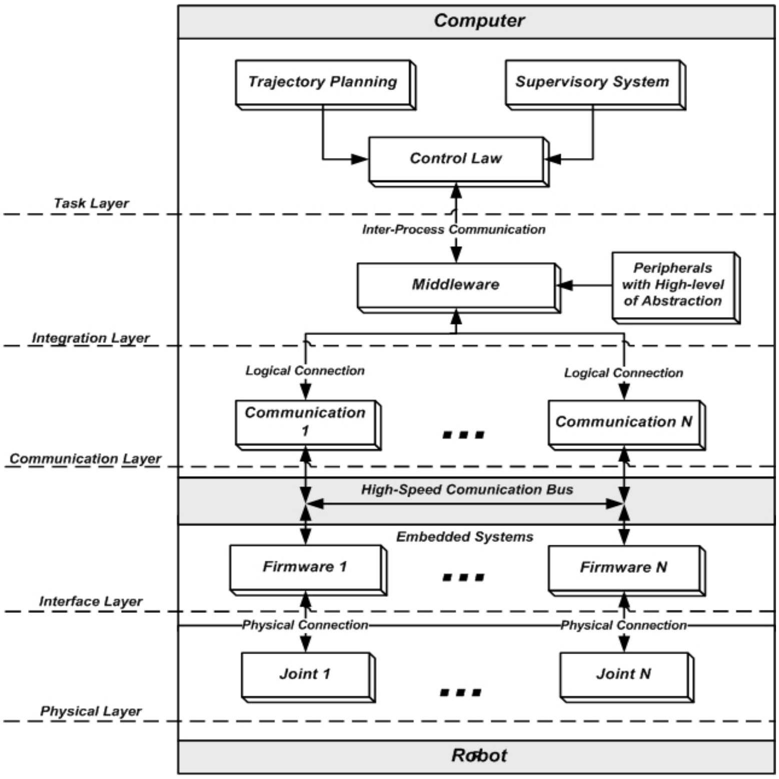


Fig. 1. Functional architecture of proposed reference model.

3.4 Interface Layer

The interface layer is composed by the embedded systems, i.e., dedicated hardware's to process specific task software (called firmware) encapsulated in internal storage memories. This organizational structure divides the system in small hardware modules and consequently, distributes the system processing. The processing distribution degree is proportional at the utilization level of dedicated processors in system. The system decomposition in task dedicated processors guarantees a fixed and minimum response time.

3.5 Physical Layer

The industrial manipulator physical access (i.e., actuators and proprioceptive sensors) occur in physical layer, composed only by the input and output robot data channels. Usually, the actuator activation is realized indirectly, because, the controller signs only access the unit power that adapts this signs for the motors.

4. Reference Model applied to Interaction Tasks

Special requirement for robot controllers that includes force control

Generally the robotic controllers are developed for applications that require only position control, and the robot end effector doesn't contact the workspace during its movement. The interaction concept is related with the contact between robot and environment, where generated force and torque profiles need to be controlled. In applications that need force control, the end effector contacts some surface in its workspace and this interaction generates contact forces that must be controlled in a way to fulfill the task correctly, without damaging both, robot tools and the working objects.

The contact force intensities, originated by tool movements commanded by the robot controller, depend on both, the tool rigidity and the object surface rigidity, and they must be also controlled. A small tool movement could originate large force intensities in case the tool and the object surface rigidity are large. It should be noted that by introducing compliance to the tool we generate a delay in the application of the forces and this could be unacceptable in some applications. Consequently, the system should have a small time response to these forces, to prevent tool, robot or object damages. The use of high performance systems is a requisite of controllers for application of force control.

Therefore, the reference model proposed was applied, considering the interaction tasks requirements, for retrofit of old industrial manipulator. The resultant functional structure for controller is presented in Fig. 2 and described as follows.

4.1 Task Layer for interaction tasks

The task layer has a mathematical environment prepared to make operations with matrices in which the control law is stored. The information proceeding from the n joints are available in matrices *nx1* corresponding to the position vector q, and the velocity vector $\dot{q}$, where the lines represent the joints. The force sensor data are stored in a matrix *6 x 1* called $h = [f_x\ f_y\ f_z\ \mu_x\ \mu_y\ \mu_z]^T$, which contains forces and moments data. The information to be directed to the motors and encoders is stored in an *n x 3* control matrix u. In this layer the user develops the control laws of position and/or force of the manipulator and it is possible to carry through the task simulation.

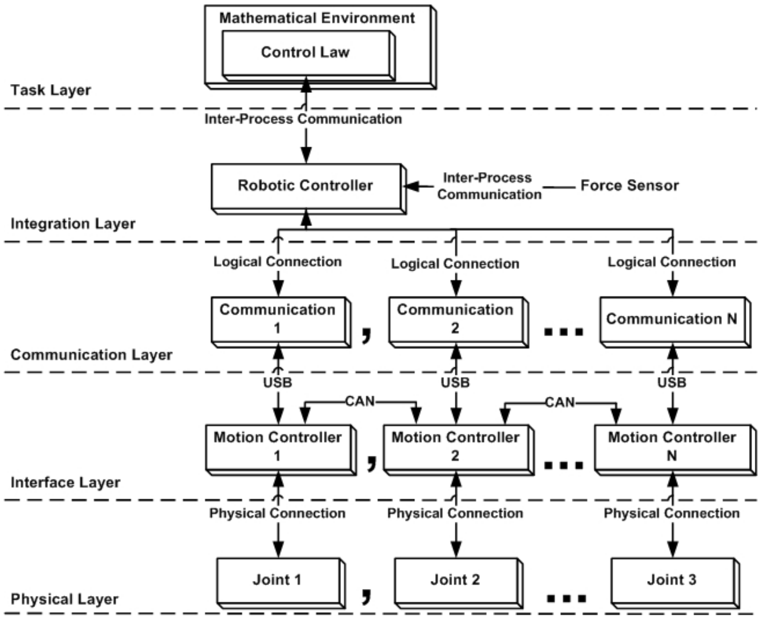


Fig. 2. Functional architecture of proposed reference model applied to interaction tasks.

4.2 Integration Layer for interaction tasks

In the integration layer the concatenation and the organization of all the information coming from the sensors and to be sent to the superior layer are done. In case of the inclusion of a new hardware to the system, it is necessary to add its control structure to this layer. This is carried through by a high-level application that manages the power unit and control unit. Preventing any irregular movements and danger situations and controlling the components of the lower level. In this software the controller's components can be activated or disabled independently.

4.3 Communication Layer for interaction tasks

The communication layer controls the data transfer by managing the interface USB (Universal Serial Bus 2.0) and the industrial protocol CAN (Campus Area Network), both high performance communication devices. The USB makes a system interconnection through a star form topology, which has the computer as a central knot. Each USB door supports up to 127 devices and, in this manner, it is possible to connect a great quantity of joints to the controller. The protocol CAN form the bus between the secondary knots (motion controllers) and the result structure is a redundant net architecture. The implementation of this bus is still being explored and intends to introduce the possibility of a joint to access information of another joint without passing through the central knot. This will increase the performance of the net and it gives the opportunity to an implementation of the system of control without the central knot: a totally embedded control. The resultant architecture communication is presented in Fig. 3.

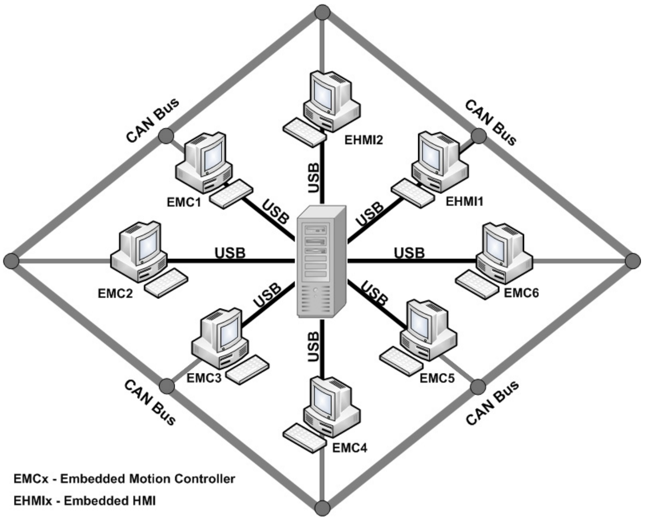


Fig. 3. Communication architecture for interaction tasks.

4.4 Interface Layer for interaction tasks

The interface layer comprises the embedded systems that carry out the control of the robotic joints, named motion controllers. Each of these motor digital controllers decodes the corresponding encoder signal and generates the modulation width pulse (PWM) to the control of the respective motor. Each of these systems has an optical isolated interface to prevent any inadequate return to the processor. It possesses a great amount of expansion doors, which allows the connection of other tools.
We developed the controller with a modular architecture to have an independent control for each joint and so, divide the mathematical complexity among the processors of the system. This results in a distributed processing organized by the central knot (computer), where the operations occur in parallel. This methodology facilitates the expansion and maintenance of the system.
Currently the system operates with a medium tax of update of the signals of 1 ms, only for a convention of literature. In case of necessity this largeness can be diminished.

4.5 Physical Layer for interaction tasks

The most inferior layer, here denominated physical layer, is the power unit of the motors and the angular position sensors.

5. Experimental Environment

The retrofitting methodology was validated with the adaptation of an old anthropomorphic manipulator, model *Rv15*, produced by the *REIS Robotics*, for interaction tasks. Where was substituted the proprietary controller by the new open-architecture controller and coupled a force sensor in system.

The *REIS Rv15* robot has six rotating joints acted by electric motors and the angular positions measurement are done using incremental optical encoders. It is a manipulator with a topology that is very common in industry applications, which constitutes an anthropomorphous arm (joints 1, 2 and 3) with a spherical wrist (joints 4, 5 and 6).
The Fig. 4 presents a complete diagram of the embedded five-layer open-architecture robotic controller for an industrial manipulator, containing it data flow and the systems interconnections

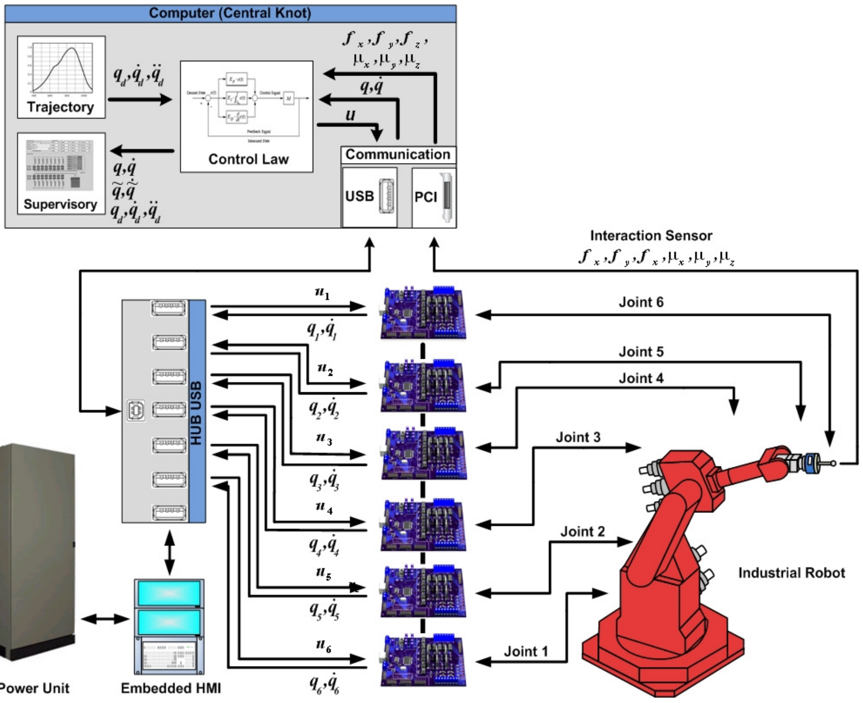
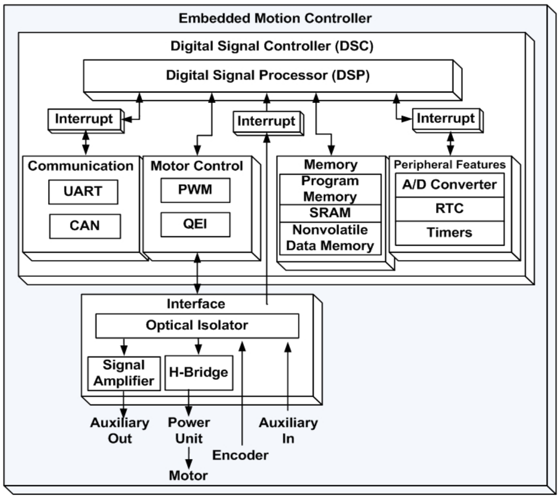


Fig. 4. Experimental environment for interaction tasks.

5.1 Hardware architecture description

The system's hardware was developed and built using high performance and reliability, low cost and easiness to be found in the market components. The Fig. 5 shows the diagram of internal blocks used in the motion controllers.
The main component is a digital signal controller (*DSC*) produced by the *Microchip Technology Inc.* named dsPIC30F6010A. It operates with 16-bits, in a 120 MHz frequency with a package TQFP of 80 pins, and is one integrant of the family of the motors control. It possesses a great amount of well differentiated modules including an ample program memory with a 144K and a non-volatile memory with 4096 bytes for information storage. It has 16 ways for A/D conversions and the necessary modules of communication. For the communication through USB we used a component which carries through the conversion of module UART for the bus. This component supports transference taxes up to 3 Megabaud and is manufactured by the *FTDI* (*Future Technology Devices International Ltd*).



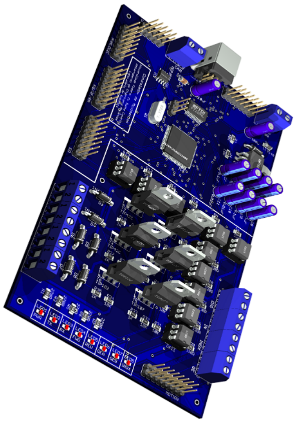


Fig. 5. Interface architecture for interaction tasks.

Moreover, it possesses other functionalities, including the generation of a digital external signal oscillator with changeable frequencies. Besides this, the same manufacturer produces available Royalty-Free drivers for many operational systems, for this form of implementation. To implement the requirements for the physical layer defined by the ISO-11898, we connect the CAN industrial protocol to a transceiver of high speed, which supports until 1Mb/s.
The system firmware implementation uses the high level language C. This is completely modulated and organized in units, to facilitate modifications. All the modules operate with interruptions of the processor with distinct priorities, such that an operation of less priority doesn't delay a higher importance process.
The module of motion controller is also composed by a 16 bits PWM generator and by a module to read the quadrature encoder (named QEI), which we extend for 32 bits. Connected to it there is an optical decoupling barrier and an H-bridge for the control of the power unit that supports 100V and 8A. There are also amplified auxiliary output channels, which operate until 100V and 6A. To protect the system, the encoder entrances and the auxiliary inputs also have been connected to optical decoupling barrier. Internally there is still a great amount of resources that had not been used and that can be useful in future upgrades of the system.
The experimental validation of the proposal was accomplished with implementation of a indirect force control strategy, the impedance control that will be presented to follow.

5.2 Firmware architecture description

The real time processing is obtained with the modularity of the embedded controller. These modules communicate only through hardware interrupts with eight priority levels. The software architecture and realistic physical modelling of the sensors and actuators provided to system a high response time.

The servo motors are controlled through a embedded self-tuning PID controller that uses the linear actuator dynamic model. The Fig. 6 presents the validation of dynamic model.

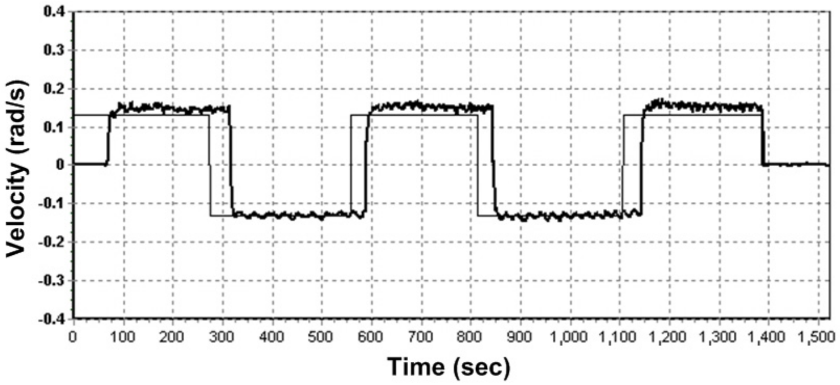


Fig. 6. Validation of the dynamic model behaviour.

5.3 Software architecture description

The software was developed using a high level object-oriented language (C++), for the Linux operating system recompiled for real-time application interface (RTAI). The control system monitors the processor activity, because most processes works through threads.
Like this, when the processor activity reaches a critical level, the threads priorities are altered favouring controller essential tasks.

5.4 Impedance control

The manipulator control strategies for interaction tasks are grouped usually in two categories: indirect force control and direct force control. The first approaches the movement with an implicit force feedback only based in movement control, the other supplies the possibility of force control for a wanted value, through an explicit force feedback (Sciavicco & Siciliano, 2000).
The classic impedance control approaches contact force indirectly modeling the interaction as a mass-spring-damper system. The indirect relationship is a consequence of force sign influence on control law, the objective is to adapt the manipulator dynamic behavior in contact with the environment and not to fulfill a position and/or force trajectory. This way, an explicit force feedback doesn't exist in system, because, this signal just supplies the system impedance in contact with a surface.
Therefore, the fundamental philosophy of impedance control, in agreement with (Hogan, 1985), is that the control system regulates the manipulator impedance, that is defined by relationship between the speed and applied force, Fig. 7 (Zeng & Hemami, 1997). The formulation of this control strategy is presented to follow, in the equation 1.

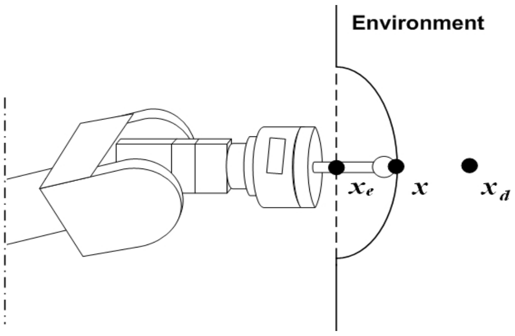


Fig. 7. Impedance force control.

$$M_D\ddot{\tilde{x}} + K_D\dot{\tilde{x}} + K_P\tilde{x} = h_A \qquad (1)$$

Where, M_D is the mass matrix, K_D is the damping matrix, K_P is the stiffness matrix and h_A is the interaction force matrix.
Equally to position control for inverse dynamics, where the impedance control is based, the integral knowledge of manipulator dynamics is admitted. In this way, the accurate knowledge of object elasticity characteristics or contact environment is not necessary in this control strategy (Yoshikawa, 2000).

5.5 Results

The implemented control strategy uses force feedback only to regulate the manipulator impedance, assuming that the manipulator is in contact with operation surface. In this way, when some force be detected the control law only will regulate the impedance to establish the system.
The application used to validate the developed controller for interaction tasks is based in this characteristic of the impedance control, however, in this case, the end-effector isn't in contact with the surface. Therefore, the manipulator is immobile, admitting to be in wanted impedance profile, and when detects external force controls the system impedance (Fig. 8).
The joint speed profiles generated in experiments are present in Fig. 9.

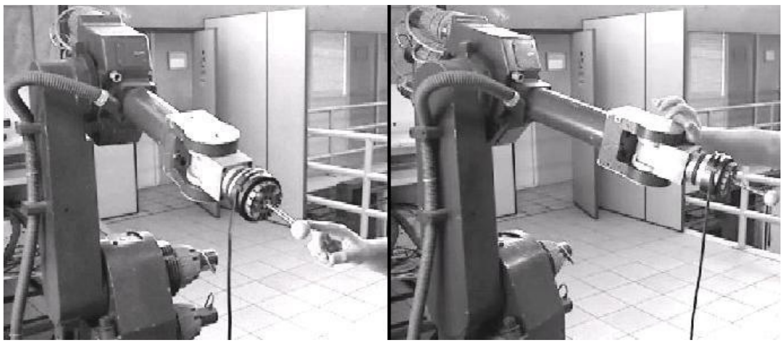


Fig. 8. Interface architecture for interaction tasks.

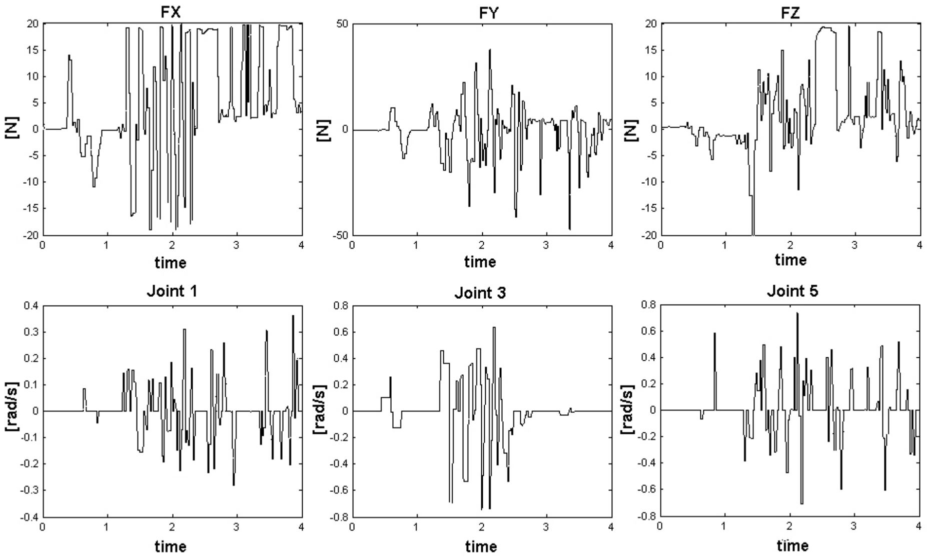


Fig. 9. Force and joint velocities profile of the impedance control.

The system validation experiment accomplished consists in a parallel manipulation on robot workspace limits. The force profiles, detected by *JR3* force and moments sensor, only adapt manipulator dynamic behaviour with environment dynamic characteristics. In this way, a profile of force control is used to drive the joints velocity behaviour to a desired impedance profile.

6. Conclusion

In this work was considered a new reference model for a control system functional architecture applied to open-architecture robot controllers. The proposed approach was applied for integrally developing of a five-layer based open-architecture robotic controller considering interaction tasks. The architecture uses parallel and distributed processing techniques and circumvents the necessity of compliance in system, allowing a real-time processing of the application and the total information control.
Old manipulator retrofitting considers the problem of including controllers with new functionalities as force control. The main characteristics of these systems are high-stiffness and position control. These characteristics restrict response time of the system. Therefore, an open-architecture system can be projected to operate in real-time.
The proposed reference model for open-architecture robot controllers was experimentally validated including the implementation of an indirect force control strategy in the robot controller. Practical tests have shown the interest of the proposed architecture in terms of controller flexibility, costs and maintenance and high capacity of processing.
This reference model clarifies the concept of robot controller and explains the internal modules that compose robot control unit. The system decomposition makes possible the optimization of internal modules for a specific task, e.g., interaction tasks. In this way, it is

possible to include new functionalities to the system, e.g., other feedback signals, new actuators or dedicated processors for a specific problem, e.g., resolution of redundancy or inverse kinematics.
In the actual stage, the researchers have been focused on the theoretical aspects of the problem. Further works will consider the model validation and experimental applications.

7. References

Abele, E.; Weigold, M. & Rothenbücher, S. (2007). Modeling and identification of an industrial robot for machining applications, *CIRP Annals- Manufacturing Technology*, Vol. 56, No. 1, page numbers 387–390.

Bona, B.; Indri, M. & Smaldone, N. (2001). Open system real time architecture and software design for robotcontrol, *Proceedings 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Vol. 1.

Donald, S. & Dunlop, G. (2001) Retrofitting path control to a unimate 2000b robot, *Proccedings 2001 Australian Conference on Robotics and Automation*, Vol. 14, page number. 15.

Ford, W. (1994). What is an open architecture robot controller?, *Proceedings of the 1994 IEEE International Symposium on Intelligent Control*, page numbers 27–32.

Hong, K.; Choi, K.; Kim, J. & Lee, S. (2001). A pc-based open robot control system: PC-ORC, *Robotics and Computer Integrated Manufacturing*, Vol. 17, No. 4, page numbers 355–365.

Hogan, N. (1985). Impedance Control: An Approach to Manipulation, Parts I-Ill, *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 107, No. 1, page numbers 1–24.

Lages, W.; Henriques, R. & Bracarense, A. (2003). Arquitetura aberta para retrofitting de robôs, *Manet Notes Workshop*, Bragança Paulista, SP, Brazil.

Lippiello, V.; Villani, L. & Siciliano, B. (2007). An open architecture for sensory feedback control of a dual-arm industrial robotic cell, *Industrial Robot: An International Journal*, Vol. 34, No. 1, page numbers 46–53.

Lutz, P. & Sperling, W. (1997). Osaca the vendor neutral control architecture, *Proceedings European Conference Integration in Manufacturing*, page numbers 247–256.

Macchelli, A. & Melchiorri, C. (2002). Real time control system for industrial robots and control applications based on real time Linux, *15th IFAC World Congress*, page numbers 21-26, Barcelona, Spain.

Nacsa, J. (2001). Comparison of three different open architecture controllers, *Proceedings of IFAC MIM*, page numbers 2–4, Prague.

Pritschow, G. ; Altintas, Y.; et al. (2001). Open controller architecture–past, present and future, *CIRP Annals-Manufacturing Technology*, Vol. 50, No. 2, page numbers 463–470.

Sciavicco, L. & Siciliano, B. (2000). Modelling and Control of Robot Manipulators. Springer.

Yoshikawa, T. (2000). Force control of robot manipulators, *Proceedings. ICRA '00 IEEE International Conference on Robotics and Automation*, Vol. 1, page numbers 220–226.

Zeng, G. & Hemami, A. (1997). An overview of robot force control, *Robotica*, Vol. 15, No. 05, page numbers 473–482.

Collaboration Planning by Task Analysis in Human-Robot Collaborative Manufacturing System

Jeffrey Too Chuan Tan, Feng Duan,
Ryu Kato and Tamio Arai
University of Tokyo
Japan

1. Introduction

The shifting manufacturing requirements to high flexibility and short production cycle have urged the emerging of human-robot collaborative type of manufacturing systems. Human-robot collaboration is a dream combination of human flexibility and machine efficiency. However, in order to materialize this paradigm shift in manufacturing history, the interaction between human and robot in the perspective of collaborative operation has to be fully investigated. Many studies had been conducted in the area of human-robot collaboration in manufacturing (Kosuge et al., 1994; Oborski, 2004). Modeling techniques (Rudas & Horvath, 1996) provide an initial step to study on this collaboration relationship even before system development. To ensure a more human-centered solution, task analysis is adapted for the modeling approach in this study. The purpose of this work is to develop a modeling framework based on task analysis approach to assist human-robot collaboration planning in manufacturing systems.

The entire development of this work is illustrated in a modeling development of an actual cable harness assembly in a prototype cellular manufacturing system (Duan et al., 2008). The outline of the paper is arranged as the following: Section 2 provides the literature reviews on human-robot collaboration in manufacturing and the overview of the prototype cellular manufacturing system setup together with the assigned cable harness assembly operation. Section 3 presents entire development of collaboration planning by task analysis including the brief introduction on task analysis approach, task decomposition by hierarchical task analysis, and collaboration analysis. Section 4 discusses the design enhancements by the modeling framework in operation process design and further extensions in human skill analysis, safety assessment and operation support. The modeling design is implemented in a prototype production cell to perform model validation and operation performance evaluation as illustrated in Section 5. Section 6 concludes the work and states the suggestions for future work.

2. Human-Robot Collaboration in Manufacturing

2.1 Human-Robot Collaboration

Many efforts had been contributed in the study on human-robot interaction. Agah had presented a general taxonomy on human interaction with intelligent systems (Agah, 2000). In manufacturing environment, Stahre had discussed several human-robot interaction problems (Stahre, 1995). Over the years, there are many proposals on robotic human operator assistance: Robot Assistant rob@work (Helms et al., 2002), COBOT (Colgate et al., 1996), KAMRO (Karlsuhe Autonomous Mobile Robot) (Laengle et al., 1997), CORA (Cooperative Robotic Assistant) (Iossifidis et al., 2002), Humanoid Service Robot HERMES (Bischoff, 2001) and The Manufacturing Assistant (Stopp et al., 2002). Although much work had been conducted on human-robot interaction, the view point of this work is quite deviated from the common goal of these studies. The ultimate aim of this work is to improve manufacturing systems by effective human-robot collaboration, rather than how much 'social' between human and robot. Therefore, manufacturing requirements become the main criterion in the collaboration planning. On the other hand, conventional assembly planning focuses on simplifying assembly process for automation. The lack in addressing human-robot collaboration in design for assembly principles has motivated this work to develop a design approach to address human-robot collaboration in assembly planning.

2.2 Practical Development in Cellular Manufacturing System

In order to ensure practicability of this work, the entire development is linked on an actual cable harness assembly system in cellular manufacturing. Also known as cell production, cellular manufacturing is a human-centered production system that catered for complex and flexible assembly requirements (Isa, K. & Tsuru, 2002). The prototype production cell design in this project is shown in Fig. 1 (Duan et al., 2008).

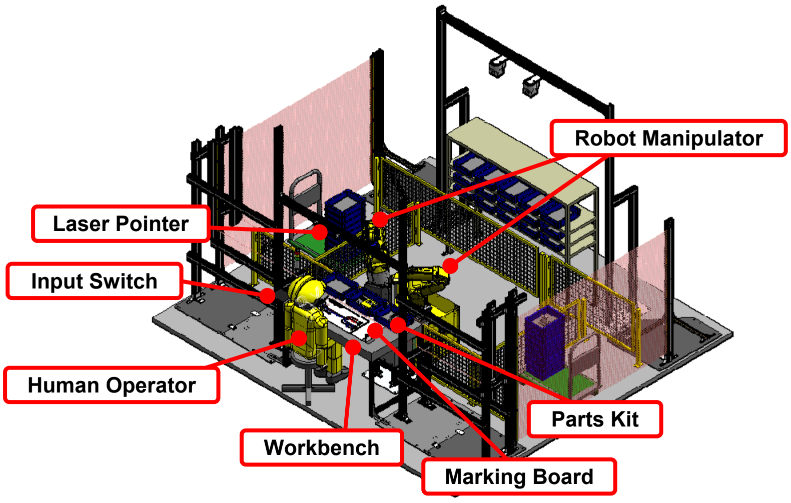


Fig. 1. Prototype production cell design for cellular manufacturing

In this cellular manufacturing system, a mobile twin robot manipulators system is assigned to collaborate with a human operator to conduct a cable harness assembly operation (Fig. 2). The robot manipulators system is able to navigate itself within the production cell (between the parts rack and the workbench) and to facilitate collaborative assembly operations on the workbench. The human operator conducts the assembly operation in sitting position and uses the input switches to control the progress of the operation. The workbench is incorporated with a liquid crystal display television (LCD TV) to provide multimedia assembly information to the human operator. Additional position information is indicated by the laser pointer system. More detailed descriptions on the prototype production cell are available in Duan's work (Duan et al., 2008).
The completed cable harness assembly is shown in Fig. 2. The human operator assembles components from the parts kit onto the marking board to form the product. The required tasks in one assembly includes cable insertion on connector and terminal, tape marking and cable tie binding, and the assembly of metal plate. This assembly process will be discussed further in the following section for collaboration planning.

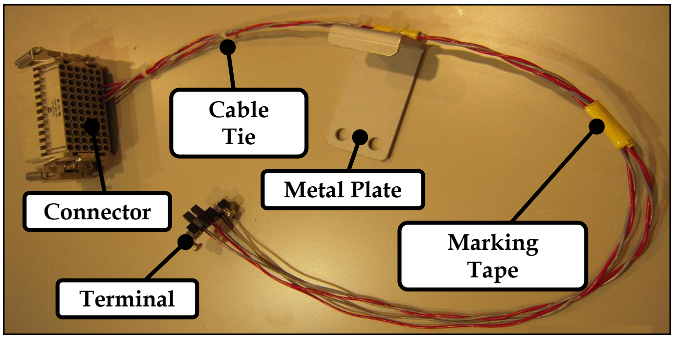


Fig. 2. Cable harness assembly

3. Collaboration Planning by Task Analysis

3.1 Task Analysis

The main challenge in human-robot collaboration study is the complexity of human nature because normal mathematical computer modeling techniques are difficult to study on the behavior. Many research studies developed the collaboration modeling from the 'machine' point of view (Kosuge et al., 1998; Mizoguchi et al., 1999) resulting 'machine-driven' collaboration. Therefore, with the motivation to develop a more 'human-centered' collaboration in production system, this work has adopted task analysis method, which provides a more 'natural' way to define and study on human activities. Task analysis is a widely used scientific methodology to model human task in various ergonomics and human factors studies (Hodgkinson & Crawshaw, 1985), medical surgery (Sarker et al., 2008), error prediction (Lane et al., 2006), and software interface design (Mills, 2007; Richardson et al., 1998). The main advantage of task analysis is the ability to describe human activities with 'abstract descriptions'. This *temporal abstraction* (Killich et al., 1999) is very useful in human-robot collaboration modeling especially when the actual optimal sequence of activities is yet to be defined. In task analysis development, the task is defined as goal and the required activities (sub goals) that must be carried out to achieve it (Annett & Duncan, 1967; Hollnagel, 2006), and continuous branch out in sub goals to form a hierarchical tree. This hierarchical task analysis (HTA) approach (Kirwan & Ainsworth, 1992; Shepherd, 1998; Stanton, 2006) is adapted in this study to extend its capability to address human-robot collaboration in production systems. In the following discussion, the cable harness assembly operation is being decomposed into hierarchical assembly tasks tree to enable further investigation on the collaboration between human and robot in the collaborative operation.

3.2 Task Decomposition by Hierarchical Task Analysis

Fig. 3 shows the general operation flow of the cable harness assembly. The whole operation consists of mainly five different tasks. The first task is parts kit preparation, which is to gather all the required assembly components in the parts kit. The assembly begins with cable insertion to the connector in second task. In third task, the cables are being arranged on the marking board and bond with marking tape and cable tie. The purpose of the marking board is as a guide for the cables and assembly positions identifications. In the fourth task, the other ends of the cables are then inserted into the terminal. The final task is the metal plate assembly.

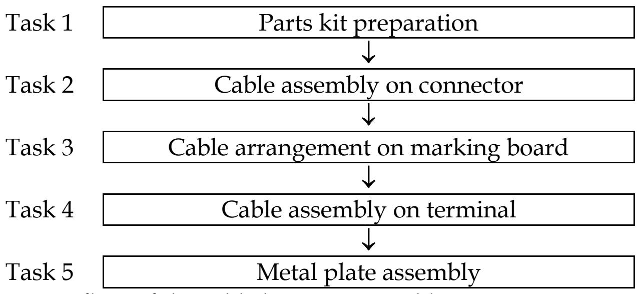

Task 1
Parts kit preparation
Task 2
Cable assembly on connector
Task 3
Cable arrangement on marking board
Task 4
Cable assembly on terminal
Task 5
Metal plate assembly

Fig. 3. General operation flow of the cable harness assembly

Referring to HTA development guideline by Stanton (Stanton, 2006), the entire cable harness assembly is being decomposed into hierarchical task tree (Tan et al., 2008a). The overall operation objective is set as the main goal followed by general tasks in the assembly plan level as the sub goals. Then, on each sub goals, the decomposition is further branched out into control plan level. Table 1 summarizes the decomposition of the cable harness assembly into a HTA table. '*Assemble cable harness*' (Super-ordinate 0) is the main goal of the entire operation. Based on the general operation flow in Fig. 3, the first hierarchical level of sub goals, '*Prepare parts kit*', '*Assemble cables on connector*', '*Arrange cables on marking board*', '*Assemble cables on terminal*', and '*Assemble metal plate*' (Super-ordinate 1, 2, 3, 4, and 5) are the general assembly tasks needed to achieve the main goal. The decompositions continue from the first level sub goals into another two hierarchical levels lower until all the task components are all well defined, as considered 'fit-for-purpose' (Stanton, 2006). In all the task levels, the execution sequence of the corresponding hierarchical level is defined in *Plan* components. With the developed HTA table, the entire cable harness assembly operation is well defined in a hierarchical task tree form for further development on collaboration planning. The HTA table can be represented in a graphical form for better visualization illustrated in the next section.

Super-ordinate	Task components – Operation or Plan
0	*Assemble cable harness* Plan 0: Do 1 then 2 then 3 then 4 then 5 then exit 1. Prepare parts kit 2. Assembly cables on connector 3. Arrange cables on marking board 4. Assemble cables on terminal 5. Assemble metal plate
1	*Prepare parts kit* Plan 1: Repeat 1.1 then 1.2 for three parts then exit 1.1 Arrange parts into tray 1.2 Check parts //
1.1	*Arrange parts into tray* Plan 1.1: Do 1.1.1 then 1.1.2 then exit 1.1.1 Retrieve part container // 1.1.2 Grab part from container //
2	*Assemble cables on connector* Plan 2: Repeat 2.1 then 2.2 for two cables then 2.3 then exit 2.1 Secure cable contacts on connector 2.2 Temporary fix cable ends // 2.3 Set connector on marking board
2.1	*Secure cable contacts on connector* Plan 2.1: Repeat 2.1.1 then 2.1.2 then 2.1.3 for two cables then exit 2.1.1 Get cable from cable kit // 2.1.2 Hold and locate insertion point // 2.1.3 Insert cable contact into connector with driver //
2.3	*Set connector on marking board* Plan 2.3: Do 2.3.1 then 2.3.2 then exit 2.3.1 Release connector // 2.3.2 Get and place connector on marked location //
3	*Arrange cables on marking board* Plan 3: Do 3.1 for two cables then 3.2 for two marking tapes then 3.3 for two cable ties then exit 3.1 Form cables on marking board 3.2 Paste marking tape on cables 3.3 Fasten cables with cable tie
3.1	*Form cables on marking board* Plan 3.1: Do 3.1.1 then 3.1.2 then exit 3.1.1 Arrange cables along marked track // 3.1.2 Fasten cable ends //
3.2	*Paste marking tape on cables* Plan 3.2: Repeat 3.2.1 then 3.2.2 for two marked locations then exit 3.2.1 Get marking tape // 3.2.2 Paste marking tape on marked location //
3.3	*Fasten cables with cable tie* Plan 3.3: Repeat 3.3.1 then 3.3.2 for two marked locations then exit

	3.3.1 Get cable tie // 3.3.2 Fasten cable tie on marked location //
4	*Assemble cables on terminal* Plan 4: Do 4.1 for two cables then 4.2 then exit 4.1 Secure cable ends on terminal 4.2 Set terminal on marking board
4.1	*Secure cable ends on terminal* Plan 4.1: Do 4.1.1 then repeat 4.1.2 then 4.1.3 for two cables then exit 4.1.1 Get terminal from part tray // 4.1.2 Hold and locate insertion point // 4.1.3 Insert cable end into terminal with driver //
4.2	*Set terminal on marking board* Plan 4.2: Do 4.2.1 then 4.2.2 then exit 4.2.1 Release terminal 4.2.2 Get and place terminal on marking board //
5	*Assemble metal plate* Plan 5: Do 5.1 then 5.2 then exit 5.1 Secure cables on metal plate 5.2 Set metal plate on marking board
5.1	*Secure cables on metal plate* Plan 5.1: Do 5.1.1 then repeat 5.1.2 then 5.1.3 then exit 5.1.1 Get metal plate from part tray // 5.1.2 Hold metal plate // 5.1.3 Fasten cables on metal plate with cable tie //
5.2	*Set metal plate on marking board* Plan 5.2: Do 5.2.1 then 5.2.2 then exit 5.2.1 Release metal plate // 5.2.2 Get and place metal plate on marking board //

Table 1. HTA table of the cable harness assembly

3.3 Collaboration Analysis

The above task decomposition development based on HTA guideline has provided a coarse task outline of the cable harness assembly. The next step is to conduct detailed analysis for collaboration planning in task level. The analysis can be done in two stages, qualitative and quantitative, based on the complexity to determine the optimum collaboration solution for a given task. In qualitative analysis, the performance requirements of the task are compared qualitatively with the capabilities of human and robot to identify possible collaboration solution. If the optimum solution is not apparent, quantitative analysis can be conducted to score the possible solutions based on the performance requirements.

Qualitative Analysis for Collaboration Task Identification. In qualitative analysis for collaboration task identification, the possible collaboration solution for each task is identified based on the comparison of the strength of human operator and robot manipulator with respect to performance requirements. Together with the definitions by Helms et al. on four types of human-robot cooperation in industrial environment: *independent operation*, *synchronized cooperation*, *simultaneous cooperation*, and *assisted cooperation*

(Helms et al., 2002), the collaboration tasks are identified and summarized in Table 2 for the first hierarchical level assembly tasks in cable harness assembly.

The objective of Task 1, '*Prepare parts kit*' (Super-ordinate 1) is to gather and arrange the assembly components into parts kit. This objective can be achieved easily by robot system using bin picking technique. Hence, it is suitable to be assigned to robot system for higher efficiency. Task 2, '*Assemble cables on connector*' (Super-ordinate 2) requires handling of flexible cables for assembly. Therefore, human operator's flexibility is needed in this task. However, based on previous study (Pongthanya et al., 2008), the mental workload for the human operator to search for the correct insertion holes from the multi-holes connector can be relatively high and time consuming. Therefore, a possible collaboration by using robot system to indicate cable insertion holes by holding the connector under a fixed beam from the laser pointer might be a good solution. However, further quantitative analysis might be needed to justify this collaboration proposal. '*Arrange cables on marking board*' in Task 3 (Super-ordinate 3) requires handling of cables, marking tape and cable tie. Hence, these highly flexible operations are suitable to be assigned to human operator. Task 4, '*Assembly cables on terminal*' (Super-ordinate 4) has the similar job requirements as in Task 2. Therefore, same collaboration solution might be applied. Task 5, '*Assembly metal plate*' (Super-ordinate 5) involves operation to fasten the cables on the metal plate with cable ties. A possible collaboration solution might be proposed, which the robot system can help to hold the metal plate to allow the human operator to use both hands to fasten the cables with cable ties.

Super-ordinate	**Task components**	**Collaboration**
1	*Prepare parts kit* 1.1 Arrange parts into tray 1.2 Check parts //	*Independent operation* by robot manipulators to prepare the parts kit
2	*Assemble cables on connector* 2.1 Secure cable contacts on connector 2.2 Temporarily fix cable ends // 2.3 Set connector on marking board	*Assisted cooperation* by robot manipulator to hold the connector and indicate assembly points while human operator inserts the cable contacts
3	*Arrange cables on marking board* 3.1 Form cables on marking board 3.2 Paste marking tape on cables 3.3 Fasten cables with cable tie	*Independent operation* by human operator due to the requirement to handle flexible cables
4	*Assemble cables on terminal* 4.1 Secure cable ends on terminal 4.2 Set terminal on marking board	*Assisted cooperation* by robot manipulator to hold the terminal and indicate assembly points while human operator inserts the cable ends
5	*Assemble metal plate* 5.1 Secure cables on metal plate 5.2 Set metal plate on marking board	*Assisted cooperation* by robot manipulator to hold the metal plate while human operator fastens the cables with cable ties

Table 2. Collaboration identification from the HTA table

Quantitative Analysis by Analytic Hierarchy Process (AHP). When multiple requirements (productivity, fatigue, safety, etc.) and solutions (human system, robot system, human-robot system, etc.) are available for a given task and the optimum solution might not be apparent by qualitative analysis, collaboration analysis by Analytic Hierarchy Process (AHP) [Saaty, 2008; Saaty, 1994] approach can be conducted to assess the task quantitatively. Taking Task 2, '*Assemble cables on connector*' (Super-ordinate 2) as example, the following description illustrates the quantitative analysis by AHP to verify the selection of human-robot collaborative system over human only system for the given task.

Four performance requirements, namely, productivity (assembly duration), quality (assembly error), human fatigue (human operator tiredness), and safety (human operation safety), are set as the *criteria* in the AHP analysis. The evaluation is done based on comparison between human only system and human-robot system as *alternatives* (fully automated system is less practical to be considered due to the complexity of flexible cable handling in this task). Fig. 4 shows the AHP model of Task 2. In order to compute the *priorities* (relative weight of the nodes) of *criteria* and *alternatives*, pairwise comparison matrix of criteria (Table 3), pairwise comparison matrix of alternatives with respect to productivity (Table 4), quality (Table 5), human fatigue (Table 6), and safety (Table 7) are developed from the analysis. Based on the fundamental scale of pairwise comparisons (Harker & Vargas, 1987), the intensity of importance values are assigned to the pairwise comparison matrixes by comparing the importance between two criteria. The *priorities* are then being calculated by summing each row and dividing each by the total sum of all the rows in the corresponding matrix.

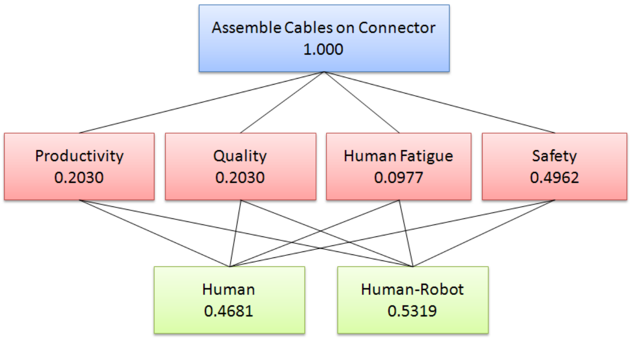


Fig. 4. AHP model of Task 2

From Table 3, the productivity and quality have the same importance (intensity of importance = 1) in achieving the assembly operation (goal). The safety criterion has been given higher priority in the pairwise comparison (intensity of importance = 2) over productivity and quality due to the high risk nature of the close range collaboration. The intensity of importance of productivity and quality over human fatigue also has been set to 2 to put more focus on mental stress of the human operator during close range collaboration with the robot system. The safety has much stronger importance (intensity of importance = 6) over human fatigue. The pairwise comparisons on the alternative systems with respect to each criterion are being judged based on the actual system performance. The improvements from human-robot design can be given a greater importance in the productivity (Table 4) and quality (Table 5). The assistance from robots also greatly reduced the workload burden of the human operator (Table 6). However, due to the close range collaboration, the safety level is much lower in human-robot design (Table 7). The final *priorities* obtained for human system is 0.4681 and human-robot system is 0.5319 (Fig. 4). These have proven that human-robot system is much preferred solution that fit well to the performance criteria.

	Productivity	**Qualit y**	**Human Fatigue**	**Safety**	**Priorities**
Productivity	1	1	2	1/2	0.2030
Quality	1	1	2	1/2	0.2030
Human Fatigue	1/2	1/2	1	1/6	0.0977
Safety	2	2	6	1	0.4962

Table 3. Pairwise comparison matrix of the performance criteria with respect to the goal

	Human	**Human-Robot**	**Local Priorities**	**Global Priorities**
Human	1	1/9	0.1	0.0203
Human-Robot	9	1	0.9	0.1827

Table 4. Pairwise comparison matrix of the alternative systems with respect to productivity

	Human	**Human-Robot**	**Local Priorities**	**Global Priorities**
Human	1	1/9	0.1	0.0203
Human-Robot	9	1	0.9	0.1827

Table 5. Pairwise comparison matrix of the alternative systems with respect to quality

	Human	**Human-Robot**	**Local Priorities**	**Global Priorities**
Human	1	1/6	0.1429	0.0014
Human-Robot	6	1	0.8571	0.0838

Table 6. Pairwise comparison matrix of the alternative systems with respect to human fatigue

	Human	**Human-Robot**	**Local Priorities**	**Global Prioritie s**
Human	1	5	0.8333	0.4135
Human-Robot	1/5	1	0.1667	0.0827

Table 7. Pairwise comparison matrix of the alternative systems with respect to safety

Collaboration Role Assignment. After the collaboration solution for each task in the first hierarchical level has been identified and justified, collaboration roles (*Human-Robot*, *Human*, and *Robot*) can be assigned to lower hierarchical task components (Table 8). The assignment can greatly assist the later control plan developments, for instance, robot system programming or human operator operation support development. The task modeling with collaboration planning is then can be completed in the task modeling tool developed in this project (Tan et al., 2009b) with color task role indicators for collaboration relationship visualization as shown in the graphical task model in Fig. 5.

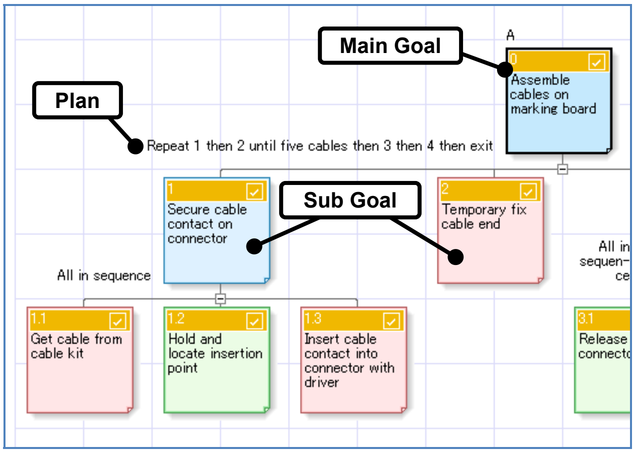

Main Goal
A
0
Assemble cables on marking board
Plan
Repeat 1 then 2 until five cables then 3 then 4 then exit
1
Secure cable contact on connector
Sub Goal
2
Temporary fix cable end
All in sequen-ce
All in sequence
1.1
Get cable from cable kit
1.2
Hold and locate insertion point
1.3
Insert cable contact into connector with driver
3.1
Release connecto

Fig. 5. Task model (partially) of a cable harness assembly (*human-robot*: blue; *human*: pink; *robot*: green)

Super-ordinate	Task components	Collaboration Roles
0	Assemble cable harness	Human-Robot
1	Prepare parts kit	Robot
1.1	Arrange parts into tray	Robot
1.1.1	Retrieve part container	Robot
1.1.2	Grab part from container	Robot
1.2	Check parts	Robot
2	Assemble cables on connector	Human-Robot
2.1	Secure cable contacts on connector	Human-Robot
2.1.1	Get cable from cable kit	Human
2.1.2	Hold and locate insertion point	Robot
2.1.3	Insert cable contact into connector with driver	Human
2.2	Temporarily fix cable ends	Human
2.3	Set connector on marking board	Human-Robot
2.3.1	Release connector	Robot
2.3.2	Get and place connector on marked location	Human
3	Arrange cables on marking board	Human
3.1	Form cables on marking board	Human
3.1.1	Arrange cables along marked track	Human
3.1.2	Fasten cable ends	Human
3.2	Paste marking tape on cables	Human
3.2.1	Get marking tape	Human
3.2.2	Paste marking tape on marked location	Human
3.3	Fasten cables with cable tie	Human
3.3.1	Get cable tie	Human
3.3.2	Fasten cable tie on marked location	Human
4	Assemble cables on terminal	Human-Robot
4.1	Secure cable ends on terminal	Human-Robot
4.1.1	Get terminal from part tray	Robot
4.1.2	Hold and locate insertion point	Robot
4.1.3	Insert cable end into terminal with driver	Human
4.2	Set terminal on marking board	Human-Robot

4.2.1	Release terminal	Robot
4.2.2	Get and place terminal on marking board	Human
5	Assemble metal plate	Human-Robot
5.1	Secure cables on metal plate	Human-Robot
5.1.1	Get metal plate from part tray	Robot
5.1.2	Hold metal plate	Robot
5.1.3	Fasten cables on metal plate with cable tie	Human
5.2	Set metal plate on marking board	Human-Robot
5.2.1	Release metal plate	Robot
5.2.2	Get and place metal plate on marking board	Human

Table 8. Collaboration role assignments

4. Design Enhancements in Collaboration Planning

4.1 Operation Process Design in Collaboration Planning

From the task analysis in task model, possible collaboration operations can be identified in the assembly level. The collaboration analysis is then further continued into control level to assign the collaboration roles between the working agents in operation. The combination of original *Plan* components and the added collaboration role assignments has represented the collaboration assembly sequences. The developed task model is used in the assembly and task planning (Barnes et al., 1997; Gottschlich et al., 2002) as well as feasible assembly sequence generations and sequence changes. The following discussion will illustrate the improvements of operation process planning by this work (Tan et al., 2008b) in cable harness assembly.

In the original human only setup of the cable harness assembly, Task 2, '*Assemble cables on connector*' requires the human operator to identify the specific insertion hole from a 12×6 holes connector to insert the cable contact. If each operation consists of five sets of cables with a total of 5×2 cable contacts, this task can be highly mentally demanding and often causes error insertions especially after a long period of working (Pongthanya et al., 2008). Fig. 6 shows the original assembly operation sequence from Task 2 to Task 3.1.

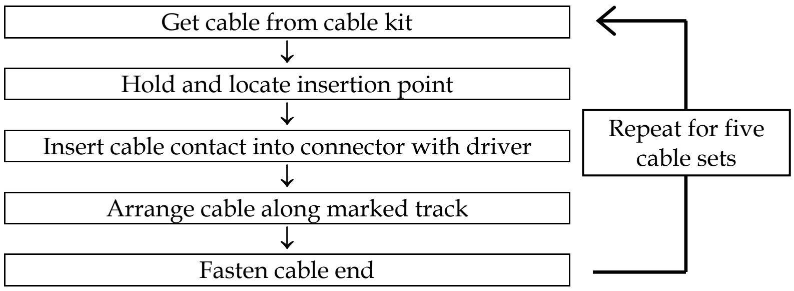

Get cable from cable kit
Hold and locate insertion point
Insert cable contact into connector with driver
Arrange cable along marked track
Fasten cable end
Repeat for five cable sets

Fig. 6. The initial assembly operation sequence from Task 2 to Task 3.1

Robot system was introduced into the operation to assist the assembly. Without collaboration planning in this work, the robot system was assigned to assist human operator by holding the connector and locating the insertion point. Fig. 7 shows the modified assembly operation sequence after adding the assistance of robot system.

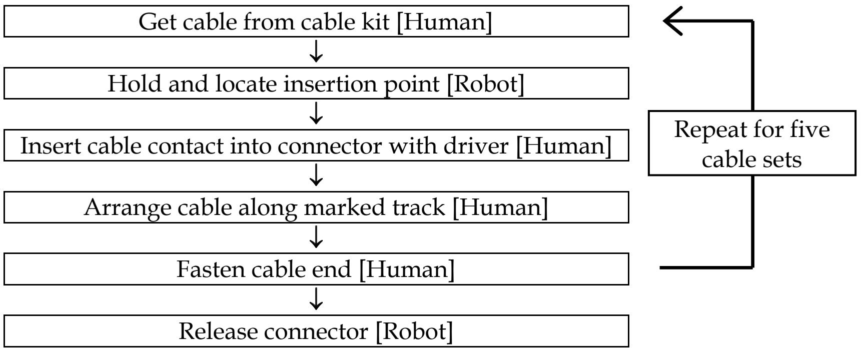

Get cable from cable kit [Human]
Hold and locate insertion point [Robot]
Insert cable contact into connector with driver [Human]
Arrange cable along marked track [Human]
Fasten cable end [Human]
Release connector [Robot]
Repeat for five cable sets

Fig. 7. Modified assembly operation sequence with robot manipulator

One of the most apparent collaboration issues from the above system was the looseness of the cables (or even being pull out from the cable fix) after the release of connector by the robot system at the end of the operation. By redesigning the operation sequence might be able to provide a solution for the issue but direct approach to revise the operation sequence planning is absence without proper task analysis.

From the task analysis and collaboration planning in this work, the cable harness assembly is being decomposed into task sequence and role assigning between human operator and robot system as explained in previous section. Based on the task model (as summarized in Fig. 8), '*Arrange cable along marked track*' and '*Fasten cable end*', are identified as repetitive steps and can be grouped into two single operation steps outside the operation loop. These steps can be simplified by adding a short step, '*Temporary fix cable end*' (Fig. 9) in the operation loop. By placing the two steps at the end of the operation, it will also solve the previous 'loosen cable' issue by fasten the cables after the release of the connector. With this, the operation sequence is being improved while the human-robot collaboration is preserved.

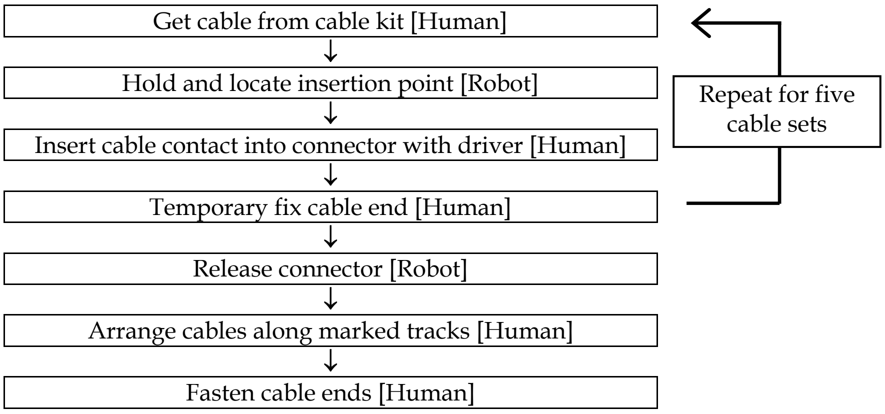

Get cable from cable kit [Human]
Hold and locate insertion point [Robot]
Insert cable contact into connector with driver [Human]
Temporary fix cable end [Human]
Release connector [Robot]
Arrange cables along marked tracks [Human]
Fasten cable ends [Human]
Repeat for five cable sets

Fig. 8. Assembly operation sequence with task analysis

From the above discussion, the design enhancements in term of (a) group repetitive steps ('*Arrange cable along marked track*' and '*Fasten cable end*'), (b) add interval step ('*Temporary fix* *cable end*'), and (c) preserve collaboration are achieved by task analysis in collaboration planning. On the other hand, by observing the first hierarchical level of tasks and plan in task modeling (Fig. 10), the precedence relationships among the tasks are well defined (red dotted arrows) to assist possible assembly sequence changes while preserving the assembly process. In the cable harness assembly, Task 3 '*Arrange cables on marking board*' and Task 4 '*Assemble cables on terminal*' are independent from each other after Task 2. Hence assembly sequence change is possible in switching the two tasks in the operation flow (Fig. 11). This assembly sequence changed operation is validated by the design implementation in the prototype production cell on the next section.

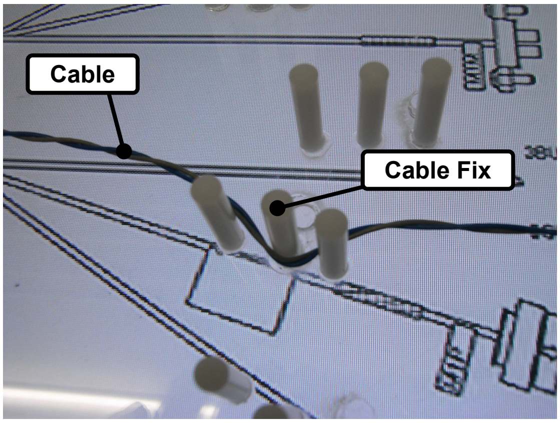
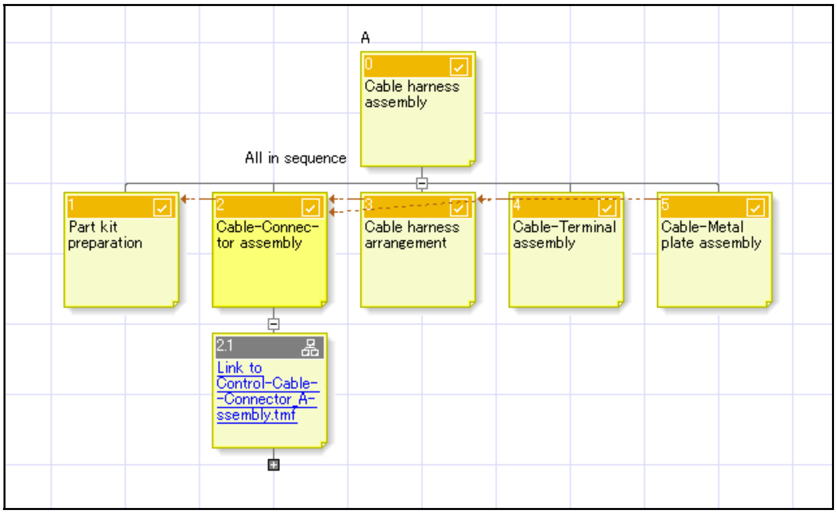


Fig. 9. Temporary cable end fixing on cable fix


A
0 Cable harness assembly
All in sequence
1 Part kit preparation
2 Cable-Connector assembly
3 Cable harness arrangement
4 Cable-Terminal assembly
5 Cable-Metal plate assembly
2.1 Link to Control-Cable--Connector A-ssembly.tmf
Fig. 10. The first hierarchical level of tasks and plan with precedence relationships (red dotted arrows)

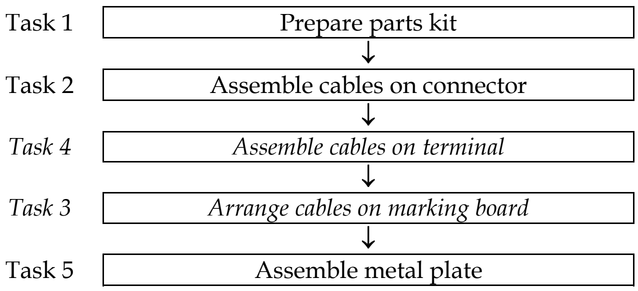

Task 1
Prepare parts kit
Task 2
Assemble cables on connector
Task 4
Assemble cables on terminal
Task 3
Arrange cables on marking board
Task 5
Assemble metal plate

Fig. 11. Cable harness assembly sequence change flow

4.2 Design Extensions for Human Skill Analysis, Safety Assessment and Operation Support

The main aim of this work is to enable collaboration planning for human-robot system in manufacturing. By adopting a more human-centered approach, task analysis enables detailed study on production operation to model and design the collaboration process. The developed task model provides several more extensions to further enhance the collaboration.

Human Skill Analysis. In human operation skill study (Duan, 2009), modeling of operation in well structured task model enables further analysis on operator's cognition and motor skill requirements in the corresponding tasks. From the task model of cable harness assembly, potential cognition and motor skills can be extracted for the corresponding tasks (Table 9) in order to evaluate the effective skills for skill transfer. The purpose of skill transfer is to provide support especially to novice operators to improve working performance and collaboration process.

Super-ordinate	Task components	Potential Cognition Skills	Potential Motor Skills
2.1.1	Get cable from cable kit	- Focus on cable's color - Focus on position of cable's head	- Grasp cable's head - Sit straightly
2.1.3	Insert cable contact into connector with driver	- Focus on position of the hole in connector - Focus on number of the hole - Focus on force feedback - Memorize order of holes in the connector	- Grasp cable's head - Elbows lower down
3.1.1	Arrange cables along marked track	- Focus on cross route - Focus on positions of cable fixes - Focus on force feedback	- Left hand holds the connector, right hand arranges cables on cable fixes - Elbows lower down - Tightly cross cables on cable fixes
3.1.2	Fasten cable ends	- Focus on cross route - Focus on positions of cable fixes - Focus on force feedback	- Hold cables while fasten - Elbows lower down - Tightly cross cables on cable fixes

Table 9. Potential cognition and motor skills for Task 2 and Task 3

Safety Assessment. Safety is the top most priority in human-robot systems. In the task modeling, safety assessment can be started as early as in the design stage. Task modeling provides a detailed analysis on human operations until lower control level to identify potential operation risk for the collaboration process. From the task model of cable harness assembly, it identifies possible human-robot collaboration in Task 2, Task 4 and Task 5. From the lower control tasks, two potential hazards can be indentified: (a) human operator's hands and/or head being trapped by robot gripper, (b) collision of robot gripper with the human operator. Table 10 shows the risk assessment on these two potential hazards in collaboration (Tan et al., 2009a) based on industrial standards ANSI/RIA R15.06 (ANSI/RIA, 1999) with reference to ISO 14121 (JIS B9702), ISO 13849-1 (JIS B9705-1), and BS EN 954-1.

Task Description	Hazard	Prior to safeguard				*PL_r (Category)*
		Severity	Exposure	Avoidance	Risk Category	
Cable harness assembly (Task 2, 4 and 5)	Trapped Risk – *Hands*	S2	E2	A2	R1	*e (4)*
	Trapped Risk – *Head*	S2	E1	A2	R2B	*d (3)*
	Collision Risk – *Hands*	S2	E2	A2	R1	*e (4)*
	Collision Risk – *Head*	S2	E1	A2	R2B	*d (3)*

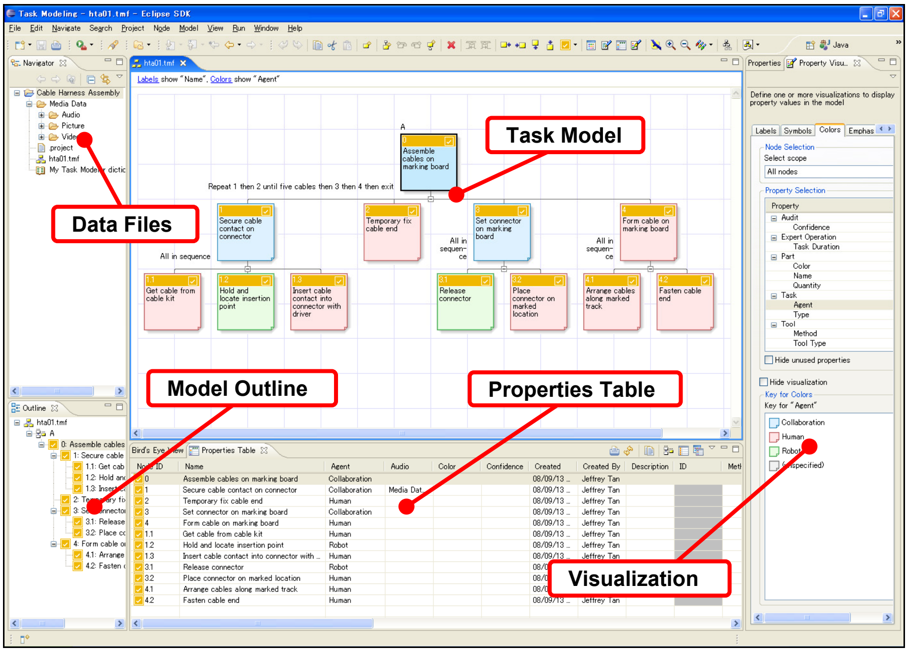
Table 10. Risk assessment on cable harness assembly (Task 2, Task 4 and Task 5)

Operation Support. One unique development in this work is to support operation by providing information to the human operator. Due to the shifting operation support from physical support, which is mainly taken care by automation, information support is one of the important factors that determine operator's working performance. In the prototype production cell for cable harness assembly in this work, a multimodal information support system (Duan et al., 2008) is developed as a man-machine interface for the human-robot collaboration system. However, in order to ensure effective information support, the content of the information has to be appropriate and relevance to the operation. Task modeling in this work has the function to extract and manage the information together with the task model. A task modeling editor (Tan et al., 2009b) (Fig. 12) is built on *IBM Task Modeler Version 6* as the development environment of task modeling. An operation properties system is developed to encapsulate relevance information to the task according to the modeling levels and support requirements. Table 11 shows the basic operation properties in the task modeling for the support information development to support the assembly operations in the prototype production cell.

Modeling Level	**Operation Properties**	**Description**
Assembly	Part	Required component
	Sub-assembly	Output of the task
Control	Agent	Subject
	Object1	Direct object
	Object2	Indirect or secondary object

	Tool	Support instrument
	Jig and Fixture	Support hardware
	Operation Duration	Time needed for the operation
	Precedence	Precedence relationship
Information Support	Operation Reference Media	Reference description in any media format
	Safety	Safety information

Table 11. Operation properties in task modeling



Bird's Eye View | Properties Table

Node ID	Name	Agent	Audio	Color	Confidence	Created	Created By	Description	ID	Meth
0	Assemble cables on marking board	Collaboration				08/09/13 ...	Jeffrey Tan			
1	Secure cable contact on connector	Collaboration	Media Dat...			08/09/13 ...	Jeffrey Tan			
2	Temporary fix cable end	Human				08/09/13 ...	Jeffrey Tan			
3	Set connector on marking board	Collaboration				08/09/13 ...	Jeffrey Tan			
4	Form cable on marking board	Human				08/09/13 ...	Jeffrey Tan			
1.1	Get cable from cable kit	Human				08/09/13 ...	Jeffrey Tan			
1.2	Hold and locate insertion point	Robot				08/09/13 ...	Jeffrey Tan			
1.3	Insert cable contact into connector with ...	Human				08/09/13 ...	Jeffrey Tan			
3.1	Release connector	Robot				08/0				
3.2	Place connector on marked location	Human				08/0				
4.1	Arrange cables along marked track	Human				08/0				
4.2	Fasten cable end	Human				08/09/13 ...	Jeffrey Tan			

Fig. 12. Task modeling editor user interface

5. Design Implementation and Operation Performance Evaluation

An actual prototype production cell for cable harness assembly (Fig. 13) is developed as design implementation to conduct validation study on the task model and collaboration planning. From the task model, low level control plan is developed to program the robot system and to generate information support to instruct the human operator during operation. From the implementation operation, the cable harness assembly operation was successfully being completed by the human-robot system based on the proposed collaboration planning. A second set of operation with the assembly sequence changed in switching Task 3 and Task 4 (Section 4.1) was also successfully being conducted to validate the assembly sequence planning in task modeling.
Operation performance evaluation was also carried out based on the comparison results between conventional manual assembly setup (Exp I) and the new human-robot collaboration setup (Exp II) in Fig. 13. A group of novice and expert operators (7 males, 22-

36 years old) had performed the assembly three times each in each of the two setups to obtain the time needed to complete the operations.

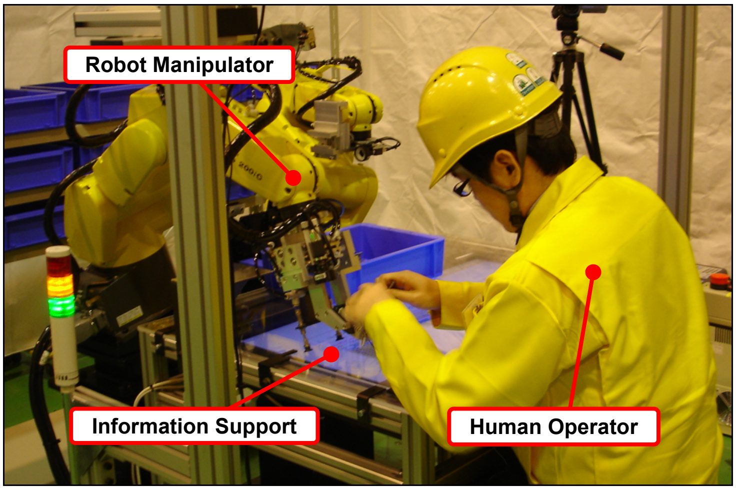


Fig. 13. Prototype production cell setup

From the results in Fig. 14, the overall performance was improved with shorter assembly duration in collaboration setup (Exp II). In the collaboration setup (Exp II), novice and expert operators had almost the same assembly duration, which meant best performance was made possible even for novice operators, who in conventional setup (Exp I) require almost double the time in first trial. The improvement in assembly quality, from 10% to 20% of assembly error (insertion error) in conventional setup (Exp I) to assembly error was totally being prevented in collaboration setup (Exp II), had proven the effectiveness of the collaboration.

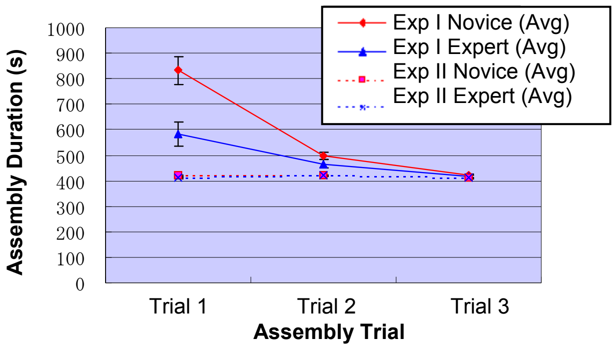


Fig. 14. Results of assembly duration in conventional setup (Exp I) and collaboration setup (Exp II)

6. Conclusions and Future Work

The challenge of this work is to develop collaboration planning between human operator and robot system in a collaborative manufacturing system by task analysis approach. The development is worked in parallel with a cable harness assembly operation in a prototype cellular manufacturing system. The core developments of this study are summarized as the following:

(a) **Task decomposition by hierarchical task analysis** – by using the capability of HTA, the entire operation is being decomposed into structured hierarchical tasks tree.
(b) **Collaboration analysis** – qualitative and quantitative analyses are conducted to identify and justify the possible collaborative solutions from the task model to further define the details of collaboration. Collaboration roles are assigned to all task components with color task role indicators to improve the collaboration relationship representation of the task model.
(c) **Design enhancements in operation process design** – improvements in term of (a) group repetitive steps, (b) add interval step, (c) preserve collaboration, and (d) assembly sequence changes are achieved in the task modeling of cable harness assembly.
(d) **Design extensions in human skill analysis, safety assessment and operation support** – extensions in facilitate human cognitive and motor skills studies, conduct risk assessment for safety design and assist information support development.
(e) **Practical implementation in prototype system** – model validation was conducted successfully with an actual cable harness assembly operation and positive results were obtained in the operation performance evaluation.

This work might have completed a preliminary modeling framework for human-robot collaboration planning in manufacturing systems based on task analysis approach. More research studies and developments are needed to further enhance the work:

(a) Quantitative studies should be conducted to investigate the effectiveness of the human-robot collaboration planning.
(b) Comparison study with other production operations to investigate the modeling capability of the proposed framework.
(c) The temporal aspects in collaboration should be taken into consideration to develop a more realistic representation for asynchronous human-robot operations.

7. References

Agah, A. (2000). Human interactions with intelligent systems: research taxonomy, *Computers & Electrical Engineering*, Vol. 27, No. 1, pp. 71-107

Annett J. & Duncan, K. D. (1967). Task analysis and training design, *Occupational Psychology*, Vol. 41, pp. 211-221

ANSI/RIA R15.06 (1999). Industrial Robots and Robot Systems – Safety Requirements, *American National Standards Institute / Robotic Industries Association*

Barnes, C. J.; Dalgleish, G. F.; Jared, G. E. M.; Swift, K. G. & Tate, S. J. (1997). Assembly sequence structures in design for assembly, *Proceedings of the IEEE International Symposium on Assembly and Task Planning*, pp. 164-169

Bischoff, R. (2001). System Reliability and Safety Concepts of the Humanoid Service Robot HERMES, *Proceedings of the First IARP/IEEE-RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments*

Colgate, J. E.; Wannasuphoprasit, W. & Peshkin, M. A. (1996). Cobots: Robots for Collaboration with Human Operators, *Proceedings of the International Mechanical Engineering Congress and Exhibition*, pp. 433-439

Duan, F. (2009). *Assembly Skill Transfer System for Cell Production*, PhD thesis, University of Tokyo, Japan

Duan, F.; Morioka, M.; Tan, J. T. C. & Arai, T. (2008). Multi-modal Assembly-Support System for Cell Production, *International Journal of Automation Technology*, Vol. 2, No. 5, pp. 384-389

Gottschlich, S.; Ramos, C. & Lyons, D. (2002). Assembly and task planning: a taxonomy, *IEEE Robotics & Automation Magazine*, Vol. 1, No. 3, pp. 4-12

Helms, E.; Schraft, R. D. & Hagele, M. (2002). rob@work: Robot assistant in industrial environments, *Proceedings of 11th IEEE International Workshop on Robot and Human Interactive Communication*, pp. 399-404

Harker, P. T. & Vargas, L. G. (1987). The theory of ratio scale estimation: Saaty's analytic hierarchy process, *Management Science*, Vol. 33, No. 11, pp. 1383-1403

Hodgkinson, G. P. & Crawshaw, C. M. (1985). Hierarchical task analysis for ergonomics research, *Applied Ergonomics*, Vol. 16, No. 4, pp. 289-299

Hollnagel, E. (2006). Chapter 14 task analysis: why, what, and how, In: *Handbook of Human Factors and Ergonomics*, Third Edition, pp. 373-383, John Wiley & Sons, Inc.

Iossifidis, I.; Bruckhoff, C.; Theis, C.; Grote, C.; Faubel, C.; Schoner, G. (2002). CORA: An Anthropomorphic Robot Assistant for Human Environment, *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, pp. 392-398

Isa, K. & Tsuru, T. (2002). Cell production and workplace innovation in Japan: towards a new model for Japanese manufacturing? *Industrial Relations*, Vol. 41, No. 4, pp. 548-578

Killich, S.; Luczak, H.; Schlick, C.; Weissenbach, M.; Wiendemaier, S. & Ziegler, J. (1999). Task modeling for cooperative work, *Behaviour and Information Technology*, Vol. 18, No. 5, pp. 325-338

Kirwan, B. & Ainsworth, L. K. (1992). *A Guide to Task Analysis*, Taylor & Francis, London

Kosuge, K.; Hashimoto, S. & Yoshida, H. (1998). Human-robots collaboration system for flexible object handling, *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1841-1846

Kosuge, K.; Yoshida, H.; Taguchi, D.; Fukuda, T.; Hariki, K.; Kanitani, K. & Sakai, M. (1994). Robot-human collaboration for new robotic applications, *Proceedings of the 20th International Conference on Industrial Electronics, Control, and Instrumentation*, Vol. 2, pp. 713-718

Laengle, T.; Hoeniger, T. & Zhu, L. (1997). Cooperation in Human-Robot-Teams, *Proceedings of the IEEE International Symposium on Industrial Electronics*, pp. 1297-1301

Lane, R.; Stanton, N. A. & Harrison, D. (2006). Applying hierarchical task analysis to medication administration errors, *Applied Ergonomics*, Vol. 37, No. 5, pp. 669-679

Mills, S. (2007). Contextualizing design: aspects of using usability context analysis and hierarchical task analysis for software design, *Behaviour & Information Technology*, Vol. 26, pp. 499-506

Mizoguchi, F.; Hiraishi, H. & Nishiyama, H. (1999). Human-robot collaboration in the smart office environment, *Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 79-84

Oborski, P. (2004). Man-machine interactions in advanced manufacturing systems, *International Journal of Advanced Manufacturing Technology*, Vol. 23, No. 3-4, pp. 227-232

Pongthanya, N.; Duan, F.; Tan, J. T. C.; Watanabe, K.; Zhang, Y.; Sugi, M.; Yokoi, H. & Arai, T. (2008). Evaluating assembly instruction methods in cell production system by physiological parameters and subjective indices, *Proceedings of The 41st CIRP Conference on Manufacturing Systems*, pp. 199-202

Richardson, J.; Ormerod, T. C. & Shepherd, A. (1998). The role of task analysis in capturing requirements for interface design, *Interacting with Computers*, Vol. 9, No. 4, pp. 367-384

Rudas, I. J. & Horvath, L. (1996). Modeling man-machine processes in CAD/CAM and flexible manufacturing systems, *Proceedings of the 22nd IEEE International Conference on Industrial Electronics, Control, and Instrumentation*, Vol. 3, pp. 1484-1489

Saaty, T. L. (2008). Decision making with the analytic hierarchy process, *International Journal of Services Sciences*, Vol. 1, No. 1, pp. 83-98

Saaty, T. L. (1994). How to make a decision: The analytic hierarchy process, *Interfaces*, Vol. 24, No. 6, pp. 19-43

Sarker, S. K.; Chang, A.; Albrani, T. & Vincent, C. (2008). Constructing hierarchical task analysis in surgery, *Surgical Endoscopy*, Vol. 22, No. 1, pp. 107-111

Shepherd, A. (1998). HTA as a framework for task analysis, *Ergonomics*, Vol. 41, No. 11, pp. 1537-1552

Stahre, J. (1995). Evaluating human/machine interaction problems in advanced manufacturing, *Computer-Integrated Manufacturing Systems*, Vol. 8, No. 2, pp. 143-150

Stanton, N. A. (2006). Hierarchical task analysis: developments, applications, and extensions. Applied Ergonomics, Vol. 37, No. 1, pp. 55-79, 2006

Stopp, A.; Baldauf, T.; Hantsche, R.; Horstmann, S.; Kristensen, S.; Lohnert, F.; Priem, C.; Ruscher, B. (2002). The Manufacturing Assistant: Safe, Interactive Teaching of Operation Sequences, *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, pp. 386-391

Tan, J. T. C.; Duan, F.; Zhang, Y.; Kato, R. & Arai, T. (2009a). Safety Design and Development of Human-Robot Collaboration in Cellular Manufacturing, *Proceedings of the 5th Annual IEEE Int. Conf. on Automation Science and Engineering*, Bangalore, India, pp. 537-542

Tan, J. T. C.; Duan, F.; Zhang, Y.; Kato, R. & Arai, T. (2009b). Task Modeling Approach to Enhance Man-Machine Collaboration in Cell Production, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 152-157

Tan, J. T. C.; Duan, F.; Zhang, Y. & Arai, T. (2008a). Extending Task Analysis in HTA to Model Man-Machine Collaboration in Cell Production, *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pp. 542-547

Tan, J. T. C.; Duan, F.; Zhang, Y. & Arai, T. (2008b). Task Decomposition of Cell Production Assembly Operation for Man-Machine Collaboration by HTA, *Proceedings of the IEEE International Conference on Automation and Logistics*, pp. 1066-1071

Collaborative rules operating manipulators

José Martins Junior[1], Luiz Camolesi Jr[2] and
Glauco Augusto de Paula Caurin[3]
[1] *Piracicaba School of Engineering*
[2]*University of Campinas*, [3]*University of São Paulo*
Brazil

1. Introduction

Collaboration among robots and human beings has inspired researchers and novelists since a long time ago. Apropos, the word "robot" first appeared in a theatre play ("R.U.R.", Karel Capek, 1921) referring to an automata character, a slave humanoid. Important advances for control strategies were presented by researchers, applied to service robots, toys and automata vehicles, concerning the interaction with human beings.
Over time, manipulator robots were massively used on industrial plants, performing predefined and repetitive tasks. Modern applications for manipulators, involving two or more robots on cooperative tasks, are now arising in industry. Most of the scientific publications on this area present solutions for some aspects involving humans, mainly related to the safety in robots' workspaces, and the flexibility to fast operate and reconfigure them. However, the way to operate manipulators remains rigidly based on imperative programming, through a HRI (Human-Robot Interface). On the other hand, a new approach proposed by (Brooks, 1986), based on behaviours, allows the definition of reactive models of control applied to mobile robots. The main limitation of this approach is its strictly reactive behaviour, i.e. all knowledge the robot will learn about the environment is unpredictable.
Current trends in several research areas are pointing to a possible occurrence of a new singularity, when the mankind will experiment the knowledge disembodiment, i.e. the human knowledge will be retrieved from brain, including its consciousness, and transferred to another place, or machine (Vinge, 2008). Psychologists (Pinker, 1999) defend that the mental states, as well as deliberations and emotions, can be represented by means of symbols of a mental language, known as "Mentalese". The free representation of signals and symbols for all mental states and their causalities is practically impossible, considering the current state of the art in technology. However, if conceived to specific domains, this can be fetched. Rules and policies for collaborative environments consist of well formed sentences, which describe states, causal relationships and their effects, applied to collaboration among human beings. These rules and policies have been used for several situations, involving computer supported cooperative work (CSCW).
This chapter presents and discusses the application of symbolic rules to coordinate collaborative environments with manipulators and humans. It also demonstrates how to express a set of collaborative rules, with common effects for machines and humans. We

know that the elephants don’t play chess (Brooks, 1990), but at the end of this chapter was presented our “robotic elephant” which plays a Tic Tac Toe game.

2. Robot control strategies

The traditional approach for robots control is feedback based and hierarchically subordinated to a trajectory generator that is responsible for mapping tasks into sets of movement sequences (rotations and translations). These sequences are obtained by combining the individual motions defined by links or mobile parts of a robot. The generated information is presented to the control modules, which are responsible for motors actuation.
A different strategy was presented by (Brooks, 1986). It describes a multilayer architecture with several levels of abstraction, allowing reactive behaviours (Nwana, 1996) for mobile robots. Each layer consists of a distinct level of competence, which may be activated. Thus, a layer can modify the resultant output, by including its component on the signal, and also inhibiting the signal produced by the lower layers.

2.1 Manipulators and the traditional approach

The large majority of industrial robot manipulators, available in the market today, use the control architecture originally proposed by Engelberger/Devol and their UNIMATE robot more than 30 years ago. The movements are decomposed by the user during the programming task into a sequence of primitive movements, i.e. point to point, straight line, circle. Normally the robot controller programming interfaces are implemented in an interpretative form. All the movements are related to a "tool center point - TCP". Time intervals or velocity to be reached during the execution of each primitive movement is also user defined using the programming interface.
A strong point of this strategy is the capability to generate complex behaviour and movements independently on which kind of end-effector the manipulator is carrying on. A weakness of this approach remains in the ability of such equipment to interact with a non-static environment, as for example, in assembly tasks. Each primitive of an individual movement is further transformed into coordinates for the joints using inverse kinematic calculation. The coordinates and its derivatives are finally transmitted to the robot controllers, for each single joint, as a function of time.

2.2 Cooperative robots, humanoids and the behavioural-based approach

Cooperative applications research for multiple robots sprung in the last decade (Parker, 2003). (Cao et al, 1997) state that cooperative behaviour can be observed on more complex animals (vertebrates, for instance), including human beings. Such behaviour has social motivation, demanding each isolated participant to feel the need or desire to cooperate. A system with multiple robots can present cooperative behaviour if, when performing some task with any cooperative mechanism, the increase on the efficiency of the whole system emerge.
Several architectures were proposed to solve the distributed control problem for mobile multi-robots, but this subject is out of the main focus of this chapter. A good description about this research area can be found in (Parker, 2003). On the other hand, there are few

published works (Lau & Ng, 2006) that discuss solutions for control problems for robotic manipulators using distributed strategies.

Reactive behaviour models

Behaviour may be defined (Bishop & Potter 2004) as an observable and repeated pattern in the relationships among spatio-temporal events associated with an agent and its environment. Behaviour based robots use the information they gather from the environment through the sensors to react to specific situations. The internal representations of environment are extremely limited when not completely inexistent. Isolated models of simple behaviours are responsible to respond to specific sensor signal conditions. The overall robot behaviour results from the output combination of each model. It is exactly the synthesis of coherent reaction, i.e. emerging intelligence as the result of the fusion of each behaviour model constitutes an open challenging task.

Reactive models are very important for strategies involving robots learning, especially mobiles ones, because they allow to assume the world as its own best model (Brooks, 1990). This feature is primordial in situations involving adaptation for robots' behaviours acting in unknown environments, like other planets.

Deliberative behaviour models

In deliberative control, the robot takes all of the available sensory information, and compares this information with its internally stored knowledge. Therefore in this approach a complete representation of the environment is stimulated using all available internal robot computer resources. To accomplish its task, the robot must further plan its future actions. This requires the robot program to look ahead. As a consequence, the control structure must provide multi task real time capabilities allowing the robot to act strategically.

If we pretend to see robots replicating behaviours based on the knowledge previously obtained by a human (in opposite to a non-deterministic learning by observing the environment), this knowledge must be formally expressed and converted into deliberative actions, according to this interpretation. Currently, two different approaches may apply: the neuronal model of the brain and the symbolic model of the mind.

The first (connectionist) aims to reproduce in computers the basic functions of the brain inspired by its topology. The main limitations for this approach refer to the enormous quantity of neurons and synaptic connections, and also the plasticity of neural networks created by the brain.

The symbolic model is presented and discussed at session 3.1 of this chapter.

2.3 Distributed and modular strategies for control architectures

Modular Robotics offers an answer for various complex tasks. Instead of designing a new and different mechanical robot for each task, simple module reconfiguration when connect in a suitable form may accomplish complicated things and meet the demands of different tasks or different working environments.

Each module is improved with individual capabilities for processing, sensing, communicating and actuating. The overall functionality of a modular system is only achieved when several modules are connected as a unique robot, i.e. a single module presents low utility.

Similarly, a manipulator robot can be decomposed and its parts individually analyzed. These parts present individual capabilities, like modules. Thus, the robot can be classified as an n-modular system, where n is the number of different types of modules.

2.4 Supporting collaborative behaviours

A multilayer control model, adapted from (Brooks, 1986), was proposed in (Martins Jr et al, 2008). This new approach includes cooperative and collaborative behaviours, and was designed to operate on distributed systems, defining different contexts – local and global – as shown in Figure 1.

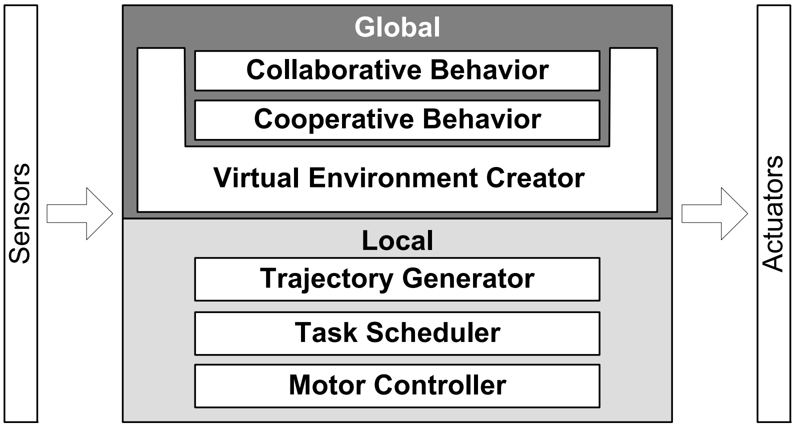


Fig. 1. Multilayer control model

Distinct parameters of criticality and strictness for agents operating on each of the two contexts (local and global) can be individually treated into its respective layer and provide means to define its coupling degree to the target (robot). Figure 2 shows the appropriated allocation of the layers on a distributed environment.

The local functions describe processes that are highly rigorous for execution time, but demand a small amount of resources (storage and processing power). They can be classified as local agents, tightly coupled to the target.

On the other hand, in the global context, the processes are less rigorous with respect to performance time, but need larger amount of resources. These features indicate that the designed agents must not be embedded into the target, but executed on loosely coupled remote computers. Local agents interact directly with the target using communication boards connected to actuators and sensors.

Each distinct part (link) of the robot, including its sensors, motors and mechanisms can be individually considered as different modules. Thus, the movements' composition for each module can be resolved on a higher level, as a cooperative task. This is one of the advantages provided by the architecture.

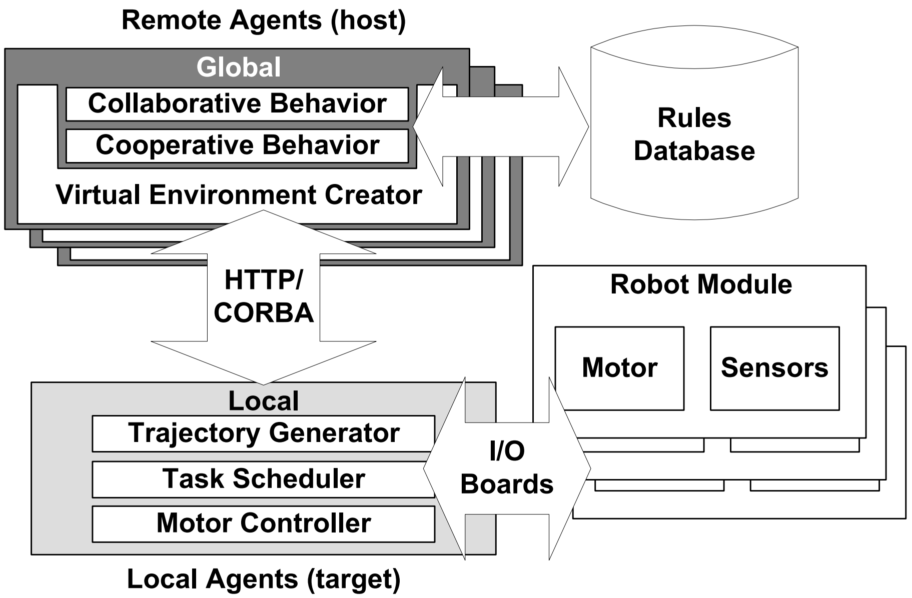


Fig. 2. Distributed architecture

Remote agents can be assigned for each module and interact with local agents to provide appropriate behaviour for the global model of the environment. This interaction can be done through a local network, and supported by data communication protocols (such as CORBA, or HTTP). The cooperative behaviour can be achieved by joint deliberation involving remote agents, based on cooperation rules and on environment model. Cooperation is frequently associated to a specific task execution.
The collaborative behaviour is placed on top of the architecture. Collaboration can be observed when a robot interacts with a set of tasks contributing to achieve a common goal to other agents in the environment, including humans. In the same manner, this behaviour is deliberated from decisions taken by agents, by analyzing collaborative rules and checking the current state of the environment.
The guiding rules for cooperation and collaboration are stored in a rules database, and can be accessed by global agents, at its respective actuation level.
A brief description of each architecture layer is presented on the following subsections.

Motor controller
Motor controller describes the bottommost layer, highly dependent and coupled to the hardware. Software agents developed to this level are locally executed (embedded) on the target. Real time requirements are highly rigorous, requiring performance time up to 1 millisecond.
The main function of this layer is to emit signals to the drivers that directly actuate on each of the motors. Data from sensors (encoders) attached to the motors are returned to the control mesh as feedback, to ensure correct performance of the actuators.

Task scheduler

Task scheduler plays an important role in the local context of the architecture, because it provides means to coordinate the processes that operate during each of the individual movement of the robot links. The execution of individual movements is part of the strategy defined by an upper layer (trajectory generator), allowing general repositioning of the robot in the environment.

Trajectory generator

Trajectory generator is the topmost layer in local context. It also imposes rigorous restrictions concerning real time, yet it allows larger deadlines, of around 10 milliseconds. Its main function is to define individual movements for each motor, by decomposing a desired trajectory between two points and sending it to the robot (inverse kinematics).

Virtual environment creator

The bottommost layer of global context is to implement the general model of the environment where the robot will be placed. At this level, geometric aspects are considered, allowing robot's workspace analysis to be performed. Visual and proximity sensors can provide data to be compared with the current state of the model, internally represented by a software agent. Global strategies to replace the robot in the environment can be defined at this level. These functions demand more computing power and, as compensation, allow flexibility to response times. The main advantage of an environment model which keeps close fidelity to the real world is the easiness to perform robot simulation on a virtual environment. In this manner, all the development and testing stages for high-level functionality, involving cooperative and collaborative behaviours, can be previously done on a simulated environment.

Cooperative behaviour

Cooperative behaviour describes relations among robots (or parts of them) and must be implemented by global agents, based on environment model and predefined rules (rules database). It is possible to notice behavioural capacities, both reactive and deliberative, of these agents. The reactive capacity is provided by environment analysis, which represents its internal model, differently from the deliberative, which results from decisions about cooperation rules.

Collaborative behaviour

In the same manner, collaborative behaviour for human interaction can be also provided. Previously defined collaborative rules are stored in a database and allow analysis by agents, by comparing to the current state of the environment model.

Rules database

The rules database is an important artifact of the whole architecture, and it was designed to store all cooperative and collaborative rules. These rules were defined using the M-Forum model (Camolesi Jr & Martins, 2006), and describe interaction policies by means of collaborative situations, involving different agents. M-Forum is presented in the next sessions.

3. Languages and rules

Languages define written or spoken symbols that are used for communication purposes. Written symbols can be jointly combined into words that, and depending on the context, provide the meaning of transmitted ideas. The sentences composition in a language is previously defined by a grammar. Sentences are constituted by a finite sequence of symbols from some finite alphabet (Slonneger & Kurtz, 1995).

The syntax of a programming language describes how the symbols may be combined to create well-formed sentences (or programs). The meaning, obtained by interpreting the words of a sentence, defines its semantics (originally conceived in “Mentalese”, the language of thought). Thus, intentions about a desired behaviour can be described by means of rules, using an interpretable language.

Restricted and non-ambiguous formal grammars were proposed to express programming languages for computers. BNF (Backus-Naur form) is a widely adopted formal model to specify grammars that describe terminal symbols of a valid alphabet, non-terminal symbols and production rules. The main benefit of using a BNF style language is the easy to implement programs that work as its lexical interpreters.

3.1 Symbolism and the mind-brain dilemma

Symbolism can be described as a movement that defends symbolic representation of mental activities, inspired by computer like way of operation. In this sense, a constructive approach is defined using a top-down strategy (Minsky, 1990): begin at the level of commonsense psychology and try to imagine processes that could simulate it. The central idea consists, assuming a greater challenge, to search for a solution by decomposing it into simpler parts. This refers to a reductionist method, a typical approach commonly applied in AI (Artificial Intelligence), and known as heuristic programming.

Two relevant aspects can be highlighted from the concept of the mind presented in (Pinker, 1999), which constitutes the foundation of computational theory of the mind. The first states that the mental computations are applied to information, and this can be expressed using an internal symbolic representation. The other refers to the functional composition of the mental modules, which perform the computations. Thus, no matter what kind of subject (physical) where mental computation is performed, the functionality of the modules that compose the mind and the symbols are submitted to it. As a consequence, beliefs and desires can be seen as information, physically embodied as configurations and symbols.

In the last decades, with the advances in AI research, a new approach for philosophy of the mind – not dualist either materialist – has emerged, the functionalism. Functionalism introduces the concept of causal role, in which a mental state can be described by their causal relations with other mental states. Functionalism is based on the distinction established by computer science about hardware (physical components) and software (programs). From this point of view of psychology, a system can describe a human being or a machine, and its basic constitution (neurons or electronics) is not what really matters, but how parts are organized (Fodor, 1981). Thus, functionalism does not rule out the possibility of a mechanical or electronic system having mental states and processes.

The central subject of this chapter is related to collaboration among robots (machines) and human beings. In this sense, we have adopted a top-down model, where the behavioural rules, with common sense for both types of actors, have been stated using a formal language (constituted by symbols). Then, the rules were implemented and their functionality observed during the coordination of a collaborative activity (more specifically, a game).

3.2 Behavioural rules, policies and collaborative environments

Interaction Policies are norms for the interactions in an environment; those can be established by logic grouping of rules with well defined goals or objectives. In the definition of a Collaboration and Control Policy (CCP) model for human-robot interaction, a policy must observe the relationship among robotic and human agents in a same environment, regarding collaborative task performance.

The research (Camolesi Jr & Martins, 2005) has achieved excellent progress for structure and ontology definition. However it still has a lot to advance on applications, such as robots control. Towards the approach of these questions, the M-Forum model supports collaborative interactions modelling through the definition of rules by providing support to five dimensions: actor; activity; object; time and space. A comparison between M-Forum and the other models for rules (Tonti et al, 2003) is presented in Table 1.

	Kaos	**Rei**	**Ponder**	**M-Forum**
Ontology based	Yes	Yes	No	Yes
Specification language	DAML/OWL	Prolog based	Ponder Language	L-Forum
Tool for specification policy	KPAT – Graphical Editor	No (GUI under development)	Graphical Editor	No (GUI under development)
Reasoning support	Java Theorem Prover	Prolog engine; Event-condition-action model.	Event calculus	Activity theorem; Deontic theorem; Event-condition-action model.
Enforcement mechanisms	Policy automation being explored for the next version	Action execution is outside the Rei engine	Java interfaces for enforcement agents	Rule execution is outside the engine
Flexibility	Ontology can be extended with domain dependent descriptions of local entities	Ontology can be extended with domain dependent descriptions of local entities	Management domain as a structuring technique for partitioning complex object	Ontology can be extended with domain dependent descriptions of local entities
Elements represented	Actors, actions, groups, places	Subject, activity, object	Subject, activity, object, domain	Actor, activity, object, time, space, association, domain, composition and generalization abstractions

Table 1. Comparison between M-Forum and other models for rules

3.3 The M-Forum model

In M-Forum (Camolesi Jr & Martins, 2006), the Actor dimension allows the representation of an agent in a collaborative environment through activity rights, prohibitions and obligations. The actors of a collaborative environment can be classified in human or not-human. Every human actor has an identifier (*Ach_id*), a current state (*AchState*) and a set of attributes (*Ach_AttS*). Given *qh* as the number of human actors at the environment and *qs*, the number of not-humans, the formal statements are:

$$\boxed{\begin{array}{l}
AchS = \{Ach_1, Ach_2, \ldots, Ach_{qh}\}\ ,\ AcsS = \{Acs_1, Acs_2, \ldots, Acs_{qs}\} \\
AchS \neq \varnothing \vee AcsS \neq \varnothing \\
AchS \cap AcsS = \varnothing \\
AcSS = AchS \cup AcsS \\
Ach_i = (Ach\_id_i\ , AchState_i\ , Ach\_AttS_i\ ) \\
Acs_i = (Acs\_id_i\ , AcsState_i\ , Acs\_AttS_i\ )
\end{array}}$$

Actors are responsible for the execution of individual or collective activities, thus being able to reach objects, an actor or actors group.
Activity is an execution unit that can be carried through by an actor or group. Normally, activities involve the manipulation or transformation of an object. Activities must be defined using Activity Operators and logic Operators representing rights, prohibitions and obligations. Activity Operators are required to specify the interaction between actors and objects. Activities have identifiers (*At_id*), a state (*AtState*), an activities subset (*At_S*), a set of operations (*OpS*) and a set of attributes (*At_AttributeS*). Given *qa* as the number of activities in an environment:

$$\boxed{\begin{array}{l}
AtSS = \{At_1\ , At_2\ , \ldots, At_{qa}\} \\
At_i = (At\_id_i\ , AtState_i, AtS_i\ , OpS_i\ , At\_AttributeS_i\ ) \\
AtS_i \subseteq AtSS \\
OpS_i \subseteq OpS
\end{array}}$$

Object is any element that can be used in actions on objects or actors. An object represents the structural characteristics and the behaviour of reality. Activities can be carried through in objects to modify its characteristics. An object modelling in such a way establishes uniformity of vision and treatment, useful for collaborative environment projects. An object may be composed by others objects (*CompObS*) and has an identifier (*Ob_id*), a state (*ObState*) and a set of attributes (*Ob_AttS*). Activities and Operations may be performed on Objects that allows its state or attributes changing.

$$\boxed{\begin{array}{l}
ObSS = \{Ob_1\ , Ob_2\ , \ldots, Ob_{qo}\ \} \vee ObSS \neq \varnothing \\
Ob_i = (Ob\_id_i\ , CompObS_i\ , ObState_i\ , Ob\_AttS_i\ ) \\
CompObS_i \subseteq ObSS
\end{array}}$$

Spaces are localization areas of actors or objects and the specific areas used for activities. Like other elements presented in this section, the spaces are essential for modelling a collaborative environment.
On the collaborative interaction, elements of the dimension space must be defined using the Space Operator (*SpOp*) to specify the position or the size of actors and objects in collaborative environments. The space element has an identifier (*Sp_id*), a state (*SpState*) and a set of attributes (*SpAttS*). If *qe* is the number of spaces into an environment:

$Sp_i = (Sp\_id_i, SpState_i, SpAttS_i)$
$SpSS = \{Sp_1, Sp_2, \ldots, Sp_{qe}\}$
$SpOp \in So = \{$ <,<=,>,>=,=,<>,==(attribution), not equal, inside, outside, intersect, meet, overlap, north, south, east, west $\}$

In time modelling, Duration, Date and Occurrence have basic semantic for temporal references establishment. These semantics are used to define a logical action with duration, occurrence date or occurrence interval of activities defined on interactions between actors and objects.
The formal basis for temporal elements describes the natural set of numbers (*N*), and representations for years (*Ty*), months (*Tm*), days (*Td*), hours (*Th*), minutes (*Tmi*) and seconds (*Ts*), for a Moment or Interval. Enumerated sets of relative values (*Tmr, Tdr, Thr, Tmir, Tsr*) are used to represent dates for a specific calendar. Given *qt* as the number of time moments or intervals occurring on an environment:

$Ty, Tm, Td, Th, Tmi, Ts \in N$
$Tmr \in \{1, 2, 3, 4, \ldots, 12\}$
$Tdr \in \{1, 2, 3, 4, 5, \ldots, 31\}$
$Thr \in \{0, 1, 2, 3, \ldots, 23\}$
$Tmir \in \{0, 1, 2, 3, \ldots, 59\}$
$Tsr \in \{0, 1, 2, 3, \ldots, 59\}$
$Time = \{Ti\_id, (Ty, Tm, Td, Th, Tmi, Ts)\}$
$Datetime = \{Ti\_id, (Ty, Tmr, Tdr, Thr, Tmir, Tsr)\}$
$Tb, Te \in Datetime$, $Interval = \{Ti\_id, (Tb, Te)\}$
$TiSS = Time \cup Datetime \cup Interval$
$TiSS = \{Ti_1, Ti_2, \ldots, Ti_{qt}\}$
$TiOp \in To = \{$ <,<=,>,>=,=,<>,== (attribution), precedes, succeeds, directly precedes, directly succeeds, overlaps, is overlapped by, occurs during, contains, starts, is stated with, finishes, is finished with, coincides with $\}$

The dimensional elements of a rule are defined in three contexts:
- **Applicability**: condition for the execution or activation of a rule and definition of the scenes (values of attributes or space aspects) in which it can be applied;
- **Execution**: a set of expressions that establishes the actions or conditions for the interactions between elements, being able to optionally involve transitory aspects of time and space;
- **Survivability**: it is an optional context specifying the other rules with the same applicability. Also the scenes can be defined (values of attributes or space aspects) to establish the instant at which the rule must be activated or deactivated.

3.4 The L-Forum syntax
L-Forum is a language developed to formalize the concepts specified by the M-Forum model. The language allows the definition of rules for an environment, increasing their precision and improving disambiguation for collaborative environment designers. Its

overall structure may be described by clauses, which are defined for three particular purposes:

- **Context:** this clause is composed by performing or activating parameters of a rule and comply with applicability conditions of the scenario (value of attributes, spatial or temporal aspects) for a rule adoption;
- **Definition (body):** it is composed by a set of expressions where actions or conditions are established for interacting elements and may involve transitory aspects of time and space. Rules and actions may be directly invocated at the body of a rule, which allows to compose the expressions;
- **Regime:** this is the scope of a rule, and refers to an optional set composed by interrelated rules having the same orientation to be performed or applied. Scenarios, involving a rule activation or deactivation, can be also described.

The main elements of L-Forum syntax are presented in Table 2.

3.5 Collaborative rules for human-robot interactions

At this point, we address to the problem of defining the collaborative rules among robots and humans. To illustrate it, a simple task was considered: a tool passing between the human and the robot.

For the robot to identify the different collaborative situations, involving the task, a visual code was established. If the human presents his(er) open hand over the common workspace (a table surface), the situation “tool passes from robot to the human” must be assumed. If the human presents his(er) hand holding the tool, the opposite situation must be considered. Summarizing, the dimensional elements to elaborate collaborative rules, are:

- **Actors:** robot and human. The robot is an actor composed by different links. The human being is also an actor established by the composition of single parts, detaching the hand;
- **Objects:** the tool, which will be collaboratively shared by the actors;
- **Space:** there are three involved spaces in the problem: the common workspace (table surface) where will be shared to pass the tool; the individual spaces, where the human and the robot stay. Each individual space is exclusive. Only the common area must be shared collaboratively by both actors;
- **Activity:** ordering and delivering are activities that may be realized by both actors (human and robot). The activities will be recognized by both actors analyzing the state their parts, i.e., the human's hand and the robot's gripper (end-effector). A hand or a gripper on “open” state means the tool ordering; a hand or a gripper holding the tool is associated to a tool delivering. Grasping and releasing are also related activities on the working process.

When modelling the collaborative actions, the human is the actor with primary actuation, and so, he (or she) establishes the frequency and sequence for actions. In this sense, considering the dimensional elements of the collaborative work scenery, previously presented, some rules to compose the Collaboration and Control Policy (CCP) are shown in Table 3.

<rule> ::=	´Rule´ <rule name> ´[´<status>´]´ ´{´ <context> ´Body ::´<definition> [<regime>] ´}´
<context> ::=	´Parameters: (´<parameters>´)´ [<applicability>]
<definition> ::=	<condition> | <action> | <rule call> [<definition>]
<regime> ::=	<survivability> [´Priorities:´ <priority>]
<parameters> ::=	(´any´ | ´all´ | <identifier>)´:´<element> [´,´ <parameters>]
<element> ::=	<actor> | <group> | <object> | <space> | <time> | <association> | <activity> | <operation>
<type> ::=	´actor´ | ´group´ | ´object´ | ´space´ | ´time´ | ´association´ | ´activity´ | ´operation´
<applicability> ::=	´Applicability::´ <condition expression>
<survivability> ::=	´Survivability::´ <condition expression>
<condition> ::=	´If ´ <condition expression> ´then {´ <definition> ´}´ [´else {´ <definition>´}´]
<action> ::=	´Action: (´ <actor> <normative operator> {<activity> (<actor> | <object>)} [<space attribution operation> <space>] [<time attribution operation> <time>]) [´);´ <action>] ´);´
<supreme action> ::=	<actor> <normative operator> <primitive operator> (<element>|<domain>|(´is part of´|´is a´) <element>) | <actor> <normative operator> <primitive group operator> <group> <element>)
<definition action> ::=	<actor> ´set´ <status>
<attribution action> ::=	<actor> ´attribute´ (<value>|<formula>| ((next | prior) (<value domain>|<domain name>)) <attribute>
<condition expression> ::=	´(´ (<attribute><attribute condition operator>(<value>| ([´all´|´any´] (<value domain>|<domain name>)) | (<condition expression> (and | or)) ´)´
<rule call> ::=	´Rule (´ <rule name> ´ (´<parameters>´)´ <normative operator> [´);´ <rule call>] ´);´
<priority> ::=	<name> [´,´ <priority>]
<group> ::=	<name>´:Group´
<actor> ::=	<name>´:Actor´
<activity> ::=	<name> ´:Activity´
<operation> ::=	<activity>´.´<name> ´:Operation´
<object> ::=	<name>´:Object´
<space> ::=	<name>´:Space´
<time> ::=	<name>´:Time´
<association> ::=	<element>´.´<name> [´.´<association>]´:Association´
<attribute> ::=	<element>´.´<name> [`:Attribute´]
<domain> ::=	<name> (<value domain> | <grouping>)
<value domain> ::=	´(´ (<numeric value> {´,´ <numeric value>}) | (<string> {´,´ <string>}) ´)´
<grouping> ::=	(<type> <name> <attribute condition operator> (<value>|([´all´| ´any´](<value domain>|<domain name>))) {(and| or) <grouping>}) | (<element> {´,´ <element>})
<element status> ::=	<element> <status attribution operator> <value>
<status> ::=	[´active´] | [´inactive´]
<primitive group operator> ::=	´insert´ | ´delete´ | ´update´
<primitive operator> ::=	´create´ | ´destroy´
<group element operator> ::=	´∈´ | ´∉´ |
<group group operator> ::=	´⊆´ | ´⊄´ | ´⊂´
<activity condition operator> ::=	[´not´] ´has´
<normative operator> ::=	´right´ | ´prohibition´ | ´obligation´ | ´dispensation´
<activity attribution operator> ::=	´receive´
<status attribution operator> ::=	´put on´ | ´move to´
<space attribution operator> ::=	´==´ | ´inside´ | ´outside´ | ´north´ | ´south´ | ´east´ | ´west´
<time attribution operator> ::=	´in´ | ´on´ | ´at´

Table 2. L-Forum syntax

Rule **Human Delivers Tool** [active] {	
Parameters::	(hu: human, ro: robot, too: tool, ts: table Surface)
Applicability::	(hu.hand not is open) and (ro.gripper is open) and (hu.hand not is on ts) and (ro.gripper not is on ts)
Body::	Action (hu obligate hand put on ts); Action (hu obligate release too on ts); Action (hu.hand obligate put of ts); Rule **Robot Moves to the work** (ro , ts) Action (ro obligate hold too); Action (ro.gripper obligate put of ts); }
Rule **Human Orders Tool** [active] {	
Parameters::	(hu: human, ro: robot, too: tool, ts: table Surface)
Applicability::	(hu.hand is open) and (ro.gripper not is open) and (hu.hand not is on ts) and (ro.gripper not is on ts)
Body::	Action (hu.hand put on ts) Action (hu.hand obligate put of ts) Rule **Robot Moves to the work** (ro , ts) Action (ro obligate release too on ts); Action (ro.gripper obligate put of ts); Action (hu.hand obligate put on ts) Action (hu obligate hold too); Action (hu.hand obligate put of ts) }
Rule **Robot Moves to the work** [active] {	
Parameters::	(ro: robot, ts: table Surface)
Applicability::	(hu.hand not is on ts)
Body::	Action (ro.vector_a prohibition move on ts) Action (ro.vector_b prohibition move on ts) Rule **Moving Vector_c** (ro, ts)
Survivability:: Priorities: **Human Delivers Tool , Human Orders Tool** }	
Rule **Moving Vector_C** [active] {	
Parameters::	(hu:human; ro: robot, ts: table Surface)
Applicability::	(hu.hand not is on ts)
Body::	Action (ro.vector_c obligate move to ts) }

Table 3. Rules for collaborative tool passing

4. Case study: Tic Tac Toe game

In this session we present an experimental case study, involving human and robot interaction faced as opponents in a board game, known as Tic Tac Toe. The game was chosen because its rules are very easy to learn, allowing it to be played by people with different levels of skill, from children to adult.

The decision was also influenced by another feature, that the game is played on a predefined board (field), facilitating the coordination of movements done by opponents (robot and human) into the common workspace.
This case study was proposed as a proof of concept for using collaborative rules to govern the interactions among different types of actors, a robot and a human. It was also important to demonstrate the need for a strategy definition when selecting rules in collaborative environments, in order to surpass the unpredictability of some human behaviour.

4.1 The game

The Tic Tac Toe is a two player game where the participants take turns drawing tokens (X or O) on a 3 x 3 grid. Winning the game involves a player placing three tokens in a row, column or diagonal. When the grid is completely full and no sequence of three equal tokens occur (row, column or diagonal), they got a draw.
Figure 3 shows a particular and possible situation during a Tic Tac Toe match. In this case, the player of X tokens won.

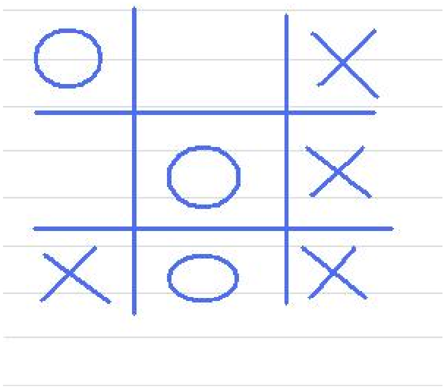


Fig. 3. A Tic Tac Toe situation

An expert performance for Tic Tac Toe game can be described as a set of few rules (Crowley & Siegler, 1993), as shown in Table 4. These rules are enough to describe every faced situation, during a Tic Tac Toe game, but often occurs that more than one rule can be applied, pointing to the need to define a criteria to select the most appropriated.

Adaptability for different levels of skill

Despite the rules simplicity, selecting them in order to make a move depends on several factors, like attention, knowledge and others. These factors are clearly influenced by the player's age, and thus, the system must be able to use appropriate strategies for different levels of skill presented by its opponent. Otherwise, a child will never be able to win, and could become bored.

Move type	**Conditions**	**Action**
Win	If there is a row, column or diagonal with two of my pieces and a blank space	Play the blank space
Block	If there is a row, column or diagonal with two of my opponent's pieces and a blank space	Play the blank space
Fork	If there are two intersecting rows, columns or diagonals with one of my pieces and two blanks AND If the intersecting space is empty	Move to the intersecting space
Block fork (1)	If there are two intersecting rows, columns or diagonals with one of my opponent's pieces and two blanks AND If the intersecting space is empty AND If there is an empty location that creates a two-in-a-row for me	Move to the location
Block fork (2)	If there are two intersecting rows, columns or diagonals with one of my opponent's pieces and two blanks AND If the intersecting space is empty AND If there is NOT an empty location that creates a two-in-a-row for me	Move to the intersecting space
Play center	If the center is blank	Play the center
Play opposite corner	If my opponent is in a corner AND If the opposite corner is empty	Play the opposite corner
Play empty corner	If there is an empty corner	Move to an empty corner
Play empty side	If there is an empty side	Move to an empty side

Table 4. Set of rules for expert performance on a Tic Tac Toe game

4.2 Defining the rules of the game

The next step consists on translating rules to L-Forum format. According to L-Forum syntax, described above, a rule may be stated using some elements of the language. A rule must have a unique name and declare its status. The parameters of a rule specify the involved elements, like actors, space and objects. The applicability refers to the conditions that cause a rule selectable. The body of a rule describes actions to be performed.

Two examples for rules mapping are presented in Table 5. The first refers to the "Play Center" rule for a Tic Tac Toe expert match, and may be applied when the center is empty and the game is not finished. The second implements a rule that may be selected in four situations, relating to each corner of the board.

```
Rule PlayCenter[active] {

Parameters::      (pl :player:actor; tboard :TicTacToeboard : space; tok :token:object, g
                  :game:object)
Applicability::   (tboard.center is empty) and (g not is finished)

Body::            Action (pl right play tok inside tboard.center); }
```

```
Rule PlayCorner[active] {

Parameters::      (pl :player:actor; tboard :TicTacToeboard : space; tok :token:object)
Applicability::   (g not is finished)

Body::            if (tboard.corner_high_right is empty)
                    then { Action (pl right play tok inside tboard.corner_high_right );
                  }
                  else {
                    if (tboard.corner_high_left is empty)
                      then { Action (pl right play tok inside tboard.corner_high_left);
                    }
                    else {
                      if (tboard.corner_low_right is empty)
                        then { Action (pl right play tok inside tboard.corner_low_right);
                      }
                      else {
                        Action (pl right play tok inside tboard.corner_low_left);
                      }
                    }
                  }
}
```

Table 5. “Play center” and “play corner” rules translated to L-Forum

4.3 The hardware infrastructure

An IBM 7545 SCARA robot retrofitted with open platform (Aroca et al, 2007), was used in this project. The target's hardware, assembled on a CompactPCI rack, contains the following components:

- **Boards**: Inova AMD K6, Acromag Carriers, National Instruments I/O;
- **Industry Packs (IP)**: Tews 48 Digital I/O, Tews IP Quadrature, Tews DAC.

Figure 4 presents the whole view of the architecture.

Since the main purpose of this project was not related to research accurate positioning and fine control, and aiming to simplify the robot operation, a strategy based on fixed points was adopted. All the nineteen positions, representing valid locations of the game, were predefined and marked into an 800x400 mm board, as shown in Figure 5. Ten of these positions (five at each side of the game field) were designed as pieces repositories.

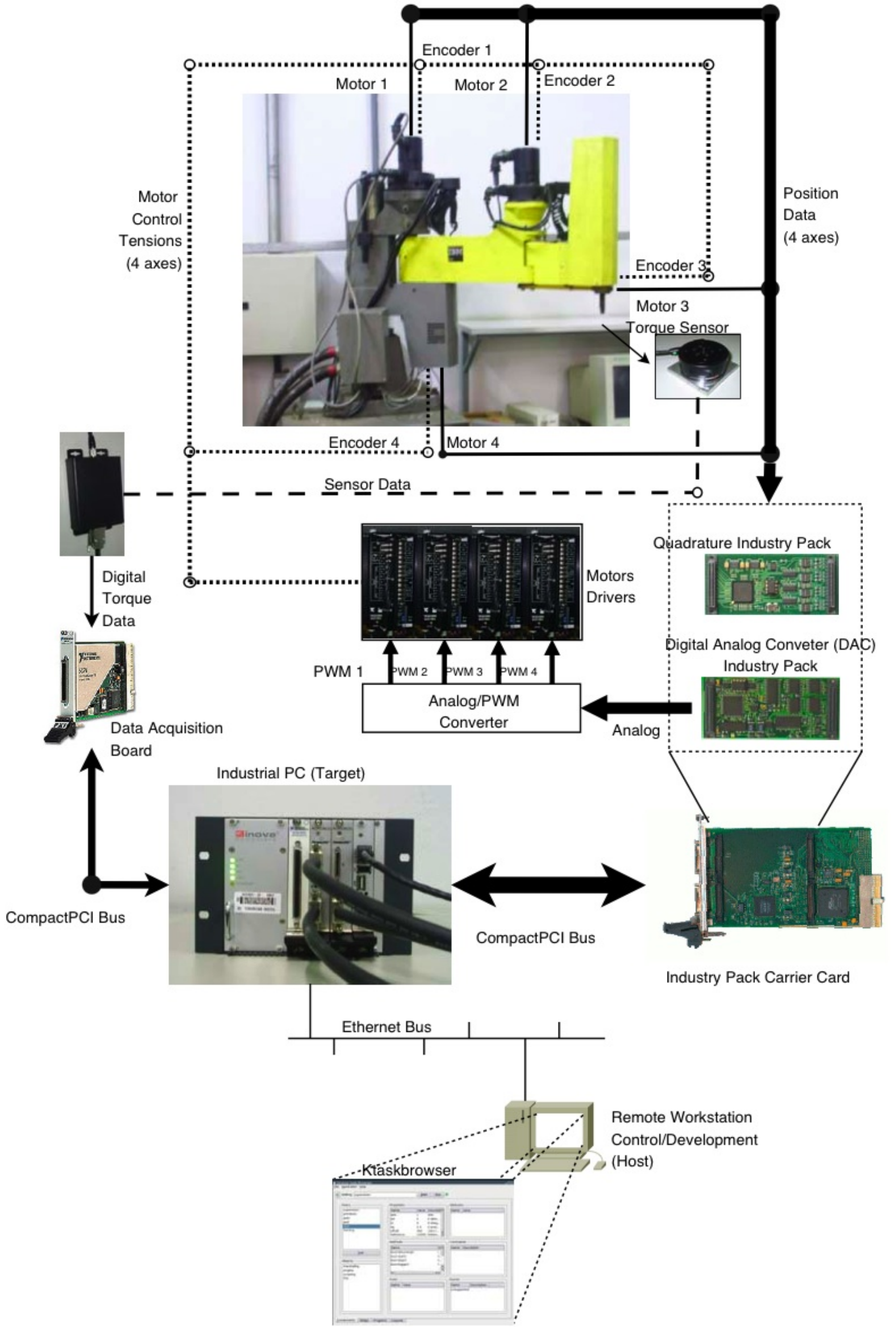


Fig. 4. The retrofitted IBM 7545 SCARA robot

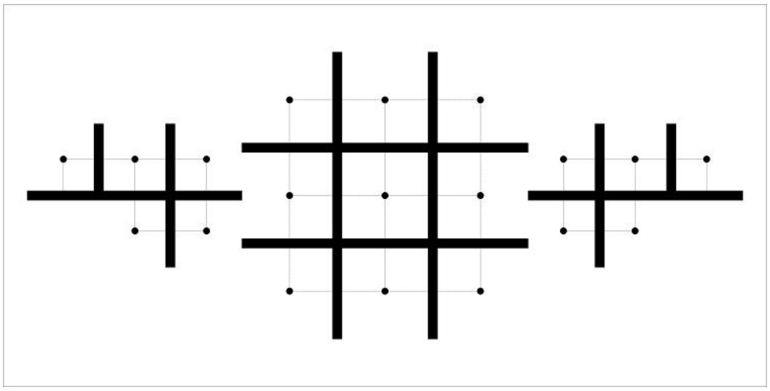


Fig. 5. Design of the Tic Tac Toe board

Nineteen reed switches (magnetic presence sensors) were fixed on the board, at each position for pieces locations (marked with small circles on the board). The reed switches are used to detect magnetic field generated by magnets, which in this case, were inserted in each piece of the Tic Tac Toe game, as shown in Figure 6.

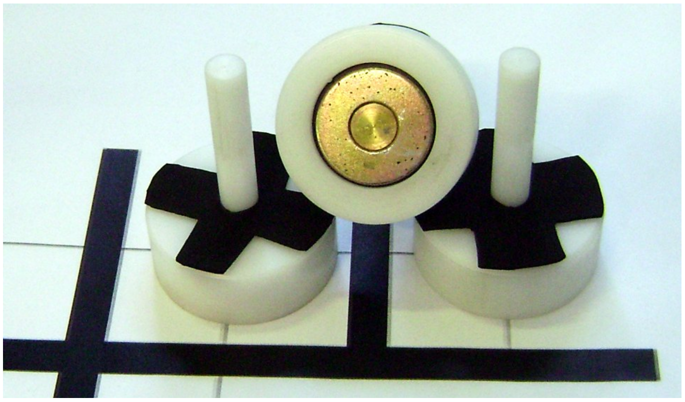


Fig. 6. Magnetic pieces

All sensors were connected to a board, with 32 digital inputs. This board multiplexes all digital entries through a parallel interface, connected to a computer port.
Since the main focus of this project was to present coherent means to allow robot and human collaboration, another subject comes up. Several researches in Robotics have pointed to the relevance of robot's appearance when interacting with humans. Human beings usually feel more comfortable to interact when the other subject looks familiar.
Considering that this robot must interact with human beings, including children, we decide to give it a playful look. The SCARA robot was dressed in an elephant costume, making it very fun and with a less formal aspect.
Figure 7 shows the robot during a Tic Tac Toe match. The picture also presents the board and the pieces over it.

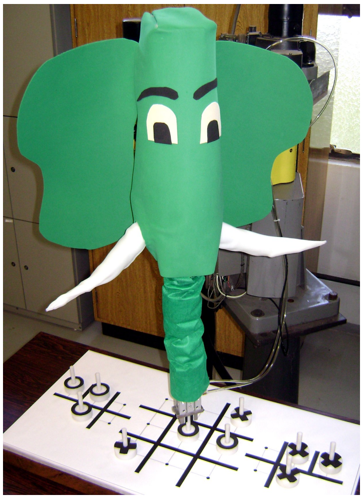


Fig. 7. Robot dressed as an elephant

4.4 Software design and implementation

The local software agents were designed to execute specific roles, implementing the bottommost levels of the distributed model, i.e., the trajectory planner, the task manager and the motor controller (Aroca et al, 2007).

The base system was developed using a real time interface to the Linux kernel – RTAI (RealTime Application Interface). Some of its features are the actuator for motors, a PID (Proportional Integral Derivative) controller, and sensor acquisition, through an industrial PC. The infrastructure also allows interacting with the robot, across the local network.

The overall solution was based on host-target model. A computer (host) was being used to develop and compile the software, before it was embedded in the industrial PC (target). Both computers run Linux operating system, but only the target's kernel was increased by the real time modules and the RTAI interface. Other facilities were also implemented, allowing more flexibility in robot reconfiguration and high level protocols for data communication (Tavares et al, 2007).

A three-dimensional HRI (Human-Robot Interface), called Scara3D, was presented in (Martins Jr et al, 2008). The interface was developed to perform tests for high-level layers integration into the architecture. The obtained results were satisfactory and proved the feasible implementation for the "Virtual Environment Creator" layer of the proposed model.

When designing the game we considered the interaction among a SCARA Robot and human beings, which can be classified as actors according M-Forum specifications. The SCARA Robot is composed by translational and rotational links and has a pneumatic gripper, while a Tic Tac Toe game contains a board and pieces (X and O); these individual parts are represented as objects in Forum model. These relationships are presented in a class diagram, as shown in Figure 8.

Current state of the real environment can be monitored by sensors integrated into the architecture, and so the virtual model can be updated, representing the interaction among several objects and actors within the game.

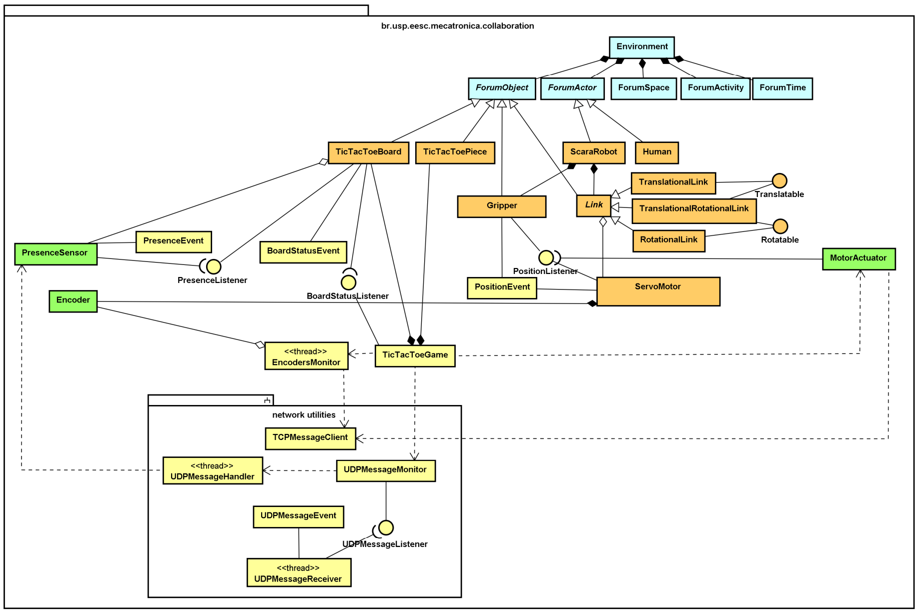


Fig. 8. Class diagram for Tic Tac Toe game

As described above, sensors were integrated to the system to detect the presence of pieces on nineteen different positions of the board. Another presence sensor (currently, a single switch) was also included to detect the presence of a human being into the shared workspace. The states of these twenty presence sensors are monitored by a client application that fires UDP (User Datagram Protocol) messages into the local network. Thus, using an UDP server (*UDPMessageReceiver*), the system allows asynchronous messages reading and performing event passing through appropriated listener implementations.
The current positions for servomotors were obtained by the system using an encoder monitor, which submits TCP (Transmission Control Protocol) requisitions to the target that controls the robot. TCP messages are also sent to the target by *MotorActuator* to reposition the servomotors, according the current states of their virtual representations by means of *ServoMotor* class instances.
As mentioned, listeners were used to provide events communication about states changing across objects into the virtual environment representation. Every change among virtual and real environments is communicated using TCP or UDP messages, allowing the distribution of the system components and the integration between high and low-level layers of the architecture.

5. Conclusion

In this chapter was presented a new architecture for robot control, which provides layers including deliberative behaviour on robot operation. The other features of the proposed model refer to the explicit definition of local and global contexts and its operating support for distributed environments.

The collaboration among robots and human beings was described using a symbolic representation, through a formal model of rules. This approach was successfully experimented in restricted situations, describing human-robot interactions. An experimental case study was also presented for this purpose, involving a collaborative game among a manipulator and humans.

Future research about this subject can be applied evolving the model to support representations of other mental states and allowing the extraction of rules from knowledge databases. It is also encouraged the use of the model for other situations, including collaboration among other subjects (mobile robots or other machines), as uncovered by this chapter.

6. Acknowledgment

The publication fee of this chapter was supported by Municipal Education Foundation of Piracicaba (FUMEP – Fundação Municipal de Ensino de Piracicaba).

The authors would like to thank researchers of Mechatronics Laboratory at São Carlos School of Engineering (University of São Paulo) by all help with the experiments described in this chapter.

7. References

Aroca, R.; Tavares, D.M. & Caurin, G.A.P. (2007). Scara Robot Controller Using Real Time Linux. *Proceedings of International Conference on Advanced Intelligent Mechatronics*, pp. 1-6, ISBN 978-1-4244-1264-8, Zürich – Switzerland, Sep 2007, IEEE, New York.

Bishop, J. N.; Potter, W.D.; (2004). Towards Developing Behavior Based Control Architectures for Mobile Robots Using Simulated Behaviors. *Proceeding of the International Conference on Artificial Intelligence (ICAI'04)*, Las Vegas, Nevada.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, Vol. 2, Issue 1, Mar 1986, pp. 14-23, ISSN 0882-4967.

Brooks, R. A. (1990). Elephants don't play chess. *Robotics and Autonomous Systems,* Vol. 6, Issue 1, Jun 1990, pp. 3-15, ISSN 0921-8890.

Camolesi Jr, L. & Martins, L. E. G. (2006). A Model for Interaction Rules to Define Governance Policies in Collaborative Environments. In: *Lecture Notes in Computer Science*, Vol. 3865, Shen, W.; Chao, K.-M.; Lin, Z.; Barthès, J.-P.A.; James, A. (Eds.), pp. 11-20, Springer Berlin, ISBN 978-3-540-32969-5, Heidelberg.

Camolesi Jr, L. & Martins, L.E.G. (2005). Specifying Powerful Rules to Govern Collaborative Environments. *Proceedings of 9th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2005)*, pp. 810-815, ISBN 1-84600-002-5, Coventry – UK, May 2005, IEEE, New York.

Cao, Y. U.; Fukunaga, A. S. & Kahng, A. B. (1997). Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, Vol. 4, No. 1, Mar 1997, pp. 7-27, ISSN 1573-7527.

Crowley, K. & Siegler, R. S. (1993). Flexible strategy use in young children's tic-tac-toe. *Cognitive Science: A Multidisciplinary Journal*, Vol. 17, Issue 4, October-December 1993, pp. 531-561, ISSN 1551-6709.

Fodor, J. A. (1981). The Mind-Body Problem. *Scientific American*, Vol. 244, No. 1, Jan 1981, pp. 114-123.

Lau, H. Y. K. & Ng, A. K. S. (2006). Immunology-based Motion Control for Modular Hyper-redundant Manipulators. *Proceedings of the 16th IFAC World Congress*, ISBN 978-0-08-045108-4, Prague, Jul 2005, Elsevier, New York.

Martins Jr, J.; Camolesi Jr, L. & Caurin, G. A. P. (2008). Scara3D: 3-Dimensional HRI integrated to a distributed control architecture for remote and cooperative actuation, *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC 2008)*, pp. 1597-1601, ISBN 978-1-59593-753-7, Fortaleza - Ceara - Brazil, Mar 2008, ACM, New York.

Minsky, M. (1990). Logical vs. analogical or symbolic vs. connectionist or neat vs. scruffy. In: *Artificial Intelligence at MIT, Expanding Frontiers*, Vol. 1, P. H. Winston & S. A. Shellard, pp. 218-243, MIT Press, ISBN 978-0262231541, Cambridge.

Nwana, H. S. (1996). Software Agents: An Overview. *Knowledge Engineering Review*, Vol. 11, Issue 3, Sep 1996, pp. 205-244.

Parker, L. E. (2003). Current research in multirobot systems. *Artificial Life and Robotics*, Vol. 7, No. 1-2, Mar 2003, pp. 1-5, ISSN 1614-7456.

Pinker, S. (1999). *How The Mind Works*, Penguin UK, ISBN 9780140244915, London.

Slonneger, K. & Kurtz, B. L. (1995). *Formal Syntax and Semantics of Programming Languages: a laboratory based approach*, Addison-Wesley Publishing Company, ISBN 0-201-65697-3, New York.

Tavares, D.M.; Aroca, R.V. & Caurin, G.A.P. (2007). Upgrade of a SCARA Robot using OROCOS. *Proceedings of the 13th IASTED International Conference on Robotics and Applications*, ISBN 978-0-88986-685-0, Würzburg – Germany, Aug 2007, ACTA Press, Calgary.

Tonti, G.; Bradshaw, J. M.; Jeffers, R.; Montanari, R.; Suri, N. & Uszok, A. (2003). Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder. In: *Lecture Notes in Computer Science*, Vol. 2870, Fensel, Dieter; Sycara, Katia; Mylopoulos, John (Eds.), pp. 419-437, Springer Berlin, ISBN 978-3-540-20362-9, Heidelberg.

Vinge, V. (2008). Signs of the Singularity. *IEEE Spectrum*, Vol. 45, No. 6, Jun 2008, pp. 68-74, ISSN 0018-9235.

Control of Lightweight Manipulators Based on Sliding Mode Technique

Jingxin Shi, Fenglei Ni and Hong Liu
Institute of Robotics, Harbin Institute of Technology, Harbin, Hilongjian Province China

Abstract
This chapter focuses on the dynamic control issues of lightweight robots as well as flexible joint robots. The goal is to increase the bandwidth and the accuracy of the trajectory tracking control. Besides the joint flexibility, the control design considers the dynamics of the electric motor in AC-form i.e. the three phase permanent magnet synchronous motor (PMSM). The final system model is a fifth order non-linear system. Based on the theory of integral sliding mode control a robust control approach for the trajectory tracking control of rigid-body robots is presented at first. This control approach has pole-placement capability despite system uncertainties. The controller is then used as the outer position controller for the control of flexible joint robots. To handle the joint flexibility, singular perturbation approach is employed, resulting in reference currents for the inner current control loop of joint motors. For the current control, sliding mode PWM technique is used to overcome the disadvantages of conventional open-loop PWM. The developed control algorithms are simple enough for practical implementation and verified by simulation studies based on a dynamic model consisting of a two-link flexible joint robot with two joint motors.

1. Introduction

The development of robotics in the past few years has been extended from the earlier standard applications of industrial robots to new fields such as service, space robotics and force-feedback systems. The design goals of the new robot generation aim at lightweight, high output torque, high speed, multi-sensory and high degree of learning capability. Such advanced features inevitably increase the complexity of the dynamic control tasks. For a lightweight robot, to avoid the disturbance torque, such as backlash etc., gearboxes with harmonic drive are often involved; this leads to flexibility in robot joints in turns. It is recognized meanwhile that the dynamic control of real world lightweight robots to reach a high system bandwidth is a challenging topic to the current development of robotics and available control technologies. The key factor which limits the system bandwidth is the "high-order", originated from the joint flexibility and the dynamics of the electric motors.
It is recognized that the state-space approach based on the feedback linearization is not adequate for the control of real world lightweight robots and even not adequate for the

control of any high-order non-linear uncertain system, despite being able to assign the closed-loop poles arbitrarily. Another methodology to control the lightweight robots is to decompose the high-order system into two or more lower order sub-systems. There are some remarkable advantages with this methodology: control approaches for rigid-body robots may be used further; the higher order time derivatives of the link position such as acceleration and jerk may be avoided; and, it is easier to set the control system into operation. One of the control methods under this category is the famous singular perturbation approach (as well as the integral manifold approach) which takes the joint torque sub-system as an algebraic system for the link position control and adds some damping for the fast motion in the joint torque. In this way, the joint torque dynamics are resolved without the need of exact tracking of a joint torque reference trajectory. Because the joint torque dynamics are almost "by-passed", this approach may possess a higher bandwidth for the link position control than the pure cascaded control structure with a joint torque control loop being inserted between the link position and motor current control loops. As a result, the composed control structure of singular perturbation approach for the joint torque dynamics can be interpreted as a feed-forward control of the joint torque added by some damping to the fast motion in the joint torque. Singular perturbation approach is verified as a simple and effective approach to stabilize the joint flexibility.

A pioneer of flexible joint robot control is Professor Mark W. Spong when he worked for University Illinois from1984 to 2008. He established the famous Spong-model for flexible joint robots and studied almost all aspects for the dynamic control of this kind of robots.

In the following, some important publications will be citied to clarify the main stream of the dynamic control issues.

The concept of new generation robotics with modular structure was proposed by Hirzinger's group as the spring-out of space robotics technologies (Hirzinger et al., 1994; Gombert et al., 1995). Later on, the concept was modified to the goals of having human arm performance with very high load/own-weight ratio as well as torque sensing and feed-back capability, with certain degree of human intelligence, providing new possibilities for space, medicine and other applications (Stieber et al., 2000; Schmidt, 2000; Hirzinger et al., 2001; Hirzinger et al., 2001; Koeppe & Hirzinger, 2001).

The fundamental control approaches for flexible joint robots were established by Spong (Marino & Spong, 1986; Spong, 1987; Spong, 1988; Spong, 1989). Since then, numerous theoretical results are developed and mainly tested with computer simulation. The developed control methods include:

(a). state-space approach based on the feedback linearization
(b). singular perturbation approach as well as integral manifold approach
(c). dynamic feedback linearization approach
(d). adaptive control technique
(e). simple PD control
(f). PD control + joint torque feedback
(g). passivity based control approach

As proposed in (Spong , 1987), for the state-space approach based on the feedback linearization, even using simplified robot model, the resulting control algorithm may not be realizable due to the state transformation and the inverse calculation of the control inputs. The control algorithm depends on the robot parameters, which are generally unknown.

As stated before, singular perturbation approach is a promising approach by solving the control problem in two time scalars: a fast joint torque damping term for the fast mode of the joint torque dynamics, and a slow joint torque feed-forward term for the outer position control loop (related to the rigid body dynamics of the robot arm) (Spong, 1987; Readman & Mark, 1994).

De Luca involves the previous system information to form the so-called dynamic feedback linearization (De Luca et al., 1998). He uses not only the actual states of the robot dynamics, but also the past states; no global state transformation is required. The resulting control structure is of 2n(n-1) order (with n being the number of robot joints). (De Luca et al., 1998) won a best paper awarded during conference IRCA98 due to the theoretical contribution.

In order to remove the requirement of exact knowledge about robot parameters, adaptive control techniques for flexible joint robots have been developed (Spong, 1989, Lin et al., 1995). These approaches can be viewed as an extension of adaptive control for rigid body robots (Slotine & Li, 1987). Though theoretically looks well, this method met the problem of over complexity for the practical implementation.

Engineers tried PD (or PID) controllers, traditionally used for industrial robots, adding some damping term for the joint flexibility. Stability proof for such control systems, if it is possible, is more involved than that of using extensive model information. Starting from (Arimoto, 1994), which provides the theoretical justification for the PD controller still used in most industrial robots, Tomei (Tomei, 1991) proved the stability of PD control with gravity compensation also for flexible joint robots. However, the stability proofs are only valid for the link position regulation and not for the trajectory tracking control.

Albu-Schaeffer (Albu-Schaeffer & Hirzinger, 2000) proposes an intermediate approach between the theoretical and the practical solutions for the link position control i.e. PD control + joint torque feedback. He uses a simple control structure in the form of joint state feedback with gravity compensation, applicable for a lightweight robot with 7DOF. A stability proof based on Lyapunov theory was provided as well. Also here, the stability proof is valid only for the case of point-to-point motion of the robot arm and not valid for the trajectory tracking control.

Ott (Ott, 2008) studied and tested several control approaches systematically including the passivity based control approach. It comes to the conclusion that the passivity based control approach doesn't show an improved performance for the trajectory tracking control despite of some other advantages. Similar to the works by Albu-Schaeffer (Albu-Schaeffer, 2002), the proposed control algorithms by Ott need often the system parameters which may not be available for general purpose lightweight robots.

In (Ozgoli & Taghirad, 2006) an extensive survey about the control of flexible joint robots is given in which 173 papers from different aspects of the control issue are cited.

It is recognized meanwhile that to design a good control system, the controller designer must have a deep understanding about the physic plant to be controlled, independent from which control approach is applied. As a result, at least a rough model for the controlled plant is required, though there are some unmodeled dynamics, external disturbances and parameter uncertainties associated with this rough model. As a candidate of physic oriented control theories, sliding mode control (Utkin et al., 2009) is selected here for the control problems of flexible joint robots. As it well known, sliding mode control theory can be applied to high-order, non-linear, uncertain MIMO systems and the resulting controllers are simple enough for practical implementations. Another advantage of sliding mode control theory is easy to understand for normal control engineers (it is the main reason why this control theory becomes more and more popular). The major disadvantage associated with sliding mode control is the chattering phenomena due to the high frequency switching of the discontinuous control input. However, if the chattering problem can be solved or the inherent discontinuous property of the plant actuators (like electric motors) can be positively utilized, sliding mode control theory will be a good design tool for deriving the control algorithms. In this chapter, the design methodology of sliding mode control will be the major theoretical tool for the control of flexible joint robots.

The rest of this chapter is organized as follows:
In Section 2, the control problems for rigid-body robot manipulators with modelling uncertainties and external disturbances will be dealt with. The resulting control algorithm will be used for the link position tracking control of flexible joint robots. Section 3 handles the joint torque dynamics based on the singular perturbation approach. We use the result of other researchers without repeating the theory of singularly perturbed systems. Section 4 presents the theoretical derivation of sliding mode PWM for the current control of PMSM. This current controller will be used as the most internal control loop for the link position tracking control. Section 5 shows the simulation study, verifying the developed control algorithms, based on a dynamic model consisting of a two-link flexible joint robot with two joint motors. In section 6 some conclusions will be given.

2. Robust control of rigid manipulators based on integral sliding mode

2.1 Problem statement

For rigid body robot manipulators, the computed torque approach provides asymptotic stability for tracking control tasks. However, the state dependent matrices needed to complete the computed torque algorithm are normally unknown and possibly too complex for a real-time implementation. This section proposes a simple controller with computed-torque-like structure enhanced by integral sliding mode, having pole-placement capability. For the reduction of the chattering effect generated by the sliding mode part, the integral sliding mode is posed as a perturbation estimator with quasi-continuous control action provided by an additional low-pass filter. The time-constant of the latter tunes the controller functionality between the perturbation compensation and a pure integral sliding mode control, as well as between chattering reduction and system robustness.
Studies on the control of chain-like mechanical systems have been a subject of intensive and profitable research over the last three decades. Robot manipulators, as dynamically coupled non-linear MIMO systems have attracted the attention of many control scientists and engineers. Arbitrary assignment of the system poles of a set of decoupled and linearised sub-systems has been the final design goal. The computed torque (Hunt et al., 1983; Gilbert & Ha, 1984), as a theoretically simplest and most comprehensive approach for the tracking control of robot manipulators, allows one to assign the poles of the closed-loop system arbitrarily at the price of an exact feedback linearization with state dependent quantities for compensation of the system non-linearity with coupling terms. Any mismatch due to parameter or modelling uncertainties in the plant will violate exact linearization and decoupling. Moreover, even when these quantities are known exactly, the real-time

implementation is still an issue, since the computational overhead might be too large to prevent the control algorithm from being realized in control hardware.

Motivated by the recent developments on integral sliding mode control (Utkin & Shi, 1996; Poznyak et al., 2004; Cao & Xu, 2004; Castaños & Fridman, 2006; Utkin et al., 2009), by taking regard on algorithm complexity, this section proposes a novel control structure with pole-placement capability for rigid body robot manipulators. Simple matrices describing the nominal model (normally they are constant, as long as the available joint torques are high enough) are used to form a computed-torque-like controller, whereas two diagonal control gain matrices are responsible for the pole-placement. In addition, an additive control vector is designed based on the concept of integral sliding mode to compensate for the overall matched system uncertainties (for systems with unmatched uncertainties, other than the case of full actuated robot manipulators, the readers are referred to (Cao & Xu, 2004; Castaños & Fridman, 2006)).

Control of robot manipulators using sliding mode technique has a rather long history. Since the first set-point sliding mode controller suggested by (Young, 1978), numerous variations have been proposed in the literature, such as the component-wise control discussed by (Slotine, 1985) and by (Chen et al., 1990). The robustness property of the conventional sliding mode control with respect to variations of system parameters and external disturbances can only be achieved after the occurrence of sliding mode. During the reaching phase, however, there is no guarantee for robustness. Integral sliding mode aims at eliminating the reaching phase by enforcing the sliding mode on the entire system response (Utkin & Shi, 1996). As a result, robustness of the system can be guaranteed starting from the initial time instant, that is, a robot manipulator is able to track the reference trajectory (with designed error dynamics given by the pole placement) throughout the entire system response despite the system uncertainties.

However, since a discontinuous term appears in the resulting joint torque, direct implementation of the integral sliding mode control algorithm may be difficult due to the chattering effect. To solve this implementation problem i.e. to reduce the chattering level, the discontinuous term is used for a perturbation estimator based on an auxiliary internal dynamic process. It will be shown that the equivalent control of such a discontinuous term is indeed able to compensate the net system perturbation.

If the equivalent control could be obtained exactly, the system perturbation could be compensated for completely, so that the system would be free of chattering and robust starting from the initial time instant. Strictly speaking, the exact equivalent control based on the system model is impossible to achieve, primarily due to model uncertainties. However, if the spectrum of the equivalent control has no overlap with the switching frequency of the discontinuous control term (it is normally the case in practice), a low-pass filter can be used to extract the equivalent control from the discontinuous control term (Utkin, 1992). Using low-pass filter to extract equivalent control from the discontinuous control term provides the basic information source of proposed control design.

From the practical point of view, the bandwidth of the low-pass filter is designed as low as possible, so that the amplitude of the chattering remains low level. However, since the frequency of the equivalent control is time-varying, a low-pass filter with a fixed time-constant and low bandwidth would "cut" the equivalent control and lose the information about the system perturbation. Thus, there is a trade-off between the system robustness (whether the system perturbation can be compensated for completely) and the chattering reduction by tuning of the time-constant of the low-pass filter.

2.2 Integral sliding mode control and perturbation estimator

In this section, the basic concept and the main result of integral sliding mode control will be outlined.

For a given dynamic system represented by the following state space equation

$$\dot{x} = f(x) + B(x)u + h(x,t) \tag{1}$$

with $x \in \Re^n$ being the state vector, $u \in \Re^m$ being the control input vector ($rank\ B(x) = m$) and $h(x,t)$ being the perturbation vector due to model uncertainties or external disturbances; $h(x,t)$ is bounded and assumed to fulfil the matching condition. The control low for system (1) is proposed as

$$u = u_0 + u_1 \tag{2}$$

where $u_0 \in \Re^m$ is responsible for the performance of the nominal system; $u_1 \in \Re^m$ is a discontinuous control action that rejects the perturbations by ensuring the sliding motion. The sliding manifold is defined as

$$\begin{gathered} s = s_0(x) + z\,, \\ \text{with} \\ s,\ s_0(x),\ z \in \Re^m \\ \dot{z} = -\frac{\partial s_0}{\partial x}\{f(x) + B(x)u_0(x)\} \\ z(0) = -s_0(x(0)) \end{gathered} \tag{3}$$

where initial condition $z(0)$ is determined under the requirement $s(0) = 0$. It can be proven that the equivalent control of u_1 will cancel out the perturbation term $h(x,t)$, see (Utkin et al. 2009). Discontinuous control u_1 has a proper selected control gain which ensures sliding motion starting from $t = 0$ i.e. $s(0) = 0$.

In real applications, however, discontinuous control u_1 may result in chattering effect, imposing high frequency vibrations. To reduce this undesired effect, the control system can be modified as follows:

$$\begin{gathered} s = s_0(x) + z \\ \dot{z} = -\frac{\partial s_0}{\partial x}\{f(x) + B(x)u - B(x)u_1\} \\ z(0) = -s_0(x(0)) \\ u = u_0 + u_{1av} \\ u_{1av} = lowpass(u_1) \end{gathered} \tag{4}$$

By solving equation $\dot{s}=0$ for u_1, it can be directly checked that the equivalent control of u_1 still cancels the system perturbation. In the above controller, relation $u_{1eq}=u_{1av}$ is used, for proof see (Utkin, 1992). Finally, the term u_{1av} is quasi-continuous (depending on the time-constant of the low-pass filter) and equal to the perturbation term to be compensated for, serving as the perturbation estimator. Moreover, since discontinuous control u_1 appears only in the control computer, its gain is more flexible to tune.

2.3 Control of robot manipulators

2.3.1 Model of rigid body robot manipulators

The model of a rigid body robot manipulator with n degrees of freedom can be written as

$$M(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)+F(\dot{q})=\tau \tag{5}$$

where $M(q)\in\Re^{n\times n}$ is the mass matrix; $C(q,\dot{q})\dot{q}\in\Re^{n}$ is the vector including centrifugal and Coriolis forces; $G(q)\in\Re^{n}$ is the gravity force vector; $F(\dot{q})\in\Re^{n}$ is the friction force vector; $q\in\Re^{n}$ represents the joint position vector and $\tau\in\Re^{n}$ denotes the joint torque vector.
For the purpose of control design, the notation of the above model can be formally changed to

$$M(q)\ddot{q}+N(q,\dot{q})=\tau \tag{6}$$

where vector $N(q,\dot{q})=C(q,\dot{q})\dot{q}+G(q)+F(\dot{q})$ does not contain term $\ddot{q}$. This model can be rewritten as the sum of an ideal model and a perturbation term:

$$M_0(q)\ddot{q}+N_0(q,\dot{q})=\tau+H(q,\dot{q},\ddot{q}) \tag{7}$$

where $M_0(q)=M(q)-\Delta M$, $N_0(q,\dot{q})=N(q,\dot{q})-\Delta N$, with ΔM and ΔN being the unknown part of matrix $M(q)$ and vector $N(q,\dot{q})$, respectively; vector $H(q,\dot{q},\ddot{q})$ denotes the overall system perturbation and has the form $H(q,\dot{q},\ddot{q})=-(\Delta M\,\ddot{q}+\Delta N)$. Note that the perturbation term $H(q,\dot{q},\ddot{q})$ satisfies the matching condition.

2.3.2 Control design using integral sliding mode

Following the design principle given in section 2.2, the joint torque vector τ can be designed as two additive terms:

$$\begin{aligned}\tau&=\tau_0+\tau_1\\ \tau_0&=M_0(q)(\ddot{q}_d-K_D\dot{q}_e-K_Pq_e)+N_0(q,\dot{q})\end{aligned} \tag{8}$$

where $M_0(q)$, $N_0(q,\dot{q})$ are the nominal value of $M(q)$, $N(q,\dot{q})$, respectively, as defined with equation (7); $K_P\in\Re^{n\times n}$, $K_D\in\Re^{n\times n}$ are positive definite diagonal gain matrices determining the closed loop performance; and the tracking error is defined as $q_e(t) = q(t) - q_d(t)$ with $[q_d(t)\ \ \dot{q}_d(t)\ \ \ddot{q}_d(t)]$ being the reference trajectory and its time derivatives. Note that τ_0 represents the computed torque part of the controller.

Discontinuous control τ_1 is now derived based on the design principle of integral sliding model control:

Step 1: Sliding Manifold

The sliding manifold is defined based on equation (3)

$$
\begin{aligned}
s &= s_0(x) + z \\
s_0 &= \begin{bmatrix} C & I \end{bmatrix} \begin{bmatrix} q_e \\ \dot{q}_e \end{bmatrix} \\
\dot{z} &= -\begin{bmatrix} C & I \end{bmatrix} \begin{bmatrix} \dot{q}_e \\ -M_0^{-1}N_0 + M_0^{-1}\tau_0 - \ddot{q}_d \end{bmatrix} \\
z(0) &= -Cq_e(0) - \dot{q}_e(0)
\end{aligned}
\tag{9}
$$

where $C \in \Re^{n\times n}$ is a positive definite gain matrix and $I \in \Re^{n\times n}$ is a $n \times n$ unit matrix.

Vector s can be further simplified by substituting τ_0 with equation (8):

$$
s = \dot{q}_e + K_D q_e + K_P \int_0^t q_e(\xi)d\xi - \dot{q}_e(0) - K_D q_e(0) \tag{10}
$$

Since the requirement $s(0) = 0$ is satisfied, sliding mode will occur starting from the initial time instant $t = 0$. Note that for the implementation of s, matrix C is not required in the final equation, see (10). As one can see from the derivations given above, equation (10) is the natural extension of the basic design equation of integral sliding mode (3).

To prepare the stability analysis, the time derivative of the sliding variable $s(t)$ can be obtained

$$
\dot{s} = \dot{s}_0 + \dot{z} = \zeta_1 + \zeta_2\tau_0 + M^{-1}\tau_1 \tag{11}
$$

where $\zeta_1 = (M_0^{-1}N_0 - M^{-1}N)$ and $\zeta_2 = (M^{-1} - M_0^{-1})$ represent the mismatches between the nominal parameters $M_0(q)$, $N_0(q,\dot{q})$, and the real system parameters $M(q)$, $N(q,\dot{q})$, respectively, viewed as system perturbation terms. Note that in this study we assume that both ζ_1 and $\zeta_2\tau_0$ are norm-bounded.

Step 2: Discontinuous control τ_1

τ_1 is the discontinuous control dedicated to reject the overall perturbation torque $H(q,\dot{q},\ddot{q})$. Here τ_1 can be selected as

$$\tau_1 = -\Gamma_0 \frac{s}{\|s\|} \tag{12}$$

where Γ_0 is a positive constant (control gain may also take other forms) and $\|s\|$ denotes the norm 2 of s i.e. $\|s\| = \sqrt{s_1^2 + s_2^2 + \cdots s_n^2}$.

Step 3: Design of the control gain Γ_0

Select a Lyapunov function candidate as $V = \frac{1}{2} s^T s > 0$ (for $s \neq 0$). The time derivative of V along the solutions of (11) is given by

$$\dot{V} = s^T \dot{s} = s^T (\zeta_1 + \zeta_2 \tau_0) - \Gamma_0 s^T M^{-1} s \big/ \|s\| \tag{13}$$

Since matrix $M^{-1}(q)$ is positive definite and Γ_0 is a positive constant, the most right term in (13) i.e. $\Gamma_0 s^T M^{-1} s \big/ \|s\|$ is positive for any $s \neq 0$. For a small enough positive number ρ, such that inequality $\Gamma_0 s^T M^{-1} s \big/ \|s\| \geq \Gamma_0 s^T \rho s \big/ \|s\|$ holds, it can be shown that

$$\dot{V} \leq -\|s\| \left(\Gamma_0 \rho - \|\zeta_1 + \zeta_2 \tau_0\| \right) \tag{14}$$

Clearly, under the norm-boundedness condition of terms ζ_1 and $\zeta_2 \tau_0$, a large enough gain Γ_0 can always be chosen to guarantee $\dot{V} < -\alpha \|s\|$ (with $\alpha > 0$ and for $\|s\| \neq 0$), implying the occurrence of sliding mode in finite time. Note that the initial conditions in (10) eliminate the reaching phase.

Step 4: Equivalent control of τ_1

Once sliding mode occurs and the system is confined to the manifold $s(t) = 0$, the equivalent control of τ_1 can be used to examine the system behaviour. The equivalent control is obtained by formally setting $\dot{s} = 0$, yielding

$$\tau_{1eq} = -M(\zeta_1 + \zeta_2 \tau_0) \tag{15}$$

Substitution of $\tau = \tau_0 + \tau_{1eq}$ in equation (6) with equivalent control (15) leads to the motion equation in sliding mode, which can be simplified as

$$M_0(q)\ddot{q} + N_0(q, \dot{q}) = \tau_0 \tag{16}$$

Control τ_0 in (8) thus achieves the designed (closed-loop) error dynamics defined by K_D and K_P, namely

$$\ddot{q}_e + K_D \dot{q}_e + K_P q_e = 0 \tag{17}$$

as if perturbation term $H(q,\dot{q},\ddot{q})$ in (7) would not have existed. Equation (16) as well as (17) represents the system motion in sliding mode. Solving $\ddot{q}$ from (16) and setting into $H(q,\dot{q},\ddot{q})$, easily shows the perturbation cancellation property, i.e. $\tau_{1eq} = -H(q,\dot{q},\ddot{q})$. The derivation above is only to show the perturbation cancellation property by the equivalent control τ_{1eq}. Actually, the designed closed loop motion presented by (17) can be obtained more easily by taking the time derivative of (10) and set $\dot{s} = 0$.
Summarization of the integral sliding mode control system for the implementation:

$$\begin{gathered}
\tau_0 = M_0(q)(\ddot{q}_d - K_D\dot{q}_e - K_P q_e) + N_0(q,\dot{q}) \\
s = \dot{q}_e + K_D q_e + K_P \int_0^t q_e(\xi)d\xi - \dot{q}_e(0) - K_D q_e(0) \\
\tau_1 = -\Gamma_0 \frac{s}{\|s\|} \\
\tau = \tau_0 + \tau_1
\end{gathered} \tag{18}$$

From (18), one can see the benefit of the control system: in order to assign the poles of the closed-loop system arbitrarily, one needs only to additionally calculate the variable s and τ_1, exact knowledge about $M(q)$ and $N(q,\dot{q})$ are not required. Depending on the available control resource, the nominal quantities $M_0(q),\ N_0(q,\dot{q})$ can even be set constant i.e. to $M_0,\ N_0$. Moreover, the robustness of the tracking control performance is ensured starting from $t = 0$.

2.3.3 Control design using integral sliding mode based perturbation estimator

Hitherto, the control system described in section 2.3.2 looks perfect. However, in some practical applications, the controller given in (18) may not be applicable to robot manipulators, as the chattering level generated by the discontinuous control term τ_1 may be very high. Following the control design approach given by (4), the control system can be modified to:

$$\begin{gathered}
\tau_0 = M_0(q)(\ddot{q}_d - K_D\dot{q}_e - K_P q_e) + N_0(q,\dot{q}) \\
s = \dot{q}_e + K_D q_e + K_P \int_0^t q_e(\xi)d\xi - \dot{q}_e(0) - K_D q_e(0) + \int_0^t M_0^{-1}(q)(\tau_1 - \tau_{1av})d\xi \\
\tau_1 = -\Gamma_0 \frac{s}{\|s\|} \\
\tau_{1av} = lowpass(\tau_1) \\
\tau = \tau_0 + \tau_{1av}
\end{gathered} \tag{19}$$

Note that for a better decoupling, the control gain of τ_1 may also be selected as $M_0\Gamma_0$ instead of Γ_0. However, since we are intended to compare the solution based on the perturbation estimator with the pure integral sliding mode control (18), the control gain is designed to have the same form for the both control systems. Now, the equivalent control of τ_1 can be obtained by setting $\dot{s}=0$

$$\begin{gathered}\dot{s}=\dot{s}_0+\dot{z}=\zeta_1+\zeta_2\tau+M_0^{-1}\tau_1=0\\ \tau_{1eq}=-M_0(\zeta_1+\zeta_2\tau)\end{gathered}\tag{20}$$

Actually, since $\tau=\tau_0+\tau_{1av}=\tau_0+\tau_{1eq}$, (20) can be further simplified to (15), implying that the equivalent control τ_{1eq} remains the same as in the case of pure integral sliding mode control.

For the convergence proof of s to zero, check that the dynamic motion about s in the closed-loop system can be derived as

$$\dot{s}=\dot{s}_0+\dot{z}=\zeta_1+\zeta_2\tau-M_0^{-1}\tau_1\tag{21}$$

For a Lyapunov function candidate $V=\frac{1}{2}s^Ts>0$ (for $s\neq0$), the time derivative of V along the solutions of (21) can be obtained as

$$\dot{V}=s^T\dot{s}=s^T(\zeta_1+\zeta_2\tau)-\Gamma_0 s^TM_0^{-1}s/\|s\|\tag{22}$$

Similar lines as in (14) can be followed to show that a large enough control gain Γ_0 can be selected such that $\dot{V}<-\alpha\|s\|$ (with $\alpha>0$ and for $\|s\|\neq0$), implying that sliding mode will be enforced in finite time. Note that τ in (22) is now quasi-continuous due to the low-pass filter, it can be assumed here that terms ζ_1 and $\zeta_2\tau$ are norm-bounded. Again, initial conditions guarantee that $s(0)=0$ in (19), thus eliminating the reaching phase.

The advantage of controller (19) over the previous controller given by (18) is: the discontinuous control term τ_1 (with gain Γ_0) appears only in the control computer and the real control τ applied to the robot manipulator (see (19)) is low-pass filtered. Control term τ_{1av} serves here as a perturbation compensator. As one can see from (19), if the time constant of the low-pass filter tends to zero, the controller given by (19) will converge to controller (18), i.e. from perturbation estimation solution to integral sliding mode control solution. For the control system under controller (19), sliding mode $s(t)\equiv0$ is guaranteed throughout the entire system response, although a low-pass filter is involved in the control loop.

2.3.4 Practical consideration

Since low-order filters do not ideally cut off the high-frequency switching signal components due to the discontinuous term τ_1, some amount will be still preserved in τ_{1av}.

Whereas for practical applications, a large time constant for the low-pass filter is normally preferred, such that the resulting control signals remain as smooth as possible. However, since the instantaneous frequency of the system perturbation (i.e. the frequency of τ_{1eq} after sliding mode occurs) is unknown and time changing, it may happen that the bandwidth of τ_{1eq} is higher than the bandwidth of the low-pass filter and the system perturbation cannot be cancelled out completely, thus the system robustness is reduced. For a high control performance, the time constant of the low-pass filter should be made small (at least during the transient period) such that the bandwidth of the low-pass filter is high enough and τ_{1eq} can get through the filter completely.
As a result, in the practical implementation the time constant of the low-pass filter can be used as a trade-off between chattering reduction and system robustness: if a high robustness as well as high control accuracy during the transient period is required, the time constant of the low-pass filter can be made small for the short time period. The trade-off between chattering reduction and system robustness by changing the time constant of the low-pass filter is demonstrated in Sections 5.2 and 5.3.

3. Singular perturbation approach to handle the joint flexibility

As mentioned in the introduction part, singular perturbation approach has at least the following advantages:
(a). the signals for the control implementation can be made available
(b). there is no need to implement an exact tracking controller for the joint torque
(c). the results for the control of rigid-body robots can be used further
(d). the implementation of the control algorithm is easy

Sure, singular perturbation approach has also disadvantages:
(a). it is not valid if the joint stiffness is too low
(b). the control law is sensitive to the change of joint stiffness
Fortunately, most of lightweight manipulators used in practice have high enough and fixed joint stiffness. The flexibility in robot joints is a side-effect to achieve lightweight and it is normally not intended by the robot designer.
The control algorithm of this section will be summarized here without repeating the theory of singularly perturbed systems. The way of treating the joint torque dynamics can be find e.g. in (Ott, 2008).
The output of the robust link position controller for rigid body manipulators given in Section 2 is denoted here as τ_d (instead of τ), which is the reference input for the joint torque implementation. Normally, when using singular perturbation approach for the control of slow dynamics, the joint inertia matrix J has to be considered in the link position controller by adding matrix J to the mass-matrix of the robot arm $M(q)$. However, since our link position controller is a robust controller, implying that no exact parameters are required, the information about the joint inertia is normally not necessary (the system robustness depends on the available control resource).

The reference current vector for the joint motors can be calculated from the slow and fast torque components i.e. $\tau_s \in R^n$ and $\tau_f \in R^n$ for stabilizing the joint torque dynamics

$$I_q^* = K_t^{-1}\tau_m = K_t^{-1}G_r^{-1}(\tau_s + \tau_f) \tag{23}$$

where K_t is the diagonal torque constant matrix of the electric motors and G_r is the diagonal gear-ratio matrix; $I_q^* = [i_{qi}^*] \in R^n$, with $i = 1 \sim n$, is the reference current vector including the reference currents for all joints; τ_m represents the motor torque vector. The slow and fast joint torque components can be given as

$$\begin{gathered}\tau_s = \tau_d \\ \tau_f = -D_\tau \dot{\tau} \\ \text{or} \\ \tau_f = -K_\tau(\tau - \tau_d) - D_\tau \dot{\tau}\end{gathered} \tag{24}$$

with $D_\tau \in R^{n\times n}$ and $K_\tau \in R^{n\times n}$ being constant diagonal control gain matrices to be determined by the control designer (if the joint stiffness is changed the control gain matrices need to be retuned accordingly).

4. Direct current control using sliding mode PWM

When using the build-in PWM unit of a micro-controller or a DSP, the required reference voltage signals generated by the current controller will be modulated in form of pulse-width and then it is hoped that the average value of the terminal voltages of the stator windings will be equal to the reference voltages that the current controller produces. In this configuration there are two problems:
(a). the PWM implementation of the terminal voltages is done in a way of open-loop, the final voltages on the stator windings may differ from the ones what current controller requires, depending on the quality of the pulse-width-modulation.
(b). it introduces some time delay, at lease a duty-cycle has to remain unchanged before the corresponding PWM signal being sent out.
Thus for a high dynamic performance, the build-in PWM unit of a micro-controller or a DSP has some disadvantages.
On the other hand, the conventional current control hardware such as Chopper-Control or Hysteresis-Control hardware do not have these disadvantages. Because no micro-processor being available, these practically used hardware were not able to implement the concept of field-oriented control. In this section we derive a current controller based on sliding mode control theory for PMSM which has the performance of field-oriented control, but without the disadvantage associated with the open-loop PWM techniques. We call this kind of current control "sliding mode PWM current control".
At first, we need the motor model to design the current controller. The motor model in the (d, q)-coordinate frame, which rotates synchronously with the motor rotor, can be given as

$$L\frac{di_d}{dt} = u_d - Ri_d + L\omega_e i_q$$
$$L\frac{di_q}{dt} = u_q - Ri_q - L\omega_e i_d - \lambda_0\omega_e \tag{25}$$

where L is the stator inductance and R is the stator resistance; i_d and i_q are the stator currents in the (d, q) coordinate frame; u_d and u_q are the stator voltages in the same coordinate frame; λ_0 is the flux constant of the rotor permanent magnet; ω_e is the rotor electric angular speed.
For the sliding mode current controller, the switching functions for the d and q current components are designed as

$$s_d = i_d - i_d^*$$
$$s_q = i_q - i_q^* \tag{26}$$

where i_q^* is the reference current i.e. one of the components of the compose controller (23) (index i is neglected here for simplicity), and reference current component $i_d^* = 0$ for constant torque operation and $i_d^* \neq 0$ for field-weakening operation (Shi & Lu, 1996). The time derivative of both switching functions along the solutions of (25) can be found as

$$\dot{s}_d = \dot{i}_d - \dot{i}_d^* = \frac{1}{L}u_d - \frac{R}{L}i_d + \omega_e i_q - \dot{i}_d^*$$
$$\dot{s}_q = \dot{i}_q - \dot{i}_q^* = \frac{1}{L}u_q - \frac{R}{L}i_q - \omega_e i_d - \frac{\lambda_0}{L}\omega_e - \dot{i}_q^* \tag{27}$$

Introducing two auxiliary variables f_d and f_q as follows

$$f_d = -\frac{R}{L}i_d + \omega_e i_q - \dot{i}_d^*$$
$$f_q = -\frac{R}{L}i_q - \omega_e i_d - \frac{\lambda_0}{L}\omega_e - \dot{i}_q^* \tag{28}$$

(27) will be simplified to

$$\dot{s}_d = f_d + L^{-1}u_d$$
$$\dot{s}_q = f_q + L^{-1}u_q \tag{29}$$

The above equation system can be summarized in vector form, resulting in

$$\begin{bmatrix} \dot{s}_d \\ \dot{s}_q \end{bmatrix} = \begin{bmatrix} f_d \\ f_q \end{bmatrix} + L^{-1} \begin{bmatrix} u_d \\ u_q \end{bmatrix} \tag{30}$$

Here stator voltages u_d and u_q are not yet the discontinuous voltages applied to the stator windings. For the sliding mode current control we need the relationship between the final discontinuous voltages applied to the stator windings i.e. $u_1 \sim u_3$ (which take the values from the set $\{-u_0, u_0\}$ with u_0 being the DC-Bus voltage) and the time derivative of both switching functions. This relationship can be given as

$$\begin{bmatrix} \dot{s}_d \\ \dot{s}_q \end{bmatrix} = \begin{bmatrix} f_d \\ f_q \end{bmatrix} + L^{-1} \begin{bmatrix} u_d \\ u_q \end{bmatrix} = \begin{bmatrix} f_d \\ f_q \end{bmatrix} + L^{-1} A_{d,q}^{1,2,3} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \tag{31}$$

where matrix $A_{d,q}^{1,2,3}$ can be expended as

$$A_{d,q}^{1,2,3} = \begin{bmatrix} \cos\theta_a & \cos\theta_b & \cos\theta_c \\ -\sin\theta_a & -\sin\theta_b & -\sin\theta_c \end{bmatrix} \tag{32}$$

with $\theta_a = \theta_e$, $\theta_b = \theta_e - 2\pi/3$, $\theta_c = \theta_e + 2\pi/3$ and θ_e being the rotor electrical angular position. Using (32), (31) can be rewritten as

$$\begin{bmatrix} \dot{s}_d \\ \dot{s}_q \end{bmatrix} = \begin{bmatrix} f_d \\ f_q \end{bmatrix} + L^{-1} \begin{bmatrix} u_1 \cos\theta_a + u_2 \cos\theta_b + u_3 \cos\theta_c \\ -u_1 \sin\theta_a - u_2 \sin\theta_b - u_3 \sin\theta_c \end{bmatrix} \tag{33}$$

To find the control signals u_1, u_2 and u_3, Lyapunov approach can be employed. Design a Lyapunov function candidate as

$$V = \frac{1}{2} \boldsymbol{S}_{dq}^T \boldsymbol{S}_{dq} \tag{34}$$

where $\boldsymbol{S}_{dq} = [s_d \ \ s_q]^T$. The time derivative of V along the solution of (33) can be found as

$$\begin{aligned} \dot{V} &= \boldsymbol{S}_{dq}^T \dot{\boldsymbol{S}}_{dq} \\ &= \begin{bmatrix} s_d & s_q \end{bmatrix} \begin{bmatrix} \dot{s}_d \\ \dot{s}_q \end{bmatrix} \\ &= \begin{bmatrix} s_d & s_q \end{bmatrix} \begin{bmatrix} f_d \\ f_q \end{bmatrix} + L^{-1} \begin{bmatrix} s_d & s_q \end{bmatrix} \begin{bmatrix} u_1 \cos\theta_a + u_2 \cos\theta_b + u_3 \cos\theta_c \\ -u_1 \sin\theta_a - u_2 \sin\theta_b - u_3 \sin\theta_c \end{bmatrix} \end{aligned} \tag{35}$$

which can be further expanded to

$$\begin{aligned}\dot{V} &= \boldsymbol{S}_{dq}^{T}\dot{\boldsymbol{S}}_{dq} \\ &= (s_d f_d + s_q f_q) + L^{-1}[u_1(s_d\cos\theta_a - s_q\sin\theta_a) + u_2(s_d\cos\theta_b - s_q\sin\theta_b) + u_3(s_d\cos\theta_c - s_q\sin\theta_c)]\end{aligned} \tag{36}$$

Introducing the following three auxiliary variables

$$\begin{aligned}\Omega_1 &= (s_d\cos\theta_a - s_q\sin\theta_a) \\ \Omega_2 &= (s_d\cos\theta_b - s_q\sin\theta_b) \\ \Omega_3 &= (s_d\cos\theta_c - s_q\sin\theta_c)\end{aligned} \tag{37}$$

equation (36) can be simplified to

$$\dot{V} = (s_d f_d + s_q f_q) + L^{-1}(u_1\Omega_1 + u_2\Omega_2 + u_3\Omega_3) \tag{38}$$

In order to guarantee $\dot{V} < 0$, the control signals u_1, u_2 and u_3 can be designed as

$$\begin{aligned}u_1 &= -u_0 sign(\Omega_1) \\ u_2 &= -u_0 sign(\Omega_2) \\ u_3 &= -u_0 sign(\Omega_3)\end{aligned} \tag{39}$$

With these notations, equation (38) can be reformulated for the final analysis

$$\begin{aligned}\dot{V} &= (s_d f_d + s_q f_q) - L^{-1}u_0[sign(\Omega_1)\Omega_1 + sign(\Omega_2)\Omega_2 + sign(\Omega_3)\Omega_3] \\ &= (s_d f_d + s_q f_q) - L^{-1}u_0[|\Omega_1| + |\Omega_2| + |\Omega_3|]\end{aligned} \tag{40}$$

In the above equation, L^{-1} is a constant (but may be unknown). If the scalar term $(s_d f_d + s_q f_q)$ is bounded and if the DC-bus voltage u_0 is high enough, $\dot{V} < 0$ can be guaranteed, implying that the real currents will converge to their reference counterparts in finite time. Thus the stability of the current control system can be ensured under two conditions
(a). the DC-bus voltage u_0 is high enough
(b). auxiliary variables f_d and f_q are bounded
Since f_d and f_q do not contain the control voltages, neither u_d and u_q, nor u_1, u_2 and u_3, the condition (b) is reasonable. Note that if the reference currents i_d^* and i_q^* change too fast, the stability condition may be violated from time to time (depending on the available DC-bus voltage u_0). In this case there exists no current controller which can do better. Some researchers design sliding mode link position controller with discontinuous joint torque commands and without taking into account the motor dynamics, would meet this problem. Other high gain link position controllers without taking into account the motor dynamics would meet the same problem.
Now the implementation procedure is summarized.

Though the derivation of the proposed current controller looks rather involved, the implementation of this controller is quite simple. The equations for the implementation are summarized as follows

$$\begin{aligned} &\begin{matrix} s_d = i_d - i_d^* \\ s_q = i_q - i_q^* \end{matrix}, \quad \begin{matrix} \Omega_1 = (s_d \cos\theta_a - s_q \sin\theta_a) \\ \Omega_2 = (s_d \cos\theta_b - s_q \sin\theta_b) \\ \Omega_3 = (s_d \cos\theta_c - s_q \sin\theta_c) \end{matrix}, \\ &u_1 = -u_0 sign(\Omega_1) \\ &u_2 = -u_0 sign(\Omega_2) \\ &u_3 = -u_0 sign(\Omega_3) \end{aligned} \tag{41}$$

with $\theta_a = \theta_e$, $\theta_b = \theta_e - 2\pi/3$, $\theta_c = \theta_e + 2\pi/3$. The final gating signals taking values from set {0, 1} (like PWM signals) feeding to the inverter can be found as

$$\begin{aligned} s_{w1} &= 0.5\left(1 + u_1/u_0\right), \\ s_{w4} &= 1 - s_{w1}, \\ s_{w2} &= 0.5\left(1 + u_2/u_0\right), \\ s_{w5} &= 1 - s_{w2}, \\ s_{w3} &= 0.5\left(1 + u_3/u_0\right), \\ s_{w6} &= 1 - s_{w3}. \end{aligned} \tag{42}$$

The switching control signals $s_{w1} \sim s_{w6}$ are pulse signals, the pulse width is not calculated from some duty-cycle, but determined directly and instantaneously by the current control errors in the field-oriented coordinates. Note that in practical implementation, several μs time delay is required between signal pair s_{wi} and s_{wi+3} ($i = 1 \sim 3$). This current control system does not require the motor parameters as well as the decoupling process, thus it is a robust current control system.

5. Simulation Studies

5.1 A two-link robot manipulator as an example

A planar, two-link manipulator with revolute joints, taken from the example in (Utkin et al., 2009), is used here to demonstrate the proposed control approaches. The manipulator and the associated variables are depicted in Figure 1.

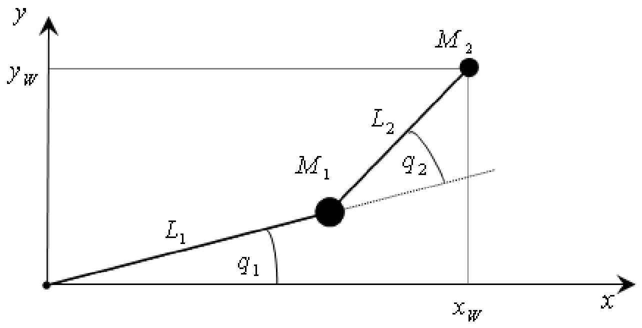


Fig. 1. Two-link manipulator with link lengths L_1 and L_2, and concentrated link masses M_1 and M_2. The manipulator is shown in joint configuration (q_1, q_2), which leads to end-effector position (x_W, y_W) in world coordinates.

The end-effector position, (x_W, y_W), i.e. the location of mass M_2 in world coordinate frame (x, y), is given by

$$\begin{aligned} x_W &= L_1 \cos q_1 + L_2 \cos(q_1 + q_2), \\ y_W &= L_1 \sin q_1 + L_2 \sin(q_1 + q_2), \end{aligned} \tag{43}$$

where (q_1, q_2) denotes the joint displacements and L_1, L_2 are the link lengths. Solving (43) for the joint displacements as a function of the end-effector position (x_W, y_W) yields the inverse kinematics as

$$\begin{aligned} &q_2 = \operatorname{atan2}(D, C), \qquad \text{with } C = \frac{x_W^2 + y_W^2 - L_1^2 - L_2^2}{2L_1L_2},\ D = \pm\sqrt{1 - C^2} \\ &q_1 = \operatorname{atan2}(y_W, x_W) - \operatorname{atan2}(L_2 \sin q_2, L_1 + L_2 \cos q_2) \end{aligned} \tag{44}$$

which obviously is not unique due to the two sign options of the square root in variable D. The function "atan2(.)" describes the arctan function normalized to the range$\pm 180^\circ$.
The dynamic model of the two-link manipulator can be given as

$$\begin{aligned} &\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} c_1 + g_1 + f_1 \\ c_2 + g_2 + f_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}, \text{i.e.} \\ &M(q) = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix},\ N(q, \dot{q}) = \begin{bmatrix} c_1 + g_1 + f_1 \\ c_2 + g_2 + f_2 \end{bmatrix}, \end{aligned} \tag{45}$$

with

$$
\begin{aligned}
&m_{22} = L_2^2 M_2, \\
&m_{12} = m_{21} = m_{22} + L_1 L_2 M_2 \cos q_2, \\
&m_{11} = L_1^2 (M_1 + M_2) + 2m_{12} - m_{22}, \\
&c_1 = -L_1 L_2 M_2 (2\dot{q}_1 \dot{q}_2 - \dot{q}_2^2) \sin q_2, \\
&c_2 = L_1 L_2 M_2 \dot{q}_1^2 \sin q_2, \\
&g_2 = L_2 M_2 g \cos(q_1 + q_2), \\
&g_1 = L_1 (M_1 + M_2) g \cos(q_1) + g_2, \\
&f_1 = k_{v1} \dot{q}_1 + k_{c1} sign(\dot{q}_1), \\
&f_2 = k_{v2} \dot{q}_2 + k_{c2} sign(\dot{q}_2),
\end{aligned}
\tag{46}
$$

where k_{vi} and k_{ci} ($i = 1,2$) are coefficients of viscous friction and coulomb friction, respectively.
The joint model for the two robot joints is given by

$$
\begin{aligned}
&J_i \ddot{\theta}_i + \tau_{dsi} + \tau_i = g_{ri} \tau_{mi} \\
&\tau_i = K_i (\theta_i - q_i) \quad i = 1 \sim 2
\end{aligned}
\tag{47}
$$

where the parameters and variables for the i th joints are

q_i :	link position
θ_i :	joint position
τ_i :	joint torque
τ_{mi} :	motor torque
τ_{dsi} :	disturbance torque
J_i :	joint inertia
K_i :	joint stiffness
g_{ri} :	gear ratio

The electric motor model for each joint is taken from equation (25) with the transformation matrix given in (32).
The plant parameters for the simulation study are selected as shown in Table 1 through Table 3. Note that for the simulation, we select the joint disturbance torque in equation (47) as pure viscous friction $\tau_{dsi} = k_{\omega i} \dot{\theta}_i$ for both joints (but at link side both viscous and coulomb frictions are applied, see equation (46) and section 5.2).

M_1	M_2	L_1	L_2
2 kg	1 kg	0.5 m	0.5 m

Table 1. Arm Parameters

L (H)	R (Ohm)	λ_0 (Wb)	P	K_t (Nm/A)	$I_{q\_\max}$ (A)	u_0 (V)
22.5×10^{-3}	0.78	0.26	4	$(3/2)P\lambda_0$	50	100

Table 2. Parameters for motor 1 and motor 2 (P = number of pole-pair)

J (Kgm^2)	K (Nm/Rad)	g_r	k_ω (Nm/(Rad/s))
1.0	12000	20	1

Table 3. Parameters of joint 1 and joint 2

For the trajectory tracking control task, we will demand the manipulator to follow a circular trajectory in its workspace. The circle with centre (x_{d0}, y_{d0}) and radius r_d is given in world coordinates by

$$\begin{aligned} x_d(t) &= x_{d_0} + r_d \cos\psi_d \\ y_d(t) &= y_{d_0} - r_d \sin\psi_d \\ \psi_d(t) &= \frac{2\pi}{t_f}t - \sin\left(\frac{2\pi}{t_f}t\right), \quad 0 \le t \le t_f, \end{aligned} \tag{48}$$

where the operation is assumed to start at time $t = 0$ and to be completed at final time $t = t_f$. Through the inverse kinematics, the reference link angles for joint 1 and joint 2 are calculated according to (44). The parameters for the reference trajectory are chosen as shown in Table 4.

x_{d0}	y_{d0}	r_d	t_f
0.25 m	0.25 m	0.5 m	2 s

Table 4. Parameters of reference circular trajectory.

5.2 Controller parameters and simulation configuration

The parameters for the outer link position control loop are selected as:

$$\begin{gathered} M_0(q) = \begin{bmatrix} 2.5 & 0 \\ 0 & 1 \end{bmatrix}, \\ N_0(q,\dot{q}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \\ K_p = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}, \ K_d = \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix}, \\ \Gamma_0 = \begin{bmatrix} 400 & 0 \\ 0 & 400 \end{bmatrix} \end{gathered} \tag{49}$$

The joint torques of both joints are limited to 400Nm. To extract the equivalent control from the discontinuous control term to obtain $\tau_{1av} = \tau_{1eq}$ in equation (19), a simple first order low-pass filter is used i.e.

$$\mu \dot{y} = -y + u \tag{50}$$

where μ is the filter time-constant. In the simulation $\mu = 0.025$ is selected. In the transition period the frequency of τ_{1eq} may be higher than the edge-frequency of the low-pass filter, see the discussion in Section 2.3.4. To solve this problem, the time constant of the low-pass filter is made time varying:

$$\mu(t) = \begin{cases} (0.025/0.5)t, \ 0 \le t \le 0.5 \\ \quad 0.025, \ t > 0.5 \end{cases} \tag{51}$$

Now the time constant of the low-pass filter is linearly increased from zero to 0.025s in half second and remains constant thereafter.
For the singular perturbation approach described in Section 3, the simple form $\tau_f = -D_\tau \dot{\tau}$ is used for the fast dynamics, where matrix D_τ is selected as

$$D_\tau = \begin{bmatrix} 0.001 & 0 \\ 0 & 0.001 \end{bmatrix} \tag{52}$$

Besides the large parameter mismatches between the values in the plant model and the nominal values used in the controller given by equation (49), some disturbances are added to the plant model to test the robustness of proposed control algorithms:
(a). the coefficients of viscous friction and coulomb friction in equation (46) are set as $k_{v1} = k_{v2} = 10 Nm/(rad/s)$ and $k_{c1} = k_{c2} = 5Nm$, respectively. The generated friction terms are sufficient large with respect to gravitation forces, centrifugal and Coriolis forces in the plant model.
(b). an additional disturbance torque during 0 ~ 0.15s with constant amplitude of $-100Nm$ is added to both robot joints to test the robustness of the control system in the transition period.

5.3. Simulation results and discussion

The simulation results of the trajectory tracking controller for rigid-body robots presented in Section 2 have been given in (Shi et al., 2008), where different sliding mode control approaches under different system uncertainties are compared. In this section, we discuss only the simulation results for flexible joint robots, which are illustrated by Figure 2 through Figure 5.

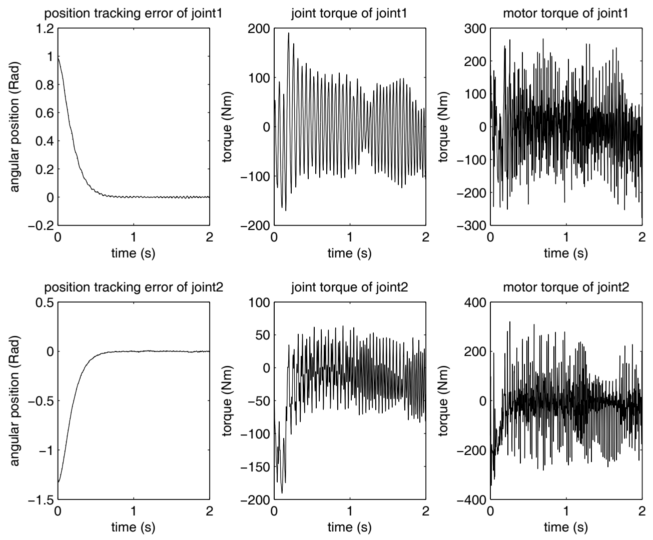


Fig. 2. Pure integral sliding mode control. Left plots: designed and real error dynamics of the link position tracking control (dotted-line: designed, solid-line: real, they are too close to be distinguished); middle plots: joint torque; right plots: required motor torque.

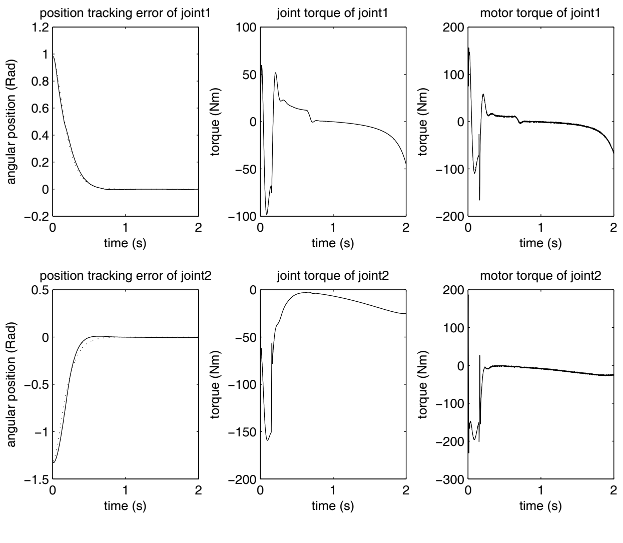


Fig. 3. Integral sliding mode based perturbation estimation approach with constant low-pass filter to extract the equivalent control. Left plots: designed and real error dynamics of the link position tracking control (dotted-line: designed, solid-line: real); middle plots: joint torque; right plots: required motor torque.

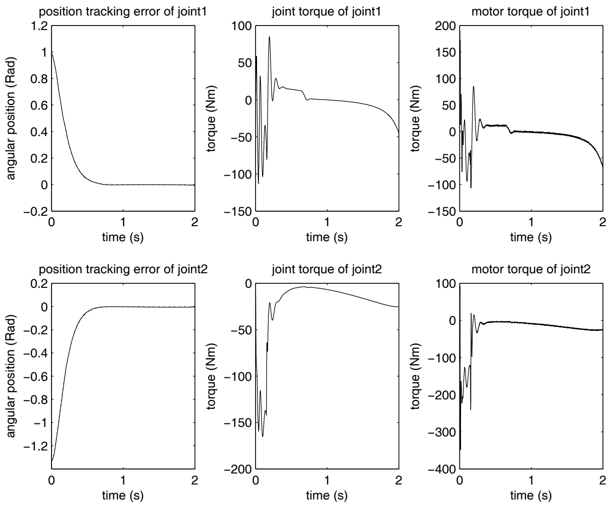


Fig. 4. Integral sliding mode based perturbation estimation approach with time varying low-pass filter to extract the equivalent control. Left plots: designed and real error dynamics of the link position tracking control (dotted-line: designed, solid-line: real, they are too close to be distinguished); middle plots: joint torque; right plots: required motor torque.

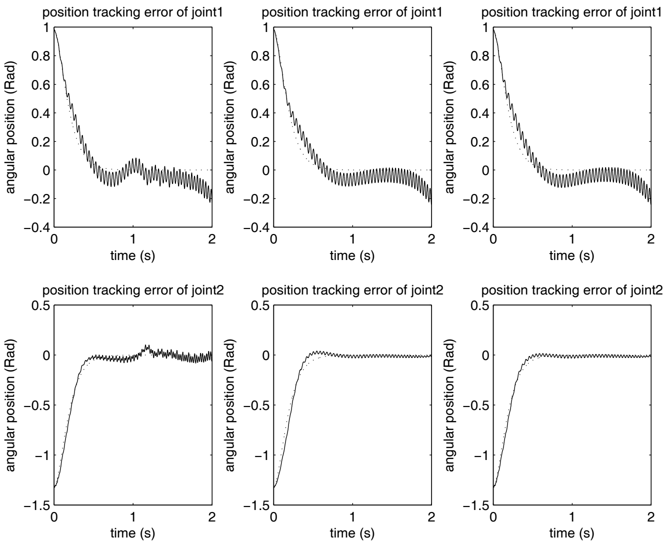


Fig. 5. Designed and real error dynamics of the link position tracking control (dotted-line: designed, solid-line: real) of the three control approaches, but without the singular perturbation treatment on the joint flexibility. Left plots: pure integral sliding mode control; middle plots: perturbation estimation approach with constant low-pass filter; right plots: perturbation estimation approach with time varying low-pass filter.

With Figure 2 the pure integral sliding mode control approach i.e. the controller given by equation (18) is demonstrated. For rigid-body robots, this controller has a prefect tracking control performance despite the large torque disturbance during the transition period (see (Shi et al., 2008)), but for flexible joint robots, the steady-state responses are not smooth enough, see both left plots in Figure 2. Similar to the case of rigid-body robots, there are high frequency oscillations in the joint torque and in the motor torque. The oscillation frequency for flexible joint robots is lower than the one for rigid-body robots because of the joint flexibility. For both types of robots this controller can not be used in practice due to the high level of chattering.
In Figure 3, the simulation result of the controller given by equation (19) is presented, where the low-pass filter is the first order linear filter given by equation (50) with constant μ. As one can see from the middle and right plots of figure, the joint torque and the required motor torques are smoothed significantly due to the perturbation estimation solution (implying that this controller can be applied to real world robot systems). However, the performance of the position tracking control is decreased a little bit, see the both left plots of the figure.
To recover the tracking control performance while keeping the joint torques and motor torques as smooth as possible, the time constant of the low-pass filter is made time varying according to equation (51). The simulation result is illustrated in Figure 4. Now, the position tracking control has a higher control accuracy, in both transition period and steady-state, see the both left plots of Figure 4. From the middle and right plots of Figure 4, one can see that the joint torques and the motor torques are still smooth enough, only in the transition period the frequency of these signals is higher than the case of Figure 3, because of the smaller time constant of the low-pass filter in this time range. Therefore, by tuning the time constant of the low-pass filter, the overall system performance can be improved.
The control approaches demonstrated by the simulation results given by Figure 2 through Figure 4 are supported by the singular perturbation treatment on the joint flexibility. Without this treatment, none of the control approaches can work properly, see Figure 5 (where all elements of matrix D_τ are set to zero). Therefore, joint torque signal as well as its time derivative is very important for the control of flexible joint robots.

6. Conclusion

The robust position tracking controller based on integral sliding mode for rigid-body manipulators is extended to the position tracking control of lightweight manipulators as well as flexible joint robots. Moreover, the control system takes the dynamics of joint motors into account. The joint flexibility is solved by singular perturbation approach which needs no parameter from the controlled system. Also, the current controller for the joint motors is a robust controller without involving the parameters of the electric motors and decoupling process. By using sliding mode PWM technique the current controller overcomes the disadvantages associated with the conventional build-in PWM in micro-processors or DSPs. For the link position tracking control only some rough nominal values are required. It is possible to achieve the pole-placement design without the exact knowledge about the manipulator system to be controlled. Moreover, the control design is mathematically easy and straightforward without involving the properties of the robot dynamics. The resulting control algorithms are simple enough for real-time implementation. The tradeoff between chattering reduction and system robustness can be adjusted by the time constant of a low-pass filter. As the chattering level being significantly reduced, the control algorithms are applicable to real-life systems. Comparative simulation studies have confirmed the effectiveness of proposed control approaches and showed the potential toward the control of lightweight manipulators for high performance applications. Moreover, the presented design methodology can also be applied to other non-linear multi-variable dynamic systems.

7. References

Albu-Schaeffer, A. & Hirzinger, G. (2000). State feedback controller for flexible joint robots: a globally stable approach implemented on DLR's lightweight robots, *IEEE International Conference on Intelligent Robotic Systems*, pp. 1087-1093, ISBN: 0-7803-6348-5, Takamatsu, Japan, 2000

Albu-Schaeffer, A. (2002). Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme, PhD Thesis of TU Munich, 2002

Arimoto, S. (1994). State-of-the-art and future research direction of robot control, *Proceedings of 4th IFAC Symposium on Robot Control*, pp. 3-14, Capri, Italy, Sept. 1994

Cao, W. & Xu, J. (2004). Nonlinear integral-type sliding mode surface for both matched and unmatched uncertain systems, *IEEE Trans. Autom. Control*, Vol. 49, No. 8, Aug. 2004, pp. 1355–1360, ISSN : 0018-9286

Castaños, F. & Fridman, L. (2006). Analysis and design of integral sliding manifolds for systems with unmatched perturbations, *IEEE Transaction on Automatic Control*, Vol.51, No.5, 2006, pp.853-858, ISSN : 0018-9286

Chen, Y.-F.; Mita, T. & Wakui, S. (1990). A new and simple algorithm for sliding mode control of robot arms, *IEEE Trans. on Automatic Control*, Vol. 35, No. 7, 1990, pp. 828-829, ISSN: 0018-9286

De Luca, A. & Lucibello, P. (1998). A general algorithm for dynamic feedback linearization of robots with elastic joints, *IEEE International Conference of Robotics and Automation*, pp. 504-510, ISBN: 0-7803-4300-X, Leuven, Belgium, May 1998

Gilbert, E. & Ha, I. (1983). An approach to nonlinear feedback control with applications to Robotics, *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 22, Dec.1983, pp. 134-138

Gombert , B.; Hirzinger, G.; Plank, G.; Schedl, M. & Shi, J. (1995). Modular concepts for the new generation of DLR's light weight robots, *Proc. Third Conference on Mechatronics and Robotics*, pp. 30-43, 1995

Hirzinger, G.; Gombert, B.; Dietrich, J. & Shi, J. (1994). Transferring space robot technologies into terrestrial applications, *Proceedings of 25th International Symposium on Industrial Robots*, 1994

Hirzinger, G. ; Albu-Schaeffer, A. ; Haehnle, M. ; Schaefer, I. & Sporer, N. (2001). On a new generation of torque controlled light-weight robots, *IEEE International Conference of Robotics and Automation*, Vol. 4, 2001, pp. 3356-3363, ISSN: 1050-4729

Hirzinger, G.; Butterfass, J.; Grebenstein, M.; Haehnle, M.; Schaeferund, I. & Sporer, N. (2001). Space robotics-driver for a new mechatronic generation of lightweight arms and multi-fingered hands, *AIM*, Vol. 2, pp. 1160-1168, ISBN: 0-7803-6736-7, 2001, Como, Italy

Hunt, L.; Su, R. & Meyer, G. (1983). Global transformation of nonlinear systems, *IEEE Trans. on Automatic Control*, Vol. 28, No. 1, 1983, pp. 24-31, ISSN: 0018-9286

Koeppe, R. & Hirzinger, G. (2001). From human arms to a new generation of manipulators: control and design principles, *ASME Int. Mechanical Engineering Congress*, 2001

Lin, T. & Goldenberg, A.A. (1995). Robust adaptive control of flexible joint robots with joint torque feedback, *IEEE International Conference of Robotics and Automation*, Vol. 1, No. 4, May. 1995, pp. 1229-1234, ISSN: 1050-4729

Marino, R. & Spong, M. (1986). Nonlinear control techniques for flexible joint manipulators: a single link case study, *IEEE International Conference of Robotics and Automation*, Vol. 3, Apr. 1986, pp. 1030-1036

Ott, C. (2008). *Cartesian impedance control of redundant and flexible-joint robots,* Springer

Ozgoli, S. & Taghirad, H. D. (2006). A survey on the control of flexible joint robots, *Asian Journal of Control*, Vol. 8, No. 4, pp. 332-344, December 2006

Poznyak, A.; Fridman, L. & Bejarano, F. J. (2004). Mini-max integral sliding mode control for multimodel linear uncertain systems, *IEEE Trans. Autom. Control*, Vol. 49, No. 1, Jan. 2004, pp. 97–102, ISSN: 0191-2216

Readman, Mark C. (1994). *Flexible Joint Robots*, CRC Press

Slotine, J.-J.-E. (1985). The robust control of robot manipulators, *Int. Journal of Robotics Research*, No. 4, 1985, pp. 49-64

Slotine, J.-J.-E. & Li, W. (1987). On the adaptive control of robot manipulators, *Int. Journal of Robotics Research*, No. 6, 1987, pp. 49-59, ISSN:0278-3649

Schmidt, G. (2000). Lecture note: Grundlagen intelligenter roboter, TU München, Lehrstuhl fuer Steuerungs- und Regelungstechnik

Shi, J. & Lu, Y.S. (1996). Field-weakening operation of cylindrical permanent-magnet motors, *IEEE International Conference on Control Applications*, pp. 864-869, ISBN: 0-7803-2975-9, Dearborn, MI, USA, September 1996

Shi, J.; Albu-Schaeffer, A. & Hirzinger, G. (1998). Key issues in dynamic control of lightweight robots for space and terrestrial applications, *IEEE International Conference of Robotics and Automation*, pp. 490-498, ISBN: 0-7803-4300-X, Leuven, Belgium, May 1998

Shi, J.; Liu, H. & Bajcinca, N. (2008). Robust control of robotic manipulators based on integral sliding mode, *International Journal of Control*, Vol. 81, No. 10, October 2008, pp.1537–1548, ISSN : 0020-7179

Spong, M.(1987). Modeling and control of elastic joint robots, *IEEE Journal of Robotics and Automation*, Vol. RA-3, No.4, 1987, pp. 291-300

Spong, M.(1988). Variable structure control of flexible joint manipulators, *IEEE Journal of Robotics and Automation*, Vol. 3, No. 2, 1988, pp.57-64

Spong, M.(1989). Adaptive control of flexible joint manipulators, *Systems and Control Letters*, No.13, 1989, pp. 15-21

Stieber, M. ; Sachdev, S. & Lymer, J. (2000). Robotics architecture of the mobile servicing system for the international space station, *International Symposium of Robotics Research*, 2000, pp. 416-421

Tomei, P. (1991). A simple PD controller for robots with elastic joints, *IEEE Transactions on Automatic Control*, Vol. 36, No. 10, 1991, pp.1208-1213, ISSN: 0018-9286

Utkin, V.I. (1992). *Sliding Modes in Control and Optimization*, London, UK: Springer-Verlag

Utkin, V.I. & Shi, J. (1996). Integral sliding mode in systems operating under uncertainty conditions, *IEEE Conf. On Decision and Control*, pp. 4591-4596, ISBN: 0-7803-3590-2, Kobe (Japan), Dec. 1996

Utkin, V.I. ; Guldner, J. & Shi, J. (2009). *Sliding Mode Control in Electromechanical Systems*, Taylor & Francis publisher, (Second Edition)

Young, K.-K.D. (1978). Controller design for a manipulator using theory of variable structure systems, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 8, No. 2, Feb.1978, pp. 210-218, ISSN: 0018-9472

Young, K.-K.D. (1988). A variable structure model following control design for robotic applications, *IEEE Journal on Robotics and Automation*, Vol. 4, Oct. 1988, pp. 556-561, ISSN: 0882-4967

Coordinate Transformation Based Contour Following Control for Robotic Systems

Chieh-Li Chen and Chao-Chung Peng
Department of Aeronautics and Astronautics, National Cheng Kung University
No. 1 University Road, Tainan City 701,
Taiwan

1. Introduction

Robots are important for industrial automation. Similar to CNC machining, robotic systems can be applied to numerous applications such as material assembling, welding, painting, manufacturing and so on. For control of robot manipulators, a conventional way is to establish their mathematical models in the joint space and therefore precise positioning of end-effector relies on control performance in the joint space.
In terms of utilizations of industrial robots, it is well known that the position of end-effector is an important factor and significantly dominates the quality of final product. Based on given trajectories in work space (also known as Cartesian space), there are two main approaches for manipulator motion control, 1) one is to determine the desired joint space positions by solving the inverse kinematics, 2) the other is to consider the dynamic model in the work space directly (Feng & Palaniswami, 1993). In both approaches, tracking performance should be good enough in order to follow real time commands and achieve prescribed contours. In respect of the joint space approach, for instance, providing the tracking errors of each joint position can not be eliminated well, the end-effector can not track the desired profile precisely; to put it another way, once good tracking capability in any one of robot arms can not be guaranteed, the synchronization control task fails and thereby gives rise to unsatisfactory machining result. Consequently, it is worthy to investigate a better control strategy to guarantee the machining quality even in the presence tolerable tracking errors. To achieve this goal, we have to deviate the topic regarding control of robotic systems for a moment and address a certain core issues about contouring control systems.

1.1 Definition of contour error

First of all, a machining quality index called contour error is introduced. As shown in Fig. 1, the contouring error ε_p corresponding to point $P(t)$ is defined geometrically as the shortest distance from $P(t)$ to the desired contour D and can be written as $\varepsilon_p = \min_D \left\| P^* - P(t) \right\|$, where $P(t) \in S$ is the actual position of end-effector at time t and $P^* \in D$. Once the shortest target P^* is found, (ideal) resultant control effort will force $P(t)$ moving towards P^* firstly and then keep it attaching on the prescribed path. This control behavior involves two stages i.e., normal and tangential errors reduction. This concept is clear and has been widely used in contouring control designs (Koren, 1980; Chin & Lin, 1997; Ho et al., 1998; Yeh & Hsu, 1999; Chiu & Tomizuka, 2001, Shih et al., 2002; Wang & Zhang, 2004; Hsieh et al., 2006; Peng & Chen, 2007a; Peng & Chen, 2007b; Chen & Lin, 2008).

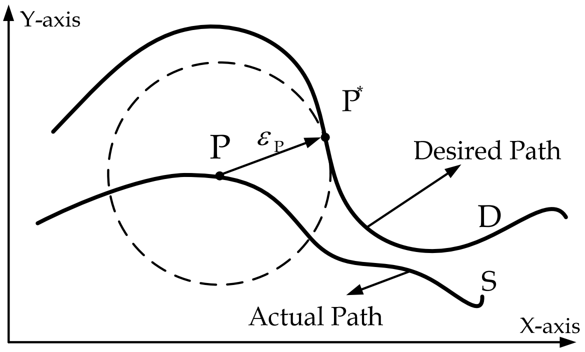


Fig. 1. Definition of contour error.

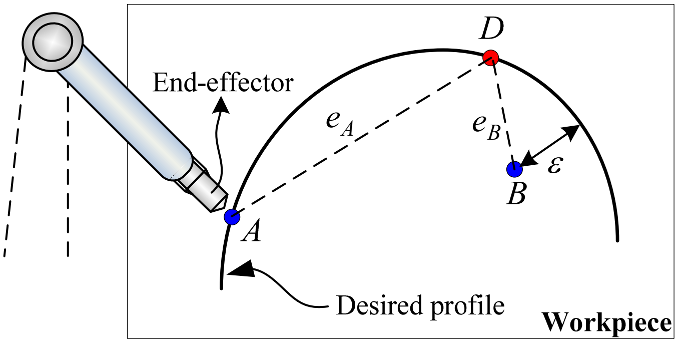


Fig. 2. Special issues appeared in contour following control systems.

1.2 Tracking and contouring control

Based on the definition of contour error, the following is to illustrate a main discrepancy between tracking control and contouring control. Consider Fig. 2, suppose that *A* is the location of the end-effector and *D* is the corresponding command position at a certain time instant. Providing the tracking error e_A exists significantly, a resultant tracking control force, where the orientation is towards from *A* to *D*, is going to be generated by controllers such that an undesirable working path away from the desired profile is induced. Moreover, Fig. 2 also reveals that a good tracking performance is just a sufficient, but not necessary, condition to reach good contouring performance. For example, the tracking error at point *A* is apparently larger than the one at point *B*, but from contouring control point of view, contouring error is defined as the shortest distance from the end-effector to the desired path, so the position *A* is preferable in the contouring process rather than *B*. A couple of features and problems of contour control systems can be found in the tutorial study by Ramesh et al (2005).

1.3 Contouring control strategies

In terms of the literature on contouring control strategies, a well known cross-coupling control (CCC) structure (Koren, 1980) has been widely applied. Using this framework, contouring error attenuation can be carried out by designing various control strategies. For example, to raise machining speed, a modified CCC structure, where an additional pre-compensated controller is involved, called cross-coupled pre-compensation method (CCPM) (Chin & Lin, 1997) was proposed. However, these control components, namely tracking controller, contouring controller and feed-forward controller, are usually designed independently such that the coupling effects between them are not manifest. It is difficult to distinguish which one dominates the eventual contouring result. Thus, a systematic design procedure for CCC structure is highly desirable. To this end, transfer function methods for CCC structure are proposed (Yeh & Hsu, 1999; Shih et al., 2002; Zhong et al., 2002). These methods make the CCC design into a unit feedback control problem and offer a systematic analysis for stability and performance of a linear contouring system. However, superior tracking level is still needed to confirm final contouring qualities when these approaches are utilized.

Some contouring control architectures, which involve coordinate transformation techniques, such as tangent-normal (*T-N*) coordinate frame (Ho et al., 1998; Chiu & Tomizuka, 2001; Wang & Zhang, 2004; Hsieh et al., 2006) and polar coordinate frame (Chen et al., 2002) have been presented in recent years. For *T-N* coordinate transformation approaches, contouring dynamics are decomposed into tangential and normal directions, where the tangential dynamics is associated with feed-rates while the normal dynamics is relevant to contouring error. Nevertheless, only the contour error of straight paths can be evaluated exactly. The normal errors just stand for approximated contour errors for arbitrary non-linear curves. On the other hand for polar coordinate transformation scheme, dynamics of radial orientation is derived in consideration of both circular and noncircular profiles. The polar coordinate based contouring control framework is adequate for circular profiles. For non-circular profiles, the precision of contouring error estimation relies upon that the radius variation with respect to angle change is small. However, for a given straight line which is (almost) perpendicular to x-axis may not satisfy this assumption.

For the preceding coordinate transformation based contouring control schemes, the main advantage is that the goal of contouring controller design becomes clear and simple; in other words, a tangential controller focuses on maintaining a desired feed-rate while a normal/radial controller is applied to eliminate normal/radial errors. However, good contouring performance still depends on good tracking results when applying the *T-N* coordinate transformation methods. Once a large tracking error occurs, the contouring quality will be degraded considerably.

In order to complete contouring tasks efficiently and accurately, an adequate control strategy is prerequisite. Computed torque method, which uses the nominal dynamic model of the robots to decouple and to linearize the nonlinear system, is one of the well-known approaches. However, when utilizing the computed torque method only, the resultant performance may not be satisfactory due to the effects caused by lump perturbations including uncertain parameters, modeling errors, load variations and nonlinear friction effects. For the sake of system robustness enhancement, several robust control theories such as learning control (Sun et al., 2006), H-infinity (Fang & Chen, 2002), sliding mode control (SMC) (Zhu et al., 1992; Chen & Xu, 1999) and adaptive control (Slotine & Li, 1988; Dong & Kuhnert, 2005; Lee et al., 2005) have been proposed. In recent decades, SMC and adaptive algorithm have been widely used to control of robot systems. Known as robust and accurate, SMC is a good candidate when systems are interfered by model uncertainties and exogenous disturbances. On the other hand, adaptive algorithm, which possesses learning behavior, is capable of estimating uncertain parameters when the parameters are not precisely known. Therefore in this study, by considering well known backstepping design technique, an integral type SMC is designed together with an adaptation law such that the controlled robotic system is robust against lumped perturbations and an adequate switching gain used in sliding controller is determined systematically without try and error and tedious analysis.

2. Main Concerned Issues and Contributions on Robot Contouring Control Systems

For concerning control of robot manipulator, the main control objective is to control the motion trajectory of the end-effector following a prescribed contour. For conventional trajectory control, a given profile in work space is decoded into independent reference joint positions and the success in contouring control task depends on tracking capability of individual robot joint. However, as argued in the preceding section, once one of the robot joint does not perform good tracking result, the end-effector may deviate from the desired path seriously. Therefore, the contouring control problem on a multi-link robot manipulator leads to a synchronization task of joint position, which can also be referred to as master-slave control scheme (Sarachik & Ragazzini, 1957). Moreover, it has been illustrated by a couple of experiments that good contouring quality doesn’t necessary depend on good tracking performance (Peng & Chen, 2007a). Consequently for manipulator contouring control, how to relax tracking capability and preserve contouring precision becomes the main concerned issue in this chapter.
To fulfill the main object mentioned above, some short-term tasks must be considered:

(i) Define an equivalent contour error.
(ii) Derive a contour error model for robotic system.
(iii) Design robust contouring controllers.

For task (i), it is well known that a closed-form solution of the exact contour error for an arbitrary curve is quite difficult to formulate. Thus, approximated contour error estimations are usually applied. Due to the use of inexact contour error models, attenuation of the approximated contour errors may not guarantee the elimination of real contour error especially in the presence of large tracking errors. As a result, an adequate equivalent error index should be defined. In this work, a new error variable named contour index (CI) is given by means of a geometric way in a virtual contouring space (VCS). The properties include: i) the CI is able to act as an equivalent contouring error and replace the normal tracking error (Hsieh et al., 2006, Chen & Lin, 2008) to be a new performance index without causing geometric over estimation and ii) the CI contributes to contour following behavior (Peng & Chen, 2007a). This phenomenon can be illustrated by referring Fig. 2, where the end-effector at A approaches to the real time command D through the desired profile instead of moving along $\overrightarrow{\mathrm{AD}}$ directly.

Regarding task (ii), the design processes are summarized as follows: dynamic correlation between joint space and work space is firstly established. The work space dynamics are further derived into the VCS, which consists of tangential and normal dynamics. Then, a dynamic equation of CI, which offers the end-effector to trail the prescribed path, is derived. Finally, based on the developed error dynamics, an adaptive sliding controller together with the idea of inverse dynamics control (Spong & Vidyasagar, 1989) is applied to provide system robustness against lumped uncertainties and fulfill the main control object.

The advantage of the proposed method for manipulator contouring control problem contains:

(1) The derivation of contour error model for robotic system is systematic
(2) The closed-loop stability analysis is clear.
(3) Final contouring quality can be maintained even if the end-effector doesn't track the real time command very well.
(4) The proposed method is also extendable to three-dimensional case (Peng & Chen, 2007b).

3. Proposed Contouring Control Framework for a Robot Manipulator

3.1 Forward kinematics

By using the Euler-Lagrange method, a dynamic equation of a vertical robot arm can be expressed as

$$\mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{G}(\boldsymbol{\theta}) = \boldsymbol{\tau} - \mathbf{F} \tag{1}$$

where $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$ and $\ddot{\boldsymbol{\theta}} \in \Re^{n}$ are the joint position, velocity and acceleration vectors, respectively. $\mathbf{M}(\boldsymbol{\theta}) \in \Re^{n\times n}$ is a positive definite inertia matrix; $\mathbf{C}(\boldsymbol{\theta},\dot{\boldsymbol{\theta}}) \in \Re^{n\times n}$ is a matrix containing Coriolis and centrifugal terms; $\mathbf{G}(\boldsymbol{\theta}) \in \Re^{n}$ stands for gravitational term; $\boldsymbol{\tau} \in \Re^{n}$ represents a torque vector and $\mathbf{F} \in \Re^{n}$ is a disturbance vector referred to nonlinear friction effects. In this study, a two degree of freedom robot ($n = 2$) is considered and depicted in Fig. 3. The position translation from joint space to work space can be calculated by considering forward kinematics as follows,

$$\mathbf{P_a} = \mathrm{h}(\boldsymbol{\theta}(\mathrm{t})) \tag{2}$$

Taking the twice time derivative of (2) gives

$$\ddot{\mathbf{P}}_\mathbf{a} = \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} + \mathbf{J}\ddot{\boldsymbol{\theta}} \tag{3}$$

where $\mathbf{P_a} = [\mathrm{x} \quad \mathrm{y}]^{T}$ is the end-effector position vector in work space, $\mathbf{J}$ is the Jacobian matrix and T denotes the transpose. Note that the Jacobian matrix is assumed to be fully known for position measurement and is nonsingular during the whole contouring control processes. The definition of matrices used in (2) and (3) are listed in Appendix.

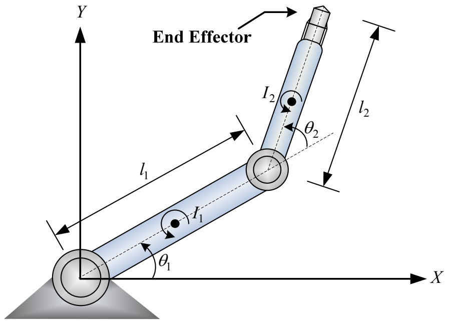


Fig. 3. Two degree of freedom robot.

3.2 Contour generation

Consider a desired profile in work space generated by

$$\mathbf{P_d} = \mathbf{T_\beta \kappa \overline{P}_d} + \mathbf{c_o} \tag{4}$$

where $\mathbf{P_d} = [x_d \quad y_d]^T$ and $\mathbf{\overline{P}_d} = [\overline{x}_d \quad \overline{y}_d]^T$ are position vectors in work space and VCS, respectively. As an example, a standard unit circle in VCS is made up of $\overline{x}_d = \sin(2\pi ft)$ and $\overline{y}_d = \cos(2\pi ft)$, where the frequency f is relative to tangential velocity. $\mathbf{T_\beta}$ denotes a rotational matrix with a angle β and the diagonal gain matrix $\boldsymbol{\kappa}$ (please see Appendix) is relative to the length of major axis and minor axis of the desired profile. The term $\mathbf{c_o} = [c_{xo} \quad c_{yo}]^T$ denotes a shifting operator. The geometric meaning of (4) is depicted in Fig. 4, where an oblique ellipse can be generated from a unit circle by the matrix operation, $\mathbf{T_\beta \kappa}$.

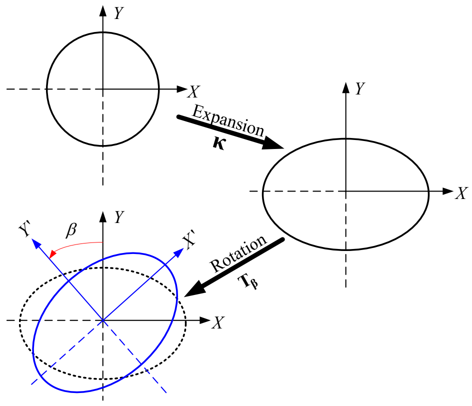


Fig. 4. Command generation via matrix operation.

3.3 Derivation of contour error dynamics

Define the tracking error in work space as $\mathbf{e} = \mathbf{P_d} - \mathbf{P_a} = [\mathrm{e_x} \quad \mathrm{e_y}]^T$ and then the resulting error dynamics is

$$\ddot{\mathbf{e}} = \ddot{\mathbf{P}}_\mathbf{d} - \ddot{\mathbf{P}}_\mathbf{a} \tag{5}$$

Subsequently, the error relationship between VCS and work space can be represented by

$$\boldsymbol{\varepsilon} = \mathbf{T}(\mathbf{T}_\beta \boldsymbol{\kappa})^{-1} \mathbf{e} \tag{6}$$

where $\boldsymbol{\varepsilon} = [\varepsilon_\mathrm{n} \quad \varepsilon_\mathrm{t}]^T$ denotes a transformed error vector including normal tracking error and tangential tracking error. The projection-matrix $\mathbf{T}$ is orthogonal, continuous and differentiable, defined by

$$\mathbf{T} = \begin{bmatrix} -\sin\gamma & \cos\gamma \\ \cos\gamma & \sin\gamma \end{bmatrix} \tag{7}$$

where γ is an instantaneous inclination of the tangential direction corresponding to the circular profile shown in Fig. 5. From the definition of tracking error in work space, Eq. (6) can be rewritten as

$$\boldsymbol{\varepsilon} = \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}(\mathbf{P_d} - \mathbf{P_a}) = \mathbf{T}\underbrace{\left[\overline{\mathbf{P}}_\mathbf{d} - \overline{\mathbf{P}}_\mathbf{a}\right]}_{\mathrm{E}} + \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\mathbf{c_o} \tag{8}$$

where $\overline{\mathbf{P}}_\mathbf{a} = (\mathbf{T}_\beta \boldsymbol{\kappa})^{-1}\mathbf{P_a}$ and $\mathbf{E} = [\mathrm{E_x} \quad \mathrm{E_y}]^T$. The corresponding geometrical meaning is shown in Fig. 5. Eq. (8) means that the information in the work space is transformed into the VCS through $\mathbf{T}(\mathbf{T}_\beta \boldsymbol{\kappa})^{-1}$; namely, the vectors $\mathbf{P_a}$, $\mathbf{P_d}$ and $\|\mathbf{e}\|$ in the work space are transformed to be $\overline{\mathbf{P}}_\mathbf{a}$, $\overline{\mathbf{P}}_\mathbf{d}$ and $\|\mathbf{E}\|$ in VCS, respectively. Then the error vector $\|\mathbf{E}\|$ can be further decomposed into tangential and normal directions through the matrix $\mathbf{T}$.

Fig. 5 shows that a moving *T-N* coordinate attaching to the profile guides $\overline{\mathbf{P}}_\mathbf{a}$ to follow $\overline{\mathbf{P}}_\mathbf{d}$ along the profile. In the control point of view, Eq. (8) also indicates that $\mathbf{P_d} - \mathbf{P_a} = 0$ if and only if $\boldsymbol{\varepsilon} = 0$. Therefore, the control problem turns into a regulation problem in both tangential and normal directions.

From (6)-(7), since $\mathbf{T}^{-1} = \mathbf{T}$, it follows

$$\ddot{\boldsymbol{\varepsilon}} + \mathbf{2T\dot{T}}\dot{\boldsymbol{\varepsilon}} + \mathbf{T\ddot{T}}\boldsymbol{\varepsilon} = \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\ddot{\mathbf{e}} \tag{9}$$

Substituting (5) into (9) yields

$$\begin{aligned} \ddot{\boldsymbol{\varepsilon}} + \mathbf{2T\dot{T}}\dot{\boldsymbol{\varepsilon}} + \mathbf{T\ddot{T}}\boldsymbol{\varepsilon} &= \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\left(\ddot{\mathbf{P}}_\mathbf{d} - \ddot{\mathbf{P}}_\mathbf{a}\right) \\ &= \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\ddot{\mathbf{P}}_\mathbf{d} - \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\left(\dot{\mathbf{J}}\dot{\boldsymbol{\theta}} + \mathbf{J}\ddot{\boldsymbol{\theta}}\right) \\ &= \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\ddot{\mathbf{P}}_\mathbf{d} - \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\dot{\mathbf{J}}\dot{\boldsymbol{\theta}} - \mathbf{T}\left(\mathbf{T}_\beta \boldsymbol{\kappa}\right)^{-1}\mathbf{J}\left[\mathbf{M}^{-1}\left(\boldsymbol{\tau} - \mathbf{f} - \mathbf{C}\dot{\boldsymbol{\theta}} - \mathbf{G}\right)\right] \\ &= \boldsymbol{\Sigma} + \mathbf{f}_\varepsilon - \boldsymbol{\Gamma}\boldsymbol{\tau} \end{aligned} \tag{10}$$

where $\mathbf{\Sigma} = \mathbf{T}(\mathbf{T}_\beta \boldsymbol{\kappa})^{-1}(\ddot{\mathbf{P}}_d - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}}) + \mathbf{\Gamma}(\mathbf{C}\dot{\boldsymbol{\theta}} + \mathbf{G})$, $\mathbf{f}_\varepsilon = \mathbf{\Gamma}\mathbf{f}$ and $\mathbf{\Gamma} = \mathbf{T}(\mathbf{T}_\beta \boldsymbol{\kappa})^{-1}\mathbf{J}\mathbf{M}^{-1}$. Eq. (10) represents the error dynamics in the VCS.

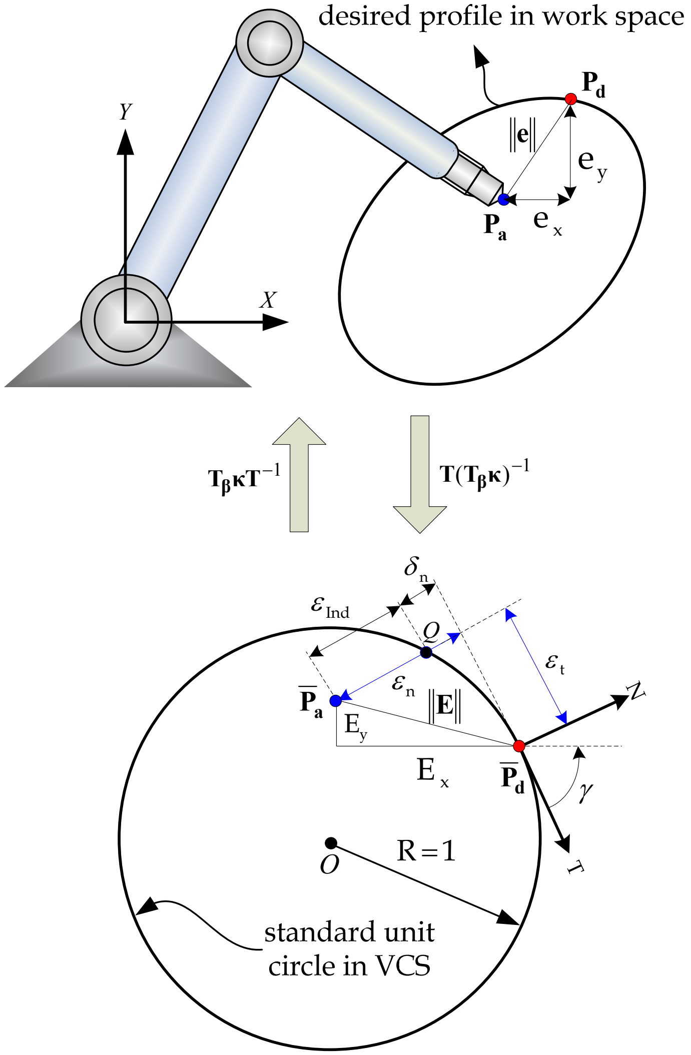

desired profile in work space
$\mathbf{P_d}$
$\|\mathbf{e}\|$
e_y
$\mathbf{P_a}$
e_x
Y
X
$\mathbf{T_\beta \kappa T}^{-1}$
$\mathbf{T(T_\beta \kappa)}^{-1}$
δ_n
ε_{Ind}
Q
ε_t
ε_n
$\overline{\mathbf{P}}_\mathbf{a}$
$\|\mathbf{E}\|$
N
E_y
E_x
$\overline{\mathbf{P}}_\mathbf{d}$
γ
O
$R=1$
T
standard unit circle in VCS

Fig. 5. Robot contouring system.

Consider the inverse dynamics compensation method (Spong & Vidyasagar, 1989), a nominal control torque $\boldsymbol{\tau} = [\tau_1 \quad \tau_2]^T$ for (10) is chosen as the form

$$\boldsymbol{\tau} = \hat{\mathbf{\Gamma}}^{-1}(\hat{\mathbf{\Sigma}} - \boldsymbol{\tau}_\varepsilon) \tag{11}$$

where $\hat{\mathbf{\Gamma}}$ and $\hat{\mathbf{\Sigma}}$ are denoted as the nominal value of $\mathbf{\Gamma}$ and $\mathbf{\Sigma}$ (i.e., $\mathbf{\Gamma} = \hat{\mathbf{\Gamma}} + \tilde{\mathbf{\Gamma}}$, $\mathbf{\Sigma} = \hat{\mathbf{\Sigma}} + \tilde{\mathbf{\Sigma}}$), respectively and $\boldsymbol{\tau}_{\varepsilon} = [\tau_{\varepsilon n} \quad \tau_{\varepsilon t}]^{\mathrm{T}}$.
Substituting (11) into (10) yields

$$\begin{aligned} \ddot{\boldsymbol{\varepsilon}} + 2\mathbf{T}\dot{\mathbf{T}}\dot{\boldsymbol{\varepsilon}} + \mathbf{T}\ddot{\mathbf{T}}\boldsymbol{\varepsilon} &= \mathbf{\Sigma} + \mathbf{f}_{\varepsilon} - \mathbf{\Gamma}\boldsymbol{\tau} \\ &= \mathbf{\Sigma} + \mathbf{f}_{\varepsilon} - \hat{\mathbf{\Gamma}}\left[\hat{\mathbf{\Gamma}}^{-1}\left(\hat{\mathbf{\Sigma}} - \boldsymbol{\tau}_{\varepsilon}\right)\right] - \tilde{\mathbf{\Gamma}}\boldsymbol{\tau} \\ &= \tilde{\mathbf{\Sigma}} - \tilde{\mathbf{\Gamma}}\boldsymbol{\tau} + \mathbf{f}_{\varepsilon} + \boldsymbol{\tau}_{\varepsilon} \\ &= \mathbf{\Lambda} + \boldsymbol{\tau}_{\varepsilon} \end{aligned} \tag{12}$$

where $\mathbf{\Lambda} = \tilde{\mathbf{\Sigma}} - \tilde{\mathbf{\Gamma}}\boldsymbol{\tau} + \mathbf{f}_{\varepsilon} = [\mathbf{\Lambda}_{\mathrm{n}} \quad \mathbf{\Lambda}_{\mathrm{t}}]^{\mathrm{T}}$ represents model uncertainty and external disturbance (for example, the friction effects). Refer to (9), it can be rewritten as

$$\begin{bmatrix} \ddot{\varepsilon}_{\mathrm{n}} \\ \ddot{\varepsilon}_{\mathrm{t}} \end{bmatrix} + 2\begin{bmatrix} 0 & \dot{\gamma} \\ -\dot{\gamma} & 0 \end{bmatrix}\begin{bmatrix} \dot{\varepsilon}_{\mathrm{n}} \\ \dot{\varepsilon}_{\mathrm{t}} \end{bmatrix} - \begin{bmatrix} \dot{\gamma}^2 & -\ddot{\gamma} \\ \ddot{\gamma} & \dot{\gamma}^2 \end{bmatrix}\begin{bmatrix} \varepsilon_{\mathrm{n}} \\ \varepsilon_{\mathrm{t}} \end{bmatrix} = \begin{bmatrix} \mathbf{\Lambda}_{\mathrm{n}} + \tau_{\varepsilon n} \\ \mathbf{\Lambda}_{\mathrm{t}} + \tau_{\varepsilon t} \end{bmatrix} \tag{13}$$

Providing the desired feed-rate is set to be constant for predetermined profiles, it follows $\ddot{\gamma} = 0$.
From Fig. 5, one can find that the mismatched term δ_{n} in normal direction is caused by the existence of ε_{t}. Thus, the elimination of normal tracking error ε_{n} causes an over-cutting segment δ_{n} when $\varepsilon_{\mathrm{t}} \neq 0$. In order to solve this problem, the CI is introduced in the VCS and is defined by

$$\varepsilon_{\mathrm{Ind}} = \varepsilon_{\mathrm{n}} - \delta_{\mathrm{n}} \tag{14}$$

where $\delta_{\mathrm{n}} = \mathrm{R} - (\mathrm{R}^2 - \varepsilon_{\mathrm{t}}^2)^{1/2}$. Note that the CI replaces ε_{n} to be a new performance index or to be regarded as an equivalent contouring error during control process; to put it simply, the purpose of the contouring control is to eliminate $\varepsilon_{\mathrm{Ind}}$ instead of ε_{n} while $\varepsilon_{\mathrm{t}} \neq 0$. Invoking (13) with (14) yields

$$\ddot{\varepsilon}_{\mathrm{Ind}} = \tau_{\varepsilon n} + \eta_1 + \eta_2 + \eta_3 + \mathbf{\Lambda}_{\mathrm{n}} \tag{15}$$

where

$$\begin{cases} \eta_1 = -2\dot{\gamma}\dot{\varepsilon}_{\mathrm{t}} + \dot{\gamma}^2\varepsilon_{\mathrm{n}} - \ddot{\gamma}\varepsilon_{\mathrm{t}} - \left(\varepsilon_{\mathrm{t}}\dot{\varepsilon}_{\mathrm{t}}\right)^2 \cdot \sigma^{-3/2} - \left[\dot{\varepsilon}_{\mathrm{t}}^2 + \varepsilon_{\mathrm{t}}\left(2\dot{\gamma}\dot{\varepsilon}_{\mathrm{n}} + \ddot{\gamma}\varepsilon_{\mathrm{n}} + \dot{\gamma}^2\varepsilon_{\mathrm{t}}\right)\right] \cdot \sigma^{-1/2} \\ \eta_2 = -\varepsilon_{\mathrm{t}}\mathrm{u}_{\mathrm{t}}\sigma^{-1/2} \\ \eta_3 = -\varepsilon_{\mathrm{t}}\mathbf{\Lambda}_{\mathrm{t}}\sigma^{-1/2} \\ \sigma = \mathrm{R}^2 - \varepsilon_{\mathrm{t}}^2 \end{cases}$$

Examining (14) again, it is necessary to remind that for all the control period, the condition $|\varepsilon_{\mathrm{t}}| < \mathrm{R}$ must be guaranteed. Consequently, the maximum allowable tracking error in tangential direction is R , which is not a crucial condition in practice.

Therefore, the modified system dynamics in VCS becomes

$$\ddot{\varepsilon}_{\mathrm{t}} - 2\dot{\gamma}\dot{\varepsilon}_{\mathrm{n}} - \dot{\gamma}^2\varepsilon_{\mathrm{t}} = \mathbf{\Lambda}_{\mathrm{t}} + \tau_{\varepsilon\mathrm{t}} \tag{16.a}$$

$$\ddot{\varepsilon}_{\mathrm{Ind}} = \eta_1 + \eta_2 + \eta_3 + \mathbf{\Lambda}_{\mathrm{n}} + \tau_{\varepsilon\mathrm{n}} \tag{16.b}$$

For (16), contouring controller design will be addressed in the next section.

4. Contouring Controller Design

In this section, design of contouring controller is considered separately for tangential and modified normal dynamics. For demonstration purpose, a general proportional-derivative (PD) controller is applied in tangential error equation. It is well known that the PD controller is capable of achieving stabilization and improving transient response, but is not adequate for error elimination. Consequently, tangential tracking errors exist unavoidably. Under this circumstance, we are going to show that precise contouring performance can still be achieved by applying the CI approach. Building on the developed contouring control framework, the tangential and normal control objects can be respectively interpreted as stabilization and regulation problems.

4.1 Design of tangential control effort

Considering the tangential dynamics of (16.a), a PD controller with the form

$$\tau_{\varepsilon\mathrm{t}} = -\mathrm{K}_{\mathrm{Vt}}\dot{\varepsilon}_{\mathrm{t}} - \mathrm{K}_{\mathrm{Pt}}\varepsilon_{\mathrm{t}} - 2\dot{\gamma}\dot{\varepsilon}_{\mathrm{n}} - \dot{\gamma}^2\varepsilon_{\mathrm{t}} \tag{17}$$

is applied. Substituting (17) into (16.a) results in

$$\ddot{\varepsilon}_{\mathrm{t}} + \mathrm{K}_{\mathrm{Vt}}\dot{\varepsilon}_{\mathrm{t}} + \mathrm{K}_{\mathrm{Pt}}\varepsilon_{\mathrm{t}} = \mathbf{\Lambda}_{\mathrm{t}} \tag{18}$$

where K_{Vt} and K_{Pt} are positive real. The selection of control gains should guarantee the criterion $|\varepsilon_{\mathrm{t}}| < \mathrm{R}$. Eq. (18) indicates that the tangential tracking error cannot be eliminated very well due to the existence of $\mathbf{\Lambda}_{\mathrm{t}}$. However, it will be shown that the existence of ε_{t} causes no harm to contouring performance with the aid of CI.

4.2 Design of normal control effort

In the following, an integral type sliding controller for the modified normal dynamics is developed by using backstepping approach. Firstly, let $\varepsilon_{\mathrm{Ind}} = \varepsilon_{\mathrm{Ind1}}$ and define an internal state w. Then the system (16.b) can be represented as

$$\dot{\mathrm{w}} = \varepsilon_{\mathrm{Ind1}} \tag{19.a}$$

$$\dot{\varepsilon}_{\mathrm{Ind1}} = \varepsilon_{\mathrm{Ind2}} \tag{19.b}$$

$$\dot{\varepsilon}_{\mathrm{Ind2}} = \eta_1 + \eta_2 + \eta_3 + \mathbf{\Lambda}_{\mathrm{n}} + \tau_{\varepsilon\mathrm{n}} \tag{19.c}$$

Assume that the system state $\varepsilon_{\mathrm{Ind1}}$ can be treated as an independent input $\phi_1(\mathrm{w})$, and let

$$\varepsilon_{\mathrm{Ind1}} = \phi_1 = -\mathrm{k}_1 \mathrm{w} \tag{20}$$

where $\mathrm{k}_1 > 0$. Then, consider as a Lyapunov function

$$\mathrm{V}_1 = \mathrm{w}^2/2 \tag{21}$$

The derivative of (21) is

$$\dot{\mathrm{V}}_1 = \mathrm{w}\phi_1 = -\mathrm{k}_1\mathrm{w}^2 \le 0 \tag{22}$$

In practice, there may exist a difference between $\varepsilon_{\mathrm{Ind1}}$ and ϕ_1. Hence, define a new error variable by $\mathrm{z}_1 = \varepsilon_{\mathrm{Ind1}} - \phi_1$, which gives

$$\dot{\mathrm{w}} = \mathrm{z}_1 + \phi_1 \tag{23.a}$$

and

$$\dot{\mathrm{z}}_1 = \varepsilon_{\mathrm{Ind2}} - \dot{\phi}_1 \tag{23.b}$$

Second, in a similar manner, consider $\varepsilon_{\mathrm{Ind2}}$ as a virtual control input and let

$$\varepsilon_{\mathrm{Ind2}} = \phi_2 = -\mathrm{w} - \mathrm{k}_2\mathrm{z}_1 + \dot{\phi}_1 \tag{24}$$

Selecting as a Lyapunov candidate

$$\mathrm{V}_2 = \mathrm{w}^2/2 + \mathrm{z}_1^2/2 \tag{25}$$

and taking its time derivative gives

$$\begin{aligned}\dot{\mathrm{V}}_2 &= \mathrm{w}(\mathrm{z}_1 + \phi_1) + \mathrm{z}_1(\varepsilon_{\mathrm{Ind2}} - \dot{\phi}_1)\\ &= \mathrm{w}(\mathrm{z}_1 + \phi_1) + \mathrm{z}_1(\phi_2 - \dot{\phi}_1)\\ &= -\mathrm{k}_1\mathrm{w}^2 - \mathrm{k}_2\mathrm{z}_1^2 \le 0\end{aligned} \tag{26}$$

Note that the criterion (26) is achieved only when the virtual control law (24) comes into effect. Taking the constraint into account, one can design a sliding surface as $\mathrm{z}_2 = \varepsilon_{\mathrm{Ind2}} - \phi_2$, and then the augmented system can be represented as

$$\dot{\mathrm{w}} = \mathrm{z}_1 + \phi_1 \tag{27.a}$$

$$\dot{z}_1 = z_2 + \phi_2 - \dot{\phi}_1 \tag{27.b}$$

$$\dot{z}_2 = \eta_1 + \eta_2 + \eta_3 + \mathbf{\Lambda}_n + \tau_{sn} - \dot{\phi}_2 \tag{27.c}$$

Suppose that the parameter uncertainties and external disturbances satisfy the inequality $\max(|\eta_3 + \mathbf{\Lambda}_n|) < \Omega$, where Ω is an unknown positive constant, then the final control object is to develop a controller that provides system robustness against Ω.
Design a control law in the following form

$$\tau_{sn} = -\eta_1 - \eta_2 - z_1 - k_3 z_2 + \dot{\phi}_2 - \xi\,\mathrm{sgn}(z_2) \tag{28}$$

where $\xi \geq \Omega$ denotes the switching gain. Select a Lyapunov candidate as

$$V_3 = w^2/2 + z_1^2/2 + z_2^2/2 \tag{29}$$

From (28) and (29), one can obtain

$$\begin{aligned}\dot{V}_3 &\leq w(z_1 + \phi_1) + z_1(z_2 + \phi_2 - \dot{\phi}_1) + z_2(\eta_1 + \eta_2 - \dot{\phi}_2 + \tau_{sn}) + \Omega|z_2| \\ &\leq -k_1 w^2 - k_2 z_1^2 - k_3 z_2^2 - z_2 \xi\,\mathrm{sgn}(z_2) + \Omega|z_2| \\ &= -2kV_3 - |z_2|(\xi - \Omega) \leq -2kV_3\end{aligned} \tag{30}$$

where $k_1 = k_2 = k_3 = k$ is applied. Therefore, system (27) is exponentially stable by using the control law (28) when the selected ξ satisfies $\xi > \Omega$. Since the upper bound of Ω may not be efficiently determined, the following well known adaptation law (Yoo & Chung, 1992), which dedicates to estimate an adequate constant value ξ_a, is applied

$$\dot{\hat{\xi}}_a = \Psi z_2 \mathrm{sat}(z_2, \rho) \tag{31}$$

where $\hat{\xi}_a$ is denoted as an estimated switching gain and

$$\begin{cases}\Psi > 0 & |z_2| > \rho \\ \Psi = 0 & , |z_2| \leq \rho\end{cases} \tag{32}$$

stands for an adaptation gain, where the use of dead-zone is needed due to the face that the ideal sliding does not occur in practical applications. For chattering avoidance, the discontinuous controller (28) is replaced by

$$\tau_{sn} = -\eta_1 - \eta_2 - z_1 - k_3 z_2 + \dot{\phi}_2 - \hat{\xi}_a \mathrm{sat}(z_2, \rho) \tag{33}$$

The saturation function is described as follows

$$\mathrm{sat}(z_2,\rho):=\begin{cases}\mathrm{sgn}(z_2) & |z_2|>\rho\\ z_2/\rho \quad , & |z_2|\le\rho\end{cases} \tag{34}$$

where ρ is relative to the thickness of the boundary layer.
Let the estimative error be $\tilde{\xi}_a$, i.e., $\tilde{\xi}_a=\xi_a-\hat{\xi}_a$ and then select a Lyapunov function as

$$V_4=w^2/2+z_1^2/2+z_2^2/2+\tilde{\xi}_a^2/2\Psi \tag{35}$$

The time derivative of (35) is

$$\begin{aligned}\dot{V}_4&=w\dot{w}+z_1\dot{z}_1+z_2\dot{z}_2+\tilde{\xi}_a\dot{\tilde{\xi}}_a/\Psi\\ &\le -k_1w^2-k_2z_1^2-k_3z_2^2-\left(z_2\hat{\xi}_a\mathrm{sat}(z_2,\rho)-\Omega|z_2|\right)-\tilde{\xi}_a z_2\mathrm{sat}(z_2,\rho)\\ &=-2kV_4-\left(z_2\xi_a\mathrm{sat}(z_2,\rho)-\Omega|z_2|\right)\\ &=-2kV_4-|z_2|\left(\xi_a|z_2|/\rho-\Omega\right)\\ &=-2kV_4+\vartheta(t)\end{aligned} \tag{36}$$

where $\vartheta(t)$ is bounded by $|\vartheta(t)|\le\vartheta_{\max}$. In general, $\xi_a\ge\Omega$ is available. Suppose that $\xi_a=\Omega$, it follows $\vartheta(t)=-\Omega|z_2|\left(|z_2|/\rho-1\right)$. The maximum value $\vartheta_{\max}=\rho\Omega/4$ occurs at $|z_2|=\rho/2$. Eq. (36) reveals that for $t\to\infty$, it follows

$$\begin{aligned}V_4(t)&\le\exp(-2kt)V_4(0)+\int_0^t\exp(-2k(t-\upsilon))\vartheta(\upsilon)d\upsilon\\ &\le\exp(-2kt)V_4(0)+\frac{\vartheta_{\max}}{2k}\left[1-\exp(-2kt)\right]\le\frac{\rho\Omega}{8k}\end{aligned} \tag{37}$$

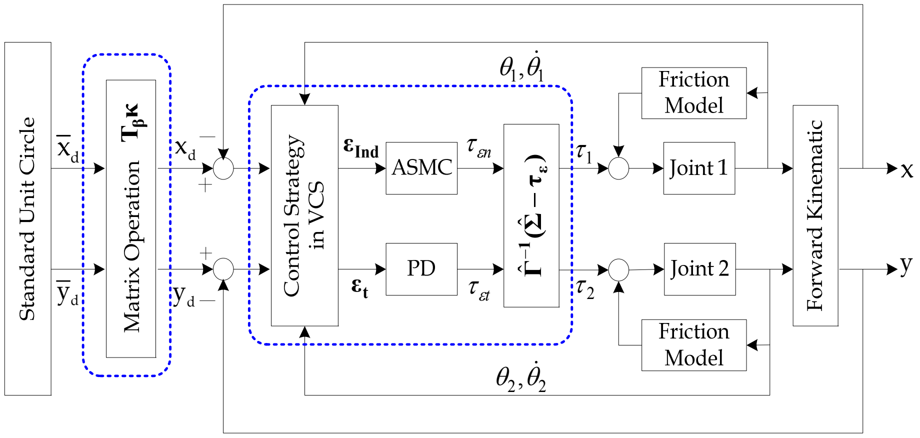

Standard Unit Circle
$\bar{x}_d$
$\bar{y}_d$
Matrix Operation $\mathbf{T}_\beta\boldsymbol{\kappa}$
x_d
y_d
−
+
+
−
Control Strategy in VCS
$\boldsymbol{\varepsilon}_{\mathbf{Ind}}$
$\boldsymbol{\varepsilon}_{\mathbf{t}}$
ASMC
PD
$\tau_{\varepsilon n}$
$\tau_{\varepsilon t}$
$\hat{\boldsymbol{\Gamma}}^{-1}(\hat{\boldsymbol{\Sigma}}-\boldsymbol{\tau}_\varepsilon)$
τ_1
τ_2
$\theta_1,\dot{\theta}_1$
$\theta_2,\dot{\theta}_2$
Friction Model
Joint 1
Joint 2
Friction Model
Forward Kinematic
x
y

Fig. 6. Block diagram of the proposed contouring control scheme for a 2-Link robotic system.

By (37), the system is exponentially stable with a guaranteed performance associated with the size of control parameters ρ and k. The overall contouring control architecture is illustrated in Fig. 6. It is similar with the standard feedback control loop, where the main control components are highlighted in the dotted blocks.
Remark. 1 For illustration purpose, a PD controller is applied to the tangential dynamics such that tangential tracking error cannot be eliminated completely. Of course one can also apply a robust controller to pursue its performance if necessary. However, the following simulations are going to show that even in the presence of tracking errors, the contouring performance will not be degraded by using the proposed contouring control framework.
Remark. 2 The action of the adaptive law activates when $|z_2| > \rho$. It means that for a given small gain value, $\hat{\xi}_a$ will be renewed in real time until the criterion $|z_2| \le \rho$ is achieved.

5. Numerical Simulations

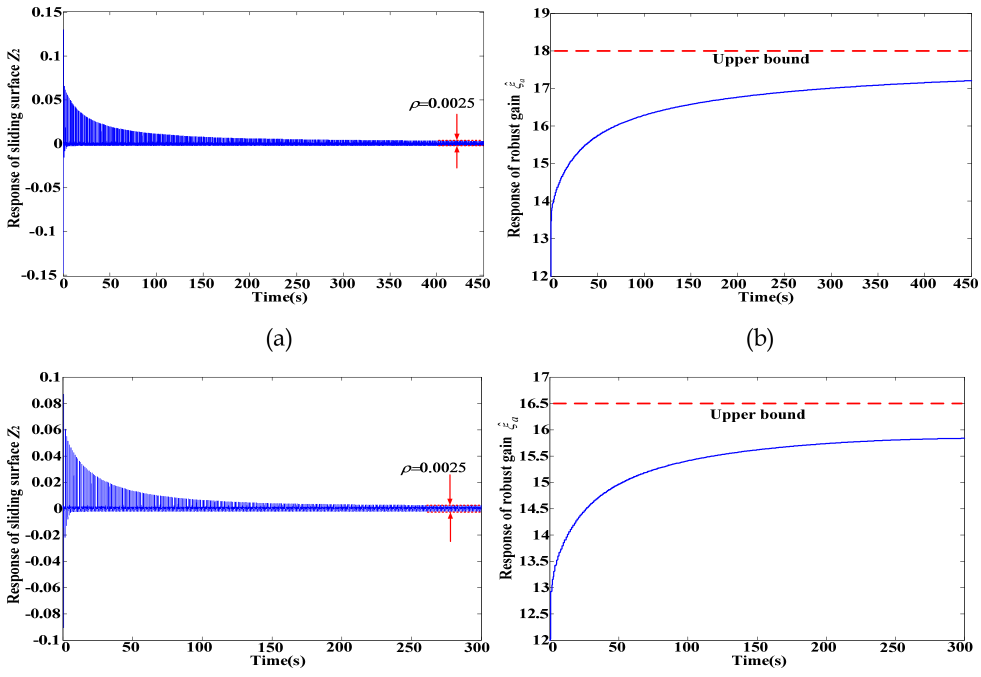
In this section, a robot system in consideration of nonlinear friction effects is taken as an example. The friction model used in numerical simulations is given by

$$F_i = \operatorname{sgn}(\dot{\theta}_i)\left[F_{ci} + (F_{si} - F_{ci})\exp\left(-(\dot{\theta}_i / \dot{\theta}_{si})^2\right)\right] + \sigma_{si}\dot{\theta}_i \tag{38}$$

where F_{ci} is the Coulomb friction and F_{si} is the static friction force. $\dot{\theta}_{si}$ denotes an angular velocity relative to the Stribeck effect and σ_{si} denotes the viscous coefficient. The suffix $i = 1,2$ indicates the number of robot joint.
The parameters used in friction model are:

$$F_{c1} = 0.025,\ F_{c2} = 0.02,\ F_{s1} = 0.04,\ F_{s2} = 0.035$$
$$\dot{\theta}_{s1} = \dot{\theta}_{s2} = 0.001,\ \sigma_{s1} = 0.005 \text{ and } \sigma_{s2} = 0.004.$$

According to the foregoing analysis, an adequate switching gain is suggested to be determined in advance for confirming system robustness. Thus, estimations are performed previously for two contouring control tasks, i.e., circular and elliptical contours. Fig. 7(a) and (c) show the responses of sliding surface and Fig. 7(b) and (d) depict the response of $\hat{\xi}_a$ during update.
From Fig. 7, it implies that the sliding surfaces were suppressed to the prescribed boundary layer by integrating with the adaptation law. The initial guess-value $\hat{\xi}_a(0) = 12$ and the adaptation gain $\Psi = 100$ are applied in (31). According to (32), an adequate value of $\hat{\xi}_a$ was determined when $|z_2| \le \rho = 0.0025$ is achieved. From the adaptation results shown in Fig. 7(b) and 7(d), the conservative switching gains $\hat{\xi}_a = 18$ and 16.5 are adopted to handle circular and elliptical profiles, respectively.



(a) (b)

(c) (d)

Fig. 7. Responses of sliding surface and estimated robust gain. (a)-(b) for circular contour, (c)-(d) for elliptical contour.

5.1 Circular contour

The following values are used for the control of circular profile:

$$\beta = 0\,,\ \kappa_x = \kappa_y = 0.1\,,\ f = 1\,,\ \left(c_{xo}\,, c_{yo}\right) = \left(0.21\,, 0.15\right)$$
$$k = 10\,,\ K_{Vt} = 9\,,\ K_{Pt} = 20\,,\ \hat{m}_1 = 7.8\,,\ \hat{m}_2 = 0.37$$

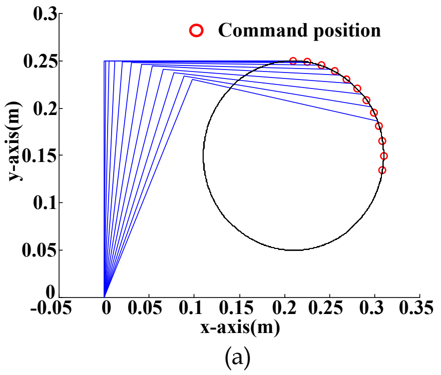
Exact system parameters of two-arm robot are

$$m_1 = 8.344\,,\ m_2 = 0.348\,,\ l_1 = 0.25\,,\ l_2 = 0.21$$

In this case, the position of starting point in the working space is set to be at $\mathbf{P_a}(0) = [x_0 \quad y_0]^T = [0.21 \quad 0.25]^T$. For a given position $\mathbf{P_a}(0)$, initial joint positions can be calculated by applying the inverse kinematics as follows

$$\begin{aligned}\theta_2(0) &= \pm\cos^{-1}\left(\frac{x_0^2+y_0^2-l_1^2-l_2^2}{2l_1l_2}\right)\\ \theta_1(0) &= \tan^{-1}\left(\frac{y_0}{x_0}\right)-\tan^{-1}\left(\frac{l_2\sin\theta_2}{l_1+l_2\cos\theta_2}\right)\end{aligned} \tag{39}$$

Referring to the simulations, Fig. 8 obviously illustrates the contour following behavior. It shows that even though the real time command position (i.e., the moving ring) goes ahead the end-effector, the end-effector still follows to the desired contour without significant deviation. The tracking responses are shown in Fig. 9(a) and (b), where the tracking errors exist significantly, but the contouring performance, evaluated by the contour index ε_{Ind}, remains in a good level. The corresponding control efforts of each robot joint are drawn in Fig.10(a)-(b). Examining Fig. 8(a) and (b) again, it can be seen that the time instants where the relative large tracking errors occur are also the time instants the relative large CIs are induced. The reason can refer to the dynamics of CI given in (16). Due to the face that (16b) is perturbed by the coupling uncertain terms η_3 when $\varepsilon_t \neq 0$, the control performance will be (relatively) degraded when large tangential tracking errors occur.



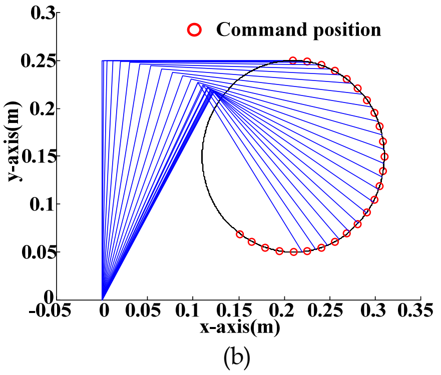
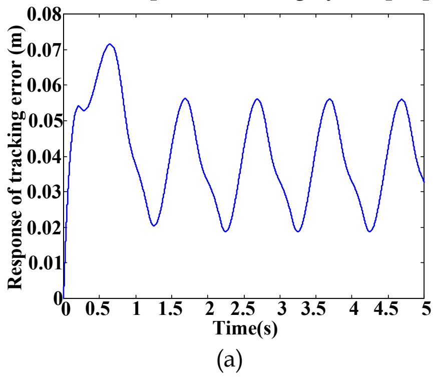


Fig. 8. Behavior of path following by the proposed method.



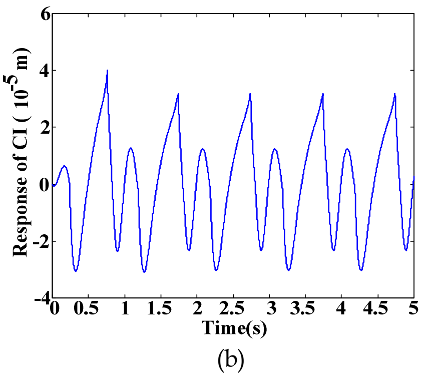


Fig. 9. Performance of tracking and equivalent contouring errors.

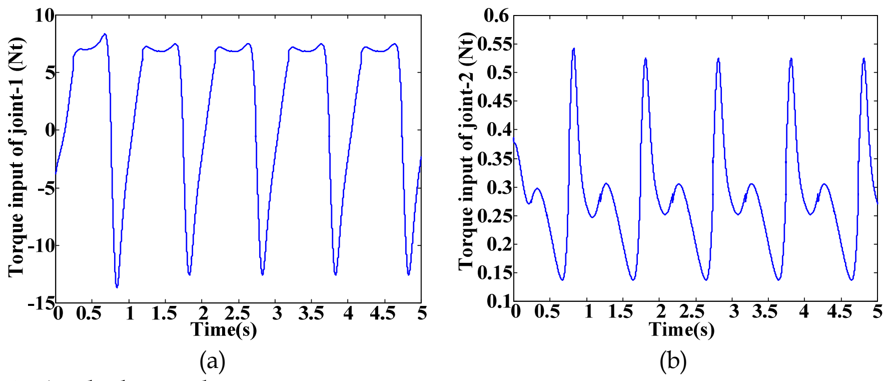


Fig. 10. Applied control torque.

5.2 Elliptical contour

In this case, the value of parameters are the same with those used in the previous case except $\beta = \pi / 3$, $\kappa_x = 0.15$ and $\kappa_y = 0.1$.

Fig. 11 evidently demonstrates that the proposed method confines the motion of the end-effector to the desired contour. The tracking performance and contouring performance are shown in Fig. 12(a) and (b), respectively. The manipulator motion remains attaching on the elliptical profile even though the end-effector doesn't track the real time command precisely. The result is consistent with the behavior illustrated in Fig. 2, i.e., the end-effector at *A* approaches to the real time command *D* through the desired path without causing short cutting phenomenon. Moreover, it has been demonstrated that the CI approach is also capable of avoiding over-cutting phenomenon, which is induced by the *T*-*N* coordinate transformation approach, in the presence of large tracking errors (Peng & Chen, 2007a). The simulation results confirm again that good contouring control performance does not necessarily rely on the good tracking level. The corresponding continuous control efforts of joint-1 and -2 are shown in Fig. 12(a) and (b), respectively.

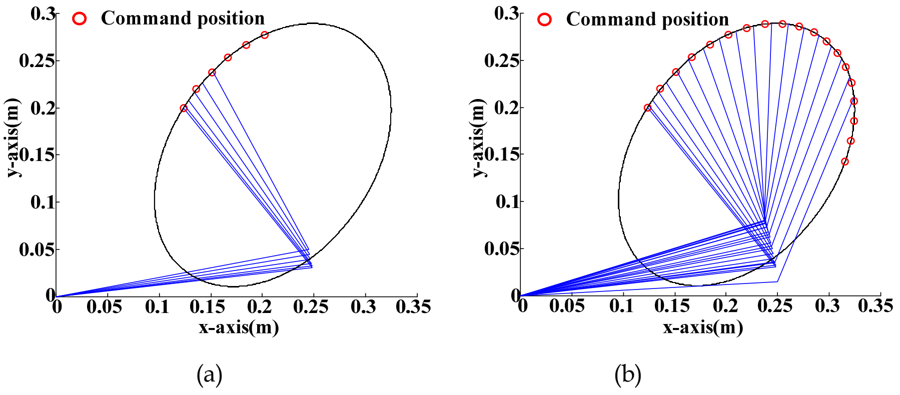
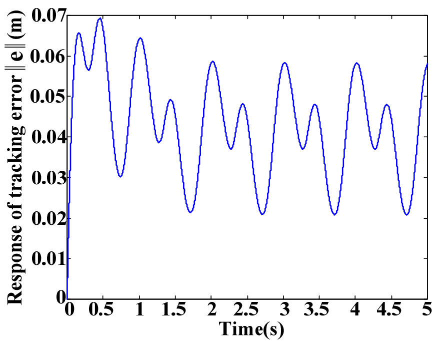
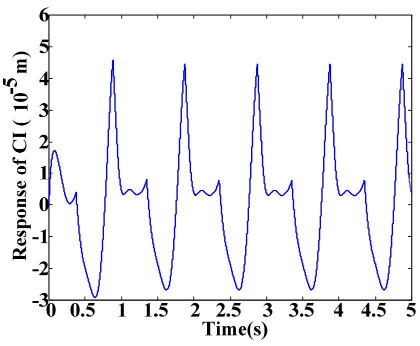


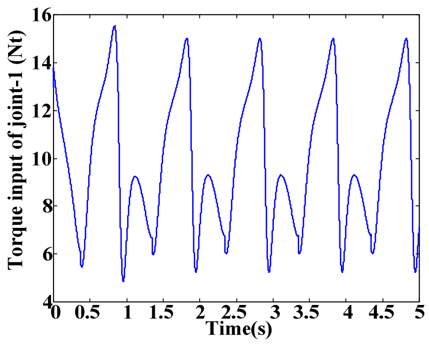
Fig. 11. Partial behavior of path following by the proposed method.

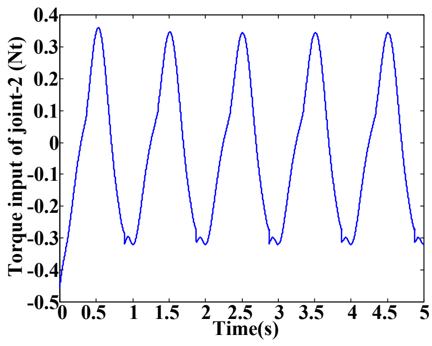




(a) (b)

Fig. 12. Performance of tracking and contouring.





(a) (b)

Fig. 13. Applied control torque.

6. Conclusion

In the robotic motion control field, positioning and tracking are considered as the main control tasks. In this Chapter, we have addressed a specific motion control topic, termed as contouring control. The core concept of the contouring control is different from the main object of the tracking control according to its goal.

For tracking control, the desired goal is to track the real time reference command as precise as possible. On the other hand, the main object is to achieve precise motion along prescribed contours for contouring control. Under this circumstance, tracking error is no longer a necessary performance index requiring to be minimized. To enhance resulting contour precision without relying on tracking performance, a contour following control strategy for robot manipulators is presented.

Different from the conventional manipulator motion control, a contour error dynamics is derived via coordinate transformation and an equivalent error called CI is introduced in VCS to evaluate contouring control performance. The contouring control task in the VCS turns into a stabilizing problem in tangential dynamics and a regulation problem in modified normal dynamics. The main advantage of the control scheme is that the final contouring accuracy will not be degraded even if the tracking performance of the robot manipulator is not good enough; that is, the existence of tracking errors will not make harm to the final contouring quality. This advantage has been apparently clarified through numerical study.

7. References

Chen, C. L. & Lin, K. C. (2008). Observer-Based Contouring Controller Design of a Biaxial State System Subject to Friction, *IEEE Transactions on Control Systems Technology*, Vol. 16, No. 2, 322-329.

Chen, C. L. & Xu, R. L. (1999). Tracking control of robot manipulator using sliding mode controller with performance robustness, *Journal of Dynamic Systems Measurement and Control-Transactions of the ASME*, Vol. 121, No. 1, 64-70.

Chen, S. L.; Liu, H. L. & Ting, S. C. (2002). Contouring control of biaxial systems based on polar coordinates, *IEEE-ASME Transactions on Mechatronics*, Vol. 7, No. 3, 329-345.

Chin, J. H. & Lin, T. C. (1997). Cross-coupled precompensation method for the contouring accuracy of computer numerically controlled machine tools, *International Journal of Machine Tools and Manufacture*, Vol. 37, No. 7, 947–967.

Chiu, G.T.-C. & Tomizuka, M. (2001). Contouring control of machine tool feed drive systems: a task coordinate frame approach, *IEEE Transactions on Control Systems Technology*, Vol. 9, No. 1, 130-139.

Dong, W. & Kuhnert, K. D. (2005) Robust adaptive control of nonholonomic mobile robot with parameter and nonparameter uncertainties, *IEEE Transactions on Robotics*, Vol. 21, No. 2, 261-266.

Fang, R. W. & Chen, J. S. (2002). A cross-coupling controller using an H-infinity scheme and its application to a two-axis direct-drive robot, *Journal of Robotic Systems,* Vol. 19, No. 10, 483-497.

Feng, G. & Palaniswami, M. (1993). Adaptive control of robot manipulators in task space, *IEEE Transactions on Automatic Control,* Vol. 38, No. 1, 100-104.

Ho, H. C.; Yen, J. Y. & Lu, S. S. (1998). A decoupled path-following control algorithm based upon the decomposed trajectory error, *International Journal of Machine Tools and Manufacture*, Vol. 39, No. 10, 1619-1630.

Hsieh, C.; Lin, K. C. & Chen, C. L. (2006). Contour Controller Design for Two-dimensional Stage System with Friction, *Material Science Forum,* Vol. 505-507, 1267-1272.

Koren, Y. (1980). Cross-Coupled Biaxial Computer Control for Manufacturing Systems, *Journal of Dynamic Systems Measurement and Control-Transactions of the ASME,* Vol. 102, No. 4, 265-272.

Lee, J. H.; Dixon, W. E.; Ziegert, J. C. & Makkar, C. (2005). Adaptive nonlinear contour coupling control for a machine tool system, *IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings*, 1629 – 1634.

Peng, C. C. & Chen, C. L. (2007a). Biaxial contouring control with friction dynamics using a contour index approach, *International Journal of Machine Tools & Manufacture*, Vol. 2007, No. 10, 1542-1555.

Peng, C. C. & Chen, C. L. (2007b). A 3-dimensional contour following strategy via coordinate transformation for manufacturing applications, *International Conference on Advanced Manufacture*, Tainan, Taiwan, November, Paper No. B4-95.

Ramesh, R.; Mannan, M. A. & Poo, A. N. (2005). Tracking and contour error control in CNC servo systems, *International Journal of Machine Tools and Manufacture*, Vol. 45, No. 3 301-326.

Sarachik, P. & Ragazzini, J. R. (1957). A Two Dimensional Feedback Control System, *Transactions AIEE*, Vol. 76, 55–61.

Shih, Y. T.; Chen, C. S. & Lee, A. C. (2002). A novel cross-coupling control design for Bi-axis motion, *International Journal of Machine Tools and Manufacture*, Vol. 42, No. 14, 1539-1548.

Slotine, J. J. E. & Li, W. P. (1988). Adaptive manipulator control: a case study, *IEEE Transactions on Automatic Control*, Vol. 33, No. 11, 995-1003.

Spong, M. W. & Vidyasagar, M. (1989). *Robot dynamics and control*, John Wiley & Sons, Inc.

Sun, M.; Ge, S. S. & Mareels, I. M. Y. (2006). Adaptive repetitive learning control of robotic manipulators without the requirement for initial repositioning, *IEEE Transactions on Robotics*, Vol. 22, No. 3, 563-568.

Wang, L. S & Zhang, J. (2004). The research of static de-coupled contour control technique of the ultra-precision machine tool, *Proceedings of the 5th World Congress on Intelligent Control and Automation*, June 15-19, Hangzhou, China.

Yeh, S. S. & Hsu, P. L. (1999) Analysis and design of the integrated controller for precise motion systems, *IEEE Transactions on Control Systems Technology*, Vol. 7, No. 6, 706-717.

Zhong, Q.; Shi, Y.; Mo, J. & Huang, S. (2002). A linear cross-coupled control system for high-speed machining, *The International Journal of Advanced Manufacturing Technology*, Vol. 19, No. 8, 558-563.

Zhu, W. H.; Chen, H. T. & Zhang, Z. J. (1992). A variable structure robot control algorithm with an observer, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 4, 1992, 486-492.

Yoo, D. S. & Chung, M. J. (1992). A variable structure control with simple adaptation laws for upper bounds on the norm of the uncertainties, *IEEE Transactions on Automatic Control*, Vol. 37, No. 6, 860-865.

Acknowledgements

Acknowledgments: Part of this work was supported by the National Science Council, Taiwan, under the Grant No. NSC96-2221-E006-052.

Appendix

The matrices used in this paper for 2-link rigid robot are listed in the following

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} -2\mathrm{m}_2 \mathrm{l}_1 \mathrm{l}_{c2} \dot{\theta}_2 \sin(\theta_2) & -\mathrm{m}_2 \mathrm{l}_1 \mathrm{l}_{c2} \dot{\theta}_2 \sin(\theta_2) \\ \mathrm{m}_2 \mathrm{l}_1 \mathrm{l}_{c2} \dot{\theta}_1 \sin(\theta_2) & 0 \end{bmatrix},$$

$$\mathbf{G} = \begin{bmatrix} \mathrm{m}_1 \mathrm{g} \mathrm{l}_{c1} \cos(\theta_1) + \mathrm{m}_2 \mathrm{g}(\mathrm{l}_1 \cos(\theta_1) + \mathrm{l}_{c2} \cos(\theta_1 + \theta_2)) \\ \mathrm{m}_2 \mathrm{g} \mathrm{l}_{c2} \cos(\theta_1 + \theta_2) \end{bmatrix}$$

where

$$\begin{aligned}
\mathbf{M}_{11} &= m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2l_1 l_{c2}\cos(\theta_2)\right) + I_1 + I_2 \\
\mathbf{M}_{12} &= m_2\left(l_1 l_{c2}\cos(\theta_2) + l_{c2}^2\right) + I_2 \\
\mathbf{M}_{21} &= m_2\left(l_1 l_{c2}\cos(\theta_2) + l_{c2}^2\right) + I_2 \\
\mathbf{M}_{22} &= m_2 l_{c2}^2 + I_2 \\
l_{ci} &= \frac{1}{2} l_i,\ I_i = \frac{1}{12} m_i l_i^2
\end{aligned}$$

The vector used in (2) is

$$h(\boldsymbol{\theta}(t)) = \begin{bmatrix} l_1\cos(\theta_1) + l_2\cos(\theta_1 + \theta_2) \\ l_1\sin(\theta_1) + l_2\sin(\theta_1 + \theta_2) \end{bmatrix}$$

and the Jacobin matrix is defined as

$$\mathbf{J} = \begin{bmatrix} -l_1\sin(\theta_1) - l_2\sin(\theta_1 + \theta_2) & -l_2\sin(\theta_1 + \theta_2) \\ l_1\cos(\theta_1) + l_2\cos(\theta_1 + \theta_2) & l_2\cos(\theta_1 + \theta_2) \end{bmatrix}$$

Regarding command generation, the operation matrices are

$$\mathbf{T}_{\beta} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix},\ \boldsymbol{\kappa} = \begin{bmatrix} \kappa_x & 0 \\ 0 & \kappa_y \end{bmatrix}$$

Design of Adaptive Controllers based on Christoffel Symbols of First Kind

Juan Ignacio Mulero-Martínez
Technical Universty of Cartagena
Spain

1. Introduction

The present chapter is aimed at systematically exposing the reader to certain modern trends in designing advanced robot controllers. More specifically, it focuses on a new and improved method for building suitable adaptive controllers guaranteeing asymptotic stability. It covers the complete design cycle, while providing detailed insight into most critical design issues of the different building blocks. In this sense, it takes a more global design perspective in jointly examining the design space at control level as well as at the architectural level.

The primary purpose is to provide insight and intuition into adaptive controllers based on Christoffel symbols of first kind for a serial-link robot arm, (Mulero-Martínez, 2007a). These controllers are referred to as static since the positional dependence of the nonlinear functions. In this context, the preferred method of nonlinear compensation is the method of building emulators. Often, however, the full power of the method is overlooked, and very few works deal with these techniques at the level of detail that the subject deserves. As a result, the chapter fills that gap and includes the type of information required to help control engineers to apply the method to robot manipulators. Developed in this chapter are several deep connections between dynamics analysis and implementation emphasizing the powerful adaptive methods that emerge when separate techniques from each area are properly assembled in a larger context.

After beginning with a comprehensive presentation of the fundamentals of these techniques, the chapter addresses the problem of factorization of the Coriolis/centripetal matrix, (Mulero-Martinez, 2009). This aspect is crucial when designing non-linear compensators by emulation. At this point, it is provided a concise and didactically structured description of the design of emulators as matters stand, (Mulero-Martinez, 2006). Specifically, emulators are split up into sub-emulators to improve and simplify the design of controllers while making faster the updating of parameters. From a practical point of view, the implementation is developed by resorting to parametric structures. This means to obtain a set of system's own function as regression functions.

Most of the adaptive schemes start from the notable property of linearity in the parameters, which lead naturally to equivalent structures when designing emulators for the nonlinear terms. When the linearity in the parameters (LIP) is considered as a first assumption in the development of adaptive schemes, it is clear that there exists a strong connection to the LIP emulators formulated in terms of a regression matrix and a vector of parameters. The main difference between standard adaptive schemes and the proposed approach stems from the idea of developing efficient controllers. The present work is aimed by attempts to mitigate the "curse of dimensionality" by exploiting the representation properties associated with the matrix of Coriolis/centripetal effects. By recalling the connection between LIP representation of robot manipulators and LIP adaptive emulators, it can be asserted that standard scheme matches perfectly with a dynamic emulator. Thus, the regression matrix, depend not only on the position joint variables but also on the velocity and acceleration variables.

As regards to the control, a novel theorem guarantees the stability for the whole system and is based on the Lyapunov energy. The proof is generalized to cope with a realistic case where both a functional reconstruction error and an external disturbance are present. It should be observed that the functional reconstruction error is caused by not using a number of regression functions appropiately distributed in the space. As a result, these considerations lead to a quite different approach, since it is required to analyze the initial conditions of the errors to guarantee the validity of the approximation. The specification of a range of validity causes that the stability holds only inside a compact set. As a consequence, the proof guarantees semi-global stability as opposed to the standard schemes where the stability is attained in the whole state space, in a global sense. Apart from these considerations, a number of remarks have been made to address some special aspects such as the boundedness of the parameters, the ultimately uniformly boundedness of all the signals and the stability in the ideal case.

The main benefit of the proposed controller is that it allows to derive tuning laws only for inertia, gravitational and frictional parameters. The Coriolis parameters are not necessary to be used because of the approximation based on Christoffel symbols. This is very useful to implement adaptive controllers since the number of nodes diminishes and the computational performance improves. Previously, an extensive analysis of the mechanical properties for a robot has been discussed. The regression functions for the adaptive controller depend on the non-linear functions associated with the inertia matrix, and therefore, a discretization of positions could be done for the inertia matrix. This is a very useful aspect because the position space for a revolute robot is compact and in consequence, the number of nodes is limited to approximate a non-linear function.

The plan of the chapter is as follows. In section 2 the representation properties for the Coriolis/centripetal matrix are analysed. An interpretation for the Coriolis/centripetal matrix is presented and the description by means of the Christoffel symbols of first kind and fundamental matrices are provided. In section 3, emulators are used to approximate the non-linearities of a robot using the properties presented in the previous section and the Kronecker product. The next section presents the design of the adaptive controller in terms of a control law and a parameter updating law. This section concludes with a theorem that guarantees the stability for the whole system and is based on the Lyapunov energy. Finally an example of a 2-dof robot arm is used to illustrate the theorem.

2. Representation of the Coriolis/Centripetal Matrix. Fundamental Matrices

In this section some notions regarding the representation of the Coriolis/centripetal matrices are introduced. All the ideas presented here constitute an original contribution and have many interesting implications in the field of robotics. To this end, fundamental matrices are introduced and described in terms of their structure. Moreover, some emerging properties are analyzed, allowing one to build the Coriolis/centripetal matrix in a simple way. Let start with the definition of the matrix M_D which from now on will be called the inertia derivative matrix.

Definition 1:

$$M_D(q,x) = \sum_{j=1}^{n} \frac{\partial M(q)}{\partial q_j} x e_j^T \tag{1}$$

where $M(q)$ is a generalized inertia matrix of dimension $n \times n$ a unitary vector of dimension n with a value 1 in the position j and x is an arbitrary vector of dimension n. It is noted that if x represents the generalized velocity vector, the matrix M_D will stand for the gradient of the generalized momentum with respect to the position coordinates q. This means that the gradient of the kinetic energy as a quadratic form $\frac{1}{2}x^T M(q)x$ relative to the joint position can be written as $\frac{1}{2}M_D^T(q,x)x$. Another representation of M_D is showed below.

Property 1: The matrix $M_D(q,x)$ can be expressed as

$$M_D(q,x) = \sum_{i=1}^{n} \frac{\partial M_i}{\partial q} x_i \tag{2}$$

Proof: It is very easy to show that $\frac{\partial M(q)}{\partial q_j} x e_j^T = \sum_{i=1}^{n} \frac{\partial M_i}{\partial q_j} x_i e_j^T$ since the following intermediate equation is obtained

$$\frac{\partial M(q)}{\partial q_j} x e_j^T = \left(\sum_{i=1}^{n} \frac{\partial M_i}{\partial q_j} e_i^T \right) x e_j^T = \sum_{i=1}^{n} \frac{\partial M_i}{\partial q_j} x_i e_j^T \tag{3}$$

Using the definition 1 and the identity (3) the hypothesis of the property is concluded

$$M_D(q,x) = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\partial M_i}{\partial q_j} x_i e_j^T = \sum_{i=1}^{n} \frac{\partial M_i}{\partial q} x_i \tag{4}$$

Q.E.D.

Now a new matrix will be introduced and from now on will be called as inertia velocity matrix, playing a central role in the representation theory.

Definition 2: Let define $M_v(q,x)$ in the following way

$$M_v(q,x) = \left(\frac{\partial M_1}{\partial q} x, \ldots, \frac{\partial M_n}{\partial q} x \right) = \sum_{i=1}^{n} \frac{\partial M_i}{\partial q} x e_i^T \tag{5}$$

The inertia velocity matrix $M_v(q,x)$ receives its name from the fact that when $x=\dot{q}$, the term $M_v(q,\dot{q})$ will be the time differentiation of the generalized inertia matrix, i.e. $\dot{M}(q)$. The following property provides an alternative way to write the matrix M_V.

Property 2: The inertia velocity matrix can be also expressed as

$$M_v(q,x)=\sum_{i=1}^{n}\frac{\partial M(q)}{\partial q_i}x_i \tag{6}$$

Proof: It is known that $\frac{\partial M_i}{\partial q}xe_i^T=\sum_{j=1}^{n}\frac{\partial M_i}{\partial q_j}x_je_i^T$ and using the definition 2 the property is proved as follows

$$M_v(q,x)=\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial M_i}{\partial q_j}x_je_i^T=\sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{\partial M_i}{\partial q_j}e_i^T\right)x_j=\sum_{j=1}^{n}\frac{\partial M(q)}{\partial q_j}x_j \tag{7}$$

Q.E.D.

2.1 Properties of the fundamental matrices

Subsequently, some properties related to the fundamental matrices are analyzed. Following a systematic methodology, the properties have been classified into two groups: commutation properties and representation properties.

2.1.1. Commutation properties

Commutation properties permit interchange of an external arbitrary vector y and a vector x passed to a fundamental matrix as an argument. The following property makes possible the commutation while keeping the type of the fundamental matrices. This means that the transpose of the inertia derivative matrix can be transformed into the same structure by simply interchanging the roles of x and y.

Property 3: $M_D^T(q,x)y=M_D^T(q,y)x$

The proof of the last property follows directly from the definition of M_D. The following property allows to pass from a type of fundamental matrix to another commuting the vectors x and y.

Property 4: $M_v(q,x)y=M_D(q,y)x$

Proof :

$$M_v(q,x)y=\sum_{i=1}^{n}\frac{\partial M_i}{\partial q}xe_i^Ty=\sum_{i=1}^{n}\frac{\partial M_i}{\partial q}xy_i=M_D(q,y)x \tag{8}$$

Q.E.D.

2.1.2. Properties of representation of the Coriolis/centripetal matrix

These properties are very important to describe the Coriolis/centripetal matrix from the fundamental structures.

Property 5:

$$M_D^T(q,\dot{q}) = \sum_{i=1}^{n} \tfrac{\partial M_i^T}{\partial q}\dot{q}e_i^T \tag{9}$$

Proof: First of all, the transpose of the inertia derivative matrix can be represented as $M_D^T(q,\dot{q}) = \sum_{j=1}^{n} e_j \dot{q}^T \sum_{i=1}^{n} \tfrac{\partial M_i}{\partial q_j} e_i^T$ using the definition 1 and the fact that the differentiation of the inertia matrix with respect to the generalized coordinate q_j is $\tfrac{\partial M(q)}{\partial q_j} = \sum_{i=1}^{n} \tfrac{\partial M_i}{\partial q_j} e_i^T$. Since scalar product is commutative the following expression is derived, $\dot{q}^T \tfrac{\partial M_i}{\partial q_j} = \tfrac{\partial M_i^T}{\partial q_j}\dot{q}$, and as a result $M_D^T(q,\dot{q}) = \sum_{j=1}^{n} e_j \sum_{i=1}^{n} \tfrac{\partial M_i^T}{\partial q_j}\dot{q}e_i^T$. The order of summation is needed to be permutated to get the proposed identity.

$$M_D^T(q,\dot{q}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\partial M_i}{\partial q_j}e_j^T\right)^T \dot{q}e_i^T = \sum_{i=1}^{n}\frac{\partial M_i^T}{\partial q}\dot{q}e_i^T \tag{10}$$

Q.E.D.

Below, some representations of the Coriolis/centripetal matrix are introduced as properties derived from M_D and M_v .

Property 6: The matrix of Coriolis/centripetal effects can be expressed as

$$C(q,\dot{q}) = \frac{1}{2}\left(M_v(q,\dot{q}) + M_D(q,\dot{q}) - M_D^T(q,\dot{q})\right) = \frac{1}{2}\left(M_v(q,\dot{q}) + J(q,\dot{q})\right) \tag{11}$$

where $J(q,\dot{q}) = M_D(q,\dot{q}) - M_D^T(q,\dot{q})$ is a skew symmetric matrix, i.e. $J(q,\dot{q}) = -J^T(q,\dot{q})$.

Proof: This is an immediate consequence of the representation of $C(q,\dot{q})$ by means of the property 1 and the fact that the inertia velocity matrix is $M_v(q,\dot{q}) = \dot{M}(q)$.

In a general way, the following representation can be derived

$$C(q,x) = \frac{1}{2}\left(M_v(q,x) + J(q,x)\right) \tag{12}$$

where x is a vector of dimension n and $J(q,x) = M_D(q,x) - M_D^T(q,x)$ is a skew symmetric matrix. It is remarkable that the definition of the Coriolis/centripetal matrix presented above, is different from the definition given by (Wen,1988) in the identity 2, $C(q,z)z = \tfrac{1}{2}\left(M_D(q,z) - J(q,z)z\right)$. An interesting property which is a direct implication of the property 4 is that, by setting $x = y$ in $C(q,x)y$.

Property 7: The Coriolis/centripetal force can be represented as

$$C(q,\dot{q})\dot{q} = \left(\dot{M}(q) - \frac{1}{2}M_D^T(q,\dot{q})\right)\dot{q} \tag{13}$$

or in a general form as

$$C(q,x)x = M_v(q,x)x - \frac{1}{2}M_D^T(q,x)x \tag{14}$$

where x is an arbitrary vector of dimension n :
Property 8: The Coriolis/centripetal matrix commutes with external vectors

$$C(q,x)y = C(q,y)x \tag{15}$$

Proof: In order to see this point the representation of the Coriolis/centripetal matrix will be used as a sum of the inertia velocity matrix, $M_v(q,x)$ and the skew symmetric matrix $J(q,x)$ given by equation (12).

$$C(q,x)y = \frac{1}{2}\left(M_v(q,x)y + J(q,x)y\right) \tag{16}$$

On one hand, it is known that $J(q,x) = M_D(q,x) - M_D^T(q,x)$ .Using the commutation properties 2 and 3 the following expression is derived

$$C(q,x)y = \frac{1}{2}\left(M_D(q,y)x - M_D^T(q,y)x + M_D(q,x)y\right) \tag{17}$$

On other hand, $J(q,y)x = M_D(q,y)x - M_D^T(q,y)x$ and then applying the commutation property 3 to $M_D(q,x)y$ the following result is achieved as claimed

$$C(q,x)y = \frac{1}{2}\left(J(q,y)x + M_v(q,y)x\right) = C(q,y)x \tag{18}$$

Q.E.D

3. Design of Emulators for Robot Manipulators.

3.1 Functional and Linear Parameterization.

The approach that follows is founded on the idea to find an emulator as a function close to the non-linear terms involved in the dynamics equations of a robot manipulator. In order to get a model from a practical point of view, uncertainties in the nonlinear terms. getting arise from the partial information about the exact structure of the dynamics, must be taken into account. The inaccuracies of a model can be classified into two classes: structured and unstructured uncertainties. The first kind of uncertainties comes out from the inaccuracies of the parameters whereas the unstructured uncertainties are related to unmodeled dynamics, see (Slotine & Li,1991). Thus, the uncertainties can be adaptatively compensated by defining each coefficient as a separate parameter so that the dynamics can be expressed in the linear in the parameters (LIP) and this means that nonlinearities can be split up into an unknown vector of physical parameters P and a known matrix of basis nonlinear functions $\Psi(q,\dot{q},x,y)$ comprising the elements of $M(q)$, $C(q,\dot{q})$, $G(q)$ and $F(q,\dot{q})$, referred to as regression matrix. Therefore, the nonlinear function $f(x)$ can be written in this sense adding a term of error ε, see (Ge et al., 1998).

$$f(q,\dot{q},x,y)=\Psi(q,\dot{q},x,y)P+\varepsilon \tag{19}$$

The linearity of the parameters is the major structural property of robot manipulators and has been analyzed in (Lewis et al., 2003). This linear factorization is always possible to be done for the rigid body dynamics of a fixed-based manipulator as long as the physical uncertainty is on the mass properties of the robot links. Furthermore, linearity of the parameters is the first assumption in the most of adaptive controllers. An alternative representation of the nonlinear component is as follows

$$f(q,\dot{q},x,y)=R(q,\dot{q})v(x,y) \tag{20}$$

where $R(q,\dot{q})=\begin{pmatrix}M(q) & C(q,\dot{q}) & G(q)\end{pmatrix}\in\mathbb{R}^{n\times(2n+1)}$ and $v=\begin{pmatrix}y^T & x^T & 1\end{pmatrix}\in\mathbb{R}^{2n+1}$. This factorization is always attainable whereas the linearity in the parameters (LIP) is only obtained under some circumstances. In the literature, emulators based on regression matrices have been used to approximate the nonlinear dynamics as a whole, as follows

$$\mathcal{N}(q,\dot{q},\ddot{q})=M(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q) \tag{21}$$

As an attempt to obtain more efficient computation, the emulator approximating the nonlinearity $f(x)$ is split up into several smaller components:

$$f(q,\dot{q},x,y)=f_m(q,y)+f_c(q,\dot{q},x)+f_g(q)+f_f(q,\dot{q}) \tag{22}$$

The function $f_m(q,x)=M(q)y$, stands for the nonlinearity of inertial terms and can be written taking into account that the components of $M(q)$ are continuous functions of their arguments so that each component can be uniformly approximated on any compact subset of the state space by an appropriately designed emulator.
From now on we assume that the number of parameters to approximate the column i of a matrix is l_i.

3.2 Inertia matrix $M(q)$.

3.2.1 Ideal Case

The inertia matrix $M(q)$ consists of column vectors $M_i(q)$ that can be approximated by regression matrices

$$M_i(q)=\Psi_{m_i}(q)\xi_{m_i} \tag{23}$$

where $\Psi_{m_i}(q)\in\mathcal{M}_{n\times l_i}(\mathbb{R})$ is a regression matrix of known robot functions and $\xi_{m_i}\in\mathbb{R}^{l_i}$ is a vector of unknown parameters (e.g. masses, inertia moments and link parameters). Then, with an arbitrary vector $x\in\mathbb{R}^n$, there follows the decomposition

$$M(q)x=\sum_{i=1}^{n}M_i(q)x_i=\left(\sum_{i=1}^{n}\Psi_{m_i}(q)x_i\xi_{m_i}\right)\tag{24}$$

On the assumption that the same regression matrix serves for each column, i.e. $\Psi_{m_1}(q)=\Psi_{m_2}(q)=\cdots\Psi_{m_n}(q)=\Psi_m(q)\in\mathcal{M}_{n\times l}(\mathbb{R})$, one may rewrite equation (24) by resorting to the Kronecker product as

$$M(q)x=\Phi_m(q,x)\xi_m\tag{25}$$

where $\xi_m\in\mathbb{R}^l$ is an l-dimensional vector of parameters and $\Phi_m(q,x)=\left(x^T\otimes\Psi_m(q)\right)\in\mathcal{M}_{n\times l}(\mathbb{R})$ is the regression matrix of the generalized inertia matrix.

3.2.2 Real Case

Given a closed, bounded subset $\Omega\subseteq\mathbb{R}^n$, and a specified accuracy ε_{m_i}, there exist values for the design parameters l_i and ξ_{m_i} so that for all $q(t)\in\Omega$ the following inequality is satisfied:

$$\left\|M_i(q)-\Psi_{m_i}(q)\xi_{m_i}\right\|\leq\varepsilon_{m_i}\tag{26}$$

For the sake of simplicity, we have assumed that $\Psi_{m_i}(q)$ are equal for all $i=1,\ldots,n$. The inertia matrix consists of column vectors that can be approximated as

$$M(q)=\sum_{i=1}^{n}M_i(q)e_i^T=\left(\sum_{i=1}^{n}\Psi_{m_i}(q)\xi_{m_i}e_i^T\right)+\varepsilon_m(q)\tag{27}$$

where $\varepsilon_m(q)\in\mathbb{R}^{n\times n}$ is the functional reconstruction error.

It is convenient to underline that the vector function $\Phi_m(q,x)$ only can be expressed by the Kronecker product whenever the hypothesis $\Phi_m(q,x)=\Psi_{m_i}(q)=\Psi_m(q)$ holds for all $i=1,\ldots,n$. As a consequence, a linear transformation with a change of basis can be derived in terms of the Kronecker product

$$M(q)x=\Phi_m(q,x)\xi_m+E_m(q,x)\tag{28}$$

The expansion given by (28) actually approximates the linear transformation $M(q)x$ to a certain degree of accuracy $E_m(q,x)=\varepsilon_m x$ by using an emulator with the position joint vector as the input. Specifically, the emulator in (28) has the vector $\left(q^T,x^T\right)^T$ as its input and n outputs, and a collection of simple computing elements $\Phi_m(q,x)$ are used to approximate the function $f_m(q,x)$.

3.3 Fundamental matrices.

For the sake of simplicity in the following, the matrix derivative of $\Psi_m(q)$ with $q\in\mathbb{R}^n$ will be denoted as

$$\frac{\partial\Psi_m(q)}{\partial q}\triangleq\begin{pmatrix}\frac{\partial\Psi_m(q)}{\partial q_1}\\ \cdots \\ \frac{\partial\Psi_m(q)}{\partial q_n}\end{pmatrix}\in\mathcal{M}_{n^2\times l}(\mathbb{R}) \tag{29}$$

3.3.1 Inertia Derivative Matrix $M_D(q,x)$

3.3.1.1 Real Case

It is possible to express the inertia derivative matrix as consisting of two terms, the approximation component $\hat{M}_D(q,x)$ and the reconstruction error $E_D(q,x)$.

$$M_D(q,x)=\sum_{j=1}^{n}\frac{\partial\hat{M}(q)}{\partial q_j}xe_j^T+\sum_{j=1}^{n}\frac{\partial E_m(q,x)}{\partial q_j}e_j^T \tag{30}$$

Following the discussion given in the ideal case, the approximation term $\hat{M}_D(q,x)y$ can be expressed in the regression form:

$$\hat{M}_D(q,x)y=\Phi_D(q,x,y)\xi_m \tag{31}$$

3.3.1.2 Ideal Case

The following lemmas are helpful for characterizing the inertia derivative matrix in LIP form.

Lemma 1 (LIP form of M_D). For arbitrary vectors $x,y\in\mathbb{R}^n$, $q\in\Omega_q\subseteq\mathbb{R}^n$, the inertia derivative matrix can be written in the LIP form as follows

$$M_D(q,x)y=\Phi_D(q,x,y)\xi_m \tag{32}$$

where $\xi_m\in\mathbb{R}^l$ is the parameter vector of the generalized inertia matrix and

$\Phi_D \in \mathcal{M}_{n\times l}(\mathbb{R})$ a regression matrix:

$$\Phi_D(q,x,y) = \left(y^T \otimes I_n\right)\psi(q,x)$$
$$\psi(q,x) = \left(x^T \otimes \frac{\partial \Psi_m(q)}{\partial q}\right) \tag{33}$$

Proof: This fact can be straightforward proved by resorting to the definition of M_D given above:

$$M_D(q,x)y = \sum_{i=1}^{n} \frac{\partial M(q)}{\partial q_i} x y_i = \left(\sum_{i=1}^{n} \frac{\partial \Phi_m(q,x)}{\partial q_i} y_i\right)\xi_m = \left(x^T \otimes \sum_{i=1}^{n} \frac{\partial \Psi_m(q)}{\partial q_i} y_i\right)\xi_m = \Phi_D(q,x,y)\xi_m \tag{34}$$

By closely examining the structure of Φ_D and using the property

$$\sum_{i=1}^{n} \frac{\partial \Psi_m(q)}{\partial q_i} y_i = \left(y^T \otimes I_n\right)\frac{\partial \Psi_m(q)}{\partial q} \tag{35}$$

it is easy to conclude that

$$\Phi_D(q,x,y) = x^T \otimes \left(\left(y^T \otimes I_n\right)\frac{\partial \Psi_m(q)}{\partial q}\right) = \left[x \otimes \left(\frac{\partial \Psi_m^T(q)}{\partial q}(y \otimes I_n)\right)\right]^T =$$
$$= \left[\left(x \otimes \frac{\partial \Psi_m^T(q)}{\partial q}\right)(y \otimes I_n)\right]^T = \left(y^T \otimes I_n\right)\psi(q,x) \tag{36}$$

3.3.2 Fundamental Matrix $M_v(q,x)$

3.3.2.1 Ideal Case

Lemma 2. For arbitrary vectors $x, y \in \mathbb{R}^n$, $q \in \Omega_q \subseteq \mathbb{R}^n$, the inertia velocity matrix can be written as

$$M_v(q,x)y = \Phi_v(q,x,y)\xi_m \tag{37}$$

where $\Phi_v(q,x,y) = \sum_{i=1}^{n} \frac{\partial \Phi_m(q,y)}{\partial q_i} x_i$.

Proof: Owing to the commutation property of fundamental matrices the result is direct.

Remark: It is easy to show that the Jacobian matrix $\Phi_v(q,x,y)$ also satisfies the commutation property, $\Phi_v(q,x,y) = \Phi_D(q,y,x)$ .This is an immediate consequence of the commutation property of $M_v(q,y)$ and $M_D(q,x)$ as given in the identity 3 .

3.3.2.2 Real Case

The inertia velocity matrix $M_v(q,x)$ can be seen as consisting of an approximation term $\hat{M}_v(q,x)$ and a reconstruction error $E_v(q,x)$. The approximation term $\hat{M}_v(q,x)$ can be written in terms of the Jacobian of $\Phi_v(q,x,y)$ as

$$\hat{M}_v(q,x)y = \Phi_v(q,x,y)\xi_m \tag{38}$$

and the error $E_v(q,x)$ is derived from the gradient of the inertia error $\varepsilon_m(q)$.

$$E_v(q,x) = \sum_{i=1}^{n} \frac{\partial \varepsilon_m}{\partial q_i} x_i = \sum_{i=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_i} x_i \tag{39}$$

3.3.3 Transpose of the Inertia Derivative Matrix

Now that it is known how M_D can be written in LIP form, the main challenge, now, is to extend this result to its transpose. This is more difficult since involves a permutation matrix as stated in the following lemma.

Lemma 3: Let $x, y \in \mathbb{R}^n$, $q \in \Omega_q \subseteq \mathbb{R}^n$ arbitrary vectors with appropriate units and P a permutation matrix1 defined as follows

$$P(n,m) = \sum_{i=1}^{n}\sum_{j=1}^{n} E_{ij} \otimes E_{ji} \tag{40}$$

where $E_{ij} = e_i e_j^T$ is a unit matrix having 1 in position (i,j) and all other entries are zero. Under these conditions, the term $M_D^T(q,x)$ can be written in LIP form as follows

$$M_D^T(q,x)y = \left(x^T \otimes I_n\right) P(n,n) \psi(q,y) \xi_m \tag{41}$$

Proof: By virtue of definition 1 and lemma 1, the generalized force $M_D^T(q,x)y$ can be directly derived in the LIP form which can be detailed into

$$\begin{aligned} M_D^T(q,x)y &= \sum_{i=1}^{n} e_i \left(x^T \frac{\partial M(q)}{\partial q_i} y \right) = \left(I_n \otimes x^T\right) \frac{\partial M(q)}{\partial q} y = \\ &= \left(I_n \otimes x^T\right) \frac{\partial \Phi_m(q,y)}{\partial q} \xi_m = \left(I_n \otimes x^T\right) \left(y^T \otimes \frac{\partial \Psi_m(q)}{\partial q} \right) \xi_m \end{aligned} \tag{42}$$

1 $P(m,n)$ is a matrix of zeroes and ones for which $P(m,n)^{-1} = P(m,n)^T = P(m,n)$.

With reference to the property $B \otimes A = P(m,p)^T (A \otimes B) P(n,q)$ for all $A \in \mathcal{M}_{m\times n}(\mathbb{R})$ and $B \in \mathcal{M}_{p\times q}(\mathbb{R})$, (see Corollary 4.3.10., p. 260, in (Horn & Johnson,1999)), the term $\left(x^T \otimes I_n\right)$ can be commutated as

$$\left(x^T \otimes I_n\right) = P(n,1)^T \left(I_n \otimes x^T\right) P(n,n) \tag{43}$$

The last identity can be simplified further by exploiting the structure of the permutation matrices. In particular, it is easy to show that $P(n,1) = P(1,n) = I_n$ (see problem 18, section 4.3, p. 265, in (Horn & Johnson, 1999)), which leads to

$$\left(I_n \otimes x^T\right) = \left(x^T \otimes I_n\right) P(n,n) \tag{44}$$

Therefore,

$$\begin{aligned} M_D^T(q,x)y &= \left(x^T \otimes I_n\right) P(n,n) \left(y^T \otimes \frac{\partial \Psi_m(q)}{\partial q}\right) \xi_m = \\ &= \left(x^T \otimes I_n\right) P(n,n) \psi(q,y) \xi_m \end{aligned} \tag{45}$$

Remark: The permutation matrix in the last lemma can be also written as

$$P(n,n) = \sum_{i=1}^{n} e_i \otimes I_n \otimes e_i^T = \sum_{i=1}^{n} e_i^T \otimes I_n \otimes e_i \tag{46}$$

For more details the reader is referred to the problem 21, section 4.3. p.286,in (Horn & Johnson, 1999).

3.3.4 Coriolis/Centripetal Matrix

On the basis of the description of M_D and M_D^T in LIP form, the skew-symmetric matrix J can be also represented as LIP.

Lemma 4: Let $x, y \in \mathbb{R}^n$, $q \in \Omega_q \subseteq \mathbb{R}^n$ arbitrary vectors with appropriate units. The skew-symmetric matrix $J(q,x)$ can be expressed is linear in the parameters,

$J(q,x)y = \Phi_J(q,x,y)\xi_m$

where $\Phi_J(q,x,y) = \left(y^T \otimes I_n\right)\left(I_{n^2} - P(n,n)\right)\psi(q,x)$.

Proof: It is straightforward that

$J(q,x)y = M_D(q,x)y - M_D^T(q,x)y = \left(y^T \otimes I_n\right)\left(I_{n^2} - P(n,n)\right)\psi(q,x)\xi_m$

Lemma 5: For arbitrary vectors $x, y \in \mathbb{R}^n$, $q \in \Omega_q \subseteq \mathbb{R}^n$, the inertia velocity matrix can be written as

$M_v(q,x)y = \Phi_v(q,x,y)\xi_m$

where $\Phi_v(q,x,y)=\sum_{i=1}^{n}\frac{\partial\Phi_m(q,y)}{\partial q_i}x_i$.

Proof: This is direct.

Remark: It is observed the following identity: $\Phi_D(q,x,y)=\Phi_v(q,y,x)$, which is consistent with the commutation property.

Remark: An alternative way for J is obtained by resorting to the commutation property:

$$\begin{aligned}M_v(q,x)y-M_D^T(q,x)y&=M_D(q,y)x-M_D^T(q,y)x=\\&=\left(x^T\otimes I_n\right)\left(I_{n^2}-P(n,n)\right)\psi(q,y)\xi_m=J(q,y)x\end{aligned}$$

Remarkably, the above lemmas can be conveniently used to write the Coriolis/centripetal matrix in LIP form.

Proposition 1 (Coriolis/Centripetal matrix in LIP form): Let $x,y\in\mathbb{R}^n$, $q\in\Omega_q\subseteq\mathbb{R}^n$ arbitrary vectors, the Coriolis/centripetal effect $C(q,x)y$ can be linearly factorized as a regression matrix $\Phi_C(q,x,y)$ and a parameter vector ξ_m , i.e. $C(q,x)y=\Phi_C(q,x,y)\xi_m$, where Φ_C is given by

$$\Phi_C(q,x,y)=\frac{1}{2}\left[\left(x^T\otimes I_n\right)\psi(q,y)+\left(y^T\otimes I_n\right)\left(I_{n^2}-P(n,n)\right)\psi(q,x)\right]$$

Proof: By restoring to the LIP form of M_v and J , matrix C can be written as

$$\begin{aligned}C(q,x)y&=\frac{1}{2}\left(M_v(q,x)y+J(q,x)y\right)=\\&=\frac{1}{2}\left[\left(x^T\otimes I_n\right)\psi(q,y)+\left(y^T\otimes I_n\right)\left(I_{n^2}-P(n,n)\right)\psi(q,x)\right]\xi_m\end{aligned}$$

3.4 Model Errors

The dynamic model of the robot manipulator is allowed to be imprecise since the nonlinear function $f(q,\dot{q},x,y)=M(q)x+C(q,\dot{q})y+G(q)$, (where $x,y\in\mathbb{R}^n$ are arbitrary vectors usually with units of acceleration and velocity respectively),is not exactly known. The imprecision comes from unstructured uncertainties, namely modeling errors caused by the truncation of the Gaussian expansion series. A detailed description of the approximation errors is demanding from a modeling viewpoint. To point out the fundamental aspects of error modeling, it is convenient to express the total error as composed by three terms, (Mulero-Martinez, 2007a),

$$E=E_m(q,x)+E_C(q,\dot{q},y)+E_G(q)$$

By referring to the expression of the Coriolis/centripetal matrix in (Mulero-Martínez,2007a) from the fundamental matrices M_D and M_V , the Coriolis/centripetal errors $E_C(q,\dot{q},y)$ can be written as

$$E_C(q,\dot{q},y)=\frac{1}{2}\left(E_D(q,\dot{q})-E_D^T(q,\dot{q})+E_V(q,\dot{q})\right)y$$

The error term $E_D(q,\dot{q})$ is the approximation error associated with the fundamental matrix $M_D(q,\dot{q})$ and $E_V(q,\dot{q})$ regards with the velocity matrix $M_V(q,\dot{q})$. For the sake of simplicity the gradient of the inertia error $\varepsilon_m(q)$ will be denoted as

$$\frac{\partial \varepsilon_m}{\partial q} \triangleq \left(\frac{\partial \varepsilon_m}{\partial q_1}, \dots, \frac{\partial \varepsilon_m}{\partial q_n} \right) \in \mathbb{R}^{n \times n^2}$$

These errors can be expressed in terms of the gradient of the inertia error $\varepsilon_m(q)$ as follows

$$\begin{aligned} E_D(q,\dot{q}) &= \sum_{i=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_i} \dot{q} e_i^T \\ E_V(q,\dot{q}) &= \sum_{i=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_i} \dot{q}_i \end{aligned} \tag{47}$$

For the following derivation, it is worth rewriting the mathematical errors in a more suitable form using the Kronecker product. This is written down in the following properties.

Claim 3: The linear transformation $E_D(q,\dot{q})\dot{q}_r$ can be formulated in terms of the Kronecker product as

$$E_D(q,\dot{q})\dot{q}_r = \frac{\partial \varepsilon_m(q)}{\partial q}(\dot{q}_r \otimes \dot{q}) \tag{48}$$

Proof: The proof is derived directly from the definition of $E_D(q,\dot{q})$

$$E_D(q,\dot{q})\dot{q}_r = \sum_{j=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_j} \dot{q} e_j^T \dot{q}_r = \sum_{j=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_j} \dot{q} \dot{q}_{r_j} = \frac{\partial \varepsilon_m(q)}{\partial q}(\dot{q}_r \otimes \dot{q})$$

Claim 4: The linear transformation $E_V(q,\dot{q})\dot{q}_r$ is expressed in terms of the Kronecker product as

$$E_V(q,\dot{q})\dot{q}_r = \frac{\partial \varepsilon_m(q)}{\partial q}(\dot{q} \otimes \dot{q}_r) \tag{49}$$

Proof: This fact can be trivially proved from the definition in equation (47)

$$E_V(q,\dot{q})\dot{q}_r = \sum_{i=1}^{n} \frac{\partial \varepsilon_m(q)}{\partial q_i} \dot{q}_i \dot{q}_r = \frac{\partial \varepsilon_m(q)}{\partial q}(\dot{q} \otimes \dot{q}_r)$$

Remark: Equations (48) and (49) are not the same since the Kronecker product is not commutative, i.e. $(\dot{q}_r \otimes \dot{q}) \neq (\dot{q} \otimes \dot{q}_r)$.

4. Design of the Adaptive Controller.

4.1 Error Dynamic Equation

In order to manage equilibrium points at the origin, it is necessary to make a coordinate transformation so that a position error variable is considered, $e(t) = q_d(t) - q(t)$. Thus, convergene of trajectory $q(t)$ to the desired trajectory $q_d(t)$ can be analysed observing position error trajectories $e(t)$ close to the origin in the phase space. The objective of the controller is both the stable tracking of trajectories and the rejection of disturbances. A good tracking performance means that the error converge to zero (asymptotic stability) or to a finite value, $\lim_{t\to\infty} e(t) = E < \infty$. This idea is also applied to $\dot{q}(t)$ and $\ddot{q}(t)$ because a position trajectory is given by three quantities $(q(t),\dot{q}(t),\ddot{q}(t))$. Measurements of velocities are easy to get by tachometers, but sensors of acceleration are noisy and are not used for implementation in robotics field. This fact conjures up to use a filtered error signal, $r(t)$, that is a derivative filter or PD (Slotine & Li, 1991):

$$r(t) = \dot{q}_r(t) - \dot{q}(t) \tag{50}$$

From (50) it is shown that $r(t)$ is a measurement of error between real velocity $\dot{q}(t)$ and reference velocity $\dot{q}_r(t)$. This reference velocity must not be confused with the desired velocity $\dot{q}_d(t)$. In fact, reference velocity is defined as follows

$$\dot{q}_r(t) = \dot{q}_d(t) + \Lambda e(t) \tag{51}$$

where Λ is a diagonal matrix of design parameters with big positive elements so that the system is BIBO stable. This matrix allows to filter errors so that no acceleration of errors $\ddot{e}(t)$ will appear in the error dynamic equation. The definition of filtered error $r(t)$ in terms of position and velocity errors can be obtained from (50) and (51)

$$r(t) = \dot{e}(t) + \Lambda e(t)$$

Substituting $q(t)$ from (50) into the plant, the error dynamic equation is derived.

$$M(q)\dot{r} = -C(q,\dot{q})r + f(x) - \tau + \tau_d$$

where $f(x) = M(q)\ddot{q}_r + C(q,\dot{q})\dot{q}_r + G(q) + F(q,\dot{q})$ stands for non-linear terms to be compensated. This non-linear function could be parametrised by $x^T = \left(q^T,\dot{q}^T,\dot{q}_r^T,\ddot{q}_r^T\right)^T$ or using the definition of $\dot{q}_r(t)$ in (51) by $x^T = \left(e^T,\dot{e}^T,q_d^T,\dot{q}_d^T,\ddot{q}_d^T\right)^T$.

A structural property of robot manipulators is the linearity of parameters (LIP) (Craig, 1989), (Sciavicco, 2002). This means that non-linearities can be split up into a parameter vector P and a vector of basis functions $\Psi(x)$. Therefore the non-linearity function $f(x)$ can be expressed in this sense adding a term of error ε .

$$f(x) = \Psi(q,\dot{q},\dot{q}_r,\ddot{q}_r)P + \varepsilon$$

Linearity of parameters (LIP) is a first assumption in most of the adaptive controllers.

Proceeding in a similar way, the approximation of the non-linear function using estimation of the parameters $\hat{P}$ can be represented by means of linearity of parameters.

$$\hat{f}(x) = \Psi(q, \dot{q}, \dot{q}_r, \ddot{q}_r)\hat{P}$$

4.2 Controller Structure, Control Law and Updtating Law.

Up unto this point, LIP property has been analyzed via fundamental matrices. The control approach employs an *inertia-related linearization approach*, i.e. a conservation of energy formulation, as an attempt to derive update laws and control laws. To be specific, it is required to define an inertia-related Lyapunov function in the stability analysis which utilizes physical properties inherent to a mechanical manipulator (such as those presented above). Thus, the stability of the tracking error system is ensured by formulating the adaptive update rule and by analyzing the stability of the tracking error system at the same time. It is well known that dynamic models even though quite complex are anyhow an idealization of reality. Specifically, robots show uncertainties that are mainly found from two sources: variability of parameters and nonlinearity terms in the system. One way of dealing with parametric uncertainties would be to use the inertia-related approach, see (Lewis et al., 2003). The benefits of this approach as compared to others is that avoids a direct measurement of acceleration and the invertibility of the generalized inertia matrix, which are restrictions of some controllers such as those inspired in the adaptive computed-torque approaches.

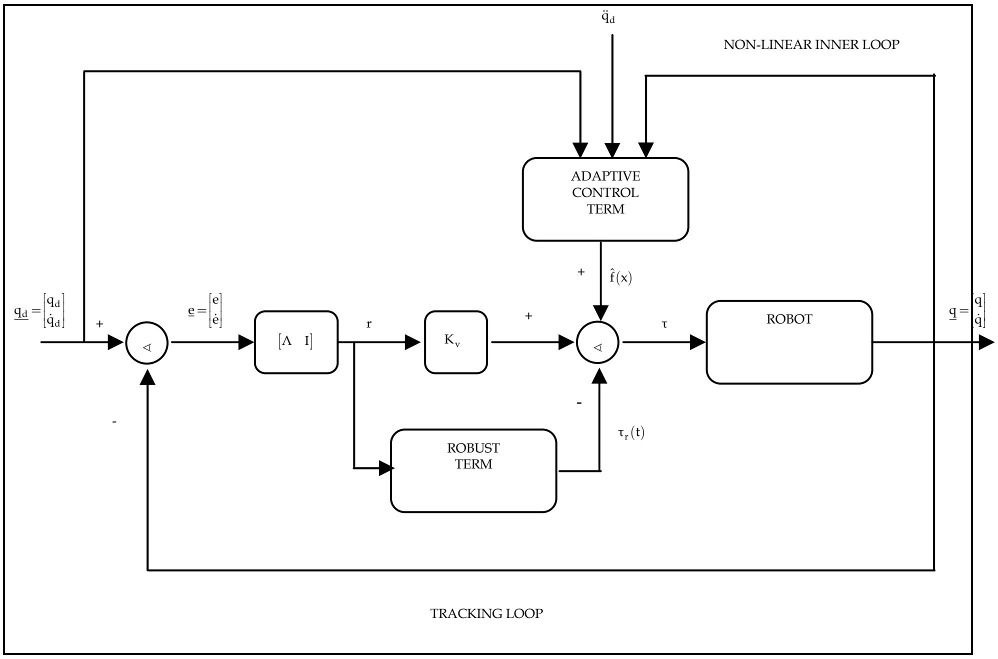

$\ddot{q}_d$
NON-LINEAR INNER LOOP
ADAPTIVE CONTROL TERM
+
$\hat{f}(x)$
$\underline{q}_d = \begin{bmatrix} q_d \\ \dot{q}_d \end{bmatrix}$
+
$\underline{e} = \begin{bmatrix} e \\ \dot{e} \end{bmatrix}$
$[\Lambda \quad I]$
r
K_v
+
τ
ROBOT
$\underline{q} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$
-
-
$\tau_r(t)$
ROBUST TERM
TRACKING LOOP

Fig. 1. Adaptive Control Structure

The controller consists of three terms: a PD-controller to guarantee a good tracking performance, $\tau_{pd} = K_v r = K_v(\dot{e} + \Lambda e)$, a compensator of non-linearities, $\tau_{nl} = \hat{f}$ and a robust controller to absorb unmodelled dynamics τ_r .

$$\tau = K_v r + \hat{f} + \tau_r \tag{52}$$

where $K_v = K_v^T > 0$ is the gain matrix.
The control structure appears in figure 1. In this scheme, two loops can be spoted: an outer loop to track signals and an inner loop to compensate non-linearities. The inner loop is driven by an adaptive control and the outer loop is driven by a robust and PD terms.
An important feature of this class of controllers is that of being based on the all-important *closed-loop error dynamics*, which results from the substitution of the filtered tracking error into the robot dynamics

$$M(q)\dot{r} = -C(q,\dot{q})r + f(x) - \tau = -C(q,\dot{q})r - K_v r + \tilde{f}(x) - \tau_r \tag{53}$$

where $\tilde{f}(x) = f(x) - \hat{f}(x)$ stands for the functional estimation error with x being a vector of appropriate variables as shown below. The nominal nonlinearity f can be computed as
$f(q,\dot{q},\dot{q}_r,\ddot{q}_r) = M(q)\ddot{q}_r + C(q,\dot{q})\dot{q}_r + G(q)$
where $\ddot{q}_r$ is the reference acceleration, $\ddot{q}_r(t) \triangleq \ddot{q}_d(t) + \Lambda\dot{e}(t)$, and $\dot{q}_r$ the reference velocity, $\dot{q}_r(t) \triangleq \dot{q}_d(t) + \Lambda e(t)$. The nonlinearity f can be conveniently partitioned into several smaller terms, resulting into an added controller structure
$f(x) = f_m(x) + f_c(x) + f_g(x)$
where $f_g(x) = G(q)$. The terms f_m and f_c can be defined in terms of $\ddot{q}_r$ and $\dot{q}_r$ as follows

$$\begin{aligned} f_m(q,\ddot{q}_r) &= M(q)\ddot{q}_r = \Phi_m(q,\ddot{q}_r)\xi_m \\ f_c(q,\dot{q},\ddot{q}_r) &= C(q,\dot{q})\dot{q}_r = \Phi_c(q,\dot{q},\ddot{q}_r)\xi_m \\ f_g(q) &= G(q) = \Phi_g(q)\xi_g \end{aligned}$$

where $\xi_m \in \mathbb{R}^{l_m}$, $\xi_g \in \mathbb{R}^{l_g}$ are parameter vectors and Φ_m, Φ_c, Φ_g regression matrices.

Theorem 1: Let the desired trajectory $q_d(t)$ bounded by q_B , i.e. $\|q_d(t)\| \le q_B$. Suppose that the approximation error ε and unmodeled disturbances $\tau_d(t)$ are upper bounded by ε_N and d_B respectively. Let the control law given by (52) with a robust term, $\tau_r(t) = K_r \mathrm{sgn}(r)$ where $K_r = \mathrm{diag}(k_{r_{ii}}) > 0$ with $k_{r_{ii}} \ge \varepsilon_n + d_B$. The next parameter updating laws are considered

$$\dot{\hat{\xi}}_m = \Gamma_m\left(\Phi_m^T(q,\ddot{q}_r) + \Phi_c^T(q,\dot{q},\dot{q}_r)\right)r =$$
$$= \Gamma_m\left(\left(\ddot{q}_r \otimes \Psi_m^T(q)\right)\frac{1}{2}\left[\psi^T(q,\dot{q}_r)(\dot{q}\otimes I_n) + \psi^T(q,\dot{q})(I_{n^2} - P(n,n))(\dot{q}_r \otimes I_n)\right]\right)r$$
$$\dot{\hat{\xi}}_g = \Gamma_g \Phi_g^T(q) r$$

with $\Gamma_m = \Gamma_m^T > 0$, $\Gamma_g = \Gamma_g^T > 0$ and $\Gamma_f = \Gamma_f^T > 0$ symmetric positive-definite constant matrices. Then tracking error $r(t) \to 0$ as $t \to \infty$ and parameters $\hat{\xi}_m$ and $\hat{\xi}_g$ are bounded $\left(\|\hat{\xi}\| \le \xi_B\right)$. It can be concluded that $e \in \mathcal{L}_2^n \cap \mathcal{L}_\infty^n$, e is continuous, $e(t) \to 0$ and $\dot{e}(t) \to 0$ as $t \to \infty$; and τ is bounded.

Proof: Select the following inertia-related Lyapunov-like function:

$$V = \frac{1}{2} r^T M(q) r + \frac{1}{2}\tilde{\xi}_m^T \Gamma_m^{-1} \tilde{\xi}_m + \frac{1}{2}\tilde{\xi}_g^T \Gamma_g^{-1}\tilde{\xi}_g \tag{54}$$

where $\tilde{\xi}_m \in \mathbb{R}^{l_m}$, $\tilde{\xi}_g \in \mathbb{R}^{l_g}$ are the parameter error vectors, $\Gamma_m \in M_{l_m \times l_m}(\mathbb{R})$, $\Gamma_g \in M_{l_g \times l_g}(\mathbb{R})$ diagonal, positive-definite matrices. Note that the term $\frac{1}{2}r^T M(q) r$ stands for the kinetic energy with filtered error instead of joint velocities, and $\frac{1}{2}\tilde{\xi}^T \Gamma^{-1}\tilde{\xi}$ is the energy attached to the non-linear terms to be compensated. It is noted that in (54) , an energy for Coriolis/centripetal terms does not appear because these terms are obtained from inertia matrix approximation using Christoffel symbols of first kind. Matrices Γ_m , Γ_g and Γ_f are diagonal and their non-null terms represent the learning rate of parameters to nominal values, for inertia and gravitational respectively.
Differentiating (54) with respect to time gives

$$\dot{V} = r^T M(q)\dot{r} + \frac{1}{2} r^T \dot{M}(q) r + \tilde{\xi}_m^T \Gamma_m^{-1} \dot{\tilde{\xi}}_m + \tilde{\xi}_g^T \Gamma_g^{-1} \dot{\tilde{\xi}}_g \tag{55}$$

From (55) it is clear that we must substitute for the variable $\dot{r}$; therefore we must write the robot equation in terms of the variable r . Using (53) the robot dynamics can be rewritten as

$$M(q)\dot{r} = -C(q,\dot{q})r - K_v r + \tilde{f}_m(q,\ddot{q}_r) + \tilde{f}_c(q,\dot{q},\ddot{q}_r) + \tilde{f}_g(q) - \tau_r$$

The functional estimation error $\tilde{f}(x)$ can be expressed in terms of regression emulators as indicated below

$$\begin{aligned}\tilde{f}_m(q,\ddot{q}_r) &= \Phi_m(q,\ddot{q}_r)\xi_m + E_m(q,\ddot{q}_r)\\ \tilde{f}_c(q,\dot{q},\ddot{q}_r) &= \Phi_c(q,\dot{q},\ddot{q}_r)\xi_m + E_c(q,\dot{q},\ddot{q}_r)\\ \tilde{f}_g(q) &= \Phi_g(q)\xi_g + \varepsilon_g(q)\end{aligned} \tag{56}$$

Owing to the passivity property of robots $\dot{M}(q) - 2C(q,\dot{q})$ is a skew symmetric matrix and its quadratic form is zero:

$$\dot{V}(r) = -r^T K_v r + r^T\left(\tilde{f}_m(q,\ddot{q}_r) + \tilde{f}_c(q,\dot{q},\ddot{q}_r) + \tilde{f}_g(q)\right) + r^T\tau_r + \tilde{\xi}_m^T \Gamma_m^{-1}\dot{\tilde{\xi}}_m + \tilde{\xi}_g^T \Gamma_g^{-1}\dot{\tilde{\xi}}_g \quad (57)$$

This is the same type of parametric separation that appears in the formulation of the adaptive computed-torque controller; however, note that regression matrices are not a function of joint acceleration $\ddot{q}$. Now, substituting (56) into (57) gives

$$\dot{V} = -r^T K_v r + \left[r^T(\Phi_m(q,\ddot{q}_r) + \Phi_c(q,\dot{q},\dot{q}_r)) + \dot{\tilde{\xi}}_m^T \Gamma_m^{-1} \right]\tilde{\xi}_m + \left[r^T\Phi_g(q) + \dot{\tilde{\xi}}_g^T \Gamma_g^{-1} \right]\tilde{\xi}_g$$
$$+ r^T(\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r) + \tau_d)$$

where

$$\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r) = E_m(q,\ddot{q}_r) + \frac{1}{2}(E_D(q,\dot{q}_r)\dot{q}_r - E_D^T(q,\dot{q}_r)\dot{q}_r + E_v(q,\dot{q})\dot{q}_r) + \varepsilon_g + \varepsilon_f \quad (58)$$

Now, the strategy consists of making zero the terms in brackets, which results in the following adaptive update rules:

$$\dot{\hat{\xi}}_m = \Gamma_m\left(\Phi_m^T(q,\ddot{q}_r) + \Phi_c^T(q,\dot{q},\dot{q}_r)\right)r =$$
$$= \Gamma_m\left(\left(\ddot{q}_r \otimes \Psi_m^T(q)\right)\frac{1}{2}\left[\psi^T(q,\dot{q}_r)(\dot{q}\otimes I_n) + \psi^T(q,\dot{q})(I_{n^2} - P(n,n))(\dot{q}_r \otimes I_n)\right]\right)r$$
$$\dot{\hat{\xi}}_g = \Gamma_g \Phi_g^T(q) r$$

As long as $\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r)$ and $\tau_d(t)$ are bounded, $\dot{V}(r)$ is also upper bounded.

$$\dot{V}(r) \le -K_{v_{min}}\|r\|^2 + r^T(\varepsilon_N + d_B) - r^T K_r \mathrm{sgn}(r) \quad (59)$$

If $K_{r_{ii}} \ge \varepsilon_n + d_B$, and using lemma 6 in the appendix, $r^T(-K_r\mathrm{sgn}(r) + \varepsilon_N + d_B) \le 0$, so that, $\dot{V}(r)$ is negative defined

$$\dot{V}(r) \le -K_{v_{min}}\|r\|^2$$

Hence, $V(r)$ is a Lyapunov function, $V(r) > 0$ and $\dot{V}(r) \le 0$ so that stability in the sense of Lyapunov is guaranteed.

$$\sigma_{min}(K_v)\int_0^t r^T r\, d\tau \le \int_0^t r^T K_v r\, d\tau \le V(0)$$

where $\sigma_{\min}(K_v)$ is the minimum eigenvalue of K_v. Since $V(0)$ and $\sigma_{\min}(K_v)$ are positive constants, $r(t)$ is a signal of finite energy $\left(r(t) \in \mathcal{L}_2^n\right)$ and this implies that is bounded. From lemma 7 in the appendix, $e \in \mathcal{L}_2^n \cap \mathcal{L}_\infty^n$, e is continuous and $e \to 0$ as $t \to \infty$, and $\dot{e} \in \mathcal{L}_2^n$. On the other hand, $\dot{V}(r)$ is semidefinite negative, so that $0 \le V(r(t)) \le V(r(0))$ $\forall t \ge 0$. Since V is lower bounded by zero and $\dot{V}$ is nonpositive, it follows that V approaches a finite limit, which can be written as

$$-\int_0^\infty \dot{V}(r)dt < \infty$$

and this implies that $V(t) \in \mathcal{L}_\infty$ and that $\hat{\xi}_m$ and $\hat{\xi}_g$ are bounded. Observing that $M^{-1}(q)$, parameters ξ_m and ξ_g, and regression functions Φ are bounded, and $r(t) \in \mathcal{L}_2^n$, $q_d, \dot{q}_d, \ddot{q}_d, \tau_r \in L_\infty^n$, then from error filtered dynamics as given by (53) the boundedness of $\dot{r}$ and $\tau(t)$ is verified ($\dot{r}, \tau \in L_\infty^n$). From the theorem 2 in the appendix and given that $e \in \mathcal{L}_2^n$, it follows that $r \in \mathcal{L}_2^n \cap \mathcal{L}_\infty^n$, $\dot{r} \in \mathcal{L}_2^n$, r is continuous and $r(t) \to 0$ as $t \to \infty$. Hence, $\dot{e} \to 0$ as $t \to \infty$.

4.2 Semi-global Stability. Initial Conditions Dependence.

At this point a number of comments are in order. First of all it is interesting to explore in-depth the initial conditions that must be satisfied by both the position errors and the velocity errors so as to guarantee the approximations to be valid. Thus, in the following remark these conditions are discussed from the Stone-Weierstrass theorem and from some results concerning with the stability of non-linear systems. The importance of developing this analysis stems from the fact that the approximations are only valid over a compact set whose size depends on a given desired accuracy.

It is assumed that the joint i has physical limits $q_{\min_i}$ and $q_{\max_i}$ so that the workspace Ω_q can be described as $[q_{\min_1}, q_{\max_1}] \times \ldots \times [q_{\min_n}, q_{\max_n}]$. For the subsequent discussion, it is worth considering the following weighting norm

$$\|x\|_{\rho,\infty} = \max_{1 \le i \le n} \left\{ \frac{|x_i|}{\rho_i} \right\}$$

where $\rho \in \mathbb{R}^n$ is a given vector. As a result, the compact set Ω_q can be represented in the form of an $n-$ ball, i.e.

$$B(q_0, \rho) = \left\{ q \in \mathbb{R}^n \middle| \|q - q_0\|_{\rho,\infty} \le 1 \right\}$$

where

$$q_{0_i} = \frac{q_{\min_i} + q_{\max_i}}{2}$$
$$\rho_i = \frac{q_{\max_i} - q_{\min_i}}{2}$$

Remark As it was claimed above, the desired trajectories remain bounded for all time. This means that the vector $q_d(t)$ is in the compact set $\Omega_d = \left\{ q_d \in \mathbb{R}^n \,|\, \|q_d\| \le q_B \right\}$ for an appropriate upper bound q_B. From the Stone-Weierstrass theorem, it is possible to find an emulator with a sufficient number of nodes and, with suitable parameters, satisfying the following condition

$$\|f_m(q,\ddot{q}_r) - \Phi_m(q,\ddot{q}_r)\xi_m\| \le \|E_m(q,\ddot{q}_r)\|$$

for a given vector $\ddot{q}_r$, and over a compact set Ω_m including the workspace of the robot manipulator. In this, it is convenient to recall that $E_m(q,\ddot{q}_r) = \varepsilon_m(q)\ddot{q}_r$ where $\varepsilon_m(q) \in \mathbb{R}^{n\times n}$ denotes the estimation error for the inertia generalized matrix. Note that the compact set Ω_q denotes an operation region of interest in which the system works on. This implies that the obtained result is regionally stable in the sense that all the states $q(t)$ must be guaranteed within the compact Ω_q. Furthermore, this compact set should include the space of desired trayectories, i.e. $\Omega_d \subseteq \Omega_q$.
In a similar way, it can be stated the following validity regions for the approximation of fundamental matrices M_D and M_V

$$\|M_D(q,\dot{q})\dot{q}_r - \Phi_D(q,\dot{q},\dot{q}_r)\xi_m\| \le \|E_D(q,\dot{q})\dot{q}_r\|$$
$$\|M_V(q,\dot{q})\dot{q}_r - \Phi_v(q,\dot{q},\dot{q}_r)\xi_m\| \le \|E_V(q,\dot{q})\dot{q}_r\|$$
$$\left\|M_D^T(q,\dot{q})\dot{q}_r - (\dot{q}\otimes I_n)P(n,n)\psi(q,\dot{q}_r)\xi_m\right\| \le \left\|E_D^T(q,\dot{q})\dot{q}_r\right\|$$

for a given vectors $\dot{q},\ \dot{q}_r \in \mathbb{R}^n$ and for all q in Ω_q describing the validity region. It is clear from this discussion that the ideal parameters are derived as the following optimization problem

$$\xi_m \triangleq \min_{\substack{\xi\in\mathbb{R}^{L_m n}\\ q\in\Omega_q}} \{\|f_m(q,\ddot{q}_r) + f_c(q,\dot{q},\dot{q}_r) - \Phi_m(q,\ddot{q}_r)\xi_m - \Phi_c(q,\dot{q},\ddot{q}_r)\xi_m\|\}$$

In the last expression L_m denotes the number of nodes required for the inertia approximation.
The same observations can be done for the gravitational vector $G(q)$. Thus, the validity region Ω_g for the approximation of this term is determined by the condition

$$\left\|G(q) - \Phi_g(q)\xi_g\right\| \le \left\|\varepsilon_g(q)\right\|$$

for all $q \in \Omega_q$. Evidently, for this region to be useful, it should be included the workspace

Ω_q . Just like for the inertia terms, the ideal parameters are derived as

$$\xi_g \triangleq \min_{\substack{W\in\mathbb{R}^{L_m\times n}\\ q\in\Omega_q}} \left\{\left\|G(q)-\Phi_g(q)\xi_g\right\|\right\}$$

It is known that the error $e(t)$ is connected to the filtered tracking error $r(t)$ via a transfer function $G_r(s)=sI+\Lambda$ with $\Lambda\in\mathbb{R}^{n\times n}$ a Hurwitz matrix. From lemma 8 in the appendix, it should be clear that being $G_r(s)$ a strictly proper, exponentially stable transfer function, then $r\in\mathcal{L}_2^n \Rightarrow e\in\mathcal{L}_2^n\cap\mathcal{L}_\infty^n$, $\dot{e}\in\mathcal{L}_2^n$, e is continuous and $e(t)\to 0$ as $t\to\infty$. If in addition $r\to 0$ as $t\to\infty$, then $\dot{e}\to 0$. Furthermore, it is assumed that the filtered tracking error $r\in\mathcal{L}_\infty^n$, i.e. there exists a constant $b_r\in\mathbb{R}$ such that $\|r\|\le b_r$. Actually, this condition should be guaranteed by the controller. If the differential equation describing the closed-loop filtered tracking error dynamics is solved, it is easy to see the following relationship

$$e(t)=e^{-\Lambda(t-t_0)}e(t_0)+\int_{t_0}^{t}e^{-\Lambda(t-\tau)}r(\tau)d\tau$$

Due to $\|r\|_{\rho,\infty}\le b_{r,\infty}$, being $b_{r,\infty}\in\mathbb{R}$ a constant, it is straightforward to derive an upper bound for the error norm

$$\|e(t)\|_{\rho,\infty}\le\|e(t_0)\|_{\rho,\infty}+\frac{b_{r,\infty}}{\lambda_{\min}\{\Lambda\}}\left\|e^{-\Lambda(t-t_o)}-1\right\|_{\rho,\infty}\le\|e(t_0)\|_{\rho,\infty}+\frac{b_{r,\infty}}{\lambda_{\min}\{\Lambda\}} \qquad (60)$$

The universal property of approximation holds so long as $q(t)\in\Omega_q$ or equivalently whenever $\|q(t)-q_0\|_{\rho,\infty}\le 1$. Applying the triangle inequality to (60) and from the boundedness of the desired trajectory (i.e. $\|q_d(t)-q_0\|_{\rho,\infty}\le q_{d,\infty}$), it is not difficult to write the following $\|q(t)-q_0\|_{\rho,\infty}\le q_{d,\infty}+\|e(t_0)\|_{\rho,\infty}+\frac{b_{r,\infty}}{\lambda_{\min}\{\Lambda\}}\le 1$.
Note that owing to $\Omega_d\subseteq\Omega_q$, the bound $q_{d,\infty}$ satisfies the condition $q_{d,\infty}\le 1$. Now the radius $b_{r,\infty}$ is determined,

$$b_{r,\infty}\le\lambda_{\min}\{\Lambda\}\left(1-q_{d,\infty}-\|e(t_0)\|_{\rho,\infty}\right) \qquad (61)$$

As a consequence, the initial filterd tracking error must satisfy the condition $\|r(t_0)\|\le b_{r,\infty}$ for the constant $b_{r,\infty}$ given in (61) .
It is known that $\|r(t_0)\|_{\rho,\infty}\le\|\dot{e}(t_0)\|_{\rho,\infty}+\lambda_{\max}\{\Lambda\}\|e(t_0)\|_{\rho,\infty}$ and then, the condition $\|r(t_0)\|\le b_{r,\infty}$ is held whenever

$$\|\dot{e}(t_0)\|_{\rho,\infty} + \lambda_{\max}\{\Lambda\}\|e(t_0)\|_{\rho,\infty} \leq \lambda_{\min}\{\Lambda\}\left(1 - q_{d,\infty} - \|e(t_0)\|_{\rho,\infty}\right) \tag{62}$$

It is concluded that the initial errors $e(t_0)$ and $\dot{e}(t_0)$ must verify the inequality (62) , where the constants Λ and $q_{d,\infty}$ are known a priori.

Remark If the robust term is removed from the control law, it is necessary to account for an additional assumption of persistence of excitation (PE) on the signals Φ_g and Φ_m . In this case, the variation of the energy is upper bounded in the following way

$$\dot{V} \leq -\lambda_{\min}\{K_v\}\|r\|^2 + \|r\|(\varepsilon_N + d_B)$$

Obviously the energy is decreasing as long as the size of the filtered tracking error remains greater than a constant, i.e.

$$\|r\| \geq \frac{\varepsilon_N + d_B}{\lambda_{\min}\{K_v\}}$$

As it was pointed out in the previous discussion, the validity of the approximation holds throughout whenever $r \in \Omega_r = \left\{ r \in \mathbb{R}^n \middle| \|r\|_{\rho,\infty} \leq b_{r,\infty} \right\}$, being $b_{r,\infty}$ given in (61) . Under these considerations, it is easy to derive a suitable selection for the derivative gains as

$$\lambda_{\min}\{K_v\} \geq \frac{(\varepsilon + d_B)}{\lambda_{\min}\{\Lambda\}\left(1 - q_{d,\infty} - \|e(t_0)\|_{\rho,\infty}\right)}$$

Unfortunately, the tracking errors do not vanish, but are bounded by small enough values. Since $\|r\|_{\rho,\infty} \leq b_{r,\infty}$, the errors e and $\dot{e}$ are also guaranteed to be bounded. Additionally to this result, the boundedness of the desired trajectory also implies that the states q and $\dot{q}$ are bounded. To complete the analysis the boundedness of the parameters is now explored. To this purpose, it is worth considering the dynamics relative to $\tilde{\xi}_m$ and $\tilde{\xi}_g$

$$\begin{cases} \dot{\tilde{\xi}}_m = -\Gamma_m\left(\left(\ddot{q}_r \otimes \Psi_m^T(q)\right)\frac{1}{2}\left[\psi^T(q,\dot{q}_r)(\dot{q} \otimes I_n) + \psi^T(q,\dot{q})\left(I_{n^2} - P(n,n)\right)(\dot{q}_r \otimes I_n)\right]\right)r \\ \dot{\tilde{\xi}}_g = -\Gamma_g \Phi_g^T(q) r \\ y_m = \Phi_m(q,\ddot{q}_r)\tilde{\xi}_m \\ y_g = \Phi_g(q)\tilde{\xi}_g \end{cases}$$

It can be shown that the boundedness can be obtained by reformulating the problem in terms of the Kronecker product. This consideration leads to the following vector dynamics

$$\begin{pmatrix} \hat{\xi}_m \\ \hat{\xi}_g \end{pmatrix} = \begin{pmatrix} \Gamma_m\left(\left(\ddot{q}_r \otimes \Psi_m^T(q)\right)\frac{1}{2}\left[\psi^T(q,\dot{q}_r)(\dot{q} \otimes I_n) + \psi^T(q,\dot{q})(I_{n^2} - P(n,n))(\dot{q}_r \otimes I_n)\right]\right)r \\ \Gamma_g \Phi_g^T(q) r \end{pmatrix}$$

$$\begin{pmatrix} y_m \\ y_g \end{pmatrix} = Q(q,\dot{q},\dot{q}_r,\ddot{q}_r)\begin{pmatrix} \hat{\xi}_m \\ \hat{\xi}_g \end{pmatrix}$$

where the matrix Q is defined as

$$Q \triangleq \begin{pmatrix} \Phi_m(q,\ddot{q}_r) & 0 \\ 0 & \Phi_g(q) \end{pmatrix}$$

Now, the PE condition is equivalent to the persistence of excitation of $\Phi_g(q)$ and of $\Phi_m(q,\ddot{q}_r)$, and consequently to the uniform complete observability of this system. From the lemma 9, the boundedness of $y_m(t)$, $y_g(t)$, and $r(t)$ insures the boundedness of $\tilde{\xi}_m$ and of $\tilde{\xi}_g$, and then the boundedness of $\hat{\xi}_m$ and of $\hat{\xi}_g$. With reference to the theorem 3, due to $\|r\|$ is greater than a constant, it can be asserted that the system is ultimately uniformly bounded (UUB).

Remark: Defining the state $x = \left(r^T, \tilde{\xi}_m^T, \tilde{\xi}_g^T\right)^T$ the energy can be written as

$$V(x) = \frac{1}{2}x^T P x$$

where

$$P = \begin{pmatrix} M(q) & 0_{n\times nL_m} & 0_{n\times nL_g} \\ 0_{nL_m\times n} & \Phi_m(q,\ddot{q}_r) & 0_{nL_m\times nL_g} \\ 0_{nL_g\times n} & 0_{nL_g\times nL_m} & \Phi_g(q) \end{pmatrix}$$

In this case the energy is lower and upper bounded,

$$\frac{1}{2}\|x\|^2 \le V(x) \le \frac{1}{2}\|x\|^2$$

where the constants λ_1 and λ_2 are defined as follows

$$\lambda_1 = \min\left\{\lambda_{\min}\{M(q)\}, \frac{1}{\lambda_{\max}\{\Gamma_m\}}, \frac{1}{\lambda_{\max}\{\Gamma_g\}}\right\}$$

$$\lambda_2 = \max\left\{\lambda_{\max}\{M(q)\}, \frac{1}{\lambda_{\min}\{\Gamma_m\}}, \frac{1}{\lambda_{\min}\{\Gamma_g\}}\right\}$$

Applying the passivity property of robot manipulators on the variation of the energy with respect to time leads to the following expression

$$\dot{V} = -r^T K_v r + r^T(\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r) + \tau_d) + \left(r^T \Phi_m(q,\ddot{q}_r) + \dot{\tilde{\xi}}_m^T \Gamma_m^{-1} \right) \tilde{\xi}_m + \left(r^T \Phi_g(q) + \dot{\tilde{\xi}}_g^T \Gamma_g^{-1} \right) \tilde{\xi}_g$$

Folding the parameter updating law (62) into the last expression, the variation of the energy becomes

$$\dot{V} = -r^T K_v r + r^T(\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r) + \tau_d)$$

Assuming that the external disturbance torque and the functional reconstruction error are uniformly bounded, i.e. $\|\tau_d\| \le d_B$ and $\|\varepsilon(q,\dot{q},\dot{q}_r,\ddot{q}_r)\| \le \varepsilon_B$, the power will be bounded by

$$\dot{V} \le -\lambda_{min}\{K_v\}\|r\|^2 + (d_B + \varepsilon_B)\|r\|$$

with $\lambda_{min}\{K_v\}$ representing the minimum eigenvalue of the gain matrix K_v . Owing to ε_B and d_B are constants, $\dot{V}$ is semidefinite negative as long as $\|r\| > \frac{(d_B+\varepsilon_B)}{\lambda_{min}\{K_v\}}$. Hence,

$$\frac{(d_B + \varepsilon_B)}{\lambda_{min}\{K_v\}} < \|x(t)\| < \infty$$

Invoking the theorem 4 in the appendix, it is concluded that

$$\|x(t)\| < \frac{(d_B + \varepsilon_B)}{\lambda_{min}\{K_v\}}\sqrt{\frac{\lambda_2}{\lambda_1}} \quad \forall t \in [t_0 + T, \infty)$$

where $0 \le T < \infty$. In this way, it is stated the ultimately uniformly boundedness of x .
Remark: In the ideal case, when $\varepsilon_N = d_B = 0$, it is easy to show the asymptotic convergence. In this case the variation of the Lyapunov energy in (59) turns to be

$$\dot{V}(r) = -r^T K_v r$$

Note that in this case no robust term is necessary to be introduced into the control law since there is no functional reconstruction errors nor external disturbances. It can be shown that the second differentiation $\ddot{V}(r) = -2r^T K_v \dot{r}$ is also bounded and uniform continuity of $V(r)$ is guaranteed. Since $V(r) > 0$, $\dot{V}(x,t)$ uniformly continuous and non-positive, involking lemma 7 in the appendix, it is easy to show that $\dot{V}$ vanishes as t goes to infinity. Indeed, by using Barbalat's lemma it can be concluded that $\dot{V}(r) \to 0$ as $t \to \infty$ and that $r(t) \to 0$. Hence, $\dot{e}(t) \to 0$ as $t \to \infty$.
The parameter tuning algorithm is hardly a continuous-time backpropagation algorithm. These parameters are initialized to zero so that there is no preliminary off-line phase and in

the first steps of working the controller behaves just as a PD controller. If the gain K_v is considered to be large, the closed loop error remains bounded and this implies the stabilization of the whole system. This initialization is very important since it is not necessary to choose initial parameters that make stable the system. It is well known that this task is very complex to do.

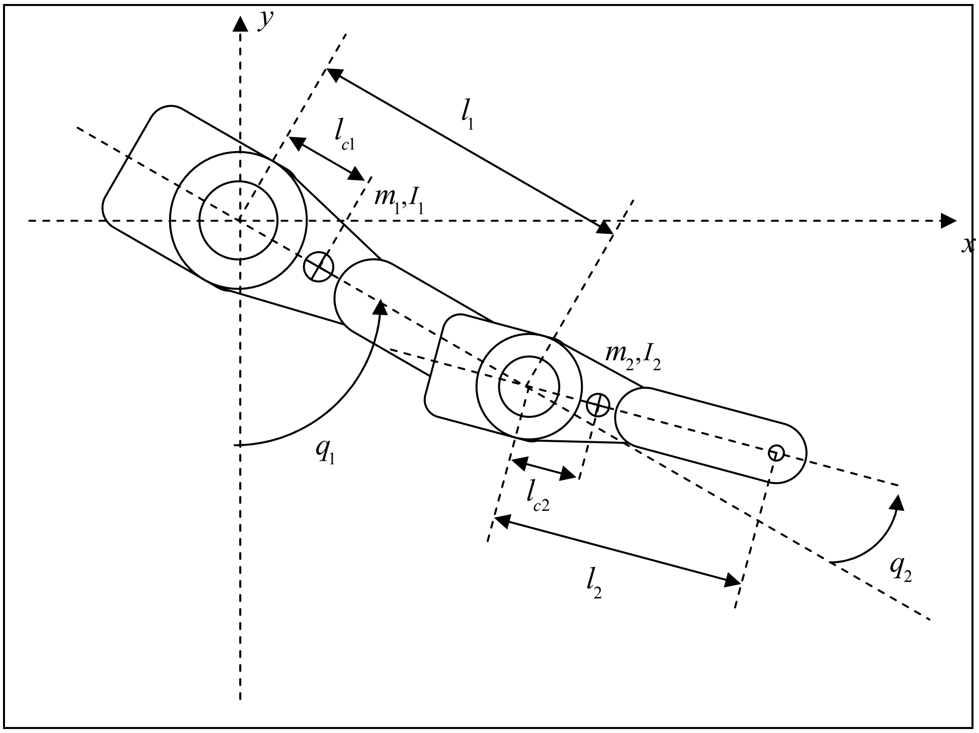


Fig. 2. Two-Link Planar Elbow Arm

5. Results

A planar two-link arm robot is used here as a platform to demonstrate the effectiveness of the adaptive controller. These kind of robots are used widely in the literature to get proof of the effectiveness of the controllers and appears in figure 2. Dynamics can be found in the literature (Spong & Vidyasagar, 1989), (Lewis et al. ,2003) and no friction is taken into account in the model. The generalized inertia matrix is given by

$$M(q)=\begin{pmatrix} m_1 l_{c_1}^2 + m_2 l_1^2 + I_1 & m_2 l_1 l_{c_2}\cos(q_2-q_1) \\ m_2 l_1 l_{c_2}\cos(q_2-q_1) & m_2 l_{c_2}^2 + I_2 \end{pmatrix} \tag{63}$$

where $q^T=(q_1,q_2)^T$ is the joint variable, m_1 , m_2 are the masses of the links, l_1 , l_2 the

length of the links, l_{c_1} , l_{c_2} the position of the center of mass respect to the frame attached to the link and I_1 , I_2 are the inertia about the Z-axis for each link.
The construction of the approximating function in the controller is illustrated here. From (Ge et al., 1998) a superset of basis functions $\varphi_m(q)$ for $M(q)$ is found and this set described completely the robotic system.

$$\varphi_m(q)=\left(1,\sin(q_1),\sin(q_2),\cos(q_1),\cos(q_2)\right)^T \in \mathbb{R}^5 \tag{64}$$

In practise, only a subset of elements is necessary to describe the whole system and this means that $M(q)$ is over parametrised. From (64) the activation function for inertia matrix, $\gamma_m(q,\ddot{q}_r)$ is built as $\gamma_m(q,\ddot{q}_r)=\ddot{q}_r \otimes \varphi_m(q)$. The dimension of $\gamma_m(q,\ddot{q}_r)$ means that the number of nodes required to approximate inertia matrix is 10 .
Coriolis-centripetal matrix is carried out from (58) and this implies to calculate the Jacobian matrix $\psi(q,x)=\sum_{i=1}^{n}\left(x\otimes\frac{\partial\varphi_m(q)}{\partial q_i}\right)e_i^T$. This Jacobian is applied to vectors of dimension 2 in order to work out the Coriolis/centripetal matrix so that the number of nodes is also 10 .
The desired trajectory is chosen as a periodic sinusoid with amplitud 1 and frequency 1 rad/sec.

$$\begin{aligned} q_{d_1}(t)&=\sin(t)\\ q_{d_2}(t)&=\cos(t)\end{aligned} \tag{65}$$

This provides bounded signals for position, velocity and acceleration. It is assumed that initially there is no knowledge about the system so that all the weights are initialized to zero.
The robot parameters are $l_1=l_2=1\text{m}$ and the mases are $m_1=0.8\text{kg}$, $m_2=2.3\text{kg}$. The gains of the controller were chosen as $K_v=\text{diag}(20,20)$, $\Gamma_m=\Gamma_g=\Gamma_f=\text{diag}(50,50)$, $\Lambda=\text{diag}(5,5)$. It is assumed the next initial conditions for signals and estimations, $q(0)=0$, $\dot{q}(0)=0$, $\hat{m}_1(0)=0$, $\hat{m}_2(0)=0$. In this scheme all the parameters are initialized at zero so that there is no preliminary off-line learning phase. The signal tracking and weight tuning works together on-line. The main benefit of this initialization is the independence of the controller as regards the initial parameters skipping the task of finding initial stabilizing parameters as necessary in other control techniques.
The whole system was simulated in SIMULINK using Runge-Kutta method with an integration fixed step of $\Delta t=10^{-3}$ sec and a simulation range from 0 to 20 sec. The response of the controller with parameter tuning (e.g. Theorem 1) appears in figure 3.
It is noted that no knowledge of the dynamics is needed for adaptive control and a good tracking performance is obtained as observed in figure 4. Position error signals remains in a bound band of $\pm 5\%$ respect to the equilibrium point. In this figure an exponential convergence response of the system is showed with a setting time about 5 seconds. It is possible to improve the tracking performance by increasing the gains of the controller.

Therefore, it is concluded that the designed neurocontroller provides a good tracking of desired trajectories.

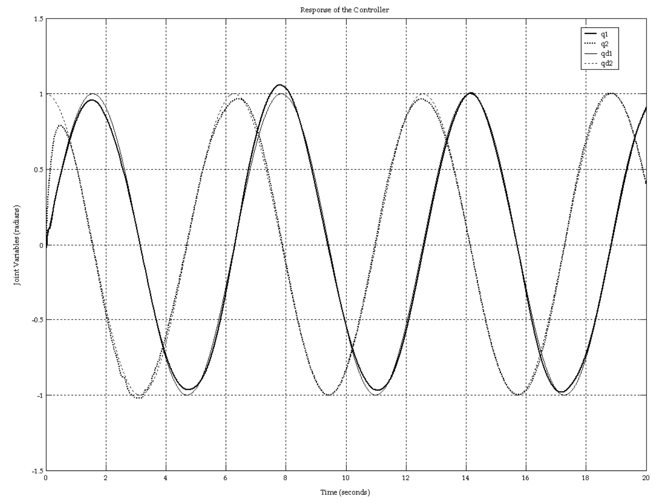


Fig. 3. Response of the Adaptive Controller with Gradient-Type tuning. Actual and desired joint angles.

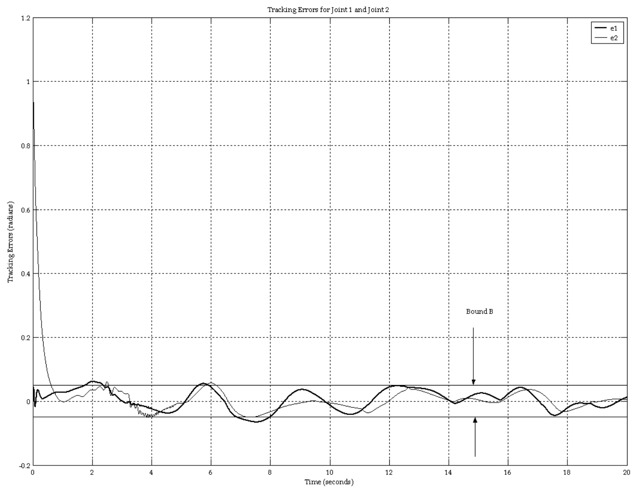


Fig. 4. Response of the controller with gradient-type parameter tuning. Representation of tracking errors

6. Appendix

Lemma 6: For $K = \mathrm{diag}(K_{ii}) \in R^{n\times n}$ and $d = (d_1, d_2, ..., d_n)^T \in R^n$, if $u = -K\mathrm{sgn}(x)$ and $k_{ii} \geq |d_i|$ then $x^T M(u-d) \leq 0$ (Ge et al., 1998).

Lemma 7: Let $V(x,t)$ be a Lyapunov function so that $V(x,t) > 0$, $\dot{V}(x,t) \leq 0$. If $\dot{V}(x,t)$ is uniformly continuous (Lewis et al., 2003), then

$$\dot{V}(x,t) \to 0 \text{ as } t \to \infty \tag{66}$$

The following theorem is very important in control of non-linear systems, and is due to Desoer and Vidyasagar, cf. (Desoer & Vidyasagar, 2008)

Theorem 2: Let the closed-loop transfer function $H(s) \in \mathbb{R}^{n\times n}(s)$ be exponentially stable and strictly proper, and $h(t)$ the corresponding impulse response (obtained by evaluating the inverse Laplace transform of $H(s)$). If $u \in \mathcal{L}_2^n$, then $y = h * u \in \mathcal{L}_2^n \cap \mathcal{L}_\infty^n$, $\dot{y} \in \mathcal{L}_2^n$, y is continuous and $y(t) \to 0$ as $t \to \infty$, where $h * u$ denotes the convolution product of h and u.

On the basis of this theorem, it is possible to state the following lemma, (Ge et al., 1998).

Lemma 8: Let $e(t) = h(t) * r(t)$, where $h = \mathcal{L}^{-1}\{H(s)\}$ and $H(s)$ is an $n \times n$ strictly proper, exponentially stable transfer function. Then $r \in \mathcal{L}_2^n \Rightarrow e \in \mathcal{L}_2^n \cap \mathcal{L}_\infty^n$, $\dot{e} \in \mathcal{L}_2^n$, e is continuous and $e(t) \to 0$ as $t \to \infty$. If in addition $r \to 0$ as $t \to \infty$, then $\dot{e} \to 0$. (Ge et al., 1998).

Theorem 3 (UUB by Lyapunov Analysis): If for system

$$\dot{x} = f(x,t) + g(t) \tag{67}$$

there exists a function $V(x,t)$ with continuous partial derivatives such that for x in a compact set $S \subseteq \mathbb{R}^n$

$$\begin{gathered} V(x,t) \text{ is positive definite, } V(x,t) > 0 \\ \dot{V}(x,t) < 0 \text{ for } \|x\| > R \end{gathered} \tag{68}$$

for some $R > 0$, such that the ball of radius R is contained in S, then the system is UUB and the norm of the state is bounded to within a neighborhood of R.

The following theorem is a modified version of the uniformly ultimately boundedness theorem of Corless and Leitmann, cf. (Corless & Leitmann, 1981). For more insights the reader may refer to theorems 1 and 2 in (Dawson et al., 1990) or the theorem 2.15. p. 65 in (Qu, 1998).

Theorem 4: If V is a Lyapunov candidate function for any given continuous-time system with the properties

$$\lambda_1 \|x(t)\|^2 \leq V(x(t)) \leq \lambda_2 \|x(t)\|^2 \tag{69}$$

$$\dot{V}(x(t)) < 0, \text{ if } \eta_2 > \|x(t)\| > \eta_1 \tag{70}$$

where

$$\eta_2 > \eta_1 \sqrt{\frac{\lambda_2}{\lambda_1}} \tag{71}$$

then

$$\|x(t)\| < \eta_1 \sqrt{\frac{\lambda_2}{\lambda_1}} \quad \forall t \in [t_0 + T, \infty) \tag{72}$$

where T is a finite positive constant.

The following lemma allows to connect the uniform complete observability (UCO) to the boundedness of the states, (Lewis et al., 1999).

Lemma 9 (Technical Lemma): Consider the linear time-varying system $(0, B(t), C(t))$ defined by

$$\begin{aligned} \dot{x} &= B(t)u \\ y &= C(t)x \end{aligned} \tag{73}$$

with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^p$ and the elements of $B(t)$ and $C(t)$ piecewise continuous functions of time. Since the state transition matrix is the identity matrix, the observability grammian is

$$N(t, t_0) = \int_{t_0}^{t} C^T(\tau) C(\tau) d\tau \tag{74}$$

Let the system be uniformly completely observable with $B(t)$ bounded. Then if $u(t)$ and $y(t)$ are bounded, the state $x(t)$ is bounded.

7. References

Corless, M. and Leitmann, G. (1981). Continuous State Feedback Guaranteeing Uniform Ultimate Boundness for Uncertain Dynamics Systems. *IEEE Transactions on Automatic Control*, Vol.26, No. 5, (October 1981) (1139- 1144), ISSN 0018-9286

Dawson, D. M., Qu, Z., Lewis, F. L., and Dorsey, J. F. (1990). Robust Control for the Tracking of Robot Motion. *International Journal of Control*, Vol.52, No. 3, (1990) (581-595), ISSN 0020-7179

Desoer, C. A., and Vidyasagar, M. (2008). *Feedback Systems: Input-Output Properties,* Society for Industrial and Applied Mathematics, ISBN 978-0898716702

Ge, S. S., Lee, T. H. and Harris, C. J. (1998). *Adaptive Neural Network Control of Robotic Manipulators,* World Scientific Publishing Company, ISBN 978-9810234522, London

Horn, R. A., and Johnson, C. R. (1999). *Topics in Matrix Analysis,* Cambridge University Press, ISBN 978-0521467131

Lewis, F. L., Jagannathan, S. and Yesildirek, A. (1999). *Neural Network Control of Robot Manipulators and Nonlinear Systems,* Taylor and Francis Ltd., ISBN 978-0748405961

Lewis, F. L., Dawson, D. M. and Abdallah, C. T. (2003). *Robot Manipulator Control: Theory and Practice,* Marcel Dekker Inc., ISBN 978-0824740726.

Mulero-Martínez, J.I. (2007). Bandwidth of Mechanical Systems and Design of Emulators with RBF. *Neurocomputing*, Vol.70, No.7-9, (2007) (1453-1465), ISSN 0925-2312

Mulero-Martínez, J.I. (2007a). An Improved Dynamic Neurocontroller Based on Christoffel Symbols. *IEEE Transactions on Neural Networks*, Vol.18, No.3, (May 2007) (865-879), ISSN 1045-9227

Mulero-Martínez, J.I. (2007b). Uniform Bounds of the Coriolis/Centripetal Matrix of Serial Robot Manipulators. *IEEE Transactions on Robotics*, Vol.23, No.5, (October 2007) (1083-1089), ISSN 1552-3098

Mulero-Martínez, J.I. (2009). A New Factorization of the Coriolis/Centripetal Matrix. *Robotica*, Vol.27, No.5, (September 2009) (689-700), ISSN 0263-5747

Qu, Z. (1998). *Robust Control of Nonlinear Uncertain Systems,* John Wiley and Sons, ISBN *978-0471115892*

Slotine, J.J. and Li, W. (1991) *Applied Nonlinear Control,* Prentice-Hall, ISBN 978-0130408907

Spong, M. W. and Vidyasagar, M. (1989). *Robot Dynamics and Control,* John Wiley and Sons Inc., ISBN 978-0471612438.

Wen, J. T. (1990). A Unified Perspective on Robot Control: The Energy Lyapunov Function Approach. *International Journal of Adaptive Control and Signal Processing*, Vol.4, No. 6 (November, 1990) (487-500)

Development of a New 2 DOF Lightweight Wrist for the Humanoid Robot ARMAR

Albert Albers, Jens Ottnad and Christian Sander
IPEK - Institute of Product Development, University of Karlsruhe (TH)
Germany

1. Introduction

The mechatronic design of a humanoid robot is fundamentally different from that of industrial robots. Industrial robots generally have to meet requirements such as mechanical stiffness, accuracy and high velocities. The key goal for this humanoid robot is not accuracy, but the ability to cooperate with humans. In order to enable a robot to interact with humans, high standards are set for sensors and control of its movements. The robot's kinematic properties and range of movements must be adjusted to humans and their environment (Schäfer, 2000).

1.1 The Humanoid Robot ARMAR

The collaborative research centre 588 "Humanoid Robots – learning and cooperating multi-modal robots" was established by the "Deutsche Forschungsgemeinschaft" (DFG) in Karlsruhe in May 2001. In this project, scientists from different academic fields develop concepts, methods, and concrete mechatronic components for a humanoid robot called ARMAR (see figure 1) that can share its working space with humans.

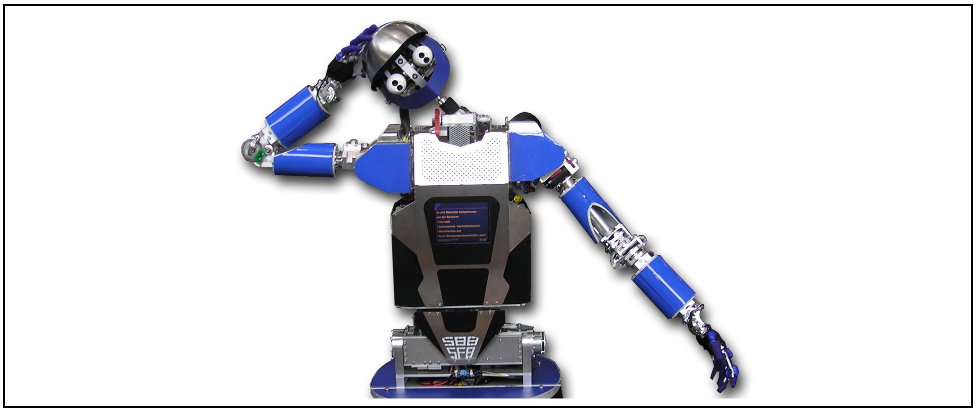


Fig. 1. Upper body of the humanoid robot ARMAR III.

The long-term target is the interactive work of robots and humans to jointly accomplish specified tasks. For instance, a simple task like putting dishes into a dishwasher requires sophisticated skills in cognition and the manipulation of objects. Communication between robots and humans should be possible in different ways, including speech, touch, and gestures, thus allowing humans to interact with the robots easily and intuitively. As this is the main focus of the collaborative research centre, a humanoid upper body on a holonomic platform for locomotion has been developed. It is planned to increase the mobility of ARMAR by replacing the platform with legs within the next years, which will lead to modifications of the upper body.

1.2 State of the Art and Motivation

The focus of this paper is the design and the development process of a new wrist for the humanoid robot ARMAR. The wrist serves as the connection between forearm and hand. An implementation of the new modules is planned for the next generations of the humanoid robot, ARMAR IV and V. The wrist of the current version, ARMAR III, has two degrees of freedom (Albers et al., 2006) and its rotational axes intersect in one point. ARMAR III has the ability to move the wrist to the side (± 60°, adduction/abduction) as well as up and down (± 30°, flexion/extension). This is realized by a universal joint in a compact construction. At the support structure of the forearm all motors for both degrees of freedom are fixed. The gear ratio is obtained by a ball screw in conjunction with either a timing belt or a cable. The load transmission is almost free from backlash. The velocity control and the angular measurement in the wrist are realized by encoders at the motors and by quasi-absolute angular sensors directly at the joint. To measure the load on the hand, a 6-axis force and torque sensor is fitted between the wrist and the hand.

One of the main points of criticism on the current version of the wrist is the offset between the rotational axes and the flange, as shown in figure 2 (left). Due to the joint design, this offset distance is necessary in order to provide the desired range of motion. Also other wrists of humanoid robots show a similar design, see (Shadow), (Kaneko et al., 2004), (Park et al., 2005), (Kaneko et al., 2008). That offset is even greater due to the placement of the 6-axis force and torque sensor. The resulting movement, a circular path performed by the carpus, does not appear as a humanlike motion, as illustrated in figure 2 (right).

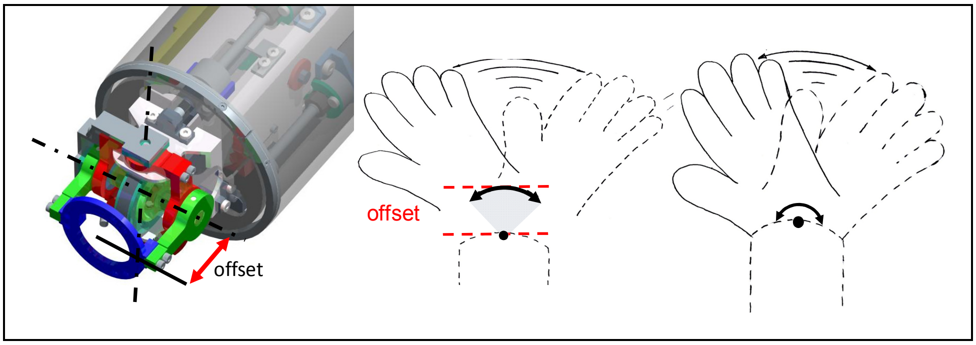


Fig. 2. Offset between the rotational axis and the hand flange at the wrist of the humanoid robot ARMAR III (left) and the resulting movement (right)

The German Aerospace Centre DLR (Deutsches Zentrum für Luft- und Raumfahrt) has been working on seven degree of freedom robot arms for several years. The result of this project is shown in figure 3 (left). Although their work is inspired by a human arm, their goal is not to design humanoid robots. The wrists of the lightweight arms of the third generation imitate human wrist movements by a pitch-pitch combination with intersecting axes (kardanic). An alternative pitch-roll configuration is also utilized, mainly for applications using tools (Albu-Schäffer et al., 2007). Both versions have an offset comparable to the current wrist of ARMAR III.
Henry J. Taylor and Philip N.P. Ibbotson designed a so called "Powered Wrist Joint" (Rosheim, 1989) in order to load and unload space shuttles. The concept of this wrist is illustrated in figure 3 (right). In a smaller version, the basic idea could be reused in humanoid robot's wrist. The second degree of freedom (pitch) of the wrist is guided by a spherical joint. Such an assembly provides a slim design and relatively wide range of motion. The actuators for the second degree of freedom (yaw) are located directly at the joint; therefore, the drive units are quite simple. On the other hand, miniaturization seems to be very difficult due to the dimensions of common gears and motors.

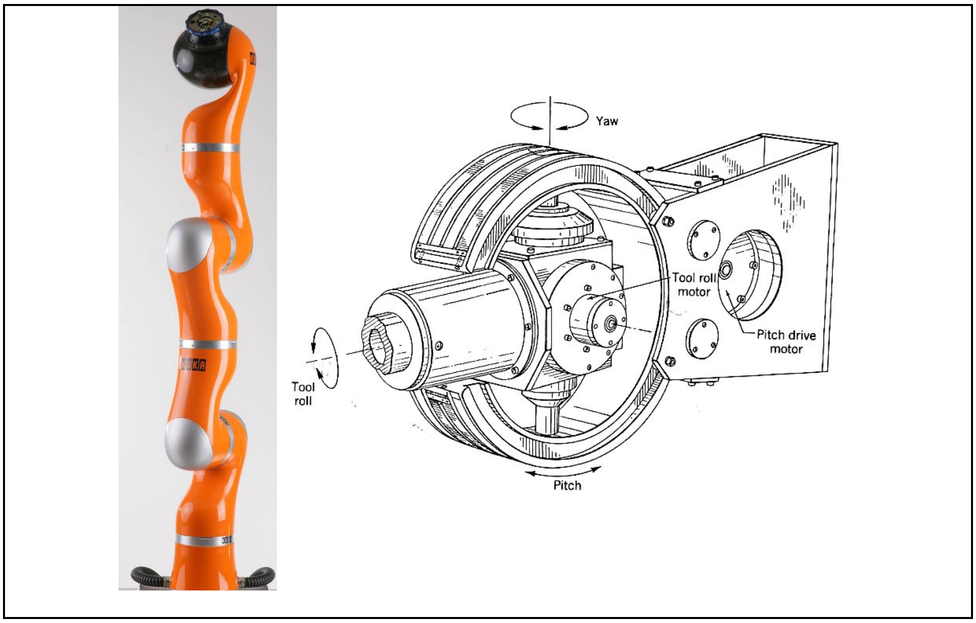


Fig. 3. The DLR/Kuka lightweight robot arm (Abu-Schäfer et al. 2007) (left) and concept for a wrist actuator (Rosheim, 1989) (right).

2. New Concept

2.1 Requirements and Design Goals

In this section the system of objectives is defined. It describes all relevant objectives, their dependence and boundary conditions, which are necessary for the development of the correct object system, outgoing from the current condition to the future condition. But the solution itself is no part of the system of objectives. It is permanently extended and concretized over the complete product lifecycle. The correct, consistently and complete definition of this system is the basis of the successful product development and a core component of the development activity (Albers et al., 2008a). Since the robot is intended to get in contact with humans in order to achieve various functions, it is inevitable that the robot is accepted by the human. The ability to move like a human is as important as a human-like appearance; therefore, specific demands (Asfour, 2003) on kinematics, dynamics and the design space must to be considered. A human wrist consists of many different elements and has a relatively wide range of motions. Figure 4 illustrates the different possible movements of the human wrist along with the corresponding reachable angular position of the joints (Whired, 2001).

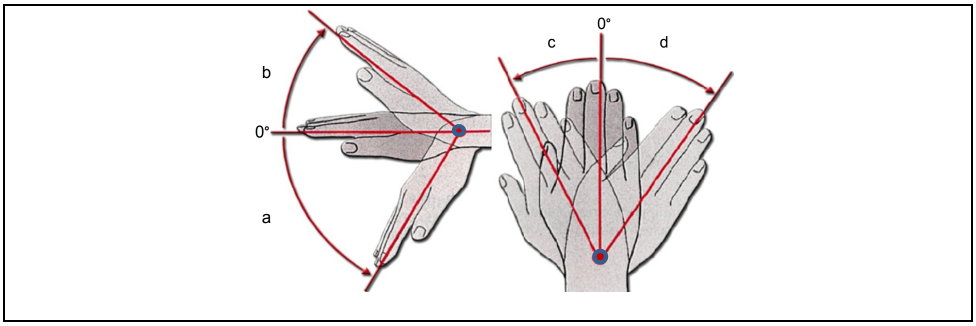


Fig. 4. Human wrist and range of motion: a = palmar flexion 70°, b = dorsal flexion 90°, c = radial abduction 20°, d = ulnar abduction 40° (Whired, 2001).

In order to implement a human-like wrist movement, two orthogonally arranged rotational degrees of freedom are necessary. Both axes are orthogonal to the forearm's axis and intersect in one point. The two degrees of freedom need to be put in a kinematical series.
The requirements and design goals for a humanoid robot's wrist can be deduced based on the range of motion of the human wrist. The first degree of freedom should have a ±30° range of motion and the second about ±90°. The wrist will be attached to the forearm's structure on one side and provides the connection to the hand. It should be possible to disconnect the mechanical joint between the hand and wrist in a simple way in order to enable a modular design. To measure the load on the hand, a 6-axis force and torque sensor must be fitted between the wrist and the hand. The electronic cables and pneumatic tubes supplying power to the hand actuators are the similar to those used in the previous models of ARMAR (Schulz, 2003; Beck et al., 2003). The design space for the robot's wrist is based on human dimensions as far as possible; therefore, one aim is to keep a sphere of approximately 100 mm in diameter as a boundary. At the same time, the control strategy aims to operate all degrees of freedom as individually as possible.
In keeping with the standardized drive concept of most modules of the robot, electronic motors are used as the source for actuation. The drive units need to be dimensioned for a load of 3 kg. All gears are designed to be free from backlash and not self-locking. But friction, e.g. in case of a loss of power, leads to a slow and damped sinking of the arm instead of abrupt movement. That is of great importance for an interactive application of the

robot in a human environment. On the other hand, stick-slip effects in the gears have been avoided, which is a clear benefit for the control system.
Finally, the mechanical structures should be as light as possible in order to save energy during dynamic movements. A lower mass of the wrist can contribute significantly to a reduced energy consumption of the whole arm and has a strong influence on the gears and motors used for the drive units for the elbow and shoulder degrees of freedom.

2.2 Concepts

A simple reduction of the wrist's length by only minor modifications is not possible. This is mainly because the current joint design in combination with the drive unit for the second degree of freedom does not allow a mounting of the hand in the rotational axis. Formulated in an abstract way, the development goal is to shift material from the intersection point to a different location in order to gain free space in the centre position.
Bodies in general have six degrees of freedom in a three dimensional space: three rotational, and three translational. Due to design complexity, the degrees of freedom must be reduced for the development of a technical joint. As technical solutions in robotics usually have only one degree of freedom, it is necessary to combine two basic joints to implement a two degree of freedom joint (Brudniok 2007). An alternative solution is a spherical joint where one rotation is blocked, but actuators for such a design have not yet been sufficiently developed. As result of these basic considerations, two principle solutions were found: a universal joint and a kind of curved track as depicted in figure 5.

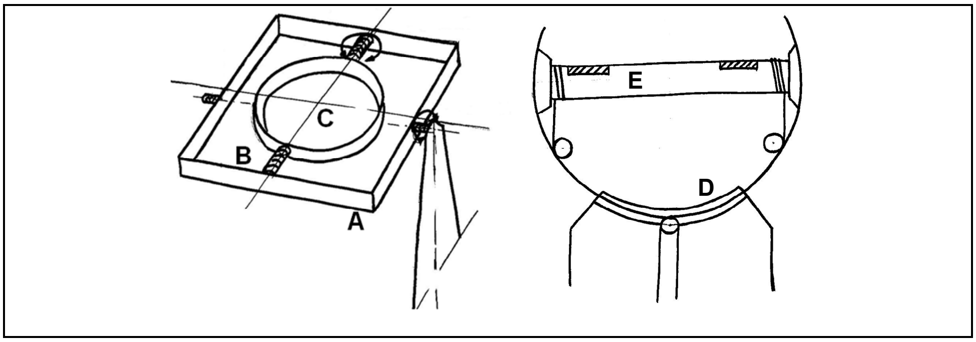


Fig. 5. Universal joint (left) and the principle curved track solution (right).

To illustrate the decision process within the development both concepts are discussed shortly. The universal joint concept (figure 5 left) is very similar to the current solution running on ARMAR III. The first degree of freedom is provided by a rectangular frame (A). On that frame there is enough space for the bearings (B) of the second degree of freedom. Finally, the hand can be mounted on the plate (C). In contrast to the current version, the reduced length was achieved by taking all elements in one plane. The disadvantage is that the outer diameter has to be enlarged in order to provide the wide range of motion described in the previous section. One possible implementation of the drive units could be a direct connection by bowden cables providing a slim and light design of the joint itself. By applying this idea to the universal joint, the total length (TL) of each cable changes. Figure 6 illustrates the parameters which are of importance for a two dimensional consideration.

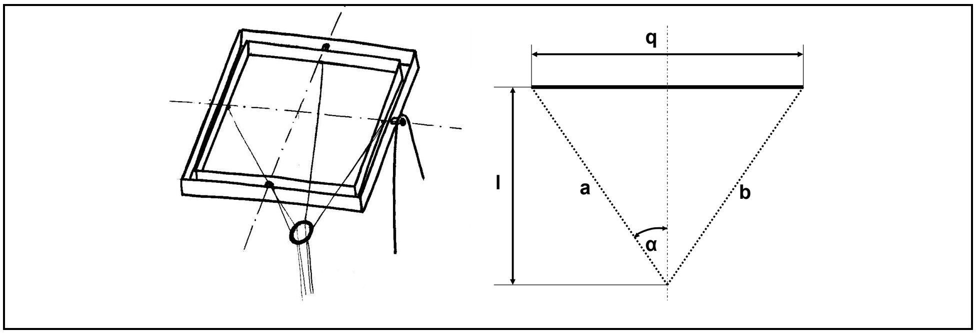


Fig. 6. "Changing" length of the cables in different angular positions of the wrist.

The total lenght can easily be calculated by the following formula, where α denotes the angle between the cables and the middle axis of the forearm:

$$TL = a + b = 2 \cdot \frac{l}{\cos(\alpha)} \tag{1}$$

As α depends on the angular position of the wrist, TL changes during each movement. That means that the different degrees of freedom can not be run independently as long as electronic motors are used as actuators. The Shadow Hand, for example, uses a different concept concerning the cables and their changing lengths (Shadow).

The second basic concept depicted in figure 5 on the right side consists of a curved track solution for the first degree of freedom (D). As this first rotation is limited to ±30°, there is enough space left for the bearings of the second degree of freedom, which may be realized, e.g., by a simple shaft (E). This configuration allows a relatively wide range of motion and a high capability for a reduction of the wrist's length. The challenges for this concept include finding a technical solution for the curved track, a suitable actuation and a design with a proper stiffness in the structures.

Overall, both basic concepts fulfill the principle requirement of length reduction. The curved track method, however, has a clear advantage in terms of size in the radial direction. The oval outer contour also shows a better similarity to a human wrist; therefore, the curved track concept was selected for further development.

2.3 Embodiment Design

By an appropriate design of the shaft (see figure 5 right, named E) it is possible to gain still more space for the 6-axis force and torque sensor. Figure 7 illustrates a cross-section view of the modified shaft. The depth of the shell corresponds with the radius of the curved track and enables a mounting of the hand exactly in the point where the rotational axes intersect. This is achieved by shifting the mechanical connection in the negative direction along the center axis of the forearm.

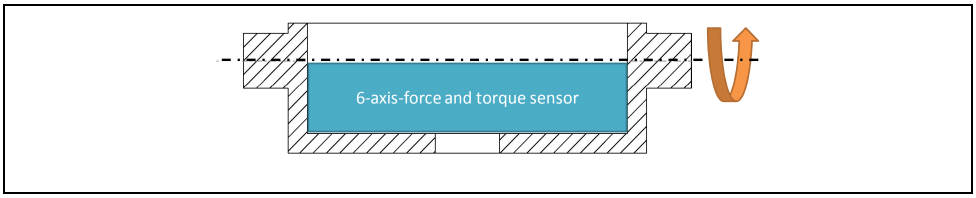


Fig. 7. Basic idea for the shaft of the second DOF of the wrist integrating the force sensor.

For the technical implementation of the curved track, a curved guide named HCR manufactured by THK was selected. Used for medical applications, THK produces ceramic curved guides with a radius of approximately 100 mm. From a technical standpoint it would have been possible to reduce the radius to meet the requirements for a humanoid robot's wrist. For economic reasons, however, this was not a feasible option for the collaborative research centre. Therefore, a different solution was necessary.
The curved guide was replaced by rollers in combination with a timing belt. This allowed the integration of two different functions in one element: the timing belt functions as part of the drive unit while also providing sufficient pre-load to avoid a gap between the rollers and the track. Figure 8 shows the basic CAD model of each design.

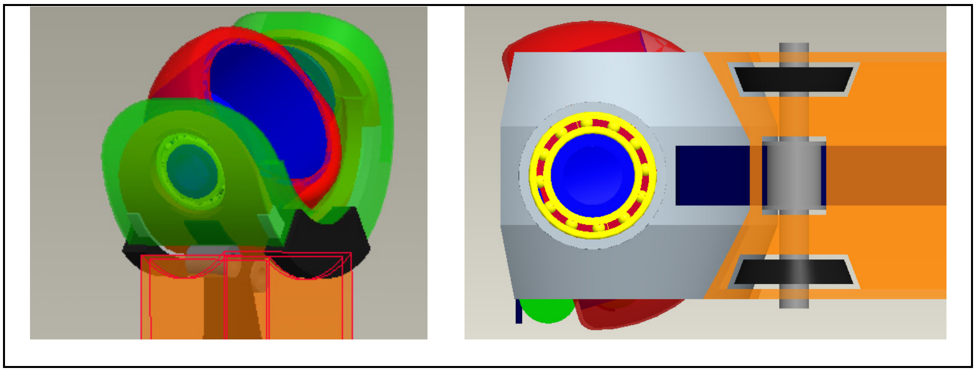


Fig. 8. First technical solution by using a curved guide (left) and the alternative using a roll timing belt combination (right).

3. Simulation

3.1 Basic Geometric Considerations

Based on the new concept an analytic model can be set up. Therefore all geometrical parameters based on the nature of the human body have to be adapted to the model. The undefined variables have to be calculated and estimated using the analytical model to get a reasonable set of values for the design. Using parameter optimization the best combination of values for a design proposal can be found in order to achieve reasonable preloads for the belt.
Two main load cases were used whereas the angle of the initiated force φ can vary. Figure 9 illustrates these two load cases. Here the calculated force F (36 N) is the substitute for all external loads and self-weight (Albers et al., 2006). M is the appropriate torque resulting

from the arm of lever and is about 3.14 Nm. To avoid a displacement of the cap the preload F_V has to be chosen great enough

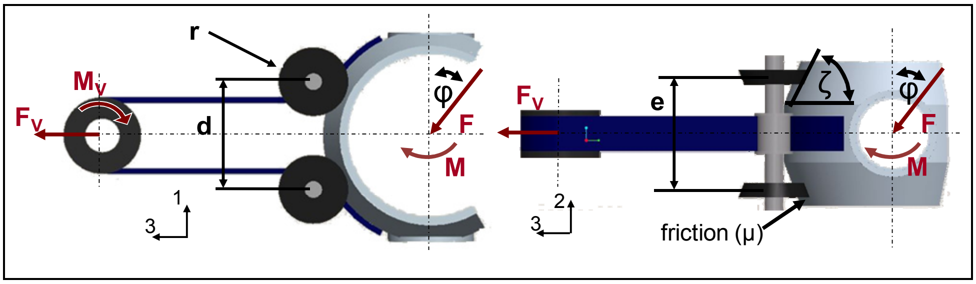


Fig. 9. Load case I (left) and load case II (right) in two different directions.

Load case I:
The external load F is applied in a variable angle φ towards the vertical line. The maximum required preload for an offset of the beveled wheels (d) of 37.5 mm is about 1 kN. When d is increased to 42.5 mm, the required force is less than 0.63 kN. Thus, the required force decreases by about 37 % when the off-set of the beveled wheels is increased by about 21 %. By doubling the distance from 35 mm to 70 mm, the required preload force is reduced by 90 %. The calculated critical angle of the load φ is 36°.

Load case II:
Calculations have shown that the influence of the substituted shear force F is negligible for this load case. Therefore only the over-all torque M is used for the analysis. F_V is dependent upon the angle ς and the wheel distance e. The calculated maximum force for the timing belt is 0.28 kN. The calculated forces are all in a reasonable range compared to the technical elements that can be used for the construction. Consequently the concept can be realized in a physical system with standard bearings and materials.

3.2 First Design and Finite Element Analysis

On the basis of the analytic results described in section 3.1, the optimal solution for the free geometrical parameters can be defined and in a further step be designed in a CAD system. An impression of the parameter optimized wrist is given in figure 10:

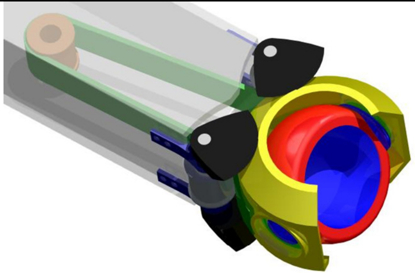


Fig. 10. CAD model based on the results of the analytical considerations.

In a next step the CAD model is simulated numerically using Finite Element Method (FEM) in order to gain further information of the system's behavior. Especially the elasticity of the different structures and the resulting interaction effects are of interest. The preload force and the orientation of the external force were varied systematically. The primary object is to get values for the displacement of the cap towards the global coordinate system. ABAQUS (Dassault Systèmes) is the used solver for the FEM. In order to reduce the computing time, the CAD model must be simplified while the fundamental behavior of the system should be modeled as accurately as possible. The following parts are taken into account for the Finite Element Analysis (FEA): The cap, the beveled wheels, the idler and the timing belt are modeled as deformable with the ABAQUS- element type 'C3D8I' (except beveled wheels, C3D8R). Analytical rigid elements are used for the connecting wheels and the driving shaft. All deformable parts are simulated with isotropic material except the timing belt. Due to the fact that the timing belt is composed of a steel cord with polyurethane backing and teethes, an anisotropic material parameter is used in the model. The angle φ of the external load takes the value of 0° and 36° which is identified in the analytic calculation as the most critical. Figure 11 illustrates the result of the FEA.

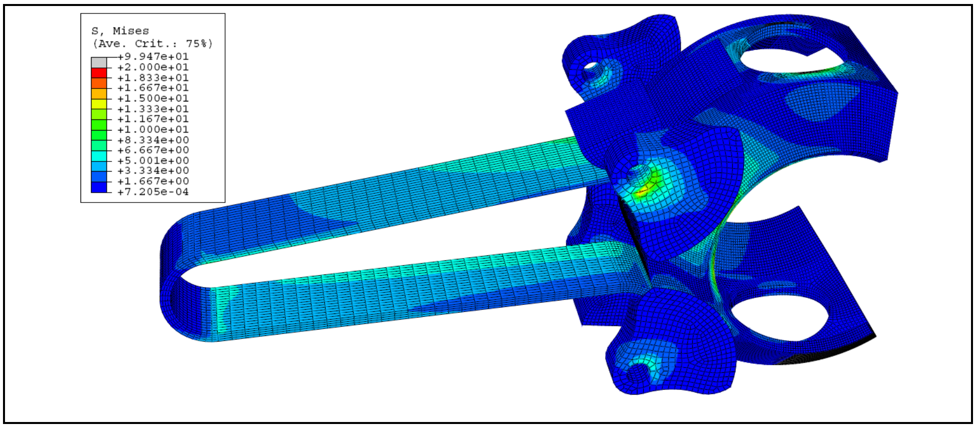


Fig. 11. Stress distribution (von-Mises criterion) for load case II.

The stresses, obtained by the FEA show a reasonable distribution. The displacement of the cap tested with high preload forces is minimal due to the FEA. For a preload of 0.6 kN the displacement for load case II is about $4.4 \cdot 10^{-3}$ mm and for load case I $1.39 \cdot 10^{-2}$ mm. Compared with a preload of 1 kN the displacement doesn't highly decrease. For load case II the displacement takes the value of about $4.07 \cdot 10^{-3}$ mm and for load case II $1.29 \cdot 10^{-2}$ mm. These values for the different preloads show that in the range between F_V=0.6 and 1.0 kN only a small increase of positioning accuracy due to less displacement can be reached. But the high additional costs in the construction of the wrist for preloads higher than 0.6 kN can't be justified. For this reason, and for practical implementation, it is not meaningful to use forces greater than 0.6 kN. For preloads lower than 0.25 kN the position deviation increases dramatically and the system becomes statically indeterminate. The displacement of load case I with φ=36° is for every point smallest compared with load case I (φ=0°) and load case II. Therefore, it appears that a preload between 0.25 kN and 0.6 kN would be most suitable.

4. Functional Prototype

Based on the positive results obtained by the different simulations, a functional prototype was developed. That was necessary mainly because different functions were integrated in the toothed belt, which is usually used in a different manner and not all material parameters were available so that estimated values were used.
As the purpose of the prototype is to prove basic functionality of the design, a few simplifications are made. For the beveled wheels complete rolls are used and the cap is designed in a simple way for instance. Further-more, the construction allows the possibility to implement an s-beam force sensor (Lorenz K-25). Figure 12 shows two pictures of the assembled functional prototype with a one kilogram weight attached at the hand's position.

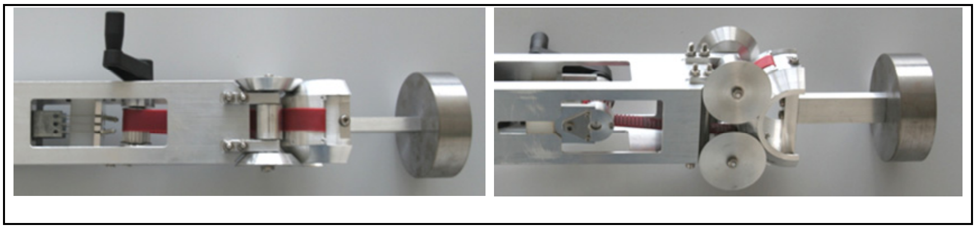


Fig. 12. Functional prototype.

Multiple static and dynamic tests show that this configuration is very accurate and has a high stiffness for small preloads of about 300 N. Hereby the wrist is hand-held at the forearm tube and statically loaded by huge forces between 20-80 N or moved dynamically in all different directions. Even for very fast "hand actuated" motions, which were approximately five times of the maximum velocity of the robot's arm, the assembly remained free from backlash.

5. Optimization and Lightweight Design

As a lightweight design is one of the main goals for the development of the new wrist, different numerical optimization methods were used.

5.1 Topology Optimization

Topology optimization is used for the determination of the basic layout of a new design. It involves the determination of features such as the number, location and shape of holes, and the connectivity of the domain. A new design is determined based upon the design space available, the loads, possible bearings, and materials of which the component is to be composed. Today topology optimization is very well theoretically studied (Bendsoe & Sigmund, 2003) and also a common tool in the industrial design process (Pedersen & Allinger, 2005). The designs, obtained using topology optimization are considered as design proposals. These topology optimized designs can often be rather different compared to designs obtained with a trial and error design process or designs obtained from improvements of existing layouts. The standard formulation in topology optimization is often to minimize the compliance corresponding to maximize the stiffness using a mass constraint for a given amount of material. That means that for a predefined amount of mass the structure with the highest stiffness is determined. Compliance optimization is based upon static structural analyses, modal analyses or even non-linear problems, such as models including contacts. A topology optimization scheme is basically an iterative process that integrates a finite element solver and an optimization module. Based on a design response supplied by the FE solver (e.g. strain energy), the topology optimization module modifies the FE model.

5.2 Material Optimization

Besides the topology optimization, it is necessary in addition to consider optimization strategies such as material optimization. Extreme lightweight design is possible only by combining both optimization strategies such as the topology optimization in combination with an optimal fiber layout. For calculation of laminates by use of the Finite Element Method (FEM), approaches are used that combine the properties of single plies to one virtual material by use of the 'Classical Lamination Theory' (CLT) (Johns, 1999). These established theories are valid for the elastic range.

Several approaches for the determination of optimal fiber orientation have been presented in the past. (Luo & Gea, 1998) use an energy based method. (Setoodeh, 2005) describes an optimality criteria approach, while (Jansson, 2007) works with a generic algorithm. Inspired by nature (Kriechbaum 1994), (Hyer & Charette, 1987) place fibres in direction of first principal stress. In that context (Lederman, 2003) presents a method placing the fibers in the direction of the first main stress in the finite element. (Pedersen, 1991) showed, that a fiber orientation according to the first main strains leads to maximization of stiffness. Most of those approaches only work for one layer, and are reduced on two dimensional problems.

The method used in that work was developed by (Albers et al., 2008b), focusing two main goals: Fast convergence, because the approach is intended to be used together with FEM, and, in a second step, combination with topology optimization. Application should be possible for 3D-geometries, and determination of a two layered laminate structure (orientation and thicknesses) had to be possible to take multi-axial load cases into account. The approach is based on a theory described by (Ledermann, 2003). Optimal fiber orientation is found, if it is equal to the orientation of the first main stress. To be able to take multi-axial load cases into account, the method creates two plies per finite element, with the second ply oriented in the direction of the second main stress. The relation of thickness of the two plies is proportional to the relation of the two main stresses. The orientation of the composite in space is defined by the surface created by the two directions of the main stresses. The third main stress is not taken into account, because 3-dimensional canvases are normally not used in real world applications.

The method is implemented in an iterative procedure, starting with a finite element model with isotropic material. Thenceforward, the isotropic material model is replaced by an anisotropic one with the parameters of a combined two-layer composite. Stress and ply directions are updated in every iteration. In detail, the following steps are undertaken in each iteration: From the preceding finite element analysis, main stress directions and -amounts are determined for each finite element. The procedure starts with the transformation of the direction vectors of main stresses from the element coordinate systems to the vector of the global system by use of the direction cosines. The cross product of the direction vectors of the two first main stresses is used to define the perpendicular to the later surface of lamina of the element. In the special case of the cross product being the zero-vector, e.g. the uniaxial stress condition, a filter is used to determine the perpendicular out of the neighboring elements. By use of the given engineering constants $E_{\|}$, $E_{\perp}$, $\upsilon_{\perp\|}$, $\upsilon_{\perp\perp}$ and $G_{\perp\|}$ of the chosen fiber-matrix-combination, the orthotropic stiffness matrix [*C*] of the UD-layers can be reduced to a transversal isotropic one as follows:

$$[C]=\begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{21} & C_{22} & C_{33} & 0 & 0 & 0 \\ C_{31} & C_{32} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{bmatrix} \tag{2}$$

with

$$\begin{gathered} C_{11}=\frac{1-\upsilon_{23}\upsilon_{32}}{E_2E_3\nabla},\ C_{22}=\frac{1-\upsilon_{13}\upsilon_{31}}{E_1E_3\nabla},\ C_{33}=\frac{1-\upsilon_{12}\upsilon_{21}}{E_{21}E_1\nabla} \\ C_{12}=\frac{\upsilon_{12}+\upsilon_{32}\upsilon_{13}}{E_1E_3\nabla},\ C_{13}=\frac{\upsilon_{13}+\upsilon_{12}\upsilon_{23}}{E_1E_2\nabla},\ C_{23}=\frac{\upsilon_{23}+\upsilon_{21}\upsilon_{13}}{E_1E_2\nabla} \\ C_{44}=C_{23},\ C_{55}=C_{13},\ C_{66}=C_{12} \end{gathered} \tag{3}$$

and

$$\nabla=\frac{1-\upsilon_{12}\upsilon_{21}-\upsilon_{23}\upsilon_{32}-\upsilon_{31}\upsilon_{13}-2\upsilon_{21}\upsilon_{32}\upsilon_{13}}{E_1E_2E_3} \tag{4}$$

Now, the engineering constants mentioned above are introduced:

$$\begin{gathered} E_2=E_3=E_{\perp} \\ G_{31}=G_{21}=G_{\perp\|} \\ \upsilon_{31}=\upsilon_{21}=\upsilon_{\perp\|} \\ G_{\perp\perp}=\frac{G_{\perp\|}}{2(1+\upsilon_{\perp\perp})} \end{gathered} \tag{5}$$

The angle α between the two layers is the angle between the two first main stresses. It can be obtained by use of the direction cosines. The volume share of the two layers is calculated as:

$$\upsilon_1 = \frac{|\sigma_1|}{|\sigma_1| + |\sigma_2|}, \ \upsilon_2 = 1 - \upsilon_1 \tag{6}$$

By use of the 'Classical Lamination Theory' (CLT), the combined stiffness matrix [C_{com}] of the two-layer-lamina can now be calculated. First, the stiffness matrix of the smaller layer is transformed into the lamina coordinate system, defined by the direction of the first main stress. This is done by rotating the stiffness matrix of the layer [C]' about α:

$$[C] = [T]^{-1}[C]'[T]^{-T} \tag{7}$$

With [*T*] the following transformation matrix:

$$[T] = \begin{bmatrix} \cos^2\alpha & \sin^2\alpha & 2\sin\alpha\cos\alpha \\ \sin^2\alpha & \cos^2\alpha & 2\sin\alpha\cos\alpha \\ -\sin\alpha\cos\alpha & \sin\alpha\cos\alpha & \cos^2\alpha - \sin^2\alpha \end{bmatrix} \tag{8}$$

The combination of the two layers is done by use of the rules defined by the CLT. The iteration is finished by formatting and writing the new anisotropic stiffness matrices in the input deck for the FEA. Depending on the FE code used, the materials has to be filtered and clustered before a FEA can be performed, as some FE algorithms are limited in the number of materials allowed. Reduced convergence speed and accuracy of the approach may result.

5.3 Model Setup

For the topology and material optimization the complete system is disassembled and only the cap is used for the optimization. This simplification is necessary to avoid enormous computing time caused by a very fine mesh for the cap and a huge number of load cases. Cutting these parts free from the total system, calls for a realistic replacement of the interaction between the components. Hereby the interaction between the beveled wheels and cap is replaced by a connection at the corresponding nodes which allows a degree of freedom in the 3-axis direction. This simplification is possible because the appearing forces can only be compressive force or the cap lifts off the wheel surface. The timing belt is replaced by a load which is tangential to the cap and transferred to the structure by 5 points on each side of the cap. The preload used in place of the timing belt is 450 N. On one side of the cap an additional force is applied to the timing belt which is the result of an inertial relief and named $F_{inertrel}$. The external load (F) is defined to 30 N and applied to the cap by a torsion arm, which is modeled as rigid element (RBE in MSC.Nastran), in a distance of 100 mm in negative 3-coordinate-axis. The force vector can be reduced to three different directions because of the symmetry conditions can be defined for the optimization process.

The additional torque (M) represents 5 Nm and rotates about the global 3-coodrinate-axis. In figure 13 eight different load cases are illustrated.

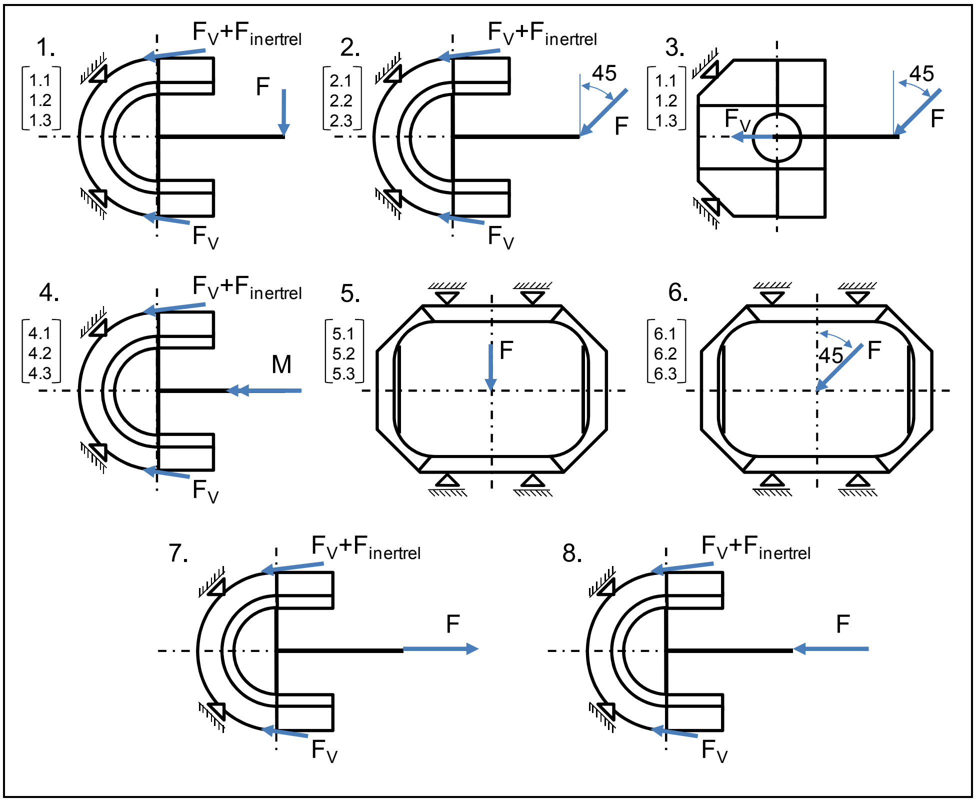


Fig. 13. Load cases for the topology optimization.

The first six load case combinations are set for three different rotational positions of the second DOF. The initial state is the neutral position and two other positions are realized by changing the direction and position of all forces on the cap as they would appear at a ±30° rotation. From this it follows that 18+2=20 different configurations of the cap are set for the topology optimization.

5.4 Results

The result of the topology optimization as a basic design proposal is shown in figure 14. The complex topography in the center is a result of the stress caused by the torsional moment applied to the structure mainly by load case 4 (4.1, 4.2, 4.3). The direct connection to the bearing by the beveled wheels is visible clearly. The function of the center link is mainly to absorb the reaction forces applied to the cap by the high preload and the beveled wheels. The side links are important to reinforce the cap structure between the two points of force transmission of the timing belt. The shape of an arrowhead as a result of the two side links is highlighted on the right side in figure 14.

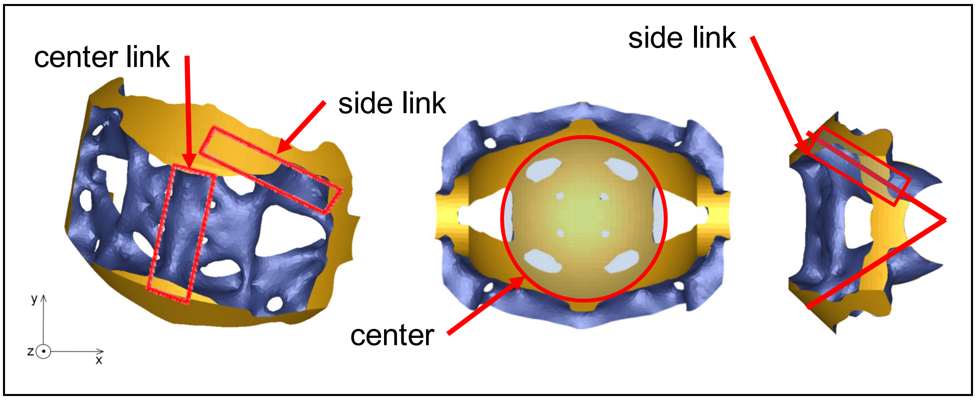


Fig. 14. Smoothed design proposal as a results by the optimization for a lightweight laminate.

In figure 15 the principle stress for an anisotropic topology optimization is illustrated. The highlighted regions are areas with preferred orientated principle stress, which is important while using fiber reinforced materials. Furthermore the stress is uniaxial in these regions. A zoomed view (round clippings) clarifies the stress orientation. In these regions the fibers in the laminate can be orientated in the direction of the principle stress which on the one hand reduces the amount of used material and by that the weight of the cap and on the other hand it improves the stiffness and accuracy. Regions which are not highlighted have changing stress orientations and different stress states. In these areas laminates have to be stacked with different orientations to absorb the multiaxial stress.

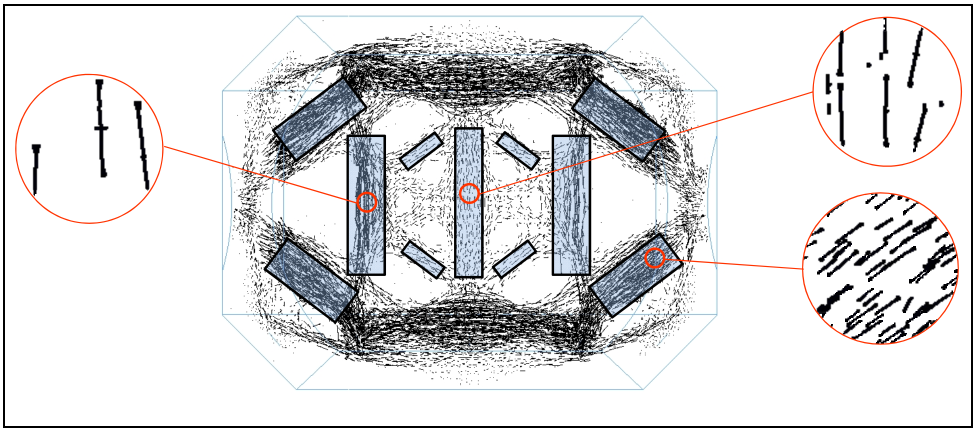


Fig. 15. Principle stress highlighted for a laminate.

Based on the optimization results a 3D-CAD model can be implemented as a first design proposal (see figure 16). In order to realize a lightweight design, the part is built up as an hollow shell. The advantage is that the material is located at the outer area which increases the bending stiffness. To reduce the comprehensive stress at the friction contact zone where the beveled wheels are crawling, a metal band is laminated into the fiber composite. This metal band also increases the stiffness. Furthermore compression proof foam can be integrated into the shell in areas with high pressure mainly introduced to the structure by high preloads. The complex suggestion for the center by the topology optimization is reduced to a thickened center link. To big holes in the structure reduce the weight. In figure 16 on the right side the arrowhead shape, which was suggested by the optimization, is visible. This shape allows a good distribution of forces within the structure.

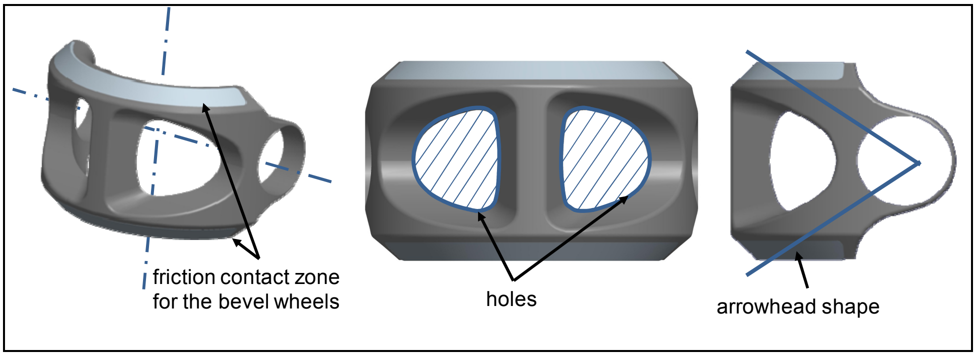


Fig. 16. 3D-CAD model of a design proposal.

Due to the two holes in the structure a new concept for the timing belt is required. Therefore two narrow belts instead of one are used to apply the preload to the cap. In figure 17 a 3D-CAD model with a suggestion for the two timing belts is illustrated. This arrangement increases the support effect and results in a better positioning accuracy.

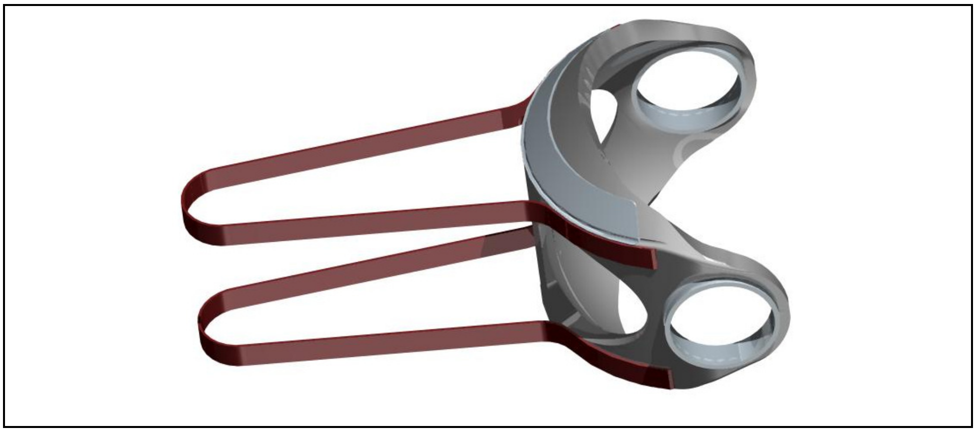


Fig. 17. 3D-CAD model of a design proposal for the timing belt.

6. Conclusion

In this paper the development of a new concept for humanoid robot's lightweight wrist is presented. Especially the different steps of the development process are described. Based on the basic ideas, different analyses and simulations are conducted. A functional prototype is presented which is a kind of proof of the concept. Due to the design proposal obtained by the topology optimization for a fiber composite, a lightweight design is implemented in the CAD model.

The next step will be the integration of the drive units for the second degree of freedom. Here different solutions are possible, e.g. like bowden cables or a direct actuation in combination with harmonic drive gears. In order to achieve a further reduction of mass, composite materials may be used for some further of the structural components. The new wrist will be developed in the next months and is to be manufactured and assembled during the next year.

7. References

Albers A.; Brudniok S.; Ottnad J.; Sauter Ch.; Sedchaicharn K. (2006) Upper Body of a new Humanoid Robot – the Design of Armar III, *Humanoids 06 - 2006 IEEE-RAS International Conference on Humanoid Robots*, December 4 to 6, 2006 in Genova, Italy.

Albers, A.; Deigendesch, T.; Meboldt, M. (2008a). Handling Complexity – A Methodological Approach Comprising Process and Knowledge Management, *Proceedings of the TMCE 2008, 2008 TMCE International Symposium on Tools and Methods of Competitive Engineering*, April 21-25, 2008, Izmir, Turkey.

Albers, A.; Ottnad, J.; Weiler, W. (2008b). Integrated Topology and Fibre Optimization for 3-Dimensional Composites, *Proceedings of IMECE 2008, 2008 ASME International Mechanical Engineering Congress and Exposition*, November 2-6, 2008, Boston, Massachusetts, USA.

Albu-Schäffer, A.; Haddadin, S.; Ott, Ch.; Stemmer, A.; Wimböck, T.; Hirzinger, G. (2007). The DLR lightweight robot: design and control concepts for robots in human environments, *Industrial Robot: An International Journal*, Vol. 34 No. 5, 2007.

Asfour, T. (2003). *Sensomotorische Bewegungskoordination zur Handlungsausführung eines humanoiden Roboters*, Dissertation Fakultät für Informatik, Universität Karlsruhe, 2003.

Beck, S.; Lehmann, A.; Lotz, Th.; Martin, J.; Keppler, R.; Mikut, R. (2003). Model-based adaptive control of a fluidic actuated robotic hand, Proc., *GMA-Congress 2003*, VDI-Berichte 1756, S. 65-72; 2003.

Bendsoe, M. & Sigmund, O. (2003). *Topology Optimization – Theory, Methods, Application*, Springer Verlag 2003.

Brudniok, S. (2007). Dissertation - Methodische Entwicklung hochintegrierter mechatronischer Systeme am Beispiel eines humanoiden Roboters, Forschungsberichte des Instituts für Produktentwicklung, Band 26, Karlsruhe 2007, ISSN 1615-8113.

Hyer, M. W. & Charette, R. F. (1987). *Innovative design of composite structures: use of curvilinear fiber format to improve structural efficiency*, Technical Report 87–5, University of Maryland, College Park, MD, USA, 1987.

Jansson, N. (2007). Optimization of hybrid thermoplastic composite structures using surrogate models and genetic algorithms. Composite Structures, Vol. 80 (2007) No. 1, pp. 21-31.

Jones, R. (1999). *Mechanics of composite materials - 2. ed*, Philadelphia: Taylor & Francis, 1999; ISBN 1-56032-712-X.

Kaneko, K.; Kanehiro, F.; Kajita S.; Hirukawa, H.;Kawasaki, T.; Hirata, M.; Akachi, K.; Isozumi, T. (2004). Humanoid Robot HRP-2, *Proc. IEEE Int. Conference on Robotics and Automation*, pp. 1083-1090, 2004.

Kaneko, K.; Harada, K.; Kanehiro, F.; Miyamori, G.; Akachi, K. (2008). Humanoid Robot HRP-3, *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Acropolis Convention Center Nice, France, Sept, 22-26, 2008.

Kriechbaum, R. (1994). *Ein Verfahren zur Optimierung der Faserverläufe in Verbundwerkstoffen durch Minimierung der Schubspannung nach Vorbildern der Natur*, FZKA 5406, Forschungszentrum Karlsruhe, Eggenstein-Leopoldshafen, Germany, 1994.

Ledermann, M. (2003) Dissertation: *Beiträge zur Optimierung von Faserverbunden nach dem Vorbild der Natur*, Institut für Materialforschung, Forschungszentrum Karlsruhe; Wissenschaftliche Berichte FZKA 6779, ISSN 0947-8620. 2003.

Luo J.H., Gea H.C. (1998) Optimal orientation of orthotropic materials using an energy based method, *Structural Optimization*, Vol. 15 (1998), No. 3/4, pp. 230-236.

Park, I. W.; Kim, J. Y.; Lee, J.; Oh, J. H. (2005). Mechanical Design of Humanoid Robot Platform KHR-3 (KAIST Humanoid Robot – 3: HUBO), *Proc. IEEE-RAS Int. Conference on Humanoid Robots*, pp. 321-326, 2005.

Pedersen, P. (1991). Optimal orientation of anisotropic materials, optimal distribution of anisotropic materials, optimal shape design with anisotropic materials, optimal design for a class of non-linear elasticity, *Optimization of large structural systems; Proceedings of the NATO/DFG Advanced Study Institute* (1991), Berchtesgaden, Germany.

Pedersen, C.B.W. & Allinger, P. (2005) Recent Developments in the Commercial Implementation of Topology Optimization, *TopoptSYMP2005 - IUTAM-Symposium*, Copenhagen, Denmark, pp. 123-132, 2005.

Rosheim, M. (1989). *Robot wrist actuators*, 1. Auflage, 1989 ISBN 0-471-61595-1.

Schulz, S. (2003). *Eine neue Adaptiv-Hand-Prothese auf der Basis flexibler Fluidaktoren*, Dissertation, Fakultät für Maschinenbau, Universität Karlsruhe (TH), 2003.

Schäfer, C. (2000). *Entwurf eines anthropomorphen Roboterarms: Kinematik, Arbeitsraumanalyse, Softwaremodellierung*, Dissertation Fakultät für Informatik, Universität Karlsruhe, 2000.

Setoodeh, S. (2005). Combined topology and fiber path design of composite layers using cellular automata, *Structural and Multidisciplinary Optimization*, Vol. 30 (2005), No. 6, pp. 413-421.

Shadow. www.shadow.org.uk: The Shadow Robot Company.

Wirhed, R. (2001). *Sportanatomie und Bewegungslehre*, Schattauer Verlag, 3. Auflage.

Development of Tendon Based Dexterous Robot Hand

Chung-Hsien Kuo and Chun-Tzu Chen
*Department of Electrical Engineering
National Taiwan University of Science and Technology
Taiwan*

1. Introduction

Dexterous robot hand development is a very challenging and interesting research topic. A dexterous robot hand may serve as a prosthesis hand for disabled patients or to serve as a gripping device for robotic arms. In most of previous studies, the dexterous robot hands are developed based on the directed gear train controls and tendon wired controls. The directed gear train control based design (Lin et al., 1996; Namiki et al., 2003) directly coupled the gear train in the finger module mechanisms. In such a configuration, the weight of the dexterous robot hand is quite heavy because of using numerous gear parts and motors. At the same time, the mechanical design and the assembly of the directed gear train dexterous robot hand are much complicated. Meanwhile, the heats resulted from high reduction of gear trains as well as the high speed rotation of motors are also challenging issues of directed gear train based dexterous robot hands. Consequently, the weights and heats are major concerns when applying this configuration to the prosthesis hand for amputees.
On the other hands, the tendon wire control based dexterous robot hand allocates the gear trains and motors at a distance location (Jacobsen et al., 1986; Kyriakopoulos et al., 1997; Challoo et al, 1994). In this manner, the weights and heats produced from motors and gear trains are resolved when compared to the directed gear train based configuration. Nevertheless, the non-rigid characteristics and frictions of the tendon wires are also important to the precise control of a dexterous robot hands.
In this chapter, a dexterous robot hand with tendon wired control is proposed to perform the characteristics of compact size and low weight. According to this purpose, the dexterous robot hand size is defined as the hand size of a twenties male. In order to reduce the weight of the dexterous robot hand, the ABS engineering plastic material is used in this study. Additionally, to emulate the hand motions of a human, the dexterous robot hand is designed as a five-finger mechanical structure. Consequently, the proposed dexterous robot hand is composed of 16 joint motions. To reduce the control complexity, 12 active joints are independently controlled; and the remaining four joints are manipulated depending on four corresponding active joints.
At the same time, five FSR pressure sensors are attached on the tips of all fingers to detect external forces applied on the corresponding fingers. Therefore, the robot hand is capable of

gripping objects with different griping force setting. In addition to the mechanical design and sensor deployments, the motion trajectories for different griping behaviors (Parka et al., 2006) are also constructed. These motion trajectories are desired to synchronously control the finger joints for achieving various griping behaviors. Consequently, the tactile force sensing (Kawasaki et al., 2007; Kawasaki et al., 2002) may combine with the motion trajectories to perform force feedback based griping control systems. Finally, the proposed dexterous robot hand prototype is developed to demonstrate its motion behaviors.

2. System architecture

In this section, the system architecture is described. The proposed system architecture is composed of the mechanical design, behaviour based gripping motion planning with tactile force sensing, and system integrations, as shown in Fig. 1. The mechanical design specifications include the size and skeleton of robot hand, degree-of-freedom, range of motions of each finger, and orientation of each finger. These specifications are defined according to the hand profile and various gripping postures of a twenties male student in our laboratory. Therefore, the proposed dexterous robot hand may demonstrate similar motions with the human beings. In addition, a tendon wired control configuration is proposed in this study to reduce the weight and heat of the hand. Especially, the ABS engineering plastic material is used to further reduce the weight of the robot hand.

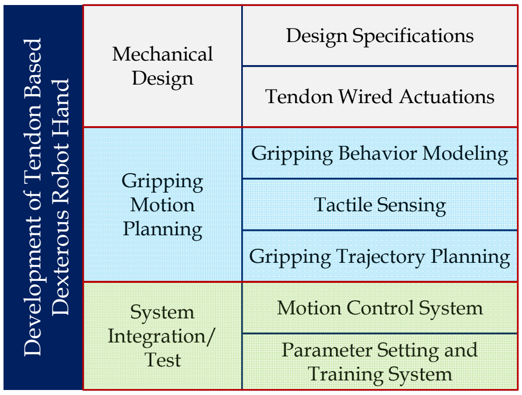


Fig. 1. System development architecture

The gripping motion planning module acts the supervisory controller of the proposed dexterous robot hand. It is responsible of modelling gripping motions in terms of setting the initial and final postures of fingers, acquiring the tactile sensor data from the tips of all fingers, and planning the gripping trajectory based on the gripping models, maximum allowable tactile forces and maximum allowable joint angles. Note that the gripping motion planning module may just model simple griping motions in the current stage. Finally, the system integration and test module is desired to cooperate with the motor controller to manipulate the robot hand. At the same time, the operation parameters such as the maximum tactile force and grip model selection can be also desired.

3. Mechanical design

In this study, a dexterous robot hand is proposed based on the design concepts of light weight, compact size, similar to human's hand gripping motions, gripping force restriction, and low cost. In order to meet these design concepts, the following design issues and directions are discussed before this research project.

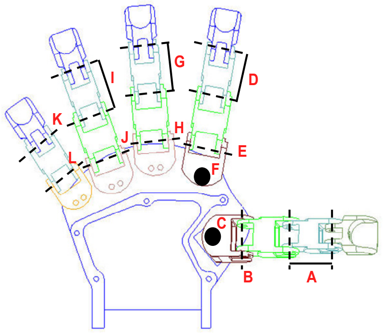
1. Light weight: Most of dexterous robot hands are designed based on directed gear train controls and tendon wired controls. Because of the distance motor and gear train configurations of the tendon wired control robot hand, the weight of the robot hand can be reduced. At the same time, an ABS engineering plastic material is also used to fabricate the hand to further reduce the weight of the robot hand. In this manner, the light robot hand may reduce the loads of the robot arms for service robots as well as the loads of the upper extremity for hand amputees.
2. Compact size: One of the study purposes of this dexterous robot hand is to produce a prosthesis hand for hand amputees preliminarily. Therefore, in addition to the robot hand weight reductions, the hand size and profile must be similar to the human beings. In this study, the proposed dexterous robot hand is designed referring to a twenties male student in our laboratory. Consequently, a five-finger robot hand with 16 degree-of-freedom is presented.
3. Emulating human's hand gripping motions: In addition to similar hand structures, the range of motions of human's hand is also evaluated. The range of angles of the joint is defined via physically measuring the angle of finger joints. In addition, several gripping motions such as gripping apples, eggs and pens are simulated using the computer aided design (CAD) software.
4. Gripping force restriction: Gripping force restriction is important to the robot hand. The maximum force restriction may provide a sufficient gripping force when the finger tip touches the objects, but the force will not damage the gripped objects. At the same time, the maximum allowable tactile force may also protect the wire and motors of the robot hand.
5. Low cost: In general, the cost of a dexterous robot hand is quite expensive because of using high performance DC/ AC servo motors. In recent years, the advance RC (radio servo) techniques provide a simple and low cost position servo control solution. Therefore, the RC servo motors are used in this work to reduce the cost of motors.
6. Other concerns: In addition to the previous considerations, the tendon wired control robot hand may also eliminate the heats resulted from the motors and gear trains when compared to the directed gear train control robot hands. At the same time, the distance actuated robot hand can be applied in more strict environments such as water. Finally, to reduce the control complexity, the joint motions of distal interphalangeal joints of the index finger, middle finger, ring finder and little finger are designed to be dependent on the joint motions of the proximal interphalangeal joints of the corresponding fingers. As a consequence, the motor number is reduced as 12 in this study.

Mechanical design of the proposed dexterous robot hand uses the Pro/E CAD software tool. As described before, the robot hand profile and structure is referred to the hand of a twenties male student in our laboratory. Fig. 2 shows the hand photo of the volunteer. Design parameters of this robot hand follow the hand structure of this hand profile. In order to increase the producing efficiency of mechanical parts, geometries of these mechanical parts are modified and designed as several uniform specifications, as shown in the right-hand-side of Fig. 2. Note that the little finger just designed as a two-phalanx structure because of infrequent uses of this finger as well as reduction in the length of this finger when these uniform mechanical parts are used.

Unit: mm

	Thumb Finger	Index Finger	Middle Finger	Ring Finger	Little Finger
Distal Phalanx	24.1	24.2	24.2	24.2	24.2
Distal Phalanx	23.5	23.5	23.5	23.5	23.5
Distal Phalanx	24.86	24.86	24.86	24.86	N/A

Fig. 2. Hand photo of volunteer and design parameter of robot hand mechanical structure



Unit: degree

Joint Symbol	Range of Angle	Joint Symbol	Range of Angle
A $_{\text{Distal}}$	89	G $_{\text{Distal}}$	75
A $_{\text{Proximal}}$	87	G $_{\text{Proximal}}$	103
B	75	H	93
C	45	I $_{\text{Distal}}$	67
D $_{\text{Distal}}$	75	I $_{\text{Proximal}}$	101
D $_{\text{Proximal}}$	101	J	91
E	90	K	80
F	32	L	93

Fig. 3. Mechanical structure design and ranges of joint angles of the proposed robot hand

In addition to design a similar structure with human being, range of joint motions are also discussed in this study. The ranges of joint motions of fourteen volunteers are evaluated as shown in the right-hand-side of Fig. 3. The joint symbols are referred to the left-hand-side CAD model. These parameters are mean-values measured from fourteen volunteers. Especially, to reduce the control complexity, the joint motions of distal interphalangeal joints and proximal interphalangeal joints of A, D, G and I (referred to Fig. 3) are designed as dependently actuated. The joint angles of distal and proximal joints are correlated in terms of the ranges of joint motions. In practice, different diameters of pulleys and cable wires are used to produce synchronous actuations of distal and proximal joint within the defined angle ranges. Fig. 4 shows the design details.

The mechanical parts of this robot hand are produced in the machining shop of our university. Bearings are used in all rotary parts to reduce the frictions, as shown in the left-hand-side of Fig. 5. In addition, the tactile sensor socket is also desired at the distal phalanx part of each finger, as shown in the right-hand-side of Fig. 5.

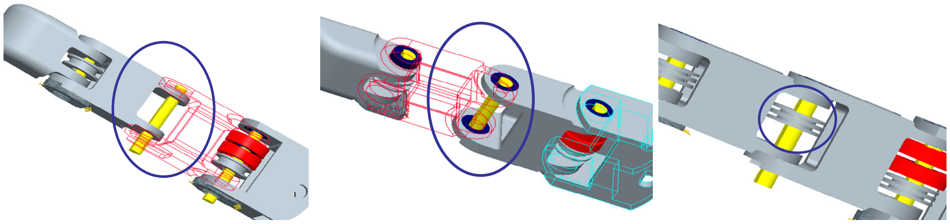


Fig. 4. Design details of a finger CAD model

In order to reduce the weights of mechanical parts, the ABS engineering plastic material is used, and all mechanical parts are shown in Fig. 6.

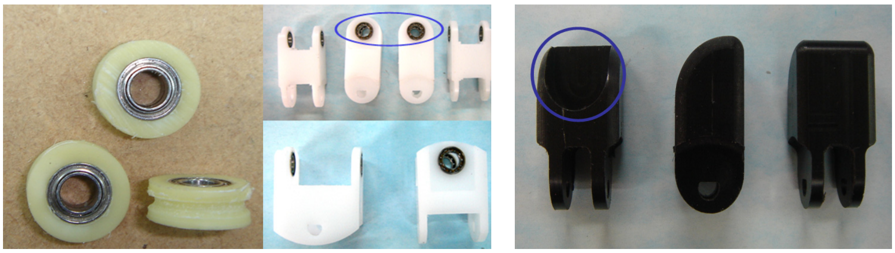


Fig. 5. Parts with bearings and tactile sensor socket

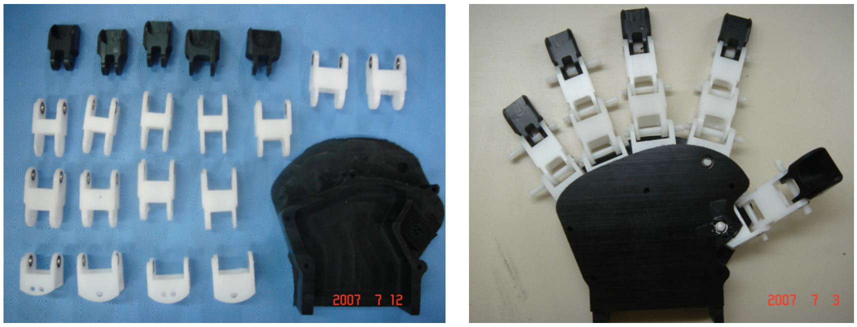


Fig. 6. Photos of produced mechanical parts and assembly of the hand

Because of referring the hand profile of a twenties volunteer, the size of the fingers are evaluated as shown in Fig. 7. In this figure, the index finger of the volunteer are compared with the robot hand. Apparently, they are in a similar finger profile. Finally, because of using the ABS engineering plastic material, the weight of the hand can be reduced. The weight for each finger is summarized in Table 1. To investigate the mechanism performance, several hand postures of human beings are simulated using the 3D CAD tool, as shown in Fig. 8. As a consequence, these hand postures can be properly desired using the proposed mechanical structure of this robot hand.

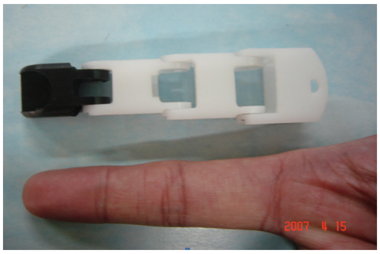


Fig. 7. Finger profile comparisons for a twenties volunteer and the proposed robot hand

Unit: gram

	Thumb Finger	Index Finger	Middle Finger	Ring Finger	Little Finger
Distal Phalanx	4	3.5	3.5	3.5	3.5
Middle Phalanx	4	4	4	4	4
Proximal Phalanx	3.5	4.5	4.5	4	3
Metacarpal Phalanx	N/A	3.5	3.5	3.5	N/A
Weight of a Finger	11.5	15.5	15.5	15	10.5

Table 1. Weights of fingers of the robot hand

Finally, the tendon wired control configuration is introduced. In this study, a multi-cord steel wire with 0.8 mm diameter is used. In addition, the tendon wire is surrounded with a spring cord as shown in Fig. 9. I addition, the assembled tendon wired control robot hand is also presented in Fig. 10. The photo of the wire actuated robot hand is shown in the left-hand-side of Fig. 10; and the photo of the distance motor site is shown in the right-hand-side of Fig. 10. All RC servos are mounted at the motor platform, and it can be installed off the hand and arm so that the load of the arm can be reduced.

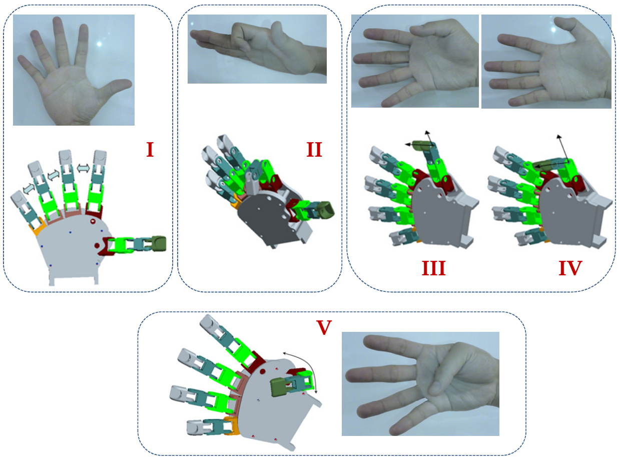


Fig. 8. 3D CAD hand posture simulations

For example, the motor platform can be installed inside the body of a wheeled robot or a humanoid robot to increase the carry loads of the arm and hand. Note that the motor platform is just installed under the robot hand for demonstrations in this chapter.

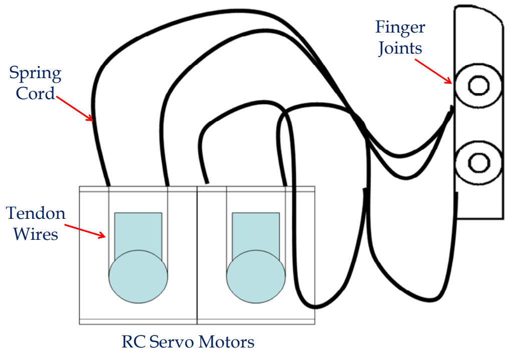


Fig. 9. Tendon wired control configuration

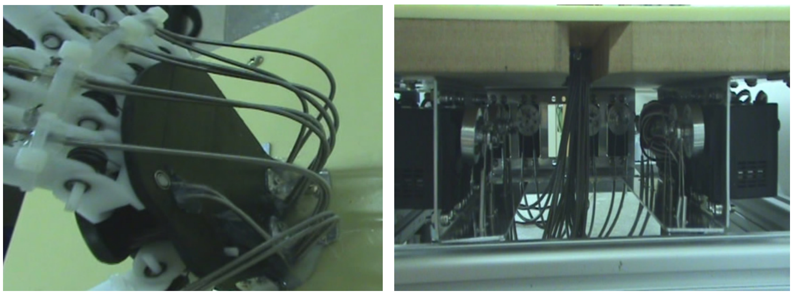


Fig. 10. Photos of wire actuated robot hand (robot hand site and distance motor site)

4. Behavior based gripping motion planning with tactile force sensing

In this study, the developments of a dexterous robot hand do not focus on the mechanical structure design, but also on the gripping behavior modelling. In order to simplify the gripping model as well as to perform more realistic approach, only simple and frequent gripping behaviors are discussed such as gripping an apple, cylinder, egg, etc. It is noted that the precise position controls of all finger tips using inverse kinematics are not the focus of this study due to the difficulties of getting a precise spatial position and orientation of the gripped object in this study. Instead, this study investigates the joint motions from an initial posture to a final posture of a hand for each interested gripping motion. Fig. 11 shows the initial and final postures of a robot hand for gripping an egg and an apple, respectively.
All joint angles of the initial hand and final postures are recorded. The gripping motion can be synchronously desired according to the interpolations of the joint angles of the initial and final postures. Therefore, the gripping model is formed based on the initial and final joint angles as the synchronous interpolations of these joint angles.

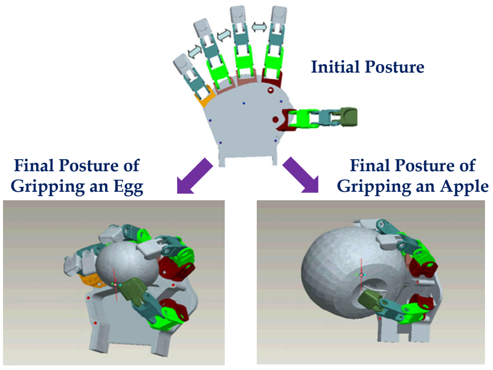

Initial Posture
Final Posture of Gripping an Egg
Final Posture of Gripping an Apple

Fig. 11. Gripping postures simulations of an apple and an egg

Most of initial postures are the same (complete extraction of all fingers, as shown in the top figure of Fig. 11). However, the final postures are quite different for various gripping models. Equation (1) shows the gripping model for a specific gripping motion.

$$n = \frac{T}{\Delta T} \tag{1}$$

$$\vartriangle\theta_k = \frac{\theta_{k,f} - \theta_{k,i}}{n} \tag{2}$$

$$\theta_k(t) = \theta_{k,i} + t\vartriangle\theta_k \tag{3}$$

where T is time required for the gripping motion; ΔT is angular position command time interval for joint motors; n is theoretical count of position commands for a specific gripping motion; $\theta_{k,i}$ is the initial joint angle for joint motor k (k = 1 to 12); $\theta_{k,f}$ is the finial joint angle for joint motor k; $\Delta\ \theta_k$ is the angle increasements of joint motor k; t is the index (from zero) of position commands.

Especially, the gripping model just defines the synchronous joint angle increasements for all joint motors in each operation command with a pre-defined time interval. The actual final (stop) joint angles will not be identical to the pre-defined final posture because it is difficult to get the actual positions and orientations of the gripped object online. That means the real final postures will not refer to the pre-defined final postures; instead, they depend on the maximum allowable tactile sensor force as well as the maximum allowable joint angles to meet practical gripping situations. Hence, the actual operation commands for a whole gripping motion may smaller or greater than the theoretical count of operation commands, as shown in Fig. 12. Consequently, based on the tactile sensor data feedback and maximum allowable joint angle mechanisms, the final (stop) joint angles are determined finger-by-finger.

Note that the tactile sensor used in this study is a conventional force sensing resistor (FSR). On the other hand, the maximum allowable joint angle is desired for the finger being not damage the gripped object during gripping. At the same time, the maximum allowable joint angle may also prevent the collisions and intersections of the hand mechanism for tactile sensor failures and gripping position uncertainties.

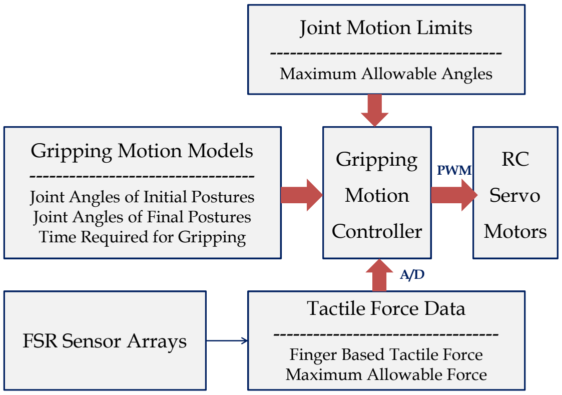


Fig. 12. Behavior-based gripping motions control architecture

The algorithm of the proposed behavior-based gripping motion control system is described in Fig. 13. At the beginning of gripping operations, a gripping model is selected. The angle increasements of al joint angles are calculated according to the angular position (operation) command time interval and the time required for such a gripping motion. The operation command index (*t*) is set as zero at the startup. The joint angled are further updated to get closing to the object. The maximum allowable tactile forces and joint angles are examined finger-by-finger.

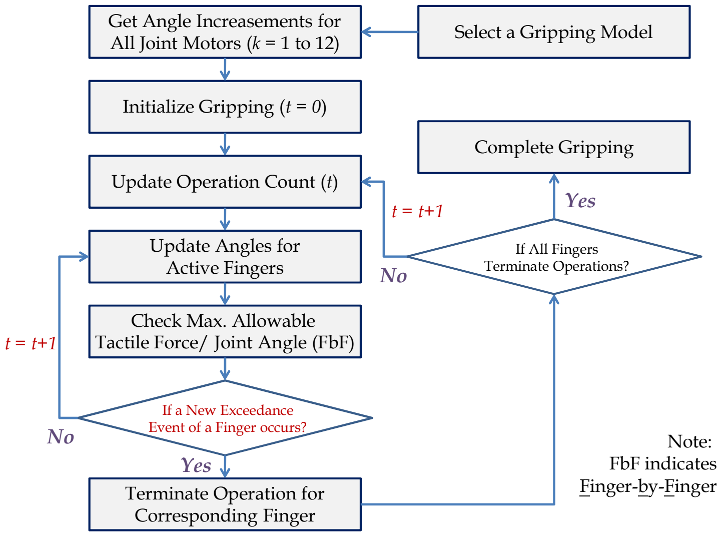

Get Angle Increasements for All Joint Motors (k = 1 to 12)
Select a Gripping Model
Initialize Gripping (t = 0)
Complete Gripping
Update Operation Count (t)
Yes
t = t+1
If All Fingers Terminate Operations?
No
Update Angles for Active Fingers
Check Max. Allowable Tactile Force/ Joint Angle (FbF)
t = t+1
If a New Exceedance Event of a Finger occurs?
No
Yes
Terminate Operation for Corresponding Finger
Note: FbF indicates Finger-by-Finger

Fig. 13. Algorithm of behavior-based gripping motion control system

The active finger is defined as a finger with neither the tactile force being below the maximum allowable force nor the joint angles belong to this finger being within the maximum allowable joint angles. The system is running for all active fingers in terms of updating operation command index (t) as well as calculating new joint angle commands for active fingers. Finally, the gripping motion is completed when no active fingers existed in this system.

5. System integrations and experiments

The system integration and test module is desired to cooperate with the motor controller to manipulate the robot hand via implementing the behavior based gripping motion control algorithm. The control system of the proposed dexterous robot hand is constructed using the PSoC (Programmable System on a Chip) processor. The PSoC is a mixed-signal array microcontroller, and it is a product of Cypress Semiconductor. The core of PSoC is built based on a very-small Harvard architecture machine, named M8C. In addition to the core of firmware, the most important feature is that PSoC provides reconfigurable integrated analog and digital peripherals. Therefore, the PSoC is very suitable to integrate the sensing and actuation devices of the dexterous robot hand. In addition, the analog multiplexer is also desired to extend the analog-to-digital channels of the FSR sensing. At the same time, the gripping models are stored in an EEPROM and the data is accessed via the I^{2}C communications.
In order to justify the tactile force variations among these five FSR sensors, the calibrations of FSR sensor are also desired in this study. At the same time, the operation parameters such as the grip model selection, maximum allowable tactile force for each finger, maximum allowable joint angles, theoretical operation time required for each gripping motion, and operation command time interval are downloaded from a host computer via RS 232 communications. Fig. 14 shows the control circuit boards and the photo of gripping an egg. Ideally, only the fingers of thumb and index and middle fingers are used for gripping. Therefore, the terminations of the thumb, index and middle fingers should be done via checking the maximum allowable tactile forces.

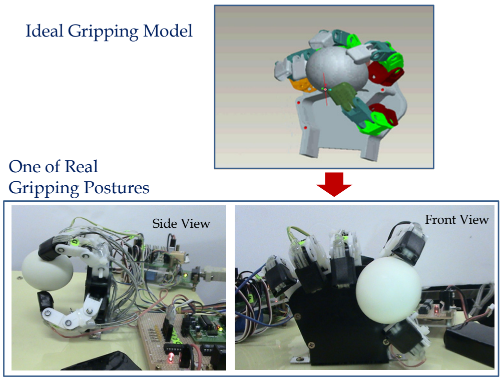


Fig. 14. Egg gripping experiment with inconsistent egg posture (middle finger cannot touch the egg, and this finger is terminated via maximum allowable joint angle of the egg gripping model)

However, due to inconsistent spatial position and posture of the egg, only the thumb and index can touch the tactile sensors in this experiment, and middle finger are terminated via the maximum allowable joint angle for this egg gripping model. Additionally, the terminations of the ring and little finger are also done in terms of the exceedances of their maximum allowable joint angles. It is important that the maximum allowable gripping joint angles of different gripping models must be determined carefully according to practical considerations.
In addition to gripping an object, the proposed dexterous robot hand may perform several hand postures. These hand postures are done via only using the maximum allowable joint angles. Fig. 15 shows several typical hand postures for presenting one-digit numbers, clenched hand, OK symbol, and YA symbol.

6. Conclusions

In this chapter, a dexterous robot hand is proposed based on the design concepts of light weight, compact size, similar to human's hand gripping motions, gripping force restriction, and low cost. The robot hand prototype is produced in this study. Especially, complicated kinematics models and position control of joint angles are not constructed due to unpredictable object orientations and positions in this study. Instead, the gripping motion behavior models for different gripping motions cooperating with maximum allowable tactile forces and joint angles are constructed to simplify the control architecture as well as to improve the practical gripping compatibility. This chapter just represent a pilot study of a low cost dexterous robot hand; however, most of design and control parameters are not well justified to perform perfect performance. On the other hand, the selection of tendon wire as well as the mechanical properties and dynamics of the tendon wire are still the major interests of the future works.

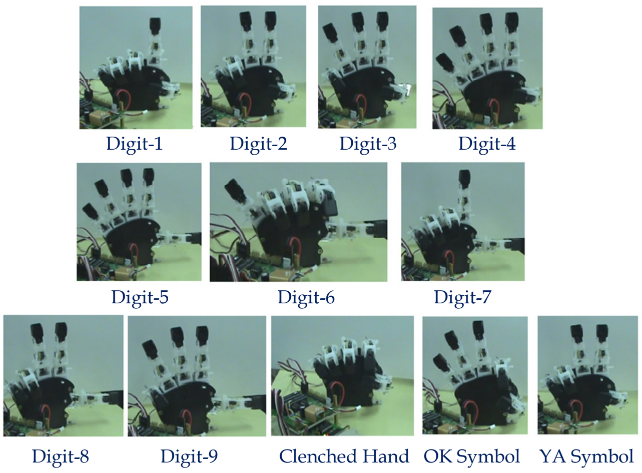


Fig. 15. Several hand posture demonstrations

7. Acknowledgement

This work was partially supported by the National Science Council, Taiwan, R.O.C., under Grants 95-2221-E-011-226-MY3.

8. References

Challoo, R.; Johnson, J.P.; McLauchlan, R.A.; & Omar, S.I. (1994). Intelligent Control of a Stanford JPL Hand Attached to a 4 DOF Robot Arm, *Proceedings of IEEE International Conference on Humans, Information and Technology*, Vol. 2, pp. 1274 – 1278.

Jacobsen, S.; Iversen, E.; Knutti, D.; Johnson, R.; & Biggers, K. (1986). Design of the Utah/M.I.T. Dextrous Hand, *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1520 – 1532, San Francisco, U.S.A.

Kawasaki, H.; & Mouri, T. (2007). Design and Control of Five-Fingered Haptic Interface Opposite to Human Hand. *IEEE Transactions on Robotics,* Vol. 23, No. 5, pp. 909 - 918.

Kawasaki, H.; Komatsu, T.; & Uchiyama, K. (2002). Dexterous Anthropomorphic Robot Hand with Distributed Tactile Sensor: Gifu Hand II. *IEEE/ASME Transactions on Mechatronics,* Vol. 7, No. 3, pp. 296 - 303.

Kyriakopoulos, K.J.; Van Riper, J.; Zink, A.; & Stephanou, H.E. (1997). Kinematic Analysis and Position/Force Control of the Anthrobot Dextrous Hand. *IEEE Transactions on Systems, Man, and Cybernetics, Part B,* Vol. 27, No. 1, pp. 95 - 104.

Lin, L.R; & Huang, H.P. (1996). Integrating Fuzzy Control of the Dexterous National Taiwan University (NTU) Hand. *IEEE/ASME Transactions on Mechatronics,* Vol. 1, No. 3, pp. 216 - 229.

Namiki, A.; Imai, Y.; Ishikawa, M.; & Kaneko, M. (2003). Development of a High-speed Multifingered Hand System and its Application to Catching, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 3, pp. 2666 - 2671.

Parka, N.H.; Oh, Y.; & Oh, S.R. (2006). Behavior-based Control of Robotic Hand by Tactile Servoing. *International Journal of Applied Electromagnetics and Mechanics,* Vol. 24, pp. 311–321.

Dimensional Synthesis and Analysis of the 2-UPS-PU Parallel Manipulator

Yunfeng Zhao, Yanhua Tang and Yongsheng Zhao[1]
School of Mechanical Engineering, Yanshan University
Qinhuangdao 066004 Hebei China

1. Introduction

Compare with conventional serial robots, parallel manipulators has the advantages of higher accuracy, higher stiffness, and higher ratio of force-to-weight, so it has been intensively researched and evaluated by industry and intuitions over the last two decades [1].
It is well known that a main drawback of parallel manipulator is their reduced workspace. Furthermore computing this workspace is not an easy task as, at the opposite of classical serial robot, the translational and orientation workspace are coupled [2]. A number of authors have described the workspace of a parallel mechanism by discretizing the Cartesian workspace [3]. In the case of three degree of freedom (3-DOF) manipulators, the workspace is limited to a region of the three-dimensional Cartesian space.
A more challenging problem is designing a parallel manipulator for a given workspace. Merlet [4] propounded an algorithm to determine all the possible geometries of Gough-type 6-DOF parallel manipulators whose workspace must include a desired one. Boudreau and Gosselin [5] proposed an algorithm that allows for the determination of some parameters of the parallel manipulators using a genetic algorithm method in order to obtain a workspace as close as possible to a prescribed one. Kosinska et al. [6] presented a method for the determination of the parameters of a Delta-4 manipulator, where the prescribed workspace has been given in the form of a set of points. Snyman et al. [7] proposed an algorithm for designing the planar 3-RPR manipulator parameters, for a prescribed two-dimensional physically reachable output workspace. Laribi et al. [8] presented an optimal dimensional synthesis method of the DELTA parallel manipulator for a prescribed workspace. This problem was generally solved numerically, and none of the authors mentioned above took driving force into account.
In this paper, the 2-UPS-PU Parallel manipulator is designed to have a specified workspace. The algorithm is proposed to solve the optimization problem, which not only takes into account the leg-length limits, the mechanical limits on the passive joints, and interference between links, but also the driving forces of the three legs.

[1] Corresponding author. Tel./Fax: +86-335-807-4581.
E-mail address: yszhao@ysu.edu.cn

This paper is organized as follows: Section 2 is devoted to the description of the 2-UPS-PU Parallel manipulator. Section 3 deals with the position analysis of the Parallel manipulator. Section 4 is devoted to the kinematic analysis of the Parallel manipulator. Section 5 deals with the Statics analysis of the Parallel manipulator. In Section 6, we carry out the formulation of the optimization problem. Section 7 contains some conclusions.

2. Displacement analysis

The 2-UPS-PU parallel manipulator is shown in Fig. 1. This manipulator consists of three kinematic chains, including two UPS legs with identical topology and one PU leg, connecting the fixed base to a moving platform. In this parallel manipulator, the UPS legs, from base to platform, consist of a fixed Universal joint, an actuated prismatic joint and a spherical joint attached to the platform. The PU leg connecting the base center to the platform consists of a prismatic joint attached to the base, a universal joint attached to the platform. This branch is used to constrain the motion of the platform to the three degrees of freedom.

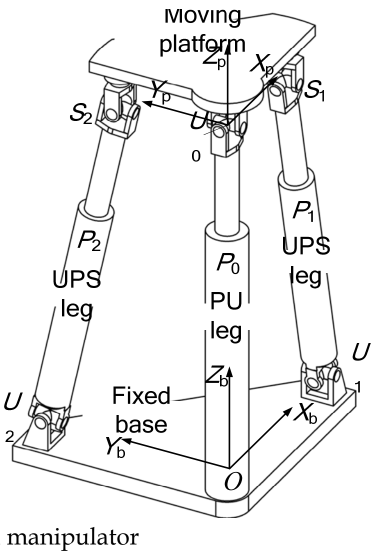


Fig. 1. The 2-UPS-PU parallel manipulator

The reference frame O-$X_bY_bZ_b$ is fixed on the base and mobile frame U_0-$X_pY_pZ_p$ is fixed on the moving platform (see Fig. 1). At the initial position, the X_p-axis and Y_p-axis of mobile frame are coincidence with the axes of the Universal joints U_0 respectively. The orientation of the first axis of U_0 is fixed. The orientation of the mobile frame can be represented by θ_1 and θ_2 shown in Fig. 2, which are two Euler angles about two axes of U_0, respectively. Such Euler angles are defined by first rotating the mobile frame about the base x_p-axis (first axis of U_0) by an angle θ_1, then about the mobile y_p-axis by an angle. For this choice of Euler angles, the rotation matrix is defined as

$$\boldsymbol{R}=\boldsymbol{R}_{X_p}\left(\theta_1\right)\cdot\boldsymbol{R}_{Y_p}\left(\theta_2\right)=\begin{bmatrix}\cos\theta_2 & 0 & \sin\theta_2\\ \sin\theta_1\sin\theta_2 & \cos\theta_1 & -\sin\theta_1\cos\theta_2\\ -\cos\theta_1\sin\theta_2 & \sin\theta_1 & \cos\theta_1\cos\theta_2\end{bmatrix} \tag{1}$$

where $\boldsymbol{R}_{X_p}(\cdot)$ and $\boldsymbol{R}_{Y_p}(\cdot)$ are the basic rotation matrices.

The position of the mobile frame can be represented by l_0, which is the distance between the Universal joint U_0 and reference point O. The homogeneous transform matrix $\boldsymbol{T}$, which represents the orientation and position of the mobile frame, is

$$\boldsymbol{T}=Trans_{Z_p}(l_0)R_{X_p}\left(\theta_1\right)\cdot R_{Y_p}\left(\theta_2\right)=\begin{bmatrix}\cos\theta_2 & 0 & \sin\theta_2 & 0\\ \sin\theta_1\sin\theta_2 & \cos\theta_1 & -\sin\theta_1\cos\theta_2 & 0\\ -\cos\theta_1\sin\theta_2 & \sin\theta_1 & \cos\theta_1\cos\theta_2 & l_0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{2}$$

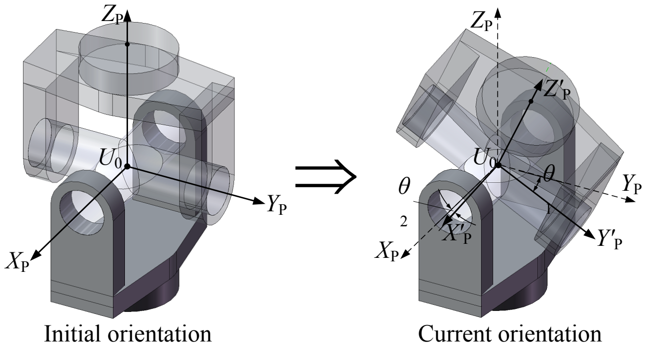


Fig. 2. The 2-UPS-PU parallel manipulator

The spherical joints (S_1 and S_2) of the UPS legs are arranged on the moving platform and their distances to the Universal joint U_0 on the moving platform is r. The Universal joints (U_1 and U_2) are fixed on the base platform and the distances to the reference point O on the base is R. The coordinate of U_0, S_1 and S_2 in mobile frame and the coordinate of U_1 and U_2 in reference frame are expressed as:

$$\boldsymbol{U}_0=\begin{pmatrix}0\\0\\0\end{pmatrix}\quad \boldsymbol{S}_1=\begin{pmatrix}r\cos\alpha\\ r\sin\alpha\\ 0\end{pmatrix}\quad \boldsymbol{S}_2=\begin{pmatrix}r\cos\beta\\ r\sin\beta\\ 0\end{pmatrix}\quad \boldsymbol{U}_1=\begin{pmatrix}R\cos\alpha\\ R\sin\alpha\\ 0\end{pmatrix}\quad \boldsymbol{U}_2=\begin{pmatrix}R\cos\beta\\ R\sin\beta\\ 0\end{pmatrix} \tag{3}$$

2.1 Inverse kinematics

For a given position and orientation of the mobile platform, we can compute the related link lengths, denoted by l_i, using the following relation:

$$\begin{aligned} l_0 &= l_0 \\ l_1 &= \|\boldsymbol{T}\cdot\boldsymbol{S}_1-\boldsymbol{U}_1\| = \sqrt{a^2+b^2+c^2} \\ l_2 &= \|\boldsymbol{T}\cdot\boldsymbol{S}_2-\boldsymbol{U}_2\| = \sqrt{d^2+e^2+f^2} \end{aligned} \tag{4}$$

where

$$\begin{aligned} a &= r\cos\alpha\cos\theta_2 - R\cos\alpha \\ b &= r\cos\alpha\sin\theta_1\sin\theta_2 + r\sin\alpha\cos\theta_1 - R\sin\alpha \\ c &= r\sin\alpha\sin\theta_1 - r\cos\alpha\cos\theta_1\sin\theta_2 + l_0 \\ d &= r\cos\beta\cos\theta_2 - R\cos\beta \\ e &= r\cos\beta\sin\theta_1\sin\theta_2 + r\sin\beta\cos\theta_1 - R\sin\beta \\ f &= r\sin\beta\sin\theta_1 - r\cos\beta\cos\theta_1\sin\theta_2 + l_0 \end{aligned}$$

Eq. (4) is the solution of the so-called inverse kinematics problem.

2.2 Direct kinematics

If set the second axis of Universal joint U_0 pass through either S_1 or S_2 ($\alpha = 0°$ or $\alpha = 90°$), we can simply get analytical direct kinematics solution. For example, let *a*=90°, then cos*a*=1 and sin*a*=0. As a result, Eq. (4) for $\alpha = 90°$ is simplified as:

$$\begin{aligned} l_1^2 &= r^2 + R^2 + l_0^2 + 2rl_0\sin\theta_1 - 2rR\cos\theta_1 \\ l_2^2 &= 2rl_0\sin\beta\sin\theta_1 - 2rR\sin\beta\sin\beta\cos\theta_1 - 2rR\cos\beta\sin\beta\sin\theta_1\sin\theta_2 \\ &\quad - 2rl_0\cos\beta\cos\theta_1\sin\theta_2 - 2rR\cos\beta\cos\beta\cos\theta_2 + r^2 + R^2 + l_0^2 \end{aligned} \tag{5}$$

Then, we can calculate θ_1 and θ_2 by

$$\begin{aligned} \theta_1 &= \sin^{-1}\left(\frac{l_2^2 - r^2 - R^2 - l_0^2}{2r\sqrt{l_0^2+R^2}}\right) + \tan^{-1}\left(\frac{R}{l_0}\right) \\ \theta_2 &= \sin^{-1}\left(\frac{a+b}{\sqrt{c^2+d^2}}\right) - \tan^{-1}\left(\frac{d}{c}\right) \end{aligned} \tag{6}$$

where

$$\begin{aligned} a &= \frac{r^2+R^2+l_0^2-l_1^2}{2r} \\ b &= l_0\sin\beta\sin\theta_1 - R\sin^2\beta\cos\theta_1 \\ c &= l_0\cos\beta\cos\theta_1 + R\cos\beta\sin\beta\sin\theta_1 \\ d &= R\cos^2\beta \end{aligned}$$

It is easy to see that Eq. (6) must satisfy:

$$\begin{aligned} \theta_1 &\in [-90° \quad 90°] \\ \theta_2 &\in [-90° \quad 90°] \end{aligned}$$

Obviously, the same calculation can be drawn when *a*=0°.

3. Velocity equation

To differentiae Eq. (4) allows us to obtain the velocities equation as:

$$\begin{pmatrix} \dot{l}_1 \\ \dot{l}_2 \\ \dot{l}_0 \end{pmatrix} = [\boldsymbol{q}] \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{l}_0 \end{pmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{l}_0 \end{pmatrix} \tag{7}$$

where $[q]$ is the kinematic jacobian matrix, and

$$q_{11} = \frac{l_0 r(\cos\alpha\sin\theta_1\sin\theta_2 + \sin\alpha\cos\theta_1) + Rr\sin\alpha(\sin\alpha\sin\theta_1 - \cos\alpha\cos\theta_1\sin\theta_2)}{l_1}$$

$$q_{12} = \frac{r\cos\alpha[R\cos\alpha\sin\theta_2 - R\sin\alpha\sin\theta_1\cos\theta_2 - l_0\cos\theta_1\cos\theta_2]}{l_1}$$

$$q_{13} = \frac{l_0 + r(\sin\alpha\sin\theta_1 - \cos\alpha\cos\theta_1\sin\theta_2)}{l_1}$$

$$q_{21} = \frac{l_0 r(\sin\beta\cos\theta_1 + \cos\beta\sin\theta_1\sin\theta_2) + Rr\sin\beta(\sin\beta\sin\theta_1 - \cos\beta\cos\theta_1\sin\theta_2)}{l_2}$$

$$q_{22} = \frac{r\cos\beta[R\cos\beta\sin\theta_2 - l_0\cos\theta_1\cos\theta_2 - R\sin\beta\sin\theta_1\cos\theta_2]}{l_2}$$

$$q_{23} = \frac{l_0 + r(\sin\beta\sin\theta_1 - \cos\beta\cos\theta_1\sin\theta_2)}{l_2}$$

4. Statics analysis

The workloads can be simplified as a wrench $\boldsymbol{F}_w$ applied onto moving platform at U_0.

$$\boldsymbol{F}_w = \begin{pmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{pmatrix}^{\mathrm{T}}$$

When ignoring the friction in all the joints and the mass of all the parts, the wrench $\boldsymbol{F}_w$ is balanced by three active forces f_i (i = 1,2, 3), two constrained forces f_x and f_y, and a constrained torque m_z. Each of f_i is applied on and along the axes of three legs; f_x, f_y and m_z are applied on the moving platform about $\boldsymbol{X}_p$, $\boldsymbol{Y}_p$, $\boldsymbol{Z}_p$, respectively. the formulae for solving active forces and constrained forces are derived as

$$\begin{pmatrix} F_x \\ F_y \\ F_z \\ M_x \\ M_y \\ M_z \end{pmatrix} = \begin{bmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 & \boldsymbol{S}_z & \boldsymbol{S}_x & \boldsymbol{S}_y & \boldsymbol{0} \\ \boldsymbol{S}_{01} & \boldsymbol{S}_{02} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{S}_z \end{bmatrix} \cdot \begin{pmatrix} f_1 \\ f_2 \\ f_0 \\ f_x \\ f_y \\ m_z \end{pmatrix} \tag{8}$$

where

$$\boldsymbol{S}_x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}; \quad \boldsymbol{S}_y = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}; \quad \boldsymbol{S}_z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix};$$

$$\boldsymbol{S}_i = \frac{\boldsymbol{T} \cdot \boldsymbol{S}_i - \boldsymbol{U}_i}{l_i}; \quad \boldsymbol{S}_{0i} = \frac{\boldsymbol{S}_i \times (\boldsymbol{T} \cdot \boldsymbol{S}_i - \boldsymbol{U}_i)}{\|\boldsymbol{S}_i \times (\boldsymbol{T} \cdot \boldsymbol{S}_i - \boldsymbol{U}_i)\|} \quad (i = 1, 2)$$

5. Workspace

5.1 Mechanical constraints

There are four main mechanical constraints that limit the workspace of a parallel manipulator [1]: (i) Workspace singularities, (ii) the actuators stroke, (iii) the range of the passive joints, and (iv) the link interference.

5.1.1 Workspace singularities

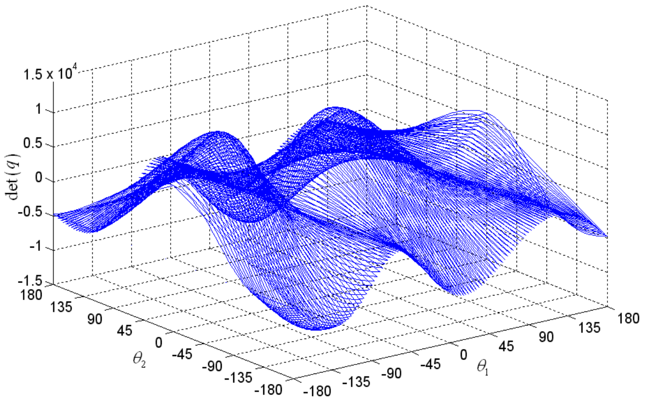


Fig. 3. The relationship between the determinant of jacobian matrix and θ1、θ2

The theoretical workspace of the manipulator is surrounded by singular surface. In the theoretical workspace, the determinant of jacobian matrix ($|\boldsymbol{q}|$) should be always greater or less than zero. The relationship between the determinant of jacobian matrix and parameters of θ_1、θ_2, for r =150, R =200, α=π/2, β=π/6 and l_0=300, is shown in Fig. 3. If we set θ_1=θ_2=0 as the initial oriention, the determinant of jacobian matrix should be always greater than zero.

$$|q| > 0 \tag{9}$$

5.1.2 Actuators' stroke

The limited stroke of actuator i imposes a length constraint on link i, such that

$$\frac{l_{i\max} - l_{i\min}}{l_{i\min}} \le 0.8 \qquad i = 0, 1, 2 \tag{10}$$

where $l_{i\min}$ and $l_{i\max}$ are, respectively, the minimum and maximum lengths of leg *i*.

5.1.3 Range of the passive joints

Each passive joint has a limited range of motion. Let the maximum misalignment angle of the Universal joint U_i be θ_{iumax}, Then, the limits on Universal joint U_i impose a constraint, such that

$$\theta_{iU} \le \theta_{iU\max} \tag{11}$$

Similarly, Let the maximum misalignment angle of the spherical joint S_i be θ_{ismax}, Then, the limits on spherical joint S_i impose a constraint, such that

$$\theta_{iS} \le \theta_{iS\max} \tag{12}$$

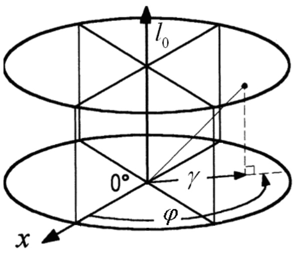
5.1.4 Link interference

Let us assume that the links can be approximated by cylinders of diameter *D*. This imposes a constraint on the relative position of all pairs of links, such that

$$distance\left(l_i \quad l_j\right) \ge D \quad i,j = 0,1,2 \quad i \ne jl \tag{13}$$

or the minimum distance between every two line segments corresponding to the links of the parallel manipulator should be greater than or equal to *D*. The minimum distance between two line segments is not given by a simple formula but can be obtained through the application of a multi-step algorithm. Due to space limitations, we will not present that algorithm here but refer the reader to the well-detailed one given in [9].

5.2 Workspace representation



According to the characteristics of the 2-UPS-PU parallel manipulator, we present a cylindrical coordinate system (see Fig. 4), where φ and γ are exactly the polar coordinates representing the orientation of the moving platform and l_0 is the z-coordinate.
The unit vector of Z_p-axis in reference frame is expressed as:

$$\boldsymbol{Z}_p = \begin{pmatrix} \sin\gamma\cos\varphi \\ \sin\gamma\sin\varphi \\ \cos\gamma \end{pmatrix} = \begin{pmatrix} \sin\theta_2 \\ -\sin\theta_1\cos\theta_2 \\ \cos\theta_1\cos\theta_2 \end{pmatrix} \tag{14}$$

Then, we can calculate θ_1 and θ_2 by

$$\begin{aligned} \theta_2 &= \sin^{-1}(\sin\gamma\cos\varphi) \\ \theta_1 &= -\sin^{-1}\left(\frac{\sin\gamma\sin\varphi}{\cos\theta_2}\right) \end{aligned} \tag{15}$$

5.3 Algorithm for the workspace

Step 1: Initialize double arrays ***A***, ***B*** and ***Z***, with dimensions (n+2)×m, where n is the number of equally spaced planes l_0 between $l_{0\min}$ and $l_{0\max}$ at which the workspace will be computed, and m is the number of points to be computed at each plane l_0=const. These arrays will store, respectively, the values of φ , γ and l_0 for the points defining the workspace boundary.

Step 2: Set r , R , h , α , β and ***D***.

Step 3: Set $\theta_{u\max}$, $\theta_{s\max}$, $l_{i\min}$ and $l_{i\max}$, Where $i = 0, 1, 2$.

Step 4: Set l_0=$l_{0\min}$.

Step 5: For the current l_0, construct a polar coordinate system at (φ, γ). Starting at m equally spaced angles, increment the polar ray, solve the inverse kinematics, and apply the constraint checks defined by Eqs. $(8) \sim (12)$ until a constraint is violated. The values for φ, γ and l_0 at the point of constraint violation are written into the three double arrays.

Step 6: Set $l_0 = l_0 + \Delta l_0$, where $\Delta l_0 = \dfrac{l_{0\max} - l_{0\min}}{n}$

Step 7: Repeat steps 5-6 until l_0 becomes greater than $l_{0\max}$.

Step 8: Transfer ***A*** and ***B*** into ***X*** and ***Y***, so that $X(i,j) = B(i,j)\cdot\cos\left[A(i,j)\right]$ and $Y(i,j) = B(i,j)\cdot\sin\left[A(i,j)\right]$, where $i = 1\cdots n+1$ and $j = 1\cdots m$.

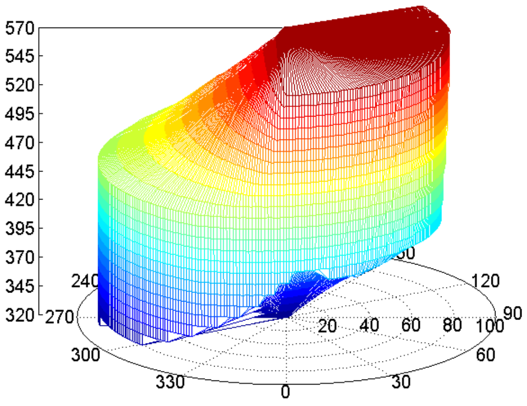


Fig. 5. Isometric view of the Parallel manipulator's workspace.

The proposed algorithm was implemented in MATLAB for the 2-UPS-PU parallel manipulator whose data are $r = 100$, $R = 120$, $\alpha = 90°$, $\beta = 25°$, $D = 5$, $l_{\min} = 320$, $l_{\max} = 578$, $l_{0\min} = 320$, $l_{0\max} = 570$, $\theta_{U\max} = 90°$, and $\theta_{S\max} = 60°$. The workspace of the manipulator is presented in Fig. 5. and the approximated projected workspaces for l_0=350, 450 and 500 shown in Fig. 6.

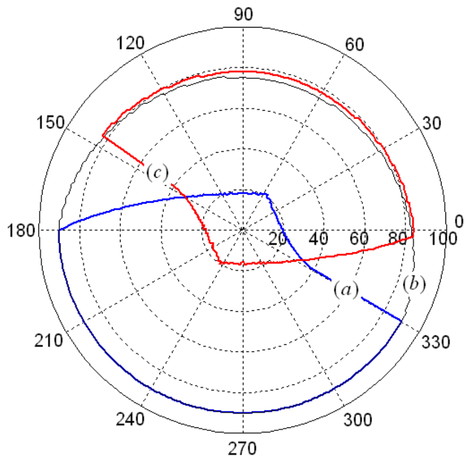


Fig. 6. The projected workspace of the Parallel manipulator for the positions (a) l0=350, (b) l0=450 and (c) l0=500

6. Optimal design

The aim of this section is to develop and to solve the multidimensional optimization problem of selecting the geometric design variables for the 2-UPS-PU parallel manipulator having a prescribed workspace with better driving capability.

The prescribed workspace of the parallel manipulator is defined as a cylinder with radius γ and height h (see Fig.7) and the actual workspace is convex (see Fig.5). The prescribed workspace is inside the workspace of the parallel manipulator if all the points in the boundary curves are inside the actual workspace.

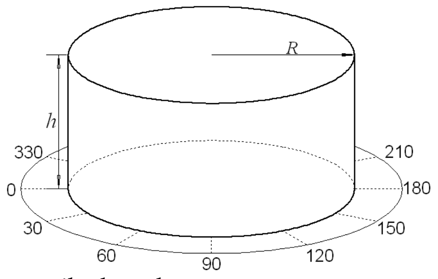


Fig. 7. The scheme of the prescribed workspace.

6.1 Objective function

The parameters to be optimized are the minimum lengths of three legs ($l_{0\min}$, $l_{1\min}$ and $l_{2\min}$), the dimension of the base and the platform (R, r), and the relative angular position of the two UPS chains (α, β). Without losing generality, let r be normalized by h, and R, $l_{0\min}$ be normalized by r such that

$$k_r = \frac{r}{h} \quad k_R = \frac{R}{r} \quad k_{l_0} = \frac{l_{0\min}}{r} \tag{16}$$

The objective function of the multi-parameter optimization problem, can be stated as

$$\begin{array}{ll} \min & k = \pi R^2 l_{0\min} F \\ \multicolumn{2}{l}{\text{Subject to } f(\boldsymbol{I}, p) \le 0 \text{ for all the points } p \text{ inside the specified workspace.}} \end{array} \tag{17}$$

where $\boldsymbol{I}$ is the unknown vector of parameters, and F is the maximum of the driving force of the three legs.

6.2 Algorithm for dimensional synthesis

Step 1: Initialize double arrays $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$, with dimensions 361×1. Set the parameters γ_u and h representing the prescribed workspace.

Step 2: Set $\alpha = 90°$, $k = 0$, and the allowable parameter ranges for β, k_r, k_R and $k_{l_{0\min}}$.

Step 3: Set the cycle number n.

Step 4: Random select the parameters β, k_r, k_R and $k_{l_{0\min}}$ by Monte Carlo method in ranges.

Step 5: Set $l_0 = k \cdot l_{0\min}$ and $\gamma = \gamma_u$. Starting at 360 equally spaced angles, increment the angle φ, solve the inverse kinematics, write the length of the UPS legs into array $\boldsymbol{A}$ and $\boldsymbol{B}$. set $l_{1\min} = l_{1\min} = \min(\min(\boldsymbol{A}), \min(\boldsymbol{B}))$, and $l_{1\max} = l_{1\max} = 1.8 \cdot l_{1\min}$.

Step 6: Set flag=0. For the current l_0, construct a polar coordinate system at (φ, γ). Starting at 360 equally spaced angles, increment the polar ray, solve the inverse kinematics, and apply the constraint checks defined by Eqs. (8)-(12) until $\gamma > \gamma_u$. Set flag=1 at the point of constraint violation and return to Step 12.

Step 7: If flag=0, then set $l_0 = k \cdot l_{0\min} + h$. For the current l_0, construct a polar coordinate system at (φ, γ). Starting at 360 equally spaced angles, increment the polar ray, solve the inverse kinematics, and apply the constraint checks defined by Eqs. (8)-(12) until $\gamma > \gamma_u$. Set flag=1 at the point of constraint violation and return to Step 12.

Step 8: If flag=0, then set $l_0 = k \cdot l_{0\min}$ and $\gamma = \gamma_u$. Starting at 360 equally spaced angles, increment the angle φ, solve the Statics, and write the maximum driving forces of the three legs into array C.

Step 9: If flag=0, then set $l_0 = k \cdot l_{0\min} + h$ and $\gamma = \gamma_u$. Starting at 360 equally spaced angles, increment the angle φ, solve the Statics, and write the maximum driving forces of the three legs into array $\boldsymbol{D}$.

Step 10: Set $F = \max\left(\max(C), \max(D)\right)$, and $k_d = FRl_{0\min}$.

Step 11: If $k = 0$ or $k_d < k$, set $O_r = k_r$, $O_R = k_R$, $O_{l_0} = k_{l_0}$, $O_{l_{\min}} = l_{1\min}$ and $O_{l_{\max}} = l_{1\max}$.

Step 12: Repeat steps 4-11 until the cycle time is n.

The proposed algorithm was again implemented in MATLAB. The optimal solution obtained for this example is presented in Table 1 for $\gamma_u = 60°$ and $h = 100$.

Fig. 8 shows the workspace of the 2-UPS-PU parallel manipulator.

Fig. 9 shows the relationship between the minimum of γ and l_0. One can notice the actual workspace include the prescribed workspace.

β	R	r	$l_{1,2\min}$	$l_{1,2\max}$	$l_{0\min}$	$l_{0\max}$
27°	186	124	396	712	450	650

Table 1. The optimal dimensions of 2-UPS-PU parallel manipulator

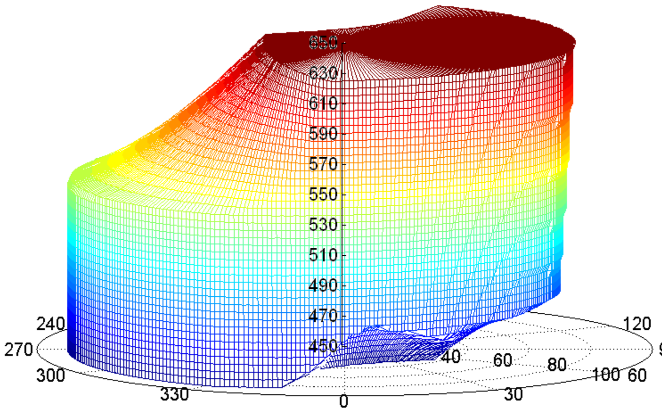


Fig. 8. Isometric view of the Parallel manipulator's workspace.

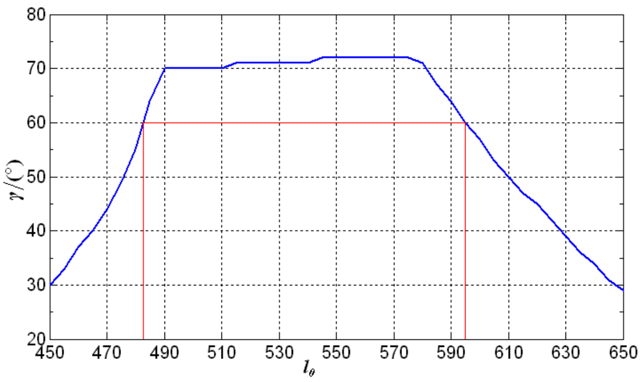


Fig. 9. The relationship between the minimum of γ and l0

7. Conclusion

This paper proposed a novel 2-UPS-PU parallel manipulator with simple structure, high rotational capability, high load carrying capacity.
Based on the cylindrical coordinate system, an algorithm for computing three-dimensional workspace of the manipulator has been proposed in this paper. The boundary of the workspace on the specific plan is found out quickly by step-searching along the selected ray line.
An optimal dimensional synthesis method was presented to optimize this manipulator for a prescribed workspace. The driving force parameters were introduced into the object function. All the dimensional parameters, including the length of the legs, were included into the optimizing algorithm.
The methods proposed in this paper can be adopted universally. The results of this paper provide solid theoretical basis for further theoretical studies and practical application of this manipulator.

8. References

1. Z. Huang, L.F. Kong, Y.F. Fang, Theory on Parallel Robotics and Control[M], Machinery Industry Press, Beijing, China, 1997. (in Chinese)
2. J-P. Merlet. Still a long way to go on the road for parallel mechanisms [R]. ASME DETC Conference, 2002 Montréal.
3. T. Huang, J.S. Wang, D.J. Whitehouse, Closed form solution to the position workspace of Stewart parallel manipulators[J], Science in China, Series E: Technological Sciences 41 (4) (1998) 394–403.
4. Merlet J-P. Designing a parallel manipulator for a specific workspace[J]. International Journal of Robotics Research, 1997, 16(8): p 545-556.
5. R. Boudreau, C.M. Gosselin, The synthesis of planar parallel manipulators with a genetic algorithm[J]. ASME J. Mech. 1999; 121:533–537.
6. A. Kosinska, M. Galicki, K. Kedzior, Design and optimization of parameters of Delta-4 parallel manipulator for a given workspace[J]. J. Robot. Syst. 20 (9) (2003) 539–548.
7. J.A. Snyman, A.M. Hay, Optimal synthesis for a continuous prescribed dexterity interval of 3-DOF parallel planar manipulator for different prescribed output workspaces[C]. In: Proceeding of CK2005, 12th International Workshop on Computational Kinematics Cassino, May 2–6, 2005.
8. M.A. Laribi, L. Romdhane, S. Zeghloul. Analysis and dimensional synthesis of the DELTA robot for a prescribed workspace[J]. Mechanism and Machine Theory 2007, 42:P859-870
9. Masory O, Wang J. Workspace evaluation of Stewart platforms[J]. ASME, Design Engineering Division, 1992, 45:337~345

Direct and Indirect Adaptive Fuzzy Control for a Class of MIMO Nonlinear Systems

Salim Labiod [1] and Thierry Marie Guerra [2]

[1] *LAMEL, Faculty of Science and Technology, University of Jijel, BP. 98, Ouled Aissa, 18000, Jijel*
Algeria

[2] *LAMIH, UMR CNRS 8530, Université de Valenciennes et du Hainaut-Cambrésis, Le Mont Houy, 59313, Valenciennes cedex 9*
France

1. Introduction

During the last two decades, there has been significant progress in the area of adaptive control design of nonlinear systems (Krstic et al., 1995; Sastry & Isidori, 1989; Slotine & Li 1991; Spooner et al., 2002). Most of the developed adaptive control schemes assume that an accurate model of the system is available and the unknown parameters appear linearly with respect to known nonlinear functions. However, this assumption is not sufficient for many practical situations, because it is difficult to precisely describe a nonlinear system by known nonlinear functions and, therefore, the problem of controlling nonlinear systems with incomplete model knowledge remains a challenging task.

As a model free design method, fuzzy control has found extensive applications for complex and ill-defined plants (Passino & Yurkovich, 1998; Wang, 1994). Basically, fuzzy control is a human knowledge-based design methodology which is driven accordingly by fuzzy membership functions and fuzzy rules. However, it is sometimes difficult to find the matched membership functions and fuzzy rules for some plants, or the need may arise to tune the controller parameters if the plant dynamics change. In the hope to overcome this problem, based on the universal approximation theorem and on-line learning ability of fuzzy systems, several stable adaptive fuzzy control schemes have been developed to incorporate the expert knowledge systematically (Spooner & Passino, 1996; Spooner et al., 2002; Su & Stepanenko, 1994; Wang, 1994). The stability analysis in such schemes is performed by using the Lyapunov approach. Conceptually, there are two distinct approaches that have been formulated in the design of a fuzzy adaptive control system: direct and indirect schemes. The direct scheme uses fuzzy systems to approximate unknown ideal controllers (Chang, 2000; Chang, 2001; Labiod & Boucherit, 2003; Li & Tong, 2003; Ordonez & Passino, 1999; Spooner & Passino 1996; Wang, 1994), while the indirect scheme uses fuzzy systems to estimate the plant dynamics and then synthesizes a control law based

on these estimates (Boulkroune et al., 2008a; Boulkroune et al., 2008b; Chang, 2000; Chang, 2001; Chekireb et al., 2003; Chiu, 2005; Golea et al., 2003; Labiod et al., 2005; Ordonez & Passino, 1999; Spooner & Passino 1996; Su & Stepanenko, 1994; Wang, 1994).
For uncertain single-input single-output (SISO) nonlinear systems, fuzzy adaptive control schemes were proposed in (Chang, 2001; Essounbouli & Hamzaoui, 2006; Labiod & Boucherit, 2003; Spooner & Passino, 1996; Su & Stepanenko, 1994; Wang, 1994). The problem of adaptive fuzzy control of uncertain multi-input multi-output (MIMO) nonlinear systems is more difficult because of the coupling that exists between the control inputs and the outputs. This problem was studied in (Boulkroune et al., 2008a; Boulkroune et al., 2008b; Chang, 2000; Chekireb et al., 2003; Chiu, 2005; Golea et al., 2003; Labiod et al., 2005; Li & Tong, 2003; Ordonez & Passino, 1999; Tlemcani et al., 2007 ; Tong et al., 2000; Zhang & YI, 2007). We note that the direct adaptive approach turns out to require more restrictive assumptions than the indirect case, but is perhaps of more interest because it does not present any possible controller singularity problem.
In the aforementioned papers, the adjustable parameters of the fuzzy systems are updated by an adaptive law based on a Lyapunov approach, i.e., the parameter adaptive laws are designed in such a way to ensure the convergence of a Lyapunov function. However, for an effective adaptation, it is more judicious to directly base the parameter adaptation process on the identification error between the unknown function and its adaptive fuzzy approximation (Labiod & Guerra, 2007a; Labiod & Guerra, 2007b).
This chapter presents direct and indirect adaptive fuzzy control schemes for a class of continuous-time uncertain MIMO nonlinear dynamic systems. The proposed schemes are based on the results in (Labiod & Guerra, 2007a). In the direct approach, since fuzzy systems are used to approximate unknown ideal controllers, the adjustable parameters of the used fuzzy systems are updated using a gradient descent algorithm that is designed to minimize the error between the unknown ideal controllers and fuzzy controllers. On the other hand, in the indirect approach, since fuzzy systems are used to approximate the system's unknown nonlinearities, the adjustable parameters of the used fuzzy systems are updated using a gradient descent algorithm that is designed to minimize the error between the system's unknown nonlinearities and the used fuzzy systems. In both approaches, the stability analysis of the closed-loop system is performed using a Lyapunov approach. In particular, it is shown that the tracking errors are uniformly ultimately bounded and converge to a neighbourhood of the origin.
The organization of this chapter is as follows. The problem formulation and fuzzy systems description are given in section 2. The MIMO direct adaptive fuzzy controller with a proof of the stability results are presented in section 3. The MIMO indirect adaptive fuzzy controller with its stability analysis is given in section 4. Section 5 presents simulation results of the proposed direct adaptive control scheme applied to a two-link robot manipulator. Finally, section 6 concludes the chapter.

2. Problem formulation

We consider a class of uncertain MIMO nonlinear systems modeled by

$$\begin{aligned} y_1^{(r_1)} &= f_1(\boldsymbol{x}) + \sum\nolimits_{j=1}^{p} g_{1j}(\boldsymbol{x}) u_j \\ &\vdots \\ y_p^{(r_p)} &= f_p(\boldsymbol{x}) + \sum\nolimits_{j=1}^{p} g_{pj}(\boldsymbol{x}) u_j \end{aligned} \tag{1}$$

where $\boldsymbol{x} = \left[y_1,\ \dot{y}_1,\dots\ ,y_1^{(r_1-1)},\dots,y_p,\ \dot{y}_p,\dots\ ,y_p^{(r_p-1)}\right]^T \in \mathbb{R}^n$ with $n = \sum\nolimits_{i=1}^{p} r_i$, is the overall state vector which is assumed available for measurement, $\boldsymbol{u} = \left[u_1,\dots,u_p\right]^T \in \mathbb{R}^p$ is the control input vector, $\boldsymbol{y} = \left[y_1,\dots,y_p\right]^T \in \mathbb{R}^p$ is the output vector, and $f_i(\boldsymbol{x})$, $g_{ij}(\boldsymbol{x})$, $i,j = 1,\dots,p$ are unknown smooth nonlinear functions.
Let us denote

$$\boldsymbol{y}^{(r)} = \left[y_1^{(r_1)},\dots,y_p^{(r_p)}\right]^T \text{ ; } \boldsymbol{f}(\boldsymbol{x}) = \left[f_1(\boldsymbol{x}),\dots,f_p(\boldsymbol{x})\right]^T \text{ ; } \boldsymbol{G}(\boldsymbol{x}) = \begin{bmatrix} g_{11}(\boldsymbol{x}) & \cdots & g_{1p}(\boldsymbol{x}) \\ \vdots & \ddots & \vdots \\ g_{p1}(\boldsymbol{x}) & \cdots & g_{pp}(\boldsymbol{x}) \end{bmatrix}$$

Then, dynamic system (1) can be written in the following compact form

$$\boldsymbol{y}^{(r)} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} \tag{2}$$

The control objective is to design adaptive control $u_i(t)$ for system (1) such that the output $y_i(t)$ follows a specified desired trajectory $y_{di}(t)$ under boundedness of all signals.
Throughout this study we need the following assumptions.
A1: The matrix $\boldsymbol{G}(\boldsymbol{x})$ is symmetric positive definite and bounded as $0 < \underline{g}\boldsymbol{I}_p \le \boldsymbol{G}(\boldsymbol{x}) \le \overline{g}\boldsymbol{I}_p$, where $\boldsymbol{I}_p$ is the $p \times p$ identity matrix, $\underline{g}$ and $\overline{g}$ are some positive constants.

A2: The desired trajectory $y_{di}(t)$ is a known bounded function of time with bounded known derivatives up to the r_i order.
Remark 1: Notice that Assumption A1 is a sufficient condition ensuring that the matrix $\boldsymbol{G}(\boldsymbol{x})$ is always regular and, therefore, system (1) is feedback linearizable by a static state feedback. Although this assumption restricts the considered class of MIMO nonlinear systems, many physical systems, such as robotic systems (Slotine & Li, 1991), fulfill such a property.
Define the tracking errors as

$$\begin{aligned} e_1(t) &= y_{d1}(t) - y_1(t) \\ &\vdots \\ e_p(t) &= y_{dp}(t) - y_p(t) \end{aligned} \tag{3}$$

and the filtered tracking errors as

$$\begin{aligned} s_1(t) &= \left(\frac{d}{dt}+\lambda_1\right)^{r_1-1} e_1(t) \ , \ \lambda_1 > 0 \\ &\vdots \\ s_p(t) &= \left(\frac{d}{dt}+\lambda_p\right)^{r_p-1} e_p(t) \ , \ \lambda_p > 0 \end{aligned} \tag{4}$$

From (4), $s_i(t)=0$ represents a linear differential equation whose solution implies that the tracking error $e_i(t)$ and its derivatives up to r_i-1 converge to zero (Slotine & Li, 1991). Thus, the control objective becomes the design of a controller to keep $s_i(t)$ at zero, $i=1,\ldots,p$. Moreover, bounds on $s_i(t)$ can be directly translated into bounds on the tracking error. Specifically, if $\left|s_i(t)\right| \le \Phi_i$ where Φ_i is a positive constant, one can conclude that (Slotine & Li, 1991): $\left|e_i^{(j)}(t)\right| \le 2^j \lambda_i^{j-r_i+1}\Phi_i$, $j=0,\ldots,r_i-1$, $i=1,\ldots,p$. These bounds can be reduced by increasing the parameters λ_i.
The time derivatives of the filtered errors (4) can be written as

$$\begin{aligned} \dot{s}_1 &= v_1 - f_1(\boldsymbol{x}) - \textstyle\sum_{j=1}^{p} g_{1j}(\boldsymbol{x}) u_j \\ &\vdots \\ \dot{s}_p &= v_p - f_p(\boldsymbol{x}) - \textstyle\sum_{j=1}^{p} g_{pj}(\boldsymbol{x}) u_j \end{aligned} \tag{5}$$

where $v_1,\ldots,v_p$, are given as follows

$$\begin{aligned} v_1 &= y_{d1}^{(r_1)} + \beta_{1,r_1-1} e_1^{(r_1-1)} + \ldots + \beta_{1,1}\dot{e}_1 \\ &\vdots \\ v_p &= y_{dp}^{(r_p)} + \beta_{p,r_p-1} e_p^{(r_p-1)} + \ldots + \beta_{p,1}\dot{e}_p \end{aligned} \tag{6}$$

with

$$\beta_{i,j} = C_{r_i-1}^{j-1} \lambda_i^{r_i-j} \ , \ i=1,\ldots p, \ \ j=1,\ldots,r_i-1$$

Denote

$$\boldsymbol{s} = \left[s_1,\ldots,s_p\right]^T ; \ \boldsymbol{v} = \left[v_1,\ldots,v_p\right]^T$$

Then equation (5) can be written in matrix form as

$$\dot{\boldsymbol{s}} = \boldsymbol{v} - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} \tag{7}$$

If the nonlinear functions $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{G}(\boldsymbol{x})$ are known, to achieve the control objectives, one can use the following ideal nonlinear control law (Labiod & Guerra, 2007b)

$$\boldsymbol{u}^* = \boldsymbol{G}^{-1}(\boldsymbol{x})\left(-\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{v} + \boldsymbol{K}\,\boldsymbol{s} + \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0)\right) \tag{8}$$

where, $\boldsymbol{K} = diag\left[k_1,\ldots,k_p\right]$, $\boldsymbol{K}_0 = diag\left[k_{01},\ldots,k_{0p}\right]$, with $k_i > 0$ and $k_{0i} > 0$, for $i = 1,\ldots,p$, ε_0 is a small positive constant, and $\tanh(\cdot)$ is the hyperbolic tangent function defined for the vector $\boldsymbol{s} = \left[s_1,\ldots,s_p\right]^T$ as $\tanh(\boldsymbol{s}/\varepsilon_0) = \left[\tanh(s_1/\varepsilon_0),\ldots,\tanh(s_p/\varepsilon_0)\right]^T$.
Effectively, when we select the control input as $\boldsymbol{u} = \boldsymbol{u}^*$, equation (7) simplifies to

$$\dot{\boldsymbol{s}} = -\boldsymbol{K}\,\boldsymbol{s} - \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0) \tag{9}$$

or, equivalently

$$\dot{s}_i = -k_i s_i - k_{0i} \tanh(s_i/\varepsilon_0), \quad i = 1,\ldots,p \tag{10}$$

From which we can conclude that $s_i(t) \to 0$ as $t \to \infty$ and, therefore, $e_i(t)$ and all its derivatives up to $r_i - 1$ converge to zero.
It is clear that if $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{G}(\boldsymbol{x})$ are completely unknown, the proposed nonlinear control law (8) is not feasible. In this case, in order to overcome this design difficulty, we propose to use fuzzy systems to construct adaptively the unknown functions. The idea is to use fuzzy systems to identity the entire unknown control function (8) in the direct approach, and to identify the unknown nonlinear functions $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{G}(\boldsymbol{x})$ in (8) in the indirect approach.
In this chapter we use the zero-order Takagi-Sugeno fuzzy system that performs a mapping from an input vector $\boldsymbol{z} = [z_1,\ldots,z_m]^T \in \Omega_z \subset \mathbb{R}^m$ to a scalar output variable $y_f \in \mathbb{R}$, where $\Omega_z = \Omega_{z_1} \times \cdots \times \Omega_{z_m}$ and $\Omega_{z_i} \subset \mathbb{R}$. If we define M_i fuzzy sets F_i^j, $j = 1,\ldots,M_i$, for each input z_i, then the fuzzy system will be characterized by a set of if-then rules of the form (Wang, 1994; Jang & Sun, 1995; Passino & Yurkovich, 1998)

$$\mathrm{R}^k : \mathrm{If}\ z_1\ \mathrm{is}\ G_1^k\ \mathrm{and}\ldots\mathrm{and}\ z_m\ \mathrm{is}\ G_m^k\ \mathrm{Then}\ y_f\ \mathrm{is}\ y_f^k \quad (k = 1,\ldots,N)$$

where $G_i^k \in \left\{F_i^1,\ldots,F_i^{M_i}\right\}$, $i = 1,\ldots,n$, y_f^k is the crisp output of the k-th rule, and N is the total number of rules.
By using the singleton fuzzifier and the product inference engine, the final output of the fuzzy system is given as follows (Wang, 1994; Jang & Sun, 1995; Passino & Yurkovich, 1998)

$$y_f(z)=\frac{\sum_{k=1}^{N}\mu_k(z)y_f^k}{\sum_{k=1}^{N}\mu_k(z)} \tag{11}$$

where

$$\mu_k(z)=\prod_{i=1}^{m}\mu_{G_i^k}(z_i)\text{, with } \mu_{G_i^k}\in\left\{\mu_{F_i^1},\ldots,\mu_{F_i^{M_i}}\right\}$$

where $\mu_{F_i^j}(x_i)$ is the membership function of the fuzzy set F_i^j.

By introducing the concept of fuzzy basis functions (Wang, 1994), the output given by (11) can be rewritten in the following compact form

$$y_f(z)=\boldsymbol{w}^T(z)\theta \tag{12}$$

where $\theta=\left[y_f^1,\ldots,y_f^N\right]^T$ is a vector grouping all consequent parameters, and $\boldsymbol{w}(z)=\left[w_1(z),\ldots,w_N(z)\right]^T$ is a set of fuzzy basis functions defined as

$$w_k(z)=\frac{\mu_k(z)}{\sum_{j=1}^{N}\mu_j(z)}\ ,\ k=1,\ \ldots,N \tag{13}$$

The fuzzy system (12) is assumed to be well-defined so that $\sum_{j=1}^{N}\mu_j(z)\neq 0$ for all $z\in\Omega_z$.

It has been proved in (Wang, 1994) that fuzzy systems in the form of (12) with Gaussian membership functions can approximate continuous functions over a compact set to an arbitrary degree of accuracy provided that enough number of rules are considered. So, for a general smooth nonlinear function $f(z)$ defined from $\mathbb{R}^n$ to $\mathbb{R}$, there exists a fuzzy system in the form of (12) with some optimal parameters θ^* such that

$$\sup_{z\in\Omega_z}\left|f(z)-\boldsymbol{w}^T(z)\theta^*\right|\leq\bar{\varepsilon}\ ),$$

where $\bar{\varepsilon}$ is a positive constant. Thus, one can express $f(z)$ as

$$f(z)=\boldsymbol{w}^T(z)\theta^*+\varepsilon(z),$$

where $\varepsilon(z)$ is the fuzzy approximation error satisfying $\left|\varepsilon(z)\right|\leq\bar{\varepsilon}$ for $z\in\Omega_z$.

In this chapter, it is assumed that the structure of the fuzzy system and the fuzzy basis function parameters are properly specified in advance by the designer. This means that the designer decision is needed to determine the structure of the fuzzy system (that is, determine relevant inputs, number of membership functions for each input, membership

function parameters, number of rules), and the consequent parameters should be calculated by learning algorithms.
It should be noticed that fuzzy systems can be replaced by any other linearly parameterized universal function approximator without any technical difficulty such as neural networks and wavelet networks. However, only fuzzy logic systems can make use of linguistic information in a systematic way.

3. Direct adaptive fuzzy control

In section 2 we have established that there exists an ideal control law $\boldsymbol{u}^*$ given by (8) that can achieve the control objective. However, this nonlinear controller cannot be used since it depends on unknown functions. In this section, to circumvent this problem, we propose to use adaptive fuzzy systems for approximating this ideal controller, and the error between the fuzzy controller and the ideal controller will be used to update the free parameters of the fuzzy controller.
To develop the control law, we represent each component of the ideal input control vector $\boldsymbol{u}^* = \left[u_1^*,\ldots,u_p^*\right]$ by a fuzzy system in the form of (12) as the following

$$u_i^*(z) = \boldsymbol{w}_i^T(z)\theta_i^* + \varepsilon_i(z); \quad i = 1,\ldots,p \tag{14}$$

where $\boldsymbol{z} = \left[\boldsymbol{x}^T, \boldsymbol{s}^T\right]^T$, $\varepsilon_i(z)$ is the fuzzy approximation error, θ_i^* is an unknown ideal parameter vector that minimizes the function $\left|\varepsilon_i(z)\right|$ over an operating compact set Ω_z, and $\boldsymbol{w}_i(z)$ is a fuzzy basis function vector assumed suitably specified by the designer. In this study, we assume that the used fuzzy systems do not violate the universal approximation property on the compact set Ω_z, which is assumed large enough so that the variable z remains inside it under closed-loop control. So it is reasonable to assume that the fuzzy approximation error is bounded for all $z \in \Omega_z$.
Let us denote

$$\varepsilon(z) = \left[\varepsilon_1(z),\ldots,\varepsilon_p(z)\right]^T; \ \theta^* = \left[\theta_1^{*T},\ldots,\theta_p^{*T}\right]^T, \ \boldsymbol{w}(z) = diag\left[\boldsymbol{w}_1(z),\ldots,\boldsymbol{w}_p(z)\right]$$

Therefore, one can write (8) as

$$\boldsymbol{u}^* = \boldsymbol{w}^T(z)\theta^* + \varepsilon(z) \tag{15}$$

Since the ideal parameter vector θ^* is unknown, let us use its estimate θ instead to form the adaptive control

$$\boldsymbol{u}(z) = \boldsymbol{w}^T(z)\theta \tag{16}$$

The next step should be the design of an adaptive law for the free parameters θ such that the control law $\boldsymbol{u}$ approximates, as best as possible, the ideal controller $\boldsymbol{u}^*$. To this end, let us define the error between the controllers $\boldsymbol{u}^*$ and $\boldsymbol{u}$ as:

$$\boldsymbol{e}_u = \boldsymbol{u}^* - \boldsymbol{u} \tag{17}$$

The error $\boldsymbol{e}_u$ represents the actual deviation between the unknown function $\boldsymbol{u}^*$ and the online fuzzy approximator (16), while the fuzzy approximation error $\varepsilon(z)$ represents the minimum possible deviation between the unknown function $\boldsymbol{u}^*$ and the online fuzzy approximator, i.e. $\varepsilon(z)$ represents the minimum possible value of $\boldsymbol{e}_u$.
Using (15) and (16), (17) becomes

$$\boldsymbol{e}_u = \boldsymbol{u}^* - \boldsymbol{w}^T(z)\theta = \boldsymbol{w}^T(z)\tilde{\theta} + \varepsilon(z) \tag{18}$$

where $\tilde{\theta} = \theta^* - \theta$ is the parameter estimation error vector.
Adding and subtracting $\boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}^*$ to the right-hand side of (7), we obtain the error equation governing the closed-loop system

$$\dot{\boldsymbol{s}} = \boldsymbol{v} - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}^* - \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}^* \tag{19}$$

With (8) and (18), (19) becomes

$$\dot{\boldsymbol{s}} = -\boldsymbol{K}\boldsymbol{s} - \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u \tag{20}$$

Now, consider a quadratic cost function; that measures the discrepancy between the ideal controller and the actual fuzzy controller, defined as

$$J(\theta) = \frac{1}{2}\boldsymbol{e}_u^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u = \frac{1}{2}\left(\boldsymbol{u}^* - \boldsymbol{w}^T(z)\theta\right)^T \boldsymbol{G}(\boldsymbol{x})\left(\boldsymbol{u}^* - \boldsymbol{w}^T(z)\theta\right) \tag{21}$$

We use the gradient descent method to minimize the cost function (21) with respect to the adjustable parameters θ. Consequently, applying the gradient method (Slotine & Li, 1991; Ioannou & Sun, 1996), the minimizing trajectory $\theta(t)$ is generated by the following differential equation

$$\dot{\theta} = -\eta\nabla_\theta J(\theta) \tag{22}$$

where η is a positive constant parameter.
From (21), the gradient of $J(\theta)$ with respect to θ is

$$\frac{\partial J(\theta)}{\partial \theta} = -\boldsymbol{w}(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u$$

Therefore, the gradient descent algorithm becomes

$$\dot{\theta} = \eta\, \boldsymbol{w}(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u \tag{23}$$

We recall here that the ideal controller $\boldsymbol{u}^*$ is unknown, so the error signal $\boldsymbol{e}_u$ defined in (17) is not available. Equation (20) will be used to overcome this design difficulty. Indeed, from (20), we see that even if the error vector $\boldsymbol{e}_u$ is not available, the vector $\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u$ is available, and it is given by

$$\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u = \dot{\boldsymbol{s}} + \boldsymbol{K}\,\boldsymbol{s} + \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0)$$

Therefore, (23) becomes

$$\dot{\theta} = \eta\, \boldsymbol{w}(\boldsymbol{z})\left\{\dot{\boldsymbol{s}} + \boldsymbol{K}\,\boldsymbol{s} + \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0)\right\} \tag{24}$$

As shown by (Ioannou & Sun, 1996), an adaptive law in the form of (24) cannot guarantee the boundedness of the parameters $\tilde{\theta}$ in the presence of approximation errors, which are unavoidable in such adaptive schemes. So, to improve the robustness of the adaptive law (24) in the presence of approximation errors, we modify it by introducing a σ-modification term as follows (Ioannou & Sun, 1996)

$$\dot{\theta} = \eta\, \boldsymbol{w}(\boldsymbol{z})\left\{\dot{\boldsymbol{s}} + \boldsymbol{K}\,\boldsymbol{s} + \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0)\right\} - \eta\sigma\theta \tag{25}$$

where σ is a small positive constant.
The following theorem summarizes the stability result for the proposed direct adaptive control scheme.

Theorem 1: *Consider the system in (1) with the control law defined by (17). Suppose that Assumptions A1 and A2 hold, the approximation error $\varepsilon(\boldsymbol{z})$ in (18) is bounded as $\|\varepsilon(\boldsymbol{z})\| \le \bar{\varepsilon}$ where $\bar{\varepsilon}$ is a positive constant, and that the free parameters θ are updated according to (25). Then, all the closed-loop signals are uniformly ultimately bounded, and the tracking errors are attracted to a neighbourhood of the origin whose size can be adjusted by control parameters.*

Proof: Let us consider the following Lyapunov function candidate

$$V = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} + \frac{1}{2\eta}\tilde{\theta}^T\tilde{\theta} \tag{26}$$

Differentiating (26) with respect to time and using (20) and (25), we get

$$\dot{V} = -\boldsymbol{s}^T\boldsymbol{K}\,\boldsymbol{s} - \boldsymbol{s}^T\boldsymbol{K}_0\tanh(\boldsymbol{s}/\varepsilon_0) + \boldsymbol{s}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u - \tilde{\theta}^T\left(\boldsymbol{w}(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u - \sigma\theta\right) \tag{27}$$

With (18), (27) becomes

$$\dot{V} = -\boldsymbol{s}^T\boldsymbol{K}\,\boldsymbol{s} - \boldsymbol{s}^T\boldsymbol{K}_0\tanh(\boldsymbol{s}/\varepsilon_0) + \boldsymbol{s}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u - \boldsymbol{e}_u{}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \varepsilon^T(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \sigma\tilde{\theta}^T\theta \tag{28}$$

Using the following inequalities

$$\sigma\tilde{\theta}^T\theta \le -\frac{\sigma}{2}\left\|\tilde{\theta}\right\|^2 + \frac{\sigma}{2}\left\|\theta^*\right\|^2$$
$$\varepsilon^T(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u \le \frac{1}{4}\boldsymbol{e}_u{}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \varepsilon^T(\boldsymbol{z})\boldsymbol{G}(\boldsymbol{x})\varepsilon(\boldsymbol{z})$$
$$\boldsymbol{s}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u \le \frac{1}{2}\boldsymbol{e}_u{}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \frac{1}{2}\boldsymbol{s}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{s}$$

we have

$$\dot{V} \le -\boldsymbol{s}^T\left(\boldsymbol{K} - \frac{1}{2}\boldsymbol{G}(\boldsymbol{x})\right)\boldsymbol{s} - \boldsymbol{s}^T\boldsymbol{K}_0\tanh(\boldsymbol{s}/\varepsilon_0) - \frac{1}{4}\boldsymbol{e}_u{}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u - \frac{\sigma}{2}\left\|\tilde{\theta}\right\|^2 + \varepsilon(\boldsymbol{z})^T\boldsymbol{G}(\boldsymbol{x})\varepsilon(\boldsymbol{z}) + \frac{\sigma}{2}\left\|\theta^*\right\|^2 \tag{29}$$

Since $\varepsilon(\boldsymbol{z})$ and $\boldsymbol{G}(\boldsymbol{x})$ are assumed bounded in this study and θ^* is a constant vector, we can define a positive constant bound ψ_1 as

$$\psi_1 = \sup_t\left(\varepsilon(\boldsymbol{z})^T\boldsymbol{G}(\boldsymbol{x})\varepsilon(\boldsymbol{z})\right) + \frac{1}{2}\sigma\left\|\theta^*\right\|^2$$

Then, (29) can be simplified to

$$\dot{V} \le -\boldsymbol{s}^T\left(\boldsymbol{K} - \frac{1}{2}\overline{g}\boldsymbol{I}_p\right)\boldsymbol{s} - \frac{\sigma}{2}\left\|\tilde{\theta}\right\|^2 - \boldsymbol{s}^T\boldsymbol{K}_0\tanh(\boldsymbol{s}/\varepsilon_0) - \frac{1}{4}\boldsymbol{e}_u{}^T\boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \psi_1 \tag{30}$$

We assume here that each design parameter k_i is chosen such that $k_i > \overline{g}/2$ and we let $\kappa = 2\min_{1\le i\le p}\left(k_i - \overline{g}/2\right)$. Consequently, (30) can be bounded by

$$\dot{V} \le -\alpha_1 V - \boldsymbol{s}^T \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0) - \frac{1}{4}\boldsymbol{e}_u{}^T \boldsymbol{G}(\boldsymbol{x})\boldsymbol{e}_u + \psi_1 \le -\alpha_1 V + \psi_1 \tag{31}$$

where $\alpha_1 = \min(\kappa, \sigma\eta)$.
From (31), one can establish that the Lyapunov function candidate satisfies the following condition

$$0 \le V(t) \le \left(V(0) - \frac{\psi_1}{\alpha_1}\right)e^{-\alpha_1 t} + \frac{\psi_1}{\alpha_1} \tag{32}$$

This last condition implies that $\boldsymbol{s}(t)$ and $\tilde{\theta}(t)$ are uniformly bounded, and that $\boldsymbol{s}(t)$ is uniformly ultimately bounded with respect to the set $\Omega_s = \left\{\boldsymbol{s} : \|\boldsymbol{s}\| \le \sqrt{2\psi_1/\alpha_1}\right\}$. This consequently leads to uniform boundedness of the tracking errors $\left|e_i^{(j)}(t)\right| \le 2^j \lambda_i^{j-r_i+1}\sqrt{2\psi_1/\alpha_1}$, $j = 0,\ldots,r_i - 1$, $i = 1,\ldots,p$ (Slotine & Li, 1991).

Remark 2: In the absence of the approximation error, i.e., $\varepsilon(\boldsymbol{x}) = 0$ in (18), by setting $\sigma = 0$ in (25), one can show that the tracking errors are asymptotically stable, i.e., $e_i(t) \to 0$ as $t \to \infty$, for $i = 1,\ldots,p$.
Remark 3: It is worth noticing that the parameter updating law (25) is not implementable in case the derivative of $\boldsymbol{s}(t)$ is not available. However, a discrete implementable version of (25) can be obtained. Rewriting (25) as

$$\theta(t) = \theta(t-\Delta t) + \int_{t-\Delta t}^{t} \phi_1(\tau)\dot{\boldsymbol{s}}(\tau)d\tau + \int_{t-\Delta t}^{t} \phi_2(\tau)d\tau,$$

where Δt is a small positive constant, $\phi_1 = \eta\boldsymbol{w}(\boldsymbol{z})$ and $\phi_2 = \eta\boldsymbol{w}(\boldsymbol{z})\left(\boldsymbol{K}\boldsymbol{s} + \boldsymbol{K}_0 \tanh(\boldsymbol{s}/\varepsilon_0)\right) - \eta\sigma\theta$. Using the fact that $\dot{\boldsymbol{s}} = d\boldsymbol{s}/dt$, the expression of $\theta(t)$ becomes

$$\theta(t) = \theta(t-\Delta t) + \int_{\boldsymbol{s}(t-\Delta t)}^{\boldsymbol{s}(t)} \phi_1(\tau)d\boldsymbol{s} + \int_{t-\Delta t}^{t} \phi_2(\tau)d\tau.$$

By assuming that $\phi_1(t)$, $\phi_2(t)$ and $\boldsymbol{s}(t)$ are continuous time functions and that Δt is small enough, a discrete implementable version of (25) is given by: $\theta(t) = \theta(t-\Delta t) + \phi_1(t-\Delta t)\left(\boldsymbol{s}(t) - \boldsymbol{s}(t-\Delta t)\right) + \phi_2(t-\Delta t)\Delta t$, which represents a good discrete approximation of the parameter update law (25) if Δt is chosen sufficiently small.

4. Indirect adaptive fuzzy control

In this section we propose to indirectly approximate the unknown ideal controller (8) by identifying the unknown functions $f_i(\boldsymbol{x})$ and $g_{ij}(\boldsymbol{x})$ using fuzzy systems. First, let us assume that the nonlinear functions $f_i(x)$ and $g_{ij}(x)$ can be approximated, over a compact set D_x, by fuzzy systems of the form of (12) as follows

$$f_i(x) = f_i^*(x) + \varepsilon_{f_i}(x); \quad f_i^*(x) = \boldsymbol{w}_{f_i}^T(x)\theta_{f_i}^*, \; i = 1,\ldots,p \tag{33}$$

$$g_{ij}(x) = g_{ij}^*(x) + \varepsilon_{g_{ij}}(x); \quad g_{ij}^*(x) = \boldsymbol{w}_{g_{ij}}^T(x)\theta_{g_{ij}}^*, \; i,j = 1,\ldots,p \tag{34}$$

where $\varepsilon_{f_i}(x)$ and $\varepsilon_{g_{ij}}(x)$ are fuzzy approximation errors, $\theta_{f_i}^*$ and $\theta_{g_{ij}}^*$ are optimal parameter vectors that minimize functions $\left|\varepsilon_{f_i}(x)\right|$ and $\left|\varepsilon_{g_{ij}}(x)\right|$, respectively, and $\boldsymbol{w}_{f_i}(x)$ and $\boldsymbol{w}_{g_{ij}}(x)$ are fuzzy basis function vectors assumed suitably specified by the designer.

In this study, we assume that the used fuzzy systems do not violate the universal approximation property on the operating compact set D_x, which is assumed large enough so that state variables remain within D_x under closed-loop control. So it is reasonable to assume that the minimum approximation errors are bounded for all $x \in D_x$.

Since the ideal parameter vectors $\theta_{f_i}^*$ and $\theta_{g_{ij}}^*$ is unknown, let us use their estimates θ_{f_i} and $\theta_{g_{ij}}$ instead to form the adaptive approximations

$$\hat{f}_i(x) = \boldsymbol{w}_{f_i}^T(x)\theta_{f_i}, \; i = 1,\ldots,p \tag{35}$$

$$\hat{g}_{ij}(x) = \boldsymbol{w}_{g_{ij}}^T(x)\theta_{g_{ij}}, \; i,j = 1,\ldots,p \tag{36}$$

Denote

$$\hat{\boldsymbol{f}}(x) = \left[\hat{f}_1(x),\ldots,\hat{f}_p(x)\right]^T, \; \varepsilon_f(x) = \left[\varepsilon_{f1}(x),\ldots,\varepsilon_{fp}(x)\right]^T$$

$$\hat{\boldsymbol{G}}(x) = \begin{bmatrix} \hat{g}_{11}(x) & \cdots & \hat{g}_{1p}(x) \\ \vdots & \ddots & \vdots \\ \hat{g}_{p1}(x) & \cdots & \hat{g}_{pp}(x) \end{bmatrix}, \; \varepsilon_G(x) = \begin{bmatrix} \varepsilon_{g11}(x) & \cdots & \varepsilon_{g1p}(x) \\ \vdots & \ddots & \vdots \\ \varepsilon_{gp1}(x) & \cdots & \varepsilon_{gpp}(x) \end{bmatrix}$$

Now we can write an expression for the adaptive control law

$$\boldsymbol{u} = \hat{\boldsymbol{G}}^{-1}(x)\left(-\hat{\boldsymbol{f}}(x) + \boldsymbol{v} + \boldsymbol{K}\boldsymbol{s} + \boldsymbol{K}_0 \tanh\left(\boldsymbol{s}/\varepsilon_0\right)\right) \tag{37}$$

This control term results from (8) by using the adaptive fuzzy approximations $\hat{\boldsymbol{f}}(x)$ and $\hat{\boldsymbol{G}}(x)$ instead of actual functions $\boldsymbol{f}(x)$ and $\boldsymbol{G}(x)$, respectively.

Adding and subtracting $\hat{\boldsymbol{f}}(x)$ and $\hat{\boldsymbol{G}}(x)\boldsymbol{u}$ to the right-hand side of (7), we get

$$\dot{\boldsymbol{s}} = \boldsymbol{v} - \left(\boldsymbol{f}(\boldsymbol{x}) - \hat{\boldsymbol{f}}(\boldsymbol{x})\right) - \left(\boldsymbol{G}(\boldsymbol{x}) - \hat{\boldsymbol{G}}(\boldsymbol{x})\right)\boldsymbol{u} - \hat{\boldsymbol{G}}(\boldsymbol{x})\boldsymbol{u} - \hat{\boldsymbol{f}}(\boldsymbol{x}) \tag{38}$$

Using the control law (37), (38) becomes

$$\dot{\boldsymbol{s}} = -\boldsymbol{K}\boldsymbol{s} - \boldsymbol{K}_0 \tanh\left(\boldsymbol{s}/\varepsilon_0\right) - \left(\boldsymbol{f}(\boldsymbol{x}) - \hat{\boldsymbol{f}}(\boldsymbol{x})\right) - \left(\boldsymbol{G}(\boldsymbol{x}) - \hat{\boldsymbol{G}}(\boldsymbol{x})\right)\boldsymbol{u} \tag{39}$$

where each element of the vector $\dot{\boldsymbol{s}}$ is given by

$$\dot{s}_i = -k_i s_i - k_{0i} \tanh\left(s_i/\varepsilon_0\right) - \left(f_i(\boldsymbol{x}) - \hat{f}_i(\boldsymbol{x})\right) - \sum_{j=1}^{p}\left(g_{ij}(\boldsymbol{x}) - \hat{g}_{ij}(\boldsymbol{x})\right)u_j \tag{40}$$

The next task should be the design of adaptive laws for the free parameters θ_{fi} and θ_{gij} such that $\hat{\boldsymbol{f}}(\boldsymbol{x}) + \hat{\boldsymbol{G}}(\boldsymbol{x})\boldsymbol{u}$ approximates, as best as possible, the unknown nonlinear function $\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}$. To this end, let us define the modelling error $\boldsymbol{e}_m$ between $\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}$ and $\hat{\boldsymbol{f}}(\boldsymbol{x}) + \hat{\boldsymbol{G}}(\boldsymbol{x})\boldsymbol{u}$ as:

$$\boldsymbol{e}_m = \left(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}\right) - \left(\hat{\boldsymbol{f}}(\boldsymbol{x}) + \hat{\boldsymbol{G}}(\boldsymbol{x})\boldsymbol{u}\right) = \left(\boldsymbol{f}(\boldsymbol{x}) - \hat{\boldsymbol{f}}(\boldsymbol{x})\right) + \left(\boldsymbol{G}(\boldsymbol{x}) - \hat{\boldsymbol{G}}(\boldsymbol{x})\right)\boldsymbol{u} \tag{41}$$

where each component of the vector $\boldsymbol{e}_m$ is given by

$$e_{mi} = \left(f_i(\boldsymbol{x}) - \hat{f}_i(\boldsymbol{x})\right) - \sum_{j=1}^{p}\left(g_{ij}(\boldsymbol{x}) - \hat{g}_{ij}(\boldsymbol{x})\right)u_j = \left(f_i^*(\boldsymbol{x}) - \hat{f}_i(\boldsymbol{x})\right) + \sum_{j=1}^{p}\left(g_{ij}^*(\boldsymbol{x}) - \hat{g}_{ij}(\boldsymbol{x})\right)u_j + \varepsilon_i(\boldsymbol{x}) \tag{42}$$

$$e_{mi} = \boldsymbol{w}_{f_i}^T(\boldsymbol{x})\tilde{\theta}_{f_i} + \sum_{j=1}^{p}\boldsymbol{w}_{g_{ij}}^T(\boldsymbol{x})\tilde{\theta}_{g_{ij}}u_j + \varepsilon_i(\boldsymbol{x}) \tag{43}$$

with $\varepsilon_i(\boldsymbol{x}) = \varepsilon_{f_i}(\boldsymbol{x}) + \sum_{j=1}^{p}\varepsilon_{g_{ij}}(\boldsymbol{x})u_j$, and $\tilde{\theta}_{f_i} = \theta_{f_i}^* - \theta_{f_i}$ and $\tilde{\theta}_{g_{ij}} = \theta_{g_{ij}}^* - \theta_{g_{ij}}$.

Then, from (39) and (41) we have

$$\boldsymbol{e}_m = -\left(\dot{\boldsymbol{s}} + \boldsymbol{K}\boldsymbol{s} + \boldsymbol{K}_0 \tanh\left(\boldsymbol{s}/\varepsilon_0\right)\right) \tag{44}$$

Now, consider a quadratic cost function; that measures the discrepancy between the unknown nonlinearities and their adaptive fuzzy approximations, defined as

$$J(\theta) = \frac{1}{2}\boldsymbol{e}_m^T\boldsymbol{e}_m = \frac{1}{2}\sum_{i=1}^{p}\left(\left(f_i(\boldsymbol{x}) - \hat{f}_i(\boldsymbol{x})\right) + \sum_{j=1}^{p}\left(g_{ij}(\boldsymbol{x}) - \hat{g}_{ij}(\boldsymbol{x})\right)u_j\right)^2 \tag{45}$$

Applying the gradient method, the minimizing trajectories $\theta_{fi}(t)$ and $\theta_{gij}(t)$ are generated by the following differential equations

$$\begin{cases} \dot{\theta}_{fi} = -\eta \nabla_{\theta_{fi}} J(\theta) \\ \dot{\theta}_{gij} = -\eta \nabla_{\theta_{gij}} J(\theta) \end{cases} \tag{46}$$

where η is a positive constant parameter.
Therefore, the gradient descent algorithm becomes

$$\begin{cases} \dot{\theta}_{fi} = \eta \boldsymbol{w}_{fi}(\boldsymbol{x}) e_{mi} \\ \dot{\theta}_{gij} = \eta \boldsymbol{w}_{gij}(\boldsymbol{x}) u_j e_{mi} \end{cases} \tag{47}$$

Since the modelling error $\boldsymbol{e}_m$ is not available, equation (44) will be used to overcome this design difficulty. Then, we obtain

$$\begin{cases} \dot{\theta}_{fi} = -\eta \boldsymbol{w}_{fi}(\boldsymbol{x})\left(\dot{s} + k_i s_i + k_{0i} \tanh(s_i/\varepsilon_0)\right) \\ \dot{\theta}_{gij} = -\eta \boldsymbol{w}_{gij}(\boldsymbol{x}) u_j \left(\dot{s} + k_i s_i + k_{0i} \tanh(s_i/\varepsilon_0)\right) \end{cases} \tag{48}$$

In order to improve the robustness of the adaptive law (48) in the presence of approximation errors, we modify it by introducing a σ -modification term as follows (Ioannou & Sun, 1996)

$$\begin{cases} \dot{\theta}_{fi} = -\eta \boldsymbol{w}_{fi}(\boldsymbol{x})\left(\dot{s} + k_i s_i + k_{0i} \tanh(s_i/\varepsilon_0)\right) - \eta\sigma\theta_{fi} \\ \dot{\theta}_{gij} = -\eta \boldsymbol{w}_{gij}(\boldsymbol{x}) u_j \left(\dot{s} + k_i s_i + k_{0i} \tanh(s_i/\varepsilon_0)\right) - \eta\sigma\theta_{gij} \end{cases} \tag{49}$$

where σ is a small positive constant.
Before proceeding we need to introduce an assumption about the approximation errors $\varepsilon(\boldsymbol{x}) = \varepsilon_f(\boldsymbol{x}) + \varepsilon_G(\boldsymbol{x})\boldsymbol{u}$. Since $\varepsilon(\boldsymbol{x})$ depends upon the control input $\boldsymbol{u}$, $\varepsilon(\boldsymbol{x})$ is not assumed bounded by constant bounds but it is assumed bounded by functional bounds.
A3: The function $\varepsilon(\boldsymbol{x}) = \varepsilon_f(\boldsymbol{x}) + \varepsilon_G(\boldsymbol{x})\boldsymbol{u}$ is bounded as follows

$$\|\varepsilon(\boldsymbol{x})\| \le \sqrt{\delta_0 + \sum_{i=1}^{p} \delta_i s_i^2} \;\; ; \quad \delta_i > 0,\, i = 0,\ldots,p\,.$$

The following theorem summarizes the stability result for the proposed indirect adaptive control scheme.

Theorem 2: *Consider the system in (1) with the control law defined by (37). Suppose that Assumptions A1-A3 hold and that the free parameters of the used fuzzy systems are updated according to (49). Then, all the closed-loop signals are uniformly ultimately bounded, and the* *tracking errors are attracted to a neighbourhood of the origin whose size can be adjusted by control parameters.*

Proof: Let us consider the following Lyapunov function candidate

$$V = \frac{1}{2}\sum_{i=1}^{p}\left(s_i^2 + \frac{1}{\eta}\tilde{\theta}_{f_i}^T\tilde{\theta}_{f_i} + \frac{1}{\eta}\sum_{j=1}^{p}\tilde{\theta}_{g_{ij}}^T\tilde{\theta}_{g_{ij}} \right) \tag{50}$$

Differentiating (26) with respect to time and using (40), (43), (44) and (49), we get

$$\dot{V} = \sum_{i=1}^{p}\left(-k_i s_i^2 - k_{0i}s_i\tanh\left(s_i/\varepsilon_0\right) - s_i e_{mi} - e_{mi}\left(e_{mi} - \varepsilon_i\left(x\right)\right) + \sigma\tilde{\theta}_{f_i}^T\theta_{f_i} + \sigma\sum_{j=1}^{p}\tilde{\theta}_{g_{ij}}^T\theta_{g_{ij}} \right) \tag{51}$$

Using the following inequalities

$$\sigma\tilde{\theta}_{f_i}^T\theta_{f_i} \le -\frac{\sigma}{2}\left\|\tilde{\theta}_{f_i}\right\|^2 + \frac{\sigma}{2}\left\|\theta_{f_i}^*\right\|^2, \quad \sigma\tilde{\theta}_{g_{ij}}^T\theta_{g_{ij}} \le -\frac{\sigma}{2}\left\|\tilde{\theta}_{g_{ij}}\right\|^2 + \frac{\sigma}{2}\left\|\theta_{g_{ij}}^*\right\|^2$$
$$\varepsilon_i\left(x\right)e_{mi} \le \frac{1}{4}e_{mi}^2 + \varepsilon_i^2\left(x\right), \quad s_i e_{mi} \le \frac{1}{4}e_{mi}^2 + s_i^2$$

we obtain

$$\dot{V} = \sum_{i=1}^{p}\left(-\left(k_i - 1\right)s_i^2 - k_{0i}s_i\tanh\left(\frac{s_i}{\varepsilon_0}\right) - \frac{1}{2}e_{mi}^2 - \frac{\sigma}{2}\left\|\tilde{\theta}_{f_i}\right\|^2 + \frac{\sigma}{2}\left\|\theta_{f_i}^*\right\|^2 - \frac{\sigma}{2}\sum_{j=1}^{p}\left\|\tilde{\theta}_{g_{ij}}\right\|^2 + \frac{\sigma}{2}\sum_{j=1}^{p}\left\|\theta_{g_{ij}}^*\right\|^2 + \varepsilon_i^2\left(x\right) \right) \tag{52}$$

Since $\left\|\varepsilon\left(x\right)\right\|^2 = \sum_{i=1}^{p}\varepsilon_i^2\left(x\right)$ and using assumption A3, we have

$$\dot{V} = \sum_{i=1}^{p}\left(-\left(k_i - \delta_i - 1\right)s_i^2 - k_{0i}s_i\tanh\left(\frac{s_i}{\varepsilon_0}\right) - \frac{1}{2}e_{mi}^2 \right) + \psi_2 \tag{53}$$

with

$$\psi_2 = \delta_0 + \sum_{i=1}^{p}\left(\frac{\sigma}{2}\left\|\theta_{f_i}^*\right\|^2 + \frac{\sigma}{2}\sum_{j=1}^{p}\left\|\theta_{g_{ij}}^*\right\|^2 \right).$$

We assume here that each design parameter k_i is chosen such that $k_i > \left(\delta_i - 1\right)$ and we let $\kappa = 2\min_{1\le i\le p}\left(k_i - \delta_i - 1\right)$. Consequently, (53) can be bounded by

$$\dot{V} \le -\alpha_2 V - \boldsymbol{s}^T \boldsymbol{K}_0 \tanh\left(\boldsymbol{s}/\varepsilon_0\right) - \frac{1}{2} \boldsymbol{e}_m{}^T \boldsymbol{e}_m + \psi_2 \ \le -\alpha_2 V + \psi_2 \tag{54}$$

where $\alpha_2 = \min\left(\kappa, \sigma\eta\right)$.

From (54), one can establish that the Lyapunov function candidate satisfies the following condition

$$0 \le V(t) \le \left(V(0) - \frac{\psi_2}{\alpha_2} \right) e^{-\alpha_2 t} + \frac{\psi_2}{\alpha_2} \tag{55}$$

This last condition implies that $\boldsymbol{s}(t)$, $\tilde{\theta}_{f_i}(t)$ and $\tilde{\theta}_{g_{ij}}(t)$ are uniformly bounded, and that $\boldsymbol{s}(t)$ is uniformly ultimately bounded with respect to the set $\Omega_s = \left\{ \boldsymbol{s} : \left\| \boldsymbol{s} \right\| \le \sqrt{2\psi_2/\alpha_2} \right\}$. This consequently leads to uniform boundedness of the tracking errors $\left| e_i^{(j)}(t) \right| \le 2^j \lambda_i^{j-r_i+1} \sqrt{2\psi_2/\alpha_2}$, $j = 0, \ldots, r_i - 1$, $i = 1, \ldots, p$ (Slotine & Li, 1991).

Remark 4: Since the matrix $\hat{\boldsymbol{G}}(\boldsymbol{x})$ is generated on line, the control law (37) is not well-defined if $\hat{\boldsymbol{G}}(\boldsymbol{x})$ becomes not regular. To overcome this singularity problem, we use a regularized inverse as in (Labiod et al., 2005) given by $\hat{\boldsymbol{G}}^{-1}(\boldsymbol{x}) \approx \hat{\boldsymbol{G}}^T(\boldsymbol{x}) \left[\gamma_0 I_p + \hat{\boldsymbol{G}}(\boldsymbol{x}) \hat{\boldsymbol{G}}^T(\boldsymbol{x}) \right]^{-1}$, where γ_0 is a small positive constant.

Remark 5: In the absence of the approximation error, i.e., $\varepsilon(\boldsymbol{x}) = 0$, by setting $\sigma = 0$ in (49), one can show that the tracking errors are asymptotically stable, i.e., $e_i(t) \to 0$ as $t \to \infty$, for $i = 1, \ldots, p$.

5. Simulation results

In this section, we test the proposed direct adaptive fuzzy control scheme on the tracking control of a two-link rigid robot manipulator with the following dynamics (Labiod et al., 2005; Slotine & Li, 1991; Tong et al., 2000):

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - \begin{bmatrix} -h\dot{q}_2 & -h\left(\dot{q}_1 + \dot{q}_2\right) \\ h\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \right\} \tag{56}$$

where

$$M_{11} = a_1 + 2a_3 \cos(q_2) + 2a_4 \sin(q_2),\ M_{22} = a_2\ , M_{21} = M_{12} = a_2 + a_3 \cos(q_2) + a_4 \sin(q_2),$$

$$h = a_3 \sin(q_2) - a_4 \cos(q_2),$$

with

$$a_1 = I_1 + m_1 l_{c1}^2 + I_e + m_e l_{ce}^2 + m_e l_1^2,\ a_2 = I_e + m_e l_{ce}^2,\ a_3 = m_e l_1 l_{ce} \cos\delta_e,\ a_4 = m_e l_1 l_{ce} \sin\delta_e .$$

In the simulation, the following parameter values are used

$$m_1 = 1,\, m_e = 2,\, l_1 = 1,\, l_{c1} = 0.5,\, l_{ce} = 0.6,\, I_1 = 0.12,\, I_e = 0.25,\, \delta_e = 30^\circ .$$

Let $\boldsymbol{y} = [q_1,\ q_2]^T$, $\boldsymbol{u} = [u_1,\ u_2]^T$, $\boldsymbol{x} = [q_1,\ \dot{q}_1,\ q_2,\ \dot{q}_2]^T$, and

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \end{bmatrix} = -\boldsymbol{M}^{-1} \begin{bmatrix} -h\dot{q}_2 & -h(\dot{q}_1 + \dot{q}_2) \\ h\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix},$$

$$\boldsymbol{G}(\boldsymbol{x}) = \begin{bmatrix} g_{11}(\boldsymbol{x}) & g_{12}(\boldsymbol{x}) \\ g_{21}(\boldsymbol{x}) & g_{22}(\boldsymbol{x}) \end{bmatrix} = \boldsymbol{M}^{-1} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}^{-1} .$$

Then, the robot system given by (56) can be expressed as

$$\ddot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u} \tag{57}$$

which is in the input-output form given by (2). Since the matrix $\boldsymbol{M}$ is positive definite (Slotine & Li, 1991), then it is always regular and $\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{M}^{-1}$ is also positive definite.

The control objective is to force the system outputs q_1 and q_2 to follow the desired trajectories $y_{d1}(t) = \sin(t)$ and $y_{d2}(t) = \cos(t)$, respectively.

To synthesize the direct adaptive fuzzy controller, two fuzzy systems in the form of (12) are used to generate the control signals u_1 and u_2. Each fuzzy system has $\boldsymbol{z} = [e_1(t), \dot{e}_1(t), e_2(t), \dot{e}_2(t)]^T$ as input, and for each input variable z_j, $j = 1,\ldots,4$, three Gaussian membership functions are defined as

$$\mu_{F_j^1}(z_j) = \exp\left(-\frac{1}{2}\left(\frac{z_j + 1.25}{0.6} \right)^2 \right),\ \mu_{F_j^2}(z_j) = \exp\left(-\frac{1}{2}\left(\frac{z_j}{0.6} \right)^2 \right),\ \mu_{F_j^3}(z_j) = \exp\left(-\frac{1}{2}\left(\frac{z_j - 1.25}{0.6} \right)^2 \right).$$

The robot initial conditions are $\boldsymbol{x}(0) = [0.25 \quad 0 \quad 0.5 \quad 0]^T$, and the initial values of the parameter estimates $\theta_1(0)$ and $\theta_2(0)$ are set equal to zero. The design parameters used in this simulation are chosen as follows:

$$\lambda_1 = 1,\ \lambda_2 = 1\ ,\ \boldsymbol{K} = diag[1,1],\ \boldsymbol{K}_0 = diag[5,5],\ \varepsilon_0 = 0.01,\ \eta = 5 \text{ , and } \sigma = 0.001 .$$

The simulation results for the first link are shown in Fig. 1, those for the second link are shown in Fig. 2, and the control input signals are shown in Fig. 3. We can note that the actual trajectories converge to the desired trajectories and the control signals are almost smooth. These simulation results demonstrate the tracking capability of the proposed direct adaptive controller and its effectiveness for control tracking of uncertain multivariable nonlinear systems.

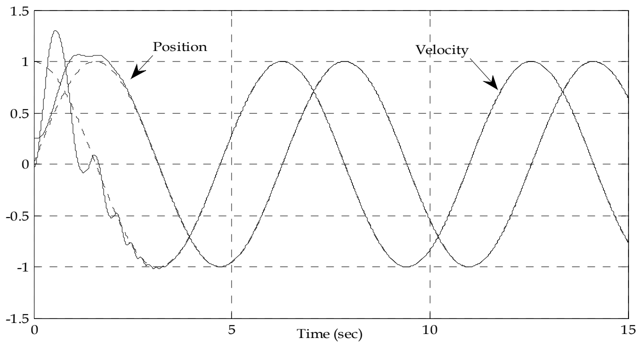


Fig. 1. Tracking curves of link 1: actual (solid lines); desired (dotted lines).

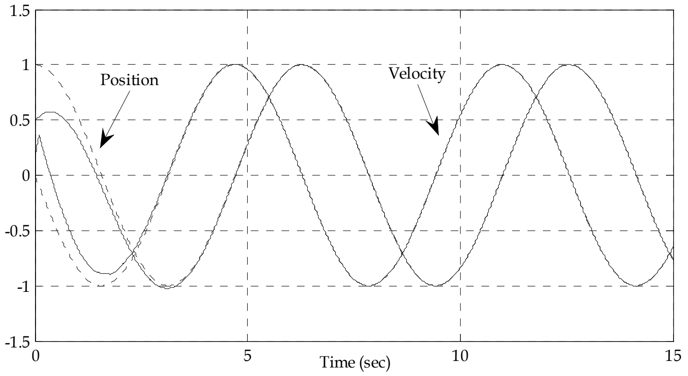


Fig. 2. Tracking curves of link 2: actual (solid lines); desired (dotted lines).

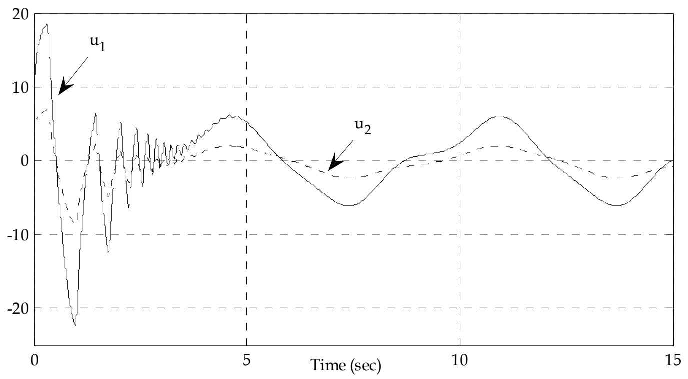


Fig. 3. Control input signals: u_1 (solid line); u_2 (dotted line).

6. Conclusion

In this chapter, stable direct and indirect adaptive fuzzy controllers for a class of MIMO nonlinear systems with uncertain model dynamics are presented. In the direct scheme, fuzzy systems are used to construct adaptively an unknown ideal controller and their adjustable parameters are updated by using the gradient descent method in order to minimize the error between the unknown controller and the fuzzy controller. In the indirect scheme, the controller design is based on the approximation of the system's unknown nonlinearities by using fuzzy systems. The free parameters of the used fuzzy systems in this case are updated using a gradient descent algorithm that is designed to minimize the identification error between the unknown nonlinearities and their adaptive fuzzy approximations. Both approaches do not require the knowledge of the mathematical model of the plant, guarantee the uniform boundedness of all the signals in the closed-loop system, and ensure the convergence of the tracking errors to a neighbourhood of the origin. Simulation results for direct adaptive control scheme performed on a two-link robot manipulator illustrate the method. Future works will focus on extension of the approach to more general MIMO nonlinear systems and its improvement by introducing a state observer to provide an estimate of the state vector.

7. References

Boulkroune, A.; M'Saad, M.; Tadjine, M. & Farza, M. (2008a). Adaptive fuzzy control of MIMO nonlinear systems with unknown hysteresis and control gain matrix sign, *Proc. 16th Mediterranean Conference on Control and Automation*, pp. 380-385, Ajaccio, June 2008.

Boulkroune, A.; M'Saad, M.; Tadjine, M. & Farza, M. (2008b). Adaptive fuzzy control for MIMO nonlinear systems with unknown dead-zone, *Proc. 4th IEEE International Conference on Intelligent Systems*, pp. 50-55, Varna, Sept. 2008.

Chang, Y.C. (2000). Robust tracking control for nonlinear MIMO systems via fuzzy approaches. *Automatica*, Vol. 36, pp. 1535-1545.

Chang, Y.C. (2001). Adaptive fuzzy-based tracking control for nonlinear SISO systems via VSS and $H\infty$ approaches. *IEEE Trans. Fuzzy Systems*, Vol. 9, pp. 278-292.

Chekireb, H.; Tadjine, M. & Bouchaffra, D. (2003). Direct adaptive fuzzy control of nonlinear system class with applications. *Control and Intelligent Systems*, Vol. 31, No. 2, pp. 113-121.

Chiu, C.-S. (2005). Robust adaptive control of uncertain MIMO non-linear systems–feedforward Takagi–Sugeno fuzzy approximation based approach. *IEE Proc.-Control Theory Appl.*, Vol. 152, pp. 157-164.

Essounbouli, N. & Hamzaoui, A. (2006). Direct and indirect robust adaptive fuzzy controllers for a class of nonlinear systems, *International Journal of Control, Automation, and Systems*, Vol. 4, No. 2, pp. 146-154.

Golea, N.; Golea, A. & Benmahammed, K. (2003). Stable indirect fuzzy adaptive control. *Fuzzy sets and Systems*, Vol. 137, pp. 353-366.

Ioannou, P.A. & Sun, J. (1996). *Robust adaptive control*, Prentice Hall, NJ.

Jang, J.S.R. & Sun, C.T. (1995). Neuro-fuzzy modeling and control. *Proc. IEEE*, vol. 83, pp. 378-405.

Krstic, M.; Kanellapoulos, I. & Kokotovic, P. (1995). *Nonlinear and adaptive control design*, Weley Interscience, New York.

Labiod, S. & Boucherit, M.S. (2003). Direct stable fuzzy adaptive control of a class of SISO nonlinear systems. *Archives of Control Sciences*, Vol. 13, pp. 95-110.

Labiod, S.; Boucherit, M.S. & Guerra, T.M. (2005). Adaptive fuzzy control of a class of MIMO nonlinear systems. *Fuzzy Sets and Systems*, Vol. 151, pp. 59–77.

Labiod, S. & Guerra, T.M. (2007a). Adaptive fuzzy control of a class of SISO nonaffine nonlinear systems. *Fuzzy Sets and Systems*, Vol. 158 (10), pp. 1098-1126

Labiod, S. & Guerra, T.M. (2007b). Direct adaptive fuzzy control for a class of MIMO nonlinear systems. *International Journal of Systems Science*, Vol. 38, No. 8, pp. 665-675.

Li, H.-X. & Tong, S.C. (2003). A hybrid adaptive fuzzy control for a class of nonlinear MIMO systems. *IEEE Trans. Fuzzy Systems*, Vol. 11, pp. 24-34.

Ordonez, R. & Passino, K.M. (1999). Stable multi-input multi-output adaptive fuzzy/neural control. *IEEE Trans. Fuzzy Systems*, Vol. 7, pp. 345–353.

Passino, K.M. & Yurkovich, S. (1998). *Fuzzy control*, Addison-Wesley Longman Inc.

Sastry, S.S. & Isidori, A. (1989). Adaptive control of linearizable systems. *IEEE Trans. Automat. Contr.*, Vol. 34, pp. 1123-1131.

Slotine, J.E. & LI, W. (1991). *Applied nonlinear control*, Prentice Hall, Englewood Cliffs, NJ.

Spooner, J.T. & Passino, K.M. (1996). Stable adaptive control using fuzzy systems and neural networks. *IEEE Trans. Fuzzy Systems*, Vol. 4, pp. 339-359.

Spooner, J.T.; Maggiore ; Ordonez, R. & Passino, K.M. (2002). *Stable adaptive control and estimation for nonlinear systems*, John Wiley & Sons.

Su, C.Y. & Stepanenko, Y. (1994). Adaptive control of a class of nonlinear systems with fuzzy logic. *IEEE Trans. Fuzzy Systems*, Vol. 2, pp. 285-294.

Tlemcani, A.; Chekireb, H.; Boucherit, M.S. & Labiod, S. (2007). Decentralized direct adaptive fuzzy control of non-linear interconnected MIMO system class, *Archives of Control Sciences*, Vol. 17, No. 4, pp. 357-374.

Tong, S.C.; Tang, J. & Wang, T. (2000), Fuzzy adaptive control of multivariable nonlinear systems. *Fuzzy Sets and Systems*, Vol. 111, pp. 153–167.

Wang, L.X. (1994). *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice-Hall, Englewood Clifs, NJ.

Zhang, T.P. & YI, Y. (2007). Adaptive fuzzy control for a class of MIMO nonlinear systems with unknown dead-zones. *Acta Automatica Sinica*, Vol. 33, No. 1, pp. 96-99.

Dynamic Trajectory-Tracking Control of an Omnidirectional Mobile Robot Based on a Passive Approach

M. Velasco-Villa, H. Rodríguez-Cortés, I. Estrada-Sanchez, H. Sira-Ramírez and J. A. Vázquez
CINVESTAV-IPN, Departamento de Ingeniería Eléctrica
Sección de Mecatrónica
México D.F., México.

1. Introduction

The traditional control problems of trajectory-tracking and regulation have been extensively studied in the field of mobile robotics. In particular, the differential and the omnidirectional mobile robots, also known, respectively, as the (2,0) and the (3,0) robots (see (Bétourné & Campion, 1996), (Campion et al., 1996), have attracted the interest of many control researchers.
It is a common practice in mobile robotics to address control problems taking into account only a kinematic representation. In (Canudas et al., 1996) and (Campion et al., 1996) the kinematic models for diverse types of mobile robots are presented.
From a kinematic perspective, the trajectory-tracking control problem of (2,0)-type robot has been addressed and solved for example in (D'Andrea-Novel et al., 1992) following a dynamic feedback linearization approach. In (Oriolo et al., 2002) a real time implementation of a dynamic feedback linearization tracking-controller is presented. For the same class of robot, a discrete time approach is considered in (Niño-Suárez et al., 2006) where a path-tracking controller based on a sliding mode control technique is presented.
The regulation and trajectory-tracking problems for the omnidirectional mobile robot (3,0), have also received sustained attention. Considering, only its kinematic model, several control strategies have been proposed. In (Liu et al., 2003), it is designed a nonlinear controller based on a Trajectory Linearization strategy and in (Velasco-Villa et al., 2007), the remote control of the (3,0) mobile robot is developed based on a discrete-time strategy assuming a time-lag model of the robot. In (Velasco-Villa et al., 2007b) the trajectory-tracking problem is solved by means of an estimation strategy that predicts the future values of the system based on the exact nonlinear discrete-time model of the robot.
A more reduced number of contributions have been focused on the dynamic representation of the omnidirectional mobile robot. For example, in (Carter et al., 2001), it is described the mechanical design of a (3,0) robot and based on its dynamic model it is proposed a PID control for each robot wheel. Authors in (Bétourné & Campion, 1996) consider an Euler-

Lagrange model formulation and present an output feedback controller that solves the trajectory-tracking problem. In the same spirit, in (Williams et al., 2002) the dynamic model of the mobile robot is considered in order to study the slipping effects between the wheels of the vehicle and the working surface. In (Chung et al., 2003), the mobile robot is analyzed in the case of a vehicle supporting castor wheels. In (Vázquez & Velasco-Villa, 2008) the trajectory tracking problem is addressed and solved by considering a modification of the well known Computed-Torque strategy. Finally, in (Kalmár-nagy et al. 2004) the time-optimization problem of a desired trajectory is considered for a mobile robot subject to admissible input limits in order to obtain feedback laws that are based on the kinematic and dynamic models.
The analysis of Euler-Lagrange systems has produced several feedback strategies that have been developed mainly in the area of robot manipulators and, lately, in the area of power electronics. In particular, passivity-based control approaches that consider the energy managing structure of the system, for instance: the interconnection and damping assignment technique developed in (Ortega et al., 2001), and passivity-based approaches that exploit the passivity properties of the exact tracking error dynamics and its natural passive output are taken in (Ortega et al., 1998), (Sira-Ramírez & Rodríguez-Cortés, 2008), (Sira-Ramrez, 2005) and (Sira-Ramrez & Silva-Ortigoza, 2006).
We address and solve the trajectory-tracking problem of an omnidirectional mobile robot taking into account its dynamic model. Contrary to the differential case, the considered mobile robot it is not affected by non-holonomics constraints. Following ideas developed in the field of power electronics, in this work, we consider a passivity based control technique that exploit the passivity properties of the exact tracking error addressed as: Exact Tracking Error Dynamics Passive Output Feedback (ETEDPOF), (Sira-Ramrez, 2005), (Sira-Ramrez & Silva-Ortigoza, 2006). The performance of the proposed control strategy is contrasted through numerical simulations with the well-known Computed-Torque Control (Vázquez & Velasco-Villa, 2008) for a desired circular trajectory.
This paper is organized as follows: Section 2 presents the dynamic model of the omnidirectional mobile robot and some structural properties are stated. A brief recall of the Computed-Torque solution is also provided. Immediately, in Section 3 the trajectory-tracking problem is solved by the Exact Tracking Error Dynamics Passive Output Feedback. Closed loop stability is formally proven. In Section 4, the performance of the developed strategy is evaluated by means of numerical simulations and compared with the solution obtained by the Computed-Torque control technique. Section 5 presents some conclusions.

2. Omnidirectional Mobile Robot

A top view of the configuration of a (3,0) mobile robot is depicted in Figure 1. The mobile reference frame $X_m - Y_m$ is located at the center of mass of the vehicle with the X_m axis aligned with respect to the wheel 3. Wheels 1 and 2 are placed symmetrically with an angle $\delta = 30^\circ$ with respect to the Y_m axis. The fixed reference frame $X - Y$ provides the absolute localization of the vehicle on the workspace. The mobile robot is of the type (Canudas et al., 1996) $(\delta_m, \delta_s) = (3,0)$, this is, it has three degrees of mobility and zero degrees of steerability allowing the displacements of the vehicle in all directions instantaneously.

2.1 Dynamic Model

Mobile robot velocity expressed in $X-Y$ coordinates is defined as (Campion et al., 1996),

$$\dot{q} = R^T(\phi)u \tag{1}$$

with,

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix}, \quad u = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}, \quad R(\phi) = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

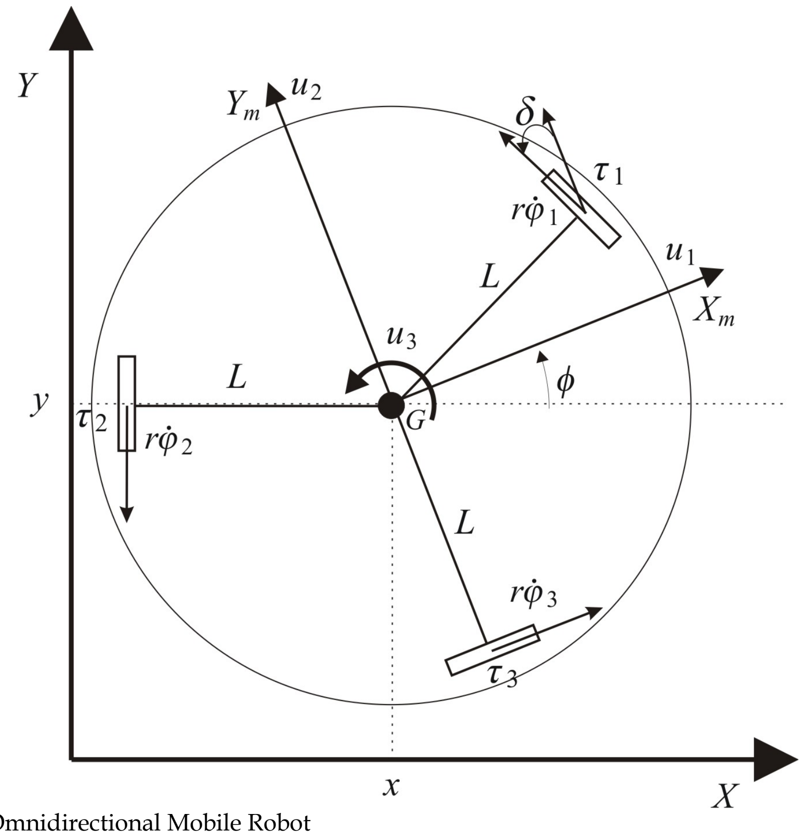


Fig. 1. Omnidirectional Mobile Robot

The point (x, y) is the position of the center of mass of the robot on the plane $X-Y$ and ϕ is the robot orientation with respect to the X-axis. u_1, u_2 are the mobile robot linear velocity expressed in the mobile reference system and u_3 is the rotational velocity measured in the mobile reference system.
A simple analysis of the velocity constrains on Figure 1 produces,

$$J_1 R(\phi)\dot{q} - J_2\dot{\varphi} = 0 \tag{2}$$

with,

$$J_1 = \begin{bmatrix} -\sin\delta & \cos\delta & L \\ -\sin\delta & -\cos\delta & L \\ 1 & 0 & L \end{bmatrix}, \quad J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix},$$

$\varphi = \begin{bmatrix} \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^{\mathrm{T}}$, where φ_1, φ_2, φ_3 represent the angular displacements of wheels one, two and three, respectively; δ is the orientation of the i-wheel with respect to its longitudinal axis; L is the distance between the center of each wheel and the center of the vehicle and r is the radius of each wheel.
Following (Campion et al., 1996)-(Canudas et al., 1996), the kinetic energy of the robot is given by the wheel rotational energy plus the translational and rotational energy of the robot. Therefore, the Lagrangian of the system is obtained as,

$$\mathsf{L} = \frac{1}{2}\dot{q}^T R^T(\phi) M R(\phi)\dot{q} + \frac{1}{2}\sum_{i=1}^{3}\dot{\varphi}_i^{\,T} I_r \dot{\varphi}_i, \tag{3}$$

with $M = diag(M_p, M_p, I_p)$ and $I_r = diag(I_\varphi, I_\varphi, I_\varphi)$. M_p is the vehicle mass I_p the moment of inertia about the Z axis of the vehicle and I_φ is the moment of inertia of each wheel about its rotation axis.
Considering that the kinematics restrictions (2) are satisfied for all t, it is possible to neglect the friction and slipping effects between the wheels and the working surface. Then, the Euler-Lagrange formulation produces the system representation,

$$[M + R^T(\phi)E^T I_r E R(\phi)]\ddot{q} + R^T(\phi)E^T I_r E\dot{R}(\phi)\dot{q} = R^T(\phi)E^T\tau$$

where $\tau = \begin{bmatrix} \tau_1 & \tau_2 & \tau_3 \end{bmatrix}^{\mathrm{T}}$, with τ_i the input torque of each wheel and $E = J_2^{-1}J_1$. Equivalently

$$D\ddot{q} + C(\dot{q})\dot{q} = B\tau, \tag{4}$$

where,

$$D = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_1 & 0 \\ 0 & 0 & d_3 \end{bmatrix}, \quad C(\dot{q}) = a\begin{bmatrix} 0 & \dot{\phi} & 0 \\ -\dot{\phi} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$B = \frac{1}{r}\begin{bmatrix} -\sin(\delta+\phi) & -\sin(\delta-\phi) & \cos\phi \\ \cos(\delta+\phi) & -\cos(\delta-\phi) & \sin\phi \\ L & L & L \end{bmatrix},$$

with $d_1 = M_p + \frac{3I_r}{2r^2}$, $d_3 = I_p + \frac{3I_r L^2}{r^2}$ and $a = \frac{3I_r}{2r^2}$.

2.2 Structural properties

In what follows, some structural properties of the dynamic model (4) are stated. These properties will be necessary to synthesize the proposed control strategy.

Property 1 *The vector* $C(\dot{q})\dot{q}$ *does not possess a unique representation, in particular, for the development of the feedback law, the following alternative parametrization will be considered,*

$$\begin{aligned} C(\dot{q})\dot{q} &= a\begin{bmatrix} 0 & \dot{\phi} & 0 \\ -\dot{\phi} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\dot{q} \\ &= \frac{a}{2}\begin{bmatrix} 0 & \dot{\phi} & \dot{y} \\ -\dot{\phi} & 0 & -\dot{x} \\ -\dot{y} & \dot{x} & 0 \end{bmatrix}\dot{q} = C_a(\dot{q})\dot{q}. \end{aligned}$$

Property 2 *The structure of matrix* $C_a(\dot{q})$ *is such that,*

$$\| C_a(\dot{q}) \| \leq k_c \| \dot{q} \|, \tag{5}$$

where it is easy to show that $k_c = \frac{a}{2}$.

2.3 Computed-Torque Control Solution

Before presenting the main result of the paper we briefly recalled the solution obtained by considering a modified version of the well know Computed-Torque control strategy. Following (Vázquez & Velasco-Villa, 2008), it is possible to consider for system (4) a feedback law of the form,

$$B\tau = D\ddot{q}_d + C(\dot{q}_d)\dot{q}_d - K_p\tilde{q} - K_d\dot{\tilde{q}} + C_r(\dot{q}_d)\dot{q} \tag{6}$$

where q_d is the desired trajectory and $\tilde{q} = q - q_d$ is the tracking error. K_p and K_d are diagonal and positive definite matrices of proportional and derivative gains and matrix $C_r(\dot{q}_d)$ is defined as,

$$C_r(\dot{q}_d) = \frac{a}{2}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 2\dot{y}_d & -2\dot{x}_d & 0 \end{bmatrix}.$$

Feedback (6) in closed loop loop with system (4) produces,

$$D\ddot{\tilde{q}} + C(\dot{q})\dot{q} - C(\dot{q}_d)\dot{q}_d + K_p\tilde{q} + K_d\dot{\tilde{q}} - C_r(\dot{q}_d)\dot{q} = 0,$$

for which, it is possible to formally proof asymptotic closed loop stability. In what follows, it will be presented an alternative feedback control law that solves the same problem.

3. Control design.

Consider the following general model of physical systems (Sira-Ramirez et al., 2006; Sira-Ramírez et al., 2008).

$$\begin{aligned} \mathsf{A}\dot{x} &= \mathsf{J}(x,u)x - \mathsf{R}(x,u)x + \mathsf{B}(x)u + \mathsf{E}(t) \\ y &= \mathsf{B}^{\mathrm{T}}x, \end{aligned} \tag{7}$$

where x is an n dimensional vector, A is a constant symmetric, positive definite matrix, $\mathsf{J}(x,u)$ is a skew symmetric matrix, $\mathsf{R}(x,u)$ is a symmetric positive definite matrix and $\mathsf{E}(t)$ is a n-dimensional smooth vector function of t or sometimes, a constant vector. The input matrix $\mathsf{B}(x)$ is a $n \times m$ matrix and the output vector y is an m dimensional vector. Moreover, we assume that,

$$\begin{aligned} \mathsf{J}(x,u) &= \mathsf{J}_0 + \sum_{j=1}^{m}\mathsf{J}_j^u u_j + \sum_{k=1}^{n}\mathsf{J}_k^x x_k \\ \mathsf{R}(x,u) &= \mathsf{R}_0 + \sum_{j=1}^{m}\mathsf{R}_j^u u_j + \sum_{k=1}^{n}\mathsf{R}_k^x x_k \\ \mathsf{B}(x) &= \mathsf{B}_0 + \sum_{k=1}^{n}\mathsf{B}_k x_k. \end{aligned} \tag{8}$$

Define $e_u = u - u^*(t)$ and the following ,

$$\begin{aligned} \mathsf{M}^*(t) &= \left[\left(\mathsf{J}_1^x - \mathsf{R}_1^x\right)x^* \quad \cdots \quad \left(\mathsf{J}_n^x - \mathsf{R}_n^x\right)x^*\right] \\ \mathsf{L}^*(t) &= \left[B_1x^* \quad \cdots \quad B_nx^*\right] \\ \mathsf{Q}^*(t) &= \left[\left(\mathsf{J}_1^u - \mathsf{R}_1^u\right)x^* \quad \cdots \quad \left(\mathsf{J}_m^u - \mathsf{R}_m^u\right)x^*\right], \end{aligned} \tag{9}$$

where $x^* = x^*(t)$. Straightforward computations show that the error dynamics reads as,

$$\begin{aligned} \mathsf{A}\dot{e} &= \left[\mathsf{J}^*(x,u,t) - \mathsf{R}^*(x,u,t)\right]e + \mathsf{B}^*(x,t)e_u \\ y_e &= \mathsf{B}^*(x,t)^{\mathrm{T}}e \end{aligned} \tag{10}$$

where,

$$
\begin{aligned}
\mathsf{J}^*(x,u,t) &= \mathsf{J}(x,u) + \frac{1}{2}\left[\mathsf{P}^*(t) - \mathsf{P}^*(t)^{\mathrm{T}}\right] \\
\mathsf{R}^*(x,u,t) &= \mathsf{R}(x,u) - \frac{1}{2}\left[\mathsf{P}^*(t) + \mathsf{P}^*(t)^{\mathrm{T}}\right] \\
\mathsf{B}^*(x,t) &= \mathsf{B}(x) + \mathsf{Q}^*(t),
\end{aligned}
\tag{11}
$$

with

$$\mathsf{P}^*(t) = \mathsf{M}^*(t) + \mathsf{L}^*(t).$$

We refer to (10) as the exact open loop error dynamics.
Following Sira-Ramirez et al., 2006; Sira-Ramírez et al., 2008 we have,

Assumption 3 *Given a feasible smooth bounded reference trajectory $x^*(t) \in R^n$, there exists a smooth open loop control input $u^*(t) \in R^m$, such that for all trajectories starting at $x(t_0) = x^*(t_0)$, the tracking error $e(t) = x(t) - x^*(t)$ is identically zero for all $t \geq t_0$.*

Assumption 4 *For any constant positive definite symmetric matrix K the following relation is uniformly satisfied*

$$\mathsf{R}^*(x,u,t) + \mathsf{B}^*(x,t)K\mathsf{B}^*(x,t)^{\mathrm{T}} > 0.$$

Theorem 5 *Consider the system (7)-(8) in closed loop with the controller,*

$$u = u^*(t) - K\mathsf{B}^*(x,t)e. \tag{12}$$

Then, under Assumptions 3 and 4, the tracking error $e(t)$ is globally asymptotically stabilized to zero.

Proof. Take now the following Lyapunov function candidate,

$$V = \frac{1}{2}e^{\mathrm{T}}\mathsf{A}e \tag{13}$$

whose time derivative is given by,

$$\dot{V} = -e^{\mathrm{T}}\mathsf{R}^*(x,u,t)e + e^{\mathrm{T}}\mathsf{B}^*(x,t)e_u.$$

Introducing (11) into the above equation, we have

$$\dot{V} = -e^{\mathrm{T}}[\mathsf{R}^*(x,u,t) + \mathsf{B}^*(x,t)K\mathsf{B}^*(x,t)^{\mathrm{T}}]e,$$

By Assumption 4, the proof is completed.

3.1 Omnidirectional mobile robot case

In the following we apply a slightly modified version of the result given in Theorem 5 to solve the trajectory-tracking control problem of the omnidirectional mobile robot. For this purpose, we express the dynamic model of the mobile robot (4) in terms of (7). Defining initially the feedback,

$$\tau = B^{-1}u, \tag{14}$$

it is obtained,

$$D\ddot{q} + C_a(\dot{q})\dot{q} = u. \tag{15}$$

Consider now the state variables,

$$q = x_1 = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \end{bmatrix} \text{ and } \dot{q} = x_2 = \begin{bmatrix} x_{21} \\ x_{22} \\ x_{23} \end{bmatrix}. \tag{16}$$

Therefore, system (15) can be rewritten as,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ D\dot{x}_2 &= -C_a(x_2)x_2 + u. \end{aligned} \tag{17}$$

Consider now a second feedback law of the form,

$$u = -x_1 - Rx_2 + v \tag{18}$$

with $R = diag\{r_1, r_2, r_3\}$. Replacing (18) into (17) we obtain the following closed-loop system,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ D\dot{x}_2 &= -C_a(x_2)x_2 - x_1 - Rx_2 + v, \end{aligned} \tag{19}$$

that can be rewritten in the form of (7) with,

$$\mathsf{A} = \begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix}, \quad \mathsf{J}(x,u) = \begin{bmatrix} 0 & I \\ -I & -C_a(x_2) \end{bmatrix},$$
$$\mathsf{R}(x,u) = \begin{bmatrix} 0 & 0 \\ 0 & R \end{bmatrix}, \quad \mathsf{B}(x) = \begin{bmatrix} 0 \\ I \end{bmatrix},$$

and I a 3×3 unity matrix. Now, we obtain the dynamics of the tracking error. Straight forward computations show that,

$$\mathsf{L}^*(t) = \mathsf{Q}^*(t) = 0,$$

and

$$\mathsf{M}^*(t) = \begin{bmatrix} 0 & 0 \\ 0 & \mathsf{C}(t) \end{bmatrix}, \tag{20}$$

with

$$\mathsf{C}(t) = a\begin{bmatrix} 0 & -x_{23}^* & -x_{22}^* \\ x_{23}^* & 0 & x_{21}^* \\ -x_{22}^* & x_{21}^* & 0 \end{bmatrix}.$$

Finally, from equation (11) we have,

$$\mathsf{J}^*(x,t) = \begin{bmatrix} 0 & I \\ -I & J_{22}^* \end{bmatrix}, \quad \mathsf{R}^*(t) = \begin{bmatrix} 0 & 0 \\ 0 & R_{22}^* \end{bmatrix}, \qquad (21)$$
$$\mathsf{B}^* = [0 \quad I]^{\mathrm{T}}$$

with

$$J_{22}^* = \begin{bmatrix} 0 & -ax_{23} - ax_{23}^* & -ax_{22} \\ ax_{23} + ax_{23}^* & 0 & ax_{21} \\ ax_{22} & -ax_{21} & 0 \end{bmatrix}$$

and

$$R_{22}^* = \begin{bmatrix} r_1 & 0 & ax_{22}^* \\ 0 & r_2 & -ax_{21}^* \\ ax_{22}^* & -ax_{21}^* & r_3 \end{bmatrix}.$$

Hence, the trajectory-tracking error dynamics is described by,

$$\begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix}\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} = \left\{ \begin{bmatrix} 0 & I \\ -I & J_{22}^* \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & R_{22}^* \end{bmatrix} \right\} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} e_v.$$

3.2 Trajectory-tracking solution: Initial proposal

From the previous developments, it is possible now to state a preliminary solution of the trajectory-tracking problem associated with the considered mobile robot.

Proposition 6 *Consider the dynamic model of the omnidirectional mobile robot (4) in closed loop with the controller,*

$$\tau = B^{-1}\left[-x_1 - Rx_2 + v(t)^* - K\mathsf{B}^* e_2\right], \qquad (22)$$

where $K = diag\{0, K_2\}$ and K_2 a symmetric positive definite matrix. Assume that $x_1(t)^*$, $x_2(t)^*$ and $v(t)^*$ satisfy Assumption 3 and

$$\frac{\lambda_m(R) + \lambda_m(K_2)}{a} > \left\| x_2^* \right\|. \qquad (23)$$

Then, the closed-loop system (4)-(22) renders the equilibrium point $e_1 = x_1 - x_1^* = 0$, $e_2 = x_2 - x_2^* = 0$ *asymptotically stable.*

Proof. The proof of this result can be seen as a particular case of the solution given in the next subsection and it is left to the interested reader.

Remark 7 *It is possible to see that by defining controller (22) as a function only of the velocity error the convergence of* e_1 *is slow. This affects the overall performance of the controller.*

3.3 Main trajectory-tracking solution

In order to improve the controller convergence produced by the feedback law (22), consider now,

$$e_v = -K_1 e_1 - K_2 e_2 \tag{24}$$

with $K_1 = diag\{k_{11}, k_{12}, k_{13}\}$ and $K_2 = diag\{k_{21}, k_{22}, k_{23}\}$, obtaining the closed loop system,

$$\begin{aligned} \dot{e}_1 &= e_2 \\ D\dot{e}_2 &= -e_1 + (J_{22}^* - R_{22}^*)e_2 - K_1 e_1 - K_2 e_2. \end{aligned} \tag{25}$$

To show the convergence of the tracking error notice first that $(e_1, e_2) = (0,0)$ is an equilibrium point of system (25). Consider, now a candidate Lyapunov function of the form,

$$V(e_1, e_2) = \frac{1}{2} e_2^T D e_2 + \frac{1}{2} e_1^T e_1 + \varepsilon e_1^T D e_2 + \frac{1}{2} \varepsilon e_1^T K_2 e_1. \tag{26}$$

It is not difficult to see that this function is positive definite for sufficiently small ε. Taking the time derivative of equation (26) along (25) it is obtained,

$$\begin{aligned} \dot{V} = &-\varepsilon e_1^T \left[I + K_1\right] e_1 - e_2^T \left[R_{22}^* + K_2 - \varepsilon D\right] e_2 \\ &+ \varepsilon e_1^T \left[(J_{22}^* - R_{22}^*) - K_1\right] e_2. \end{aligned} \tag{27}$$

Note now that R_{22}^* can be written as $R_{22}^* = R + aF(x_{2d})$ with,

$$F(x_{2d}) = \begin{bmatrix} 0 & 0 & x_{22d} \\ 0 & 0 & -x_{21d} \\ x_{22d} & -x_{21d} & 0 \end{bmatrix},$$

and also, $J_{22}^* - R_{22}^*$ can be rewritten as $J_{22}^* - R_{22}^* = -R + C_a(e_2) + aG(x_{2d})$ with

$$G(x_{2d}) = \begin{bmatrix} 0 & -x_{23d} & -x_{22d} \\ x_{23d} & 0 & x_{21d} \\ 0 & 0 & 0 \end{bmatrix}.$$

From the fact that,

$$\|F(x_{2d})\| \le \|x_{2d}\| \quad \text{and} \quad \|G(x_{2d})\| \le \|x_{2d}\|,$$

lengthy but simple computations show that,

$$\begin{aligned}\dot{V} \le -\{&[\lambda_m(K_1)+\varepsilon]\|e_1\|^2 + [\lambda_m(K_2)+\lambda_m(R_{22}) \\ &- \varepsilon\lambda_M(D) - a\|x_{2d}\| - \varepsilon a\|e_1\|]\|e_2\|^2 \\ &-[\varepsilon\lambda_M(R_{22}) + \varepsilon a\|x_{2d}\| + \lambda_M(k_1)]\|e_1\|\|e_2\|\}.\end{aligned}$$

Notice now that the above relation can be rewritten as:

$$\dot{V} \le -\begin{bmatrix}\|e_1\| & \|e_2\|\end{bmatrix} P \begin{bmatrix}\|e_1\| \\ \|e_2\|\end{bmatrix}$$

with

$$P = \begin{bmatrix}[\lambda_m(K_1)+\varepsilon] & p_{12} \\ p_{12} & p_{22}\end{bmatrix},$$

where

$$\begin{aligned}p_{12} &= -\frac{1}{2}\left[\varepsilon\lambda_M(R_{22}) + \varepsilon a\|x_{2d}\| + \lambda_M(K_1)\right] \\ p_{22} &= \lambda_m(K_2) + \lambda_m(R_{22}) - \varepsilon\lambda_M(D) - a\|x_{2d}\| - \varepsilon a\|e_1\|.\end{aligned}$$

Therefore, the closed-loop system will be stable if the conditions,

$$\begin{aligned}&(i) \quad \lambda_m(K_1)+\varepsilon > 0 \\ &(ii) \quad \det\{P\} > 0\end{aligned}$$

are satisfied. Condition (i) is trivially satisfied while condition (ii) can be written as a second order function of ε, that is,

$$-\gamma_1\varepsilon^2 + \gamma_2\varepsilon + \gamma_3 > 0 \tag{28}$$

where,

$$\begin{aligned}\gamma_1 &= \lambda_M(D) + a\|e_1\| + \frac{1}{4}\left[\lambda_M(R_{22}) + a\|x_{2d}\|\right]^2 \\ \gamma_2 &= \lambda_m(K_2) + \lambda_m(R_{22}) - a\|x_{2d}\| - \lambda_m(K_1)\left[\lambda_M(D) + a\|e_1\|\right] \\ &\quad - \lambda_M(K_1)\left[\lambda_M(R_{22}) + a\|x_{2d}\|\right] \\ \gamma_3 &= \lambda_m(K_1)\left[\lambda_m(K_2) + \lambda_m(R_{22}) - a\|x_{2d}\|\right] - \frac{1}{4}\lambda_M^2(K_1).\end{aligned}$$

It is clear now, from equation (28), that the system will be asymptotically stable for a sufficiently small ε. Notice that when $\varepsilon \to 0$ the required stability condition is reduced to $\gamma_3 > 0$ that can be easily obtained by an adequate selection of the control gains together with a bounded desired velocity. Hence, we have shown:

Proposition 8 *Consider the dynamic model of the omnidirectional mobile robot (4) in closed loop with the controller*

$$\tau = B_a^{-1}[-x_1 - Rx_2 + v(t)^* - K_1e_1 - K_2e_2] \tag{29}$$

where K_1 *and* K_2 *are symmetric positive definite matrices. Assume that* $x_1(t)^*$, $x_2(t)^*$ *and* $v(t)^*$ *satisfy Assumption 3 and* $\gamma_3 > 0$. *Then, the closed-loop system (4)-(22) renders the equilibrium point* $e_1 = 0$, $e_2 = 0$ *asymptotically stable.*

4. Numerical simulations

We carried out numerical simulations to assess the performance of the controller given in Proposition 8. The values of the parameters correspond to a laboratory prototype built in our institution and they are $M_p = 9.58$ Kg, $I_r = 0.52\ \text{Kgm}^2$, $L = 0.1877$ m, $r = 0.03812$ m and $\delta = 30^o$. The initial conditions of the mobile robot are $x_1(0) = [0, \quad 0, \quad 1.5]^{\mathrm{T}}$ and $x_2(0) = [0, \quad 0, \quad 0]^{\mathrm{T}}$. Finally, the controller parameters are summarized in Table 1.

Parameter	Value	Parameter	Value
k_{11}	200	k_{21}	200
k_{12}	200	k_{22}	200
k_{13}	200	k_{23}	100
r_1, r_2	200	r_3	30

Table 1. Feedback control law parameters

It is desired to follow a circular trajectory or radius 0.5 m centered at the origin with initial conditions $x_{1d}(0) = [0.5, \quad 0, \quad \pi/2]^{\mathrm{T}}$.

Figure 2 shows the evolution on the plane of the mobile robot when it is considered the control strategy proposed on this paper (P) and the one obtained when the Computed-Torque control (CT) (6) is considered. The torque input signals are shown on Figure 3 for the passive approach and on Figure 4 for the Computed-Torque control. It is clear that by selecting the control gains under the restriction of equivalent torque magnitude, the control strategy proposed in this work has a better performance than the one obtained by the Computed-Torque scheme. Finally, the evolution of the position and velocity errors for the passivity control strategy are shown on Figures 5 and 6 respectively.

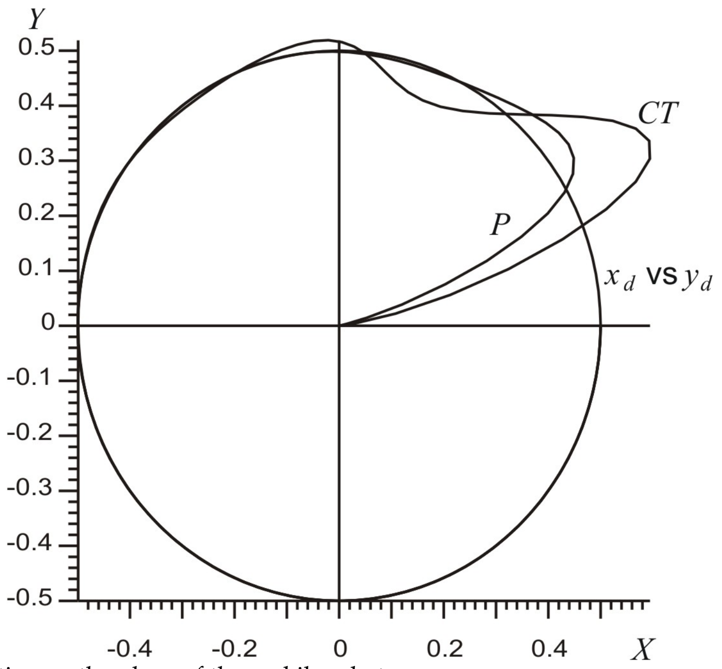


Fig. 2. Evolution on the plane of the mobile robot.

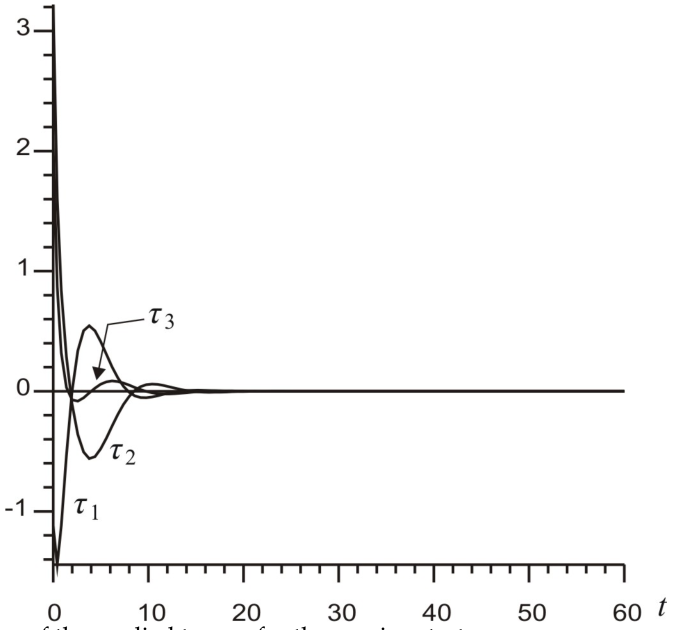


Fig. 3. Evolution of the applied torque for the passive strategy.

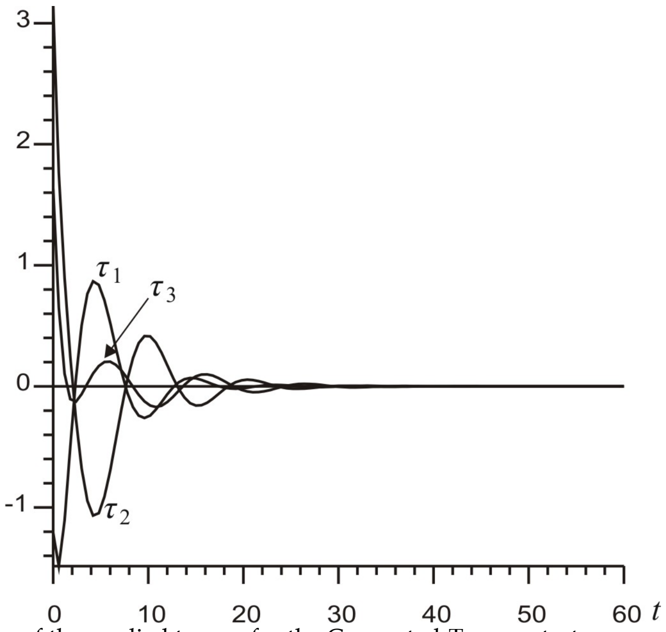


Fig. 4. Evolution of the applied torque for the Computed-Torque strategy.

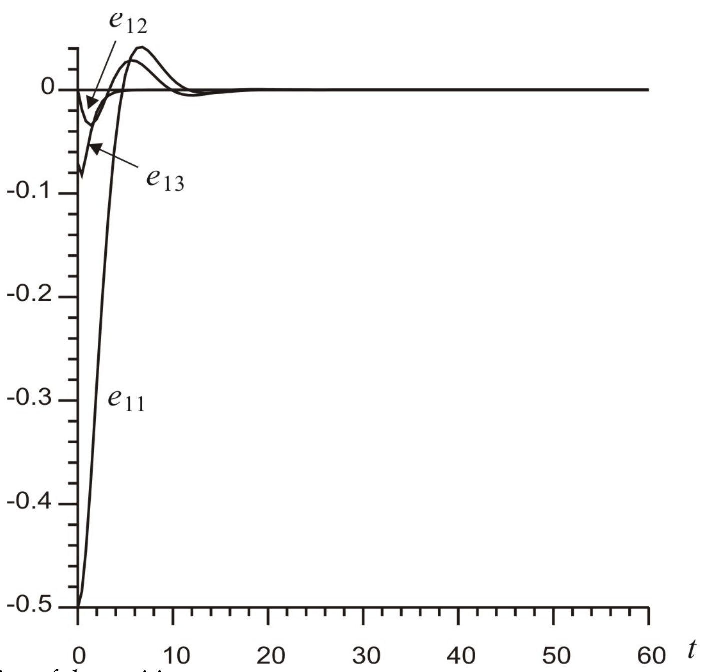


Fig. 5. Evolution of the position errors.

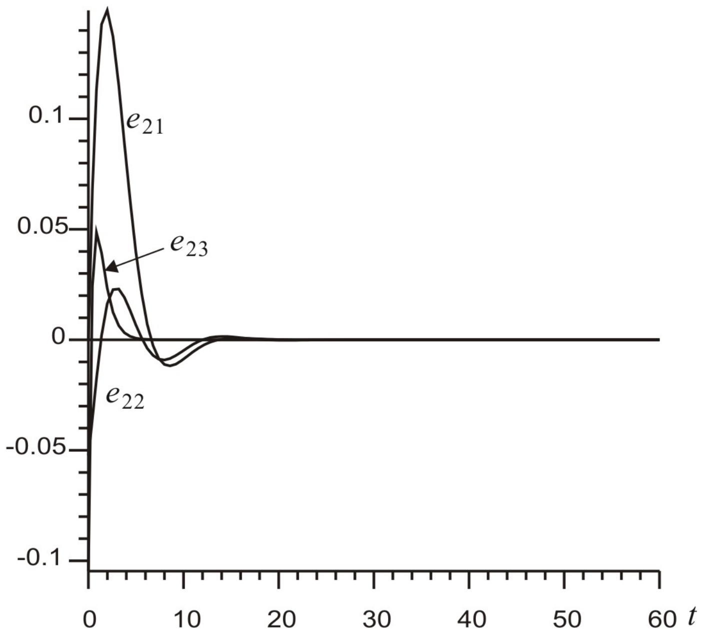


Fig. 6. Velocity errors.

5. Conclusions

The trajectory-tracking problem for the omnidirectional mobile robot considering its dynamic model has been addressed and solved by means of a full state information time varying feedback based on a methodology that exploits the passivity properties of the exact tracking error dynamics. The asymptotic stability of the closed loop system is formally proved. Numerical simulations are proposed to illustrate the properties of the closed-loop system showing a better performance than the control obtained by the well known Computed-Torque approach.

6. Acknowledgment

This work was partially supported by CONACyT México, under Grants: 61713 and 82741.

7. References

Bétourné, A. & Campion G. (1996) Dynamic Modelling and Control Design of a Class of Omnidirectional Mobile Robots. *Proceedings of the 1996 IEEE Int. Conference on Robotics and Automation*, pp. 2810-2815, Minneapolis, USA.

Campion, G.; Bastin, G. & D'Andréa-Novel, B. (1996) Structural Properties and Clasification of Kinematics and Dynamics Models of Wheeled Mobile Robots. *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 1, pp. 47-61.

Canudas, C.; Siciliano, B.; Bastin, G.; Brogliato, B.; Campion, G.; D'Andrea-Novel, B. ; De Luca, A.; Khalil, W.; Lozano, R.; Ortega, R.; Samson, C. & Tomei, P. (1996) *Theory of Robot Control*. Springer-Verlag, London.

Carter, B.; Good, M.; Dorohoff, M.; Lew, J.; Williams II, R. L. & Gallina, P. (2001) Mechanical design and modeling of an omni-directional robocup player. *Proceedings RoboCup 2001 International Symposium*, Seattle, WA, USA.

Chung, J. H.; Yi, B. J.; Kim, W. K. & Lee, H. (2003) The Dynamic Modeling and Analysis for An Omnidirectional Mobile Robot with Three Caster Wheels. *Proceedings of the 2003 IEEE Int. Conference on Robotics and Automation*, pp. 521-527, Taipei, Taiwan.

D'Andrea-Novel, B.; Bastin, G. & Campion, G. (1992) Dynamic Feedback Linearization of Nonholonomic Wheeled Mobile Robots. *Proceedings of the IEEE International Conference on Robotic and Automation*, pp. 2527-2532, Nice, France.

Kalmár-Nagy, T.; D'Andrea, R. & Ganguly, P. (2004) Near-Optimal Dynamic Trajectory and Control of an Omnidirectional Vehicle. *Robotics and Autonomous Systems*, Vol. 46, pp. 47-64.

Liu, Y.; Wu, X.; Zhu, J. and Lew, J. (2003) Omni-directional mobile robot controller design by trajectory linearization. *Proceedings of the American Control Conference*, pp. 3423-3428, Denver, Colorado, USA.

Niño-Suárez, P. A.; Aranda-Bricaire, E. & Velasco-Villa, M. (2006) Discrete-time sliding mode path-tracking control for a wheeled mobile robot. *Proc. of the 45th IEEE Conference on Decision and Control*, pp. 3052-3057, San Diego, CA, USA.

Oriolo, G.; De Luca, A. & Venditteli, M. (2002) WMR control via dynamic feedback linearization: Design, implementation, and experimental validation. *IEEE Transaction on Control Systems Technology*, Vol. 10, No. 6, pp. 835-852.

Ortega, R.; Loria, A.; Nicklasson, P. J. & Sira-Ramírez H. (1998) *Passivity-based Control of Euler Lagrange Systems*. Springer, New York, USA.

Ortega, R.; van der Schaft, A.; Mareels, I. & Maschke, B. (2001) Putting energy back in control. *IEEE Control Syst. Magazine*, Vol. 21, No. 2, pp. 18-33.

Sira-Ramrez H. (2005) Are non-linear controllers really necessary in power electronics devices?. *European Power Electronics Conference EPE-2005*, Dresden, Germany.

Sira-Ramrez, H. & Silva-Ortigoza, R. (2006) *Design Techniques in Power Electronics Devices*. Springer-Verlag, Power Systems Series,, London. ISBN: 1-84628-458-9.

Sira-Ramírez, H. & Rodríguez-Cortés, H. (2008) Passivity Based Control of Electric Drives. Internal Report, Centro de Investigación y de Estudios Avanzados, 2008.

Velasco-Villa M.; Alvarez-Aguirre, A. & Rivera-Zago G. (2007) Discrete-Time control of an omnidirectional mobile robot subject to transport delay. *IEEE American Control Conference 2007*, pp. 2171-2176, New York City, USA.

Velasco-Villa M.; del-Muro-Cuellar B. &Alvarez-Aguirre, A. (2007) Smith-Predictor compensator for a delayed omnidirectional mobile robot. *15th Mediterranean Conference on Control and Automation*, T30-027, Athens, Greece.

Vázquez J. A. & Velasco-Villa M. (2008) Path-Tracking Dynamic Model Based Control of an Omnidirectional Mobile Robot. *17th IFAC World Congress*, Seoul, Korea.

Williams, R. L.; Carter, B. E.; Gallina, P. & G. Rosati. (2002) Dynamic Model With Slip for Wheeled Omnidirectional Robots. *IEEE Transactions on Robotics and Automation*, Vol. 18, pp. 285-293.

Eclectic Theory of Intelligent Robots

E. L. Hall, S. M. Alhaj Ali*, M. Ghaffari,
X. Liao and Ming Cao
Center for Robotics Research
University of Cincinnati
Cincinnati, OH 45221-0072 USA
** The Hashemite Univ. (Jordan)*

1. Introduction

The purpose of this paper is to describe a concept of eclecticism for the design, development, simulation and implementation of a real time controller for an intelligent, vision guided robot or robots. The use of an eclectic perceptual, creative controller that can select its own tasks and perform autonomous operations is illustrated. This eclectic controller is a new paradigm for robot controllers and is an attempt to simplify the application of intelligent machines in general and robots in particular. The idea is to uses a task control center and dynamic programming approach. However, the information required for an optimal solution may only partially reside in a dynamic database so that some tasks are impossible to accomplish. So a decision must be made about the feasibility of a solution to a task before the task is attempted. Even when tasks are feasible, an iterative learning approach may be required. The learning could go on forever. The dynamic database stores both global environmental information and local information including the kinematic and dynamic models of the intelligent robot. The kinematic model is very useful for position control and simulations. However, models of the dynamics of the manipulators are needed for tracking control of the robot's motions. Such models are also necessary for sizing the actuators, tuning the controller, and achieving superior performance. Simulations of various control designs are shown. Much of the model has also been used for the actual prototype Bearcat Cub mobile robot. This vision guided robot was designed for the Intelligent Ground Vehicle Contest. A novel feature of the proposed approach lies in the fact that it is applicable to both robot arm manipulators and mobile robots such as wheeled mobile robots. This generality should encourage the development of more mobile robots with manipulator capability since both models can be easily stored in the dynamic database. The multi task controller also permits wide applications. The use of manipulators and mobile bases with a high-level control are potentially useful for space exploration, manufacturing robots, defense robots, medical robotics, and robots that aid people in daily living activities.
An important question in the application of intelligent machines is: can a major paradigm shift can be effected from industrial robots to a more generic service robot solution? That is, can we perform an eclectic design? (Hall, et al. 2007)

The purpose of this paper is to examine the theory of robust learning for intelligent machines. A main question in the application of intelligent machines is: can a major paradigm shift can be effected?

Eclecticism as defined by Wikipedia as " a conceptual approach that does not hold rigidly to a single paradigm or set of assumptions, but instead draws upon multiple theories, styles, or ideas to gain complementary insights into a subject, or applies different theories in particular cases." http://en.wikipedia.org/wiki/Eclecticism

A scientific paradigm had been defined by Kuhn as "answers to the following key questions:

- what is to be observed and scrutinized,
- what kind of questions are supposed to be asked and probed for answers in relation to this subject,
- how are these questions are to be structured,
- how should the results of scientific investigations be interpreted.
- how is an experiment to be conducted, and what equipment is available to conduct the experiment.

"Thus, within normal science, the paradigm is the set of exemplary experiments that are likely to be copied or emulated. The prevailing paradigm often represents a more specific way of viewing reality, or limitations on acceptable programs for future research, than the much more general scientific method."

In the eclectic control, some answers to the key questions are:

- *The performance of the intelligent machine will be observed*
- *Actual or simulated behaviors will lead to questions of normal or useful responses*
- *Questions should be structured to permit answers from queries of the database*
- *Objectively by anyone in the world*
- *Simulations are much more cost effective than actual performance tests*

The proposed theory for eclectic learning is also based on the previous perceptual creative controller for an intelligent robot that uses a multi- modal adaptive critic for performing learning in an unsupervised situation but can also be trained for tasks in another mode and then is permitted to operate autonomously. The robust nature is derived from the automatic changing of task modes based on a dynamic data base and internal measurements of error at appropriate locations in the controller.
The eclectic controller method is designed for complex real world environments. However, analysis and simulation is needed to clarify the decision processes and reduce the danger in real world operations.
The eclectic controller uses a perceptual creative learning architecture to integrate a Task Control Center (TCC) and a dynamic database (DD) with adaptive critic learning algorithms to permit these solutions. Determining the tasks to be performed and the data base to be updated are the two key elements of the design. These new decision processes encompass both decision and estimation theory and can be modeled by neural networks and implemented with multi-threaded computers.
The main thrust of this paper is to present the eclectic theory of learning that can be used for developing control architectures for intelligent machines. Emphasis will be placed on the

missing key element, the dynamic data base, since the control architectures for neural network control of vehicles in which the kinematic and dynamic models are known but one or more parameters must be estimated is a simple task that has been demonstrated.
The mathematical models for the kinematics and dynamics were developed and the main emphasis was to explore the use of neural network control and demonstrate the advantages of these learning methods. The results indicate the method of solution and its potential application to a large number of currently unsolved problems in complex environments. The adaptive critic neural network control is an important starting point for future learning theories that are applicable to robust control and learning situations.
The general goal of this research is to further develop an eclectic theory of learning that is based on human learning but applicable to machine learning and to demonstrate its application in the design of robust intelligent systems. To obtain broadly applicable results, a generalization of adaptive critic learning called Creative Control (CC) for intelligent robots in complex, unstructured environments has been used. The creative control learning architecture integrates a Task Control Center (TCC) and a Dynamic Knowledge Database (DKD) with adaptive critic learning algorithms.
Recent learning theories such as the adaptive critic have been proposed in which a critic provides a grade to the controller of an action module such as a robot. The creative control process which is used is “beyond the adaptive critic.”
A mathematical model of the creative control process is presented that illustrates the use for mobile robots.

1.1 Dynamic Programming

The intelligent robot in this paper is defined as a decision maker for a dynamic system that may make decisions in discrete stages or over a time horizon. The outcome of each decision may not be fully predictable but may be anticipated or estimated to some extent before the next decision is made. Furthermore, an objective or cost function can be defined for the decision. There may also be natural constraints. Generally, the goal is to minimize this cost function over some decision space subject to the constraints. With this definition, the intelligent robot can be considered as a set of problems in dynamic programming and optimal control as defined by Bertsekas (Bertsekas, 2000).
Dynamic programming (DP) is the only approach for sequential optimization applicable to general nonlinear and stochastic environments. However, DP needs efficient approximate methods to overcome its dimensionality problems. It is only with the presence of artificial neural network (ANN) and the invention of back propagation that such a powerful and universal approximate method has become a reality.
The essence of dynamic programming is Bellman's *Principle of Optimality.* (White and Sofge, 1992) *“An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision”* (Bertsekas, 2000) *(p.83).*

The original Bellman equation of dynamic programming for adaptive critic algorithm may be written as shown in Eq (1):

$$J(R(t)) = \max_{u(t)}(U(R(t),u(t)) + < J(R(t+1)) >)/(1+r) - U_0 \qquad (1)$$

Where R(t) is the model of reality or state form, U(R(t),u(t)) is the utility function or local cost, u(t) is the action vector, J(R(t)) is the criteria or cost-to-go function at time t, r and U_0 are constants that are used only in infinite-time-horizon problems and then only sometimes, and where the angle brackets refer to expected value.
The user provides a utility function, U, and a stochastic model of the plant, R, to be controlled. The expert system then tries to solve the Bellman equation for the chosen model and utility function to achieve the optimum value of J by picking the action vector u(t). If an optimum J cannot be determined, an approximate or estimate value of the J function is used to obtain an approximate optimal solution.
Regarding the finite horizon problems, which we normally try to cope with, one can use Eq (2):

$$J(R(t)) = \max_{u(t)}(U(R(t),u(t)) + < J(R(t+1)) >)/(1+r) \qquad (2)$$

Dynamic programming gives the exact solution to the problem of how to maximize a utility function U(R(t), u(t)) over the future times, t, in a nonlinear stochastic environment. Dynamic programming converts a difficult long-term problem in optimization over time <U(R(t))>, the expected value of U(R(t)) over all the future times, into a much more straightforward problem in simple, short-term function maximization – after we know the function J. Thus, all of the approximate dynamic programming methods discussed here are forced to use some kind of general-purpose nonlinear approximation to the J function, the value function in the Bellman equation, or something closely related to J(Werbos, 1999).

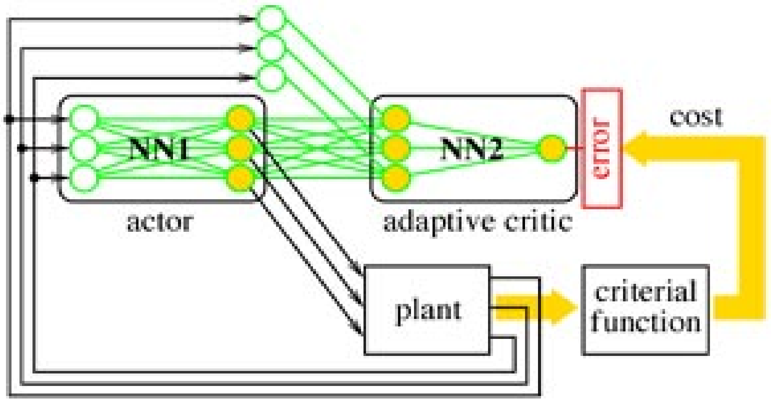
In most forms of adaptive critic design, we approximate J by using a neural network. Therefore, we approximate J(R) by some function $\hat{J}(R,W)$, where W is a set of weights or parameters, $\hat{J}$ is called a critic network (Widrow, et al., 1973)

If the weights **W** are adapted or iteratively solved for, in real time learning or offline iteration, we call the Critic an Adaptive Critic (Werbos, 1999).

An adaptive critic design (ACD) is any system which includes an adapted critic component; a critic, in turn, is a neural net or other nonlinear function approximation which is trained to converge to the function **J(X).**
In adaptive critic learning or designs, the critic network learns to approximate the cost-to-go or strategic utility function J and uses the output of an action network as one of its' inputs, directly or indirectly. When the critic network learns, back propagation of error signals is possible along its input feedback to the action network. To the back propagation algorithm, this input feedback looks like another synaptic connection that needs weights adjustment. Thus, no desired control action information or trajectory is needed as supervised learning.

2. Adaptive Critic And Creative Control

Most advanced methods in neurocontrol are based on adaptive critic learning techniques consisting of an action network, adaptive critic network, and model or identification network as show in Figure 1. These methods are able to control processes in such a way, which is approximately optimal with respect to any given criteria taking into consideration of particular nonlinear environment. For instance, when searching for an optimal trajectory to the target position, the distance of the robot from this target position can be used as a criteria function. The algorithm will compute the proper steering, acceleration signals for control of vehicle, and the resulting trajectory of the vehicle will be close to optimal. During trials (the number depends on the problem and the algorithm used) the system will improve performance and the resulting trajectory will be close to optimal. The freedom of choice of the criteria function makes the method applicable to a variety of problems. The ability to derive a control strategy only from trial/error experience makes the system capable of semantic closure. These are very strong advantages of this method.



Structure of the adaptive critic controller based on artifical neural networks.

Fig. 1. Structure of the adaptive critic controller (Jaska and Sinc, 2000)

Creative Learning Structure

It is assumed that we can use a kinematic model of a mobile robot to provide a simulated experience to construct a value function in the critic network and to design a kinematic based controller for the action network. A proposed diagram of creative learning algorithm is shown in Figure 2 (Jaska and Sinc, 2000). In this proposed diagram, there are six important components: the task control center, the dynamic knowledge database, the critic network, the action network, the model-based action and the utility funtion. Both the critic network and action network can be constructed by using any artificial neural networks with sigmoidal function or radial basis function (RBF). Furthermore, the kinematic model is also used to construct a model-based action in the framework of adaptive critic-action approach. In this algorithm, dynamic databases are built to generalize the critic network and its training process and provide evironmental information for decision making. It is especially critical when the operation of mobile robots is in an unstructured environments. Furthermore, the dynamic databases can also used to store environmental parameters such as Global Position System (GPS) way points, map information, etc. Another component in the diagram is the utility function for a tracking problem (error measurement). In the diagram, X_k, X_{kd}, X_{kd+1} are inputs and Y is the ouput and J(t), J(t+1) is the critic function at the time.

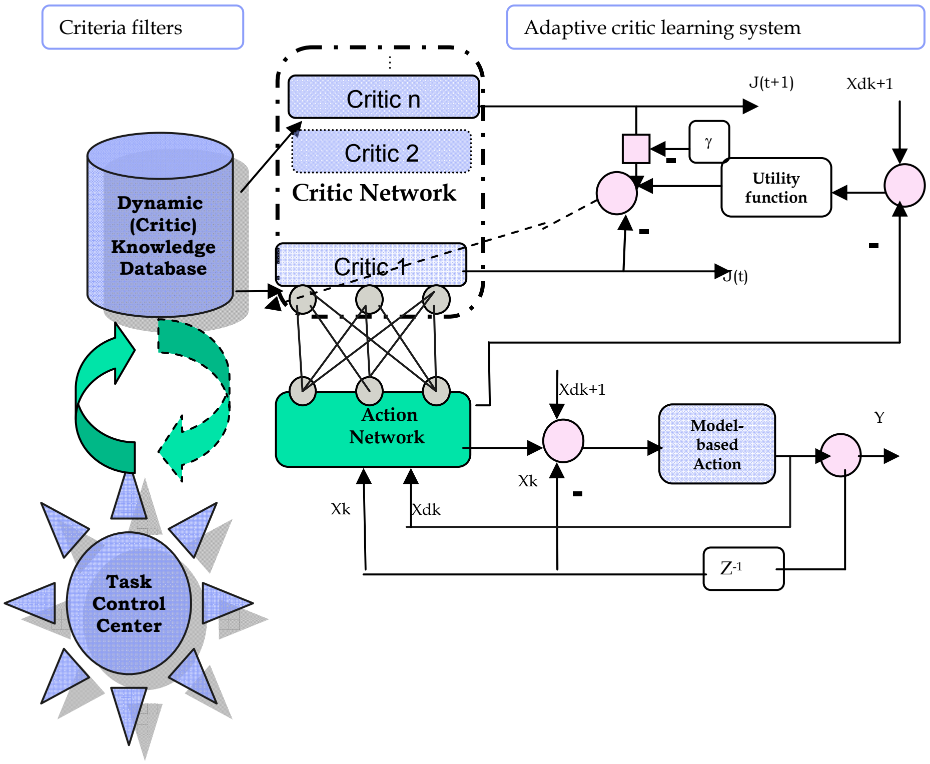


Fig. 2. Proposed Creative Learning Algorithm Structure

Dynamic Knowledge Database (DKD)

The dynamic databases contain domain knowledge and can be modified to permit adaptation to a changing environment. Dynamic knowledge databases may be called a “neurointerface” (Widrow and Lamego, 2002) in a dynamic filtering system based on neural networks (NNs) and serves as a “coupler” between a task control center and a nonlinear system or plant that is to be controlled or directed. The purpose of the coupler is to provide the criteria functions for the adaptive critic learning system and filter the task strategies commanded by the task control center. The proposed dynamic database contains a copy of the model (or identification). Action and critic networks are utilized to control the plant under nominal operation, as well as make copies of a set of parameters (or scenario) previously adapted to deal with a plant in a known dynamic environment. The database also stores copies of all the partial derivatives required when updating the neural networks using backpropagation through time (Yen and Lima, 2002). The dynamic database can be expanded to meet the requirements of complex and unstructured environments.
The data stored in the dynamic database can be uploaded to support offline or online training of the dynamic plant and provide a model for identification of nonlinear dynamic

environment with its modeling function. Another function module of the database management is designed to analyze the data stored in the database including the sub-task optima, pre-existing models of the network and newly added models. The task program module is used to communicate with the task control center. The functional structure of the proposed database management system (DBMS) is shown in Figure 3. The DBMS can be customized from an object-relational database.
In existing models the database is considered to be static. The content of the data base may be considered as information. However, our experience with the World Wide Web is that the "information" is dynamic and constantly changing and often wrong.

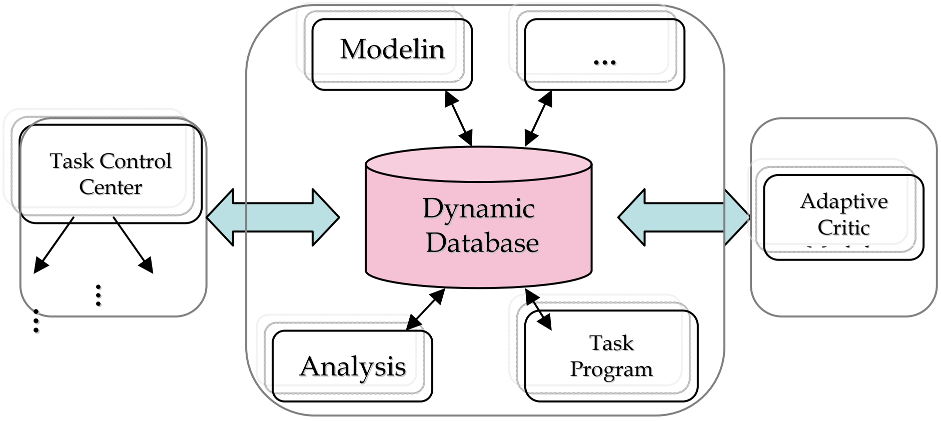


Fig. 3. Functional structure of dynamic database

2.3 Task Control Center (TCC)

The task control center (TCC) can build task-level control systems for the creative learning system. By "task-level", we mean the integration and coordination of perception, planning and real-time control to achieve a given set of goals (tasks) (Lewis, et al., 1999). TCC provides a general task control framework, and it is to be used to control a wide variety of tasks. Although the TCC has no built-in control functions for particular tasks (such as robot path planning algorithms), it provides control functions, such as task decomposition, monitoring, and resource management, that are common to many applications. The particular task built-in rules or criteria or learning J functions are managed by the dynamic database controlled with TCC to handle the allocation of resources. The dynamic database matches the constraints on a particular control scheme or sub-tasks or environment allocated by TCC.
The task control center acts as a decision-making system. It integrates domain knowledge or criteria into the database of the adaptive learning system. According to Simmons (Simmons, 2002), the task control architecture for mobile robots provides a variety of control constructs that are commonly needed in mobile robot applications, and other autonomous mobile systems. The goal of the architecture is to enable autonomous mobile robot systems to easily specify hierarchical task-decomposition strategies, such as how to navigate to a particular location, or how to collect a desired sample, or how to follow a track in an unstructured environment. This can include temporal constraints between sub-goals, leading to a variety of sequential or concurrent behaviors. TCC schedules the execution of planned behaviors, based on those temporal constraints acting as a decision-making control center.

Integrating the TCC with the adaptive critic learning system and interacting with the dynamic database, the creative learning system provides both task-level and real-time control or learning within a single architectural framework. Through interaction with human beings to attain the input information for the system, the TCC could decompose the task strategies to match the dynamic database for the rules of sub-tasks by constructing a distributed system with flexible mechanisms, which automatically provide the right data at the right time. The TCC also provides orderly access to the resources of the dynamic database with built-in learning mechanisms according to a queue mechanism. This is the inter-process communication capability between the task control center and the dynamic database. The algorithm on how to link the task control center and the dynamic database is currently done by the human designers.

Creative learning controller for intelligent robot control

Creative learning may be used to permit exploration of complex and unpredictable environments, and even permit the discovery of unknown problems, ones that are not yet recognized but may be critical to survival or success. By learning the domain knowledge, the system should be able to obtain the global optima and escape local optima. The method attempts to generalizes the highest level of human learning – imagination. As a ANN robot controller, the block diagram of the creative controller can be presented in Figure 4.

Experience with the guidance of a mobile robot has motivated this study and has progressed from simple line following to the more complex navigation and control in an unstructured environment. The purpose of this system is to better understand the adaptive critic learning theory and move forward to develop more human-intelligence-like components into the intelligent robot controller. Moreover, it should extend to other applications. Eventually, integrating a criteria knowledge database into the action module will develop a powerful adaptive critic learning module.

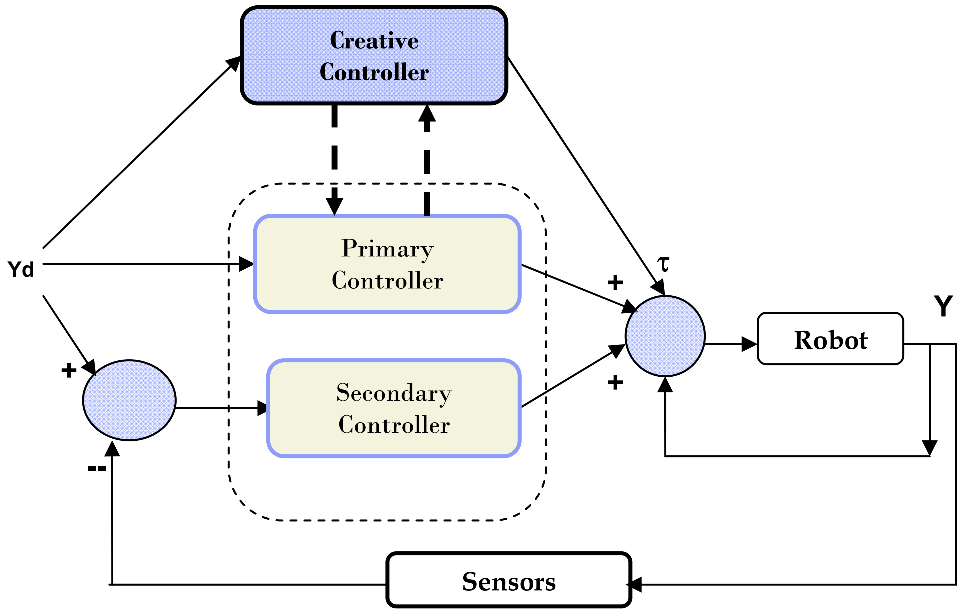


Fig. 4. Block diagram of creative controller

A creative controller is designed to integrate domain knowledge or criteria database and the task control center into the adaptive critic neural network controller. It provides a needed and well-defined structure for autonomous mobile robot application. In effect, it replaces a human doing remote control. We have used the intelligent mobile robot as the test-bed for the creative controller.
The task control center of the creative learning system can be considered hierarchically as follows:

- ❋ Mission for robot – e.g. mobile robot
  - ❋ Task for robot to follow – J : task control
    - ❋ Track for robot to follow
      - ❋ Learn non-linear system model- model discovery
        - ❋ Learn unknown parameters

Adaptive Critic system Implementation

Adaptive Critic system and NN

In order to develop the creative learning algorithm addressed above, we have taken a bottom-up approach to implement adaptive critic controllers by first using neural network for on-line or off-line learning methods. [16] Then the proposed dynamic knowledge database and task control center are added with some to be realized in future research projects.

Tuning algorithm and stability analysis

For linear time invariant systems it is straightforward to examine stability by investigating the poles in the s-plane. However, stability of a nonlinear dynamic systems is much more complex, thus the stability criteria and tests are much more difficult to apply than those for linear time invariant systems[17-19]. For general nonlinear continuous time systems, the state space model is

$$\begin{aligned} \dot{x} &= f[x(t),u(t)] \\ y &= g[x(t),u(t)] \end{aligned} \tag{3}$$

where the nonlinear differential equation is in state variable form, x(t) is the state vector and u(t) is the input and the second equation y(t) is the output of the system.

Creative controller and nonlinear dynamic system

For a creative controller, the task control center and the dynamic database are not time-variable systems; therefore, the adaptive critic learning component determines the stability of the creative controller. As it is discussed in the previous section, the adaptive critic learning is based on critic and action network designs, which are originated from artificial neural network (ANN), thus stability of the system is determined by the stability of the neural networks (NN) or convergence of the critic network and action network training procedure.
The creative controller is a nonlinear system. It is not realistic to explore all the possibilities of the nonlinear systems and prove that the controller is in a stable state. We have used both robot arm manipulators and mobile robot models to examine a large class of problems known as tracking in this study. The objective of tracking is to follow a reference trajectory as closely as possible. This may also be called optimal control since we optimize the tracking error over time.

Critic and Action NN Weights Tuning Algorithm

In adaptive critic learning controller, both the critic network and action network use multilayer NN. Multilayer NN are nonlinear in the weights V and so weight tuning algorithms that yield guaranteed stability and bounded weights in closed-loop feedback systems have been difficult to discover until a few years ago.

3. Some Eclectic Control Scenarios

Urban Rescue Scenarios

Suppose a mobile robot is used for urban rescue as shown in Figure 5. It waits at a start location until a call is received from a command center. Then it must go rescue a person. Since it is in an urban environment, it must use the established roadways. Along the roadways, it can follow pathways. However, at intersections, it must choose between various paths to go to the next block. Therefore, it must use a different criteria at the corners than along the track. The overall goal is to arrive at the rescue site with minimum time. To clarify the situations consider the following steps.

1. Start location – the robot waits at this location until it receives a task command to go to a certain location.
2. Along the path, the robot follows a road marked by lanes. It can use a minimum mean square error between its location and the lane location during this travel.
3. At intersections, the lanes disappear but a database gives a GPS waypoint and the location of the rescue goal.

This example requires the use of both continuous and discrete tracking, a database of known information and multiple criteria optimization. It is possible to add a large number of real-world issues including position estimation, perception, obstacles avoidance, communication, etc.

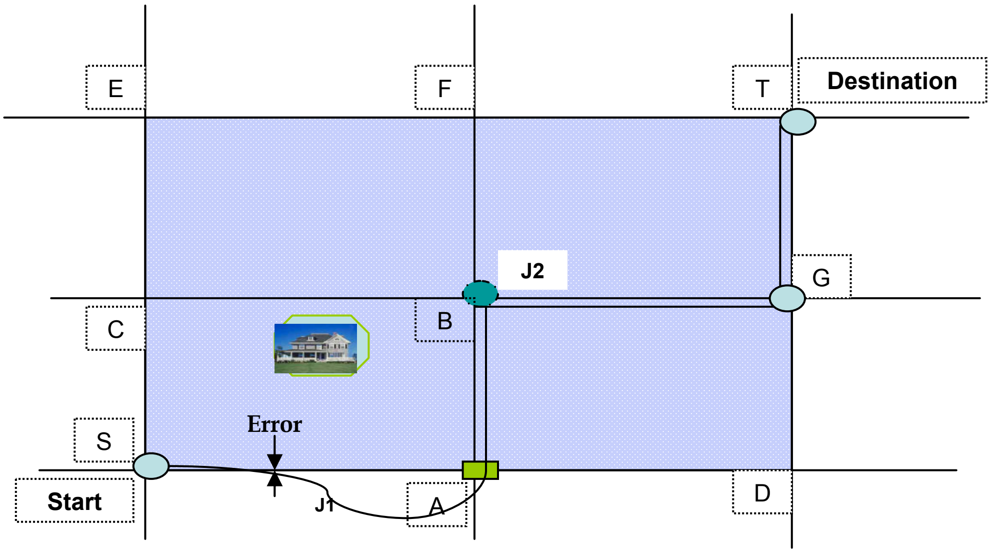


Fig. 5. Simple urban rescue site

In an unstructured environment as shown in Figure 5, we assume that information collected about different potions of the environment could be available to the mobile robot, improving its overall knowledge. As any robot moving autonomously in this environment must have some mechanism for identifying the terrain and estimating the safety of the movement between regions (blocks), it is appropriate for a coordination system to assume that both local obstacle avoidance and a map-building module are available for the robot which is to be controlled. The most important module in this system is the adaptive system to learn about the environment and direct the robot action.[18]
A Global Position System (GPS) may be used to measure the robot position and the distance from the current site to the destination and provide this information to the controller to make its decision on what to do at next move. The GPS system or other sensors could also provides the coordinates of the obstacles for the learning module to learn the map, and then aid in avoiding the obstacles when navigating through the intersections A, B or G, D to destination T.

Task control center
The task control center (TCC) acts a decision-making command center. It takes environmental perception information from sensors and other inputs to the creative controller and derives the criteria functions. We can decompose the robot mission at the urban rescue site shown as Figure 5 into sub-tasks as shown in Figure 6. Moving the robot between the intersections, making decisions is based on control-center-specified criteria functions to minimize the cost of mission. It's appropriate to assume that J1 and J2 are the criteria functions that the task control center will transfer to the learning system at the beginning of the mission from the Start point to Destination (T). J1 is a function of t related to tracking error. J2 is to minimize the distance of the robot from A to T since the cost is directly related to the distance the robot travels.

- From Start (S) to intersection A: robot follow the track SA with the J1 as objective function
- From intersection A to B or D: which one will be the next intersection, the control center takes both J1 and J2 as objective functions.

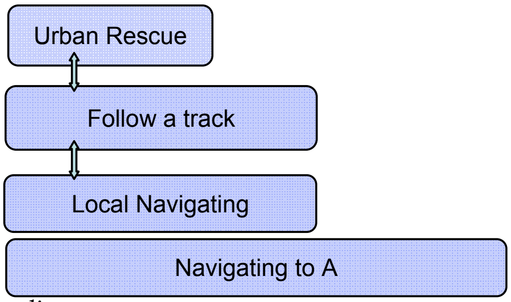


Fig. 6. Mission decomposition diagrams

Dynamic databases
Dynamic databases would store task-oriented environment knowledge, adaptive critic learning parameters and other related information for accomplishing the mission. In this scenario, the robot is commanded to reach a dangerous site to conduct a rescue task. The

dynamic databases saved a copy of the GPS weight points S, A, B, C, D, E, F, G and T. The map for direction and possible obstacle information is also stored in the dynamic databases. A copy of the model parameters can be saved in the dynamic database as shown in the simplified database Figure 7. The action model will be updated in the dynamic database if the current training results are significantly superior to the previous model stored in the database.

Database fields	
Field	**Description**
MODEL_ID	Action model ID
MODEL_NAME	Action model name
UTILITY_FUN	Utility function
CRITERIA_FUN	Criteria function
…	…
Adaptive Critic Training Parameters	
INPUT_CRITIC	Input to critic network
DELT_J	J(t+1)-J(t)
…	…

Fig. 7. Semantic dynamic database structure

Robot Learning Module

Initial plans such as road tracking and robot navigating based on known and assumed information, can be used to incrementally revise the plan as new information is discovered about the environment. The control center will create criteria functions according to the revised information of the world through the user interface. These criteria functions along with other model information of the environment will be input to the learning system. There is a data transfer module from the control center to the learning system as well as a module from the learning system to the dynamic database. New knowledge is used to explore and learn, training according to the knowledge database information and then decide which to store in the dynamic database and how to switch the criteria. The simplest style in the adaptive critic family is heuristic dynamic programming (HDP). This is NN on-line adaptive critic learning. There is one critic network, one action network and one model network in the learning structure. U(t) is the utility function. R is the critic signal as J (criteria function). The learning structure and the parameters are saved a copy in the dynamic database for the system model searching and updating.

Other Demonstrations

The UC Robot Team is attempting to exploit its many years of autonomous ground vehicle research experience to demonstrate its capabilities for designing and fabricating a smart vehicle control for unmanned systems operation as shown in Figures 8, 9 and 10. The purpose of this research is to perform a *proof by demonstration* through system design and integration of a new autonomous vehicle that would integrate advanced technologies in Creative Control with advanced autonomous robotic systems.

The main thrust of our effort is the intelligent control software which provides not only adaptation but also learning and prediction capabilities. However, since a proof by demonstration is needed, further efforts in simulation and implementation are necessary. This new Creative Control has been developed over the past several years and has been the subject of many UC dissertations and papers.

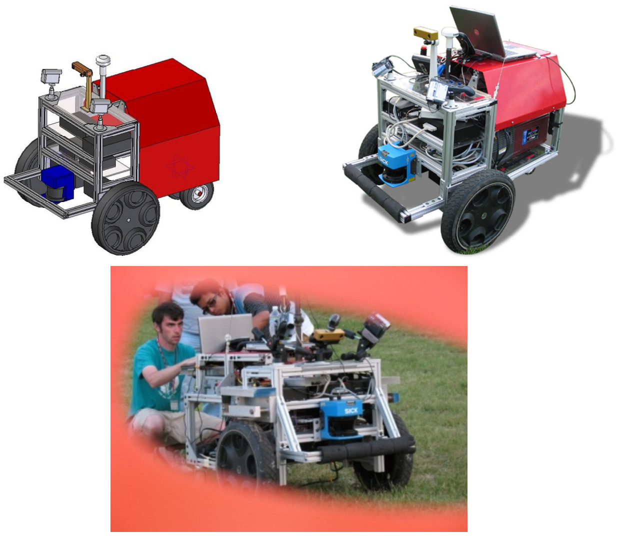


Fig. 8. Bearcat Cub intelligent vehicle designed for IGVC

Fig. 9. NAC Jeep prototype at UC

Fig. 10. Hybrid Vehicle

4. CONCLUSIONS AND RECOMMENDATIONS

The eclectic control is proposed and described as a general perceptual creative adaptive critic learning system. The task control center is a decision-making command center for the intelligent creative learning system. The dynamic knowledge database integrates task control center and adaptive critic learning algorithm into one system. It also provides a knowledge domain for the task command center to perform decision-making. Furthermore, creative learning can be used to explore complex and unpredictable environments, and even permit the discovery of unknown problems. By learning the domain knowledge, the system should be able to obtain the global optima and escape local optima. The challenge is now in implementing such concepts in practical applications.

5. REFERENCES

Bertsekas, D. P., Dynamic Programming and Optimal Control, Vol. I, Second Edition, Athena Scientific, Belmont, MA, 2000, pp. 2, 364.

Brumitt, B.L., A Mission Planning System for Multiple Mobile Robots in Unknown, Unstructured, and Changing Environments. 1998, Carnegie Mellon University.

Campos, J., and F.L. Lewis. Adaptive Critic Neural Network for Feedforward Compensation. in American Control Conference, 1999.

Cao, P.M, „Autonomous Runway Soil Survey System with the Fusion Of Global and Local Navigation Mechanism‟, Ph.D. Dissertation, June 2004.

Ghaffari, M., X. Liao, E. Hall, A Model for the Natural Language Perception-based Creative Control of Unmanned Ground Vehicles. in SPIE Conference Proceedings. 2004.

Hall, E.L. , Ghaffari, M. , Liao, X., Alhaj Ali, S.M. , Sarkar, S., Reynolds, S. and Mathur, K. , "Eclectic Theory of Intelligent Robots," Proc. of Intelligent Robots and Computer Vision, Boston, MA, SPIE 2007

Jaksa, R., and P. Sinc, L*arge Adaptive Critics and Mobile Robotics.* July 2000.

Lewis, F.L., S. Jagannathan, and A. Yesildirek, N*eural Network Control of Robot manipulators and Nonlinear Systems.* 1999, Philadelphia: Taylor and Francis.

Lewis, F.L., D.M. Dawson, and C.T. Abdallah, R*obot Manipulator Control: Theory and Practice.* 2nd Rev&Ex edition ed. 2003: Marcel Dekker (December 1, 2003). 430.

Liao, X., and E. Hall. Beyond Adaptive Critic - Creative Learning for Intelligent Autonomous Mobile Robots. in Intelligent Engineering Systems Through Artificial Neural Networks, ANNIE, in Cooperation with the IEEE Neural Network Council. 2002. St. Louis - Missouri.

Liao, X., et al. Creative Control for Intelligent Autonomous Mobile Robots. in Intelligent Engineering Systems Through Artificial Neural Networks, ANNIE. 2003.

Pang, X. and Werbos, P.J., "Generalized Maze Navigation: SRN Critics Solve What Feedforward or Hebbian Nets Cannot", *Systems, Man, and Cybernetics, IEEE International Conference on*, pp.1764 -1769, v.3, 1996.

Simmons, R., T*ask Control Architecture.* http://www.cs.cmu.edu/afs/cs/project/ TCA/www/ TCA-history.html, 2002.

Stubberud, A.R. and S.C. Stubberud, S*tability,* in H*andbook of Industrial Automation,* R.L. Shell and E.L. Hall, Editors. 2000, MARCEL DEKKER, INC.: New York.

Syam, R. et al. Control of Nonholonomic Mobile Robot by an Adaptive Actor-Critic Method with Simulated Experience Based Value-Functions. in Proc. of the 2002 IEEE International Conference on Robotics and Automation. 2002.

Werbos, P.J. "Tutorial on Neurocontrol, Control Theory and Related Techniques: From Backpropagation to Brain-Like Intelligent Systems," *the Twelfth International Conference on Mathematical and Computer Modelling and Scientific Computing (12th ICMCM & SC)*, http://www.iamcm.org/pwerbos/, 1999.

Systems," *the Twelfth International Conference on Mathematical and Computer Modelling and Scientific Computing (12th ICMCM & SC)*, , 1999.

Werbos, P.J., "Backpropagation and Neurocontrol: a Review and Prospectus," *IJCNN Int Jt Conf Neural Network*, pp.209-216,1989.

White, D. and Sofge, D. Handbook of Intelligent Control, Van Nostrand, 1992

Widrow, B., Gupta, N. and Maitra, S. "Punish/reward: Learning with a Critic in Adaptive Threshold Systems," *IEEE Trans. Systems, Man, Cybemetics*, v.5 pp. 455-465, 1973.

Widrow, B. and Lamego, M.M. N*eurointerfaces.* Control Systems Technology, IEEE Transactions on, 2002. 1**0(**2): p. 221 -228.

Yen, G.G. and Lima, P.G., "Dynamic Database Approach for Fault Tolerant Control Using Dual Heuristic Programming". in Proceedings of the American Control Conference. May 2002.

This new Creative Control has been developed over the past several years and has been the subject of many UC dissertations and papers (Cao, 2004)(Liao et al. 2003) (Hall, et al, 2007)

Enhanced stiffness modeling of serial manipulators with passive joints

Anatol Pashkevich[1,2], Alexandr Klimchik[1,2] and Damien Chablat[2]
[1]*Ecole des Mines de Nantes*
[2]*Institut de Recherches en Communications et Cybernetique de Nantes*
France

Abstract
The chapter focuses on the enhanced stiffness modeling and analysis of serial kinematic chains with passive joints, which are widely used in parallel robotic systems. In contrast to previous works, the stiffness is evaluated for the loaded working mode corresponding to the static equilibrium of the elastic forces and the external wrench acting upon the manipulator end point. It is assumed that the manipulator elasticity is described by a multidimensional lumped-parameter model, which consists of a chain of rigid bodies connected by 6-dof virtual springs. Each of these springs characterize flexibility of the corresponding link or actuating joint and takes into account both their translational/rotational compliance and the coupling between them. The proposed technique allows finding the full-scale "load-deflection" relation for any given workspace point and to linearise it taking into account variation of the manipulator Jacobian due to the external load. These enable evaluating critical forces that may provoke non-linear behavior of the manipulator, such as sudden failure due to elastic instability (buckling). The advantages of the developed technique are illustrated by several examples that deal with kinematic chains employed in typical parallel manipulators.

Keywords
Stiffness model, external loading, kinetostatic analysis, passive joints, buckling, divergence of equilibrium, static stability

1. Introduction

Due to the increasing industrial needs, novel approaches in mechanical design of robotic manipulators are targeted at essential reduction of moving masses and achieving high dynamic performances with relatively low energy consumption. This motivates using advanced kinematical architectures and light-weight materials, as well as minimization of the cross-sections of all manipulator elements (Siciliano & Khatib, 2008). The primary constraint for such minimization is the mechanical stiffness of the manipulator, which must be evaluated taking into account external disturbances (loading) imposed by a relevant manufacturing process. However, in robotic literature, the manipulator stiffness is usually evaluated by a linear model, which defines the static response to the external force/torque, assuming that the compliant deflections are small and the external loading is insignificant (Zhang et al., 2009; Majou et al., 2007). At the same time, in many practical applications (such as milling, for instance), the loading is essential and conventional stiffness modeling techniques must be used with great caution (Los et al., 2008). Moreover, for the manipulators with light-weight links, there is a potential danger of buckling phenomena that is known from general theory of elastic stability (Timoshenko & Goodier, 1970). Hence, the existing stiffness modeling techniques for high-performance robotic manipulators must be revised and enhanced, in order to add ability of detecting non-linear effects and avoid structural failures caused by the loading.

The existing approaches for the manipulator stiffness modeling may be roughly divided into three main groups: the Finite Element Analysis (FEA) (Piras et al., 2005; Hu et al., 2007; Nagai & Liu 2007), the matrix structural analysis (SMA) (Deblaise et al. 2006, Martin, 1966, Li et al., 2002), and the virtual joint method (VJM) that is often called the lumped modeling (Gosselin, 1990; Pashkevich et. al. 2008; Quennouelle & Gosselin 2008 a). The most accurate of them is the Finite Element Analysis, which allows modeling links and joints with its true dimension and shape. However it is usually applied at the final design stage because of the high computational expenses required for the repeated remeshing of the complicated 3D structure over the whole workspace. The SMA also incorporates the main ideas of the FEA, but operates with rather large elements – 3D flexible beams that are presented in the manipulator structure. This leads obviously to the reduction of the computational expenses, but does not provide clear physical relations required for the parametric stiffness analysis. And finally, the VJM method is based on the expansion of the traditional rigid model by adding the virtual joints (localized springs), which describe the elastic deformations of the links, joints and actuators (Salisbury, 1980; Gosselin, 1990). The VJM technique is widely used at the pre-design stage and will be extended in this paper for the case of the preloaded manipulators.

It should be noted, that there are a number of variations and simplifications of the VJM, which differ in modeling assumptions and numerical procedures. Recent modification of this method allows to extend it to the over-constrained manipulator and to apply it at any workspace point, including the singular ones (Pashkevich et. al. 2009 a, b). Besides, to take into account real shape of the manipulator components, the stiffness parameters may be evaluated using the FEA modeling. The latter provided the FEA-accuracy throughout the whole workspace without exhaustive remeshing required for the classical FEA.

At present, there is very limited number of publication that directly addressed the problem of the stiffness modeling for loaded manipulators. The most essential results were obtained in (Alici, & Shirinzadeh; 2005; Quennouelle & Gosselin, 2008 b; Kovecses & Angeles, 2007) where the stiffness matrix was computed taking into account the change in the manipulator configuration due to the preloading. However, the problem of finding the corresponding loaded equilibrium was omitted, so the Jacobian and Hessian were computed in a traditional way, i.e. for the neighborhood of the unloaded equilibrium. The latter yielded essential computational simplification but also imposed crucial limitations, not allowing detecting the buckling and other non-liner effects.

This chapter focuses on the stiffness modeling of serial kinematic chains with passive joints, which are widely used in parallel robotic systems. It presents an enhanced solution of the considered problem, taking into account influence of the external force/torque on the manipulator configuration as well as change in the Jacobian due to the external loading. It implements the virtual joint technique that describes the compliance of the manipulator elements by a set of localized six-dimensional springs separated by rigid links and perfect joints. In contrast to previous works, the developed technique allows to obtain the full-scale “load-deflection” relation for any given workspace point and to compute the desired matrix for any manipulator configuration (including singular ones), implicitly taking into account the kinematic redundancy imposed by the passive joints. Besides, it enables designer to evaluate critical forces that may provoke non-linear manipulator behaviour, such as sudden failure due to elastic instability (buckling) which has not been previously studied in robotic literature. Another contribution is a numerical algorithm for computing the loaded equilibrium and its analytical criteria for its stability analysis.

The remainder of the chapter is organized as follows. Section 2 defines the research problem and basic assumptions. In Section 3, it is proposed a numerical algorithm for computing of the loaded static equilibrium and its stability analysis. Section 4 focuses on the stiffness matrix evaluation taking into account external loading and presence of passive joints. Section 5 contains a set of illustrative examples that demonstrate possible nonlinear behavior of loaded serial kinematic chains. And finally, Section 6 summarizes the main results and contributions.

2. Problem of Stiffness modelling

2.1 Manipulator Architecture

Let us consider a general serial kinematic chain, which consists of a fixed “Base”, a number of flexible actuated joints “Ac”, a serial chain of flexible “Links”, a number of passive joints “Ps” and a moving “Platform” at the end of the chain (Fig. 1). It is assumed that all links are separated by the joints (actuated or passive, rotational or translational) and the joint type order is arbitrary. Besides, it is admitted that some links may be separated by actuated and passive joints simultaneously. Such architecture can be found in most of parallel manipulators (Fig. 2) where several similar kinematic chains are connected to the same base and platform in a different way (with rotation of 90° or 120°, for instance), in order to eliminate the redundancy caused by the passive joints. It is obvious that such kinematic chains are statically *under-constrained* and their stiffness analysis can not be performed by direct application of the standard methods.

Typical examples of the examined kinematic chains can be found in 3-PUU translational parallel kinematic machine (Li & Xu, 2008), in Delta parallel robot (Clavel, 1988) or in parallel manipulators of the Orthoglide family (Chablat & Wenger, 2003) and other manipulators (Merlet, 2006). It worth mentioning that here a specific spatial arrangement of under-constrained chains yields the *over-constrained* mechanism that posses a high structural rigidity with respect to the external force. In particular, for Orthoglide, each kinematic chain prevents the platform from rotating about two orthogonal axes and any combination of two kinematic chains suppresses all possible rotations of the platform. Hence, the whole set of three kinematic chains produces non-singular stiffness matrix while for each separate chain the stiffness matrix is singular. This motivates development of dedicated stiffness analysis techniques that are presented below.

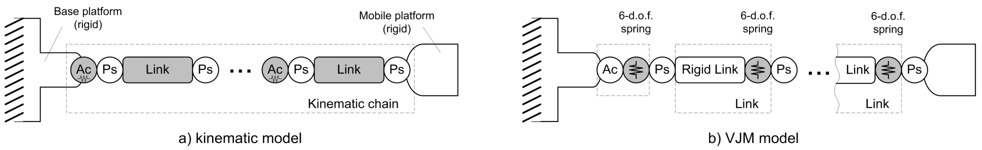


Fig. 1. General serial kinematic chain and its VJM model (Ac – actuated joint, Ps – passive joint).

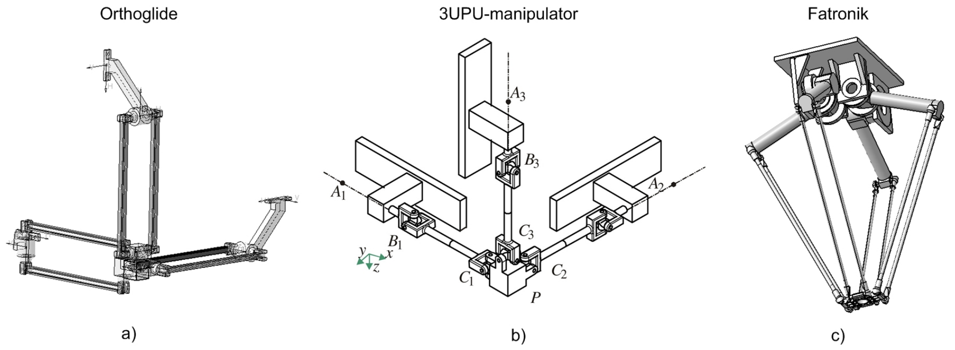


Fig. 2. Architecture of typical parallel manipulators and their kinematics chains

2.2 Basic Assumptions

To evaluate the stiffness of the considered serial manipulator, let us apply a modification of the virtual joint method (VJM), which is based on the lump modeling approach (Gosselin, 1990). According to this approach, the original rigid model should be extended by adding virtual joints (localized springs), which describe elastic deformations of the links. Besides, virtual springs are included in the actuating joints, to take into account the stiffness of the control loop. Under such assumptions, the kinematic chain can be described by the following serial structure:

(a) a rigid link between the manipulator base and the first actuating joint described by the constant homogenous transformation matrix $\mathbf{T}_{Base}$;

(b) the 6-d.o.f. actuating joints defining three translational and three rotational actuator coordinates, which are described by the homogenous matrix function $\mathbf{T}_{3D}\left(\boldsymbol{\theta}_a^i\right)$ where $\boldsymbol{\theta}_a^i = \left(\theta_x^{ai}, \theta_y^{ai}, \theta_z^{ai}, \theta_{\varphi x}^{ai}, \theta_{\varphi y}^{ai}, \theta_{\varphi z}^{ai}\right)$ are the virtual spring coordinates;

(c) the 6-d.o.f. passive joints defining three translational and three rotational passive joins coordinates, which are described by the homogenous matrix function $\mathbf{T}_{3D}\left(\mathbf{q}_p^i\right)$ where $\mathbf{q}_p^i = \left(q_x^i, q_y^i, q_z^i, q_{\varphi x}^i, q_{\varphi y}^i, q_{\varphi z}^i\right)$ are the passive joint coordinates;

(d) the rigid links, which are described by the constant homogenous transformation matrix $\mathbf{T}_{Link}^i$;

(e) a 6-d.o.f. virtual joint defining three translational and three rotational link-springs, which are described by the homogenous matrix function $\mathbf{T}_{3D}\left(\boldsymbol{\theta}_{Link}^i\right)$, where

$\boldsymbol{\theta}^i_{Link} = \left(\theta^i_x, \theta^i_y, \theta^i_z, \theta^i_{\varphi x}, \theta^i_{\varphi y}, \theta^i_{\varphi z}\right)$, $\left(\theta^i_x, \theta^i_y, \theta^i_z\right)$ and $\left(\theta^i_{\varphi x}, \theta^i_{\varphi y}, \theta^i_{\varphi z}\right)$ correspond to the elementary translations and rotations respectively;

(f) a rigid link from the last link to the end-effector, described by the homogenous matrix transformation $\mathbf{T}_{Tool}$.

In the frame of these notations, the final expression defining the end-effector location subject to variations of all joint coordinates of a single kinematic chain may be written as the product of the following homogenous matrices

$$\mathbf{T} = \mathbf{T}_{Base} \cdot \prod_i \left(\mathbf{T}_{3D}\left(\boldsymbol{\theta}^i_a\right) \cdot \mathbf{T}_{3D}\left(\mathbf{q}^{2i-1}_p\right) \cdot \mathbf{T}^i_{Link} \cdot \mathbf{T}_{3D}\left(\boldsymbol{\theta}^i_{Link}\right) \cdot \mathbf{T}_{3D}\left(\mathbf{q}^{2i}_p\right)\right) \cdot \mathbf{T}_{Tool} \tag{1}$$

where the components $\mathbf{T}_{Base}, \mathbf{T}_{3D}(...), \mathbf{T}^i_{Link}, \mathbf{T}_{Tool}$ may be factorized with respect to the terms including the joint variables, in order to simplify computing of the derivatives (Jacobian and Hessian) .

This expression includes both traditional geometric variables (passive and active joint coordinates) and stiffness variables (virtual joint coordinates). Explicit position and orientation of the end-effector can by extracted from the matrix $\mathbf{T}$ in a standard way (Angeles, 2007) , so finally the kinematic model can be rewritten as the vector function

$$t = g(\mathbf{q}, \boldsymbol{\theta}) \tag{2}$$

where the vector $\mathbf{t} = (\mathbf{p}, \boldsymbol{\varphi})^T$ includes the position $\mathbf{p} = (x, y, z)^T$ and orientation $\boldsymbol{\varphi} = (\varphi_x, \varphi_y, \varphi_z)^T$ of the end-platform, the vector $\mathbf{q} = (q_1, q_2, ..., q_n)^T$ contains all passive joint coordinates, the vector $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_m)^T$ collects all virtual joint coordinates, n is the number of passive joins, m is the number of virtual joints.

2.3 Problem statement

In general, the stiffness model describes the resistance of an elastic body or a mechanism to deformations caused by an external force or torque. It can be defined by the relation $\mathbf{F} = f(\Delta\mathbf{t})$, where $f(...)$ is the function that associates a deformation $\Delta\mathbf{t}$ with an external force $\mathbf{F}$ that causes it. It worth mentioning that the function $f(...)$ can de determined even for the singular configurations (or redundant kinematics) while the inverse statement is not generally true. For relatively small deformations, this function is defined through the "stiffness matrix" $\mathbf{K}$, which defines the linear relation

$$\mathbf{F} = \mathbf{K}(\mathbf{q}_0, \boldsymbol{\theta}_0) \cdot \Delta\mathbf{t} \tag{3}$$

between the six-dimensional translational/rotational displacements $\Delta\mathbf{t} = (\Delta x, \Delta y, \Delta z, \Delta\varphi_x, \Delta\varphi_y, \Delta\varphi_z)^T$, and the static forces/torques $\mathbf{F} = \left(F_x, F_y, F_z, M_x, M_y, M_z\right)$ causing this transition. Here, the vector $\mathbf{q}_0 = (q_{01}, q_{02}, ..., q_{0n})^T$ includes all passive joint coordinates, the vector $\boldsymbol{\theta}_0 = (\theta_{01}, \theta_{02}, ..., \theta_{0m})^T$ collects all virtual joint coordinates, n is the number of passive joins, m is the number of virtual joints. Usually, the manipulator is assembled without internal preloading, so the vector $\boldsymbol{\theta}_0$ is equal to zero.

However, for the loaded mode, similar relation is defined in the neighborhood of another static equilibrium, which corresponds to a different manipulator configuration $(\mathbf{q},\boldsymbol{\theta})$, that is caused by external forces/torques $\mathbf{F}$. Respectively, in this case, the stiffness model describes the relation between the increments of the force $\boldsymbol{\delta}\mathbf{F}$ and the position $\boldsymbol{\delta}\mathbf{t}$

$$\boldsymbol{\delta}\mathbf{F} = \mathbf{K}^F(\mathbf{q},\boldsymbol{\theta}) \cdot \boldsymbol{\delta}\mathbf{t} \tag{4}$$

where $\mathbf{q} = \mathbf{q}_0 + \boldsymbol{\Delta}\mathbf{q}$ and $\boldsymbol{\theta} = \boldsymbol{\theta}_0 + \boldsymbol{\Delta\theta}$ denote the new configuration of the manipulator, and $\boldsymbol{\Delta}\mathbf{q}$, $\boldsymbol{\Delta\theta}$ are the deviations of the passive joint and virtual spring coordinates respectively.

Hence, the considered problem may be divided into three sequential subtasks: (i) finding the static equilibrium for the loaded configuration and checking its stability, (ii) linearization of the relevant force/position relations in the neighborhood of this equilibrium, and finally (iii) determining the critical force for the kinematic chain that may cause undesired buckling phenomena.

3. Static equilibrium for loaded mode

Computing of the static equilibrium is a key issue for the stiffness analysis, since it defines the manipulator configuration $(\mathbf{q},\boldsymbol{\theta})$ required for the linearization of the "load-deflection" relation. In previous works, this issue was usually ignored and the linearization was performed in the neighborhood of the unloaded configuration assuming that the external load is small enough. It is obvious that the latter essentially limits relevant results and do not allow to detect non-linear effects such as the buckling. From mathematical point of view, the problem is reduced to finding solutions of a system of non-linear equations that may be unique or non-unique, stable or unstable.

3.1 Configuration of loaded manipulator

Let us assume that, due to the external force $\mathbf{F}$, the end-effector of the manipulator is relocated from the initial (unloaded) position $\mathbf{t}_0 = g(\mathbf{q}_0, \boldsymbol{\theta}_0)$ to a new position $\mathbf{t} = g(\mathbf{q}, \boldsymbol{\theta})$, which satisfies the condition of the mechanical equilibrium. Here $\mathbf{q}_0$ is computed via the inverse kinematics and $\boldsymbol{\theta}_0$ is equal to zero (since there are no external loading in the springs), $\mathbf{q}, \boldsymbol{\theta}$ are passive and virtual joint coordinate in the loaded mode respectively. For rather small displacement $\boldsymbol{\Delta}\mathbf{t} = \mathbf{t} - \mathbf{t}_0$, a new position of the end-effector $\mathbf{t} = P(\mathbf{q}_0 + \boldsymbol{\Delta}\mathbf{q}, \boldsymbol{\theta}_0 + \boldsymbol{\Delta\theta})$ may be expressed as

$$\mathbf{t} = \mathbf{t}_0 + \mathbf{J}_\theta \cdot \boldsymbol{\Delta\theta} + \mathbf{J}_q \cdot \boldsymbol{\Delta}\mathbf{q} \tag{5}$$

where $\mathbf{J}_\theta$ and $\mathbf{J}_q$ are the kinematic Jacobians with respect to the coordinates $\boldsymbol{\theta}$, $\mathbf{q}$, which may be computed from (1), (2) analytically or semi-analytically, using the factorization technique. However, in general case, the model is highly non-linear and computing $\mathbf{J}_\theta$ and $\mathbf{J}_q$ requires some additional efforts.

For computational reasons, let us consider the dual problem that deals with determining the external force $\mathbf{F}$ and the manipulator configuration $(\mathbf{q}, \boldsymbol{\theta})$ that correspond to the output position $\mathbf{t}$.

Let us assume that the joints are given small, arbitrary virtual displacements $\delta\mathbf{q}, \delta\boldsymbol{\theta}$ in the equilibrium neighborhood.

According to the principle of virtual displacements, the virtual work of the external force $\mathbf{F}$ applied to the end-effector along the corresponding displacement $\delta\mathbf{t} = \mathbf{J}_\theta \cdot \delta\boldsymbol{\theta} + \mathbf{J}_q \cdot \delta\mathbf{q}$ is equal to the sum $\left(\mathbf{F}^T\mathbf{J}_\theta\right)\cdot\delta\boldsymbol{\theta} + \left(\mathbf{F}^T\mathbf{J}_q\right)\cdot\delta\mathbf{q}$. Since the passive joints do not produce the force/torque reactions, the virtual work includes only one component $-\boldsymbol{\tau}_\theta{}^T \cdot \delta\boldsymbol{\theta}$ (the minus sign takes into account the force-displacement directions for the virtual spring). In the static equilibrium, the total virtual work of all forces is equal to zero for any virtual displacement, therefore the equilibrium conditions may be written as

$$\mathbf{J}_\theta{}^T \cdot \mathbf{F} = \boldsymbol{\tau}_\theta; \qquad \mathbf{J}_q{}^T \cdot \mathbf{F} = \mathbf{0} \tag{6}$$

Taking into account (3), the latter system of equations can be rewritten as

$$\mathbf{F}^T \cdot \mathbf{J}_\theta = \mathbf{K}_\theta \cdot \boldsymbol{\theta}; \qquad \mathbf{F}^T \cdot \mathbf{J}_q = 0 \tag{7}$$

It is evident that there is no general method for analytical solution of this system and it is required to apply numerical techniques. To derive the numerical algorithm, let us linearize the kinematic equation in the neighborhood of the current position $(\mathbf{q}_i, \boldsymbol{\theta}_i)$

$$\mathbf{t} = P(\mathbf{q}_i, \boldsymbol{\theta}_i) + \mathbf{J}_q(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot(\mathbf{q}_{i+1} - \mathbf{q}_i) + \mathbf{J}_\theta(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot(\boldsymbol{\theta}_{i+1} - \boldsymbol{\theta}_i) \tag{8}$$

and rewrite the static equilibrium equations as

$$\mathbf{J}_\theta{}^T(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{F}_{i+1} = \mathbf{K}_\theta\, \boldsymbol{\theta}_{i+1}; \qquad \mathbf{J}_q{}^T(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{F}_{i+1} = \mathbf{0} \tag{9}$$

This leads to a linear algebraic system of equations with respect to $(\mathbf{q}_{i+1}, \boldsymbol{\theta}_{i+1}, \mathbf{F}_{i+1})$

$$\begin{aligned}
&\mathbf{J}_q(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{q}_{i+1} + \mathbf{J}_\theta(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\boldsymbol{\theta}_{i+1} = \mathbf{t} - \mathbf{f}(\mathbf{q}_i, \boldsymbol{\theta}_i) + \mathbf{J}_q(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{q}_i + \mathbf{J}_\theta(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\boldsymbol{\theta}_i \\
&-\mathbf{K}_\theta\cdot\boldsymbol{\theta}_{i+1} + \mathbf{J}_\theta{}^T(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{F}_{i+1} = \mathbf{0} \\
&\mathbf{J}_q{}^T(\mathbf{q}_i, \boldsymbol{\theta}_i)\cdot\mathbf{F}_{i+1} = \mathbf{0}
\end{aligned} \tag{10}$$

which gives the following iterative scheme

$$\begin{bmatrix}\mathbf{F}_{i+1}\\ \mathbf{q}_{i+1}\end{bmatrix}=\begin{bmatrix}\mathbf{J}_\theta(\mathbf{q}_i,\boldsymbol{\theta}_i)\mathbf{K}_\theta^{-1}\mathbf{J}_\theta^T(\mathbf{q}_i,\boldsymbol{\theta}_i) & \mathbf{J}_q(\mathbf{q}_i,\boldsymbol{\theta}_i)\\ \mathbf{J}_q^T(\mathbf{q}_i,\boldsymbol{\theta}_i) & 0\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{t}-\mathbf{f}(\mathbf{q}_i,\boldsymbol{\theta}_i)+\mathbf{J}_q(\mathbf{q}_i,\boldsymbol{\theta}_i)\mathbf{q}_i+\mathbf{J}_\theta(\mathbf{q}_i,\boldsymbol{\theta}_i)\boldsymbol{\theta}_i\\ 0\end{bmatrix}$$
$$\boldsymbol{\theta}_{i+1}=\mathbf{K}_\theta^{-1}\cdot\mathbf{J}_\theta^T(\mathbf{q}_i,\boldsymbol{\theta}_i)\cdot\mathbf{F}_{i+1} \tag{11}$$

where the starting point $(\mathbf{q}_0,\boldsymbol{\theta}_0)$ can be chosen using the non-loaded configuration, and computed via the inverse kinematics.
As follows from computational experiments, for typical values of deformations the proposed iterative algorithm possesses rather good convergence (3-5 iterations are usually enough). However, in the case of buckling or in the area of multiple equilibriums, the problem of convergence becomes rather critical and highly depends on the initial guess. To overcome this problem, the value of the joint variables $(\boldsymbol{\theta}_i,\mathbf{q}_i)$ computed at each iteration were disturbed by adding small random noise. Further enhancement of this algorithm may be based on the full-scale Newton-Raphson technique (i.e. linearization of the static equilibrium equations in addition to the kinematic one), this obviously increases computational expenses but potentially improves convergence.

3.2 Stability of the static equilibrium

To evaluate stability of the computed static equilibrium $(\mathbf{q},\boldsymbol{\theta})$, let us assume that the manipulator end-effector is fixed at the point $\mathbf{p}$ corresponding to the external load $\mathbf{F}$, but the joint coordinates are given small virtual displacements $\delta\mathbf{q}$, $\delta\boldsymbol{\theta}$ satisfying the geometrical constraint (2), i.e.

$$\mathbf{p}=\mathbf{g}(\mathbf{q},\boldsymbol{\theta});\qquad \mathbf{p}=\mathbf{g}(\mathbf{q}+\delta\mathbf{q},\boldsymbol{\theta}+\delta\boldsymbol{\theta}) \tag{12}$$

For these assumptions, let us compute the total virtual work in the joints that must be positive for a stable equilibrium and negative for an unstable one.
To achieve the virtual configuration $(\mathbf{q}+\delta\mathbf{q},\boldsymbol{\theta}+\delta\boldsymbol{\theta})$ and restore the equilibrium conditions, each of the joints must include virtual motors that generate the generalized forces/torques $\delta\boldsymbol{\tau}_q$, $\delta\boldsymbol{\tau}_\theta$ which satisfies the equations:

$$\begin{aligned}&\mathbf{J}_\theta^T\,\mathbf{F}=\mathbf{K}_\theta\,\boldsymbol{\theta}; &&(\mathbf{J}_\theta+\delta\mathbf{J}_\theta)^T\mathbf{F}=\mathbf{K}_\theta(\boldsymbol{\theta}+\delta\boldsymbol{\theta})+\delta\boldsymbol{\tau}_\theta\\ &\mathbf{J}_q^T\,\mathbf{F}=0; &&(\mathbf{J}_q+\delta\mathbf{J}_q)^T\mathbf{F}=\delta\boldsymbol{\tau}_q\end{aligned} \tag{13}$$

After relevant transformations, the virtual torques may be expressed as

$$\delta\boldsymbol{\tau}_\theta=\delta(\mathbf{J}_\theta^T\,\mathbf{F})-\mathbf{K}_\theta\,\delta\boldsymbol{\theta};\qquad \delta\boldsymbol{\tau}_q=\delta(\mathbf{J}_q^T\,\mathbf{F}) \tag{14}$$

where $\delta(.)$ denotes the differential with respect to $\delta\mathbf{q}$, $\delta\boldsymbol{\theta}$ that may be expanded via the Hessians of the scalar function $\Psi=\mathbf{g}(\mathbf{q},\,\boldsymbol{\theta})^T\,\mathbf{F}$:

$$\delta(\mathbf{J}_\theta^T \mathbf{F}) = \mathbf{H}_{\theta q}^F \, \delta\mathbf{q} + \mathbf{H}_{\theta\theta}^F \, \delta\boldsymbol{\theta}; \qquad \delta(\mathbf{J}_q^T \mathbf{F}) = \mathbf{H}_{qq}^F \, \delta\mathbf{q} + \mathbf{H}_{q\theta}^F \, \delta\boldsymbol{\theta} \tag{15}$$

provided that

$$\mathbf{H}_{qq}^F = \partial^2\Psi / \partial\mathbf{q}^2; \qquad \mathbf{H}_{\theta\theta}^F = \partial^2\Psi / \partial\boldsymbol{\theta}^2; \qquad \mathbf{H}_{q\theta}^F = \mathbf{H}_{\theta q}^F = \partial^2\Psi / \partial\mathbf{q}\,\partial\boldsymbol{\theta} \tag{16}$$

Further, taking into account that the virtual displacement from $(\mathbf{q}, \boldsymbol{\theta})$ to $(\mathbf{q} + \delta\mathbf{q}, \boldsymbol{\theta} + \delta\boldsymbol{\theta})$ leads to a gradual change of the virtual torques from $(\mathbf{0}, \mathbf{0})$ to $(\delta\boldsymbol{\tau}_q, \delta\boldsymbol{\tau}_\theta)$, the virtual work may be computed as a half of the corresponding scalar products

$$\delta W = -\frac{1}{2}\left(\delta\boldsymbol{\tau}_\theta^T \, \delta\boldsymbol{\theta} + \delta\boldsymbol{\tau}_q^T \, \delta\mathbf{q}\right), \tag{17}$$

where the minus sign takes into account the adopted conventions for the positive directions of the forces and displacements. Hence, after appropriate substitutions and transforming to the matrix form, the desired stability condition may be written as

$$\delta W = -\frac{1}{2}\begin{bmatrix} \delta\boldsymbol{\theta}^T & \delta\mathbf{q}^T \end{bmatrix} \begin{bmatrix} \mathbf{H}_{\theta\theta}^F - \mathbf{K}_\theta & \mathbf{H}_{q\theta}^F \\ \mathbf{H}_{\theta q}^F & \mathbf{H}_{qq}^F \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\theta} \\ \delta\mathbf{q} \end{bmatrix} > 0 \tag{18}$$

where $\delta\mathbf{q}$ and $\delta\boldsymbol{\theta}$ must satisfy to the geometrical constraints (12).
In order to take into account the relation between $\delta\mathbf{q}$ and $\delta\boldsymbol{\theta}$ that is imposed by (12), let us apply the first-order expansion of the function $\mathbf{g}(\boldsymbol{\theta}, \mathbf{q})$ that yields the following linear relation

$$\begin{bmatrix} \mathbf{J}_\theta & \mathbf{J}_q \end{bmatrix} \cdot \begin{bmatrix} \delta\boldsymbol{\theta} \\ \delta\mathbf{q} \end{bmatrix} = \mathbf{0}\,. \tag{19}$$

Then, applying the SVD- factorization (Strang, 1998) of the integrated Jacobian

$$\begin{bmatrix} \mathbf{J}_\theta & \mathbf{J}_q \end{bmatrix} = \begin{bmatrix} \mathbf{U}_\theta & \mathbf{U}_q \end{bmatrix} \cdot \begin{bmatrix} \mathbf{S}_r & \\ & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_\theta^T \\ \mathbf{V}_q^T \end{bmatrix} \tag{20}$$

and extracting from $\mathbf{V}_\theta$, $\mathbf{V}_q$ the sub-matrices $\mathbf{V}_\theta^{\circ}$, $\mathbf{V}_q^{\circ}$ corresponding to the zero singular values, a relevant null-space of the system (19) may be presented as

$$\delta\boldsymbol{\theta} = \mathbf{V}_\theta^{\circ} \cdot \delta\boldsymbol{\mu}; \qquad \delta\mathbf{q} = \mathbf{V}_q^{\circ} \cdot \delta\boldsymbol{\mu} \tag{21}$$

where $\delta\boldsymbol{\mu}$ is the arbitrary vector of the appropriate dimension (equal to the rank-deficiency of the Integrated Jacobian). Hence, the stability condition (18) may be rewritten as inequality

$$\delta W = -\frac{1}{2}\delta\boldsymbol{\mu}^T \cdot \begin{bmatrix} \mathbf{V}_\theta^o \\ \mathbf{V}_q^o \end{bmatrix}^T \cdot \begin{bmatrix} \mathbf{H}_{\theta\theta}^F - \mathbf{K}_\theta & \mathbf{H}_{q\theta}^F \\ \mathbf{H}_{\theta q}^F & \mathbf{H}_{qq}^F \end{bmatrix} \cdot \begin{bmatrix} \mathbf{V}_\theta^o \\ \mathbf{V}_\theta^o \end{bmatrix} \cdot \delta\boldsymbol{\mu} > 0 \tag{22}$$

that must be satisfied for all non-zero $\delta\boldsymbol{\mu}$. In other words, the considered static equilibrium $(\mathbf{q}, \boldsymbol{\theta})$ is stable if (and only if) the matrix

$$\begin{bmatrix} \mathbf{V}_\theta^o \\ \mathbf{V}_q^o \end{bmatrix}^T \cdot \begin{bmatrix} \mathbf{H}_{\theta\theta}^F - \mathbf{K}_\theta & \mathbf{H}_{q\theta}^F \\ \mathbf{H}_{\theta q}^F & \mathbf{H}_{qq}^F \end{bmatrix} \cdot \begin{bmatrix} \mathbf{V}_\theta^o \\ \mathbf{V}_\theta^o \end{bmatrix} < 0 \tag{23}$$

is positive-negative. It is worth mentioning that the obtained result is in a good agreement with previous studies (Alici & Shirinzadeh, 2005), where (for manipulators without passive joints) the stiffness properties were defined by the matrix $\mathbf{K}_\theta - \mathbf{H}_{\theta\theta}^F$ that must be positive-definite.

4. Stiffness model for the loaded mode

The previous section presents a technique that allows obtaining an exact relation between the elastic deformations and corresponding external force/torque. It is based on sequential computations of loaded equilibriums (and relevant force/torque) for various displacements of the manipulator end-point with respect to its unloaded location. However, in general case, this relation is highly non-linear while common engineering practice operates with the stiffness matrix derived via the linearization.

To compute the desired stiffness matrix, let us consider the neighborhood of the loaded configuration and assume that the external force and the end-effector location are incremented by some small values $\delta\mathbf{F}$, $\delta\mathbf{t}$. Besides, let us assume that a new configuration also satisfies the equilibrium conditions. Hence, it is necessary to consider simultaneously two equilibriums corresponding to the manipulator state variables $(\mathbf{F}, \mathbf{q}, \boldsymbol{\theta}, \mathbf{t})$ and $(\mathbf{F}+\delta\mathbf{F}, \mathbf{q}+\delta\mathbf{q}, \boldsymbol{\theta}+\delta\boldsymbol{\theta}, \mathbf{t}+\delta\mathbf{t})$. Relevant equations of statics may be written as

$$\mathbf{F}\cdot\mathbf{J}_\theta^{\;T} = \mathbf{K}_\theta \cdot \boldsymbol{\theta}; \qquad \mathbf{F}\cdot\mathbf{J}_q^{\;T} = 0 \tag{24}$$

and

$$\begin{aligned} (\mathbf{F}+\delta\mathbf{F})\cdot(\mathbf{J}_{\boldsymbol{\theta}}+\delta\mathbf{J}_{\boldsymbol{\theta}})^T &= \mathbf{K}_{\boldsymbol{\theta}}\cdot(\boldsymbol{\theta}+\delta\boldsymbol{\theta}); \\ (\mathbf{F}+\delta\mathbf{F})\cdot(\mathbf{J}_{\mathbf{q}}+\delta\mathbf{J}_{\mathbf{q}})^T &= 0 \end{aligned} \tag{25}$$

where $\delta\mathbf{J}_q(\mathbf{q},\boldsymbol{\theta})$ and $\delta\mathbf{J}_\theta(\mathbf{q},\boldsymbol{\theta})$ are the differentials of the Jacobians due to changes in $(\mathbf{q}, \boldsymbol{\theta})$. Besides, in the neighborhood of $(\mathbf{q},\boldsymbol{\theta})$, the kinematic equation may be also presented in the linearized form:

$$\boldsymbol{\delta}\mathbf{t} = \mathbf{J}_{\theta}(\mathbf{q},\, \boldsymbol{\theta}) \cdot \boldsymbol{\delta\theta} + \mathbf{J}_{q}(\mathbf{q},\, \boldsymbol{\theta}) \cdot \boldsymbol{\delta}\mathbf{q}\,, \tag{26}$$

Hence, after neglecting the high-order small terms and expending the differentials via the Hessians of the function $\Psi = \mathbf{g}(\mathbf{q},\boldsymbol{\theta})^T\mathbf{F}$ (similar to sub-section 3.2), equations (24), (25) may be rewritten as

$$\begin{aligned} &\mathbf{J}_{\theta}{}^{T}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\mathbf{F} + \mathbf{H}^{F}_{\theta q}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\mathbf{q} + \mathbf{H}^{F}_{\theta\theta}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\boldsymbol{\theta} = \mathbf{K}_{\theta} \cdot \delta\boldsymbol{\theta} \\ &\mathbf{J}_{q}{}^{T}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\mathbf{F} + \mathbf{H}^{F}_{qq}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\mathbf{q} + \mathbf{H}^{F}_{q\theta}(\mathbf{q},\boldsymbol{\theta}) \cdot \delta\boldsymbol{\theta} = \mathbf{0} \end{aligned} \tag{27}$$

and the general relation between the increments $\delta\mathbf{F}$, $\delta\mathbf{t}$, $\delta\boldsymbol{\theta}$, $\delta\mathbf{q}$ can be presented as

$$\begin{bmatrix} \mathbf{0} & \mathbf{J}_q & \mathbf{J}_\theta \\ \mathbf{J}_q^T & \mathbf{H}^F_{\theta\theta} - \mathbf{K}_\theta & \mathbf{H}^F_{\theta q} \\ \mathbf{J}_\theta^T & \mathbf{H}^F_{q\theta} & \mathbf{H}^F_{qq} \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{F} \\ \delta\boldsymbol{\theta} \\ \delta\mathbf{q} \end{bmatrix} = \begin{bmatrix} \delta\mathbf{t} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{28}$$

The latter gives a straightforward numerical technique for computing of the desired stiffness matrix: direct inversion of the matrix in the left-hand side of (28) and extracting from it the upper-left sub-matrix of size 6×6. Similarly, there can be computed the matrices defining linear relations between the end-effector increment $\delta\mathbf{t}$ and the increments of the joint variables $\delta\boldsymbol{\theta}$, $\delta\mathbf{q}$, i.e.:

$$\delta\mathbf{F} = \mathbf{K}_F \cdot \delta\mathbf{t}; \qquad \delta\boldsymbol{\theta} = \mathbf{K}_\theta \cdot \delta\mathbf{t}; \qquad \delta\mathbf{q} = \mathbf{K}_q \cdot \delta\mathbf{t} \tag{29}$$

where

$$\begin{bmatrix} \mathbf{0} & \mathbf{J}_q & \mathbf{J}_\theta \\ \mathbf{J}_q^T & \mathbf{H}^F_{\theta\theta} - \mathbf{K}_\theta & \mathbf{H}^F_{\theta q} \\ \mathbf{J}_\theta^T & \mathbf{H}^F_{q\theta} & \mathbf{H}^F_{qq} \end{bmatrix}^{-1} = \left[\begin{array}{c:c:c} \mathbf{K}_F & \mathbf{K}_\theta & \mathbf{K}_q \\ \hdashline * & * & * \\ \hdashline * & * & * \end{array}\right] \tag{30}$$

In the case when the above matrix inverse is computationally hard, the variable $\delta\boldsymbol{\theta}$ can be eliminated analytically, using corresponding static equation: $\delta\boldsymbol{\theta} = \mathbf{k}^F_\theta\, \mathbf{J}^T_\theta \cdot \delta\mathbf{F} + \mathbf{k}^F_\theta\, \mathbf{H}^F_{\theta q} \cdot \delta\mathbf{q}$, . where $\mathbf{k}^F_\theta = \left(\mathbf{K}_\theta - \mathbf{H}^{\mathbf{F}}_{\theta\theta}\right)^{-1}$. This leads to a reduced system of matrix equations with unknowns $\delta\mathbf{F}$ and $\delta\mathbf{q}$

$$\begin{bmatrix} \mathbf{J}_\theta \cdot \mathbf{k}^F_\theta \cdot \mathbf{J}_\theta{}^T & \mathbf{J}_{\mathbf{q}} + \mathbf{J}_\theta \cdot \mathbf{k}^F_\theta \cdot \mathbf{H}^F_{\theta q} \\ \mathbf{J}_q{}^T + \mathbf{H}^F_{q\theta} \cdot \mathbf{k}^F_\theta \cdot \mathbf{J}_\theta{}^T & \mathbf{H}^F_{qq} + \mathbf{H}^F_{q\theta} \cdot \mathbf{k}^F_{\boldsymbol{\theta}} \cdot \mathbf{H}^F_{\theta q} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\delta}\mathbf{F} \\ \boldsymbol{\delta}\mathbf{q} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\delta}\mathbf{t} \\ \mathbf{0} \end{bmatrix}. \tag{31}$$

that may be treated in the similar way, i.e. the desired stiffness matrix is also obtained by direct inversion of the matrix in the left-hand side of (31) and extracting from it the upper-left sub-matrix of size 6×6:

$$\begin{bmatrix} \mathbf{J}_{\theta}\mathbf{k}_{\theta}^{F}\mathbf{J}_{\theta}^{T} & \mathbf{J}_{q}+\mathbf{J}_{\theta}\mathbf{k}_{\theta}^{F}\mathbf{H}_{\theta q}^{F} \\ \mathbf{J}_{q}^{T}+\mathbf{H}_{q\theta}^{F}\mathbf{k}_{\theta}^{F}\mathbf{J}_{\theta}^{T} & \mathbf{H}_{qq}^{F}+\mathbf{H}_{q\theta}^{F}\mathbf{k}_{\theta}^{F}\mathbf{H}_{\theta q}^{F} \end{bmatrix}^{-1} = \left[\begin{array}{c:c} \mathbf{K}_{F} & \mathbf{K}_{q} \\ \hdashline * & * \end{array}\right] \qquad (32)$$

It worth mentioning that the structure of the latter matrix is similar to one obtained for the unloaded manipulator in (Pashkevich et al., 2009 c) and differs only by Hessians that take into account influence of the external load. It should be also noted that, because of presence of the passive joints, the stiffness matrix of a separate serial kinematic chain is always singular, but aggregation of all the manipulator chains of a parallel manipulator produce a non-singular stiffness matrix.
Hence, the presented technique allows computing the stiffness matrix in the presence of the external load and to generalize previous results both for serial kinematic chains and for parallel manipulators. It the following Section, it will be applied to several examples that deal with kinematic chains employed in typical parallel manipulators.

5. Illustrative examples

Let us apply the developed technique to the stiffness analysis of a serial kinematic chain consisting of three similar links separated by two similar rotating actuated joints. It is assumed that the chain is a part of a parallel manipulator and it is connected to the robot base via a universal passive joint and the end-platform connection is achieved via a spherical passive joint. In order to investigate possible non-linear effects in the stiffness behavior of such architecture, let us consider several cases that differ in stiffness models of the links and actuated joints.

5.1 Examined models

5.1.1 Manipulator geometry

In general, the geometry of the examined kinematic chain (Fig. 2) can be defined as $U_pR_aR_aS_p$ where R, U and S denote respectively the rotational, universal and spherical joints, and the subscripts '*p*' and '*a*' refer to passive and active joints respectively. Using the homogenous matrix transformations, the chain geometry may be described by the equation

$$\mathbf{T} = \mathbf{R}_u(\mathbf{q}_0)\cdot\mathbf{T}_x(L)\cdot\mathbf{T}_s(\boldsymbol{\theta}_1)\cdot\mathbf{R}_z(q_{a1})\cdot\mathbf{T}_x(L)\cdot\mathbf{T}_s(\boldsymbol{\theta}_2)\cdot\mathbf{R}_z(q_{a2})\cdot\mathbf{T}_x(L)\cdot\mathbf{T}_s(\boldsymbol{\theta}_3)\cdot\mathbf{R}_s(\mathbf{q}_t) \qquad (33)$$

where $\mathbf{R}_z(...)$ and $\mathbf{T}_x(...)$ are the elementary rotation/translation matrices around/along the *z*- and *x*-axes, $\mathbf{R}_u(...)$ is the homogeneous rotation matrix of the universal joint (incorporating two elementary rotations), $\mathbf{R}_S(.)$ is the homogeneous rotation matrix of the universal joint (incorporating three elementary rotations), q_{a1}, q_{a2} are the coordinates of the actuated joints, L is the length of the links, $\mathbf{q}_0$ is the coordinate vector of the universal passive joint located at the robot base, $\mathbf{q}_t$ is the coordinate vector corresponding to the passive spherical joint at the end-platform, $\mathbf{T}_s(.)$ is the homogenous vector-function describing elastic deformations in the links and actuators (they are represented by the virtual coordinates incorporated in the vectors $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3$). It is obvious that this model can be easily transformed into the form $\mathbf{t} = \mathbf{g}(\mathbf{q}, \boldsymbol{\theta})$ used in the frame of the developed technique.

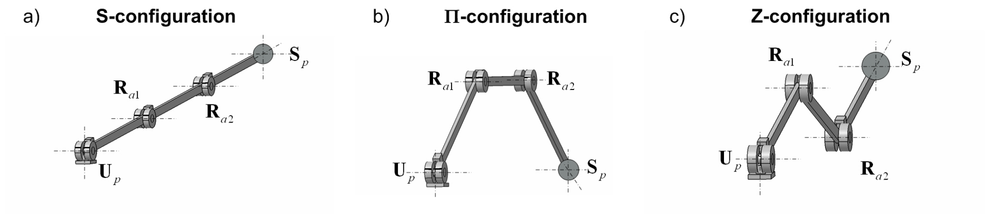


Fig. 3. Examined kinematical chain and its typical configurations (Up – passive universal joint, Ra1, Ra2 – actuated rotating joints, Sp – passive spherical joint)

To investigate particularities of this architecture, let us also define three typical postures that differ in values of the actuated coordinates:

S-configuration: the links are located along the straight line (Fig. 2a),
the actuated coordinates are $q_{a1} = q_{q2} = 0$

Π-configuration: the chain takes a trapezoid shape (Fig. 2b),
the actuated coordinates are $q_{a1} = q_{q2} = -30^\circ$

Z-configuration: the chain takes a zig-zag shape (Fig. 2c),
the actuated coordinates are $q_{a1} = -q_{q2} = 30^\circ$

For presentational convenience, let us also assume that the coordinates $\mathbf{q}_0$ of the universal passive joint are computed to ensure location of the end-effector on the Cartesian axis *x*.
For each of these configurations, let us consider three types of the virtual springs corresponding to different physical assumptions concerning the stiffness properties of the actuators/links. They cover the cases, in which the main flexibility is caused by the torsion in the actuators, by the link bending, and by the combination of elementary deformations of the links.

5.1.2 Case of 1D-springs: Model A

Here, it is assumed that the flexible elements are localized in the actuating drives while the links are considered as strictly rigid. It allows, without loss of generality, to reduce the original $U_pR_aR_aS_p$ model down to $R_pR_aR_aR_p$ and define a single stiffness parameter K_θ (similar for both actuators) that will be used as a reference value for the further analysis. Besides, it is possible to ignore the end-effector orientation and consider a single passive joint coordinate q (at the base) and two virtual joint coordinates θ_1, θ_2 (at actuators). This restricts the end-effector motions to Cartesian *xy*-plane where the geometrical model is defined by equations

$$\begin{aligned} x &= L \cdot \cos q + L \cdot \cos q_{12} + L \cdot \cos q_{13}, \\ y &= L \cdot \sin q + L \cdot \sin q_{12} + L \cdot \sin q_{13} \end{aligned} \tag{34}$$

where $q_{12} = q + \theta_1$ and $q_{13} = q + \theta_1 + \theta_2$. In this case, the Jacobian matrices are also computed easily

$$\mathbf{J}_q = L \cdot \begin{bmatrix} -\sin q - \sin q_{12} - \sin q_{13} \\ \cos q + \cos q_{12} + \cos q_{13} \end{bmatrix}; \quad \mathbf{J}_\theta = L \cdot \begin{bmatrix} -\sin q_{12} - \sin q_{13} & -\sin q_{13} \\ \cos q_{12} + \cos q_{13} & \cos q_{13} \end{bmatrix} \tag{35}$$

and corresponding stiffness analysis will be performed analytically and compared with numerical results that were obtained using the developed methodology.

5.1.3 Case of 2D springs: Model B

For this model, let us assume that the actuators do not include flexible components but the manipulator links are subject to non-negligible deformations in Cartesian xy-plane (bending and compression). Correspondingly, the link flexibility is defined by a 3×3 matrix that includes elements describing deformation in x- and y- directions and rotational deformation with respect to z-axis. Relevant stiffness matrix may be written as (Connor, 1976)

$$\mathbf{K} = \frac{E}{L^3} \cdot \begin{bmatrix} A \cdot L^2 & 0 & 0 \\ 0 & 12 \cdot I & -6 \cdot I \cdot L \\ 0 & -6 \cdot I \cdot L & 4 \cdot I \cdot L^2 \end{bmatrix} \tag{36}$$

where L is the length of the links, I and A are respectively its second moment and area of the cross-section , and E is the Young module. Further, for comparison purposes, let us re-parameterize this matrix $\boldsymbol{K}$ to be closer to model A. In particular, let us denote the element $k_{3,3}$ (corresponding to z-rotation) of the compliant matrix $\mathbf{k} = \mathbf{K}^{-1}$ as $1/K_\theta$ and eliminate the Young module. This yields expression

$$\mathbf{k} = \frac{1}{K_\theta} \cdot \begin{bmatrix} I/A & 0 & 0 \\ 0 & L^2/3 & L/2 \\ 0 & L/2 & 1 \end{bmatrix} \tag{37}$$

where, for a rectangular cross-section $a \times b$, the required parameters may be computed as $A = ab$ and $I = ab^3/12$.

From kinematical point of view, model B is also restricted to Cartesian *xy*-plane and is described by the expression $R_pR_aR_aR_p$. However, in addition to a single passive joint coordinate q , here there are nine coordinates of the virtual spring (three for each link). The kinematic model of this manipulator is defined by equations

$$\begin{aligned} x &= L_1 \cdot \cos q + L_2 \cdot \cos q_{12} + \theta_2 \cdot \sin q_{12} + L_3 \cdot \cos q_{13} + \theta_5 \cdot \sin q_{13} + L_4 \cdot \cos q_{14} + \theta_8 \cdot \sin q_{14}, \\ y &= L_1 \cdot \sin q + L_2 \cdot \sin q_{12} + \theta_2 \cdot \cos q_{12} + L_3 \cdot \sin q_{13} + \theta_5 \cdot \cos q_{13} + L_4 \cdot \sin q_{14} + \theta_8 \cdot \cos q_{14} \end{aligned} \tag{38}$$

where $L_1 = L$, $L_2 = L + \theta_1$, $L_3 = L + \theta_4$, $L_4 = \theta_7$, $q_{12} = q + \theta_3$, $q_{13} = q + \theta_3 + \theta_6$, $q_{14} = q + \theta_3 + \theta_6 + \theta_9$, and $\boldsymbol{\theta}_1 = (\theta_1, \theta_2, \theta_3)$, $\boldsymbol{\theta}_2 = (\theta_4, \theta_5, \theta_6)$, $\boldsymbol{\theta}_3 = (\theta_7, \theta_8, \theta_9)$ are the spring joint coordinates for the first, second and third links respectively. The Jacobian matrices in this case can be also computed analytically but their dimensions are too high for analytical computations. Hence, in this case this stiffness analysis will be performed numerically.

5.1.4 Case of 3D springs: Model C

This case also assumes that that the actuators are strictly rigid but the link flexibility is described by a full-scale 3D model that incorporates all deflections along and around x-,y-,z-axes of the three-dimensional Cartesian space. Relevant 6×6 stiffness matrix of the link may be expresses as (Connor, 1976)

$$\mathbf{K} = \frac{E}{L^3} \cdot \begin{bmatrix} A \cdot L^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 12 \cdot I_z & 0 & 0 & 0 & -6 \cdot I_z \cdot L \\ 0 & 0 & 12 \cdot I_y & 0 & 6 \cdot I_y \cdot L & 0 \\ 0 & 0 & 0 & G \cdot J \cdot L^2 / E & 0 & 0 \\ 0 & 0 & 6 \cdot I_y \cdot L & 0 & 4 \cdot I_y \cdot L^2 & 0 \\ 0 & -6 \cdot I_z \cdot L & 0 & 0 & 0 & 4 \cdot I_z \cdot L^2 \end{bmatrix} \tag{39}$$

where A, I_y, I_z are the area and the second moments of the link cross-section, J is the polar moment, E and G are the Young Coulomb modules of the link material. For a rectangular cross-section $a \times b$, the required parameters may be computed as $A = ab$ and $I_y = a^3 b / 12$, $I_z = ab^3 / 12$.

Similar to previous subsection, let apply the re-parameterization by defining the compliance with respect the z-axis as $1/K_\theta$ (here, it is element $k_{6,6}$ of the compliant matrix $\mathbf{k} = \mathbf{K}^{-1}$). This leads to expression

$$\mathbf{k} = \frac{1}{K_\theta} \cdot \begin{bmatrix} I_z / A & 0 & 0 & 0 & 0 & 0 \\ 0 & L^2 / 3 & 0 & 0 & 0 & L/2 \\ 0 & 0 & k_I \cdot L^2 / 3 & 0 & k_I \cdot L / 2 & 0 \\ 0 & 0 & 0 & k_J \cdot I_z / \left(2 \cdot L \cdot (1+\nu)\right) & 0 & 0 \\ 0 & 0 & k_I \cdot L / 2 & 0 & k_I & 0 \\ 0 & L/2 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{40}$$

where the coefficient k_J depends on cross-section shape, $k_I = I_y / I_z$, and ν is the Poisson ratio coefficient.

The kinematics of model C corresponds to the general expression $U_pR_aR_aS_p$ (see sub-section 5.1.1), it is described by the complete product of homogeneous matrices (33) that includes two passive joints $(\mathbf{q}, \mathbf{q}_t)$ incorporating five passive coordinates and three virtual-springs with 18 virtual coordinates totally (six for each link). It is obvious that analytical computation in this case is rather cumbrous, so the stiffness analysis will be performed numerically.

5.2 Stiffness analysis for model A

Let us examine first the model A that includes minimum number of flexible elements (two 1D virtual springs in the actuated joints) and may be tackled analytically. However, in spite of its simplicity, this model is potentially capable to detect the buckling phenomena at least if the initial posture of the kinematic chain is straight (S-configuration), because of evident mechanical analogy to straight columns behavior under axial compression. It is matter of research interest to evaluate other types of initial configurations with respect to the multiple loaded equilibriums, their stability and to compare with numerical results provided by the developed technique.

5.2.1 Computing static equilibriums

As follows from the kinematic equations (see subsection 5.1.2), model A includes there joint variables (q , θ_1, θ_2) one of which may be treated as a kinematically redundant one. Let us assume that the redundant variable is the passive joint coordinate q while the manipulator end-effector is located at the point $(x, y) = (3L - \delta, 0)$, where δ is a linear displacement along x-axis. Then, assuming that the initial values of the actuating coordinates (i.e. before the loading) are denotes as θ_1^0, θ_2^0 , the potential energy stored in the virtual springs may be expressed as the following function of the redundant variable

$$E(q) = \frac{1}{2} K_\theta \left(\theta_1(q) - \theta_1^0\right)^2 + \frac{1}{2} K_\theta \left(\theta_2(q) - \theta_2^0\right)^2 \tag{41}$$

where the θ_1, θ_2 are computed via the inverse kinematics as

$$\begin{aligned} &\theta_2(q) = \pm\arccos\left(\frac{(3-\Delta)^2 - 2(3-\Delta)\cos q - 1}{2}\right); \qquad \Delta = \delta / L \\ &\theta_1(q) = \operatorname{atan2}\left(\frac{-\sin q}{3 - \Delta - \cos q}\right) - \operatorname{atan2}\left(\frac{2\sin\theta_2}{(3-\Delta)^2 - 2(3-\Delta)\cos q + 1}\right) - q \end{aligned} \tag{42}$$

Using these equations, the desired equilibriums may be computed from the extrema of $E(q)$. In particular, stable equilibriums correspond to minima of this function, and unstable ones correspond to maxima:

$dE(q)/dq = 0;\;\; dE^2(q)/dq^2 > 0$: stable equilibrium ($E_{\min}$)

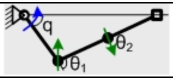
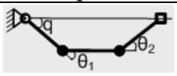
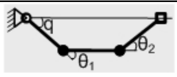
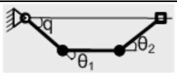
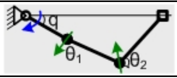
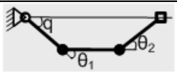
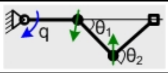
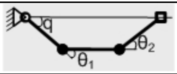
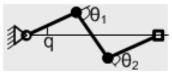
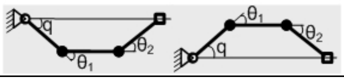
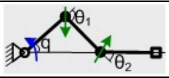
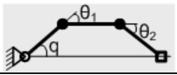
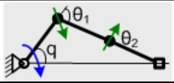
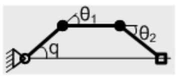
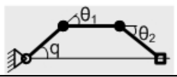
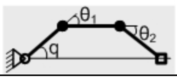
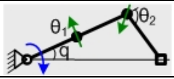
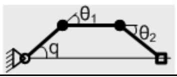
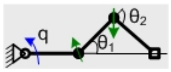
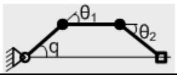
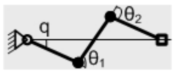
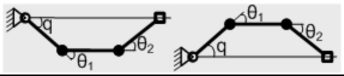
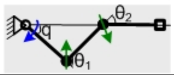
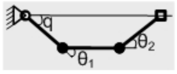
$dE(q)/dq = 0;\;\; dE^2(q)/dq^2 < 0$: unstable equilibrium ($E_{\max}$)

To illustrate this approach, Fig. 4 and Table 1 present a case study corresponding to the initial S-configuration of the examined kinematic chain (i.e. when $\theta_1^0 = \theta_2^0 = 0$). They allow comparing 12 different shapes of the deformated chain and selecting the best and the worst case with respect to the energy. As follows from these results, here there are two symmetrical maxima and two minima, i.e. two stable and two unstable equilibriums. Besides, the stable equilibriums correspond to Π-shaped deformated postures, and the unstable ones correspond to Z-shaped postures, as it is shown in Fig. 5. More detailed analysis allows deriving analytical expressions for the force and energy for small values of δ that will be used in the following subsection:

stable equilibrium: $E_{\min} \approx \delta \cdot K_\theta / L$; $F_s \approx K_\theta / L$

unstable equilibrium: $E_{\max} \approx \delta \cdot 3K_\theta / L$; $F_s \approx 3K_\theta / L$

It worth also mentioning that only stable equilibriums may be observed in practice and only this type of solutions is produced by the algorithm proposed in Section 3.

Configuration	q	θ_1	θ_2	Potential Energy	Configuration for stable static equilibrium
	$-\varphi_5$	φ_2	0	$1.5\dfrac{K_\theta}{L}$	
	$-\varphi_1$	φ_1	φ_1	$1.0\dfrac{K_\theta}{L}$	
	$-\varphi_4$	0	φ_2	$1.5\dfrac{K_\theta}{L}$	
	0	$-\varphi_1$	$-\varphi_3$	$2.5\dfrac{K_\theta}{L}$	
	φ_4	$-\varphi_2$	φ_2	$3.0\dfrac{K_\theta}{L}$	
	φ_1	$-\varphi_3$	φ_1	$2.5\dfrac{K_\theta}{L}$	
	φ_5	$-\varphi_2$	0	$1.5\dfrac{K_\theta}{L}$	
	φ_1	$-\varphi_1$	$-\varphi_1$	$1.0\dfrac{K_\theta}{L}$	
	φ_4	0	$-\varphi_2$	$1.5\dfrac{K_\theta}{L}$	
	0	φ_1	$-\varphi_3$	$2.5\dfrac{K_\theta}{L}$	
	$-\varphi_4$	φ_2	$-\varphi_2$	$3.0\dfrac{K_\theta}{L}$	
	$-\varphi_1$	$-\varphi_3$	$-\varphi_1$	$2.5\dfrac{K_\theta}{L}$	

$$\varphi_1 = \arccos(1-\frac{1}{2}\Delta)\;;\;\varphi_2 = \arccos(1-\frac{3}{2}\Delta+\frac{1}{4}\Delta^2)\;;\;\varphi_3 = \arccos(1-2\Delta+\frac{1}{4}\Delta^2)\;;$$

$$\varphi_4 = \arccos\left(\frac{12-6\Delta+\Delta^2}{4(3-\Delta)}\right);\;\varphi_5 = \arccos\left(\frac{6-6\Delta+\Delta^2}{2(3-\Delta)}\right)$$

Table 1. Selected postures of the deformated kinematic chain and their corresponding equilibriums (case of unloaded S-configuration, $\delta = L/10$)

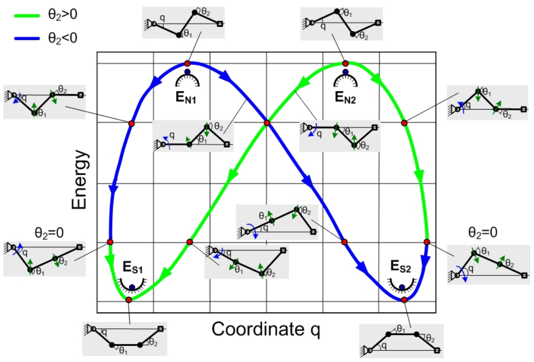


Fig. 4. Potential energy $E(q)$ and manipulator postures for different values of passive coordinate q (case of unloaded S-configuration, $\delta = L/10$)

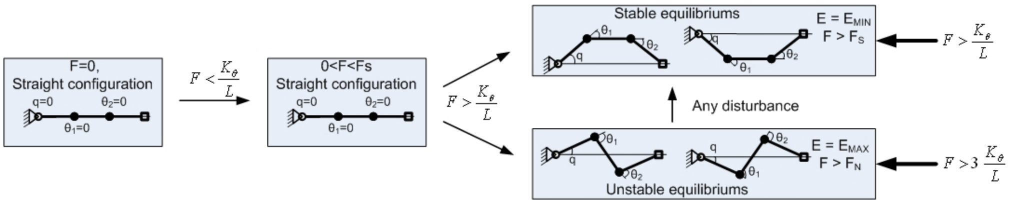


Fig. 5. Evolution of the S-configuration under external loading

5.2.2 Buckling behavior of S-configuration

Let us apply the above results to detailed analysis of S-configuration under external loading in the axial direction. As follows from the previous subsection, the external force $F \le K_\theta / L$ can not change the manipulator shape, similar to small compressing of straight columns that can not cause lateral deflections. Hence, in this case the straight configuration is stable. Further, for $K_\theta / L < F \le 3K_\theta / L$, the straight configuration may be hypothetically restored but becomes unstable, so any small disturbance will case sudden reshaping in the direction of a stable trapezoid-type posture. And finally, for $F > 3K_\theta / L$, there may exist two types of unstable equilibriums: the trivial straight-type and a more complicated zig-zag one. Hence, S-configuration demonstrates classical buckling phenomena that must be taken into account in the manipulator stiffness analysis.

If the assumption concerning small values of δ is released, analytical solutions for the non-trivial equilibriums may be still derived from the static equations. In particular, for the stable equilibrium, one can get

$$F_S(\Delta) = \frac{K_\theta}{L} \cdot \frac{\varphi}{\sin\varphi} \tag{43}$$

where $\varphi = \pm\arccos(1 - \Delta / 2)$. For the unstable equilibrium similar equation may be written as

$$F_N(\Delta) = \frac{K_\theta}{L} \cdot \frac{\cos(q+\theta) + 2 \cdot \cos q}{\sin\theta} \cdot \theta \tag{44}$$

where $q = \pm\arccos\left(\frac{12 - 6\Delta + \Delta^2}{12 - 4\Delta}\right)$, $\theta = \mp\arccos\left(1 - \frac{3\Delta}{2} + \frac{\Delta^2}{4}\right)$.

Corresponding plots are presented in Fig. 6 and 7 where there are also defined the bifurcation points, linear approximations of the force-deflection relations and relationship between external force and virtual joint coordinates. Their interpretation is similar to the axial compression of a straight column, which is a classical example in the strength of materials (Alfutov, 2000). It should be noted, that the developed numerical algorithm exactly produces the curve (11), including "Bifurcation point 1" which defines a critical force that can not be exceeded in practice. For practical application, it be useful linear approximation at the neighborhood of this bifurcation that yields the stiffness coefficient $0.17\,K_\theta / L^2$.

Therefore, for the S-configuration, the proposed technique is able to detect and evaluate numerically the buckling, and it provides good agreement with engineering intuition and relevant mechanical analogy (compressing of the straight column).

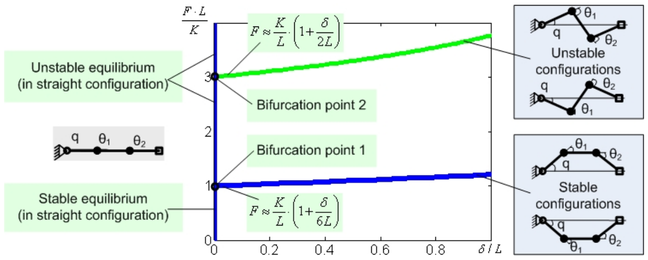


Fig. 6. Model A: Force-deflection relations for S-configuration (initial unloaded posture with coordinates $\theta_1^0 = \theta_2^0 = 0$)

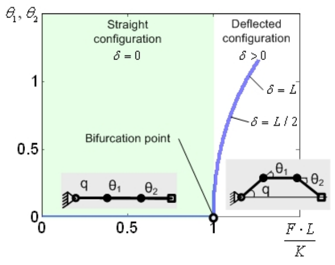


Fig. 7. Model A: Relationship between external force and virtual joint coordinates (case of S-configuration)

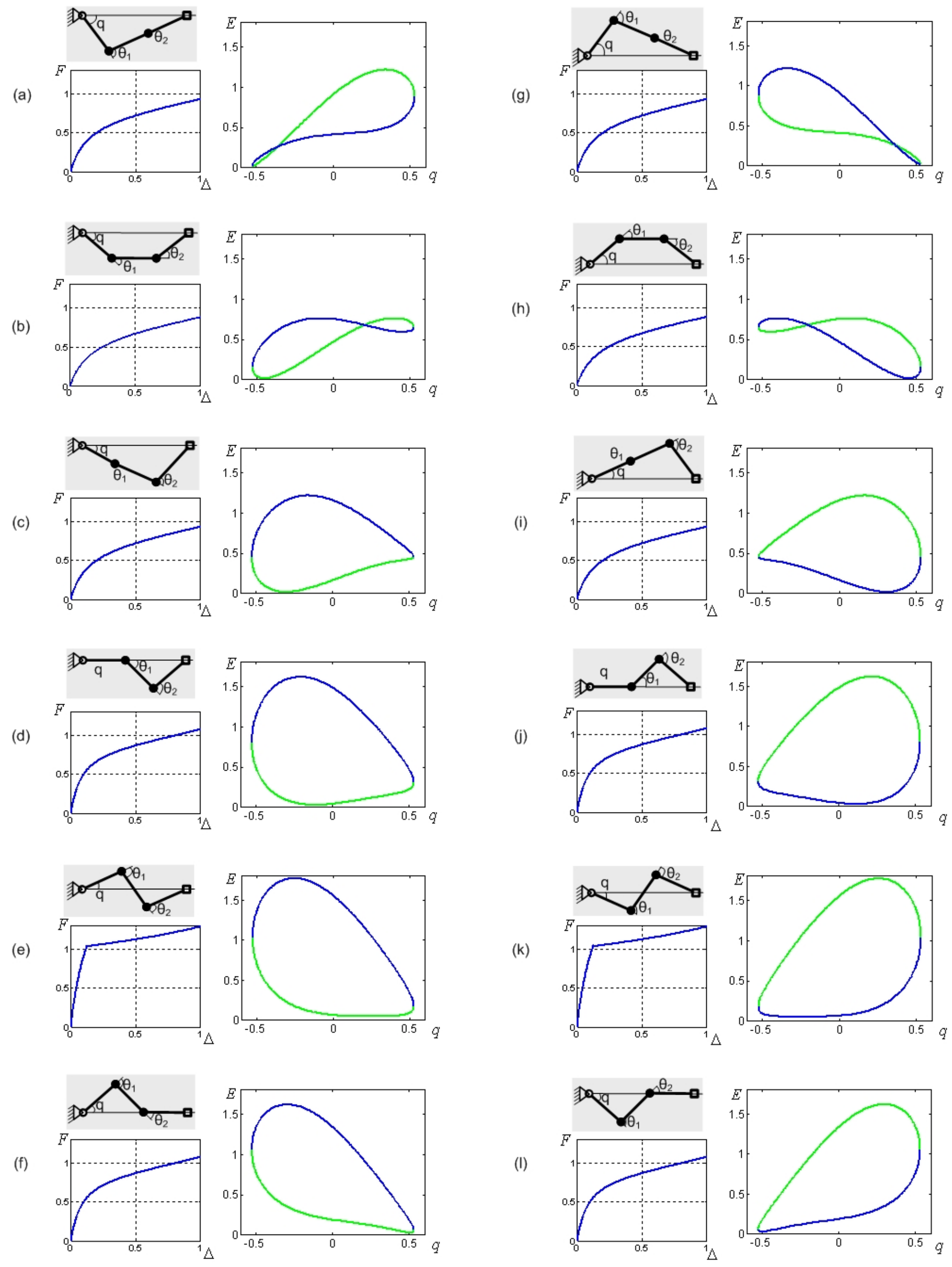
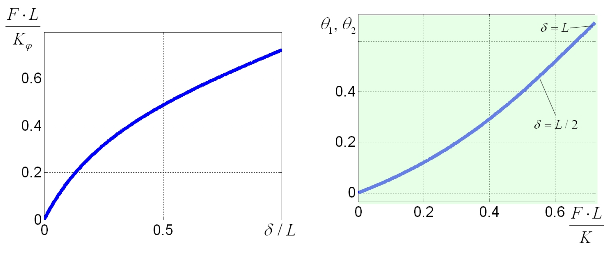


Fig. 8. Model A: Potential energy curves $E(q)$ and force-deflection relations $F(\Delta)$ for selected non-straight postures

5.2.3 Nonlinear phenomena for other configurations

Let us investigate now another unloaded shapes corresponding to Π-configuration, Z-configuration and several intermediate cases. Corresponding results are presented in Fig. 8 that contains the potential energy curves $E(q)$ for the end-point deflection $\delta = L/10$ and relevant force-deflection relations $F(\Delta)$. As follows from them, in most of the cases there exist a single stable and a single unstable equilibrium, so the kinematic chain can not suddenly change its shape due to external loading. The only exception is the case of Π-configuration (see Fig. 8,- b, h) where there are two stable and two unstable equilibriums. Another conclusion concerns the profile of the force-deflection plots that are highly nonlinear in all cases. Moreover, for Z-configuration, there exists a bifurcation of the stable equilibriums corresponding to the cuspidal point of the function $F(\Delta)$ where the stiffness reduces sharply.
More detailed analysis shows that Π-configuration demonstrates good analogy with axially compressed imperfect column where the deflection starts from the beginning of the loading and there is no sudden buckling, but the stiffness essentially reduces while the loading increases. Relevant plots are presented in Fig. 9 where the stiffness coefficient is about $1.78\,K_\theta / L^2$ at the beginning and $0.43\,K_\theta / L^2$ at the end of the curve $F(\Delta)$.



(a) (b)

Fig. 9. Model A: Force-deflection relations and deformations in actuated joints for Π-configuration (initial unloaded posture with coordinates $\theta_1^0 = \theta_2^0 = -30°$)

However, for Z-configuration that corresponds to the unloaded zig-zag shape, the stiffness behavior demonstrates the buckling that leads to sudden transformation from a symmetrical to a non-symmetrical posture as shown in Fig. 10. Here, there exist two stable equilibriums that differ in the values of the potential energy (see Fig. 8 e, k). Relevant plots are presented in Fig. 11 where the stiffness coefficient is about $16.7\,K_\theta / L^2$ at the beginning and $0.39\,K_\theta / L^2$ at the end of the curve $F(\Delta)$.

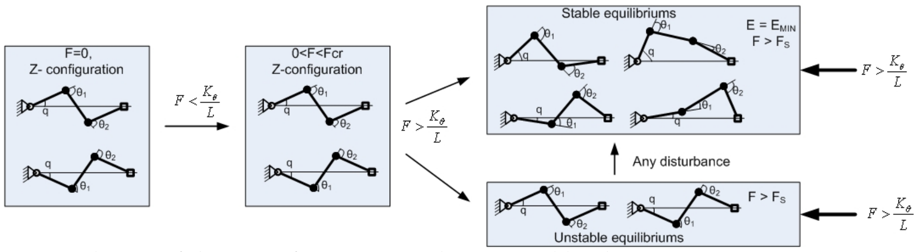


Fig. 10. Evolution of the Z-configuration under external loading

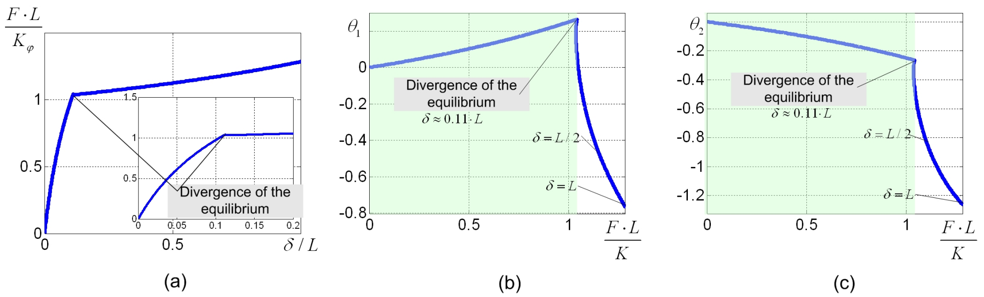


Fig. 11. Model A: Force-deflection relations and deformations in actuated joints for Z-configuration (initial unloaded posture with coordinates $\theta_1^0 = -30°;\ \theta_2^0 = 30°$)

Therefore, the stiffness analysis of model A (Table 2) allowed detecting more general class of manipulator postures that are dangerous with respect to the buckling. They include all configurations that posses an axial symmetry with respect to the direction of the external force (S- and Z-configurations for instance). These postures will be in the focus of the stiffness analysis for models B and C.

Configuration	Critical force	Stiffness for unloaded mode	Stiffness near the buckling ($\delta \approx 0$)		Stiffness for large deformations ($\delta \approx L$)
			$F < F_{cr}$	$F > F_{cr}$	
S-configuration $\theta_1^0 = \theta_2^0 = 0$	$\frac{K_\theta}{L}$	∞	∞	$0.20\,\frac{K_\theta}{L^2}$	$0.22\,\frac{K_\theta}{L^2}$
Π-configuration $\theta_1^0 = \theta_2^0 = -30°$	-	$1.78\,\frac{K_\theta}{L^2}$	-	-	$0.43\,\frac{K_\theta}{L^2}$
Z-configuration $\theta_1^0 = -30°;\ \theta_2^0 = 30°$	$1.03\,\frac{K_\theta}{L}$	$16.7\,\frac{K_\theta}{L^2}$	$5.50\,\frac{K_\theta}{L^2}$	$0.20\,\frac{K_\theta}{L^2}$	$0.39\,\frac{K_\theta}{L^2}$

Table 2. Summary of the Stiffness analysis for model A

5.3 Stiffness analysis for model B

In this case, it is assumed that the manipulator stiffness is caused by elasticity of the links while the actuating joints are rigid enough. The elastic deflections (bending and compression) are still restricted by the Cartesian xy-plane and each link includes only three virtual springs with joint variables θ_x^i, θ_y^i and $\theta_{\phi z}^i$, which describe respectively linear displacements in x- and y-directions and angular rotation around z-axis. Totally, the stiffness model has 11 variables (two for a passive joint and nine for the virtual springs of three links), so it was studied numerically, using the proposed technique. The stiffness parameters were evaluated assuming that the links are rectangular beams of the length L and the cross-section $a \times b$, where $a = 0.02L$ and $b = 0.05L$. For comparison purposes, corresponding stiffness matrices were scaled with respect to the bending coefficient to keep similarity with model A (see sub-section 5.1.3). The stiffness analysis was performed for three above mentioned typical configurations, assuming that the external force is directed along the x-axis causing compression of the examined kinematic chain.

For S-configuration, the results are presented in Fig. 12 that includes both the force-deflection plot and plots for deflections in the virtual springs. As follows from these results, here also there is very strong analogy with the compression of the straight column. In particular, first the links are subject the compression and the deflection starts from the beginning of the loading but the stiffness is very high (about $2500\, K_\theta / L^2$, for the assumed link shape). Then, after the buckling, the kinematic chain changes its shape to become non-symmetrical and the stiffness falls down to $0.20\, K_\theta / L^2$. The critical force may be also computed using the previous results, as $F_0 = K_\theta / L$.

For Π-configuration (Fig. 13), the stiffness properties are also qualitatively equivalent to the case of model A but the stiffness coefficient is slightly lower (in the frame of the adopted parameterization). For the presented curve $F(\Delta)$, it varies from $5.31\, K_\theta / L^2$ to $0.34\, K_\theta / L^2$.

For Z-configuration (Fug. 14), it has been also detected the buckling that occurs if the loading approaches to the critical value $F_0 = 1.07\, K_\theta / L$. At this point, the stiffness falls down from $100 \cdot K_\theta / L^2$ to $0.13 \cdot K_\theta / L^2$, which essentially differs from model A due to different nature of the virtual springs and to the cross-coupling between them. Here, it should be taken into account that the adopted parameterization ensure equivalence of the rotational compliance $1/K_\theta$ in virtual springs of models A and B, but their rotational stiffness is different.

Hence, the obtained results (Table 3) demonstrate qualitative similarity but some quantitative difference compared to model A. The latter is caused by different arrangement of the elastic elements in the virtual joints that corresponds to other physical assumptions. These results confirm essential influence of the external loading on the manipulators stiffness and potential instability of symmetrical postures.

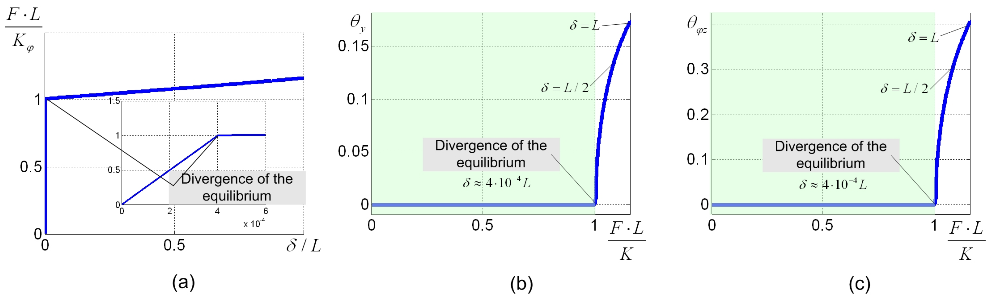


Fig. 12. Model B: Force-deflection relations and deflections in virtual springs for S-configuration (initial unloaded posture with coordinates $\theta_1^o = \theta_2^o = 0$)

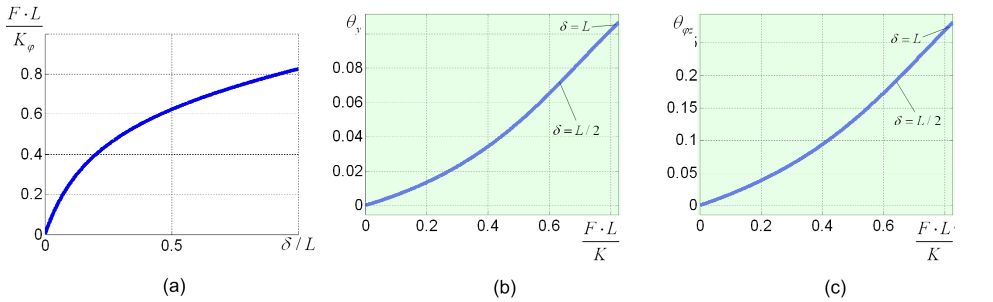


Fig. 13. Model B: Force-deflection relations and deflections in virtual springs for Π-configuration (initial unloaded posture with coordinates $\theta_1^o = \theta_2^o = -30°$)

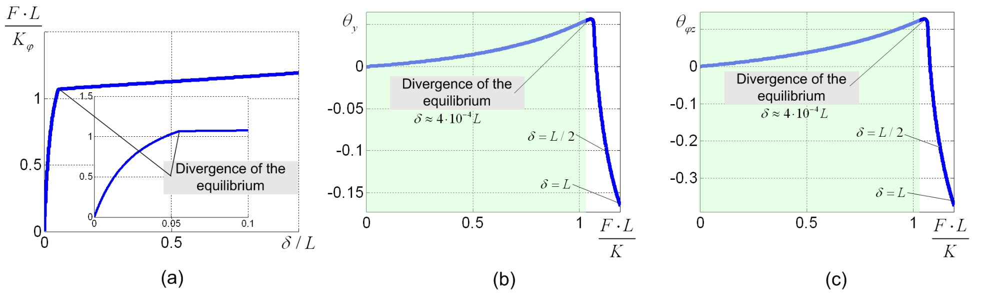


Fig. 14. Model B: Force-deflection relations and deflections in virtual springs for Z-configuration (initial unloaded posture with coordinates $\theta_1^o = -30°;\ \theta_2^o = 30°$)

Configuration	Critical force	Stiffness for unloaded mode	Stiffness near the buckling ($\delta \approx 0$)		Stiffness for large deformatio ns ($\delta \approx L$)
			$F < F_{cr}$	$F > F_{cr}$	
S-configuration $\theta_1^0 = \theta_2^0 = 0$	$\frac{K_\theta}{L}$	$2500\frac{K_\theta}{L^2}$	$2500\frac{K_\theta}{L^2}$	$0.20\frac{K_\theta}{L^2}$	$0.22\frac{K_\theta}{L^2}$
Π-configuration $\theta_1^0 = \theta_2^0 = -30°$	-	$5.31\frac{K_\theta}{L^2}$	-	-	$0.34\frac{K_\theta}{L^2}$
Z-configuration $\theta_1^0 = -30°;\ \theta_2^0 = 30°$	$1.07\frac{K_\theta}{L}$	$100\frac{K_\theta}{L^2}$	$0.13\frac{K_\theta}{L^2}$	$0.13\frac{K_\theta}{L^2}$	$0.16\frac{K_\theta}{L^2}$

Table 3. Summary of the Stiffness analysis for model B

5.4 Stiffness analyses for the model C

Finally, let us consider model C where the link elasticity is described in 3D space and corresponding stiffness matrices have dimension 6×6 (the actuating joints are assumed perfect and rigid, similar to model B). It is also assumed that the links are rectangular beams of the length L with the cross-section $a \times b$, where $a = 0.02L$, $b = 0.05L$ and the smaller value a corresponds to z-direction that was not studied above. The latter assumption agrees with real dimensions of links used in some parallel manipulators, such as Orthoglide (Chablat & Wenger, 2003).

To ensure comparability of all examined cases, the link stiffness matrices were parameterized with respect to the bending coefficient of the z-axis K_θ (see sub-section 5.1.4). In total, the stiffness model includes 23 variables (five for passive joints and 18 for the virtual springs of three links) and it was studied numerically. The stiffness analysis was performed for the same manipulator configurations (S, Π and Z) in the unloaded mode and the same direction of the external force as for models A and B.

For S-configuration, the results (Fig. 15) are qualitatively similar to ones obtained for model B. Besides, numerical value of the stiffness for the non-loaded case is the same, $2500\,K_\theta / L^2$. However, here the buckling occurs for essentially lower critical force, $0.16\,K_\theta / L$, that corresponds to sudden lateral deflection in z-direction. Then, after the buckling, the stiffness falls down to $0.20\,K_\theta / L^2$. It worth mentioning that the axial deflection corresponding to the critical force is very low, it is equal to $7 \cdot 10^{-5} \cdot \delta / L$. But further increase of the force by only 20% leads to extremely high increase of the deflection, in more then 1000 times.

In contrast, for Π-configuration (Fig. 16), it was detected buckling that does not exist in models A and B. In particular, if the external force exceeds the critical value $0.20\,K_\theta / L$ the stiffness suddenly reduces from $1.03\,K_\theta / L^2$ to $0.04\,K_\theta / L^2$ (for comparison, the stiffness coefficient for unloaded mode is $1.70\,K_\theta / L^2$). Physically it is also explained by sudden deflection in z-direction that it was beyond capabilities of previous models. It worth also mentioned that, in this case study, the stiffness of manipulator links in z-direction is essentially lower than in y-direction. Another interpretation of this buckling phenomena may be presented as sudden loss of symmetry with respect to xy-plane.

For Z-configuration (Fig. 17), the results remain qualitatively the same, but corresponding numerical values are changed. Thus, manipulator stiffness for the unloaded mode is $7.52\,K_\theta / L$, it gradually reduces to $5.88\,K_\theta / L$ and then, after the buckling, falls down to $0.03\,K_\theta / L$. Corresponding value of the critical force is $0.17\,K_\theta / L$ and is also defined by the z-direction deflection.

More detailed numerical results concerning model C are presented in Table 4. As follows from them, a full-scale 3D stiffness analysis yields lower values of critical force compared to models A and B. Besides, for model C, all examined postures demonstrated buckling related to sudden deflections in the z- direction. This presents another source of potential structural instability of kinematic chains that posses the symmetry with respect to a plane.

Generally, summarizing all presented case studies, it should be concluded that the developed technique produces reliable results, it is able to evaluate manipulator stiffness and to compute the range of the loading that prevents buckling.

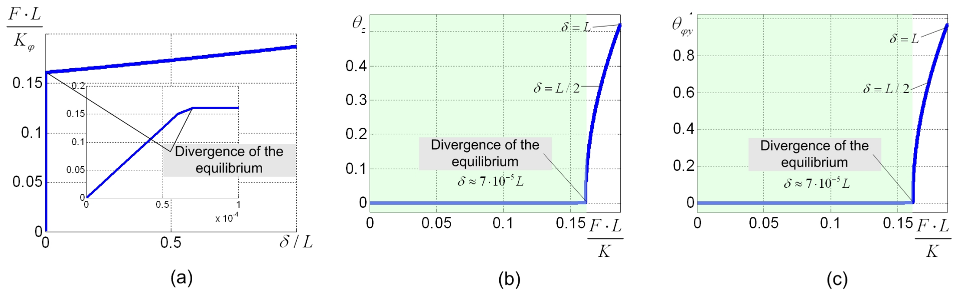


Fig. 15. Model C: Force-deflection relations and deflections in virtual springs for S-configuration (initial unloaded posture with coordinates $\theta_1^0 = \theta_2^0 = 0$)

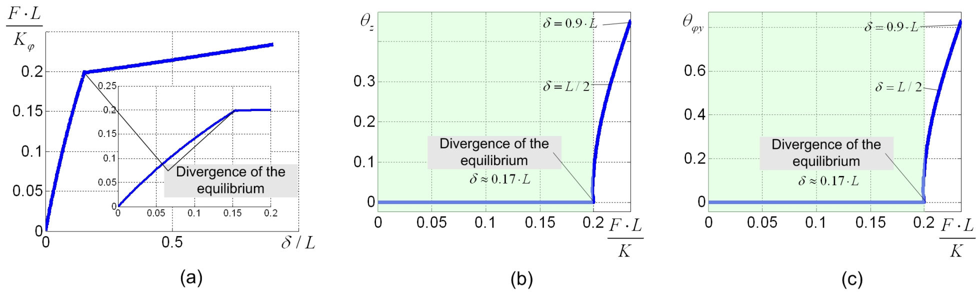
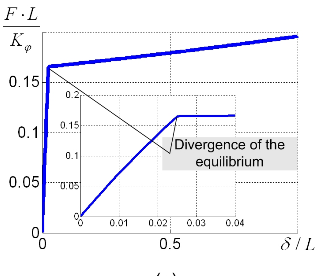
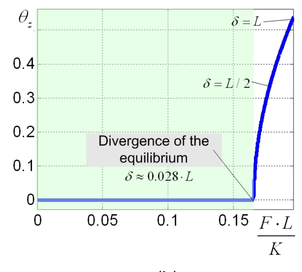


Fig. 16. Model C: Force-deflection relations and deflections in virtual springs for Π-configuration (initial unloaded posture with coordinates $\theta_1^0 = \theta_2^0 = -30°$)





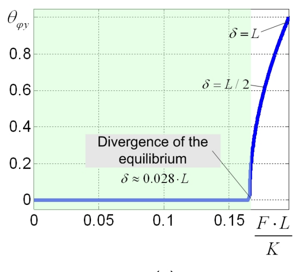


Fig. 17. Model C: Force-deflection relations and deflections in virtual springs for Z-configuration (initial unloaded posture with coordinates $\theta_1^0 = -30°;\ \theta_2^0 = 30°$)

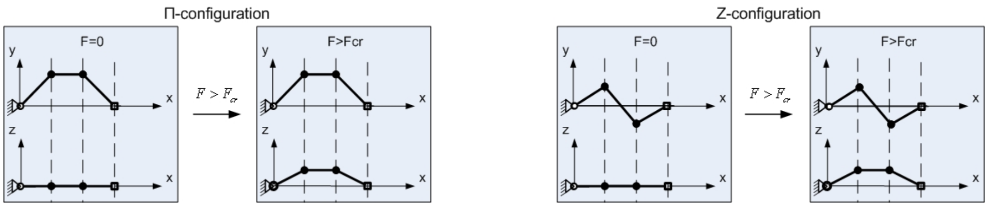


Fig. 18. Model C: Evolution of the Π- and Z-configurations under external loading

Configuration	Critical force	Stiffness for unloaded mode	Stiffness near the buckling ($\delta \approx 0$)		Stiffness for big deformations ($\delta \approx L$)
			$F < F_{cr}$	$F > F_{cr}$	
S-configuration $\theta_1^0 = \theta_2^0 = 0$	$0.16\,\frac{K_\theta}{L}$	$2500\,\frac{K_\theta}{L^2}$	$2500\,\frac{K_\theta}{L^2}$	$0.023\,\frac{K_\theta}{L^2}$	$0.029\,\frac{K_\theta}{L^2}$
Π-configuration $\theta_1^0 = \theta_2^0 = -30°$	$0.20\frac{K_\theta}{L}$	$1.70\frac{K_\theta}{L^2}$	$1.03\,\frac{K_\theta}{L^2}$	$0.043\,\frac{K_\theta}{L^2}$	$0.13\frac{K_\theta}{L^2}$
Z-configuration $\theta_1^0 = -30°;\ \theta_2^0 = 30°$	$0.17\,\frac{K_\theta}{L}$	$7.52\,\frac{K_\theta}{L^2}$	$5.88\frac{K_\theta}{L^2}$	$0.027\,\frac{K_\theta}{L^2}$	$0.035\,\frac{K_\theta}{L^2}$

Table 4. Summary of the Stiffness analysis for model C

6. Conclusions

In modern robot-based manufacturing, the stiffness analysis becomes a critical issue. It is motivated by current trends in manipulator design that are targeted at achieving high dynamic performances with relatively small link masses and low energy consumption in actuators. These demand a revision of the existing stiffness modeling techniques that must take into account the external loading imposed by a manufacturing process. Accordingly, this chapter focuses on the enhanced stiffness modeling and analysis of serial kinematic chains with passive joints, which are widely used in parallel robotic systems. In contrast to previous works, the stiffness is evaluated for the loaded working

mode corresponding to the static equilibrium of the elastic forces and the external wrench acting upon the manipulator end point. The proposed technique allows finding the full-scale "load-deflection" relation for any given workspace point and to linearise it taking into account variation of the manipulator Jacobian due to the external force/torque. These enables designer to evaluate critical forces that may provoke non-linear behavior of the manipulators, such as sudden failure due to elastic instability (buckling) which has not been previously studied in robotic literature.
One of the essential novelties proposed here is a new solution strategy of the kinetostatic equations, which takes into account the passive joints in the straightforward way, allowing computing a stiffness matrix even for singular Jacobian and Hessian. Besides, the method does not require manual model reduction that usually deals with elimination of the redundant springs corresponding to the passive joints, since this operation is inherently included in the numerical algorithm. Another advantage is the computational simplicity that requires low-dimensional matrix inversion compared to other techniques. The theoretical contributions also include the matrix criteria for the stability of the static equilibrium in the loaded mode.
The advantages of the developed technique are illustrated by several examples that deal with kinematic chains employed in typical parallel manipulators. They demonstrate possible non-linear effects that may arise in loaded mode, including essential dependence of the stiffness on the applied force/torque and sudden change of the stiffness if the external wrench exceeds the critical value. Besides, there were detected several typical configurations of serial kinematic chains that are potentially dangerous with respect to buckling. It is shown that such configurations possess either axial or planar symmetry with respect to direction of the external loading. This research may be also extended for more sophisticated architectures that include parallel manipulators with intermediate links between the main kinematic chains, kinematic parallelograms and other structures improving rigidity of the manipulating system.

7. Acknowledgements

The work presented in this paper was partially funded by the Region "Pays de la Loire", France and by the EU commission (project NEXT).

8. References

Alfutov N.A. (2000), Stability of Elastic Structures, Series: *Foundations of Engineering Mechanics*, 2000, IX, 356 p. 128 illus., Hardcover ISBN: 978-3-540-65700-2

Alici, G. & Shirinzadeh, B. (2005). Enhanced stiffness modeling, identification and characterization for robot manipulators, Proceedings of IEEE Transactions on Robotics vol. 21, No. 4, pp. 554–564.

Angeles. J (2007). Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms, Springer, ISBN: 978-0-387-29412-4, New York

Chablat D. & Wenger P. (2003). Architecture Optimization of a 3-DOF Parallel Mechanism for Machining Applications, the Orthoglide, Proceedings of IEEE Transactions on Robotics and Automation, vol. 19, no. 3, pp. 403-410.

Clavel R. (1988). DELTA, a fast robot with parallel geometry, Proceedings, of the 18th International Symposium of Robotic Manipulators, IFR Publication, pp. 91–100.

Connor J. (1976), *Analysis of Structural Member Systems*, Ronald Press, 1976.

Deblaise D., Hernot X. & Maurine P. (2006). A systematic analytical method for pkm stiffness matrix calculation, Proceedings of the 2006 IEEE International Conference on Robotics and Automation, pp. 4213-4219, May 2006, Orlando, Florida

Gosselin C.M. (1990). Stiffness mapping for parallel manipulators, Proceedings of IEEE Transactions on Robotics and Automation, vol. 6, pp. 377–382.

Hu X.; Wang R.; Wu F.;Jin D.; Jia X.;Zhang J.; Cai F., & Zheng Sh. (2007) Finite Element Analysis of a Six-Component Force Sensor for the Trans-Femoral Prosthesis, In Digital Human Modeling, V.G. Duffy (Ed.), pp. 633–639, © Springer-Verlag Berlin Heidelberg 2007

Kovecses, J. & Angeles, J. (2007). The stiffness matrix in elastically articulated rigid-body systems. In Multibody System Dynamics Vol. 18, No. 2, pp. 169–184

Li Y.W., Wang J.S., & Wang L.P. (2002). Stiffness analysis of a Stewart platform-based parallel kinematic machine, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), vol. 4, pp. 3672–3677, Washington, US, May 11–15, 2002

Li Y. & Xu Q. (2008). Stiffness analysis for a 3-PUU parallel kinematic machine, Mechanism and Machine Theory, vol. 43, no. 2, pp. 186-200.

Los J.; Tomiyama T.; Shibukawa T. & Takeuchi Y. (2008). Expanding the possibilities of position error compensation in CAM for PKM milling machines, Proceedings of The 41st CIRP Conference on Manufacturing Systems 2008, pp. 399-404

Martin H. C. (1966). Introduction to matrix methods of structural analysis, McGraw-Hill Book Company,

Majou F.; Gosselin C.; Wenger P. & Chablat D. (2007). Parametric stiffness analysis of the Orthoglide. Mechanism and Machine Theory Vol. 42, No. 3, pp. 296–311.

Merlet J.-P. (2006). Parallel Robots, Kluwer Academic Publishers, Dordrecht.

Nagai K. & Liu Zh. (2007). A Systematic Approach to Stiffness Analysis of Parallel Mechanisms and its Comparison with FEM, Proceeding of SICE Annual Conference 2007, pp 1087-1094, Sept. 17-20, 2007, Kagawa University, Japan

Pashkevich A., Chablat D. & Wenger P. (2008). Stiffness analysis of 3-d.o.f. overconstrained translational parallel manipulators, Proceedings of *IEEE International Conference on Robotics and Automation*, pp. 1562-1567.

Pashkevich A.; Klimchik A.; Chablat D. & Wenger P. (2009 a). Accuracy Improvement for Stiffness Modeling of Parallel Manipulators, Proceedings of 42nd CIRP Conference on Manufacturing Systems, June 2009

Pashkevich A.; Klimchik A.; Chablat D. & Wenger P. (2009 b). Stiffness analysis of multichain parallel robotic systems with loading, Jornal of Automation, Mobile Robotics & Intelligent Systems, vol. 3, No. 3, pp. 75-82

Pashkevich A., Chablat D. & Wenger P. (2009 c). Stiffness analysis of overconstrained parallel manipulators, Mechanism and Machine Theory, vol. 44, pp. 966-982

Piras G.; Cleghorn W.L. & Mills J.K. (2005). Dynamic finite-element analysis of a planar high-speed, high-precision parallel manipulator with flexible links. Mechanism and Machine Theory, Vol. 40, No. 7, pp. 849-862.

Quennouelle C. & Gosselin C. M. (2008 a). Stiffness Matrix of Compliant Parallel Mechanisms, In Springer Advances in Robot Kinematics: Analysis and Design, pp. 331-341

Quennouelle C. & Gosselin C. M. (2008 b). Instantaneous Kinemato-Static Model of Planar Compliant Parallel Mechanisms, Proceedings of ASME International Design Engineering Technical Conferences Brooklyn, NY, USA, August 3-6.

Salisbury, J., (1980). Active Stiffness Control of a Manipulator in Cartesian Coordinates, Proceedings of 19th IEEE Conference on Decision and Control, pp. 87–97.

Siciliano, B. & Khatib, O. (2008). Springer Handbook of Robotics, ISBN: 978-3-540-23957-4, Berlin

Strang G. (1998) Introduction to Linear Algebra. Wellesley, MA: Wellesley Cambridge Press

Timoshenko S. & Goodier J. N. (1970), Theory of elasticity, 3d ed., McGraw-Hill, New York

Zhang Dan ; Bi Zhuming & Li Beizhi (2009). Design and kinetostatic analysisofa new parallel manipulator. Robotics and Computer-Integrated Manufacturing Vol. 25, pp. 782–791

Fast Dynamic Model of a Moving-base 6-DOF Parallel Manipulator

António M. Lopes
Unidade de Integração de Sistemas e Processos Automatizados,
Faculdade de Engenharia, Universidade do Porto
Portugal

1. Introduction

Dynamic models play an important role in parallel manipulators simulation and control. Mainly in the later case, the efficiency of the involved computations is of paramount importance, because manipulator real-time control is usually necessary (Zhao & Gao, 2009).
The dynamic model of a parallel manipulator operating in free space can be represented in Cartesian coordinates by a system of nonlinear differential equations, which may be written in matrix form as

$$\mathbf{I}(\mathbf{x})\cdot\ddot{\mathbf{x}}+\mathbf{V}(\mathbf{x},\dot{\mathbf{x}})\cdot\dot{\mathbf{x}}+\mathbf{G}(\mathbf{x})=\mathbf{f} \tag{1}$$

$\mathbf{I}(\mathbf{x})$ being the inertia matrix, $\mathbf{V}(\mathbf{x},\dot{\mathbf{x}})$ the Coriolis and centripetal terms matrix, $\mathbf{G}(\mathbf{x})$ a vector of gravitational generalized forces, $\mathbf{x}$ the generalized position of the moving platform (or end-effector) and $\mathbf{f}$ the controlled generalized force applied on the end-effector. Thus,

$$\mathbf{f}=\mathbf{J}^{T}(\mathbf{x})\cdot\boldsymbol{\tau} \tag{2}$$

where τ is the generalized force developed by the actuators and $\mathbf{J}(\mathbf{x})$ is the inverse kinematics jacobian matrix (Merlet, 2006).
Dynamic modelling of parallel manipulators presents an inherent complexity, mainly due to system closed-loop structure and kinematic constraints. Several approaches have been applied to the dynamic analysis of parallel manipulators, the Newton-Euler and the Lagrange methods being the most popular ones (Do & Yang, 1988; Reboulet & Berthomieu, 1991; Ji, 1994; Dasgupta & Mruthyunjaya, 1998; Khalil & Ibrahim, 2007; Riebe & Ulbrich, 2003; Guo & Li, 2006; Nguyen & Pooran, 1989; Lebret et al., 1993; Di Gregório & Parenti-Castelli, 2004; Caccavale et al., 2003; Dasgupta & Choudhury, 1999). These methods use classical mechanics principles, as is the case for all the approaches found in the literature, namely the ones based on the principle of virtual work (Staicu et al., 2007; Tsai, 2000; Wang & Gosselin, 1998), screw theory (Gallardo et al., 2003), recursive matrix method (Staicu & Zhang, 2008), Hamilton's principle (Miller, 2004), and Kane's equation (Liu et al., 2000).

Thus, all approaches are equivalent, leading to equivalent dynamic equations. Nevertheless, these equations can present different levels of complexity and associated computational loads (Zhao & Gao, 2009). Minimizing the number of operations involved in the computation of the manipulator dynamic model has been the main goal of recent proposed techniques (Zhao & Gao, 2009; Staicu & Zhang, 2008; Abdellatif & Heimann, 2009; Wang et al., 2007; Sokolov & Xirouchakis, 2007; Bhattacharya et al., 1997; Carricato & Gosselin, 2009; Lopes, 2009).
This book chapter presents the generalized momentum concept to model the dynamics of a Stewart platform manipulator having a non-stationary base platform. This is important, for example, in macro/micro robotic applications, where a small manipulator is attached in series to a big one. The later performs large amplitude movements, while the former is only responsible for the small motions. The book chapter is organized as follows. Section 2 presents a brief description of the parallel manipulator under study. In section 3 a complete dynamic model is developed. The generalized momentum approach is used and the motion of the manipulator base platform is considered. Conclusions are drawn in section 4.

2. Manipulator Kinematic Structure

A Stewart platform manipulator is being considered. It comprises a (usually) fixed platform (the base) and a moving platform (the payload platform), linked together by six independent, identical, open kinematic chains (Raghavan, 1993). In this book chapter a particular design will be considered as shown in Figure 1 (Fichter, 1986). In this case, each chain (leg) comprises a cylinder and a piston (or spindle) that are connected together by a prismatic joint, l_i. The upper end of each leg is connected to the moving platform by a spherical joint whereas the lower end is connected to the fixed base by a universal joint. Points B_i and P_i are the connecting points to the base and moving platforms, respectively (Figure 2). They are located at the vertices of two semi-regular hexagons inscribed in circumferences of radius r_B and r_P. The separation angles between points B_1 and B_6, B_2 and B_3, and B_4 and B_5 are denoted by $2\phi_B$. In a similar way, the separation angles between points P_1 and P_2, P_3 and P_4, and P_5 and P_6 are denoted by $2\phi_P$ (Figure 2).

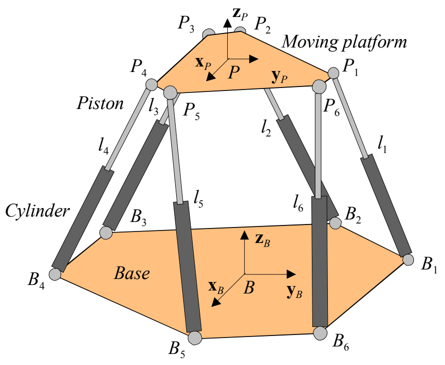


Fig. 1. Stewart platform kinematic structure

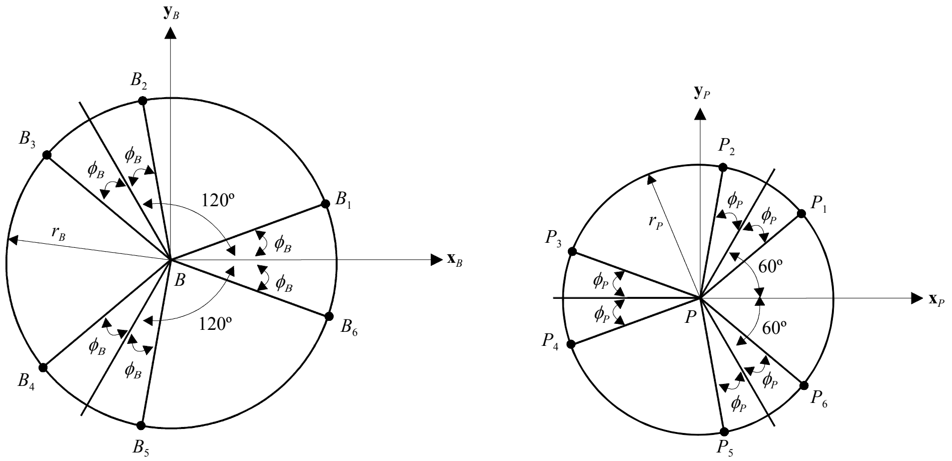


Fig. 2. Position of the connecting points to the base and payload platforms

For kinematic modelling purposes, two frames, {P} and {B}, are attached to the moving and base platforms, respectively. Its origins are the platforms centres of mass. The generalized position of frame {P} relative to frame {B} may be represented by the vector:

$$^{B}\mathbf{x}_{P}\,|_{B|E} = [x_P \quad y_P \quad z_P \quad \psi_P \quad \theta_P \quad \varphi_P]^T = [\,^{B}\mathbf{x}^T_{P(pos)}\,|_B \quad ^{B}\mathbf{x}^T_{P(o)}\,|_E\,]^T \tag{3}$$

where $^{B}\mathbf{x}_{P(pos)}\,|_B = [x_P \quad y_P \quad z_P]^T$ is the position of the origin of frame {P} relative to frame {B}, and $^{B}\mathbf{x}_{P(o)}\,|_E = [\psi_P \quad \theta_P \quad \varphi_P]^T$ defines an Euler angles system representing orientation of frame {P} relative to {B}. The used Euler angles system corresponds to the basic rotations (Vukobratovic & Kircanski, 1986): ψ_P about $\mathbf{z}_P$; θ_P about the rotated axis $\mathbf{y}_{P'}$; and φ_P about the rotated axis $\mathbf{x}_{P''}$. The rotation matrix is given by:

$$^{B}\mathbf{R}_P = \begin{bmatrix} C\psi_P C\theta_P & C\psi_P S\theta_P S\varphi_P - S\psi_P C\varphi_P & C\psi_P S\theta_P C\varphi_P + S\psi_P S\varphi_P \\ S\psi_P C\theta_P & S\psi_P S\theta_P S\varphi_P + C\psi_P C\varphi_P & S\psi_P S\theta_P C\varphi_P - C\psi_P S\varphi_P \\ -S\theta_P & C\theta_P S\varphi_P & C\theta_P C\varphi_P \end{bmatrix} \tag{4}$$

$S(\cdot)$ and $C(\cdot)$ correspond to the sine and cosine functions, respectively.
The manipulator position and velocity kinematic models are well known, being obtainable from the geometrical analysis of the kinematics chains. The velocity kinematics is represented by the Euler angles jacobian matrix, $\mathbf{J}_E$, or the kinematic jacobian, $\mathbf{J}_C$ (Merlet, 2006). These jacobians relate the velocities of the active joints (the actuators) to the generalized velocity of the moving platform:

$$\dot{\mathbf{l}} = \mathbf{J}_E \cdot {}^B\dot{\mathbf{x}}_P|_{B|E} = \mathbf{J}_E \cdot \begin{bmatrix} {}^B\dot{\mathbf{x}}_{P(pos)}|_B \\ {}^B\dot{\mathbf{x}}_{P(o)}|_E \end{bmatrix} \tag{5}$$

$$\dot{\mathbf{l}} = \mathbf{J}_C \cdot {}^B\dot{\mathbf{x}}_P|_B = \mathbf{J}_C \cdot \begin{bmatrix} {}^B\dot{\mathbf{x}}_{P(pos)}|_B \\ {}^B\boldsymbol{\omega}_P|_B \end{bmatrix} \tag{6}$$

with

$$\dot{\mathbf{l}} = \begin{bmatrix} \dot{l}_1 & \dot{l}_2 & \cdots & \dot{l}_6 \end{bmatrix}^T \tag{7}$$

$${}^B\boldsymbol{\omega}_P|_B = \mathbf{J}_A \cdot {}^B\dot{\mathbf{x}}_{P(o)}|_E \tag{8}$$

and (Vukobratovic & Kircanski, 1986)

$$\mathbf{J}_A = \begin{bmatrix} 0 & -S\psi_P & C\theta_P C\psi_P \\ 0 & C\psi_P & C\theta_P S\psi_P \\ 1 & 0 & -S\theta_P \end{bmatrix} \tag{9}$$

Vectors ${}^B\dot{\mathbf{x}}_{P(pos)}|_B \equiv {}^B\mathbf{v}_P|_B$ and ${}^B\boldsymbol{\omega}_P|_B$ represent the linear and angular velocity of the moving platform relative to {B}, and ${}^B\dot{\mathbf{x}}_{P(o)}|_E$ represents the Euler angles time derivative.

3. Complete Dynamic Modelling Using the Generalized Momentum Approach

It is well known the generalized momentum of a rigid body, $\mathbf{q}_c$, may be computed from the following general expression:

$$\mathbf{q}_c = \mathbf{I}_c \cdot \mathbf{u}_c \tag{10}$$

Vector $\mathbf{u}_c$ represents the generalized velocity (linear and angular) of the body and $\mathbf{I}_c$ is its inertia matrix. Vectors $\mathbf{q}_c$ and $\mathbf{u}_c$ and inertia matrix $\mathbf{I}_c$ must be expressed in the same frame of reference.

Equation (10) may also be written as

$$\mathbf{q}_c = \begin{bmatrix} \mathbf{Q}_c \\ \mathbf{H}_c \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{c(tra)} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{c(rot)} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{bmatrix} \tag{11}$$

where $\mathbf{Q}_c$ is the linear momentum vector due to rigid body translation and $\mathbf{H}_c$ is the angular momentum vector due to body rotation. $\mathbf{I}_{c(tra)}$ is the translational inertia matrix and $\mathbf{I}_{c(rot)}$ is the rotational inertia matrix. $\mathbf{v}_c$ and $\boldsymbol{\omega}_c$ are the body linear and angular velocities.

The inertial component of the generalized force (force and moment) acting on the body can be obtained from the time derivative of equation (10):

$$\mathbf{f}_{c(ine)} = \dot{\mathbf{q}}_c = \dot{\mathbf{I}}_c \cdot \mathbf{u}_c + \mathbf{I}_c \cdot \dot{\mathbf{u}}_c \tag{12}$$

with force and moment expressed in the same frame and $\mathbf{f}_{c(ine)} = \left[\mathbf{F}^T_{c(ine)} \quad \mathbf{M}^T_{c(ine)}\right]^T$.
Equivalently, force and moment vectors are:

$$\mathbf{F}_{c(ine)} = \dot{\mathbf{Q}}_c = \mathbf{I}_{c(tra)} \cdot \dot{\mathbf{v}}_c + \dot{\mathbf{I}}_{c(tra)} \cdot \mathbf{v}_c \tag{13}$$

$$\mathbf{M}_{c(ine)} = \dot{\mathbf{H}}_c = \mathbf{I}_{c(rot)} \cdot \dot{\boldsymbol{\omega}}_c + \dot{\mathbf{I}}_{c(rot)} \cdot \boldsymbol{\omega}_c \tag{14}$$

3.1 Payload Platform Modelling

Considering the Stewart platform manipulator base motion, i.e., the motion of frame {B} relative to a fixed world frame {W}≡{$\mathbf{x}_W$, $\mathbf{y}_W$, $\mathbf{z}_W$}, the position of the payload platform, {P}, relative to {W} and expressed in {W}, may be given by

$${}^W\mathbf{p}_P|_W = {}^W\mathbf{p}_B|_W + {}^B\mathbf{p}_P|_W \tag{15}$$

where ${}^W\mathbf{p}_B|_W$ is the position of frame {B}, and ${}^B\mathbf{p}_P|_W$ represents the position of frame {P} relative to {B} and expressed in {W}.
The linear velocity of the payload platform, {P}, relative to {W} and expressed in {W}, may be obtained taking the time derivative of the previous equation, that is,

$${}^W\mathbf{v}_P|_W = {}^W\mathbf{v}_B|_W + {}^B\mathbf{v}_P|_W + {}^W\boldsymbol{\omega}_B|_W \times {}^B\mathbf{p}_P|_W \tag{16}$$

where ${}^W\mathbf{v}_B|_W$ is the linear velocity of frame {B}, ${}^B\mathbf{v}_P|_W$ is the linear velocity of frame {P} as seen by an observer fixed in {B}, ${}^W\boldsymbol{\omega}_B|_W$ represents the angular velocity of frame {B} relative to {W}, and ${}^B\mathbf{p}_P|_W \equiv {}^B\mathbf{x}_{P(pos)}|_W$ represents the position of {P} relative to {B} and expressed in {W}.
In the following analysis, knowledge of the generalized position of frame {B} relative to {W}, ${}^W\mathbf{x}_B|_{W|E} = [x_B \quad y_B \quad z_B \quad \psi_B \quad \theta_B \quad \varphi_B]^T = [{}^W\mathbf{x}^T_{B(pos)}|_W \quad {}^W\mathbf{x}^T_{B(o)}|_E]^T$, shall be assumed: ${}^W\mathbf{x}_{B(pos)}|_W$ represents the position vector expressed in frame {W}, and ${}^W\mathbf{x}_{B(o)}|_E$ represents the orientation expressed in an Euler angles system. Knowledge of its first and second time derivatives shall also be supposed i.e., ${}^W\dot{\mathbf{x}}_B|_{W|E}$ and ${}^W\ddot{\mathbf{x}}_B|_{W|E}$, respectively. Therefore, the orientation matrix, ${}^W\mathbf{R}_B$, of frame {B} relative to {W} can be easily computed, and the jacobian, $\mathbf{J}_G$, relating the angular velocity of the base frame relative to {W}, ${}^W\boldsymbol{\omega}_B|_W$, to the first time derivative of the Euler angles, ${}^W\dot{\mathbf{x}}_{B(o)}|_E$, is given by

$${}^W\boldsymbol{\omega}_B|_W = \mathbf{J}_G \cdot {}^W\dot{\mathbf{x}}_{B(o)}|_E \tag{17}$$

Considering equation (16), in frame {B}, the following equation can be written:

$$ {}^{W}\mathbf{v}_{P}|_{B} = {}^{W}\mathbf{v}_{B}|_{B} + {}^{B}\mathbf{v}_{P}|_{B} + {}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\mathbf{p}_{P}|_{B} \tag{18} $$

Therefore, the total linear momentum of {P} expressed in frame {B} will be

$$ \mathbf{Q}_{P(tot)}|_{B} = m_{P} \cdot {}^{W}\mathbf{v}_{P}|_{B} \tag{19} $$

m_P being the payload platform mass.
Taking the time derivative of the previous equation results in

$$ \dot{\mathbf{Q}}_{P(tot)}|_{B} = m_{P} \cdot {}^{W}\dot{\mathbf{v}}_{P}|_{B} \tag{20} $$

Knowing that,

$$ {}^{W}\boldsymbol{\omega}_{P}|_{B} = {}^{B}\boldsymbol{\omega}_{P}|_{B} + {}^{W}\boldsymbol{\omega}_{B}|_{B} \tag{21} $$

$$ {}^{W}\dot{\boldsymbol{\omega}}_{P}|_{B} = {}^{B}\dot{\boldsymbol{\omega}}_{P}|_{B} + {}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\boldsymbol{\omega}_{P}|_{B} + {}^{W}\dot{\boldsymbol{\omega}}_{B}|_{B} \tag{22} $$

results in

$$ {}^{W}\dot{\mathbf{v}}_{P}|_{B} = {}^{W}\dot{\mathbf{v}}_{B}|_{B} + {}^{B}\dot{\mathbf{v}}_{P}|_{B} + 2\,{}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\mathbf{v}_{P}|_{B} + {}^{W}\dot{\boldsymbol{\omega}}_{B}|_{B} \times {}^{B}\mathbf{p}_{P}|_{B} + {}^{W}\boldsymbol{\omega}_{B}|_{B} \times \left({}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\mathbf{p}_{P}|_{B} \right) \tag{23} $$

The inertial part of the total force applied in {P}, due to the payload platform translation, expressed in frame {B} will be

$$ {}^{P}\mathbf{F}_{P(ine)(tot)}|_{B} = \dot{\mathbf{Q}}_{P(tot)}|_{B} \tag{24} $$

That is,

$$ {}^{P}\mathbf{F}_{P(ine)(tot)}|_{B} = {}^{P}\mathbf{F}_{P(ine)(fix)}|_{B} + {}^{P}\mathbf{F}_{P(ine)(man)}|_{B} \tag{25} $$

$$ {}^{P}\mathbf{F}_{P(ine)(fix)}|_{B} = m_{P} \cdot {}^{B}\dot{\mathbf{v}}_{P}|_{B} \tag{26} $$

$$ {}^{P}\mathbf{F}_{P(ine)(man)}|_{B} = m_{P} \cdot \left({}^{W}\dot{\mathbf{v}}_{B}|_{B} + 2\,{}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\mathbf{v}_{P}|_{B} + {}^{W}\dot{\boldsymbol{\omega}}_{B}|_{B} \times {}^{B}\mathbf{p}_{P}|_{B} + {}^{W}\boldsymbol{\omega}_{B}|_{B} \times \left({}^{W}\boldsymbol{\omega}_{B}|_{B} \times {}^{B}\mathbf{p}_{P}|_{B} \right) \right) \tag{27} $$

where ${}^{P}\mathbf{F}_{P(ine)(fix)}|_{B}$ represents the inertial part of the force, considering the base platform is not moving, and ${}^{P}\mathbf{F}_{P(ine)(man)}|_{B}$ represents the inertial part of the force which results from the base motion.

On the other hand, the angular momentum of the moving platform, about its centre of mass and expressed in frame {B} will be

$$\mathbf{H}_{P(tot)}|_B = \mathbf{I}_{P(rot)}|_B \cdot {}^W\boldsymbol{\omega}_P|_B \tag{28}$$

$\mathbf{I}_{P(rot)}|_B$ represents the rotational inertia matrix of the moving platform, expressed in the base frame, {B}. This inertia matrix is given by:

$$\mathbf{I}_{P(rot)}|_B = {}^B\mathbf{R}_P \cdot \mathbf{I}_{P(rot)}|_P \cdot {}^B\mathbf{R}_P^T \tag{29}$$

where $\mathbf{I}_{P(rot)}|_P$ is a constant matrix representing the rotational inertia matrix of the moving platform, expressed in frame {P}. Considering that $I_{P_{xx}}$, $I_{P_{yy}}$ and $I_{P_{zz}}$ are the moments of inertia of the moving platform expressed in its own frame, this matrix may be written as:

$$\mathbf{I}_{P(rot)}|_P = \operatorname{diag}([I_{P_{xx}} \quad I_{P_{yy}} \quad I_{P_{zz}}]) \tag{30}$$

Taking the time derivative of equation (28) results in

$$\dot{\mathbf{H}}_{P(tot)}|_B = \dot{\mathbf{I}}_{P(rot)}|_B \cdot {}^W\boldsymbol{\omega}_P|_B + \mathbf{I}_{P(rot)}|_B \cdot {}^W\dot{\boldsymbol{\omega}}_P|_B \tag{31}$$

The inertial part of the total moment applied in {P}, due to the payload platform rotation, expressed in frame {B} will be

$${}^P\mathbf{M}_{P(ine)(tot)}|_B = \dot{\mathbf{H}}_{P(tot)}|_B \tag{32}$$

that is,

$${}^P\mathbf{M}_{P(ine)(tot)}|_B = {}^P\mathbf{M}_{P(ine)(fix)}|_B + {}^P\mathbf{M}_{P(ine)(man)}|_B \tag{33}$$

where,

$${}^P\mathbf{M}_{P(ine)(fix)}|_B = \mathbf{I}_{P(rot)}|_B \cdot {}^B\dot{\boldsymbol{\omega}}_P|_B + \dot{\mathbf{I}}_{P(rot)}|_B \cdot {}^B\boldsymbol{\omega}_P|_B \tag{34}$$

$${}^P\mathbf{M}_{P(ine)(man)}|_B = \dot{\mathbf{I}}_{P(rot)}|_B \cdot {}^W\boldsymbol{\omega}_B|_B + \mathbf{I}_{P(rot)}|_B \cdot \left({}^W\dot{\boldsymbol{\omega}}_B|_B + {}^W\boldsymbol{\omega}_B|_B \times {}^B\boldsymbol{\omega}_P|_B\right) \tag{35}$$

${}^P\mathbf{M}_{P(ine)(fix)}|_B$ represents the inertial part of the moment, considering the base platform is not moving, and ${}^P\mathbf{M}_{P(ine)(man)}|_B$ represents the inertial part of the moment which results from the base motion.

The total inertial component of the generalized force applied to {P} and expressed in {B} will be

$$^{P}\mathbf{f}_{P(ine)(tot)}\,|_{B} = [\,^{P}\mathbf{F}^{T}_{P(ine)(tot)}\,|_{B} \quad ^{P}\mathbf{M}^{T}_{P(ine)(tot)}\,|_{B}\,]^{T} \tag{36}$$

The inertial components of the forces in the manipulator actuators (actuating forces) will be

$$\boldsymbol{\tau}_{P(ine)(fix)} = \mathbf{J}_{C}^{-T} \cdot {}^{P}\mathbf{f}_{P(ine)(fix)}\,|_{B} \tag{37}$$

$$\boldsymbol{\tau}_{P(ine)(man)} = \mathbf{J}_{C}^{-T} \cdot {}^{P}\mathbf{f}_{P(ine)(man)}\,|_{B} \tag{38}$$

On the other hand, regarding the gravitational part of the generalized force, if the base platform orientation changes, then the force applied to {P} and expressed in {B} results in

$$^{P}\mathbf{f}_{P(gra)}\,|_{B} = {}^{W}\Re_{B}^{T} \cdot {}^{P}\mathbf{f}_{P(gra)}\,|_{W} \tag{39}$$

where,

$$^{W}\Re_{B} = \begin{bmatrix} ^{W}\mathbf{R}_{B} & \mathbf{0} \\ \mathbf{0} & ^{W}\mathbf{R}_{B} \end{bmatrix} \tag{40}$$

and $^{P}\mathbf{f}_{P(gra)}\,|_{W}$ is the gravitational generalized force applied to {P} and expressed in {W}. This force can be computed using a simplified model that considers both a non-moving base platform, frame {B} parallel to {W}, and $\mathbf{z}_{B} \equiv -\hat{\mathbf{g}}$, i.e.,

$$^{P}\mathbf{f}_{P(gra)}\,|_{W} = \frac{\partial P_{P}\left(^{B}\mathbf{x}_{P}\,|_{B|E}\right)}{\partial\, ^{B}\mathbf{x}_{P}\,|_{B|E}} \tag{41}$$

$P_{P} = m_{P} \cdot g \cdot z_{P}$ representing the mobile platform potential energy.
The gravitational component of the actuating forces due to the moving platform, $\boldsymbol{\tau}_{P(gra)}$, is given by equation (42), which can be added to equations (37) and (38).

$$\boldsymbol{\tau}_{P(gra)} = \mathbf{J}_{C}^{-T} \cdot {}^{P}\mathbf{f}_{P(gra)}\,|_{B} \tag{42}$$

3.2 Cylinder Modelling

Position of the cylinder *i*, relative to {W} and expressed in {W}, may be computed using the following equation:

$$^{W}\mathbf{p}_{C_i}\,|_{W} = {}^{W}\mathbf{p}_{B}\,|_{W} + {}^{B}\mathbf{p}_{C_i}\,|_{W} \tag{43}$$

The linear velocity of the cylinder, relative to {W} and expressed in {W}, may be obtained taking the time derivative of the previous equation, that is,

$$^{W}\mathbf{v}_{C_i}|_W = {}^{W}\mathbf{v}_B|_W + {}^{B}\mathbf{v}_{C_i}|_W + {}^{W}\boldsymbol{\omega}_B|_W \times {}^{B}\mathbf{p}_{C_i}|_W \tag{44}$$

Considering that frame {C*i*} is attached to the cylinder *i* and positioned at its centre of mass, then $^{B}\mathbf{v}_{C_i}|_W$ is the linear velocity of frame {C*i*} as seen by an observer fixed in {B}, and $^{B}\mathbf{p}_{C_i}|_W$ represents the position of {C*i*} relative to {B} and expressed in {W}.
In frame {B} the following equation can be written:

$$^{W}\mathbf{v}_{C_i}|_B = {}^{W}\mathbf{v}_B|_B + {}^{B}\mathbf{v}_{C_i}|_B + {}^{W}\boldsymbol{\omega}_B|_B \times {}^{B}\mathbf{p}_{C_i}|_B \tag{45}$$

Considering the centre of mass of each cylinder is located at a constant distance, b_C, from the cylinder to base platform connecting point, B_i, (Figure 3), then its position relative to frame {B} is

$$^{B}\mathbf{p}_{C_i}|_B = b_C \cdot \hat{\mathbf{l}}_i + \mathbf{b}_i \tag{46}$$

where,

$$\hat{\mathbf{l}}_i = \frac{\mathbf{l}_i}{\|\mathbf{l}_i\|} = \frac{\mathbf{l}_i}{l_i} \tag{47}$$

$$\mathbf{l}_i = {}^{B}\mathbf{x}_{P(pos)}|_B + {}^{P}\mathbf{p}_i|_B - \mathbf{b}_i \tag{48}$$

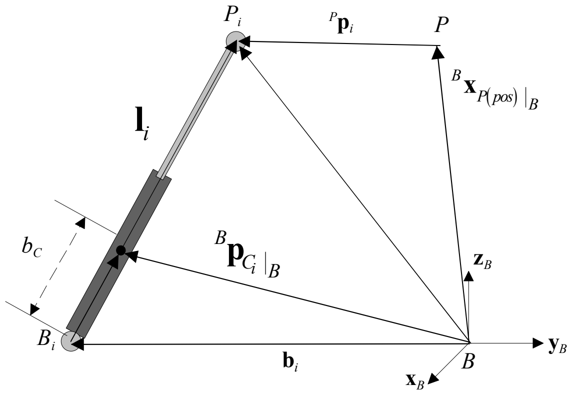


Fig. 3. Position of the centre of mass of the cylinder *i*

The linear velocity of the cylinder centre of mass, $^{B}\dot{\mathbf{p}}_{C_i}|_B$, relative to {B} and expressed in the same frame, may be computed as:

$$^{B}\dot{\mathbf{p}}_{C_i}|_B = {}^{B}\boldsymbol{\omega}_{l_i}|_B \times b_C \cdot \hat{\mathbf{l}}_i \tag{49}$$

where $^{B}\boldsymbol{\omega}_{l_i}|_B$ represents the leg angular velocity, which can be found from:

$$ {}^{B}\boldsymbol{\omega}_{l_i}|_B \times \mathbf{l}_i = {}^{B}\mathbf{v}_P|_B + {}^{B}\boldsymbol{\omega}_P|_B \times {}^{P}\mathbf{p}_i|_B \tag{50} $$

As the leg (both the cylinder and piston) cannot rotate along its own axis, the angular velocity along $\hat{\mathbf{l}}_i$ is always zero, and vectors $\mathbf{l}_i$ and ${}^{B}\boldsymbol{\omega}_{l_i}|_B$ are always perpendicular. This enables equation (50) to be rewritten as:

$$ {}^{B}\boldsymbol{\omega}_{l_i}|_B = \frac{1}{\mathbf{l}_i^T \cdot \mathbf{l}_i} \cdot \left[\mathbf{l}_i \times \left({}^{B}\mathbf{v}_P|_B + {}^{B}\boldsymbol{\omega}_P|_B \times {}^{P}\mathbf{p}_i|_B\right)\right] \tag{51} $$

or,

$$ {}^{B}\boldsymbol{\omega}_{l_i}|_B = \mathbf{J}_{D_i} \cdot \begin{bmatrix} {}^{B}\mathbf{v}_P|_B \\ {}^{B}\boldsymbol{\omega}_P|_B \end{bmatrix} \tag{52} $$

where jacobian $\mathbf{J}_{D_i}$ is given by:

$$ \mathbf{J}_{D_i} = \begin{bmatrix} \tilde{\bar{\mathbf{l}}}_i & \tilde{\bar{\mathbf{l}}}_i \cdot {}^{P}\tilde{\mathbf{p}}_i^T|_B \end{bmatrix} \tag{53} $$

$$ \bar{\mathbf{l}}_i = \frac{\mathbf{l}_i}{\mathbf{l}_i^T \cdot \mathbf{l}_i} \tag{54} $$

and, for a given vector $\mathbf{a} = \begin{bmatrix} a_x & a_y & a_z \end{bmatrix}^T$,

$$ \tilde{\mathbf{a}} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \tag{55} $$

On the other hand, equation (49) can be rewritten as:

$$ {}^{B}\dot{\mathbf{p}}_{C_i}|_B = \mathbf{J}_{B_i} \cdot \begin{bmatrix} {}^{B}\mathbf{v}_P|_B \\ {}^{B}\boldsymbol{\omega}_P|_B \end{bmatrix} \tag{56} $$

where the jacobian $\mathbf{J}_{B_i}$ is given by:

$$ \mathbf{J}_{B_i} = \begin{bmatrix} b_C \cdot \tilde{\hat{\mathbf{l}}}_i^T \cdot \tilde{\bar{\mathbf{l}}}_i & b_C \cdot \tilde{\hat{\mathbf{l}}}_i^T \cdot \tilde{\bar{\mathbf{l}}}_i \cdot {}^{P}\tilde{\mathbf{p}}_i^T|_B \end{bmatrix} \tag{57} $$

The total linear momentum of the cylinder *i*, expressed in frame {B} will be

$$\mathbf{Q}_{C_i(tot)}|_B = m_C \cdot {}^W\mathbf{v}_{C_i}|_B \tag{58}$$

m_C being the cylinder mass.
Taking the time derivative of the previous equation results in

$$\dot{\mathbf{Q}}_{C_i(tot)}|_B = m_C \cdot {}^W\dot{\mathbf{v}}_{C_i}|_B \tag{59}$$

where,

$${}^W\dot{\mathbf{v}}_{C_i}|_B = {}^W\dot{\mathbf{v}}_B|_B + {}^B\dot{\mathbf{v}}_{C_i}|_B + 2\,{}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{v}_{C_i}|_B + {}^W\dot{\boldsymbol{\omega}}_B|_B \times {}^B\mathbf{p}_{C_i}|_B + {}^W\boldsymbol{\omega}_B|_B \times \left({}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{p}_{C_i}|_B\right) \tag{60}$$

The inertial part of the total force applied in {C$_i$}, due to the cylinder translation, expressed in frame {B} will be

$${}^{C_i}\mathbf{F}_{C_i(ine)(tot)}|_B = \dot{\mathbf{Q}}_{C_i(tot)}|_B \tag{61}$$

That is,

$${}^{C_i}\mathbf{F}_{C_i(ine)(tot)}|_B = {}^{C_i}\mathbf{F}_{C_i(ine)(fix)}|_B + {}^{C_i}\mathbf{F}_{C_i(ine)(man)}|_B \tag{62}$$

$${}^{C_i}\mathbf{F}_{C_i(ine)(fix)}|_B = m_C \cdot {}^B\dot{\mathbf{v}}_{C_i}|_B = m_C \cdot \dot{\mathbf{J}}_{B_i} \cdot {}^B\dot{\mathbf{x}}_P|_B + m_C \cdot \mathbf{J}_{B_i} \cdot {}^B\ddot{\mathbf{x}}_P|_B \tag{63}$$

$${}^{C_i}\mathbf{F}_{C_i(ine)(man)}|_B = m_C \cdot \left({}^W\dot{\mathbf{v}}_B|_B + 2\,{}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{v}_{C_i}|_B + {}^W\dot{\boldsymbol{\omega}}_B|_B \times {}^B\mathbf{p}_{C_i}|_B + {}^W\boldsymbol{\omega}_B|_B \times \left({}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{p}_{C_i}|_B\right)\right) \tag{64}$$

where ${}^{C_i}\mathbf{F}_{C_i(ine)(fix)}|_B$ represents the inertial part of the force, considering the base platform is not moving, and ${}^{C_i}\mathbf{F}_{C_i(ine)(man)}|_B$ represents the inertial part of the force which results from the base motion.
When equation (61) is pre-multiplied by $\mathbf{J}_{B_i}^T$, the inertial component of the generalized force applied to {P}, due to each cylinder translation is obtained in frame {B}:

$${}^P\mathbf{f}_{C_i(ine)(tot)(tra)}|_B = \mathbf{J}_{B_i}^T \cdot {}^{C_i}\mathbf{F}_{C_i(ine)(tot)}|_B \tag{65}$$

The inertial components of the actuating forces will be

$$\boldsymbol{\tau}_{C_i(ine)(fix)(tra)} = \mathbf{J}_C^{-T} \cdot {}^P\mathbf{f}_{C_i(ine)(fix)(tra)}|_B \tag{66}$$

$$\boldsymbol{\tau}_{C_i(ine)(man)(tra)} = \mathbf{J}_C^{-T} \cdot {}^P\mathbf{f}_{C_i(ine)(man)(tra)}|_B \tag{67}$$

On the other hand, the total angular momentum of the cylinder about its centre of mass and expressed in frame {B} will be:

$$\mathbf{H}_{C_i(tot)}\,|_B = \mathbf{I}_{C_i(rot)}\,|_B \cdot {}^{W}\boldsymbol{\omega}_{C_i}\,|_B \tag{68}$$

Taking the time derivative of the previous equation results in

$$\dot{\mathbf{H}}_{C_i(tot)}\,|_B = \dot{\mathbf{I}}_{C_i(rot)}\,|_B \cdot {}^{W}\boldsymbol{\omega}_{C_i}\,|_B + \mathbf{I}_{C_i(rot)}\,|_B \cdot {}^{W}\dot{\boldsymbol{\omega}}_{C_i}\,|_B \tag{69}$$

where,

$${}^{W}\boldsymbol{\omega}_{C_i}\,|_B = {}^{B}\boldsymbol{\omega}_{C_i}\,|_B + {}^{W}\boldsymbol{\omega}_{B}\,|_B \tag{70}$$

$${}^{W}\dot{\boldsymbol{\omega}}_{C_i}\,|_B = {}^{B}\dot{\boldsymbol{\omega}}_{C_i}\,|_B + {}^{W}\boldsymbol{\omega}_{B}\,|_B \times {}^{B}\boldsymbol{\omega}_{C_i}\,|_B + {}^{W}\dot{\boldsymbol{\omega}}_{B}\,|_B \tag{71}$$

Considering that $\mathbf{I}_{C_i(rot)}\,|_{C_i}$ is the inertia constant matrix of the rotating cylinder *i*, expressed in the frame fixed to the cylinder itself, {C*i*}≡{ $\mathbf{x}_{C_i}$,$\mathbf{y}_{C_i}$,$\mathbf{z}_{C_i}$ }, then

$$\mathbf{I}_{C_i(rot)}\,|_B = {}^{B}\mathbf{R}_{C_i} \cdot \mathbf{I}_{C_i(rot)}\,|_{C_i} \cdot {}^{B}\mathbf{R}_{C_i}^{T} \tag{72}$$

where ${}^{B}\mathbf{R}_{C_i}$ is the orientation matrix of each cylinder frame, {C*i*}, relative to the base frame, {B}.

Cylinder frames were chosen in the following way: axis $\mathbf{x}_{C_i}$ coincides with the leg axis and points towards *Pi*; axis $\mathbf{y}_{C_i}$ is perpendicular to $\mathbf{x}_{C_i}$ and always parallel to the base plane, this condition being possible given the existence of a universal joint at point *Bi*, that negates any rotation along its own axis; axis $\mathbf{z}_{C_i}$ completes the referential following the right hand rule, and its projection along axis $\mathbf{z}_B$ is always positive. Thus, matrix ${}^{B}\mathbf{R}_{C_i}$ becomes:

$${}^{B}\mathbf{R}_{C_i} = \begin{bmatrix}\mathbf{x}_{C_i} & \mathbf{y}_{C_i} & \mathbf{z}_{C_i}\end{bmatrix} \tag{73}$$

where

$$\mathbf{x}_{C_i} = \hat{\mathbf{l}}_i \tag{74}$$

$$\mathbf{y}_{C_i} = \begin{bmatrix} -\dfrac{l_{iy}}{\sqrt{l_{ix}^2 + l_{iy}^2}} & \dfrac{l_{ix}}{\sqrt{l_{ix}^2 + l_{iy}^2}} & 0 \end{bmatrix}^T \tag{75}$$

$$\mathbf{z}_{C_i} = \mathbf{x}_{C_i} \times \mathbf{y}_{C_i} \tag{76}$$

So, the inertia matrices of the cylinders can be written as:

$$\mathbf{I}_{C_i(rot)}\,|_{C_i} = \mathrm{diag}([I_{C_{xx}} \quad I_{C_{yy}} \quad I_{C_{zz}}]) \tag{77}$$

where $I_{C_{xx}}$, $I_{C_{yy}}$ and $I_{C_{zz}}$ are the cylinder moments of inertia expressed in its own frame.

The inertial part of the total moment applied in {C$_i$}, due to the cylinder rotation, expressed in frame {B} will be

$$^{C_i}\mathbf{M}_{C_i(ine)(tot)}|_B = \dot{\mathbf{H}}_{C_i(tot)}|_B \tag{78}$$

that is,

$$^{C_i}\mathbf{M}_{C_i(ine)(tot)}|_B = {}^{C_i}\mathbf{M}_{C_i(ine)(fix)}|_B + {}^{C_i}\mathbf{M}_{C_i(ine)(man)}|_B \tag{79}$$

where,

$$^{C_i}\mathbf{M}_{C_i(ine)(fix)}|_B = \frac{d}{dt}\left(\mathbf{I}_{C_i(rot)}|_B \cdot \mathbf{J}_{D_i}\right) \cdot {}^B\dot{\mathbf{x}}_P|_B + \mathbf{I}_{C_i(rot)}|_B \cdot \mathbf{J}_{D_i} \cdot {}^B\ddot{\mathbf{x}}_P|_B \tag{80}$$

$$^{C_i}\mathbf{M}_{C_i(ine)(man)}|_B = \dot{\mathbf{I}}_{C_i(rot)}|_B \cdot {}^W\boldsymbol{\omega}_B|_B + \mathbf{I}_{C_i(rot)}|_B \cdot \left({}^W\dot{\boldsymbol{\omega}}_B|_B + {}^W\boldsymbol{\omega}_B|_B \times {}^B\boldsymbol{\omega}_{C_i}|_B\right) \tag{81}$$

$^{C_i}\mathbf{M}_{C_i(ine)(fix)}|_B$ represents the inertial part of the moment, considering the base platform is not moving, and $^{C_i}\mathbf{M}_{C_i(ine)(man)}|_B$ represents the inertial part of the moment which results from the base motion.

When equation (78) is pre-multiplied by $\mathbf{J}^T_{D_i}$, the inertial component of the generalized force applied to {P}, due to each cylinder rotation is obtained in frame {B}:

$$^{P}\mathbf{f}_{C_i(ine)(tot)(rot)}|_B = \mathbf{J}^T_{D_i} \cdot {}^{C_i}\mathbf{M}_{C_i(ine)(tot)}|_B \tag{82}$$

The inertial components of the actuating forces will be

$$\boldsymbol{\tau}_{C_i(ine)(fix)(rot)} = \mathbf{J}^{-T}_C \cdot {}^P\mathbf{f}_{C_i(ine)(fix)(rot)}|_B \tag{83}$$

$$\boldsymbol{\tau}_{C_i(ine)(man)(rot)} = \mathbf{J}^{-T}_C \cdot {}^P\mathbf{f}_{C_i(ine)(man)(rot)}|_B \tag{84}$$

Now, with reference to the gravitational part of the generalized force, if the base platform orientation changes, then the force applied in {P} and expressed in {B} results in

$$^{P}\mathbf{f}_{C_i(gra)}|_B = {}^W\Re^T_B \cdot {}^P\mathbf{f}_{C_i(gra)}|_W \tag{85}$$

where $^{P}\mathbf{f}_{C_i(gra)}|_W$ is the gravitational generalized force applied in {P} and expressed in {W}. This force can be computed using the model that considers both a non-moving base platform, frame {B} parallel to {W}, and $\mathbf{z}_B \equiv -\hat{\mathbf{g}}$, i.e.,

$$^{P}\mathbf{f}_{C_i(gra)}|_W = \frac{\partial \mathcal{P}_{C_i}\left({}^B\mathbf{x}_P|_{B|E}\right)}{\partial\, {}^B\mathbf{x}_P|_{B|E}} \tag{86}$$

P_{C_i} being the cylinder potential energy. Using equation (46) it can be computed by:

$$P_{C_i} = m_C \cdot g \cdot \frac{b_C}{l_i}\left(z_P + {}^P p_i\,|_{Bz}\right) \tag{87}$$

The gravitational component of the actuating forces due to each cylinder, $\boldsymbol{\tau}_{C_i(gra)}$, is

$$\boldsymbol{\tau}_{C_i(gra)} = \mathbf{J}_C^{-T} \cdot {}^P\mathbf{f}_{C_i(gra)}\,|_B \tag{88}$$

3.3. Piston Modelling

Position of the piston *i*, relative to {W} and expressed in {W}, may be computed using the following equation:

$${}^W\mathbf{p}_{S_i}\,|_W = {}^W\mathbf{p}_B\,|_W + {}^B\mathbf{p}_{S_i}\,|_W \tag{89}$$

The linear velocity of the piston, relative to {W} and expressed in {W}, may be obtained taking the time derivative of the previous equation, that is,

$${}^W\mathbf{v}_{S_i}\,|_W = {}^W\mathbf{v}_B\,|_W + {}^B\mathbf{v}_{S_i}\,|_W + {}^W\boldsymbol{\omega}_B\,|_W \times {}^B\mathbf{p}_{S_i}\,|_W \tag{90}$$

Considering that frame {S*i*} is attached to the piston *i* and positioned at its centre of mass, then ${}^B\mathbf{v}_{S_i}\,|_W$ is the linear velocity of frame {S*i*} as seen by an observer fixed in {B}, and ${}^B\mathbf{p}_{S_i}\,|_W$ represents the position of {S*i*} relative to {B} and expressed in {W}.
In frame {B} the following equation can be written:

$${}^W\mathbf{v}_{S_i}\,|_B = {}^W\mathbf{v}_B\,|_B + {}^B\mathbf{v}_{S_i}\,|_B + {}^W\boldsymbol{\omega}_B\,|_B \times {}^B\mathbf{p}_{S_i}\,|_B \tag{91}$$

If the centre of mass of each piston is located at a constant distance, b_S, from the piston to moving platform connecting point, P_i, (Figure 4), then its position relative to frame {B} is:

$${}^B\mathbf{p}_{S_i}\,|_B = -b_S \cdot \hat{\mathbf{l}}_i + {}^B\mathbf{p}_i\,|_B + {}^B\mathbf{x}_{P(pos)}\,|_B \tag{92}$$

The linear velocity of the piston centre of mass, ${}^B\dot{\mathbf{p}}_{S_i}\,|_B$, relative to {B} and expressed in the same frame, may be computed as:

$${}^B\dot{\mathbf{p}}_{S_i}\,|_B = \dot{\mathbf{l}}_i + {}^B\boldsymbol{\omega}_{l_i}\,|_B \times \left(-b_S \cdot \hat{\mathbf{l}}_i\right) \tag{93}$$

$$ {}^{B}\dot{\mathbf{p}}_{S_i}|_B = \mathbf{J}_{G_i} \cdot \begin{bmatrix} {}^{B}\mathbf{v}_P|_B \\ {}^{B}\boldsymbol{\omega}_P|_B \end{bmatrix} \tag{94} $$

where the jacobian $\mathbf{J}_{G_i}$ is given by:

$$ \mathbf{J}_{G_i} = \begin{bmatrix} \mathbf{I} - b_S \cdot \tilde{\tilde{\hat{\mathbf{l}}}}_i^T \cdot \tilde{\tilde{\hat{\mathbf{l}}}}_i & \left(\mathbf{I} - b_S \cdot \tilde{\tilde{\hat{\mathbf{l}}}}_i^T \cdot \tilde{\tilde{\hat{\mathbf{l}}}}_i\right) \cdot {}^{P}\tilde{\mathbf{p}}_i^T|_B \end{bmatrix} \tag{95} $$

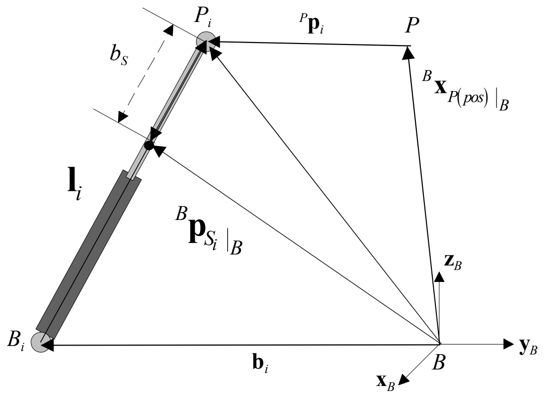


Fig. 4. Position of the centre of mass of the piston *i*

The total linear momentum of the piston *i*, expressed in frame {B} will be

$$ \mathbf{Q}_{S_i(tot)}|_B = m_S \cdot {}^{W}\mathbf{v}_{S_i}|_B \tag{96} $$

m_S being the piston mass.
Taking the time derivative of the previous equation results in

$$ \dot{\mathbf{Q}}_{S_i(tot)}|_B = m_S \cdot {}^{W}\dot{\mathbf{v}}_{S_i}|_B \tag{97} $$

where,

$$ {}^{W}\dot{\mathbf{v}}_{S_i}|_B = {}^{W}\dot{\mathbf{v}}_B|_B + {}^{B}\dot{\mathbf{v}}_{S_i}|_B + 2\,{}^{W}\boldsymbol{\omega}_B|_B \times {}^{B}\mathbf{v}_{S_i}|_B + {}^{W}\dot{\boldsymbol{\omega}}_B|_B \times {}^{B}\mathbf{p}_{S_i}|_B + {}^{W}\boldsymbol{\omega}_B|_B \times \left({}^{W}\boldsymbol{\omega}_B|_B \times {}^{B}\mathbf{p}_{S_i}|_B\right) \tag{98} $$

The inertial part of the total force applied in {S*i*}, due to the piston translation, expressed in frame {B} will be

$$ {}^{S_i}\mathbf{F}_{S_i(ine)(tot)}|_B = \dot{\mathbf{Q}}_{S_i(tot)}|_B \tag{99} $$

That is,

$$ {}^{S_i}\mathbf{F}_{S_i(ine)(tot)}|_B = {}^{S_i}\mathbf{F}_{S_i(ine)(fix)}|_B + {}^{S_i}\mathbf{F}_{S_i(ine)(man)}|_B \tag{100} $$

$$ {}^{S_i}\mathbf{F}_{S_i(ine)(fix)}|_B = m_S \cdot {}^B\dot{\mathbf{v}}_{S_i}|_B = m_S \cdot \dot{\mathbf{J}}_{G_i} \cdot {}^B\dot{\mathbf{x}}_P|_B + m_S \cdot \mathbf{J}_{G_i} \cdot {}^B\ddot{\mathbf{x}}_P|_B \tag{101} $$

$$ {}^{S_i}\mathbf{F}_{S_i(ine)(man)}|_B = m_S \cdot \left({}^W\dot{\mathbf{v}}_B|_B + 2\,{}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{v}_{S_i}|_B + {}^W\dot{\boldsymbol{\omega}}_B|_B \times {}^B\mathbf{p}_{S_i}|_B + {}^W\boldsymbol{\omega}_B|_B \times \left({}^W\boldsymbol{\omega}_B|_B \times {}^B\mathbf{p}_{S_i}|_B \right) \right) \tag{102} $$

where ${}^{S_i}\mathbf{F}_{S_i(ine)(fix)}|_B$ represents the inertial part of the force, considering the base platform is not moving, and ${}^{S_i}\mathbf{F}_{S_i(ine)(man)}|_B$ represents the inertial part of the force which results from the base motion.
When equation (99) is pre-multiplied by $\mathbf{J}_{G_i}^T$, the inertial component of the generalized force applied to {P}, due to each piston translation is obtained in frame {B}:

$$ {}^P\mathbf{f}_{S_i(ine)(tot)(tra)}|_B = \mathbf{J}_{G_i}^T \cdot {}^{S_i}\mathbf{F}_{S_i(ine)(tot)}|_B \tag{103} $$

The inertial components of the actuating forces will be

$$ \boldsymbol{\tau}_{S_i(ine)(fix)(tra)} = \mathbf{J}_C^{-T} \cdot {}^P\mathbf{f}_{S_i(ine)(fix)(tra)}|_B \tag{104} $$

$$ \boldsymbol{\tau}_{S_i(ine)(man)(tra)} = \mathbf{J}_C^{-T} \cdot {}^P\mathbf{f}_{S_i(ine)(man)(tra)}|_B \tag{105} $$

On the other hand, the total angular momentum of the piston about its centre of mass, expressed in frame {B}, will be:

$$ \mathbf{H}_{S_i(tot)}|_B = \mathbf{I}_{S_i(rot)}|_B \cdot {}^W\boldsymbol{\omega}_{S_i}|_B \tag{106} $$

Taking the time derivative of the previous equation results in

$$ \dot{\mathbf{H}}_{S_i(tot)}|_B = \dot{\mathbf{I}}_{S_i(rot)}|_B \cdot {}^W\boldsymbol{\omega}_{S_i}|_B + \mathbf{I}_{S_i(rot)}|_B \cdot {}^W\dot{\boldsymbol{\omega}}_{S_i}|_B \tag{107} $$

where,

$$ {}^W\boldsymbol{\omega}_{S_i}|_B = {}^B\boldsymbol{\omega}_{S_i}|_B + {}^W\boldsymbol{\omega}_B|_B \tag{108} $$

$$ {}^W\dot{\boldsymbol{\omega}}_{S_i}|_B = {}^B\dot{\boldsymbol{\omega}}_{S_i}|_B + {}^W\boldsymbol{\omega}_B|_B \times {}^B\boldsymbol{\omega}_{S_i}|_B + {}^W\dot{\boldsymbol{\omega}}_B|_B \tag{109} $$

Considering that $\mathbf{I}_{S_i(rot)}|_{S_i}$ is the inertia constant matrix of the rotating piston *i*, expressed in the frame fixed to the piston itself, {S*i*}, then

$$ \mathbf{I}_{S_i(rot)}|_B = {}^B\mathbf{R}_{S_i} \cdot \mathbf{I}_{S_i(rot)}|_{S_i} \cdot {}^B\mathbf{R}_{S_i}^T \tag{110} $$

where ${}^B\mathbf{R}_{S_i}$ is the orientation matrix of each piston frame, {S*i*}, relative to the base frame, {B}.

As the relative motion between cylinder and piston is a pure translation, {S$_i$} can be chosen parallel to {C$_i$}. Therefore, ${}^{B}\mathbf{R}_{S_i} = {}^{B}\mathbf{R}_{C_i}$.

So, the inertia matrices of the pistons can be written as:

$$\mathbf{I}_{S_i(rot)}\big|_{S_i} = \text{diag}([I_{S_{xx}} \quad I_{S_{yy}} \quad I_{S_{zz}}]) \tag{111}$$

where $I_{S_{xx}}$, $I_{S_{yy}}$ and $I_{S_{zz}}$ are the piston moments of inertia expressed in its own frame.

The inertial part of the total moment applied in {S$_i$}, due to the piston rotation, expressed in frame {B} will be

$${}^{S_i}\mathbf{M}_{S_i(ine)(tot)}\big|_B = \dot{\mathbf{H}}_{S_i(tot)}\big|_B \tag{112}$$

that is,

$${}^{S_i}\mathbf{M}_{S_i(ine)(tot)}\big|_B = {}^{S_i}\mathbf{M}_{S_i(ine)(fix)}\big|_B + {}^{S_i}\mathbf{M}_{S_i(ine)(man)}\big|_B \tag{113}$$

where,

$${}^{S_i}\mathbf{M}_{S_i(ine)(fix)}\big|_B = \frac{d}{dt}\left(\mathbf{I}_{S_i(rot)}\big|_B \cdot \mathbf{J}_{D_i}\right) {}^{B}\dot{\mathbf{x}}_P\big|_B + \mathbf{I}_{S_i(rot)}\big|_B \cdot \mathbf{J}_{D_i} \cdot {}^{B}\ddot{\mathbf{x}}_P\big|_B \tag{114}$$

$${}^{S_i}\mathbf{M}_{S_i(ine)(man)}\big|_B = \dot{\mathbf{I}}_{S_i(rot)}\big|_B \cdot {}^{W}\boldsymbol{\omega}_B\big|_B + \mathbf{I}_{S_i(rot)}\big|_B \cdot \left({}^{W}\dot{\boldsymbol{\omega}}_B\big|_B + {}^{W}\boldsymbol{\omega}_B\big|_B \times {}^{B}\boldsymbol{\omega}_{S_i}\big|_B\right) \tag{115}$$

${}^{S_i}\mathbf{M}_{S_i(ine)(fix)}\big|_B$ represents the inertial part of the moment, considering the base platform is not moving, and ${}^{S_i}\mathbf{M}_{S_i(ine)(man)}\big|_B$ represents the inertial part of the moment which results from the base motion.

When equation (112) is pre-multiplied by $\mathbf{J}_{D_i}^{T}$, the inertial component of the generalized force applied to {P}, due to each piston rotation is obtained in frame {B}:

$${}^{P}\mathbf{f}_{S_i(ine)(tot)(rot)}\big|_B = \mathbf{J}_{D_i}^{T} \cdot {}^{S_i}\mathbf{M}_{S_i(ine)(tot)}\big|_B \tag{116}$$

The inertial components of the actuating forces will be

$$\boldsymbol{\tau}_{S_i(ine)(fix)(rot)} = \mathbf{J}_C^{-T} \cdot {}^{P}\mathbf{f}_{S_i(ine)(fix)(rot)}\big|_B \tag{117}$$

$$\boldsymbol{\tau}_{S_i(ine)(man)(rot)} = \mathbf{J}_C^{-T} \cdot {}^{P}\mathbf{f}_{S_i(ine)(man)(rot)}\big|_B \tag{118}$$

Now, with reference to the gravitational part of the generalized force, if the base platform orientation changes, then the force applied in {P} and expressed in {B} results in

$${}^{P}\mathbf{f}_{S_i(gra)}\big|_B = {}^{W}\Re_B^{T} \cdot {}^{P}\mathbf{f}_{S_i(gra)}\big|_W \tag{119}$$

where ${}^{P}\mathbf{f}_{S_i(gra)}|_W$ is the gravitational generalized force applied in {P} and expressed in {W}. This force can be computed using the model that considers both a non-moving base platform, frame {B} parallel to {W}, and $\mathbf{z}_B \equiv -\hat{\mathbf{g}}$, i.e.,

$$ {}^{P}\mathbf{f}_{S_i(gra)}|_W = \frac{\partial P_{S_i}\left({}^{B}\mathbf{x}_P|_{B|E}\right)}{\partial\, {}^{B}\mathbf{x}_P|_{B|E}} \tag{120} $$

P_{S_i} being the piston potential energy. Using equation (92) it can be computed by:

$$ P_{S_i} = m_S \cdot g \cdot \left(1 - \frac{b_S}{l_i}\right) \cdot \left(z_P + {}^{P}p_i|_{Bz}\right) \tag{121} $$

The gravitational component of the actuating forces due to each piston, $\boldsymbol{\tau}_{S_i(gra)}$, is

$$ \boldsymbol{\tau}_{S_i(gra)} = \mathbf{J}_C^{-T} \cdot {}^{P}\mathbf{f}_{S_i(gra)}|_B \tag{122} $$

It should be noted that the base platform motion originates new inertial contributions to the parallel manipulator dynamic model, expressed by equations (38), (67), (84), (105) and (118). These contributions should the added to the corresponding ones resulting from the model that considers a fixed-base: equations (37), (66), (83), (104) and (117). Regarding the gravitational part of the dynamic model, the base platform motion modifies the gravitational force components, resulting in the equations (42), (88) and (122), which can also be added to the previous ones.
Computational efficiency of the proposed model has been evaluated by counting the number of scalar operations needed in the calculations (sums, multiplications and divisions). For this purpose, the Maple® software package was used. The results were then compared with the ones obtained by using the Lagrange formulation. The generalized momentum approach resulted in a much more efficient dynamic model. Regarding the total number of sums and multiplications involved in the two models, the ratio is five, approximately. This might be a great advantage if real-time simulation and control is needed.

4. Conclusion

A parallel manipulator is a complex multi-body dynamic system having several closed loops. Typically, it is composed of a (usually) fixed base platform and a moving payload platform, connected by at least two independent open kinematic chains. Dynamic modelling of parallel manipulators presents an inherent difficulty. Despite the intensive study in this topic of robotics, mostly conducted in the last two decades, additional research still has to be done in this area.

In this book chapter an approach based on the manipulator generalized momentum was explored and applied to the dynamic modelling of a Stewart platform manipulator. The system dynamic equations were obtained for the general case of a non-fixed base platform. It was shown the base platform motion originates new inertial force contributions to the parallel manipulator dynamic model. Compact analytical expressions for these contributions were presented and it was shown they can be easily added to the corresponding ones resulting from the model that considers a fixed base. On the other hand, regarding the gravitational part of the dynamic model, the base platform motion modifies the gravitational force components derived when considering a fixed base platform. Analytical expressions for these components were also presented. Computational efficiency of the dynamic model was evaluated by counting the number of scalar operations that are needed. The results were then compared with the ones obtained by using the Lagrange formulation. The generalized momentum approach results in a much more efficient dynamic model.

5. References

Abdellatif, H. & Heimann, B. (2009). Computational efficient inverse dynamics of 6-DOF fully parallel manipulators by using the Lagrangian formalism. *Mechanism and Machine Theory*, Vol. 44, 192-207

Bhattacharya, S., Hatwal, H. & Ghosh, A. (1997). An on-line estimation scheme for generalized Stewart platform type parallel manipulators. *Mech. Mach. Theory*, Vol. 32, 79-89

Caccavale, F., Siciliano, B. & Villani, L. (2003). The tricept robot: Dynamics and impedance control. *IEEE/ASME Transactions on Mechatronics*, Vol. 8, 263-268

Carricato, M. & Gosselin, C. (2009). On the Modeling of Leg Constraints in the Dynamic Analysis of Gough/Stewart-Type Platforms. *J. Comput. Nonlinear Dynamics*, Vol. 4, DOI:10.1115/1.3007974

Dasgupta, B. & Choudhury, P. (1999). A general strategy based on the Newton-Euler approach for the dynamic formulation of parallel manipulators. *Mech. Mach. Theory*, Vol. 34, 801-824

Dasgupta, B. & Mruthyunjaya, T. (1998). A Newton–Euler formulation for the inverse dynamics of the Stewart platform manipulator. *Mechanism and Machine Theory*, Vol. 34, 711-725

Di Gregório, R. & Parenti-Castelli, V. (2004). Dynamics of a class of parallel wrists. *Journal of Mechanical Design*, Vol. 126, 436-441

Do, W. & Yang, D. (1988). Inverse dynamic analysis and simulation of a platform type of robot. *Journal of Robotic Systems*, Vol. 5, 209-227

Fichter, E. (1986). *A Stewart Platform-Based Manipulator: General Theory and Practical Construction*. The Int. Journal of Robotics Research, Vol. 5, 157-182

Gallardo, J., Rico, J., Frisoli, A., Checcacci, D. & Bergamasco, M. (2003). Dynamics of parallel manipulators by means of screw theory. *Mechanism and Machine Theory*, Vol. 38, 1113-1131

Guo, H. & Li, H. (2006). Dynamic analysis and simulation of a six degree of freedom Stewart platform manipulator. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, Vol. 220, 61-72

Ji, Z. (1994). Dynamics decomposition for Stewart platforms. *ASME Journal of Mechanical Design*, Vol. 116, 67-69

Khalil, W. & Ibrahim, O. (2007). General solution for the dynamic modelling of parallel robots. *Journal of Intelligent and Robot Systems*, Vol. 49, 19-37

Lebret, G., Liu, K. & Lewis, F. (1993). Dynamic analysis and control of a Stewart platform manipulator. *Journal of Robotic Systems*, Vol. 10, 629-655

Liu, M-J., Li, C-X. & Li, C-N. (2000). Dynamics analysis of the Gough-Stewart platform manipulator. *IEEE Trans. Robotics and Automation*, Vol. 16, 94-98

Lopes, A. M. (2009). Dynamic modeling of a Stewart platform using the generalized momentum approach. *Communications in Nonlinear Science and Numerical Simulation*, Vol. 14, 3389–3401

Merlet, J-P. (2006). *Parallel Robots*, Springer, Dordrecht, Netherlands

Miller, K. (2004). Optimal design and modeling of spatial parallel manipulators. *Int. J. Robotics Research*, Vol. 23, 127-140

Nguyen, C. & Pooran, F. (1989). Dynamic analysis of a 6 DOF CKCM robot end-effector for dual-arm telerobot systems. *Robotics and Autonomous Systems*, Vol. 5, 377-394

Raghavan, M. (1993). The Stewart Platform of General Geometry has 40 Configurations. *ASME Journal of Mechanical Design*, Vol. 115, 277-282

Reboulet, C. & Berthomieu, T. (1991). Dynamic Models of a Six Degree of Freedom Parallel Manipulators, *Proceedings of the IEEE Int. Conf. on Robotics and Automation*, pp. 1153-1157

Riebe, S. & Ulbrich, H. (2003). Modelling and online computation of the dynamics of a parallel kinematic with six degrees-of-freedom. *Archive of Applied Mechanics*, Vol. 72, 817-829

Sokolov, A. & Xirouchakis, P. (2007). Dynamics analysis of a 3-DOF parallel manipulator with R-P-S joint structure. *Mechanism and Machine Theory*, Vol. 42, 541-557

Staicu, S. & Zhang, D. (2008). A novel dynamic modelling approach for parallel mechanisms analysis. *Robotics and Computer-Integrated Manufacturing*, Vol. 24, 167-172

Staicu, S., Liu, X.-J. & Wang, J. (2007). Inverse dynamics of the HALF parallel manipulator with revolute actuators. *Nonlinear Dynamics*, Vol. 50, 1-12

Tsai, L.-W. (2000). Solving the inverse dynamics of Stewart-Gough manipulator by the principle of virtual work. *Journal of Mechanical Design*, Vol. 122, 3-9

Vukobratovic, M. & Kircanski, M. (1986). *Kinematics and Trajectory Synthesis of Manipulation Robots*, Springer-Verlag, Berlin

Wang, J. & Gosselin, C. (1998). A new approach for the dynamic analysis of parallel manipulators. *Multibody System Dynamics*, Vol. 2, 317-334

Wang, J., Wu, J., Wang, L. & Li, T. (2007). Simplified strategy of the dynamic model of a 6-UPS parallel kinematic machine for real-time control. *Mechanism and Machine Theory*, Vol. 42, 1119-1140

Zhao, Y. & Gao, F. (2009). Inverse dynamics of the 6-dof out-parallel manipulator by means of the principle of virtual work. *Robotica*, Vol. 27, 259–268

19

Improving the Pose Accuracy of Planar Parallel Robots using Mechanisms of Variable Geometry

Jens Kotlarski, Bodo Heimann and Tobias Ortmaier
Institute of Mechatronic Systems, Leibniz Universität Hannover
Germany

1. Introduction

In comparison to classical serial mechanisms parallel kinematic machines (PKM) provide a higher accuracy, a higher stiffness, and higher dynamic properties. However, parallel robots suffer from the presence of singularities within their workspace (Gosselin & Angeles, 1990). In such configurations the moving platform gains at least one degree of freedom (DOF) and the actuation forces become (in theory) infinite. As a result, the kinematic structure can be damaged or even destroyed. Additionally, several performance indices, e.g. the achievable accuracy, are directly related to the singularity loci (Kotlarski, Abdellatif & Heimann, 2008). The closer the endeffector (EE) is 'located' to a singularity the higher is the pose error resulting from the influence of active joint errors, e.g. from limited encoder resolution.
In order to minimize the singularity loci of parallel mechanisms and to increase their performance, e.g. their achievable accuracy, redundancy can be used (Merlet, 1996). Two redundancy approaches are established for PKM, actuation redundancy and kinematic redundancy (Kock & Schumacher, 1998; Wang & Gosselin, 2004). Actuation redundancy can be realized whether by adding a kinematic chain to the mechanism or by actuating a passive joint. Amongst others, it reduces singular configurations and leads to internal preload that can be controlled in order to prevent backlash (Kock, 2001). However, the control of such mechanisms is a challenging task (Müller, 2005). Furthermore, an additional kinematic chain mostly reduces the total workspace. Therefore, kinematic redundancy is proposed realized by adding at least one actuated joint to one kinematic chain (Cha et al., 2007; Mohamed & Gosselin, 2005).
It is well known that the singularity loci as well as the achievable accuracy are greatly affected by the geometrical parameters of a mechanism (Kotlarski, de Nijs, Abdellatif & Heimann, 2009; Merlet & Daney, 2005), and are therefore highly dependent on the mechanism's actual configuration. In this chapter, as examples, kinematically redundant versions of the well known planar $3\underline{\text{R}}\text{RR}$ and $3\text{R}\underline{\text{P}}\text{R}$ mechanisms (Gosselin & Angeles, 1988; Zein et al., 2006) are considered. In each case, an additional prismatic actuator is added to an arbitrary base joint. The introduced mechanisms are denoted as $3(\underline{\text{P}})\underline{\text{R}}\text{RR}$ and $3(\underline{\text{P}})\text{R}\underline{\text{P}}\text{R}$. Thanks to the additional prismatic actuator, the inverse displacement problem has an infinite number of solutions (Ebrahimi et al., 2007). Hence, reconfigurations of the mechanisms can be performed selectively in order to avoid singularities and to affect their performance directly (Kotlarski, Do Thanh, Abdellatif & Heimann, 2008). It is important to note that with respect to the work of Arakelian et al. (Arakelian et al., 2008), kinematic redundancy can be used to rather change the geometrical parameters of a mechanism than its basic structure. This can be done at the

task planning stage or while operating the manipulator using several different strategies (see Sec. 3.2).
Here, the key idea is to change the position of the redundant actuator in a discrete manner while operating the mechanism, in particular just before shifts in direction of the moving platform. This allows for optimizing the accuracy of the manipulator for a given trajectory segment. After each switching operation, i.e. after each reconfiguration, the additional prismatic actuator is supposed to remain locked. Therefore, the joint clearance as well as the control error corresponding to the redundant actuator can be minimized. The resulting set of discrete actuator positions is called the switching pattern. The optimization of the switching patterns is achieved according to a performance index denoted as the gain of the maximal homogenized pose error (see Sec. 3.2).
The rest of this chapter is organized as follows. In Sec. 2 the geometric and the inverse kinematic models of the proposed mechanisms are given as well as fundamental definitions related to their Jacobian analysis. Sec. 3 clarifies the idea of kinematic redundancy in order to avoid singularities and to increase the performance of a PKM. Additionally, it gives a brief theoretical overview on the determination of the achievable moving platform pose accuracy and introduces the optimization strategy developed for the redundant actuator position. In Sec. 4 several analysis examples are presented in order to validate the proposed redundant schemes with the optimized switching patterns. It is demonstrated that the kinematically redundant mechanisms in combination with the developed optimization procedure lead to a great improvement in terms of singularity avoidance and, therefore, in terms of accuracy and precision. Furthermore, it is shown that the proposed optimization criterion is able to outperform the classical accuracy related performance index, i.e. the condition number of the Jacobian matrix. Sec. 5 concludes this chapter.

2. Kinematically redundant mechanisms

In the following, both the geometrical and the inverse kinematic models of the exemplarily analyzed planar, kinematically redundant mechanisms are given along with fundamental definitions related to the Jacobian analysis. An additional prismatic actuator is proposed allowing one base joint to move linearly. Hence, reconfiguration of the mechanisms can be performed selectively while operating the manipulators.

2.1 Redundant 3(P̲)R̲RR mechanism

In (Kotlarski et al., 2007) the kinematically redundant 3(P̲)R̲RR planar mechanism (see Fig. 1) was introduced. It is basically similar to the non-redundant 3R̲RR mechanism studied amongst others in (Gosselin & Angeles, 1988). Three kinematic chains $G_iM_iP_i$ ($i = 1, 2, 3$) connect the moving platform $P_1P_2P_3$ to the base $G_1G_2G_3$. Each kinematic chain consists of two links $l_{i,1}$ and $l_{i,2}$. The position of the joints G_i, M_i, and P_i with respect to the inertial coordinate frame $(CF)_0$ is given by $\boldsymbol{g}_i = (x_{G_i}, y_{G_i})^T$, $\boldsymbol{m}_i = (x_{M_i}, y_{M_i})^T$, and $\boldsymbol{p}_i = (x_{P_i}, y_{P_i})^T$. The base-fixed revolute joints are active while the remaining ones are passive. The orientation of the redundant actuator with respect to the x-axis of $(CF)_0$ is denoted by α. Positions referenced with respect to the platform fixed coordinate frame $(CF)_E$ are marked with ($'$).
In the following the configuration of the moving platform is given by

$$\boldsymbol{x} = (x_E, y_E, \phi)^T, \tag{1}$$

where x_E and y_E represent the point of origin of the platform fixed coordinate frame $(CF)_E$ with respect to $(CF)_0$ and ϕ is its orientation. The mechanism is driven by the four actuators.

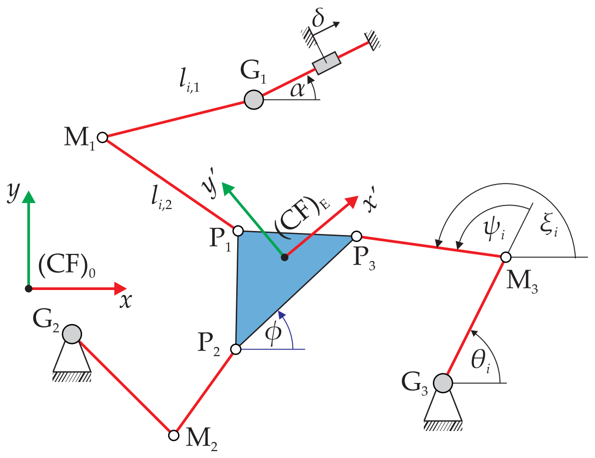


Fig. 1. Kinematically redundant 3(P)RRR mechanism

Therefore, the system input is given by the according actuator coordinates

$$\boldsymbol{\theta} = (\theta_1,\ \theta_2,\ \theta_3,\ \delta)^{\mathrm{T}}. \tag{2}$$

2.1.1 Inverse kinematics

For each kinematic chain i the geometric constraints can be written as

$$\begin{pmatrix} x_{\mathrm{P}_i} \\ y_{\mathrm{P}_i} \end{pmatrix} = \begin{pmatrix} x_{\mathrm{G}_i} \\ y_{\mathrm{G}_i} \end{pmatrix} + \begin{pmatrix} \cos(\theta_i) & \cos(\theta_i+\psi_i) \\ \sin(\theta_i) & \sin(\theta_i+\psi_i) \end{pmatrix} \begin{pmatrix} l_{i,1} \\ l_{i,2} \end{pmatrix}, \tag{3}$$

where the position of the moving platform's passive joints with respect to $(\mathrm{CF})_0$ is defined as

$$\begin{pmatrix} x_{\mathrm{P}_i} \\ y_{\mathrm{P}_i} \end{pmatrix} = \begin{pmatrix} x_{\mathrm{E}} \\ y_{\mathrm{E}} \end{pmatrix} + \begin{pmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{pmatrix} \begin{pmatrix} x'_{\mathrm{P}_i} \\ y'_{\mathrm{P}_i} \end{pmatrix}. \tag{4}$$

In the presented redundant case the position of G_1 depends on the actuator position δ:

$$\begin{pmatrix} x_{\mathrm{G}_1} \\ y_{\mathrm{G}_1} \end{pmatrix} = \begin{pmatrix} x_{\mathrm{G}_1} \\ y_{\mathrm{G}_1} \end{pmatrix}\Bigg|_{\delta=0\,\mathrm{m}} + \begin{pmatrix} \delta\ \cos(\alpha) \\ \delta\ \sin(\alpha) \end{pmatrix}. \tag{5}$$

From (3) the passive joint angles ψ_i can be determined:

$$\psi_i = \pm \arccos\left(\frac{x^2_{\mathrm{GP}_i} + y^2_{\mathrm{GP}_i} - l^2_{i,1} - l^2_{i,2}}{2\, l_{i,1}\, l_{i,2}} \right), \tag{6}$$

where $x_{\mathrm{GP}_i} = x_{\mathrm{P}_i} - x_{\mathrm{G}_i}$ and $y_{\mathrm{GP}_i} = y_{\mathrm{P}_i} - y_{\mathrm{G}_i}$. The active joint angles θ_i are finally obtained using (3) and (6):

$$\theta_i = \arctan\left(\frac{(l_{i,1} + l_{i,2}\ \cos(\psi_i))\ y_{\mathrm{GP}_i} - (l_{i,2}\ \sin(\psi_i))\ x_{\mathrm{GP}_i}}{(l_{i,1} + l_{i,2}\ \cos(\psi_i))\ x_{\mathrm{GP}_i} + (l_{i,2}\ \sin(\psi_i))\ y_{\mathrm{GP}_i}} \right), \tag{7}$$

Additionally, for simplification reasons, the angles ξ_i are defined as the counterclockwise angles that the passive links form with the x-axis:

$$\xi_i = \arctan\left(\frac{y_{\mathrm{GP}_i}\,\left(l_{i,2}+l_{i,1}\,\cos(\psi_i)\right)+\left(l_{i,1}\,\sin(\psi_i)\right)\,x_{\mathrm{GP}_i}}{x_{\mathrm{GP}_i}\,\left(l_{i,2}+l_{i,1}\,\cos(\psi_i)\right)-\left(l_{i,1}\,\sin(\psi_i)\right)\,y_{\mathrm{GP}_i}}\right). \tag{8}$$

2.1.2 Jacobian formulation

Several performance criteria and indices, e.g. the achievable accuracy, can be calculated based on the Jacobian matrices of a PKM (see Sec. 4). After summing the squares of (3) the Jacobians can be obtained by a derivation of the resulting inverse kinematic equations f_i

$$f_i \equiv 0 = \left(x_{\mathrm{P}_i} - x_{\mathrm{G}_i} - l_{i,1}\,\cos(\theta_i)\right)^2 + \left(y_{\mathrm{P}_i} - y_{\mathrm{G}_i} - l_{i,1}\,\sin(\theta_i)\right)^2 - l_{i,2}^2, \tag{9}$$

with respect to time (Gosselin & Angeles, 1990):

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\,\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\theta}}\,\dot{\boldsymbol{\theta}} = \boldsymbol{0} \;\Leftrightarrow\; \boldsymbol{A}\,\dot{\boldsymbol{x}} + \boldsymbol{B}\,\dot{\boldsymbol{\theta}} = \boldsymbol{0}. \tag{10}$$

For the 3(P̲)R̲RR mechanism the direct and the inverse Jacobian matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ result to

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \qquad \boldsymbol{B} = \begin{pmatrix} b_{11} & 0 & 0 & b_{14} \\ 0 & b_{22} & 0 & 0 \\ 0 & 0 & b_{33} & 0 \end{pmatrix}, \tag{11}$$

with (for $i = 1,2,3$)

$$\begin{aligned} a_{i1} &= x_{\mathrm{P}_i} - x_{\mathrm{G}_i} - l_{i,1}\,\cos\theta_i, \\ a_{i2} &= y_{\mathrm{P}_i} - y_{\mathrm{G}_i} - l_{i,1}\,\sin\theta_i, \\ a_{i3} &= -\,a_{i1}\,\left(x'_{\mathrm{P}_i}\,\sin(\phi) + y'_{\mathrm{P}_i}\,\cos(\phi)\right) + a_{i2}\,\left(x'_{\mathrm{P}_i}\,\cos(\phi) - y'_{\mathrm{P}_i}\,\sin(\phi)\right), \end{aligned} \tag{12}$$

and

$$\begin{aligned} b_{ii} &= l_{i,1}\,\left(a_{i1}\,\sin(\theta_i) - a_{i2}\,\cos(\theta_i)\right), \\ b_{14} &= -\,\left(a_{11}\,\cos(\alpha) + a_{12}\,\sin(\alpha)\right). \end{aligned} \tag{13}$$

As long as the Jacobian $\boldsymbol{A}$ is nonsingular, its inverse $\boldsymbol{A}^{-1}$ can be determined analytically:

$$\boldsymbol{A}^{-1} = \frac{1}{\det(\boldsymbol{A})}\begin{pmatrix} a_{33}a_{22} - a_{32}a_{23} & -(a_{33}a_{12} - a_{32}a_{13}) & a_{23}a_{12} - a_{22}a_{13} \\ -(a_{33}a_{21} - a_{31}a_{23}) & a_{33}a_{11} - a_{31}a_{13} & -(a_{23}a_{11} - a_{21}a_{13}) \\ a_{32}a_{21} - a_{31}a_{22} & -(a_{32}a_{11} - a_{31}a_{12}) & a_{22}a_{11} - a_{21}a_{12} \end{pmatrix}. \tag{14}$$

It will be useful for calculating the achievable accuracy as demonstrated in Sec. 3.1.

2.2 Redundant 3(P̲)RP̲R mechanism

Additionally, the kinematically redundant 3(P̲)RP̲R planar mechanism presented in Fig. 2 is considered. It was firstly introduced in (Kotlarski, Abdellatif, Ortmaier & Heimann, 2009). It is based on the well known 3RP̲R mechanism (Zein et al., 2006). Three kinematic chains $\mathrm{G_iP_i}$ ($i = 1,2,3$) consisting of active prismatic joints connect the moving platform $\mathrm{P_1P_2P_3}$ to the base $\mathrm{G_1G_2G_3}$. In the following, notations and definitions similar to the 3(P̲)R̲RR mechanism

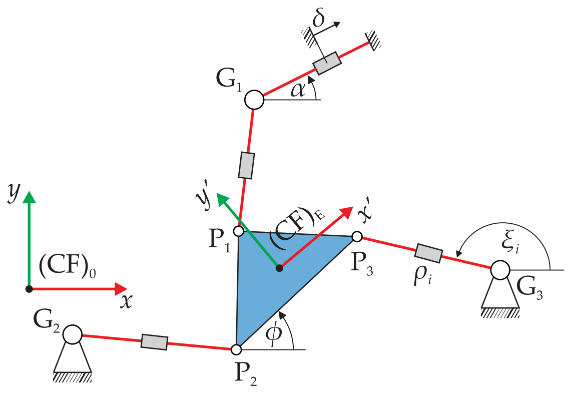


Fig. 2. Kinematically redundant 3(P)RPR mechanism

and already introduced are not mentioned again.
The system input is given by the four actuator coordinates

$$\boldsymbol{\theta} = (\rho_1, \rho_2, \rho_3, \delta)^{\mathrm{T}}, \tag{15}$$

where ρ_i defines the lengths of the kinematic chain i.

2.2.1 Inverse kinematics

The geometric constraints of the 3(P)RPR mechanism can be written as

$$\begin{pmatrix} x_{\mathrm{P}_i} \\ y_{\mathrm{P}_i} \end{pmatrix} = \begin{pmatrix} x_{\mathrm{G}_i} \\ y_{\mathrm{G}_i} \end{pmatrix} + \rho_i \begin{pmatrix} \cos(\xi_i) \\ \sin(\xi_i) \end{pmatrix}, \tag{16}$$

where the passive joint angles ξ_i are obtained by

$$\xi_i = \arctan\left(\frac{y_{\mathrm{GP}_i}}{x_{\mathrm{GP}_i}}\right). \tag{17}$$

From the Euclidean norm of the vector connecting point G_i to point P_i the lengths of each kinematic chain i are obtained

$$\rho_i^2 = x_{\mathrm{GP}_i}^2 + y_{\mathrm{GP}_i}^2. \tag{18}$$

2.2.2 Jacobian formulation

Similar to (11) and using f_i (cp. (18))

$$f_i \equiv 0 = x_{\mathrm{GP}_i}^2 + y_{\mathrm{GP}_i}^2 - \rho_i^2, \tag{19}$$

for the 3(P)RPR mechanism the elements of the direct and inverse Jacobian matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ result to

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \qquad \boldsymbol{B} = \begin{pmatrix} b_{11} & 0 & 0 & b_{14} \\ 0 & b_{22} & 0 & 0 \\ 0 & 0 & b_{33} & 0 \end{pmatrix}, \tag{20}$$

with (for $i = 1, 2, 3$)

$$\begin{aligned} a_{i1} &= x_{\mathrm{GP}_i}, \\ a_{i2} &= y_{\mathrm{GP}_i}, \\ a_{i3} &= -a_{i1}\left(x'_{\mathrm{P}_i}\sin(\phi) + y'_{\mathrm{P}_i}\cos(\phi)\right) + a_{i2}\left(x'_{\mathrm{P}_i}\cos(\phi) - y'_{\mathrm{P}_i}\sin(\phi)\right), \end{aligned} \tag{21}$$

and

$$\begin{aligned} b_{ii} &= -\rho_i, \\ b_{14} &= -\left(a_{11}\cos(\alpha) + a_{12}\sin(\alpha)\right). \end{aligned} \tag{22}$$

3. Singularity avoidance and accuracy improvement using kinematic redundancy

The condition for type-two singularities of planar mechanisms can be formulated geometrically (Hunt, 1978; Yang et al., 2002). For the here treated and common case of revolute passive joints at the moving platform, a pose of the robot is singular if the three lines passing through the passive links of the kinematic chains (passive lines) intersect at a common point or are all parallel. Thanks to the kinematic redundancy, the direction of the passive lines and, therefore, the Jacobians' elements can be directly affected. As a result, the singularity loci change as shown in Fig. 3, exemplarily for the introduced kinematically redundant versions of the planar 3$\underline{\text{R}}$RR (top) and 3R$\underline{\text{P}}$R (bottom) mechanisms with a base joint mounted on an additional prismatic actuator.

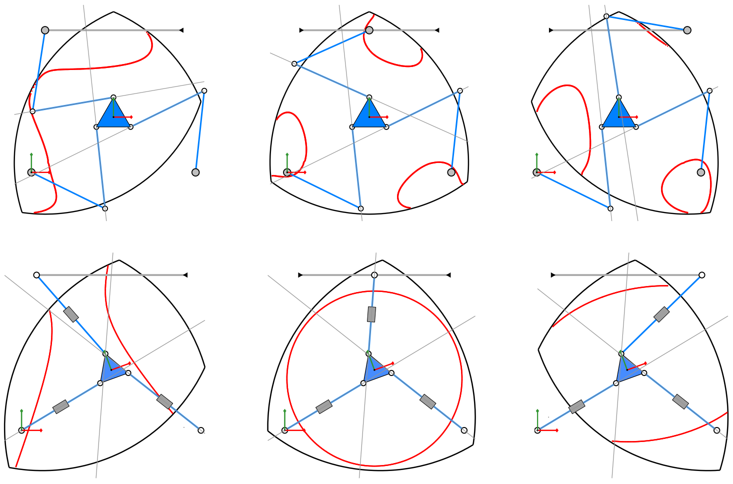


Fig. 3. Variation of the singularity loci (solid red) due to the redundant actuator configuration, i.e. the base joint position; top: redundant 3($\underline{\text{P}}$)$\underline{\text{R}}$RR, bottom: redundant 3($\underline{\text{P}}$)R$\underline{\text{P}}$R

The use of kinematic redundancy to avoid singularities is demonstrated in Fig. 4, left. The given path would cross a singularity for the symmetric, i.e. the 'classical', configuration. By moving the redundant actuator towards the right, the singularity loci could be completely removed. In case of the mechanism performance, e.g. in case of the achievable accuracy, similar effects are given. Regions suffering from high pose errors, i.e. workspace regions that do not provide a certain desired accuracy, can be 'moved' and, therefore, avoided when following a desired trajectory as shown in Fig. 4, right. Hence, the achievable accuracy increases

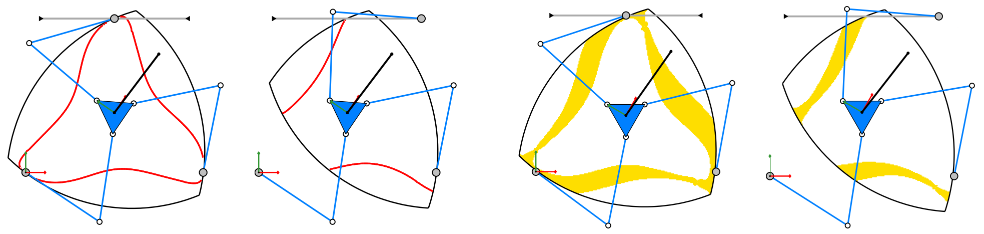


Fig. 4. Trajectory (solid black) going through a singular configuration (left, solid red) / a region where a certain accuracy in one or more directions is not given (right, yellow); a reconfiguration (from the left to the right) allows to follow the desired path

significantly as demonstrated in Sec. 4.

3.1 The moving platform's pose error

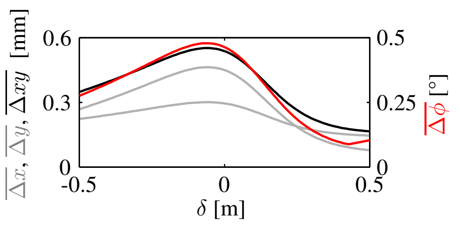
Due to several factors, like manufacturing errors, joint clearance, and active joint errors, the pose of the moving platform can be provided only within a given accuracy. An approach to determine the achievable accuracy of a PKM while considering any kind of uncertainties can be found in (Kotlarski, Abdellatif, Ortmaier & Heimann, 2009). Referring to (Merlet, 2006a), the active joint errors, e.g. the limited resolution of the encoders, are the major sources of error in a calibrated and precisely manufactured PKM. Therefore, the analysis is focussed on the achievable accuracy of a moving platform in the presence of active joint errors only.
By rewriting the velocity equation (10) in incremental form, an approximation that relates the active joint errors (the input error) $\Delta\boldsymbol{\theta}$ to the pose error (the output error) $\Delta\boldsymbol{x}$ is obtained:

$$\boldsymbol{A}\,\Delta\boldsymbol{x} + \boldsymbol{B}\,\Delta\boldsymbol{\theta} \;\Leftrightarrow\; \boldsymbol{A}\,\Delta\boldsymbol{x} = -\boldsymbol{B}\,\Delta\boldsymbol{\theta} \quad\Rightarrow\quad \Delta\boldsymbol{x} = \boldsymbol{A}^{-1}\,(-\boldsymbol{B})\Delta\boldsymbol{\theta} = -\boldsymbol{J}\,\Delta\boldsymbol{\theta}. \tag{23}$$

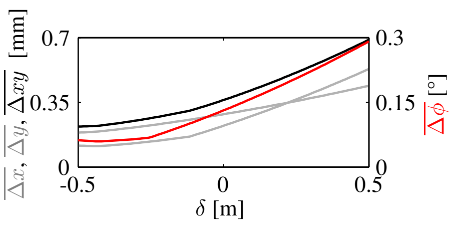
Using (23) and incorporating the fact that $|ab| = |a||b|$ and $|a+b| \leq |a| + |b| \;\forall\; (a,b) \in R$ the maximal pose error vector $\overline{\Delta\boldsymbol{x}}$ can be calculated by

$$\overline{\Delta\boldsymbol{x}} = \begin{pmatrix} \overline{\Delta x} \\ \overline{\Delta y} \\ \overline{\Delta\phi} \end{pmatrix} = \begin{pmatrix} |J_{11}| & |J_{12}| & |J_{13}| & |J_{14}| \\ |J_{21}| & |J_{22}| & |J_{23}| & |J_{24}| \\ |J_{31}| & |J_{32}| & |J_{33}| & |J_{34}| \end{pmatrix} \begin{pmatrix} |\Delta\theta_1| \\ |\Delta\theta_2| \\ |\Delta\theta_3| \\ |\Delta\delta| \end{pmatrix} \geq |\Delta\boldsymbol{x}|. \tag{24}$$

The Jacobian element of row i and column j is denoted as J_{ij}. Since the Jacobian matrix $\boldsymbol{J}$ highly depends on the actuator position δ, i.e. on the robot geometry, the additional DOF of the proposed kinematically redundant mechanisms can be used to affect the Jacobian elements and, therefore, the robot accuracy directly. Fig. 5 shows the elements of $\overline{\Delta\boldsymbol{x}}$ with respect to the actuator position δ for the chosen kinematically redundant mechanisms (left: 3(P)RRR, right 3(P)RPR) and an arbitrary constant EE pose $\boldsymbol{x}$. The elements of the active joint error vector $\Delta\boldsymbol{\theta}$ are taken from data sheets of available actuators. The authors forbear from giving the concrete parameters, i.e. the robot geometry, the EE pose, and the active joint errors, because the characteristics of $\overline{\Delta\boldsymbol{x}}$ with respect to δ is similar in all cases. The dependency of



(a) 3(P)RRR mechanism



(b) 3(P)RPR mechanism

Fig. 5. Positioning error $\overline{\Delta x}$ and $\overline{\Delta y}$ (solid gray), the overall translational error $\overline{\Delta xy}$ (solid black) and the orientational error $\overline{\Delta\phi}$ (solid red) with respect to the redundant actuator position δ and for a constant EE pose $\boldsymbol{x}$

the achievable accuracy on the redundant actuator position is well noticeable. Additionally, it can be seen that the elements of the maximal pose error $\overline{\Delta x}$, $\overline{\Delta y}$, and $\overline{\Delta\phi}$ as well as the overall translational error $\overline{\Delta xy} = \| \left(\overline{\Delta x},\ \overline{\Delta y}\right) \|_2$ have similar minima. Hence, in most cases, the maximal accuracy of each DOF can be increased for almost identical actuator positions δ by an appropriate reconfiguration of the mechanism. Therefore, an optimization procedure is required in order to find the best solution for δ.

3.2 Optimization of the redundant actuator position

The optimization of the redundant actuator position δ can be performed based on two main strategies: a classical continuous optimization and a selective discrete optimization. The latter is the key idea of this chapter and is discussed in the following.

Undoubtedly, a continuous optimization leads to an instantly influenceable, i.e. maximal achievable, accuracy. In contrast to the mentioned advantage, it results in a more challenging task concerning the robot control and usually in a higher energy demand. The proposed approach is based on the optimization of δ in a discrete manner while operating the system. Therefore, the trajectory is divided into segments. The starting and final points of the segments are certain poses, e.g. shifts in direction. Appropriate constant values of the actuator position δ corresponding to the different segments of the desired trajectory are determined. The resulting set of discrete actuator positions is called the optimized switching pattern. While moving along the desired trajectory, the position of the redundant actuator is changed according to the switching pattern. This allows for the reconfiguration of the mechanism to influence its accuracy for a given path segment. While performing a reconfiguration the pose of the moving platform is kept constant. After each switching operation, e.g. while moving along a trajectory segment, the additional prismatic actuator is supposed to remain locked. Therefore, compliance, e.g. resulting from joint clearance, as well as the control error corresponding to the redundant actuator are minimized.

In order to further minimize the switching operations the mentioned discrete optimization, i.e. the 'main idea', can be additionally modified in several ways. One possibility is to only change the redundant actuator position once before starting the desired movement. As a result, the number of reconfigurations is minimized. But, regarding complex trajectories, i.e. trajectories going through a large area of the robot workspace, this may lead to an unacceptable performance, e.g. with respect to the accuracy. Thus, another possible modification is to perform a reconfiguration only if the mechanism is unable to perform the desired operation, e.g. following a singularity-free trajectory and providing a certain accuracy, in its current configuration (Kotlarski, Do Thanh, Abdellatif & Heimann, 2008). Therefore, before moving the EE, for the upcoming trajectory segment the required performance criteria have to be calculated. If any criteria is less than its corresponding threshold a reconfiguration of the mechanism has to be performed. The mentioned modification of the proposed selective discrete optimization strategy further reduces the inconvenient switching operations and guarantees a desired performance. Nevertheless, in this chapter it is focussed on the main idea of the discrete reconfiguration strategy without any of the modifications mentioned in this paragraph. It mostly clarifies the influence of the additional prismatic actuator which is the authors' main purpose.

The optimization can be realized with respect to several criteria and performance indices: a well accepted criterion is the condition number (in general the two-norm condition number) of the Jacobian matrix $\kappa(\boldsymbol{J})$ and its inverse $\eta = \kappa^{-1}$ called dexterity. In (Gosselin, 1992) it is defined as:

$$\kappa(\boldsymbol{J}) = \kappa(\boldsymbol{J}^{-1}) = \|\boldsymbol{J}^{-1}\|_2 \|\boldsymbol{J}\|_2, \qquad 1 \leq \kappa \leq \infty, \tag{25}$$

where $\kappa = 1$ represents an isotropic configuration without an amplification of the active joint error $\Delta\boldsymbol{\theta}$ and $\kappa = \infty$ represents a singular configuration with an infinite amplification of $\Delta\boldsymbol{\theta}$ leading to (in theory) an infinite $\Delta\boldsymbol{x}$. However, the moving platforms of the considered mechanisms have two translational as well as one rotational DOF. As a result, the Jacobian matrix $\boldsymbol{J}$ is not homogeneous in terms of physical units. Therefore, the value of the condition number depends on the unit choice. Hence, a modification of the Jacobian matrix is required in order to obtain appropriate values for κ. Amongst others, the homogeneity can be achieved by transforming the moving platform velocity $\dot{\boldsymbol{x}}$ into the linear velocity $\dot{\boldsymbol{x}}_\mathrm{h} = (\dot{x}_{\mathrm{P}_1}, \dot{y}_{\mathrm{P}_1}, \dot{x}_{\mathrm{P}_2}, \dot{y}_{\mathrm{P}_2})^\mathrm{T}$ of two arbitrary points P_1 and P_2 (Pond & Carretero, 2006). Therefore, a transformation matrix $\boldsymbol{Q}$ has to be found that satisfies the following equation:

$$\dot{\boldsymbol{x}}_\mathrm{h} = \boldsymbol{Q}\dot{\boldsymbol{x}}, \tag{26}$$

where the subscript 'h' indicates homogeneous. But, instead of describing a manipulator with three DOF by the four parameters $\dot{\boldsymbol{x}}_\mathrm{h}$, a reduction of the terms describing the velocities of the moving platform to three can be performed (Gosselin, 1992). As a result, the dimension of the Jacobian matrix $\boldsymbol{J}$ remains constant. Therefore, a coordinate frame $(\mathrm{CF})_{\mathrm{E_h}}$, located at P_1, is attached to the moving platform such that its x-axis passes through P_1 and P_2. For the proposed mechanisms, by choosing $\dot{\boldsymbol{x}}_\mathrm{h} = (\dot{x}_\mathrm{E}, \dot{y}_\mathrm{E}, \dot{y}_{\mathrm{P}_3})^\mathrm{T}$ (cp. Fig. 1 and Fig. 2), the modified transformation matrix $\boldsymbol{Q}$ results to:

$$\boldsymbol{Q} = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ -\sin\beta & \cos\beta & \|\boldsymbol{p}_3\|_2 \end{bmatrix}, \tag{27}$$

where the angle β gives the orientation of $(\mathrm{CF})_0$ to $(\mathrm{CF})_{\mathrm{E_h}}$. It is important to note that the points P_1 and P_2 can be chosen arbitrary as long as they fulfill the mentioned characteristics (Gosselin, 1992). The homogenized Jacobian matrix $\boldsymbol{J}_\mathrm{h}$ can finally be determined using (23) and (26):

$$\boldsymbol{J}_\mathrm{h} = \boldsymbol{Q}\boldsymbol{J}. \tag{28}$$

Hence, an optimization of the actuator position δ can be performed by a minimization of the condition number $\kappa(\boldsymbol{J}_\mathrm{h})$ and by a maximization of the dexterity $\eta(\boldsymbol{J}_\mathrm{h})$, respectively:

$$\delta_\mathrm{opt} = \arg\left(\min_{\delta} \kappa(\boldsymbol{J}_\mathrm{h})\right) \quad \hat{=} \quad \delta_\mathrm{opt} = \arg\left(\max_{\delta} \eta(\boldsymbol{J}_\mathrm{h})\right). \tag{29}$$

As demonstrated by Merlet (Merlet, 2006b) and shown later in Sec. 4 the condition number does not necessarily exhibit a complete consistent behavior with respect to the pose error of a robot. Therefore, an optimization of the actuator position δ based on minimizing the two-norm of the maximal homogenized pose error $\overline{\Delta\boldsymbol{x}_\mathrm{h}}$ is proposed:

$$\gamma(\overline{\Delta\boldsymbol{x}_\mathrm{h}}) = ||\overline{\Delta\boldsymbol{x}_\mathrm{h}}||_2 = \left|\left| \begin{pmatrix} |J_{\mathrm{h}_{11}}| & |J_{\mathrm{h}_{12}}| & |J_{\mathrm{h}_{13}}| & |J_{\mathrm{h}_{14}}| \\ |J_{\mathrm{h}_{21}}| & |J_{\mathrm{h}_{22}}| & |J_{\mathrm{h}_{23}}| & |J_{\mathrm{h}_{24}}| \\ |J_{\mathrm{h}_{31}}| & |J_{\mathrm{h}_{32}}| & |J_{\mathrm{h}_{33}}| & |J_{\mathrm{h}_{34}}| \end{pmatrix} \begin{pmatrix} |\Delta\theta_1| \\ |\Delta\theta_2| \\ |\Delta\theta_3| \\ |\Delta\delta| \end{pmatrix} \right|\right|_2, \tag{30}$$

where the elements of the active joint error vector $\Delta\boldsymbol{\theta}$, i.e. their limited resolutions, are well known from the data sheets of the actuators. This index is called the gain $\gamma(\overline{\Delta\boldsymbol{x}_\mathrm{h}})$ of the maximal homogenized pose error $\overline{\Delta\boldsymbol{x}_\mathrm{h}}$. Although the influence of the prismatic actuator joint error $\Delta\delta$ on the pose error $\Delta\boldsymbol{x}$ is small only (see Sec. 4) it should not be neglected. The cost function to be minimized results to:

$$\delta_\mathrm{opt} = \arg\left(\min_{\delta} \gamma(\overline{\Delta\boldsymbol{x}_\mathrm{h}})\right). \tag{31}$$

There might be trajectories, e.g. regarding special applications, for which a high accuracy is required in certain DOF only. In this case, the optimization criterion (31) can be adopted such that the corresponding elements (or a single element) of $\overline{\Delta\boldsymbol{x}_\mathrm{h}}$ are solely minimized.

4. Accuracy analysis - numerical results

Several examples are presented in order to validate the proposed redundant scheme with the developed optimized switching patterns. The advantage of the approach is verified for different trajectories. Additionally, the influence of the redundant prismatic actuator on the moving platform pose accuracy is demonstrated. Moreover, in order to further confirm the results given in Sec. 4.1, the useable workspace, i.e. the singularity-free part of the workspace providing a certain performance, of the considered mechanisms is determined (and compared).

4.1 Simulation of single trajectories

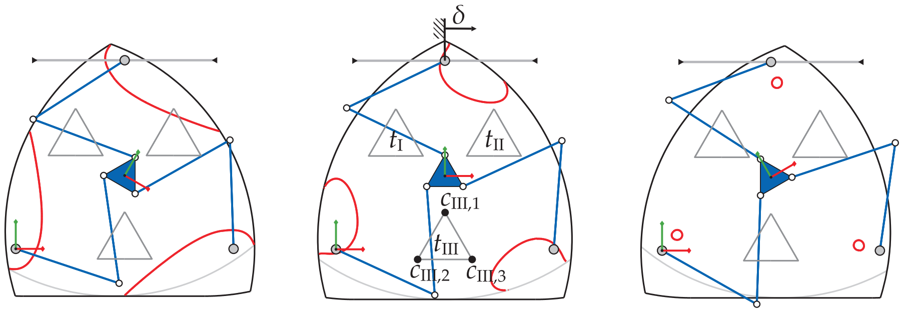
Accuracy analysis along selected simulated trajectories were performed. The geometrical parameters of the analyzed kinematically redundant mechanisms and their non-redundant counterparts are given in Table 1. In the redundant case, one prismatic actuator is attached to G_1 of the basic structure. Keeping the design space in mind the orientation of the redundant actuator was set to $\alpha = 0°$. At this point, it is important to note that the design of the additional prismatic actuator, i.e. its stroke as well as its orientation, was more or less chosen intuitively. Future work will deal with an optimization of the parameters related to the redundant actuator.

	$i=1$	$i=2$	$i=3$	
x_{G_i} [m]	0.6	0	1.2	3(P)RRR & 3(P)RPR
y_{G_i} [m]	$\sqrt{27}/5$	0	0	
x'_{P_i} [m]	0	-0.125	0.125	
y'_{P_i} [m]	0	$-\sqrt{3}/8$	$-\sqrt{3}/8$	
$l_{i,1}$ [m]	0.6	0.6	0.6	3(P)RRR
$l_{i,2}$ [m]	0.6	0.6	0.6	
$\rho_{i,\min}$ [m]	0.1	0.1	0.1	3(P)RPR
$\rho_{i,\max}$ [m]	1.2	1.2	1.2	

Table 1. Design parameters of the analyzed 3(P)RRR and 3(P)RPR mechanisms ($-0.5\,\text{m} \leq \delta \leq 0.5\,\text{m}$)

4.1.1 Redundant 3(P)RRR mechanism

First, the accuracy analysis of the kinematically redundant 3(P)RRR mechanism is performed. Exemplarily, simulation results of the three triangular trajectories (t_{I}, t_{II}, t_{III}) shown in Fig. 6 are presented. In order to clarify the effectiveness of the proposed concept the trajectories were chosen within the workspace of the mechanisms (solid black) such that the non-redundant mechanism ($\delta = 0\,\text{m}$) does not pass any singular configurations when $\phi = 0°$. Without loss of generality, the regarded 3RRR-based mechanisms are in the following assumed to remain in the same working mode which is shown in Fig. 1.



(a) 3(P)RRR ($\phi = -30°$) (b) 3(P)RRR ($\phi = 0°$) (c) 3(P)RRR ($\phi = 30°$)

Fig. 6. Exemplarily chosen trajectories t_{I}, t_{II}, t_{III} (solid gray) for the 3RRR-based mechanisms, the solid red lines represent the singularity loci within the workspace (solid black)

The EE was moved counterclockwise along the depicted trajectories with a constant orientation. The trajectories were divided such that each side of a triangular represents a segment. Hence, at every corner $c_{i,1}$, $c_{i,2}$, and $c_{i,3}$ ($i = \text{I}, \text{II}, \text{III}$) the position of the redundant actuator δ is switched according to the optimized switching pattern. During each switching operation the moving platform pose is kept constant. The optimization was performed based on the introduced cost functions (29) and (31). Even though the prismatic joint is locked between two switching phases, its joint error, e.g. the limited resolution of the encoder, has to be taken into account in order to obtain a realistic and practical accuracy analysis. Therefore, the active

joint errors were chosen based on data sheets of commercially available standard actuators to $\Delta\boldsymbol{\theta}(3(\underline{\mathrm{P}})\underline{\mathrm{R}}\mathrm{RR}) = (0.025°,\ 0.025°,\ 0.025°,\ 40\,\mu\mathrm{m})^{\mathrm{T}}$. It is important to note that in the non-redundant case the last element of $\Delta\boldsymbol{\theta}$ vanishes.
In Fig. 7 the optimized switching patterns δ_{opt} of the actuator position δ as well as the resulting mechanism pose errors $\overline{\Delta xy}$ and $\overline{\Delta\phi}$ are presented. The EE was moved along trajectory t_{I} with a constant orientation of $\phi = -30°$, $\phi = 0°$, and $\phi = 30°$ denoted as $t_{\mathrm{I}}(-30°)$, $t_{\mathrm{I}}(0°)$, and $t_{\mathrm{I}}(30°)$, respectively. The distance the EE moved along the trajectory is denoted as s. A significant improvement of the accuracy due to the kinematic redundancy

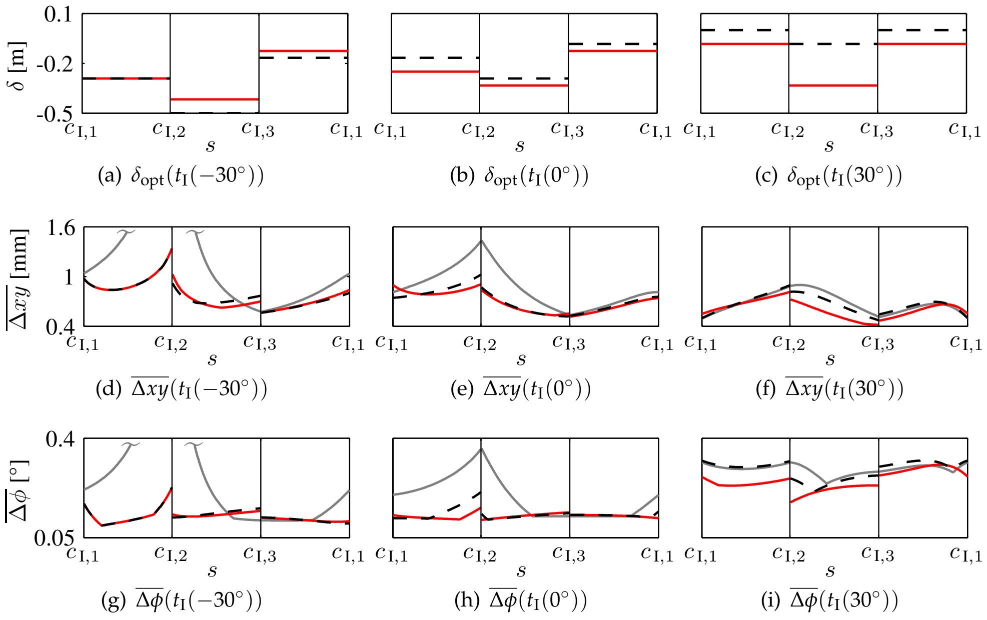


Fig. 7. Simulation results while moving along trajectory $t_{\mathrm{I}}(-30°)$ (left), $t_{\mathrm{I}}(0°)$ (center), and $t_{\mathrm{I}}(30°)$ (right); solid gray: non-redundant mechanism; dashed black: optimized redundant mechanism using $\eta(\boldsymbol{J}_{\mathrm{h}})$; solid red: optimized redundant mechanism using $\gamma(\overline{\Delta\boldsymbol{x}_{\mathrm{h}}})$

is well noticeable. E.g. regarding $t_{\mathrm{I}}(-30°)$ and $t_{\mathrm{I}}(0°)$, the maximal pose error occurring close to $c_{\mathrm{I},2}$ is minimized by a reconfiguration of the mechanism according to the optimized switching patterns. Fig. 7 shows that both optimization criteria ($\eta(\boldsymbol{J}_{\mathrm{h}})$ and $\gamma(\overline{\Delta\boldsymbol{x}_{\mathrm{h}}})$) lead to similar switching patterns and to similar achievable accuracies. In Table 2 an overview of the maximal errors of the three triangular trajectories shown in Fig. 6 are given. In order to quantify the accuracy improvement the maximal translational $\overline{\Delta xy}_{\mathrm{max}}$ and rotational error $\overline{\Delta\phi}_{\mathrm{max}}$ of the moving platform over a complete trajectory was determined. The values represent the achievable accuracy of the associated mechanism. Additionally, the percentage increase/decrease of the kinematically redundant PKM in comparison to its non-redundant counterpart is given. Significant improvements of the achievable accuracy are well noticeable in most cases. Furthermore, e.g. for $t_{\mathrm{III}}(30°)$, it can be seen that an optimization based on the gain $\gamma(\overline{\Delta\boldsymbol{x}_{\mathrm{h}}})$ leads

$t_i(\phi)$	Value	3$\underline{\text{R}}$RR	3($\underline{\text{P}}$)$\underline{\text{R}}$RR	
			using $\eta(\boldsymbol{J}_\text{h})$	using $\gamma(\overline{\Delta \boldsymbol{x}_\text{h}})$
$t_\text{I}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	7.13	1.34 (-88.3%)	1.34 (-88.3%)
	$\overline{\Delta\phi}_\text{max}$ [°]	1.93	0.23 (-81.2%)	0.23 (-81.2%)
$t_\text{I}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	1.44	1.02 (-28.8%)	0.91 (-36.9%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.36	0.21 (-42.3%)	0.16 (-57.2%)
$t_\text{I}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.90	0.90 (-0.5%)	0.81 (-9.5%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.32	0.32 (+2.2%)	0.31 (-2.6%)
$t_\text{II}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	∞	0.69 (-)	0.58 (-)
	$\overline{\Delta\phi}_\text{max}$ [°]	∞	0.14 (-)	0.12 (-)
$t_\text{II}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	3.25	0.75 (-77.1%)	0.69 (-78.9%)
	$\overline{\Delta\phi}_\text{max}$ [°]	1.50	0.22 (-85.3%)	0.22 (-85.3%)
$t_\text{II}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.63	0.70 (+10.3%)	0.68 (+6.8%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.37	0.43 (+14.5%)	0.40 (+7.6%)
$t_\text{III}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.57	0.48 (-15.3%)	0.48 (-15.3%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.30	0.26 (-12.0%)	0.26 (-12.0%)
$t_\text{III}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.70	0.86 (+22.8%)	0.60 (-14.9%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.41	0.31 (-25.0%)	0.35 (-13.3%)
$t_\text{III}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	1.40	1.43 (+2.2%)	1.10 (-21.6%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.41	0.44 (+7.0%)	0.35 (-14.9%)

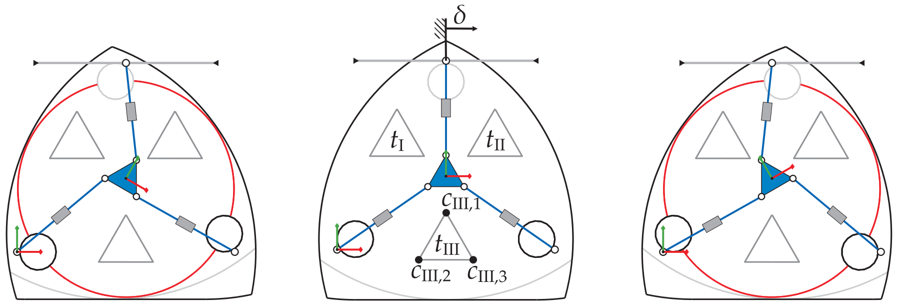
Table 2. Redundant 3($\underline{\text{P}}$)$\underline{\text{R}}$RR mechanism: maximal translational $\overline{\Delta xy}_\text{max}$ and rotational error $\overline{\Delta\phi}_\text{max}$ of the moving platform while moving along trajectory t_I, t_II, and t_III

to more appropriate switching patterns in comparison to an optimization based on the condition of the Jacobian $\eta(\boldsymbol{J}_\text{h})$. Regarding $t_\text{II}(30°)$, due to the additional active joint error $\Delta\delta$ there might be trajectory segments suffering from a decreased performance when using the proposed discrete optimization, i.e. the proposed switching patterns. This could be avoided using a continuous optimization. However, due to the mentioned advantages of the discrete switching patterns and due to the minimal decrease of the achievable accuracy only (Δxy : 0.05 mm and $\Delta\phi$: 0.03°), the authors still propose the selective discrete optimization of the redundant actuator position.

4.1.2 Redundant 3($\underline{\text{P}}$)R$\underline{\text{P}}$R mechanism

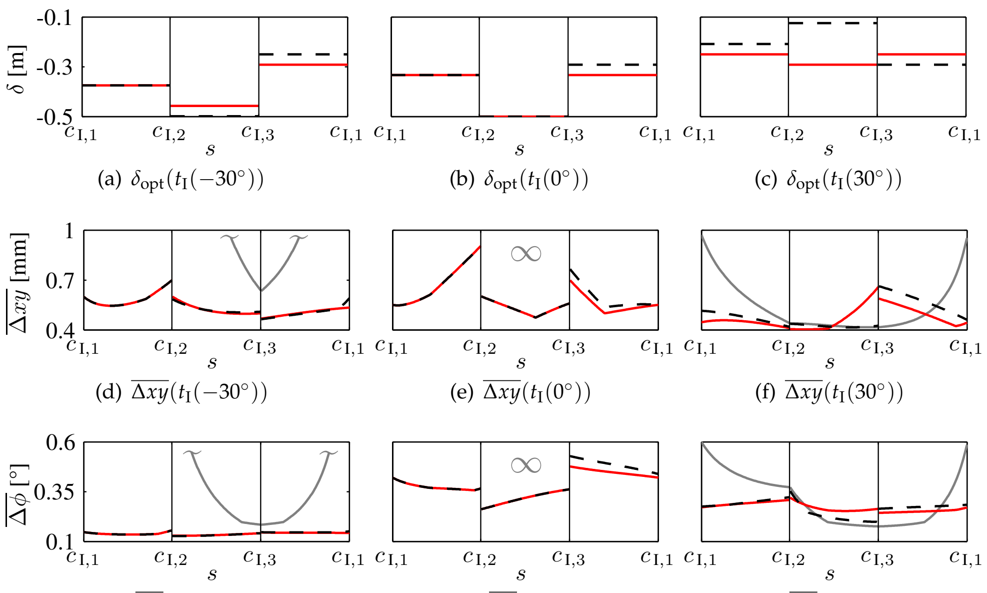
Similar to Sec. 4.1.1, an accuracy analysis of the kinematically redundant 3($\underline{\text{P}}$)R$\underline{\text{P}}$R mechanism is performed. Exemplarily, simulation results of the three triangular trajectories (t_I, t_II, t_III) which are shown in Fig. 8 are presented. In the following, facts and definitions similar to the analysis of the 3($\underline{\text{P}}$)$\underline{\text{R}}$RR mechanism and already introduced are not mentioned again. Based on the data sheets of commercially available standard actuators, the active joint errors were chosen to $\Delta\boldsymbol{\theta}$(3($\underline{\text{P}}$)R$\underline{\text{P}}$R) = (0.2 mm, 0.2 mm, 0.2 mm, 40 μm)$^\text{T}$. As well, in the non-redundant case the last element of $\Delta\boldsymbol{\theta}$ vanishes.

In Fig. 9 the optimized switching patterns δ_opt of the actuator position δ as well as the resulting pose errors $\overline{\Delta xy}$ and $\overline{\Delta\phi}$ of the mechanisms are presented. Again, the EE was moved counterclockwise along trajectory t_I with a constant orientation of $\phi = -30°$, $\phi = 0°$, and $\phi = 30°$. It is important to note that the symmetrical non-redundant mechanism suffers from a completely singular. i.e. useless, workspace for $\phi = 0°$ (indicated by $\overline{\Delta xy} = \overline{\Delta\phi} = \infty$). This is



(a) $3(\underline{\mathrm{P}})\mathrm{R}\underline{\mathrm{P}}\mathrm{R}$ ($\phi = -30°$) (b) $3(\underline{\mathrm{P}})\mathrm{R}\underline{\mathrm{P}}\mathrm{R}$ ($\phi = 0°$) (c) $3(\underline{\mathrm{P}})\mathrm{R}\underline{\mathrm{P}}\mathrm{R}$ ($\phi = 30°$)

Fig. 8. Exemplarily chosen trajectories t_{I}, t_{II}, t_{III} (solid gray) for the $3(\underline{\mathrm{P}})\mathrm{R}\underline{\mathrm{P}}\mathrm{R}$ mechanism, the solid red lines represent the singularity loci within the workspace (solid black); note: the workspace for $\phi = 0°$ is completely singular



(a) $\delta_{\mathrm{opt}}(t_{\mathrm{I}}(-30°))$ (b) $\delta_{\mathrm{opt}}(t_{\mathrm{I}}(0°))$ (c) $\delta_{\mathrm{opt}}(t_{\mathrm{I}}(30°))$

(d) $\overline{\Delta xy}(t_{\mathrm{I}}(-30°))$ (e) $\overline{\Delta xy}(t_{\mathrm{I}}(0°))$ (f) $\overline{\Delta xy}(t_{\mathrm{I}}(30°))$

(g) $\overline{\Delta\phi}(t_{\mathrm{I}}(-30°))$ (h) $\overline{\Delta\phi}(t_{\mathrm{I}}(0°))$ (i) $\overline{\Delta\phi}(t_{\mathrm{I}}(30°))$

Fig. 9. Simulation results while moving along trajectory $t_{\mathrm{I}}(-30°)$ (left), $t_{\mathrm{I}}(0°)$ (center), and $t_{\mathrm{I}}(30°)$ (right); solid gray: non-redundant mechanism; dashed black: optimized redundant mechanism using $\eta(\boldsymbol{J}_{\mathrm{h}})$; solid red: optimized redundant mechanism using $\gamma(\overline{\Delta \boldsymbol{x}_{\mathrm{h}}})$

not the case for the kinematically redundant $3(\underline{\mathrm{P}})\mathrm{R}\underline{\mathrm{P}}\mathrm{R}$ mechanism where the symmetry can be affected, i.e. avoided, thanks to the additional prismatic actuator. Regarding Fig. 9 and Table 3 similar to the $3\underline{\mathrm{R}}\mathrm{RR}$-based structure (see Sec. 4.1.1) a significant improvement of the achievable accuracy due to the kinematic redundancy is well noticeable. Again, in most cases (except

$t_i(\phi)$	Value	3R$\underline{\text{P}}$R	3($\underline{\text{P}}$)R$\underline{\text{P}}$R	
			using $\eta(\boldsymbol{J}_\text{h})$	using $\gamma(\overline{\Delta\boldsymbol{x}_\text{h}})$
$t_\text{I}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	4.87	0.70 (-85.7%)	0.70 (-85.7%)
	$\overline{\Delta\phi}_\text{max}$ [°]	1.54	0.16 (-89.9%)	0.16 (-89.9%)
$t_\text{I}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	∞	0.90 (-)	0.90 (-)
	$\overline{\Delta\phi}_\text{max}$ [°]	∞	0.53 (-)	0.48 (-)
$t_\text{I}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.97	0.66 (-31.8%)	0.66 (-32.5%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.60	0.35 (-41.1%)	0.32 (-46.6%)
$t_\text{II}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.97	0.66 (-31.9%)	0.86 (-11.7%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.60	0.32 (-46.6%)	0.35 (-41.9%)
$t_\text{II}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	∞	0.91 (-)	0.78 (-)
	$\overline{\Delta\phi}_\text{max}$ [°]	∞	0.48 (-)	0.44 (-)
$t_\text{II}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	4.87	0.70 (-85.7%)	0.64 (-86.8%)
	$\overline{\Delta\phi}_\text{max}$ [°]	1.54	0.16 (-89.9%)	0.15 (-90.2%)
$t_\text{III}(-30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	0.98	0.93 (-4.6%)	0.93 (-4.9%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.35	0.29 (-17.4%)	0.28 (-21.2%)
$t_\text{III}(0°)$	$\overline{\Delta xy}_\text{max}$ [mm]	∞	∞ (-)	∞ (-)
	$\overline{\Delta\phi}_\text{max}$ [°]	∞	∞ (-)	∞ (-)
$t_\text{III}(30°)$	$\overline{\Delta xy}_\text{max}$ [mm]	1.20	0.93 (-22.2%)	0.93 (-22.2%)
	$\overline{\Delta\phi}_\text{max}$ [°]	0.41	0.27 (-34.1%)	0.27 (-34.1%)

Table 3. Redundant 3($\underline{\text{P}}$)R$\underline{\text{P}}$R mechanism: maximal translational $\overline{\Delta xy}_\text{max}$ and rotational error $\overline{\Delta\phi}_\text{max}$ of the moving platform while moving along trajectory t_I, t_II, and t_III

for $t_\text{II}(-30°)$) the optimization based on the gain $\gamma(\overline{\Delta\boldsymbol{x}_\text{h}})$ leads to more appropriate switching patterns (in terms of accuracy improvement) in comparison to an optimization based on the Jacobian's condition $\eta(\boldsymbol{J}_\text{h})$. It is important to note, that even the redundant mechanism suffers from singularities (see $t_\text{III}(0°)$). This might be overcome by an optimization of the redundant actuator's design which will be subject to future work.

4.1.3 Influence of the redundant actuator's joint error

An additional test was performed to clarify the influence of the redundant prismatic actuator joint error $\Delta\delta$ on the moving platform pose error $\Delta\boldsymbol{x}$. Therefore, for different $\Delta\delta$ the EE was moved along I($-30°$). The actuator position δ was changed according to the optimized switching pattern shown in Fig. 7 and Fig. 9 (based on the gain $\gamma(\overline{\Delta\boldsymbol{x}_\text{h}})$). The results are presented in Fig. 10. The plots clearly demonstrate the marginal influence of $\Delta\delta$ on $\Delta\boldsymbol{x}$ when realistic values for the remaining active joint errors are chosen (cp. Sec. 4.1.1 and 4.1.2). It can be seen that even in the case of an unrealistic high joint error $\Delta\delta$ a significant increase of the mechanism's achievable accuracy in comparison to the non-redundant case is still obtained (cp. Fig. 7, left column).

4.1.4 Switching operations - accuracy progress

There might be the case that the EE passes a singular configuration while performing a reconfiguration of the mechanism, i.e. while changing the singularity loci. As a result, the performance of the PKM decreases dramatically. Hence, the switching operations have to be

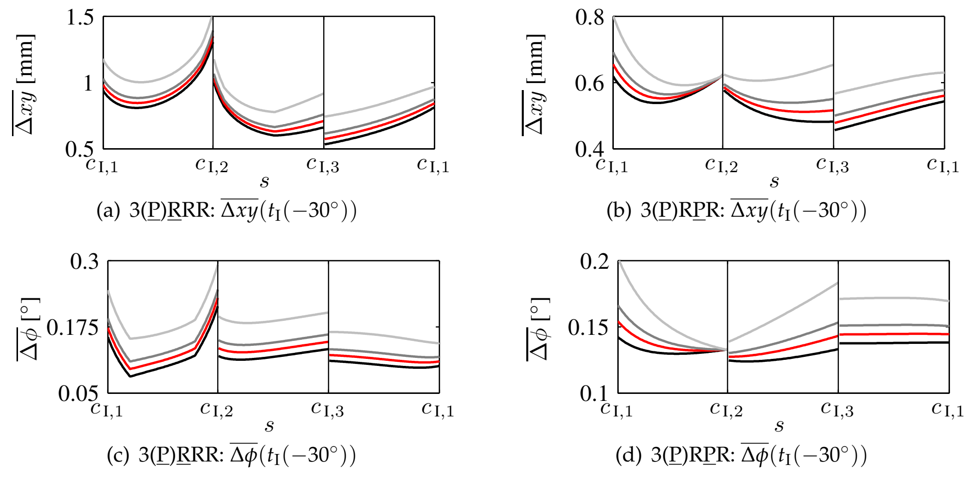


Fig. 10. Influence of $\Delta\delta$ on $\Delta\boldsymbol{x}$ while moving the EE along trajectory I$(-30°)$ (solid black: $\Delta\delta = 0\,\mu$m; solid red: $\Delta\delta = 50\,\mu$m; solid gray: $\Delta\delta = 100\,\mu$m; solid light gray: $\Delta\delta = 250\,\mu$m

considered within the optimization procedure. While performing a reconfiguration (moving δ while keeping $\boldsymbol{x}$ constant) the possibility of passing any singularities is taken into account. Additionally, configurations of low performance are avoided. Exemplarily, the behavior of the achievable accuracy obtained while moving the EE along $t_\mathrm{I}(-30°)$ (including the switching operations) is given in Fig. 11. It can be clearly seen that the achievable accuracy does not increase during reconfigurations of the mechanism. This is valid for all the trajectories the authors tested so far. A problem however is the additional operation time necessary to follow a desired path. This, i.e. the number of reconfigurations, could be reduced according to the modifications mentioned in Sec. 3.2, e.g. only change δ once before starting the desired movement or if the mechanism is unable to perform a desired operation. Furthermore, the switching time itself could be reduced by a 'semi discrete' optimization strategy, e.g. start moving δ shortly before arriving at the ending point $c_{i,j}$ of the segment j of trajectory i.

4.2 Comparing the useable workspace

In order to further clarify the effect of an additional prismatic actuator on the mechanism pose accuracy, in the following, the size of the useable workspaces w_u is determined. The useable workspace is defined as the singularity-free part of the total workspace w_t providing a certain desired performance, in this case a certain desired accuracy. Mathematically, it can be expressed as the largest region where the sign of the determinant of the Jacobian $\det(\boldsymbol{A})$ does not change and the output error $\Delta\boldsymbol{x}$ (23) satisfies any thresholds $\Delta\boldsymbol{x}_\mathrm{thr} = (\Delta xy_\mathrm{thr}, \Delta\phi_\mathrm{thr})^\mathrm{T}$, corresponding to $\overline{\Delta xy}$ and $\overline{\Delta\phi}$. Therefore, the Jacobian determinant as well as the moving platform pose error are calculated over the whole workspace. An example clarifying the procedure leading to w_u is given in Fig. 12. The analyzed workspaces for three different EE orientations of the non-redundant 3RRR mechanism ($\delta = 0\,$m $=$ const.) is given. The green part is the largest region where the sign of $\det(\boldsymbol{A})$ does not change whereas the red part is the smallest. The black area is the overlayed region where a required performance, i.e. a required accuracy,

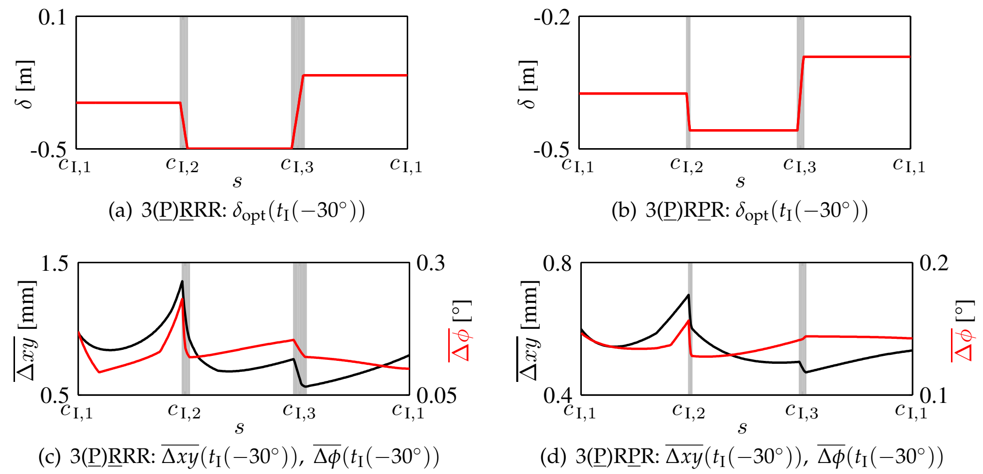


Fig. 11. Simulation results (including switching operations) while moving along trajectory $t_I(-30°)$, reconfigurations are performed based on the gain; left: 3(P)RRR, right: 3(P)RPR, the switching operation is marked by the gray background

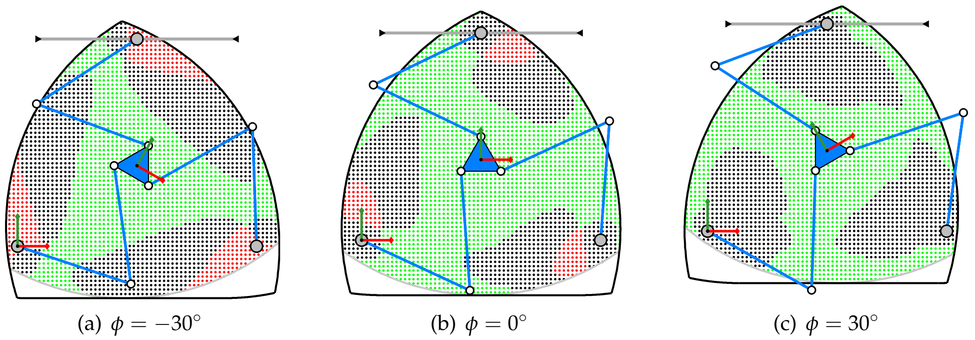
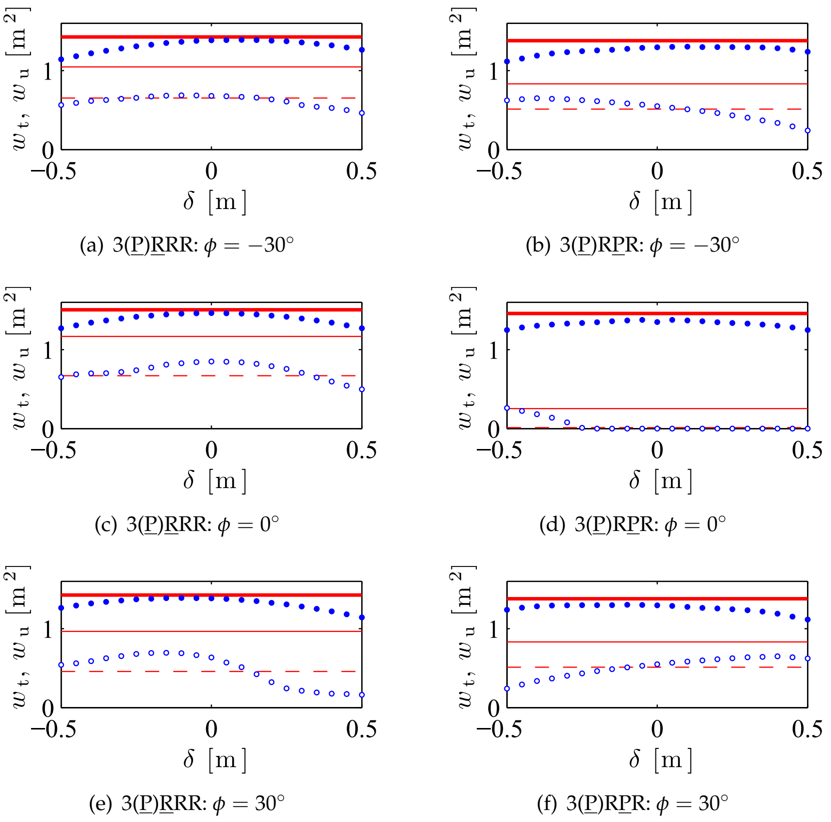


Fig. 12. Analyzed workspace of the non-redundant 3RRR mechanism ($\delta = 0\,\text{m} = \text{const.}$); green is largest region where the sign of det$(\boldsymbol{A})$ does not change whereas red is the smallest, in the black area the required accuracy can not be provided

can not be provided. Hence, the green color represents the useable workspace with respect to the mentioned requirements. That followed, the connected green area can be determined, i.e. the shape as well as the size of the useable workspace.
Three constant EE orientations $\phi = \{-30°, 0°, 30°\}$ were considered. The design of the exemplarily chosen mechanisms as well as the input error $\Delta\boldsymbol{\theta}$ are equal to the ones chosen in Sec. 4.1. The thresholds are set to $\Delta xy_{\text{thr}} = 0.75\,\text{mm}$ and $\Delta\phi_{\text{thr}} = 0.5°$. The results are given in Fig. 13. In case of the non-redundant mechanisms the total and useable workspace w_t and



(a) 3(P)RRR: $\phi = -30°$ (b) 3(P)RPR: $\phi = -30°$

(c) 3(P)RRR: $\phi = 0°$ (d) 3(P)RPR: $\phi = 0°$

(e) 3(P)RRR: $\phi = 30°$ (f) 3(P)RPR: $\phi = 30°$

Fig. 13. Total (bold lines, filled dots) and useable (light lines, unfilled dots) workspace of the kinematically redundant 3(P)RRR mechanism (left, solid red), the 3(P)RPR mechanism (right, solid red), and their non-redundant counterparts (left/right, dotted blue); the dashed red line gives the useable workspace of the redundant mechanisms for $\Delta \boldsymbol{x}_{\mathrm{thr}} = (0.5\,\mathrm{mm},\ 0.35°)^{\mathrm{T}}$

w_{u} was calculated for different base joint positions G_{1_i}, i.e. for different but constant δ_i. The solid horizontal lines represent w_{t} and w_{u} for the redundant case when the base joint G_1 can be moved linearly for $-0.5\,\mathrm{m} \leq \delta \leq 0.5\,\mathrm{m}$. Having a look at Fig. 13 a significant improvement concerning the workspace areas for all the considered EE orientations is well noticeable. Furthermore, for the redundant case the useable workspace for $\Delta \boldsymbol{x}_{\mathrm{thr}} = (0.5\,\mathrm{mm},\ 0.35°)^{\mathrm{T}}$ was determined, i.e. the requested accuracy is increased about one third. It can be clearly seen that in this case similar workspace sizes are obtained in comparison the non-redundant mechanisms with less accuracy requirements. This further demonstrates the use of kinematic redundancy in terms of accuracy improvements.

5. Conclusion

In this paper, the kinematically redundant 3($\underline{\mathrm{P}}$)$\underline{\mathrm{R}}$RR and 3($\underline{\mathrm{P}}$)R$\underline{\mathrm{P}}$R mechanisms were presented. In each case, an additional prismatic actuator was applied to the structure allowing one base joint to move linearly. After a description of some fundamentals of the proposed PKM, the effect of the additional DOF on the moving platform pose accuracy was clarified. An optimization of the redundant actuator position in a discrete manner was introduced. It is based on a minimization of a criterion that the authors denoted the gain $\gamma(\overline{\Delta \boldsymbol{x}_{\mathrm{h}}})$ of the maximal homogenized pose error $\overline{\Delta \boldsymbol{x}_{\mathrm{h}}}$. Using several exemplarily chosen trajectories a significant improvement in terms of accuracy of the proposed redundant mechanisms in combination with the developed optimization procedure was demonstrated. It could be seen that the suggested index $\gamma(\overline{\Delta \boldsymbol{x}_{\mathrm{h}}})$ leads to more appropriate switching patterns than the well known condition number of the Jacobian. Additional simulations demonstrated the marginal influence of the redundant actuator joint error $\Delta\delta$ on the moving platform pose error $\Delta \boldsymbol{x}$.
Furthermore, a comparative study on the usable workspaces, i.e. the singularity-free part of the total workspace providing a certain desired performance, of the mentioned mechanisms and their non-redundant counterparts was performed. The results demonstrate a significant increase of the useable workspace of all considered EE orientations thanks to the applied additional prismatic actuator.
To further increase the overall and the operational workspace, future work will deal with the design optimization of the prismatic actuator, e.g. its orientation with respect to the x-axis of the inertial coordinate frame as well as its stroke ('length'). In addition, the simulation will be extended to PKM with higher DOF and an experimental validation of the obtained numerical results will be performed.

6. References

Arakelian, V., Briot, S. & Glazunov, V. (2008). Increase of singularity-free zones in the workspace of parallel manipulators using mechanisms of variable structure, *Mech Mach Theory* **43**(9): 1129–1140.

Cha, S.-H., Lasky, T. A. & Velinsky, S. A. (2007). Singularity avoidance for the 3-RRR mechanism using kinematic redundancy, *Proc. of the 2007 IEEE International Conference on Robotics and Automation*, pp. 1195–1200.

Ebrahimi, I., Carretero, J. A. & Boudreau, R. (2007). 3-$\underline{\mathrm{PR}}$RR redundant planar parallel manipulator: Inverse displacement, workspace and singularity analyses, *Mechanism & Machine Theory* **42**(8): 1007–1016.

Gosselin, C. M. (1992). Optimum design of robotic manipulators using dexterity indices, *Robotics and Autonomous Systems* **9**(4): 213–226.

Gosselin, C. M. & Angeles, J. (1988). The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, *Journal of Mechanisms, Transmissions, and Automation in Design* **110**(1): 35–41.

Gosselin, C. M. & Angeles, J. (1990). Singularity analysis of closed-loop kinematic chains, *IEEE Transactions on Robotics and Automation* **6**(3): 281–290.

Hunt, K. H. (1978). *Kinematic Geometry of Mechanisms*, Clarendon Press.

Kock, S. (2001). *Parallelroboter mit Antriebsredundanz*, PhD thesis, Institute of Control Engineering, TU Brunswick, Germany.

Kock, S. & Schumacher, W. (1998). A parallel x-y manipulator with actuation redundancy for high-speed and active-stiffness applications, *Proc. of the 1998 IEEE International Conference on Robotics and Automation*, pp. 2295–2300.

Kotlarski, J., Abdellatif, H. & Heimann, B. (2007). On singularity avoidance and workspace enlargement of planar parallel manipulators using kinematic redundancy, *Proc. of the 13th IASTED International Conference on Robotics and Applications*, pp. 451–456.

Kotlarski, J., Abdellatif, H. & Heimann, B. (2008). Improving the pose accuracy of a planar 3$\underline{\text{R}}$RR parallel manipulator using kinematic redundancy and optimized switching patterns, *Proc. of the 2008 IEEE International Conference on Robotics and Automation*, Pasadena, USA, pp. 3863–3868.

Kotlarski, J., Abdellatif, H., Ortmaier, T. & Heimann, B. (2009). Enlarging the useable workspace of planar parallel robots using mechanisms of variable geometry, *Proc. of the ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots*, London, United Kingdom, pp. 94–103.

Kotlarski, J., de Nijs, R., Abdellatif, H. & Heimann, B. (2009). New interval-based approach to determine the guaranteed singularity-free workspace of parallel robots, *Proc. of the 2009 International Conference on Robotics and Automation*, Kobe, Japan, pp. 1256–1261.

Kotlarski, J., Do Thanh, T., Abdellatif, H. & Heimann, B. (2008). Singularity avoidance of a kinematically redundant parallel robot by a constrained optimization of the actuation forces, *Proc. of the 17th CISM-IFToMM Symposium on Robot Design, Dynamics, and Control*, Tokyo, Japan, pp. 435–442.

Merlet, J.-P. (1996). Redundant parallel manipulators, *Laboratory Robotics and Automation* **8**(1): 17–24.

Merlet, J.-P. (2006a). Computing the worst case accuracy of a pkm over a workspace or a trajectory, *Proc. of the 5th Chemnitz Parallel Kinematics Seminar*, pp. 83–96.

Merlet, J.-P. (2006b). *Parallel Robots (Second Edition)*, Springer.

Merlet, J.-P. & Daney, D. (2005). Dimensional synthesis of parallel robots with a guaranteed given accuracy over a specific workspace, *Proc. of the 2005 IEEE International Conference on Robotics and Automation*, pp. 942–947.

Mohamed, M. G. & Gosselin, C. M. (2005). Design and analysis of kinematically redundant parallel manipulators with configurable platforms, *IEEE Transactions on Robotics* **21**(3): 277–287.

Müller, A. (2005). Internal preload control of redundantly actuated parallel manipulators & its application to backlash avoiding control, *IEEE Transactions on Robotics and Automation* **21**(4): 668–677.

Pond, G. & Carretero, J. A. (2006). Formulating jacobian matrices for the dexterity analysis of parallel manipulators, *Mechanism & Machine Theory* **41**(12): 1505–1519.

Wang, J. & Gosselin, C. M. (2004). Kinematic analysis and design of kinematically redundant parallel mechanisms, *Journal of Mechanical Design* **126**(1): 109–118.

Yang, G., Chen, W. & Chen, I.-M. (2002). A geometrical method for the singularity analysis of 3-RRR planar parallel robots with different actuation schemes, *Pro. of the 2002 IEEE International Conference on Intelligent Robots and Systems*, pp. 2055–2060.

Zein, M., Wenger, P. & Chablat, D. (2006). Singular curves and cusp points in the joint space of 3-RPR parallel manipulators, *Proc. of the 2006 IEEE International Conference on Robotics and Automation*, pp. 777–782.

Kinematic Singularities of Robot Manipulators

Peter Donelan
Victoria University of Wellington
New Zealand

1. Introduction

Kinematic singularities of robot manipulators are configurations in which there is a change in the expected or typical number of instantaneous degrees of freedom. This idea can be made precise in terms of the rank of a Jacobian matrix relating the rates of change of input (joint) and output (end-effector position) variables. The presence of singularities in a manipulator's effective joint space or work space can profoundly affect the performance and control of the manipulator, variously resulting in intolerable torques or forces on the links, loss of stiffness or compliance, and breakdown of control algorithms. The analysis of kinematic singularities is therefore an essential step in manipulator design. While, in many cases, this is motivated by a desire to avoid singularities, it is known that for almost all manipulator architectures, the theoretical joint space must contain singularities. In some cases there are potential design advantages in their presence, for example fine control, increased load-bearing and singularity-free posture change.

There are several distinct aspects to singularity analysis—in any given problem it may only be necessary to address some of them. Starting with a given manipulator architecture, manipulator kinematics describe the relation between the position and velocity (instantaneous or infinitesimal kinematics) of the joints and of the end-effector or platform. The physical construction and intended use of the manipulator are likely to impose constraints on both the input and output variables; however, it may be preferable to ignore such constraints in an initial analysis in order to deduce subsequently joint and work spaces with desirable characteristics. A common goal is to determine maximal singularity-free regions. Hence, there is a *global* problem to determine the whole locus of singular configurations. Depending on the architecture, one may be interested in the singular locus in the joint space or in the work space of the end-effector (or both). A more detailed problem is to classify the types of singularity within the critical locus and thereby to stratify the locus. A *local* problem is to determine the structure of the singular locus in the neighbourhood of a particular point. For example, it may be important to know whether the locus separates the space into distinct subsets, a strong converse to this being that a singular configuration is isolated.

Typically, there will be a number of design parameters for a manipulator with given architecture—link lengths, twists and offsets. *Bifurcation* analysis concerns the changes in both local and global structure of the singular locus that occur as one alters design parameters in a given architecture. The design process is likely to involve optimizing some desired characteristic(s) with respect to the design parameters.

The aim of this Chapter is to provide an overview of the development and current state of kinematic singularity research and to survey some of the specific methodology and results in the literature. More particularly, it describes a framework, based on the work of Müller (2006; 2009) and of the author (Donelan, 2007b; 2008), in which singularities of both serial and parallel manipulators can be understood.

2. Origins and Development

The origins of the the study of singularities in mechanism and machine research literature go back to the 1960s and relate particularly to determination of the degree of mobility via screw theory (Baker, 1978; Waldron, 1966), the study of over-constrained closed chains (Duffy & Gilmartin, 1969) and the analysis of inverse kinematics for serial manipulators. Pieper (1968) showed that the inverse kinematic problem could be solved explicitly for wrist-partitioned manipulators, typical among serial manipulator designs. Generally, this remains a major problem in manipulator kinematics and singularity analysis. In the context of control methods, Whitney introduced the Jacobian matrix (Whitney, 1969) and this has become the central object in the study of instantaneous kinematics of manipulators and their singularities—a number of significant articles appeared in the succeeding years (Featherstone, 1982; Litvin et al., 1986; Paul & Shimano, 1978; Sugimoto et al., 1982; Waldron, 1982; Waldron et al., 1985); now the literature on kinematic singularities is very extensive, numbering well over a thousand items.

Interest in parallel mechanisms also gained momentum in the 1980s. Hunt (1978) proposed the use of in-parallel actuated mechanisms, such as the Gough–Stewart platform (Dasgupta & Mruthyunjaya, 2000), as robot manipulators, given their advantages of stiffness and precision. In contrast to serial manipulators, where the forward kinematic mapping is constructible and its singularities correspond to a loss of degrees of freedom in the end-effector, for parallel manipulators, the inverse kinematics is generally more tractable and its singularities correspond to a gain in freedom for the platform or end-effector (Fichter, 1986; Hunt, 1983). While screw theory already played a role in analyzing singularities, Merlet (1989) showed that Grassmann line geometry, which could be viewed as a subset of screw theory (see Section 4.2), is sufficiently powerful to explain the singular configurations of the Gough–Stewart platform. Thereafter, a Jacobian-based approach to understanding parallel manipulator singularities was provided by Gosselin & Angeles (1990), who showed that they could experience both direct and inverse kinematic singularities and indeed a combination of these. Subsequently, the subtlety of parallel manipulator kinematics has become even more apparent, in part as a result of the development of manipulators with restricted types of mobility, such as translational and Schönflies manipulators (Carricato, 2005; Di Gregorio & Parenti-Castelli, 2002).

The difficulty in resolving the precise configuration space and singularity locus have meant that a great deal of the singularity analysis takes a localized approach—one assumes a given configuration for the manipulator and then determines whether it is a singular configuration. It may also be possible to determine some local characteristics of the locus of singularities. This is remarkably fruitful: by choosing coordinates so that the given configuration is the identity or home configuration it is possible to reason about necessary conditions for singularity in terms of screws and screw systems. The difficulty that arises in deducing the global structure of the singularity locus is that there is no straightforward way to solve the necessary inverse kinematics. A good deal of progress can be made in some problems using Lie algebra and properties of the closure subalgebra of a chain (open or closed). This approach can be found in (Hao, 1998; Rico et al., 2003) but it appears to fail for the mechanisms dubbed "paradoxical" by Hervé (1978).

A number of authors have sought to apply methods of mathematical singularity theory to the study of manipulator singularities, for example Gibson (1992); Karger (1996); Pai & Leu (1992); Tchoń (1991). A recent survey of this approach can be found in (Donelan, 2007b). There are several salient features. Firstly, the kinematic mapping is explicitly recognized as a function between manifolds, though it may not be given explicit form. Secondly, singularities may be classified not only on the basis of their kinematic status but also in terms of intrinsic characteristics of the mapping. For example, the rank deficiency (corank) of the kinematic mapping is a simple discriminator. More subtle higher-order distinctions can be made that distinguish between the topological types of the local singularity locus and enable it to be stratified. Thirdly, it provides a precise language and machinery for determining *generic* properties of the kinematics.

Following the results of Merlet (1989), another approach has been to use geometric algebra, especially in the analysis of parallel manipulator kinematics and singularities. It is a common theme that singular configurations correspond to special configurations of points, lines and planes associated with a manipulator—for example coplanarity of joint axes or collinearity of spherical joints. Such conditions can be expressed as simple equations in the appropriate algebra. Some examples of recent successful application of these techniques are Ben-Horin & Shoham (2009); Torras et al. (1996); White (1994); Wolf et al. (2004).

3. Manipulator Architecture and Mobility

A robot manipulator is assumed to consist of a number of rigid components (links), some pairs of which are connected by joints that are assumed to be Reuleaux lower pairs, so representable by the contact of congruent surfaces in the connected pair of links (Hunt, 1978). These include three types with one degree of freedom (dof): revolute R, prismatic P and helical or screw H (the first two correspond to purely rotational and purely translational freedom respectively) and three types having higher degree of freedom: cylindrical C with 2 dof, planar E and spherical S each with 3 dof. Some manipulators include universal U joints consisting of two R joints with intersecting axes, also denoted (RR).

The *architecture* of a manipulator is essentially a topological description of its links and joints: it can be determined by a graph whose vertices are the links and whose edges represent joints (Müller, 2006). A *serial manipulator* is an open chain consisting of a sequence of pairwise joined links, the initial (base) and final (end-effector) links only being connected to one other link. If the initial and final links of a serial manipulator are connected to each other, the result is a *simple closed chain*. This is the most basic example of a *parallel manipulator*, that is a manipulator whose topological representation includes at least one cycle or loop. Note that manipulators such as multi-fingered robot hands are neither serial nor parallel—their graph is a tree and the relevant kinematics are likely to concern the simultaneous placement of each finger.

Associated with the architecture is a combinatorial invariant, the *(full-cycle) mobility* μ of the manipulator, which is its total internal or relative number of degrees of freedom. This is given by the formula of Chebychev–Grübler–Kutzbach (CGK) (Hunt, 1978; Waldron, 1966):

$$\mu = n(l-1) - \sum_{i=1}^{k}(n-\delta_i) = \sum_{i=1}^{k}\delta_i - n(k-l+1) \tag{1}$$

where n is the number of degrees of freedom of an unconstrained link ($n = 6$ for spatial, $n = 3$ for planar or spherical manipulators), k is the number of joints, l the number of links and δ_i the dofs of the ith joint. The first expression represents the difference between the total freedom of the links and the constraints imposed by the joints. The second version emphasizes that the mobility is the difference between the total joint dofs and the number of constraints as expressed by the dimension of the cycle space of the associated graph (Gross & Yellen, 2004). A specific manipulator requires more information, determining the variable design parameters inherent in the architecture. The formula (1) is *generic* (Müller, 2009): there may be specific realizations of an architecture for which the formula does not give the true mobility. For example, the Bennett mechanism consists of 4 links connected by 4 revolute joints into a closed chain and is designed so that the axes lie pairwise on the two sets of generators of a hyperboloid. The CGK formula gives $\mu = 6 \times (4 - 1) - \sum_{i=1}^{4}(6 - 1) = -2$ but in fact the mechanism is mobile with 1 dof. Instances of an architecture in which (1) underestimates the true mobility are termed *over-constrained*. In other cases, there are specific configurations in which μ does not coincide with the infinitesimal freedom of the manipulator. This has given rise to a search for a more universal formula that takes into account the non-generic cases, see for example (Gogu, 2005). It is precisely the discrepancies that arise which correspond to forms of singularity that are explored in subsequent sections.

4. The Kinematic Mapping

In the robotics literature, the Jacobian matrix for a serial manipulator is the linear transformation that relates joint velocities to end-effector velocities. Explicitly, suppose the joint variables are $\mathbf{q} = (q_1, \ldots, q_k)$ and the end-effector's position is described by coordinates $\mathbf{x} = (x_1, \ldots, x_n)$. Thus, k is the total number of the joints (or total degrees of freedom, if any have ≥ 2 dof) of the joints, while n is the dimension of the displacement space of the end-effector, typically either $n = 3$ for planar or spherical motion or $n = 6$ for full spatial motion. The *kinematic mapping* is a function $\mathbf{x} = f(\boldsymbol{\theta})$ that determines the displacement of the end-effector corresponding to given values of the joint variables. Then for a time-dependent motion described by $\mathbf{q} = \mathbf{q}(t)$, at a configuration $\mathbf{q}_0 = \mathbf{q}(t_0)$ say,

$$\dot{\mathbf{x}}(t_0) = Jf(\mathbf{q}_0)\dot{\mathbf{q}}(t_0) \tag{2}$$

where $J = Jf(\mathbf{q}_0)$ is the $n \times k$ matrix of partial derivatives of f with ijth entry $\partial f_i / \partial q_j$. It is important to note that the linear relation expressed by (2) holds infinitesimally; the Jacobian matrix is itself dependent on the joint variables. In many practical situations, for example in control algorithms, the requirement is to find an inverse matrix for J, which is only possible when $k = n$ and the determinant of the Jacobian is non-zero. In the case $k > n$, the kinematics are said to be *redundant* and one may seek a pseudo-inverse. In the case of wrist-partitioned serial manipulators, the Jacobian itself partitions in a natural way and so the singularity loci of such manipulators can also be analyzed relatively simply (Stanišić & Engleberth, 1978). It is not essential to consider the time-dependence of a motion: from the point of view of the manipulator's capabilities, the Jacobian is determined by the choice of coordinates for joints and end-effector so the properties of interest are those that are invariant under change of coordinates. This is made clearer if the domain and range of the kinematic mapping f are properly defined.

4.1 Displacement groups

The range is the set of rigid displacements of the end-effector. The rigidity of the links of a manipulator, including its end-effector, means that their motion in space is assumed to be isometric (distance between any pair of points is preserved) and orientation-preserving. Assuming the ambient space to be Euclidean, the resulting set of possible displacements is the *spatial Euclidean (isometry) group* $SE(3)$ (Murray et al., 1994; Selig, 2005). Composition of displacements and the existence of an inverse displacement ensure that, mathematically, this set is a group. Moreover it can be, at least on a neighbourhood of every point, given Euclidean coordinates, and hence forms a Lie group having compatible spatial or topological (differentiable manifold) and algebraic (group) structures. The number of independent coordinates required is the dimension of the Lie group and $SE(3)$ is 6-dimensional. It is, via choices of an origin and 3-dimensional orthonormal coordinates in the ambient space and in the link, isomorphic (that is to say topologically and algebraically equivalent to) to the semi-direct product $SO(3) \ltimes \mathbb{R}^3$, where the components of the product correspond to the orientation-preserving rotations about a fixed point (the origin) and translations, respectively.

For planar manipulators, analogously, the Euclidean isometry group is the 3–dimensional group $SE(2) \cong SO(2) \ltimes \mathbb{R}^2$. Every element of the rotation group $SO(2)$ may be written, with respect to given orthonormal basis for $\mathbb{R}^2$, in the form:

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{3}$$

where θ measures the angle of counter-clockwise rotation. In this case, θ is a coordinate for the 1-dimensional group $SO(2)$ which can, in fact, be used globally, though it is not one-to-one. Indeed, it is clear that $SO(2)$ is, in a topological sense, the same as a unit circle, denoted S^1 and θ and $\theta + 2p\pi$ represent the same point for any integer p. Topologically, $SO(3)$ is 3-dimensional projective space. Coordinates may be chosen in a number of ways. The dimensions correspond locally to the rotation about each of three perpendicular axes, or one can use Euler angles. Alternatively, there is a 2:1 representation by points of a 3-dimensional sphere, and unit quaternions or Euler–Rodrigues parameters are often used.

Regarding the joint variables, the motion associated with each Reuleaux pair corresponds to a subgroup of $SE(3)$ of the same dimension as its degrees of freedom (Hervé, 1978). In particular R, P and H joints can be represented by one-dimensional subgroups of the Euclidean group. For an R joint, the subgroup is topologically S^1, while for P and H joints the group is $\mathbb{R}$. For an S joint, the subgroup is (a copy of) $SO(3)$; for C and E joints the subgroups are equivalent to $S^1 \times \mathbb{R}$ and $\mathbb{R} \times \mathbb{R}$ respectively. Depending on the architecture of the manipulator, its joint variables therefore lie in a product of components, each of the form either S^1, $\mathbb{R}$ or $SO(3)$, and this product Q, say, forms the theoretical domain of the kinematic mapping. As mentioned previously, however, there are in practice almost certainly restrictions on the joint variables so that the actual domain is some subset of the theoretical joint space. The set Q is also a manifold for which the joint variables give coordinates, say $\mathbf{q} = (q_1, \ldots, q_k)$, at least locally though not necessarily in a one-to-one manner for the whole space at once.

The kinematic mapping has the form of a function $f : Q \to G$, where Q is the joint space and G the displacement group for the end-effector, are well-defined manifolds. Local coordinates enable f to be expressed explicitly as a formula. While G is usually taken to be $SE(3)$, for spherical manipulators in which there is a fixed point for the end-effector $G = SO(3)$. For a robot hand or multi-legged walking robot, G may be a product of several copies of $SE(3)$. For a positional manipulator, for example a 3R arm assembly that determines the wrist-centre for a wrist-partitioned serial manipulator, the range is simply $\mathbb{R}^3$.

4.2 Infinitesimal kinematics

Associated with a given point in either of these spaces $q \in Q$, $g \in G$, is its *tangent space*, denoted T_qQ, T_gG, consisting of tangent vectors of paths through that point. The tangent spaces are vector spaces of the same dimension as the corresponding manifold. In terms of a choice of local coordinates $\mathbf{q}$ on a neighbourhood of $q \in Q$ and $\mathbf{x}$ near $g \in G$, these tangent vectors will correspond to the vectors $\dot{\mathbf{q}}$, $\dot{\mathbf{x}}$. If $g = f(q)$ then there is a linear transformation $T_qf : T_qQ \to T_gG$ whose matrix representation is simply the Jacobian matrix $Jf(\mathbf{q})$.
Working locally, we are free to choose coordinates so that the given configuration is the identity $e \in G$ and so we are interested in T_eG especially. This vector space represents infinitesimal displacements of the end-effector. It has additional structure, namely that of a Lie algebra, characteristically denoted $\mathfrak{g}$: it has a bilinear, skew-symmetric "bracket" product $[u_1, u_2] \in \mathfrak{g}$ for $u_1, u_2 \in \mathfrak{g}$, that satisfies an additional property, the Jacobi identity. See, for example, Murray et al. (1994); Selig (2005) for further details in the context of robot kinematics. The bracket provides a measure of the failure of a pair of displacements to commute.
For the Euclidean group of displacements of a rigid body in $\mathbb{R}^3$, its Lie algebra $\mathfrak{se}(3)$ inherits the semi-direct product structure of $SE(3)$ and is isomorphic to $\mathfrak{so}(3) \oplus \mathfrak{t}(3)$; the infinitesimal rotations $\mathfrak{so}(3)$ can be represented by 3×3 skew-symmetric matrices

$$\Omega = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \tag{4}$$

or equivalently the corresponding 3-vector $\boldsymbol{\omega}$ that spans the kernel of the matrix ($\mathbf{0}$ for the zero matrix); the infinitesimal translations are also 3-vectors $\mathbf{v}$. Hence elements $X \in \mathfrak{se}(3)$, termed *twists* are represented by pairs of 3-vectors $(\boldsymbol{\omega}, \mathbf{v})$. For $X \neq 0$, the line in $\mathfrak{se}(3)$ spanned by X is called a *screw*.
The Lie bracket on $\mathfrak{se}(3)$ can be defined in terms of the standard vector product $\times$ on $\mathbb{R}^3$ by

$$[(\boldsymbol{\omega}_1, \mathbf{v}_1), (\boldsymbol{\omega}_2, \mathbf{v}_2)] = (\boldsymbol{\omega}_1 \times \boldsymbol{\omega}_2, \boldsymbol{\omega}_1 \times \mathbf{v}_2 + \mathbf{v}_1 \times \boldsymbol{\omega}_2) \tag{5}$$

Thought of as a 6-vector, the components of this representation are generally known as *screw coordinates* though more properly they are twist coordinates. The *pitch* of a twist $(\boldsymbol{\omega}, \mathbf{v})$ is the ratio of the two fundamental invariants, the Klein and Killing forms, expressed as scalar products of the component 3-vectors of the twists as:

$$h = \frac{\boldsymbol{\omega} \cdot \mathbf{v}}{\boldsymbol{\omega} \cdot \boldsymbol{\omega}} \tag{6}$$

A twist X of pitch 0 corresponds to infinitesimal rotation about an axis and the corresponding screw can be identified with the line in $\mathbb{R}^3$ that is its axis; in that case, screw coordinates are identical with classical Plücker line coordinates. When $\boldsymbol{\omega} = \mathbf{0}$, the pitch is not well-defined by this formula but is conventionally said to be ∞. Note that the pitch is in fact an invariant of screws, not just twists.
The Klein form arises as the quadratic form associated with the non-degenerate bilinear form

$$Q_0((\boldsymbol{\omega}_1, \mathbf{v}_1), (\boldsymbol{\omega}_2, \mathbf{v}_2)) = \tfrac{1}{2}(\boldsymbol{\omega}_1 \cdot \mathbf{v}_2 + \boldsymbol{\omega}_2 \cdot \mathbf{v}_1) \tag{7}$$

The form Q_0 gives rise to a natural pairing between the Lie algebra and its dual space of wrenches (force plus torque) so it is possible to identify wrenches and twists. As Q_0 is indefinite, there exist non-zero *reciprocal twists* X_1, X_2 satisfying $Q_0(X_1, X_2) = 0$. In statics, a wrench X_2 is reciprocal to a twist X_1 if it performs no work on a body free to move along X_1.

In order to describe the infinitesimal capabilities of a rigid body with several degrees of freedom, define a *screw system* to be any subspace $S \subseteq \mathfrak{se}(3)$ (Davidson & Hunt, 2004; Gibson & Hunt, 1990). If $X_1, \ldots, X_k$ form a set of independent twists, then they span a k-(screw) system. Associated with S is a reciprocal $(6-k)$-system $S^\perp$ consisting of the constraints or wrenches that perform no work when acting on the end-effector. (These are not necessarily distinct from S.)

4.3 Product of exponentials

It is valuable to have a standard form in which to express the kinematic mapping. In particular, the *product of exponentials*, allows us to make use of the rich theory of Lie groups. For any Lie group G there is a natural *exponential mapping*, exp, from the Lie algebra $\mathfrak{g}$ to the group G itself. When the elements of G, and hence $\mathfrak{g}$, are written as matrices then for any matrix $U \in \mathfrak{g}$ and $q \in \mathbb{R}$,

$$\exp(qU) = \sum_{n=0}^{\infty} \frac{U^n}{n!} q^n \in G \tag{8}$$

The *one-parameter subgroups* (i.e. 1-dimensional) of a Lie group G always have the form the form $\exp(qX)$, where $X \neq 0$ is an element of the Lie algebra $\mathfrak{g}$ and $q \in \mathbb{R}$. As noted previously, these correspond to the motions generated by R, P and H joints. Note that any non-zero multiple of X may be used to represent the same joint: in effect, the joint is uniquely represented by a *screw*. For an R joint the pitch $h = 0$, while for a P joint $h = \infty$.
Brockett (1984) adapted an approach for representing kinematic mappings originally due to Denavit & Hartenberg (1955) and demonstrated that the motion of the end-effector of a serial manipulator can be written as a product of exponentials:

$$f(q_1, \ldots, q_k) = \exp(q_1 X_1) \cdot \exp(q_2 X_2) \cdots \exp(q_k X_k) \tag{9}$$

where q_i denotes the joint variable of the ith joint and X_i its twist relative to a fixed set of base link coordinates, for $i = 1, \ldots, k$. An alternative approach uses coordinates in each link and expresses the invariant relation between successive links by a standard form of matrix in terms of its (Denavit–Hartenberg) design parameters. However the pure product of exponentials formulation permits the use of a classical formula from Lie theory, the Baker–Campbell–Hausdorff (BCH) formula (Donelan, 2007b; Selig, 2005). Given f in the form (9), since $\frac{d}{dq}\exp(qX)\Big|_{q=0} = X$ then at $q_1 = \cdots = q_k = 0$, the twists $X_1, \ldots, X_k$ span a screw system S of dimension $\delta \leq k$, that describes the infinitesimal capabilities of the end-effector. Here, δ is the *infinitesimal mobility*.
The product-of-exponentials formalism (9) can be extended to chains that include spherical joints. For any point $\mathbf{a} \in \mathbb{R}^3$ there is a 3-dimensional subgroup $G_\mathbf{a} \in SE(3)$ of rotations about $\mathbf{a}$, leaving $\mathbf{a}$ fixed. If $\mathfrak{g}_\mathbf{a} \subset \mathfrak{se}(3)$ is the corresponding Lie subalgebra then the restriction of the exponential on $\mathfrak{se}(3)$ to $\mathfrak{g}_\mathbf{a}$ is surjective so that every element of $G_\mathbf{a}$ can be written (not uniquely) in the form $\exp(q_1X_1 + q_2X_2 + q_3X_3)$ where X_i, $i = 1, 2, 3$ form a basis for $\mathfrak{g}_\mathbf{a}$; for example they can be taken to be infinitesimal rotations about three orthogonal lines through $\mathbf{a}$. The product then includes an exponential of this extended form.

5. Serial Manipulator Singularities

The most important characteristic of a linear transformation, invariant under linear change of coordinates, is its *rank*, the dimension of its image. Since a linear transformation cannot increase dimension and the image is itself a subspace of the range, the rank can be no greater than either the dimension of the domain or of the range of the transformation. This provides the basis for the precise definition of a singularity:

Definition 1. *A serial manipulator with kinematic mapping $f : Q \to G$, where the joint space Q has dimension k and the displacement group G has dimension n, has a* **kinematic singularity** *at $q \in Q$ if rank $T_q f$ has rank less than both k and n.*

In particular, if $k = n$ then there is a kinematic singularity when rank $T_q f < n$, or equivalently $\det Jf(q) = 0$. In terms of mobility, the definition is equivalent to $\delta < \mu$.
Using topological methods, it can be shown that for standard architectures such as 6R serial manipulators, where the joint space is the 6-dimensional torus $Q = T^6$ and the end-effector space is $SE(3)$, the kinematic mapping must have singularities (Gottlieb, 1986). These can only be excluded from the the joint space by imposing restrictions on the joint variables. In particular, if the joint space is compact, such as T^6, then so its image, the *work space*, and is hence bounded in $SE(3)$. The kinematic mapping will have singularities at the boundary configurations. However it may also have singularities interior to the workspace.
A fundamental problem is to determine the locus of kinematic singularities in the joint space of a manipulator and, if possible, to stratify it in a natural way. As has been mentioned before, actually determining the locus remains a difficult problem for most manipulators. However one can address the question of whether the locus is itself a submanifold of the joint space using singularity theory methods. The singularities of rank deficiency 1 (corank 1) can be recognized by whether, at a given configuration, the first-order Taylor polynomial of the kinematic mapping $f : Q \to SE(3)$ lies in a certain manifold. Transversality to this manifold, a linear-algebraic condition that holds when a certain set of twists span the Lie algebra $\mathfrak{se}(3)$, is enough to guarantee that the corank 1 part of the singularity locus is a manifold of dimension $|6-k|+1$. The resulting condition involves the joint twists and certain Lie brackets involving them (Donelan, 2008).
The approach can also, in principle, be extended to a manipulator architecture by including the design parameters. In this case, one may expect to encounter more degenerate singularities for specific values of the design parameter. An important problem is to determine the bifurcation set that separates classes within the architecture of manipulators with distinct topological type of singularity locus.

6. Parallel Manipulator Singularities

6.1 Infinitesimal analysis of closure

In contrast to serial manipulators, in which all the joints are actuated, for a parallel manipulator only some of the joints will be actuated, the remainder being passive. The approach to parallel manipulator singularities pioneered by Gosselin & Angeles (1990) makes use of the actuated joint variables $\mathbf{q}$ and output (end-effector) variables $\mathbf{x}$, constrained by an equation of the form:

$$F(\mathbf{q}, \mathbf{x}) = 0 \tag{10}$$

Differentiating (10), the infinitesimal kinematics can be written in the form:

$$\begin{pmatrix} \dfrac{\partial F}{\partial \mathbf{q}} & \vdots & \dfrac{\partial F}{\partial \mathbf{x}} \end{pmatrix} \begin{pmatrix} \dot{\mathbf{q}} \\ \dot{\mathbf{x}} \end{pmatrix} = J_q \dot{\mathbf{q}} + J_x \dot{\mathbf{x}} = \mathbf{0} \tag{11}$$

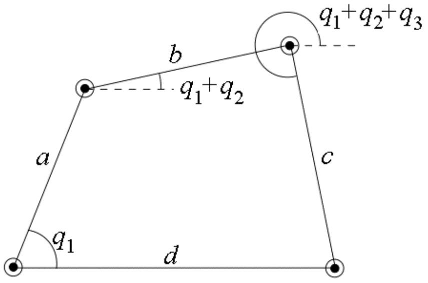


Fig. 1. Planar 4R manipulator

where the two Jacobian matrices $J_q = \partial F/\partial \mathbf{q}, J_x = \partial F/\partial \mathbf{x}$ depend on both $\mathbf{q}$ and $\mathbf{x}$. If one assumes no redundancy, then the number of actuator variables equals the number of output variables, i.e. the mobility μ of the manipulator. The number of constraints, the dimension of the range of F, should then be the difference between the number of variables 2μ and the mobility μ, hence also μ. Thus the Jacobians are both $\mu \times \mu$ matrices.

This leads to three types of singularity, depending on loss of rank of J_q (type I), or J_x (type II), or both (type III). A type I singularity is comparable to the kinematic singularity of a serial manipulator in Definition 1, where there exist theoretical end-effector velocities that are not achievable by means of any input joint variable velocity. On the other hand, in a type II singularity, there is a non-zero vector $\dot{\mathbf{x}}$ in the kernel of J_x which therefore gives a solution to (11) with $\dot{\mathbf{q}} = \mathbf{0}$, in other words when the actuated joints are static or locked. Such a solution may be isolated but may also correspond to a finite branch of motion, referred to as an *architecture singular motion* (Ma & Angeles, 1992) or *self-motion* (Karger & Husty, 1998). (Note that this term is also used for motions of a serial redundant manipulator in which the end-effector remains static.)

In practice, as noted by Gosselin & Angeles (1990), the constraint equations one actually formulates do not necessarily have the form (10). Denavit & Hartenberg (1955) observed that for a closed chain the matrix product must be the identity and, likewise, for the product-of-exponentials formulation. For a simple closed chain, the end-effector is identified with the base, so gives rise to a *closure equation* for the kinematic mapping of the form:

$$\phi(q_1, \ldots, q_k) = \exp(q_1 X_1) \cdot \exp(q_2 X_2) \cdots \exp(q_k X_k) = I \qquad (12)$$

where I is the identity transformation that leaves the end-effector fixed with respect to the base link.

By way of a relatively simple example, consider a planar 4R closed chain (Gibson & Newstead, 1986) where the closure equation involves the design parameters (link lengths) a, b, c, d and four joint variables q_i, $i = 1, 2, 3, 4$ as in Figure 1. Since the kinematics concern $SE(2)$, which is 3-dimensional, there are three scalar closure equations, two for translation and one for rotation:

$$a\cos q_1 + b\cos(q_1+q_2) + c\cos(q_1+q_2+q_3) + d\cos(q_1+q_2+q_3+q_4) = 0 \qquad (13)$$
$$a\sin q_1 + b\sin(q_1+q_2) + c\sin(q_1+q_2+q_3) + d\sin(q_1+q_2+q_3+q_4) = 0 \qquad (14)$$
$$q_1+q_2+q_3+q_4 = 0 \mod 2\pi \qquad (15)$$

Usually one joint variable is eliminated via (15), which is then omitted, and (13,14) simplified. One of q_1, q_3 is taken as actuator variable. Hence, to obtain a constraint of the form (10), it is necessary first to eliminate the remaining passive joint variables. This can be done by using $\cos^2\theta_i + \sin^2\theta_i = 1, i = 1, \ldots, 4$, at the cost of obtaining a formula involving the two branches of a quadratic. In particular, one loses differentiability where the discriminant vanishes. Alternatively, the equations can be rewritten as algebraic equations either using tan half-angle as variable, or replacing both $\cos\theta_i$ and $\sin\theta_i$ by new variables, together with corresponding Pythagorean constraint equations. In either case it is, in theory, possible to use Gröbner basis techniques to eliminate variables (Cox et al., 2004).
The situation can, however, become much more complex. For the Gough–Stewart platform in its most general architecture, for a given set of actuator variables $\mathbf{q}$ there may be up to 40 possible configurations $\mathbf{x}$ for the platform (Dietmaier, 1998; Lazard, 1993; Mourrain, 1993).
The omission of passive joint variables has additional drawbacks. Firstly, the choice of actuated variable may be arbitrary, whereas it is sometimes preferable to allow freedom in this choice. Secondly, a manipulator may gain finite or infinitesimal freedom with respect to only its passive joints in certain configurations. This phenomenon was observed by Di Gregorio & Parenti-Castelli (2002) who showed that a 3-UPU manipulator designed for translational motion could undergo rotation in some configurations. Such solutions may not be apparent when solving the restricted constraint equations (10).
The need to include passive joints for the instantaneous kinematics of a general parallel manipulator was recognized by Zlatanov et al. (1994a;b). They distinguished six types of singularity in total. In addition to those arising from inclusion of the passive joints, they also allowed that the instantaneous mobility could exceed the full-cycle mobility μ in (1). The resulting singularity type was denoted *increased instantaneous mobility* (IIM). The phenomenon had already been idenitified for simple closed chains by Hunt (1978), who termed it *uncertainty configuration*, as it corresponds to an intersection of branches of the configuration space, allowing the end-effector motion in at least two different modes. This has also been termed a *configuration space singularity* (Zlatanov et al., 2001) or *topological singularity* (Shvalb et al., 2009). Indeed, even in the relatively simple case of the planar 4R closed chain, if the link lengths $e_1 \leq e_2 \leq e_3 \leq e_4$ (in increasing order) satisfy the non-Grashof condition $e_1 + e_4 = e_2 + e_3$ then the 4-bar can fold flat, with all joints collinear, and the configuration space has a singularity (Gibson & Newstead, 1986).

6.2 A unified approach

Consider a general parallel manipulator with l links and k joints, the ith joint having δ_i degrees of freedom, $i = 1, \ldots, k$. Suppose Q is the product of of the individual joint spaces, so that Q is a manifold of dimension $d = \sum_{i=1}^{k} \delta_i$. The *configuration space* $C \subseteq Q$ is the set of joint variable vectors that satisfy the constraints imposed by the manipulator's structure. As noted in Section 3, the number of independent constraints is $k - l + 1$, the dimension of the cycle space of the graph. Let $q_1, \ldots, q_d$ denote the joint variables associated with twists $X_1, \ldots, X_d$. Each constraint can be written in the form $\phi_j(\mathbf{q}) = I, j = 1, \ldots, k - l + 1$ with ϕ_j as in (12). The map

$$\Phi : Q \to SE(3)^{k-l+1}, \qquad \Phi(\mathbf{q}) = (\phi_1(\mathbf{q}), \ldots, \phi_{k-l+1}(\mathbf{q})) \tag{16}$$

defines the configuration space as $C = \{\mathbf{q} \in Q : \Phi(\mathbf{q}) = (I, \ldots, I)\}$. The Pre-Image Theorem of differential topology asserts that C is a manifold of dimension $\mu = \dim Q - \dim SE(3)^{k-l+1}$ provided that $T_{\mathbf{q}}\Phi$ has maximum rank for all $\mathbf{q} \in C$. Consider, for example, a Gough–Stewart platform having 6 UPS legs connecting the base to the platform, as in Figure 2. Each leg has

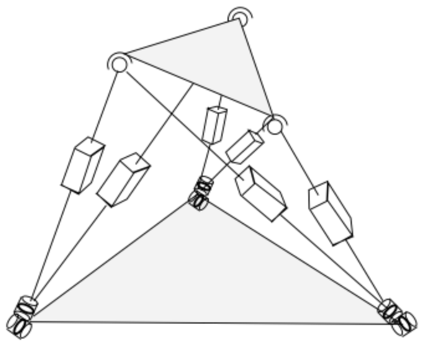


Fig. 2. 6 UPS parallel manipulator

6 joint variables, hence $d = 36$, while the number of links, including the base is $l = 14$ and the number of joints $k = 18$. Given $\dim SE(3) = 6$, it follows that the Jacobian is 30×36. Although this is rather large, we can determine its structure quite simply. Suppose the legs are numbered $r = 1, \dots, 6$ and the twists in each leg denoted X_{rs}, $r, s = 1, \dots, 6$. Five independent closure equations arise from the closed chains consisting of leg 1, leg r, the base and the platform, for $r = 2, \dots, 6$ as follows:

$$\begin{aligned}&\exp(q_{11}X_{11})\exp(q_{12}X_{12})\exp(q_{13}X_{13})\exp(q_{14}X_{14} + q_{15}X_{15} + q_{16}X_{16})\\ &\quad\cdot \exp(-q_{r4}X_{r4} - q_{r5}X_{r5} - q_{r6}X_{r6})\exp(-q_{r3}X_{r3})\exp(-q_{r2}X_{r2})\exp(-q_{r1}X_{r1}) = I \qquad (17)\end{aligned}$$

The Jacobian matrix consists of 5 rows of 36 twists (6-vectors), the jth row having in the column of the ith joint variable either the corresponding twist X_i, if that joint is involved in the loop described by the closure equation for ϕ_j, or else $\mathbf{0}$; for example the first row is

$$\begin{pmatrix} X_{11} & X_{12} & \cdots & X_{16} & X_{21} & X_{22} & \cdots & X_{26} & \mathbf{0} & \cdots & \mathbf{0} \end{pmatrix} \qquad (18)$$

It follows that a necessary and sufficient condition for maximum rank is that the pairwise vector sum of the screw systems of legs span $\mathfrak{se}(3)$. Alternatively, there is a singularity if there exists a common reciprocal wrench. This is a *configuration space singularity*.

Suppose now that the the Jacobian is of full rank, so that the configuration space is a manifold whose dimension is the full-cycle mobility μ in the CGK formula (1), or simply that this is true in a neighbourhood of a configuration. Can the full mobility of the platform be achieved using a given set of μ actuated joints? This can be answered using the Implicit Function Theorem. One requires the corresponding joint variables to give local coordinates for C. The theorem asserts that this is the case if the square matrix, obtained by deleting the μ columns of the Jacobian corresponding to the actuator variables, is non-singular. If that fails to be the case at some configuration $\mathbf{q} \in C$, then there is a direction in the tangent space $T_{\mathbf{q}}C$ that projects to $\mathbf{0}$ in the actuator velocity space but which corresponds to a non-zero velocity of the platform. This is therefore a type II singularity in the nomenclature above, or what might be termed an *actuator singularity*.

Finally, the type I or *kinematic singularities* for a parallel manipulator at a non-singular point of the configuration space are simply the singularities of the forward kinematics from C to $SE(3)$.

Even though this model gives a reasonably complete picture of the singularity types possible for a parallel manipulator, the paradoxical mechanisms such as Bennett, Bricard, Goldberg etc., are not explained; they correspond here to manipulators for which every configuration is a configuration space singularity, as the constraint Jacobian rank is non-maximal. Nevertheless, their configuration spaces are manifolds, but of the “wrong" dimension.

7. Outlook

The analysis of manipulator singularities is a burgeoning area of research. This chapter has touched briefly on some key themes. There are significant open problems, including understanding over-constrained mechanisms, genericity theorems for manipulator architectures, higher-order analysis and topology of the singularity loci, singularities of compliant mechanisms, bifurcation analysis and many more. The practical problems of manipulator design and control remain central goals, but their resolution involves many branches of mathematics including algebraic geometry, differential topology, Lie group theory, geometric algebra, analysis, numerical methods and combinatorics.

8. References

Baker, J. E. (1978). On the investigation of extreme in linkage analysis, using screw system algebra. *Mechanism and Machine Theory*, Vol. 13, No. 3, pp. 333–343

Ben-Horin, P. & Shoham, M. (2009). Application of Grassmann–Cayley algebra to geometrical interpretation of parallel robot singularities. *Int. J. Robotics Research*, Vol. 28, No. 1, pp. 127–141

Bonev, I. & Zlatanov, D., (2001). The mystery of the singular SNU translational parallel robot. *ParalleMIC Reviews*, No. 4, http://www.parallemic.org/Reviews/Review004.html

Brockett, R. (1984). Robotic manipulators and the product of exponentials formula. *Proc. Mathematical Theory of Networks and Systems*, pp. 120–129, Beer-Sheva, June 1983, Springer, Berlin/Heidelberg

Carricato, M. (2005). Fully isotropic four-degrees-of-freedom parallel mechanisms for Schönflies motion. *Int. J. Robotics Research*, Vol. 24, No. 5, pp. 397–414

Cox D., Little J. & O'Shea, D. (2008). *Ideals, Varieties, and Algorithms, 3rd edition*, Springer, Berlin/Heidelberg

Dasgupta, B. & Mruthyunjaya, T. S. (2000). The Stewart platform: a review. *Mechanism and Machine Theory*, Vol. 35, No. 1, pp15–40

Davidson, J. K. & Hunt, K. H. (2004). *Robots and Screw Theory*, Oxford University Press, Oxford

Denavit, J. & Hartenberg, R. S. (1955). A kinematic notation for lower pair mechanisms based on matrices. *J. Applied Mechanics*, Vol. 22, June pp. 215–221

Dietmaier, P. (1998). The Stewart–Gough Platform of general geometry can have 40 real postures. *Advances in Robot Kinematics: Analysis and Control*, pp. 7–16, Kluwer Academic, Dordrecht, The Netherlands

Di Gregorio, R. & Parenti-Castelli, V., (2002). Mobility Analysis of the 3-UPU Parallel Mechanism Assembled for a Pure Translational Motion. *J. Mechanical Design*, Vol. 124, No. 2, pp. 259–264

Donelan, P. S. (2007a). Singularities of robot manipulators, in *Singularity Theory*, pp. 189–217, World Scientific, Hackensack NJ

Donelan, P. S. (2007b). Singularity-theoretic methods in robot kinematics. *Robotica*, Vol. 25, No. 6, pp. 641–659

Donelan, P. S. (2008). Genericity conditions for serial manipulators. *Advances in Robot Kinematics: Analysis and Design*, pp. 185–192, Springer, Berlin/Heidelberg

Duffy, J. & Gilmartin, M. J. (1969). Limit positions of four-link spatial mechanisms—1. Mechanisms having revolute and cylindric pairs. *J. Mechanisms*, Vol. 4, No. 3, pp. 261–272

Featherstone, R. (1983). Position and velocity transformations between robot end-effector coordinates and joint angles. *Int. J. Robotics Research*, Vol. 2, No. 2, pp. 35–45

Fichter, E. F. (1986). A Stewart platform-based manipulator: general theory and practical construction. *Int. J. Robotics Research*, Vol. 5, No. 2, pp. 157–182

Gibson, C. G. (1992). Kinematic singularities—a new mathematical tool. *Advances in Robot Kinematics*, pp. 209–215, Ferrara, Italy, September 1992

Gibson, C. G. & Hunt, K. H. (1990). Geometry of screw systems I & II. *Mechanism and Machine Theory*, Vol. 25, No. 1, pp. 1–27

Gibson, C. G. & Newstead, P. E. (1986). On the geometry of the planar 4–bar mechanism. *Acta Applicandae Mathematicae*, Vol. 7, pp. 113–135

Gogu, G. (2005). Mobility of mechanisms: a critical review. *Mechanism and Machine Theory*, Vol. 40, No. 9. pp. 1068–1097

Gosselin, C. & Angeles, J. (1990). Singularity analysis of closed-loop kinematic chains. *IEEE Trans. Robotics and Automation*, Vol. 6, No. 3, pp. 281–290

Gottlieb, D. H. (1986). Robots and fibre bundles. *Bull. Soc. Mathématique de Belgique*, Vol. 38, pp. 219–223

Gross, J. L. & Yellen, J. (2004). Fundamentals of graph theory, in *Handbook of Graph Theory*, pp. 2-19, CRC Press, Boca Raton, FL

Hao, K. (1998). Dual number method, rank of a screw system and generation of Lie subalgebras. *Mechanism and Machine Theory*, Vol. 33, No. 7. pp. 1063–1084

Hervé, J. M. (1978). Analyse structurelle des mécanismes par groupe des déplacements. *Mechanism and Machine Theory*, Vol.13, pp. 437–450

Hunt, K. H. (1978). *Kinematic Geometry of Mechanisms*, Clarendon Press, Oxford

Hunt, K. H. (1983). Structural kinematics of in-parallel-actuated robot arms. *ASME J. Mechanisms, Transmission and Automation in Design*, Vol. 105, No. 2, pp. 705–712

Karger, A. (1996). Singularity analysis of serial robot–manipulators. *J. Mechanical Design*, Vol. 118, No. 4, pp. 520–525

Karger, A. & Husty , M. (1998). Classification of all self-motions of the original Stewart–Gough platform. *Computer-Aided Design*, Vol. 30, No. 3, pp. 202–215

Lazard, D. (1993). On the representation of rigid-body motions and its application to generalized platform manipulators. *Computational Kinematics*, pp. 175–181, Kluwer Academic, Dordrecht, The Netherlands

Litvin, F. L., Yi, Z., Parenti Castelli, V. & Innocenti, C. (1986). Singularities, configurations, and displacement functions for manipulators. *Int. J. Robotics Research*, Vol. 5, No. 2, pp. 52–65

Ma, O. & Angeles, J. (1992). Architecture singularities of parallel manipulators. *Int. J. Robotics and Automation*, Vol. 7, No. 1, pp. 23–29

Merlet, J. P. (1989). Singular configurations of parallel manipulators and Grassmann geometry. *Int. J. Robotics Research*, Vol. 8, No.5, pp. 45–56

Mourrain, B. (1993). The 40 "generic" positions of a parallel robot. *Proc. 1993 Int. Symposium on Symbolic and Algebraic Computation*, pp. 173–182, Kiev, Ukraine 1993, ACM, New York NY

Müller, A. (2006). A conservative elimination procedure for permanently redundant closure constraints in MBS-models with relative coordinates. *Multibody System Dynamics*, Vol. 16, No. 4, pp. 309–330

Müller, A. (2009). Generic mobility of rigid body mechanisms. *Mechanism and Machine Theory*, Vol. 44, No. 6, pp. 1240–1255

Murray, R. M., Li, Z. & Shastry, S. S. (1994). *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton

Pai, D. K. & Leu, M. C. (1992). Genericity and singularities of robot manipulators. *IEEE Trans. Robotics and Automation*, Vol. 8, No. 5, pp. 545–559

Paul, R. P. & Shimano, B. (1979). Kinematic control equations for simple manipulators. *Proc. 1978 IEEE Conference on Decision and Control*, pp. 1398–1406, San Diego, January 1978, IEEE, Piscataway NY

Pieper, D. L. (1968). The Kinematics of Manipulators Under Computer Control. Ph.D. Thesis, Stanford University

Rico, J. M., Gallardo, J. & Ravani, B. (2003). Lie algebras and the mobility of kinematic chains. *J. Robotic Systems*, Vol. 20, No. 8, pp. 477–499

Selig, J. (1996). *Geometrical Fundamentals of Robotics*, Springer Verlag, New York NY

Shvalb, N., Shoham, M.,Bamberger, H. & Blanc, D. (2009). Topological and kinematic singularities for a class of parallel mechanisms. *Math. Problems in Engineering*, Vol. 2009, doi:10.1155/2009/249349

Stanišić, M. M. & Engleberth, J. W. (1988). A geometric description of manipulator singularities in terms of singular surfaces. *Advances in Robot Kinematics*, pp. 132–141, Ljubljana, 1988

Sugimoto, K., Duffy, J. & Hunt, K. H. (1982). Special configurations of spatial mechanisms and robot arms. *Mechanism and Machine Theory*, Vol. 17, No. 2, pp. 119–132

Tchoń, K. (1991). Differential topology of the inverse kinematic problem for redundant robot manipulators. *Int. J. Robotics Research*, Vol. 10, No. 5, pp. 492–504

Torras, C., Thomas, F. & Alberich-Carraminana, M. (2006). Stratifying the singularity loci of a class of parallel manipulators. *IEEE Trans. Robotics*, Vol. 22, No. 1, pp. 23–32

Waldron, K. J. (1966). The constraint analysis of mechanisms. *J. Mechanisms*, Vol. 1, No. 2, pp. 101–114

Waldron, K. J. (1982). Geometrically based manipulator rate control algorithms. *Mechanism and Machine Theory*, Vol. 17, No. 6, pp. 379–385

Waldron, K. J., Wang, S. L. & Bolin, S. J. (1985). A study of the Jacobian matrix of serial manipulators. *J. Mechanisms, Transmission and Automation in Design*, Vol. 107, No. 2, pp. 230–238

White, N. L.. (1994). Grassmann–Cayley algebra and robotics. *J. Intelligent & Robotic Systems*, Vol. 11, No. 1, pp. 91–107

Whitney, D. E. (1969). Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man–Machine Systems*, Vol. 10, No. 2, pp. 47–53

Wolf, A., Ottaviano, E., Shoham, M. & Ceccarelli, M. (2004). Application of line geometry and linear complex approximation to singularity analysis of the 3-DOF CaPaMan parallel manipulator. *Mechanism and Machine Theory*, Vol. 39, No. 1, pp. 75–95

Zlatanov, D., Bonev, I. & Gosselin, C. (2001). Constraint singularities as configuration space singularities. *ParalleMIC Reviews*, No. 8, http://www.parallemic.org/Reviews/Review008.html

Zlatanov, D., Fenton, R. G. & Benhabib, B., (1994). Singularity analysis of mechanisms and robots via a motion–space model of the instantaneous kinematics. *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 980–985, San Diego CA, May 1994, IEEE, Piscataway NY

Zlatanov, D., Fenton, R. G. & Benhabib, B., (1994). Singularity analysis of mechanisms and robots via a velocity–equation model of the instantaneous kinematics. *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 986–991, San Diego CA, May 1994, IEEE, Piscataway NY

Motion Control of Industrial Robots in Operational Space: Analysis and Experiments with the PA10 Arm

Ricardo Campa[1], César Ramírez[1], Karla Camarillo[2], Víctor Santibáñez[1] and Israel Soto[1]
[1]*Instituto Tecnológico de la Laguna*
[2]*Instituto Tecnológico de Celaya*
Mexico

1. Introduction

Since their appearance in the early 1960's, industrial robots have gained wide popularity as essential components in the construction of automated systems. Reduction of manufacturing costs, increase of productivity, improvement of product quality standards, and the possibility of eliminating harmful or repetitive tasks for human operators represent the main factors that have determined the spread of the robotics technology in the manufacturing industry. Industrial robots are suitable for applications where high precision, repeatability and tracking accuracy are required. These facts give a great importance to the analysis of the actual control schemes of industrial robots (Camarillo et al., 2008).
It is common to specify the robotic tasks in terms of the pose (position and orientation) of the robot's end–effector. In this sense, the *operational space*, introduced by O. Khatib (1987), considers the description of the end–effector's pose by means of a position vector, given in Cartesian coordinates, and an orientation vector, specified in terms of Euler angles. On the other hand, the motion of the robot is produced by control signals applied directly to the joint actuators; further, the robot configuration is usually measured through sensors located in the joints. These facts lead to consider two general control schemes:

- The *joint–space control* requires the use of inverse kinematics to convert the pose desired task to joint coordinates, and then a typical position controller using the joint feedback signals is employed.
- The *operational–space control* uses direct kinematics to transform the measured positions and velocities to operational coordinates, so that the control errors are directly computed in this space.

The analysis of joint–space controllers is simpler than that of operational–space controllers. However, the difficulty of computing the inverse kinematics, especially for robots with many degrees of freedom, is a disadvantage for the implementation of joint–space controllers in real–time applications.

Most of industrial robots are driven by *brushless DC* (BLDC) servoactuators. Typically, this kind of servos include electronic drives which have internal joint velocity and torque controllers (Campa et al., 2008). Thus, the drive's input signal can be either the desired joint velocity or torque, defining the so–called velocity or torque operation modes, respectively. In practice, these drive inner controllers are fixed (they are usually PI controllers tuned with a high proportional and integral gains), and an overall outer loop is necessary to achieve the control goal in operational space.
Aicardi et al. (1995) were the first in analyzing such a hierarchical structure to solve the problem of pose control, but considering an ad-hoc velocity inner loop. Kelly & Moreno (2005) used the same inner controller, together with the so–called *resolved motion rate controller* (RMRC) proposed by Whitney (1969), to solve the problem of operational space control. More recently, Camarillo et al. (2008) invoked some results from (Qu & Dorsey, 1991) to the analysis of a two–loop hierarchical controller using the RMRC as the outer loop and the typical PI velocity controller in the inner loop.
The Mitsubishi PA-10 arm is a general–purpose robotic manipulator with an open architecture, which makes it a suitable choice for both industrial and research applications (Oonishi, 1999). The PA-10 system has a hierarchical structure with several control levels and standard interfaces among them; the lowermost level is at the robot joint BLDC servoactuators, which can be configured either in velocity or torque mode; however, the original setup provided by the manufacturer allows only the operation in velocity mode.
The aim of this work is twofold. Firstly, we review the theoretical results that have been recently proposed in (Camarillo et al., 2008), regarding the stability of the operational space control of industrial robots, assuming inner joint velocity PI controllers, plus the RMRC as the outer loop. Secondly, as a practical contribution of this chapter, we describe the steps required to configure the PA-10 robot arm in torque mode, and include some experimental results obtained from the real–time implementation of two–loop operational space controllers in a PA-10 robot arm which is in our laboratory.
After stating the concerns of the chapter, and recalling the main aspects of modeling and controlling industrial robots (sections 1 and 2) we review the stability analysis of the two–loop operational–space motion–control scheme in Section 3. The description of the PA-10 open–architecture is given in Section 4, while Section 5 explains the hardware and software interfaces that were developed in order to make the PA-10 work in torque–mode. To show the feasibility of the operational–space control scheme in Section 3, we carried out some real–time experiments using the platform described before in a real PA-10 robot; this is explained in Section 6. Finally, concluding remarks are set in Section 7.

2. Modeling and Control of Industrial Robots

2.1 Kinematic and Dynamic Modeling

From a mechanical point of view, industrial robots are considered to have an open kinematic chain, conformed by rigid links and actuated joints. One end of the chain (the *base*) is fixed, while the other (usually called the *end–effector*) is supposed to have a tool or device for executing the task assigned to the robot. As this structure resembles a human arm, industrial robots are also known as robot manipulators.

2.1.1 Kinematics

In a typical industrial robot the number of degrees of freedom of the robot equals the number of joints in the kinematic chain; let n be that number. The configuration of the robot can then be described by a set of n joint coordinates:

$$\boldsymbol{q} = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}^T \in \mathbb{R}^n.$$

On the other hand, let m be the number of degrees of freedom required to specify the task to be executed by the robot. Notice that, in order to ensure the accomplishment of the desired task, then $n \geq m$. If $n > m$ we have a *redundant manipulator*, which has more degrees of freedom that those required to execute the given task.
It is common that the task is specified in terms of the pose (position and orientation) of the robot's end–effector (usually time–varying). In the general 3D–motion case, we require three degrees of freedom for the position and other three for the orientation, thus $m = 6$. Let $\boldsymbol{x}$ be the set of operational coordinates for describing the pose of the end–effector, then

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_6 \end{bmatrix}^T \in \mathbb{R}^6.$$

The direct kinematic model of the robot can be written as

$$\boldsymbol{x} = \boldsymbol{h}(\boldsymbol{q}), \tag{1}$$

where $\boldsymbol{h}(\boldsymbol{q}) : \mathbb{R}^n \to \mathbb{R}^6$ is a function describing the relation between the joint coordinates and the operational coordinates. Furthermore, the differential kinematics equation can be obtained as the time derivative of the direct kinematics equation (1), i.e. (Sciavicco & Siciliano, 2000):

$$\dot{\boldsymbol{x}} = J(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{2}$$

where $J(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ is the so–called analytic Jacobian matrix.
To obtain the inverse differential kinematics, we use the right pseudoinverse of $J(\boldsymbol{q})$, which is given by (Sciavicco & Siciliano, 2000):

$$J(\boldsymbol{q})^{\dagger} = J(\boldsymbol{q})^T \left[J(\boldsymbol{q})J(\boldsymbol{q})^T \right]^{-1}, \tag{3}$$

in such a way that $J(\boldsymbol{q})J(\boldsymbol{q})^{\dagger} = I$, being $I \in \mathbb{R}^{6\times 6}$ the identity matrix. Notice that if $n = 6$ then $J(\boldsymbol{q})^{\dagger}$ reduces to the conventional inverse matrix $J(\boldsymbol{q})^{-1}$.
In the general case, the *inverse differential kinematics* is given by

$$\dot{\boldsymbol{q}} \quad = \quad J(\boldsymbol{q})^{\dagger}\dot{\boldsymbol{x}} + \left[I - J(\boldsymbol{q})^{\dagger}J(\boldsymbol{q}) \right] \dot{\boldsymbol{q}}_0 \quad \in \mathbb{R}^m \tag{4}$$

where the first term is known as the minimal norm solution of (2), and matrix $\left[\boldsymbol{I} - \boldsymbol{J}(\boldsymbol{q})^{\dagger}\boldsymbol{J}(\boldsymbol{q})\right]$ is the orthogonal projector to the null space of $J(\boldsymbol{q})$, $\mathcal{N}(J)$, defined as

$$\mathcal{N}(J) = \{\boldsymbol{w} \in \mathbb{R}^n : J\boldsymbol{w} = \boldsymbol{0}\}.$$

Any vector belonging to $\mathcal{N}(J)$ has no effect in the motion of the robot's end–effector, but in the *self–motion* of its joints (Nakamura, 1991). And as the second term of (4) always belongs to $\mathcal{N}(\mathcal{J})$, independently of the vector $\dot{\boldsymbol{q}}_0$ (that can be easily shown by left multiplying (4) by $J(\boldsymbol{q})$), then $\dot{\boldsymbol{q}}_0$ can be considered as a joint velocity vector chosen in a way that allows performing a secondary task, without affecting the motion of the robot's end–effector.
There exist different approaches for choosing vector $\dot{\boldsymbol{q}}_0$, most of them considering that it is the gradient of a suitable cost function. See (Sciavicco & Siciliano, 2000) for more information on this.

In the analysis forthcoming, it is assumed that the robot's end–effector motion makes $J(\boldsymbol{q})$ to be full rank during the whole task. Also, it is assumed that the following inequalities stand, for some positive constants k_J, k_{Jp}, and k_{dJp}:

$$\|J(\boldsymbol{q})\| \leq k_J \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n, \tag{5}$$

$$\|J(\boldsymbol{q})^{\dagger}\| \leq k_{Jp}, \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n, \tag{6}$$

$$\|\dot{J}(\boldsymbol{q})^{\dagger}\| = \left\| \frac{d}{dt} \left\{ J(\boldsymbol{q})^{\dagger} \right\} \right\| \leq k_{dJp}, \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n. \tag{7}$$

2.1.2 Dynamics

The dynamic model of a serial n–joint robot free of friction is written (Kelly et al., 2005):

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \tau, \tag{8}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector of joint coordinates, $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ is the vector of joint velocities, $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$ is the vector of joint accelerations, $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of applied torque inputs, $M(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite manipulator inertia matrix, $C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ is the matrix of centripetal and Coriolis torques, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is the vector of gravitational torques.
Some useful properties of the dynamic model (8) that will be used in this document, are the following (Kelly et al., 2005):
Property 1: $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ can be chosen so that matrix $\frac{1}{2}\dot{M}(\boldsymbol{q}) - C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew–symmetric, i.e.,

$$\boldsymbol{y}^T \left[\frac{1}{2}\dot{M}(\boldsymbol{q}) - C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \right] \boldsymbol{y} = 0, \quad \forall\, \boldsymbol{y} \in \mathbb{R}^n. \tag{9}$$

Property 2: There exist positive constants k_M, k_C and k_g such that, for all $\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{y} \in \mathbb{R}^n$, we have

$$\|M(\boldsymbol{u})\boldsymbol{y}\| \leq k_M \|\boldsymbol{y}\|, \tag{10}$$

$$\|C(\boldsymbol{u}, \boldsymbol{w})\boldsymbol{y}\| \leq k_C \|\boldsymbol{w}\| \|\boldsymbol{y}\|, \tag{11}$$

$$\|\boldsymbol{g}(\boldsymbol{u})\| \leq k_g. \tag{12}$$

2.2 Hierarchical control

It is well–known that most of industrial robots have internal joint velocity PI controllers, which are usually tuned with very high proportional and integral gains. In practice, these internal controllers are fixed (they belong to each of the joint actuator's servodrives), and an outer loop is necessary to achieve the control goal in operational space. This two–loop hierarchical structure is shown in Figure 1.

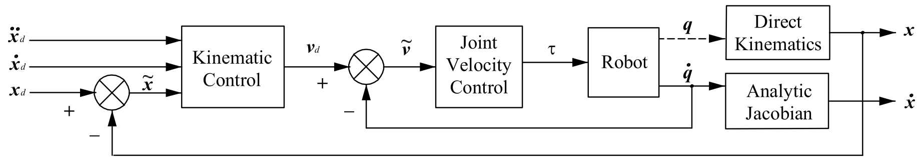


Fig. 1. Typical scheme of a hierarchical structure for operational space control.

2.2.1 Kinematic controller

By kinematic control we refer to any scheme that uses an inverse Jacobian algorithm to resolve the desired joint velocities directly from the pose variables of the desired task. Thus, a kinematic controller is used as the outer loop in the hierarchical controller of Figure 1, which provides the desired joint velocities for the inner velocity controller.
In this chapter we use as kinematic controller the so–called resolved motion rate control (RMRC), which was first proposed by (Whitney, 1969). Using this scheme, the desired joint velocity for the inner loop, can be written as

$$\boldsymbol{v}_d = J(\boldsymbol{q})^{\dagger}\left[\dot{\boldsymbol{x}}_d + K\tilde{\boldsymbol{x}}\right] + \left[I - J(\boldsymbol{q})^{\dagger}J(\boldsymbol{q})\right]\dot{\boldsymbol{q}}_0, \tag{13}$$

where $\boldsymbol{v}_d \in \mathbb{R}^n$ is the desired joint velocity vector, $\dot{\boldsymbol{x}}_d \in \mathbb{R}^6$ is the time derivative of the desired pose vector $\boldsymbol{x}_d$ in operational space, $\tilde{\boldsymbol{x}} = \boldsymbol{x}_d - \boldsymbol{x} \in \mathbb{R}^6$ is the pose error vector in operational space, and $K \in \mathbb{R}^{6\times 6}$ is a symmetric positive definite matrix of control gains. The second term in (13) is added to include the possibility of controlling a secondary task in redundant robots.

2.2.2 Joint velocity PI controller

For the inner loop in Figure 1, we consider the classical joint velocity proportional integral PI controller, commonly used in industrial robots, which can be written as

$$\boldsymbol{\tau} = K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi}, \tag{14}$$

$$\dot{\boldsymbol{\xi}} = \tilde{\boldsymbol{v}}, \tag{15}$$

where

$$\tilde{\boldsymbol{v}} = \boldsymbol{v}_d - \dot{\boldsymbol{q}} \in \mathbb{R}^n, \tag{16}$$

$\boldsymbol{\xi} = \int_0^t \tilde{\boldsymbol{v}}(\sigma)\; d\sigma$, and $K_p, K_i \in \mathbb{R}^{n\times n}$ are diagonal positive definite matrices.
Without loss of generality, let us take

$$K_p = \left[k_p + \gamma\right] I, \tag{17}$$

$$K_i = \alpha K_p, \tag{18}$$

where k_p, γ and α are strictly positive constants. Notice that in such case (18) can be written as

$$K_i = \left[k_i + \alpha\gamma\right] I, \tag{19}$$

with $k_i = \alpha k_p$, or

$$\alpha = \frac{k_i}{k_p}. \tag{20}$$

3. Stability Analysis

The stability analysis presented here is taken from (Camarillo et al., 2008). More detail can be found in that reference.
First, some remarks on notation: We use $\lambda_{\min}\{A\}$ and $\lambda_{\max}\{A\}$ to represent, respectively, the smallest and the largest eigenvalues of a symmetric positive definite matrix $A(\boldsymbol{y})$, for any $\boldsymbol{y} \in \mathbb{R}^n$. Given $\boldsymbol{y} \in \mathbb{R}^n$ and a matrix $A(\boldsymbol{y})$ the Euclidean norm of $\boldsymbol{y}$ is defined as $\|\boldsymbol{y}\| = \sqrt{\boldsymbol{y}^T\boldsymbol{y}}$, and the induced norm of $A(\boldsymbol{y})$ is defined as $\|A(\boldsymbol{y})\| = \sqrt{\lambda_{\max}\{A^TA\}}$.

Note in Figure 1 that the closed–loop system is formed by the RMRC controller (13), the joint velocity PI controller (14)–(15), and the robot model, given by (8), (1) and (2).
Left multiplying (13) by $J(\boldsymbol{q})$, using (2), and (16), we get

$$\dot{\tilde{\boldsymbol{x}}} = -K\tilde{\boldsymbol{x}} + J(\boldsymbol{q})\tilde{\boldsymbol{v}}. \tag{21}$$

Substituting (14) into the robot dynamics (8) we have

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi}. \tag{22}$$

Now, taking into account (16) and its derivative, we get

$$M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\tilde{\boldsymbol{v}} + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} = M(\boldsymbol{q})\dot{\boldsymbol{v}}_d + C(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{v}_d + \boldsymbol{g}(\boldsymbol{q}).$$

We can add the terms $\alpha M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \alpha C(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{\xi}$ to both sides of the last equation, to obtain

$$\begin{aligned} M(\boldsymbol{q})\left[\dot{\tilde{\boldsymbol{v}}} + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} &= \\ M(\boldsymbol{q})\left[\dot{\boldsymbol{v}}_d + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\boldsymbol{v}_d + \alpha\boldsymbol{\xi}\right] + \boldsymbol{g}(\boldsymbol{q}).& \end{aligned} \tag{23}$$

Finally, from (21), (15) and (23), we get the closed–loop system in terms of the state vector $\boldsymbol{z} = \begin{bmatrix}\tilde{\boldsymbol{x}}^T & \boldsymbol{\xi}^T & \tilde{\boldsymbol{v}}^T\end{bmatrix}^T \in \mathbb{R}^{6+2n}$:

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} - \boldsymbol{d}(t,\boldsymbol{z})\right]\end{bmatrix}, \tag{24}$$

where the term

$$\boldsymbol{d}(t,\boldsymbol{z}) = M(\boldsymbol{q})\left[\dot{\boldsymbol{v}}_d + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\boldsymbol{v}_d + \alpha\boldsymbol{\xi}\right] + \boldsymbol{g}(\boldsymbol{q}) \tag{25}$$

is considered as a disturbance and, as it is shown in (Camarillo et al., 2008), is upper bounded for all $t \geq t_0$ by a quadratic polynomial:

$$\|\boldsymbol{d}(t,\boldsymbol{z})\| \leq \varsigma_2\|\boldsymbol{z}(t)\|^2 + \varsigma_1\|\boldsymbol{z}(t)\| + \varsigma_0, \tag{26}$$

where $\varsigma_0, \varsigma_1, \varsigma_2$ are positive constants, assuming that $\|\boldsymbol{v}_d\|$ and $\|\dot{\boldsymbol{v}}_d\|$ are also bounded.
Substituting the definitions (17) and (19) in (24), we get

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + k_p\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + \gamma\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] - \boldsymbol{d}(t,\boldsymbol{z})\right]\end{bmatrix}. \tag{27}$$

In accordance with the theory of perturbed systems (Khalil, 2005), we define

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + k_p\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right]\right]\end{bmatrix} \tag{28}$$

as the nominal system from the closed–loop system (27). As we can see, (28) is free of disturbances, and the origin of the state space $\boldsymbol{z} = \boldsymbol{0}$ is an equilibrium of (28).
In order to show the stability of the overall closed–loop system (27), we use the methodology proposed in (Khalil, 2005), that is:

(a) First, to prove the asymptotic stability of the nominal system (28).

(b) Then, to prove the uniform ultimate boundedness of the overall closed–loop system (27).

3.1 Stability of the nominal system

To prove that the origin of the nominal system (28) is exponentially stable, we propose the following Lyapunov function candidate, inspired from (Qu & Dorsey, 1991):

$$V(t,z) = \frac{1}{2}\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}} + \alpha\boldsymbol{\xi}^T\left[k_p I + \frac{1}{2}\alpha M(\boldsymbol{q})\right]\boldsymbol{\xi} + \alpha\boldsymbol{\xi}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \frac{1}{2}\tilde{\boldsymbol{v}}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}}. \tag{29}$$

It is easy to show that

$$\begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix}^T P_1 \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix} \le V(t,z) \le \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix}^T P_2 \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix},$$

with

$$\begin{aligned} P_1 &= \begin{bmatrix}\frac{1}{2} & 0 & 0\\ 0 & \alpha k_p + \frac{1}{2}\alpha^2\lambda_{\min}\{M\} & -\frac{1}{2}\alpha\lambda_{\max}\{M\}\\ 0 & -\frac{1}{2}\alpha\lambda_{\max}\{M\} & \frac{1}{2}\lambda_{\min}\{M\}\end{bmatrix},\\ P_2 &= \begin{bmatrix}\frac{1}{2} & 0 & 0\\ 0 & \alpha k_p + \frac{1}{2}\alpha^2\lambda_{\max}\{M\} & \frac{1}{2}\alpha\lambda_{\max}\{M\}\\ 0 & \frac{1}{2}\alpha\lambda_{\max}\{M\} & \frac{1}{2}\lambda_{\max}\{M\}\end{bmatrix}, \end{aligned}$$

so that the Lyapunov function candidate (29) satisfies

$$\lambda_1\|z\|^2 \le V(t,z) \le \lambda_2\|z\|^2, \tag{30}$$

for some positive constants λ_1, λ_2, given by

$$\lambda_1 = \lambda_{\min}\{P_1\},\ \lambda_2 = \lambda_{\max}\{P_2\}. \tag{31}$$

Notice that $V(t,z)$ is a globally positive definite function if

$$\alpha \le \frac{2k_p\lambda_{\min}\{M\}}{\lambda^2_{\max}\{M\} - \lambda^2_{\min}\{M\}}, \tag{32}$$

or, using (20), if

$$k_i \le \frac{2k_p^2\lambda_{\min}\{M\}}{\lambda^2_{\max}\{M\} - \lambda^2_{\min}\{M\}}. \tag{33}$$

On the other hand, the time derivative of the Lyapunov function candidate (29) is given by

$$\begin{aligned}\dot{V}(t,z) &= \tilde{\boldsymbol{x}}^T\dot{\tilde{\boldsymbol{x}}} + \alpha k_p\boldsymbol{\xi}^T\dot{\boldsymbol{\xi}} + \alpha^2\boldsymbol{\xi}^T M(\boldsymbol{q})\dot{\boldsymbol{\xi}} + \frac{1}{2}\alpha^2\boldsymbol{\xi}^T\dot{M}(\boldsymbol{q})\boldsymbol{\xi} + \alpha\dot{\boldsymbol{\xi}}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}^T\dot{M}(\boldsymbol{q})\tilde{\boldsymbol{v}}\\ &\quad + \alpha\boldsymbol{\xi}^T M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + \tilde{\boldsymbol{v}}^T M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + \frac{1}{2}\tilde{\boldsymbol{v}}^T\dot{M}(\boldsymbol{q})\tilde{\boldsymbol{v}}, \end{aligned} \tag{34}$$

which along the solutions of the nominal system (28) results in

$$\dot{V}(t,z) = -\tilde{\boldsymbol{x}}^T K\tilde{\boldsymbol{x}} - \alpha k_i\boldsymbol{\xi}^T\boldsymbol{\xi} - k_p\tilde{\boldsymbol{v}}^T\tilde{\boldsymbol{v}} + \tilde{\boldsymbol{x}}^T J(\boldsymbol{q})\tilde{\boldsymbol{v}},$$

where we have used the definition of α in (20), and the skew–symmetry property (9) to eliminate some terms.

Then, under the assumption that the Jacobian matrix $J(\boldsymbol{q})$ is bounded, see (5), it is easy to show that

$$\dot{V}(t,\boldsymbol{z}) \leq - \begin{bmatrix} \|\tilde{\boldsymbol{x}}\| \\ \|\boldsymbol{\xi}\| \\ \|\tilde{\boldsymbol{v}}\| \end{bmatrix}^T Q \begin{bmatrix} \|\tilde{\boldsymbol{x}}\| \\ \|\boldsymbol{\xi}\| \\ \|\tilde{\boldsymbol{v}}\| \end{bmatrix}, \tag{35}$$

with

$$Q = \begin{bmatrix} \lambda_{\min}\{K\} & 0 & -\frac{1}{2}k_J \\ 0 & \alpha k_i & 0 \\ -\frac{1}{2}k_J & 0 & k_p \end{bmatrix}. \tag{36}$$

So that (35) can be written as

$$\dot{V}(t,\boldsymbol{z}) \leq -\lambda_3\|\boldsymbol{z}\|^2, \tag{37}$$

where

$$\lambda_3 = \lambda_{\min}\{Q\}. \tag{38}$$

Notice that we can choose

$$\lambda_{\min}\{K\} > \frac{k_J^2}{4k_p}, \tag{39}$$

so that Q is positive definite and $\dot{V}(t,\boldsymbol{z})$ is globally negative definite.
Thus, provided that (33) and (39) are satisfied, and according to Lyapunov's direct method (Khalil, 2005), we conclude the exponential stability of the origin of the nominal system (28). This implies that the origin of the state space is attractive, i.e.

$$\lim_{t\to\infty} \boldsymbol{z}(t) = \lim_{t\to\infty} \begin{bmatrix} \tilde{\boldsymbol{x}}(t) \\ \boldsymbol{\xi}(t) \\ \tilde{\boldsymbol{v}}(t) \end{bmatrix} = \boldsymbol{0}.$$

3.2 Stability of the overall closed–loop system

We have shown in the previous subsection that the equilibrium of the nominal system (28) is exponentially stable by choosing the feedback gains k_i, and $\lambda_{\min}\{K\}$ according to (33) and (39), respectively. Now, we are able to show that, by properly choosing the feedback gain γ of the joint velocity PI controller, the solutions $\boldsymbol{z}(t)$ of the overall closed–loop system (27) are uniformly ultimately bounded.
First, we recall the following fact (Qu & Dorsey, 1991), that will be useful for our purposes.
Fact 1: *Let*

$$g(\|\boldsymbol{z}\|) = -\alpha_1\|\boldsymbol{z}\|^2 + \alpha_2\|\boldsymbol{z}\| + \alpha_3, \tag{40}$$

be a quadratic polynomial with $\alpha_1 > 0$, *and* $\alpha_2, \alpha_3 \geq 0$. *Given*

$$\eta = \frac{\alpha_2 + \sqrt{\alpha_2^2 + 4\alpha_1\alpha_3}}{2\alpha_1}, \tag{41}$$

then

- $g(\eta) = 0$,

- *$g(\|z\|) < 0$, and strictly decreasing, for all $\|z\| > \eta$.*

$\diamondsuit$

Now, we recall the following theorem in (Camarillo et al., 2008), which is a variation of Theorem 4.18 in (Khalil, 2005).
Theorem 1: *Let*

$$\dot{z} = f(t, z), \tag{42}$$

be a system that describes a first–order vector differential equation, where $f : [0, \infty) \times D \to I\!R^m$ and $D \subset I\!R^m$, is a domain that contains the origin. Let $V(t, z)$, with $V : [0, \infty) \times D \to I\!R$, be a Lyapunov–like function that satisfies

$$\lambda_1\|z\|^2 \leq V(t,z) \leq \lambda_2\|z\|^2, \tag{43}$$
$$\dot{V}(t,z) \leq g(\|z\|) < 0, \quad \forall\, \|z\| > \eta, \tag{44}$$

for all $t \geq t_0$, and for all $z \in D$, with $\lambda_1, \lambda_2 > 0$, $g(\|z\|)$ and η defined in Fact 1 by (40) *and* (41)*, respectively. Then, there exists $T \geq 0$ (dependent on $z(t_0)$ and η) such that the solution of* (42)*, satisfies the following properties, for any initial state $z_0 = z(t_0)$:*

A. *Uniform boundedness*

$$\|z(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}} \max\{\|z_0\|, \eta\}, \quad \forall\, t \geq t_0. \tag{45}$$

B. *Uniform ultimate boundedness*

$$\|z(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}}\eta', \quad \forall\, t \geq t_0 + T, \tag{46}$$

for all $\eta' > \eta$.

$\bigtriangledown$

The proof of Theorem 1 can be found in (Camarillo et al., 2008). Theorem 1 is the base for the following result:
Proposition 1: *Consider the overall closed–loop system* (27) *with the outer–loop kinematic controller*

$$v_d = J(q)^{\dagger}[\dot{x}_d + K\tilde{x}],$$

and the inner–loop velocity controller

$$\tau = K_p\tilde{v} + K_i\xi = [k_p + \gamma]\tilde{v} + [k_i + \gamma\alpha]\xi. \tag{47}$$

Suppose the control gains are chosen so that k_p, k_i and K satisfy (33) *and* (39)*. Moreover, γ is a positive scalar such that*

$$\gamma \geq \frac{1}{\epsilon}\left[\sigma^2\varsigma_2 + \sigma\varsigma_1 + \varsigma_0\right], \tag{48}$$

where ς_0, ς_1 and ς_2 are the coefficients of the upper bound of $\|d(t,z)\|$, given in (26)*, ϵ is a positive constant such that*

$$0 < \epsilon < \frac{\lambda_3}{\varsigma_2}, \tag{49}$$

and

$$\sigma = \sqrt{\frac{\lambda_2}{\lambda_1}} \max\{\|z(t_0)\|, \bar{\eta}\}, \tag{50}$$

with

$$\bar{\eta} = \frac{\epsilon\varsigma_1 + \sqrt{\epsilon^2\varsigma_1^2 + 4\epsilon\varsigma_0\left[\lambda_3 - \epsilon\varsigma_2\right]}}{2\left[\lambda_3 - \epsilon\varsigma_2\right]}, \tag{51}$$

where λ_1, λ_2, are defined in (31), *and λ_3 is in* (38).
Then, the system is stabilizable in the sense that exists $T \geq 0$ such that the solution $z(t)$, with initial state $z_0 = z(t_0)$ satisfies

$$\|z(t)\| \leq \sigma, \quad \forall\, t \geq t_0 \text{ (uniform boundedness)}, \tag{52}$$

$$\|z(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}}\bar{\eta}', \quad \forall\, t \geq t_0 + T \text{ (uniform ultimate boundedness)}. \tag{53}$$

for all $\bar{\eta}' > \bar{\eta}$, where (52) *and* (53) *regard to the uniform and uniform ultimate boundedness, respectively.*

$\bigtriangledown$

The proof of Proposition 1 is also in (Camarillo et al., 2008); it employs (25) and Theorem 1. The following remarks can be set from this result:
Remark 1. *Notice that if $k_p \to \infty$ in the PI controller then, from* (20), *$\alpha \to 0$ and, considering* (36), *and* (49), *we get $\lambda_3 = \lambda_{\min}\{Q\} \to 0$ and $\epsilon \to 0$. Moreover, from* (51), *$\epsilon \to 0$ implies that $\bar{\eta} \to 0$, the uniform bound σ becomes $\sigma = \sqrt{\frac{\lambda_2}{\lambda_1}}\|z(t_0)\|$, and the system tends to be exponentially stable.*
Remark 2. *It is noteworthy that the joint velocity PI controller*

$$\begin{aligned} \boldsymbol{\tau} &= [k_p + \gamma]\tilde{\boldsymbol{v}} + [k_i + \gamma\alpha]\boldsymbol{\xi} \\ \dot{\boldsymbol{\xi}} &= \tilde{\boldsymbol{v}} \end{aligned}$$

can be also written as

$$\begin{aligned} \boldsymbol{\tau} &= [k_p + \gamma]\tilde{\boldsymbol{v}} + \alpha\boldsymbol{y} \\ \frac{1}{k_p + \gamma}\dot{\boldsymbol{y}} &= \tilde{\boldsymbol{v}}. \end{aligned}$$

Thus, if $\gamma \to \infty$, then, $\tilde{\boldsymbol{v}} \to 0$, and $\dot{\boldsymbol{q}} \to \boldsymbol{v}_d$. Also, considering (17) *and* (19), *$\gamma \to \infty$ implies $K_p, K_i \to \infty$.*
From the previous remarks we can conclude that the common assumption of having high–value gains in the inner velocity PI controller of industrial robots leads to small tracking errors during the execution of a motion control task.

4. The PA10 Robot System

The Mitsubishi PA10 arm is an industrial robot manipulator which completely changes the vision of conventional industrial robots. Its name is an acronym of *Portable General-Purpose Intelligent Arm*. There exist two versions (Higuchi et al., 2003): the PA10-6C and the PA10-7C, where the suffix digit indicates the number of degrees of freedom of the arm. This work focuses on the study of the PA10-7CE model, which is the enhanced version of the PA10-7C. The PA10 arm is an open architecture robot; it means that it possesses (Oonishi, 1999):

- A hierarchical structure with several control levels.
- Communication between levels, via standard interfaces.
- An open general–purpose interface in the higher level.

This scheme allows the user to focus on the programming of the tasks at the PA10 system's higher level, without regarding on the operation of the lower levels. The programming can be performed using a high–level language, such as Visual BASIC or Visual C++, from a PC with Windows operating system. The PA10 robot is currently the open–architecture robot more employed for research (see, e.g., Kennedy & Desai (2003), Jamisola et al. (2004), Pholsiri (2004)).

4.1 Basic structure of the PA10 robot

The PA10 system is composed of four sections or levels, which conform a hierarchical structure (Mitsubishi, 2002a):

Level 4: Operation control section (OCS); formed by the PC and the teaching pendant.

Level 3: Motion control section (MCS); formed by the motion control and optical boards.

Level 2: Servo drives.

Level 1: Robot manipulator.

Figure 2 shows the relation existing among each of the mentioned components. The following subsections give a more detailed description of them.

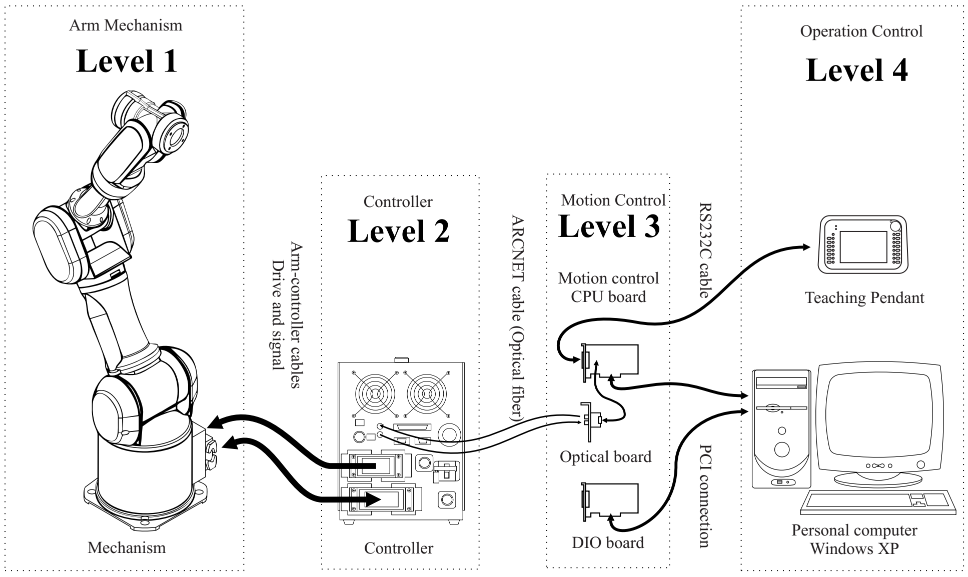


Fig. 2. Components of the PA10 system

4.1.1 Operation control section

This section is the higher level in the PA10 system's hierarchy; it receives its name because it is in this section in which the operator programs the tasks to be performed by the robot. This can be done by a control computer or a teaching pendant. The control computer is a PC with an MS-Windows operating system. For the programming of the robot tasks, the operator can employ the programs provided by the manufacturer, or he can develop his own applications using Visual C++ or Visual BASIC and some API functions from the so–called *PA library*.
It is through this functions that the user can select the control scheme he wants for the robot, being the most common the joint velocity control, the joint position control and the pose control. More information on how these schemes can be implemented, as well as an experimental comparison among them, can be found in (Camarillo et al., 2006). It is worth mentioning, however, that all of these schemes work with the servo–drives configured in velocity mode.
Independently on the way the tasks to be performed are specified in the operation control section, the information which is passed to the next level is always the same in this basic configuration. This allows to keep a hierarchical structure, but at the same time, it sets limitations to the versatility of the whole system.
For the experimental evaluation shown in Section 6, we decided to use the control computer and a real–time application, developed in Visual C++. See section 5.3 for a description of this.

4.1.2 Motion control section

It is in this section where the joint velocity commands for the servo–drives (Level 2) are computed, using the information programmed by the user in Level 4. To do so, the original PA10 system uses the MHI-D7281 motion control board, from Mitsubishi Heavy Industries, which is plugged in the PCI bus of the control computer. It is in this board where, among other things, the joint position and pose kinematic controllers, together with the interpolation algorithms for generating smooth trajectories, are implemented.
Communication among the control computer and the driver cabinet is done via optical fibers, using the ARCNET protocol. The motion control board counts on a converter to encode signals to the ARCNET format. More information on this can be found in Section 5.1.
A board model MHI-D7210 is dedicated to convert the electrical signals from the motion control board into optical signals. This board is called the optical board in Figure 2. For more information on the motion control board, and the optical board, see (Mitsubishi, 2002a).

4.1.3 Servo–drives

The seven drives of the PA10 servomotors are contained in a cabinet (driver) which connects to the optical–interface board via optical fiber. In addition, two cables connect the cabinet to the robot; one contains the power signals for the servomotors, while the other transmits the feedback signals from the joint position sensors.
PA10's servomotors are brushless DC type, and they are coupled to the links via harmonic drives. The servo–drives can be configured in torque or velocity modes. However, if the motion control board in the MCS is used, then it is only possible to configure the drives in velocity mode. In order to use the PA10 robot in torque mode, it is necessary to disable the MHI-D7281 board, and replace it by one which allows the operator to have direct access to the drive signals from the PC. This is explained in Section 5.2.
In velocity mode, the drives include an inner velocity loop which, according to the documentation provided by Mitsubishi (Mitsubishi, 2002b) is a digital PI controller with a 400 μs sampling period. Drives also include a current (torque) PI controller, with a 100 μs sampling

period, which, as indicated in (Campa et al., 2008) has an insignificant effect in the servomotor's dynamics at low velocities.

4.1.4 Robot manipulator

The PA10 robot is a 7–dof redundant manipulator with revolute joints. Figure 3 shows a diagram of the PA10 arm, indicating the positive rotation direction and the respective names of each of the joints.

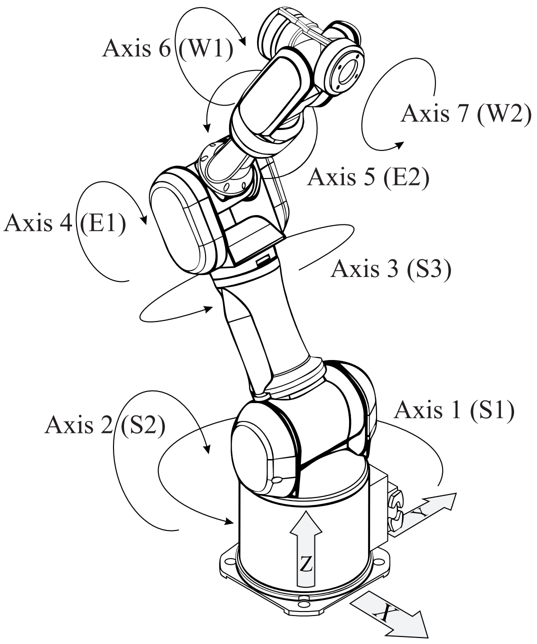


Fig. 3. Mitsubishi PA10-7CE robot

4.2 Modeling of the PA10 arm

As the PA10-7CE is a robot with seven degrees of freedom, then $n = 7$, and the direct kinematics function $\boldsymbol{f} : \mathbb{R}^7 \to \mathbb{R}^6$ has the form $\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q})$.
In this work, we consider the vector of operational coordinates $\boldsymbol{x}$ is formed by three Cartesian coordinates (x,y,z) and three Euler angles (α,β,γ), given by the ZYX convention (Craig, 2004). Using the traditional methodology, based on the Denavit-Hartenberg convention (Denavit & Hartenberg, 1955), we obtain for the PA10 arm the following expressions:

$$
\begin{aligned}
x &= [-([(\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3)\mathrm{c}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{s}_4]\mathrm{c}_5 - [-(-\mathrm{c}_1\mathrm{c}_2\mathrm{s}_3 - \mathrm{s}_1\mathrm{c}_3)]\mathrm{s}_5)\mathrm{s}_6 \\
&\quad +(-[\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3]\mathrm{s}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{c}_4)\mathrm{c}_6]\, d_7 \\
&\quad + [-(\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3)\mathrm{s}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{c}_4]\, d_5 - \mathrm{c}_1\mathrm{s}_2 d_3, \\
y &= [-([(\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3)\mathrm{c}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{s}_4]\mathrm{c}_5 - (-(-\mathrm{s}_1\mathrm{c}_2\mathrm{s}_3 + \mathrm{c}_1\mathrm{c}_3))\mathrm{s}_5)\mathrm{s}_6 \\
&\quad +(-[\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3]\mathrm{s}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{c}_4)\mathrm{c}_6]\, d_7 \\
&\quad + [-(\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3)\mathrm{s}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{c}_4]\, d_5 - \mathrm{s}_1\mathrm{s}_2 d_3, \\
z &= [-([\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 + \mathrm{c}_2\mathrm{s}_4]\mathrm{c}_5 - \mathrm{s}_2\mathrm{s}_3\mathrm{s}_5)\mathrm{s}_6 + (-\mathrm{s}_2\mathrm{c}_3\mathrm{s}_4 + \mathrm{c}_2\mathrm{c}_4)\mathrm{c}_6]\, d_7 \\
&\quad + [-\mathrm{s}_2\mathrm{c}_3\mathrm{s}_4 + \mathrm{c}_2\mathrm{c}_4]\, d_5 + \mathrm{c}_2 d_3, \\
\alpha &= \operatorname{atan2}(b, a), \\
\beta &= \operatorname{atan2}\left(-c, \frac{a}{\cos(\alpha)}\right), \\
\gamma &= \operatorname{atan2}(d, e).
\end{aligned}
$$

where the terms c_i y s_i correspond to $\cos(q_i)$ y $\operatorname{sen}(q_i)$, respectively, and d_i is the length of the i-th link, with $i = 1, 2, \ldots, 7$. The required values for this robot are $d_1 = 0.317$ m, $d_3 = 0.450$ m, $d_5 = 0.480$ m, and $d_7 = 0.070$ m. For computing the Euler angles it is employed the inverse tangent function with two arguments (atan2) and the following expressions:

$$
\begin{aligned}
a &= [([(\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3)\mathrm{c}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{s}_4]\mathrm{c}_5 + [-\mathrm{c}_1\mathrm{c}_2\mathrm{s}_3 - \mathrm{s}_1\mathrm{c}_3]\mathrm{s}_5)\mathrm{c}_6 \\
&\quad -(-[-(\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3)\mathrm{s}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{c}_4])\mathrm{s}_6]\, \mathrm{c}_7 \\
&\quad + [-([\mathrm{c}_1\mathrm{c}_2\mathrm{c}_3 - \mathrm{s}_1\mathrm{s}_3]\mathrm{c}_4 - \mathrm{c}_1\mathrm{s}_2\mathrm{s}_4)\mathrm{s}_5 + (-\mathrm{c}_1\mathrm{c}_2\mathrm{s}_3 - \mathrm{s}_1\mathrm{c}_3)\mathrm{c}_5]\, \mathrm{s}_7 \\
b &= [([(\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3)\mathrm{c}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{s}_4]\mathrm{c}_5 + [-\mathrm{s}_1\mathrm{c}_2\mathrm{s}_3 + \mathrm{c}_1\mathrm{c}_3]\mathrm{s}_5)\mathrm{c}_6 \\
&\quad -(-(-(\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3)\mathrm{s}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{c}_4))\mathrm{s}_6]\, \mathrm{c}_7 \\
&\quad + [-([\mathrm{s}_1\mathrm{c}_2\mathrm{c}_3 + \mathrm{c}_1\mathrm{s}_3]\mathrm{c}_4 - \mathrm{s}_1\mathrm{s}_2\mathrm{s}_4)\mathrm{s}_5 + (-\mathrm{s}_1\mathrm{c}_2\mathrm{s}_3 + \mathrm{c}_1\mathrm{c}_3)\mathrm{c}_5]\, \mathrm{s}_7 \\
c &= [([-\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 - \mathrm{c}_2\mathrm{s}_4]\mathrm{c}_5 + \mathrm{s}_2\mathrm{s}_3\mathrm{s}_5)\mathrm{c}_6 - (-[\mathrm{s}_2\mathrm{c}_3\mathrm{s}_4 - \mathrm{c}_2\mathrm{c}_4])\mathrm{s}_6]\, \mathrm{c}_7 \\
&\quad +[-(-\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 - \mathrm{c}_2\mathrm{s}_4)\mathrm{s}_5 + \mathrm{s}_2\mathrm{s}_3\mathrm{c}_5]\mathrm{s}_7 \\
d &= -[([-\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 - \mathrm{c}_2\mathrm{s}_4]\mathrm{c}_5 + \mathrm{s}_2\mathrm{s}_3\mathrm{s}_5)\mathrm{c}_6 - (-[\mathrm{s}_2\mathrm{c}_3\mathrm{s}_4 - \mathrm{c}_2\mathrm{c}_4])\mathrm{s}_6]\, \mathrm{s}_7 \\
&\quad +[-(-\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 - \mathrm{c}_2\mathrm{s}_4)\mathrm{s}_5 + \mathrm{s}_2\mathrm{s}_3\mathrm{c}_5]\mathrm{c}_7 \\
e &= -[-([-\mathrm{s}_2\mathrm{c}_3\mathrm{c}_4 - \mathrm{c}_2\mathrm{s}_4]\mathrm{c}_5 + \mathrm{s}_2\mathrm{s}_3\mathrm{s}_5)\mathrm{s}_6 - (-[\mathrm{s}_2\mathrm{c}_3\mathrm{s}_4 - \mathrm{c}_2\mathrm{c}_4])\mathrm{c}_6]
\end{aligned}
$$

By reasons of space, we do not include here the analytic Jacobian of the PA10 robot; however, it is possible to obtain it from the direct kinematics, since

$$J(\boldsymbol{q}) = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}}.$$

Other way of computing the analytic Jacobian is via the geometric Jacobian, $J_G(\boldsymbol{q})$, which relates the joint velocities with the linear and angular velocities, $\boldsymbol{v}$ and $\boldsymbol{\omega}$, respectively, according to (Sciavicco & Siciliano, 2000)

$$\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} = J_G(\boldsymbol{q})\dot{\boldsymbol{q}}.$$

After computing the geometric Jacobian, using standard algorithms (Sciavicco & Siciliano, 2000), we can use the following expression:

$$J(\boldsymbol{q}) = \begin{bmatrix} I & 0 \\ 0 & T(\alpha,\beta,\gamma) \end{bmatrix} J_G(\boldsymbol{q})$$

where $T(\alpha,\beta,\gamma)$ is a representation transformation matrix which depends on the Euler angle convention employed. In the case of the ZYX convention, we have that

$$T(\alpha,\beta,\gamma) = \begin{bmatrix} 0 & -\sin(\alpha) & \cos(\alpha)\cos(\beta) \\ 0 & \cos(\alpha) & \sin(\alpha)\cos(\beta) \\ 1 & 0 & -\sin(\beta) \end{bmatrix}.$$

Regarding the dynamic model of the PA10-7CE robot, it was not required for the experiments described in Section 6, although it can be found in (Ramírez, 2008).

5. Real–time Control Platform

In this section we first describe the general characteristics of the ARCNET protocol, and give the required specifications for its use in the PA10 system. After that, we describe the modifications to the hardware, and the software application we have developed for controlling the PA10 robot in real–time.

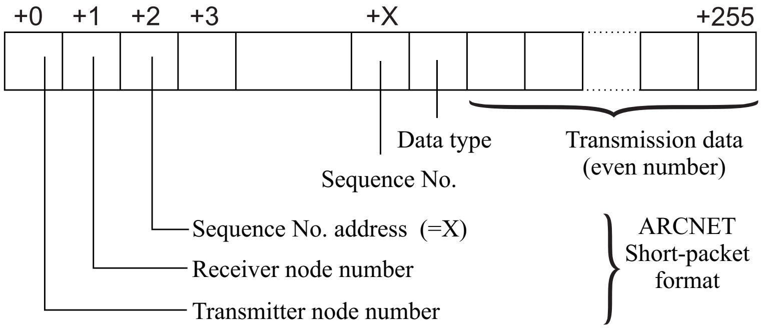
5.1 The ARCNET protocol

ARCNET (acronym of Attached Resource Computer Network) is a protocol for data transmission in a local area network (LAN). It was developed by Datapoint Corporation and it became very popular in the 80's. In the current days, even though it has been relegated by Ethernet in commercial and academic networks, it is still used in the industrial field, since its deterministic behavior and robustness are appropriate for the control of devices.
ARCNET handles data packets of variable length, from 0 to 507 bytes, and operates at velocities from 2.5 Mbps to 10 Mbps, so that it is possible to have a fast response for short messages. Another advantage is that it allows different types of transmission media, being the twisted cable, the coaxial cable and the optical fiber the most common. Moreover, the data interchange is implemented by hardware, in an ARCNET controller IC, so that the basic functions such as error checking, flux control and network configuration are performed directly in the IC.
Each device connected to an ARCNET network is known as a node, and it must contain a controller IC as well as an adapter for the type of cable employed. A MAC address identifier is assigned to each node. An ARCNET network can have up to 255 nodes with a unique identifier (or node number) assigned to each of them.
The ARCNET protocol employs an scheme known as token–passing, to manage the data stream among the active nodes in the network. When a node possesses the token, it can transmit data through the net or pass the token to the neighboring node, which, even though it can be located physically in any place, it has a consecutive node number. The header of each sent packet includes the address of the receiver node, in a way that all the nodes, with the exception of the pretended receiver, overlook those data. Once a message has been sent, the token passes to the next node, in a consecutive way, up to returning to the original node.
If the message is very long (more than 507 bytes) it is split in several packets. The packets can be short or long depending of the selected mode. Short packets are from 0 to 252 bytes, and

the long ones from 256 to 507 bytes. It is not possible to send packets of 253, 254 or 255 bytes; in those cases null data are added and they are sent either as two short packets or one long.
As each node can only send data when it has the token, no collisions occur when using the ARCNET protocol. Moreover, thanks to the token–passing scheme, the time a node lasts in sending a message can be computed; this is an important feature in industrial networks, where it is required that the control events occur in a precise moment.

5.1.1 Specifications for the PA10 system

Communication between the motion control board and the PA10's motor drives is carried out via an optical fiber, using the ARCNET protocol. In its original configuration a transmission rate of 10 Mbps is employed; the node number of the ARCNET controller IC in the driver cabinet is 254 (hexadecimal FE), while the one in the motion control board has number 255 (hexadecimal FF). As explained in (Mitsubishi, 2002b), the data packets are short, and they follow the format which is shown in Figure 4.


+0
+1
+2
+3
+X
+255
Data type
Transmission data (even number)
Sequence No.
Sequence No. address (=X)
Receiver node number
Transmitter node number
ARCNET Short-packet format
Fig. 4. ARCNET data transmission format

The first two bytes in each packet indicate the node numbers of the transmitter and receiver, respectively; next byte specifies the address (X) in which the data block start, in a way that the last byte always be the number 256.
The first byte of data (identified as "Data type" in Figure 4) can be either 'S', 'E' or 'C'. Commands 'S' and 'E' indicate, respectively, the start and end of a control sequence; when using these commands, no more data is expected. Command 'C' is used to transmit control data, which must be done periodically. When executing the 'S' command, a timer starts its operation, which checks that the time between each 'C' command does not exceeds a preestablished time limit; in case of occurring this, an alarm signal is activated, and the operation of the robot is blocked. The sequence of commands for normal operation is 'S'→'C'→ ··· →'C'→'E'.
The control computer (either using the motion control board or not) must send periodically the necessary signals to the servo-drives; these must acknowledge, returning to the computer the same command that they receive.
When executing the 'C' command, the control computer sends 44 bytes of data to the motor drives, being two bytes with general information regarding the communication and six bytes with particular information for each of the seven servos; among the latter, two bytes contain flags for turning on/off the servo, activating the brake, selecting the torque/velocity mode, etc.; the other four bytes indicate the desired torque and velocity for the corresponding actuator. On the other hand, 72 bytes are transmitted from the drives to the computer, being two

with general information and 10 for each servomotor, among which we find the status data (on/off, brake, etc.), and the actual joint position, velocity and torque of each motor. More details on this can be found in (Mitsubishi, 2002b).
It is important to highlight that, when using the motion control board and the PA library functions, the user has no access to all this information, and, in fact, he cannot configure the robot in torque mode. In the following subsections we describe the hardware and software interfaces we developed for overcoming such difficulties.

5.2 Hardware interface

In order to work with the PA10 arm in torque mode, it was necessary to have access to the signals commanded to the motor drives. For doing so, we decided to use another PC, equipped with an ARCNET board. This has been done by other authors, for example (Jamisola et al., 2004).
In (Contemporary, 2005) it is explained how a PCI22-48X ARCNET board from Contemporary Controls can be modified to make it work with the PA10 robot. In the same document it is mentioned that even though the PCI22-485X boards are now discontinued, the replacement parts are the PCI20U-485X and the PCI20U-400, from the same brand; all of them have the same ARCNET controller (COM20022); the only difference is the circuit for transmission/reception (known as *transceiver*), which is in the daughter board soldered to the base board.
For the application presented in this work, we acquired a PCI20U-485X board, which was installed in the PCI bus of a PC with a double-core processor, running at 2.4 GHz. This board was configured with the same ARCNET setup of the motion control board. On the other hand, for the sake of compatibility, we made an optical interface board, similar to the one in the original PA10 system.
Following the steps in (Contemporary, 2005), the daughter board with the transceiver was withdrawn from the PCI20U-485X and in its place we put a connector for the cable linking the signals from the main board with the optical interface; this latter plays the role of transceiver and converts the electric signals in optical signals to be transmitted by the optical fiber.
Figure 5 shows the connections required to use the PCI20U-485X board.

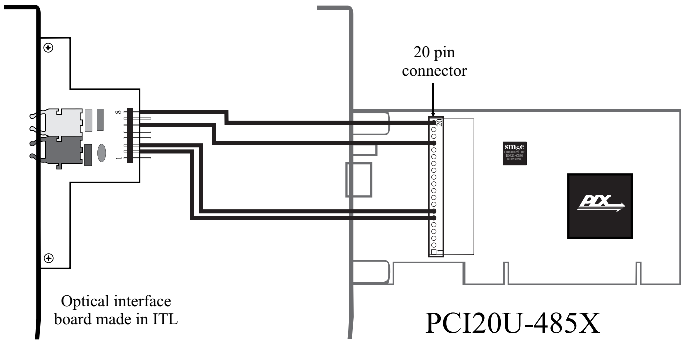


Fig. 5. Connection between the ARCNET board and the optical board

5.3 Real–time software

Real–time control of industrial processes has become an important topic in the recent years. Real–time systems are those in which it is imperative that their response occurs within a limit time. There are real–time operating systems that facilitate the development of this kind of applications. Microsoft Windows is not a real–time operating system, but it can behave as if one, by using an additional software, such as the RTX (real–time extension) from IntervalZero (formerly Ardence).

RTX allows Windows to handle two type of executable programs at the same time: (a) those with extension .RTSS, which are handled completely by the RTX real–time subsystem, thus being more predictable; and (b) those with extension .EXE, handled by the basic Windows subsystem (Win32), which are not so severe with time constraints, but they can include RTX functions and also use graphic interfaces. Thanks to RTX it is possible to simultaneously execute several real–time programs, and share information among them, by means of a shared memory.

To develop real–time applications with RTX, the Visual C++ IDE is required. It was with this platform that we designed the software interface to control the PA10 robot, without using the motion control board. Figure 6 shows a general diagram of the components of the developed software.

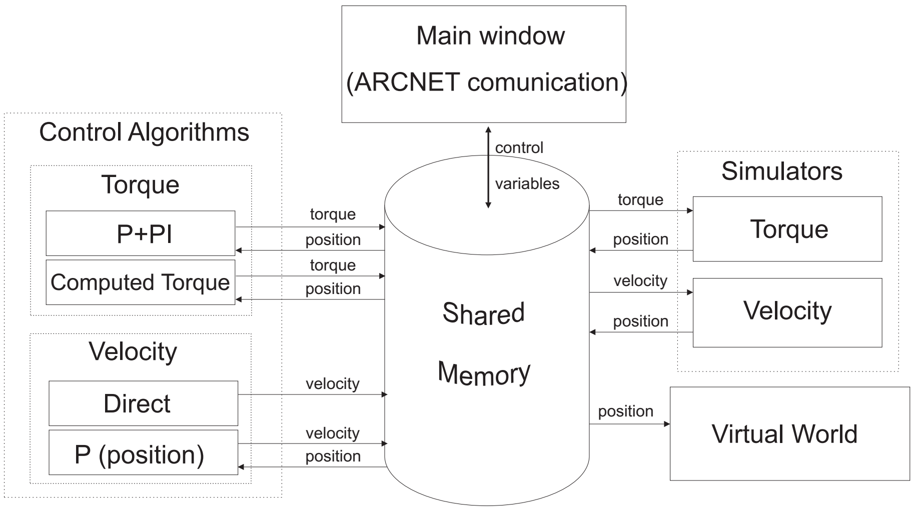
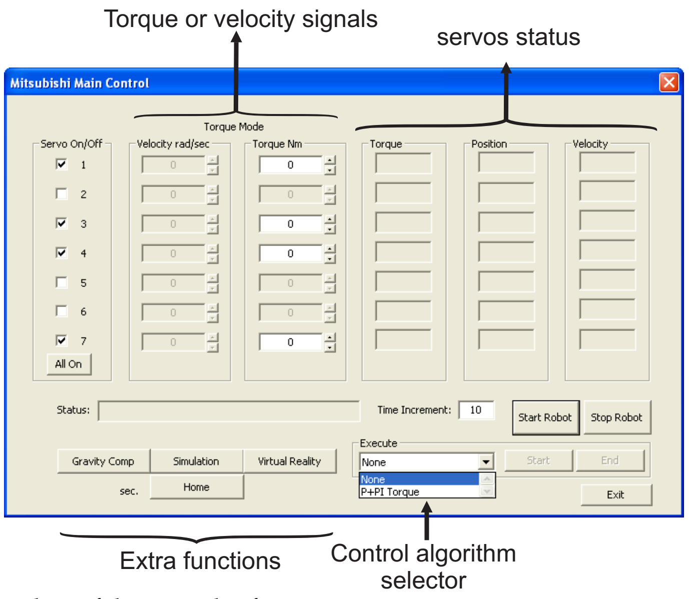


Fig. 6. Components of the control software

The central part of the system is the interchange of information among the different programs via shared memory. These programs are briefly described in the following subsections.

5.3.1 Main window

When starting the main program the user is being asked on what operation mode (velocity or torque) he wants to use for controlling the robot. After that, the main window of Figure 7 shows up; by using this window the user can select the control algorithm to execute, as well as enabling the simulator and the virtual world; further, the window is useful for monitoring and, if necessary, modifying the torque or velocity signals.



Time Increment: 10

Execute: None

Fig. 7. Main window of the control software

It is important to mention that this program establishes the communication with the servo drives via ARCNET. This communication starts and stops by pressing the “Start Robot” and “Stop Robot” buttons in the window.

5.3.2 Simulators

Depending on the operation mode selected, a simulator of the robot can be executed, which resolves the corresponding model in order to obtain the joint coordinates of the robot, starting from the torque or velocity input.

5.3.3 Control algorithms

They are a series of Visual C++ programs which implement different control laws, which can be selected depending on the operation mode selected at the beginning. The number of programs increases when new controllers are added.

5.3.4 Virtual world

It shows, via a 3D animation, the behavior of the robot in real–time, either in simulation or in a real experiment (see Figure 8).

6. Experimental Results

In order to evaluate the performance of the developed real–time software for controlling the PA10-7CE robot arm, we carried out some experiments. The idea was to test the implementation of the two–loop controller studied in Section 2.2 in three different operation modes.

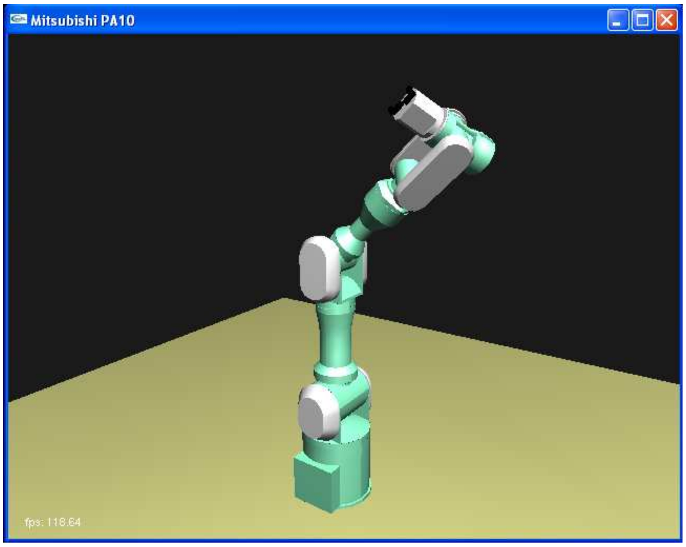


Fig. 8. Virtual world of the PA10 robot

Summing up, the following three cases were considered, being the difference how the inner and outer controllers were implemented (either in the control computer or in the servo–drives):

A. Both RMRC and PI controllers in the real–time software; drives in torque mode.
B. RMRC controller in the real–time software; PI in drives, configured in velocity mode.
C. Both RMRC and PI controllers in the original system, with the motion control board.

The control aim of the experiments was to track a desired time–varying trajectory. In order to show the benefits of redundancy in the PA10-7CE, a secondary task was designed, consisting in maximizing the distance of the joint positions to the actuator limits.

6.1 Desired trajectory

The primary task consisted in tracking a straight line in Cartesian coordinates, subject to the following constraints:

- The robot's end–effector should move only along the Y axis, with respect to the base coordinate frame; hence, x and z coordinates should remain constant.
- The end–effector should perform a turn around the Z axis of the base coordinate frame; hence, β and γ should remain constant.
- Variable operational coordinates (y and α) should perform a cycloidal–type motion, which is given by the following expression (Sciavicco & Siciliano, 2000):

$$s(t) = s_i + (s_f - s_i)\left[\frac{t}{T} - \frac{1}{2\pi}\text{sen}\left(\frac{2\pi\, t}{T}\right)\right] \tag{54}$$

where $s(t)$ is the corresponding operational coordinate (as a function of time t), s_i and s_f are, respectively, the initial and final values of such $s(t)$, and T indicates the duration of the cycloidal motion.

As explained in (Sciavicco & Siciliano, 2000), the cycloidal motion given by (54) produces smooth velocity and acceleration profiles, which is a convenient feature in robotic tasks.
For the experiments, we chose the parameters in Table 1.

Parameter	Value	Unit
x	-0.339	m
y_i	-0.339	m
y_f	0.339	m
z	0.837	m
α_i	0	rad
α_f	1.5708	rad
β	0	rad
γ	0	rad
T	20	s

Table 1. Parameters for the desired trajectory

6.2 Secondary (redundant) task

The PA10-7CE arm is a 7–dof redundant manipulator for pose control tasks (requiring only 6 dof). Thus, we decided to take advantage of the redundant degree of freedom to perform a secondary task, which should not affect the primary task (tracking of the desired trajectory) but only the self–motion of the robot joints.
According to the analysis in Section 2, the secondary task is added to the controller by means of vector $\dot{\boldsymbol{q}}_0$ in (13). Among the different methods of choosing vector $\dot{\boldsymbol{q}}_0$, those considering it as the gradient of a suitable cost function, $H(\boldsymbol{q})$, are the most common (Sciavicco & Siciliano, 2000). For the purpose of the experiments in this chapter, we chose the following cost function:

$$H(\boldsymbol{q}) = \sum_{i=1}^{7} \left(\frac{q_i - \bar{q}_i}{q_{i_{max}} - q_{i_{min}}} \right)^2 \tag{55}$$

where $q_{i_{max}} - q_{i_{min}}$ is the range of operation for the i-th joint, and $\bar{q}_i$ is the central value of this range. Minimizing this cost function implies keeping each of the joints near to their corresponding central values, that is, far from their joint limits.
In order to minimize (55) we should make $\dot{\boldsymbol{q}}_0$ equal to its negative gradient; in other words, the i-th element of $\dot{\boldsymbol{q}}_0$ should be

$$\dot{\boldsymbol{q}}_{0_i} = -\lambda \frac{2(q_i - \bar{q}_i)}{(q_{i_{max}} - q_{i_{min}})^2}$$

where λ is a weighting factor.
For the experiments we used the actual joint limits of the robot, shown in Table 2, with $q_{i_{max}} = -q_{i_{min}}$, and $\bar{q}_i = 0$ for each link i.

6.3 Results

We carried out three sets of experiments, each to test the performance during the execution of the two–loop controller shown in Figure 1, but using different operation modes. The same primary and secondary tasks (mentioned previously) were chosen for the three cases. The

Joint	Limit	Unit
1	180	deg
2	97	deg
3	180	deg
4	143	deg
5	270	rad
6	180	deg
7	270	deg

Table 2. Joint limits for the secondary task

gains of each controller were tuned so that a good performance of the tracking task could be perceived.
Prior to executing the desired task in operational space, a joint position controller was used to move the robot to the required initial pose.

6.3.1 Case A

In this case, the two controllers (outer RMRC and inner velocity PI) where implemented in the developed real–time application, using a sampling period of 10 ms. The control gains were chosen as follows,

- RMRC controller:
$$K = \text{diag}\{40, 50, 40, 12, 12, 12\}$$
- PI controller:
$$K_p = \text{diag}\{80, 120, 40, 48, 5, 8, 15\}$$
$$K_i = \text{diag}\{36, 50, 24, 20, 10, 10, 10\}$$

The weighting factor for the secondary task was chosen to be $\lambda = 2.5$. The servo–drives where configured in torque mode.

6.3.2 Case B

For this case, only the RMRC controller was implemented in the control computer, with the control gains:
$$K = \text{diag}\{24, 24, 24, 8, 8, 8\}$$
The servo–drives were configured in velocity mode, so that the inner PI velocity controller was implemented in them. Factor λ had to be reduced in order to obtain a better performance; its final value was $\lambda = 0.7$.

6.3.3 Case C

In this case we did not used the same software program in the control computer. Instead, we employed an application using a Windows multimedia timer and some API functions from the PA library, to send the desired operational coordinates to the motion control board. The control scheme was also chosen via API functions, in a way of implementing the RMRC in the control board and configuring the servo–drives in velocity mode so as to use their inner PI velocity loops.

6.3.4 Discussion

In order to compare the performance of the three cases, we decided to compute the Euclidean norm of both the position error $\|\tilde{\boldsymbol{p}}\|$ and the orientation error $\|\tilde{\boldsymbol{\phi}}\|$. Figures 9 and 10 show the evolution of such norms for the three cases.

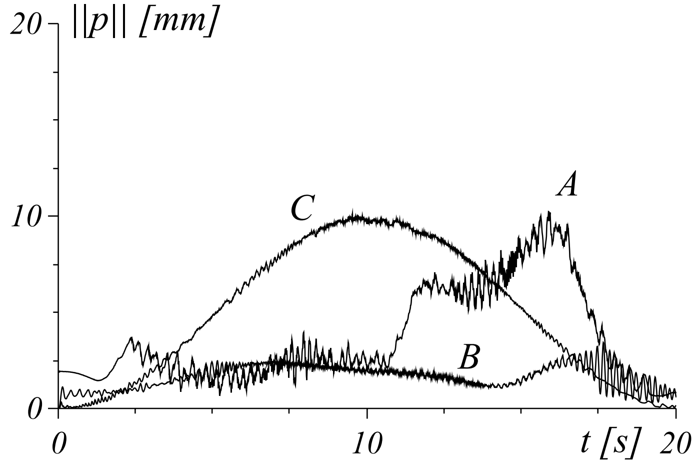


Fig. 9. Time evolution of the norm of the position error.

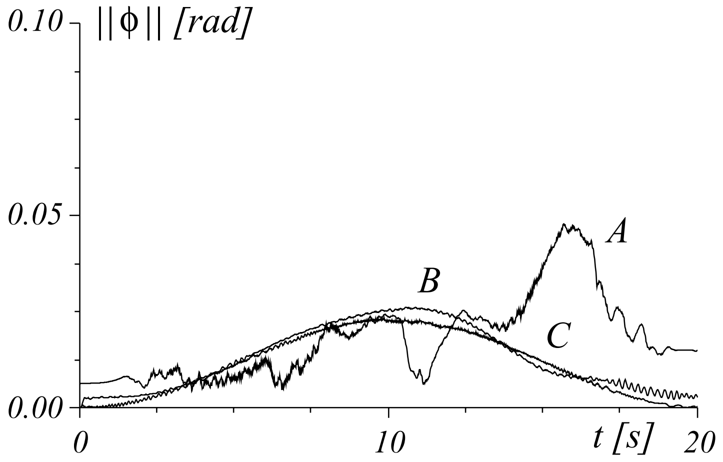


Fig. 10. Time evolution of the norm of the orientation error.

Figures show that even though Case C corresponds to the original setup for the PA10, in the case of the position error, we obtain a better performance when using the drives configured in velocity mode (Case B). As shown in Figure 10, the orientation errors for cases B and C are quite similar. Regarding the results of Case A (drives configured in torque mode) we notice a

more irregular shape of the norms. Some error peaks appear in the last half of the trajectory (when the velocity is decreasing); this is probably due to static friction, which appears at low velocities.
To have a better appreciation of the performance for the three cases, we decided to use a standard index: the root mean square (RMS) value of the norm of the corresponding error, computed in a trip of time T, that is

$$RMS(\tilde{\boldsymbol{u}}) = \sqrt{\frac{1}{T}\int_0^T \|\tilde{\boldsymbol{u}}(\sigma)\|^2 \, d\sigma}$$

where $\tilde{\boldsymbol{u}}$ can be either $\tilde{\boldsymbol{p}}$ or $\tilde{\boldsymbol{\phi}}$. Table 3 shows the performance indexes obtained during the execution of the trajectory ($T = 20$ s):

Index	Case A	Case B	Case C
$RMS(\tilde{\boldsymbol{p}})$	4.379	1.816	6.065
$RMS(\tilde{\boldsymbol{\phi}})$	0.021	0.015	0.014

Table 3. Performance indexes

It is worth noticing that the best performance is obtained with Case B (velocity mode). Further, even though Case A (torque mode) seems to produce more oscillations (see figures 9 and 10), the performance index for the position is lower for Case A than for Case C.

7. Conclusions

In this chapter, we have dealt with the motion control problem in operational space, using the resolved motion rate controller RMRC (kinematic control) plus the intrinsic joint velocity PI controller of the industrial robots, whose solutions of the overall closed–loop system have been proved to have uniformly ultimately boundedness.
Due to its open architecture system, the PA10 has become a suitable robot for both research and industrial applications. This leads to the need of studying the operation of each of the sections that compose its hierarchical structure.
Some experiments were carried out in a PA10-7CE arm, using a software program we have developed for control in real–time. The same hierarchical structure was tested in different operation modes. The results show good performance in all cases, even though the operation in torque mode is more affected by mechanical vibrations, perhaps due to friction and to the discretization of the controllers.

8. Acknowledgements

This work is partially supported by DGEST, PROMEP and CONACyT (grant 60230), Mexico.

9. References

Aicardi, M.; Caiti, A.; Cannata, G & Casalino, G. (1995). Stability and robustness analysis of a two layered hierarchical architecture for a closed loop control of robots in the operational space. *Proceedings of IEEE International Conference on Robotics and Automation*, Nagoya, Japan, 1995.

Camarillo, K.; Campa, R. & Santibáñez, V. (2006). Control of the Mitsubishi PA10-7CE robot using inner velocity PI loops (in Spanish). *Proceedings of 2006 Mexican Congress on Robotics (COMRob06)*, Mexico City, Mexico, October 2006.

Camarillo, K.; Campa, R.; Santibáñez, V. & Moreno-Valenzuela, J. (2008). Stability analysis of the operational space control for industrial robots using their own joint velocity PI controllers. *Robotica*, Vol. 26, No. 6, November 2008, 729-738.

Campa, R.; Torres, E.; Salas, F. & Santibáñez, V. (2008). On modeling and parameter estimation of brushless DC servoactuators for position control tasks. *Proceedings of IFAC World Congress*, Seoul, Korea, July 2008.

Craig, J. J. (2004). *Introduction to Robotics: Mechanics and Control*, Prentice–Hall, 2004.

Contemporary Control Systems (2005). Modification of a Contemporary Controls ARCNET card to control a Mitsubishi Heavy Industries, Inc. PA10 robot arm. Online technical document: www.ccontrols.com/pdf/PA10 procedure.pdf

Denavit, J. & Hartenberg, E. (1955). A kinematic notation for lower–pair mechanisms based on matrices. *Journal of Applied Mechanics*. Vol. 77, 1955, 215-221.

Higuchi, M., Kawamura, T.; Kaikogi, T.; Murata, T. & Kawaguchi, M. (2003) Mitsubishi clean room robot. Mitsubishi Heavy Industries, Ltd., Technical Review, Vol. 40, No. 5, 2003.

Jamisola, R. S.; Maciejewski, A. A. & Roberts, R. G. (2004). Failure–tolerant path planning for the PA-10 robot operating among obstacles. *Proceedings of IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 2004.

Kelly, R. & Moreno, J. (2005). Manipulator motion control in operational space using joint velocity inner loops. *Automatica*, Vol. 41, 2005, 1423-1432.

Kelly, R.; Santibáñez, V. & Loría, A. (2005). *Control of Robot Manipulators in Joint Space*, Springer–Verlag, London, 2005.

Kennedy, C. & Desai, J. P. (2003). Force feedback using vision. *Proceedings of IEEE International Conference on Advanced Robotics*, Coimbra, Portugal, 2003.

Khalil, H. (2005). *Nonlinear Systems*, Prentice Hall, New York, 2005.

Khatib, O. (1987). A unified approach for motion and force control of robot manipulators. *IEEE Journal on Robotics and Automation*, Vol. 3, No. 1, 1987, 43-52.

Mitsubishi Heavy Industries (2002a). Instruction manual for installation, maintenance & safety. General Purpose Robot PA10 series, document 91-10023, 2002.

Mitsubishi Heavy Industries (2002b). Instruction manual for servo driver. General Purpose Robot PA10 series, document SKC-GC20004, 2002.

Nakamura, Y. *Advanced Robotics: Redundancy and Optimization*, Addison-Wesley, 1991.

Oonishi, K. (1999). The open manipulator system of the MHIPA-10 robot. *Proceedings of International Symposium on Robotics*, Tokio, Japan, October 1999.

Pholsiri, C. (2004). Task decision making and control of robotic manipulators. Ph.D. Thesis, The University of Texas at Austin, Austin, TX, 2004.

Qu, Z. & Dorsey, J. (1991). Robust tracking control of robots by a linear feedback law. *IEEE Transactions on Automatic Control*, Vol. 36, No. 9, 1991, 1081-1084.

Ramírez, C. (2008). Dynamic modeling and torque–mode control of the Mitsubishi PA10-7CE robot. Master's Thesis (in Spanish). Instituto Tecnológico de la Laguna, Torreón, Mexico, December 2008.

Ramírez, C. & Campa, R. (2008). Development of a system for real–time control of the Mitsubishi PA10 robot (in Spanish). *Proceedings of 2008 Mexican Congress on Robotics (COMRob08)*, Mexico City, Mexico, September 2008.

Sciavicco, L. & Siciliano, B. (2000). *Modelling and Control of Robot Manipulators*, Springer–Verlag, London, 2000.

Whitney, D. E. (1969). Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on Man-Machine Systems*, Vol. 10, No. 2, June 1969, 47-53.

MRI Compatible Robot Systems for Medical Intervention

Ming Li, Dumitru Mazilu, Ankur Kapoor
and Keith A. Horvath
National Institutes of Health
USA

1. Introduction

Magnetic resonance imaging (MRI) is a non-invasive medical imaging tool that helps physicians diagnose and treat medical conditions. It offers excellent visualization of soft-tissue in any imaging plane without using of iodinated contrast agent or ionizing radiation. In addition to anatomical morphology, MRI can also provide functional information.

With novel fast imaging technologies, MRI became an imaging tool for guiding and monitoring various interventions and biopsy procedures on various organs including the brain, breast, prostate, liver and spine (Jolesz, 1998, Melzer & Seibel, 1999). High performance magnetic field gradients, multi-channel receivers, and advanced reconstruction systems improve the clinical applicability of real time MRI procedures (Nayak et al., 2004, Bock et al., 2004, Guttman et al., 2003, Lederman, 2005). Modern scanners designed for the interventional environment can provide real-time images of acceptable quality in excess of 10 frames per second. As a result, real-time MRI (rtMRI) is becoming an attractive method for many minimally invasive cardiac interventions (Kuehne et al., 2003, Raval et al., 2006, Henk et al., 2005, McVeigh et al., 2006, Horvath et al., 2007).

However the confined physical space of MRI scanner challenges medical intervention, even the magnets with open architecture provide only a limited working space, at the expense of image quality. The use of robots inside the MRI scanner is a very attractive solution: a robot manipulates the intervention instruments while MR images continuously give feedback of the position of the instruments which are controlled by the robot. MR compatible robotic systems have been researched and developed for prostate biopsy and brachytherapy (Chinzei et al., 2000, Krieger et al., 2005, Fischer et al., 2006, Stoianovici et al., 2007a), breast intervention (Kaiser et al., 2000, Larson et al., 2004), interventional spinal procedure (Hempel et al., 2003), neurosurgery (Masamune et al., 1995, Koseki et al., 2002), interventional liver therapy (Hata et al., 2005, Kim et al., 2002), and cardiac intervention (Li et al., 2008).

Design of a system operating inside or close to the bore of a high field MRI scanner is of significant complexity. Due to the strong magnetic field of the MRI, standard materials, sensors and actuators cannot be employed. Due to the confined space of MRI bore, the mechanical design should be compact and simultaneously functional. In this chapter, we discuss MRI robots for medical interventions, including the MR compatibility, mechanical design and control method. As an example, we present our work on the development of a robotic valve delivery module for transapical aortic valve replacement under real-time MRI guidance. Furthermore, we present our work on an MR compatible hands-on cooperative control of a pneumatically actuated robot.

2. MR compatibility

Three basic features for high quality MRI are high and homogenous magnetic fields, fast-switching magnetic field gradients and radiofrequency (RF) pulses. These extreme environmental conditions require adequate construction materials, actuation assemblies, sensors, and proper shielding of electronics. For example, in strong magnetic fields, ferromagnetic materials can be dangerous projectiles. Also, the homogeneity of the main magnetic field is strongly affected by ferromagnetic materials inside the MRI scanner, resulting in substantial distortion of images. Fast changing gradient magnetic fields can induce electrical fields and eddy-currents inside conductive materials. These eddy-currents can heat conductive materials and distort the homogeneity of the main magnetic field and thus severely affect the quality of MR images. RF pulses can heat conductive elements, such as wires or interventional devices. Devices containing closed metallic loops should be avoided in the MR environment.

MRI is also very sensitive to electromagnetic noise. Electronic equipment and wires needed for the operation of mechatronic devices placed inside the MRI scanner room can generate electromagnetic noise if these are not properly shielded.

MRI compatibility of robots has different levels: 1) MR safety, no risk to the patient or medical personnel; 2) image quality not affected by the introduction and operation of the robot; and 3) the operation of the robot not affected by the magnetic fields (Schenck, 1998, Schenck, 2000, Chinzei, 1999). We present MR compatibility of different materials, actuators and sensors and discuss the level of the compatibility of each of these in this section.

2.1 Materials

Based on their reaction in a magnetic field, the materials can be characterized in to three categories: ferromagnetic, paramagnetic and diamagnetic materials. Ferromagnetic materials, such as iron, magnetic stainless steel, cobalt and nickel, exhibit a strong attraction to a magnetic field. These materials are not MR safe. They could be a dangerous projectile to patients, physicians, and the MRI scanner when they are close to the scanner. Moreover, ferromagnetic materials create positive susceptibility to an external magnetic field. The homogeneity of an MR static field will be disturbed by the presence of ferromagnetic materials; therefore the image will be distorted. Yet, the ferromagnetic materials, such as magnetic stainless steel, have desirable mechanical properties. Paramagnetic materials, such as aluminium, platinum and austenitic stainless steels, become slightly magnetized when exposed to a magnetic field. Paramagnetic metals have a small and positive susceptibility to magnetic fields; hence they require some compromise when they are used inside an MRI scanner and close to imaging areas. Diamagnetic metals have a very weak and negative susceptibility to magnetic fields. Copper, silver, and gold are diamagnetic. They are MR safe, and will not affect the homogeneity of a static magnetic field when they are placed

close to an MRI scanner. They have very limited or negligible image artifacts even they are located close to imaging areas.
Another effect of MR compatibility is the generation of the eddy-currents inside conductive materials due to fast-switching gradient magnetic fields. These eddy-currents may alter the local homogeneity of the main magnetic field and affect the quality and linearity of MR images, causing image artifacts. Moreover, eddy-currents may also cause unwanted heating of the materials.
Materials suitable for MR-compatible devices in level 3 are nonmagnetic and nonconductive. Plastics, ceramics and composite material have been used in the development of MR-compatible systems. But plastics are often too soft and present limited structural stiffness, whereas, ceramics are too hard, brittle and could lead to manufacturing difficulties or part failures. Bearings, ball screws and gears made of non-ferromagnetic metals, such as aluminium, copper, brass, titanium, and tantalum are often used in MR compatible systems. These metal parts do not present substantial problems or image artifacts as long as they are of small size and appropriately positioned relative to the imaging areas.
In conclusion, MR-compatible materials can be characterized into three categories depending on their magnetic compatibility when they are placed inside an MRI scanner:

1) materials that produce no image distortion: nylon, vespel, PEEK (poly-ether-ether-ketone), polysulfone, zirconia, plexiglass. Polyimide (from class of vespel materials) for example, presents outstanding structural strength, good chemical resistance, very low electrical and thermal conductivity, low friction and have no detectable MR artifacts.

2) materials that produce low level of image distortion, but for many applications are acceptable: brass, zinc, lead.

3) materials that produce noticeable image distortion, but for some application are still acceptable: titanium, tantalum, tungsten, zirconium, molybdenum, aluminium.

The MR compatibility of certain types of materials and devices have been reported in (Chinzei et al., 1999, Schenck, 2000). In order to be suitable for medical devices or medical robotics, these materials also should be biocompatible, sterilizable and economical.

2.2 Actuators

Electromagnetic motors are commonly used to actuate conventional robotic system, yet they are not safe and compatible in an MRI environment. The principle of electromagnetic motor operation is that a mechanical force is produced by the interaction of an electric current and a magnetic field. Electromagnetic motors usually are equipped with magnets to maintain a magnetic field. When such an electromagnetic motor is placed close to MRI bore, it will obviously disturb the homogeneity of the MR static field. On the other hand, the static magnetic field and the fast-switching gradient magnetic field will also affect the operation of the motor.
Ultrasonic or piezoelectric motors are used in various MRI compatible robotic systems (Masamune et al., 1995, Chinzei et al., 2000, Kaiser et al., 2000, Koseki et al., 2002, Elhawary et al., 2007). These motors are driven by the ultrasonic vibration of a piezoelectric transducer when high-frequency voltage is applied; therefore theoretically they do not produce magnetic fields and are immune to the magnetic field. According to (Chinzei et al., 1999 and Fischer et al., 2008a), some commercially available ultrasonic motors are prone to cause image artifacts and significant signal-to-noise ratio (SNR) loss. The advantages of ultrasonic

motors include compact shape, small size, bidirectional and high torque output. The high breaking torque allows the robot to maintain its position even when it is not be powered. But these motors are not back-drivable, mechanical clutches need to be implemented to manually move the robot without detaching the mechanical part and the motor in the case of a medical emergency. For interventions which require simultaneous use of the imager and the robot, the ultrasonic motors should be placed outside of the MRI scanner. A motion transmission system, such as shafts, belts/chains, cables and linkages, is required to transfer the motion from the ultrasonic motor to the area of interest. Remote actuation can present additional limitations such as, backlash, friction and joint flexibility.

Hydraulic actuators use pressurized hydraulic fluid, such as oil or saline in medical environments (Kim et al., 2002). A hydraulic actuator includes a hydraulic cylinder, a piston with sliding rings and seals, valves, and a supply pressure. Hydraulic actuators can transfer large forces, but the leakage of fluid is an issue in a medical environment. Air bubbles in the fluid could create control problems. Hydraulic actuators are recommended for applications that require high position accuracy, or slow and smooth movements.

Pneumatic actuators have similar construction to hydraulic actuators. Howerver, they use compressed air instead of liquid. Without the threat of the liquid leakage, they are cleaner and are more desirable for a medical environment. Pneumatic actuators can be operated at higher speeds than hydraulic actuators, but due to compressibility of air, they usually provide limited stiffness. Pneumatic actuators are suitable for relatively low-force applications, such as actuating a robot in soft-tissue intervention.

Both hydraulic and pneumatic cylinders made of non-magnetic material can be safely placed close to the imaging areas and will not cause image artifacts. The control valve and control unit should be placed outside of MRI room, or at least outside of the 5 Gauss line.

Pneumatic actuators recently have been used in MRI compatible robotic systems for real-time imaging guided intervention (Hempel et al., 2003, Fischer et al., 2008b, Li et al., 2008). Because of the inter-relationship between temperature, pressure and volume of air, the dynamic control of the pneumatic actuator is challenging. A new concept of pneumatic actuator-PneuStep was presented by Stoianovici and colleagues (Stoianovici et al., 2007b). PneuStep uses a stepper motor principle to achieve precise motion on the order of 0.050mm.

2.3 Sensors

Position and/or force sensors are used to provide feedback for close-loop control of medical robotic devices to provide a safe and accurate operation. Sometimes redundant sensors are used to guarantee the safety in medical intervention systems.

Optical signals do not interfere with magnetic field. Optical sensors have been developed and used for sensing position (Fischer et al., 2008b, Hempel et al., 2003, Hata et al., 2008) and force (Harja et al., 2007, Takahashi et al., 2003) in an MRI environment. A commercially available optical encoder from US digital (Vancouver, Washington, USA) uses transmissive strips for positioning the motion. The reading heads have small electronics to convert the optical signal into an electronic signal. The optical encoder functions well and does not cause image artifacts even if it is placed 5cm from the imaging areas (Fischer et al., 2006). Innomedic (Herxheim, Germany) has developed both rotary and linear optical encoders. These magneto-translucent fiber-optical incremental encoders are made of glass. Optical fibers transfer the signals to optoelectronic conversion circuits remotely located. These encoders do not cause image artifacts even if they are placed at the isocenter of the magnet.

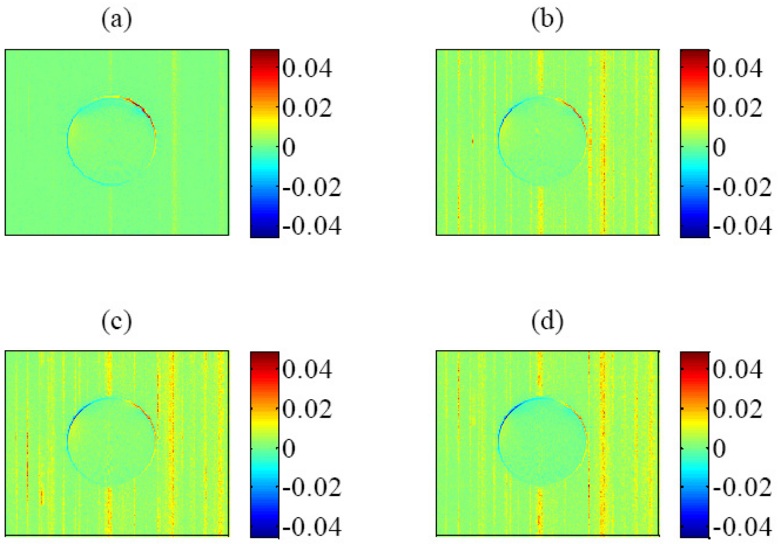


Fig. 1. Mean percentage difference between images under different conditions and baseline image (Phantom only). (a) with robot only (b) with robot inside bore and sensor at 1.0 m (c) sensor moved to 0.5 m and (d) sensor moved to 0.25 m

Another type of important optical sensor is for measuring force in an MRI environment. These optical sensors (Harja et al., 2007, Takahashi et al., 2003, Tada & Kanade, 2005) are used in MRI-compatible robots developed as haptic devices for neuroscience and diagnostics. But they use some electronics which may not function during imaging. Our group modified a commercially available "3D SpaceNavigator" (3dconnexion, Fremont, CA) to create a 6 degrees-of-freedom (DoF) user input sensor which can be placed close to an MRI bore (Kapoor et al., 2009). A "3D SpaceNavigator" optical signal is converted to an electronic signal via a signal PC board, which connects to a USB interface board. After carefully rewiring and shielding, this ergonomic force sensor functions well when it is brought up to 50cm of the bore. Figure 1 shows the percentage difference between images under different conditions with the baseline image.
Force sensing can also be achieved by using hydraulics/pneumatics (Liu et al., 2000, Yu et al., 2007). In their work, Gassert and colleagues (Gassert et al., 2006) presented an MR compatible 1-DoF force sensor based on transmission of hydrostatic force and motion which is located outside the MRI room, where traditional electronics can then be used. Due to friction losses and compressibility, there is a significant dead zone in the force sensed output. However, the advantage of using their technique is that it allows use of the device during imaging.

3. Mechanical Designs

Scanners with open configuration present a major advantage regarding the accessibility to the target area. Double-donut MRI scanner offers a 50cm vertical gap between two facing superconductive magnets. This gap can accommodate a physician and interventional robots. Another type of open scanner with vertical arrangement of magnet poles at 40-45cm distance, also offers good accessibility to the patient. Open-configuration MRI systems with low magnetic fields, in the range of 0.02–0.7 Tesla, result in a lower available signal and slower speeds of image acquisition. The most popular MRI scanners at this time are based on a single superconductive magnet with a horizontal bore. This type of scanner offers better quality imaging with real-time acquisition speeds. The major drawback of these scanners for interventions is the limited space inside the bore, which typically is 60cm in diameter and 200cm in length. A new generation of scanners (Siemens Espree) with short bore (120cm) and increased diameter (70cm) make intervention access and interventional robot assistance within the magnet quite feasible.

MRI scanner configuration, the target organ, and the intervention procedure greatly impact the mechanical designs of MRI robots. MRI compatible robots for percutaneous biopsy, drug injection or radiotherapy seed implantation are currently an active research area. We discuss the mechanism of some of these robots and focus on a robotic valve delivery module for beating heart transapical valve replacement in this section.

Chinzei and colleagues presented a robotic system for assisting a physician who stands in the limited space between the two magnets of a double-donut MRI scanner (Chinzei et al., 2000). The base of the robot is mounted above the physician's head in the open MRI magnet and two rigid arms reach down into the surgical field. The robotic device consists of five linear motion stages arranged to form a 2-DoF orienting mechanism, which is attached to a 3-DoF Cartesian positioning mechanism. The ends of the arms are linked to form a tool holder. Recently, Hata and colleagues developed a semi-active needle-guiding manipulator for microwave thermotherapy of liver tumors in a double-donut open MRI (Hata et al., 2008). The manipulator is installed on the side of the patient table at the gap between the magnets. It consists of a 3-DoF Cartesian stage and an encoded passive remote center of motion configuration. The synergistic control keeps the virtual remote center of motion at the pre-planned target using encoder outputs from the needle holder as input to the motorized base stage.

Koseki and colleagues developed a robotic system for endoscope manipulation working inside a vertical-magnetic-field open configuration MRI scanner (Koseki et al., 2002). This 4-DoF robot takes advantage of the available horizontal space between the vertical poles of such scanners. They adopted a 5-bar linkage mechanism for 2-DoF translational motions. The planar parallel mechanism reduces the size and enables the actuators to be distant from imaging areas. Tajima and colleagues (Tajima et al., 2004) introduced a master-slave MR robotic system for intraoperative diagnosis and minimally invasive surgery inside an open vertical MRI scanner. Each of the slave manipulators has three translational DoF that are located on its base and three rotational DoF on its arm component. The ultrasonic actuators and sensors were located a meter away from the imaging areas. A mechanism consisting of a pair of differential gears and wires was implemented to transmit movement to the tool tip.

Masamune et al. first designed a MRI-compatible manipulator for stereotactic needle therapy (Masamenu et al., 1995) in a conventional closed MRI. The device was designed to

be mounted on the MRI table and above the head of the subject. It has a stereotactic frame with 6-DoF. The isocentric structure avoids collision with the patient.
There are several reports on MRI compatible manipulators and/or robots for prostate biopsy and brachytherapy. These compact devices are mounted on the MRI table and below the torso. Krieger and colleagues developed a 3-DoF manipulator for MRI guided transrectal prostate (Krieger et al., 2005). The manipulator is equipped with active fiducial tracking to encode the position of the needle path and is remotely manually actuated by the physician from outside of the MRI scanner. The device is comprised of a rectal sheath, which is placed adjacent to the prostate in the rectum of the patient and a needle guide, containing a curved needle channel. Stoianovici and colleagues developed a 5-DoF MrBot actuated by a new type of pneumatic stepper motor for brachytherapy seed placement (Stoianovici et al., 2007a). The MrBot robot is constructed in the form of a platform supported by articulated linear actuators in a 5-DoF parallel link structure. The patient is in the decubitus position and seeds are placed in the prostate transperineally. Fischer and colleagues described a 6-DoF robot assistant system for transperineal prostate needle placement (Fischer et al., 2008b). The mechanism consists of two decoupled planar motions. Motion in the vertical plane is achieved using a modified version of a scissor lift mechanism that is traditionally used for parallel plane motion, whereas motion in the horizontal plane is accomplished by coupling two straight line motion mechanisms generally referred to as Scott–Russell mechanisms.
For accessing the organs located in the middle of the torso inside a closed MRI scanner, long flat arm like structure are usually used. Kaiser and colleagues presented ROBITOM, the first MR-compatible robotic system dedicated to MR-guided breast biopsies (Kaiser et al., 2000). The entire assembly is a box-like structure that is anchored on top of the standard MRI scanner table. The patient lies prone on the elevated table with the breasts placed through two openings. Larson and colleagues developed a similar breast-dedicated MR-compatible robotic system (Larson et al., 2004) with multiple degrees of freedom and remote control. Their device is remotely actuated with ultrasonic motors and a motion transmission system consisting of shafts and universal joints. Innomotion (Innomedic GmbH, Herxheim, Germany) is a commercially available MR-compatible interventional robotic system, original developed for precise needle insertion for MR-guided therapy of spinal diseases. An arch is mounted to the MRI patient bed with specially made rails. An arm sitting on the arch takes advantage of the clearance between the patient and the bore. The robot has 5-DoF with a remote center of motion structure to place and orient an interventional tool.

3.1 Robotic module for transapical aortic valve replacement

Our group has developed a robotic system for transapical aortic valve replacement under real-time MRI guidance (Li et al., 2008). The robotic component consists of a 5-DoF robotic arm and a 3-DoF valve delivery module. The 5-DoF Innomotion robotic arm is employed as the 3-DoF valve delivery module holder. The compact valve delivery module is designed for manipulating and placing the prosthesis into a beating heart, while in the MRI scanner. The valve delivery module consists of a sterilizable valve delivery unit and an active mechanism that provides the essential manipulation of the delivery device for valve placement.

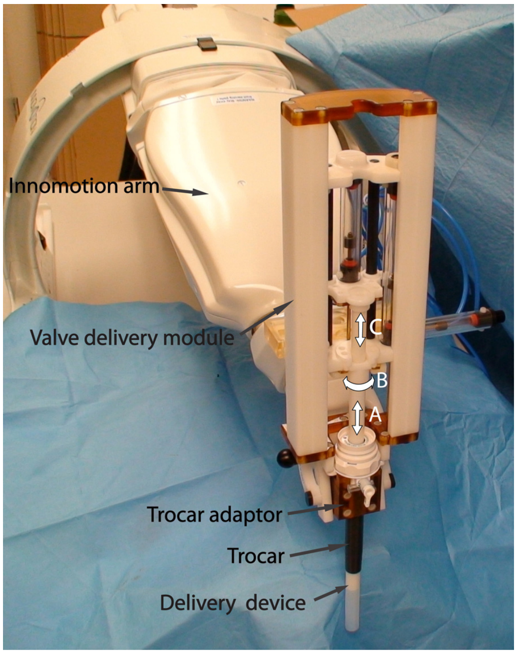


Fig. 2. Picture of the robotic system for MRI guided transapical aortic valve replacement. It shows an Innomotion arm for the positioning module and the prototype of the new developed 3-DoF valve delivery module affixed on the robotic arm.

The valve delivery unit includes a delivery device, a trocar, and a trocar adaptor. The delivery device consists of a straight plastic rod, outside of which is a sheath protecting the prosthetic valve before it is deployed. The robotic active mechanism comprises one rotational joint (B) and two linear joints: the translation joint (A) and the insertion joint (C) (fig 2). The operations of linear and rotational joints are independent.
The translation joint provides linear displacement of the delivery device along its axis. This joint is directly actuated by two parallel linear pneumatic actuators. The two-actuator structure guarantees balanced motion of a delivery device. The bodies of two linear actuators, together with a base plate, a top plate and two inner rails, form a rigid sliding frame. The sliding frame can slide on two frame rails. These two frame rails, along with a connecting plate (for mounting on a positioning arm) and a top plate form a rigid base frame. The position of the motion is sensed by a linear optical encoder that is attached to a frame rail of the rigid frame. This travel range of this joint is 90mm.
The rotational joint rotates the delivery device around its axis. This changes the orientation of the prosthesis relative to coronary ostia before it is deployed. It is actuated by a linear pneumatic actuator attached on the base plate of the sliding frame. The linear movement is transmitted to the rotational movement by a rack-gear transmission. The rotation angle is sensed by a linear optical encoder. A sliding base rack actuated by the pneumatic actuator transmits linear movement to the gears of the rotation axis. The travel range of this joint is ±60 degree.

The other linear joint is an insertion joint. This linear movement advances the prosthetic valve out of the delivery device. One central sliding stage with a semicircular groove actuated by a linear actuator can slide on the inner rails of the rigid sliding frame. The whole travel length of the insertion is 45mm, which guarantees the prosthesis be driven out from the sheath. This linear movement is also encoded with an optical linear encoder.
The prototype of the valve delivery module is shown in Figure 2. All parts of the prototype delivery module are made from non-conductive plastic materials. Pneumatic actuators and optical sensors are used for operating and positioning each joint. The robotic module allows precise and repeatable positioning of a bioprosthetic valve in the correct location. The maximum measured force load of the translation stage, under 75 psi operation air pressure, was 34N; and the maximum torque of the rotation stage was 0.4Nm. With operational velocity of 10mm/sec, the accuracy of translation stage was found to be 0.19±0.14mm. With operational velocity of 5deg/sec, the accuracy of the rotation stage was found to be 0.46±0.27deg. With a load of 10N (a reasonable force if we consider the friction and reaction force from a real case), the module moved smoothly and stably and the accuracy was in the same range. The presence and motion of the robotic module inside the scanner was found to have no noticeable disturbance to the image. The observed signal-noise-ratio (SNR) loss was 6.1% to 6.5% for the valve delivery module placed in the bore and in motion respectively.

4. Control Strategies

Improving precision and accuracy, while maintaining compatibility and safety with MRI are the prime concerns for most MRI compatible robotic systems. Point-to-point position motion is the basic control mode for these robots. However, a shared control between a user and a MRI compatible robot makes it a more intuitive instrument especially during the setup phases of interventions. We describe the needs and method to implement the hands-on coorperative control mode on a MR compatible robot in this section.
Planning a minimally invasive intervention on living tissue is not a trivial task. The organs move around and thus the pre-planned motion based on pre-operative images alone is not sufficient.

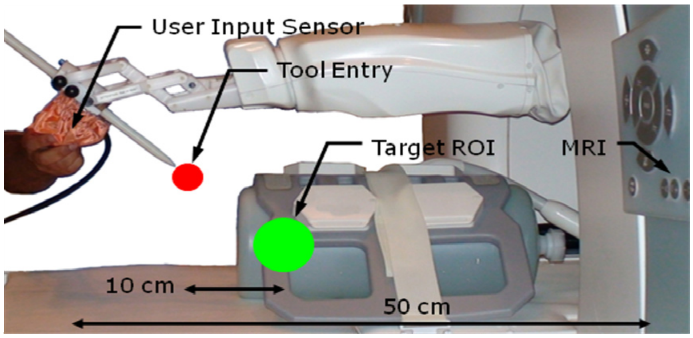


Fig. 3. Hands-on cooperative control

Most of the contemporary systems make use of intra-operative images and a graphical user interface to update the planned motions. It may be not necessary and sometimes impractical to have an image guided interface for the robot during procedure setup. In the robotic assisted intervention of beating heart transapical valve replacement (Li et al., 2008), before the prosthetic valve delivery process can be started, the surgeon must perform preparatory procedures of placing the trocar into the apex of the heart. Thereafter, the surgeon loads the delivery device with the prosthetic valve and inserts the delivery device into the trocar. Using imaging to guide the placement at this stage was unnecessary and impractical because of the large motion and variability in the localization of the trocar. Typically, the trocar port is about 10-15cm from imaging center, thus requiring a very large image acquisition volume. Breathing motion of up to 20mm also causes localization and registration errors. Moreover, adjustments may also be required to the entry point after a preliminary scan of the delivery device is acquired.

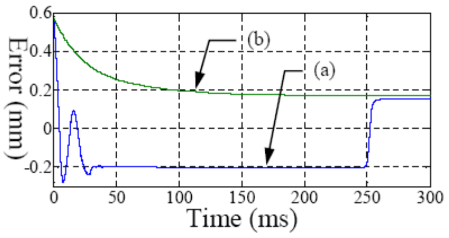


Fig. 4. Position error in the PIV controller for one of the robot axis under two different modes. (a) Gains optimized for point-to-point mode (b) Gains for hands-on mode.

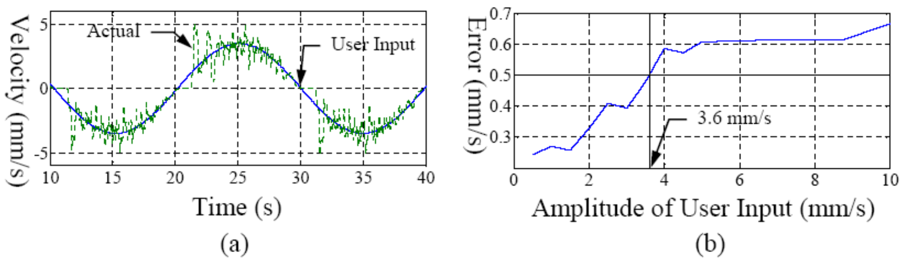


Fig. 5. (a) User input with amplitude of 3.5mm/s and corresponding actual Cartesian velocity (b) Average Cartesian velocity error over time (2 cycles) with respect to amplitude of user input.

Figure 3 shows the setup of the Innomotion robotic arm and user input sensor in the MRI room. We describe the method to implement the hands-on coorperative control mode on a modified pneumatically actuated Innomotion robotic arm (Innomedic, Herxheim, Germany). The actuators of the Innomotion robot are controlled by an off-the-shelf controller (Motion Engineering Inc/Danaher Motion, Santa Barbara, CA). A PIV (proportional position loop integral and proportional velocity) controller runs on the proprietary DSP based hardware.

4.1 Low level control

One important difference between hands-on control mode and other modes is that the later prefers zero or low steady state errors. Hands-on mode has a user in the control loop who can tolerate a fair amount of steady-state error by adjusting his/her input to achieve the desired result. This mode demands a fast and smooth response without oscillations during the transient as well as steady state. Low level PIV gains should satisfy this requirement. It is straight forward to obtain such a response on traditional motor driven robots. However, for a pneumatic robot a compromise between smoothness, speed and error has to be made. To achieve this behavior, we applied a sine wave as user input (velocity) with varying amplitudes. Iteratively, we adjusted the PIV gains and the maximum amplitude of user input that would produce reasonable error, till we saw no further improvement. In figure 4, the fast oscillatory response for point-to-point gains is shown along with a smoother, reasonably fast response for hands-on mode. Figure 5(a) shows a typical profile for a user input with amplitude of 3.5mm/sec and the actual Cartesian velocity measured using robot encoders. The error is defined as the difference between the user input and the actual Cartesian velocity at that instant. The average error is computed by averaging the error over the number of samples collected during two cycles of user input. Figure 5(b) shows this value for different amplitudes of user input. Our current system can tolerate a maximum user input speed of 3.6mm/sec and 3.6deg/sec, if the required error is to be kept below 0.5mm/sec and 0.5deg/sec.

4.2 High level admittance control

Depending on how the inertias and friction forces compare with the forces applied by the environment to the robot, it is possible to classify robotic devices into two categories: impedance-type and admittance-type. More importantly, it is the control scheme that distinguishes the two types. The impedance-type robots act as a force/torque source, where the controller outputs a force based on desired input positions (and their derivatives). In admittance-type robots, the controller outputs a desired position (and its derivatives) based on user inputs such as force measurement or joystick motion. Innomotion robot is practically "non-back-drivable", i.e., it requires significant effort to overcome internal friction to maneuver it. Thus, it is more suited for an admittance control scheme such as those explored with the Johns Hopkins University Steady-Hand Robot (Taylor et al., 1999). In their work, force sensors could be used to obtain user input. However, MRI compatibility poses certain challenges in obtaining a 6-DoF user input. The outline of our hands-on controller is as follows:

1) Switch the low-level PIV gains to ones optimized for hands-on mode.

2) Obtain incremental motion desired by the user, that is $\dot{x} = K_c \cdot f$, where $\dot{x}$ is desired incremental motion, $f \in R^6$ is user input and K_c is a scaling matrix.

3) Compute current joint state, q , from current actuator values, a , and incremental motion in the actuators, Δa .

4) A constrained least squares problem is solved for the optimal incremental joint motion, Δq^* by the high-level controller. The least-squares (LS) problem has an objective function describing the desired outcome. It includes constraints that consider joint limits, and importantly velocity limits.

$$
\begin{aligned}
\Delta q^* &= \arg\min_{\Delta q} \left\| W_c \cdot \left(\Delta x - K_c f \right) \right\| + \left\| W_q \cdot \Delta q \right\| \\
s.t. \quad & |q| \le \dot{q}_U \cdot \Delta t \\
& q_L \le q + \Delta q \le q_U \\
& \Delta x = J \cdot \Delta q
\end{aligned}
$$

The incremental joint motion, Δq, is a variable for LS problem. Matrix J is the Jacobian of the robot. q_L and q_U are the lower and upper bounds of the joint variables. $\dot{q}_U$ is the upper bound of the joint velocities that are obtained as described in earlier section. Δt is the small time interval of the high-level control loop. Without any constraints, the above constrained LS problem, which is implemented in the high-level block, is equivalent to resolved rate control. W_c and W_q are diagonal weight matrices.

5) Numerically integrate the incremental joint motion to arrive at a new set of joint positions. We assume that for each iteration loop, the incremental motions are sufficiently small and $\Delta x / \Delta t = J \cdot \Delta q / \Delta t$ represents a good approximation to the instantaneous kinematics.

6) Compute desired actuator values, a^d and desired actuator velocities, $\dot{a}^d = \Delta a^d / \Delta t$ from q and Δq^*. These are new set points for the low-level controller.

7) Repeat steps 2-6 while in hands-on mode. On exit, switch PIV gains to point-to-point mode values.

Numerical integration and rate control laws such as these are known to be "non-conservative" and may result in positional errors. However, this is not a concern here, since a human-in-the-loop can easily account for any positional error by applying required additional input to reach the desired goal. The desirable behavior here is that the robot continues to follow the user input as best as it can, even in the advent of certain limits, such as joint or velocity extremity.

4.3 Experiments and Results

We tested the system functionality, as well as, the qualitative ease of use of the sensor. We also evaluated the hands-on sensor interface with respect to current and alternate interfaces. We used a quintessential peg-in-the-hole task to emulate the clinical scenario that inserts a dexterous tool into a trocar inside a thoracic cavity. Our peg was a cylinder 12.5mm in diameter and 140mm long, which mimicked the dimension of the delivery device. To design our experiments to be in line with Fitt's law (Zhai, 1995), we used two different hole dimensions (13.5mm and 16.5mm) for fine and coarse positioning accuracy, respectively. Further, the larger hole had a larger bevel at the entry point. The hole was placed at a known position with respect to the robot and the starting configuration of the robot was chosen at random for each trial. The starting position was chosen using an uniform random distribution from a spherical annular region of radii $15\sqrt{3}$ to $20\sqrt{3}$ mm, and the orientations were picked in the interval of ±15 - ±20deg. These are clinically relevant distances, because the robot can be easily and quickly positioned to within these regions using available passive adjustments.

For comparision, we developed a simplified GUI that resembles the commands acceptable to the AESOP, that is, the surgeon could annunciate one of ”move“-“left”, “right”, “up”, “down”,“front”,“back”, or “rotate”-“left”, “right”, “front” and “back” to move the 5-DoF robot in the appropiate direction. The person at console would have to press the corresponding button. The buttons to increase/decrease speed would change the current set speed by 0.5mm/sec. In our experimental setup, the person at console could not directly visualize the robot to replicate the clinical setting. The person at console was picked at random from the pool of users to avoid bias.
Each user was asked to perform the task as quickly as possible, on the “start” cue of the person on console and indicate verbally when they completed the task by announcing “done”. Comparisons of task times recorded using a stop watch have been shown in figure 6. As seen, hands-on approach is efficient for both the levels of accuracy.

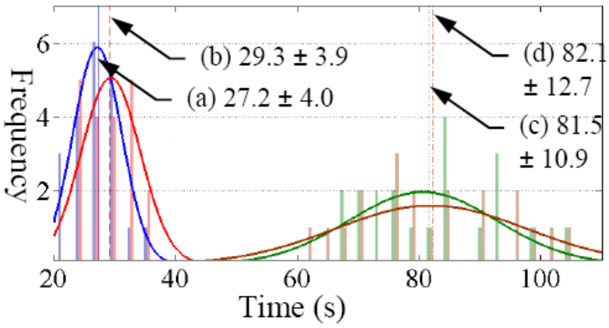


Fig. 6. Histogram of time required to complete the peg-in-hole task (a,b) hands-on and hole size of 16.5mm and 13.5mm, respectively (c,d) AESOP like interface and hole size of 16.5mm and 13.5mm, respectively. The solid curves are the fitted Gaussian distributions with means and deviations as indicated.

This can be attributed to the ability to perform complex continuous motions involving multiple translational and rotational degrees of freedom at one time. Further, using the GUI interface, the user announcing the commands has to contemplate his/her next moves. As expected, there is an increase in time required with an increase in the difficulty of the task. Though, this difference is not significant when compared with the difference between the two approaches.

5. Conclusion

MRI provides excellent visualization of soft tissue, its sub-structure and surrounding tissues. The unique features of MRI, such as oblique image planes, multi-slice imaging, real-time visualization, and freedom from radiation exposure risk, enable MRI to be a critical tool in the guidance of many interventional procedures. Additionally, a mechatronic system can provide more accurate and smooth access to targeting organs in a confined space. The marriage of a MRI and a robot makes the benefit of minimally invasive interventions substantial.

The high magnetic field and the confined space of MR bore presented many technical challenges. The design of robotic system is dependent of MRI scanner configuration type, medical intervention and target areas. The mechatronic components including actuators, sensors and controllers must be able to work in accurate, stable and robust way in MR environment. Materials used for robotic system should have low magnetic susceptibilities (comparable with air, water or human tissue), low electrical conductibility, adequate mechanical strength and good manufacturing properties. Certain conductive materials can be used, but only for small parts and without loops. If it is necessary, electrical components may be brought into MR environment if their electrical signals are of low frequency and are shielded.

Control strategy and human machine interface for MRI compatible robotic systems for medical interventions need to be studied. A shared control between a user and an MRI compatible robot makes it a more intuitive instrument especially during setup phases of interventions. An image-guided interface would be ideal to plan and manipulate the robotic holder or adjust the interventional tool when the robot is inside the magnet bore. In the engineering of robots for medical applications, detailed analyses of the functions of the entire system, that is, robot, interfaces and application, taken as single entity, are arguably more important than the individual performance of the subsystems (robot, surgeon, interfaces and application, separately). Thus, having a combination of more than one interface such as an image-guided, console guided or hands-on based on application might yield a higher performance from the entire system.

6. References

Bock, M.; Volz, S.; Zuhlsdorff, S. & et al. (2004). MR-guided intravascular procedures: Real-time parameter control and automated slice positioning with active tracking coils. *Journal of Magnetic Resonance Imaging.* Vol.19, 580-589

Chinzei, K.; Kikinis, R. & Jolesz, F. (1999). MR Compatibility of Mechatronic Devices: Design Criteria. *Proc. of MICCAI '99 Lecture Notes in Computer Science.* Vol.1679/1999, 1020-1031.

Chinzei, K.; Hata, N.; Jolesz, F.A. & et al. (2000). MR compatible surgical assist robot: system integration and preliminary feasibility study. *Proc. of MICCAI '00 Lecture notes in Computer Science.* Vol.1935/2000, 921-930.

Elhawary, H.; Zivanovic, A.; Rea, M & et al. (2007). A MR compatible mechatronic system to facilitate magic angle experiments in vivo. *Proc. of MICCAI '07 Lecture notes in Computer Science.* Vol.4792/2007, 604-611

Fischer, G.S.; Iordachita, I.; DiMaio, S.P. & et al. (2006). Design of a Robot for Transperineal Prostate Needle Placement in an MRI Scanner, *Proc. of IEEE International Conference on Mechatronics.* pp 592-597, Budapest, Hungary.

Fischer, G.S.; Krieger, A.; Iordachita, I. & et al. (2008a). MRI Compatibility of Robot Actuation Techniques – A Comparative Study. *Proc. of MICCAI '08 Lecture Notes in Computer Science.* Vol.5242/2008, 509-517

Fischer, G.S.; Iordachita, I.; Csoma, C. & et al. (2008b). MRI-Compatible Pneumatic Robot for Transperineal Prostate Needle Placement. *IEEE/ASME Transactions on Mechatronics.* Vol.12(3), 295-305.

Gasset, R.; Moser, R.; Burdet, E. & et al. (2006). MRI/fMRI-compatible robotic system with force feedback for interaction with human motion. *IEEE/ASME Transactions on Mechatronics*. Vol. 11(2), 216–224

Guttman, M.A.; Kellman, P.; Dick, A.J. & et al. (2003). Real-time accelerated interactive MRI with adaptive TSENSE and UNFOLD. *Magnetic Resonance in Medicine*. Vol.50, 315-321

Harja, J.; Tikkanen, J.; Sorvoja, H. & et al. (2007). Magnetic resonance imaging-compatible, three-degrees-of-freedom joystick for surgical robot. *International Journal of Medical Robotics and Computer Assisted Surgery*. Vol.3(4), 365–371

Hata, N.; Hashimoto, R.; Tokuda, J. & et al. (2005). Needle guiding robot for MR-guided microwave thermotherapy of liver tumor using motorized remote-center-of-motion constraint. *Proc. of IEEE International Conference on Robotics & Automation*. pp. 1652-1656, Barcelona, Spain.

Hempel, E.; Fischer, H.; Gumb, L. & et al. (2003). An MRI-compatible surgical robot for precise radiological interventions. *Computer Aided Surgery*. Vol.8(4), 180–191

Henk, C.B.; Higgins, C.B.; & Saeed, M. (2005). Endovascular interventional MRI. *Journal of Magnetic Resonance Imaging*. Vol.22, 451–460.

Horvath, K.A.; Guttman, M.; Li, M. & et al. (2007) Beating heart aortic valve replacement using real-time MRI guidance. *Innovations*. Vol.2, 51-55

Jolesz, F.A. (1998). Interventional and intraoperative MRI: a general overview of the field. *Journal of Magnetic Resonance Imaging*. Vol.8, 3–7

Kaiser, W.A.; Fischer, H.; Vagner, J. & et al. (2000). Robotic system for biopsy and therapy of breast lesions in a high-field whole-body magnetic resonance tomography unit. *Investigative Radiology*. Vol.35, 513–519

Kapoor, A.; Wood, B.; Mazilu, D. & et al. (2009). MR-compatible Hands-on Cooperative Control of a Pneumatically actuated robot. *Proc. of IEEE International Conference on Robotics & Automation*. pp. 2681-2686, Kobe, Japan.

Kim, D.; Kobayashi, E.; Dohi, T. & et al. (2002). A new, compact MR-compatible surgical manipulator for minimally invasive liver surgery. *Proc. of MICCAI '02 Lecture Notes in Computer Science*. Vol.2488/2002, 164-169

Koseki, Y.; Washio, T.; Chinzei, K. & et al. (2002). Endoscope manipulator for trans-nasal neurosurgery, optimized for and compatible to vertical field open MRI. *Proc. of MICCAI '02 Lecture Notes in Computer Science*. Vol.2488/2002, 114–21.

Krieger, A.; Susil, R.C.; Menard, C. & et al. (2005). Design of a novel MRI compatible manipulator for image guided prostate interventions. *IEEE Transactions on Biomedical Engineering*. Vol. 52, 306–313

Kuehne, T.; Saeed, M.; Higgins, C.B. & et al. (2003). Endovascular stents in pulmonary valve and artery in swine: feasibility study of MR imaging-guided deployment and postinterventional assessment. *Radiology*. Vol.226, 475–481.

Larson, B.T.; Erdman, A.G.; Tsekos, N.V. & et al. (2004). Design of an MRI-compatible robotic stereotactic device for minimally invasive interventions in the breast. *Journal of Biomechanical Engineering*. Vol.126, 458–65

Lederman, R.J. (2005). Cardiovascular interventional magnetic resonance imaging. *Circulation*. Vol.112, 3009–3017.

Li, M.; Mazilu, D. & Horvath, K. (2008). Robotic system for transapical aortic valve replacement with MRI guidance. *Proc. of MICCAI '08 Lecture Notes in Computer Science.* Vol.5242/2008, 476-484

Liu, J.Z.; Dai, T.H.; Elster. T.H. & et al. (2000). Simultaneous measurement of human joint force, surface electromyograms, and functional MRI-measured brain activation. *Journals of Neuroscience Methods*. Vol.101(1), 49-57

McVeigh, E.R.; Guttman, M.A.; Lederman, R.J. & et al. (2006). Real-Time Interactive MRI-Guided Cardiac Surgery: Aortic Valve Replacement Using a Direct Apical Approach. *Magnetic Resonance in Medicine.* Vol.56, 958-964.

Melzer, A. & Seibel, R. (1999). MRI-guided treatment of degenerative spinal diseases. *Minimally Invasive Therapy & Allied Technologies.* Vol.8(5), 327-335

Nayak, K.S.; Cunningham, C.H.; Santos, J.M. & et al. (2004). Real-time cardiac MRI at 3 Tesla. *Magnetic Resonance in Medicine.* Vol.51, 655-660

Masamune, K.; Kobayashi, E.; Masutani, Y. & et al. (1995). Development of an MRI-compatible needle insertion manipulator for stereotactic neurosurgery. *Computer Adied Surgery*. Vol.1, 242–248

Raval, A.N.; Karmarkar, P.V.; Guttman, M.A. & et al. (2006). Real-time magnetic resonance imaging-guided endovascular recanalization of chronic total arterial occlusion in a swine model. *Circulation*. Vol.113, 1101–1107.

Schenck, J.F. (1998). MR safety at high magnetic fields. *Magnetic Resonance Imaging Clinics of North America.* Vol.6, 715–30

Schenck, J.F. (2000). Safety of strong, static magnetic fields. *Journal of Magnetic Resonance Imaging.* Vol.12, 2-19

Stoianovici, D.; Song, D.; Petrisor, D. & et al. (2007a). "MRI Stealth" robot for prostate interventions. *Minimally Invasive Therapy & Allied Technologies.* Vol.16, 241-248

Stoianovici, D.; Patriciu, D; Petrisor, D. & et al. (2007b). A new type of motor: Pneumatic step motor. *IEEE/ASME Transactions on Mechatronics.* Vol.12(1), 98-106

Tada, M. & Kanade, T. (2005). Design of an MR-compatible three-axis force sensor. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3505–3510, Edmonton, Canada.

Tajima, F.; Kishi, K.; Nishizawa, K. & et al. (2004). A Prototype Master-Slave System Consisting of Two MR-Compatible Manipulators with Interchangeable Surgical Tools Part of a Unified Support System for Diagnosis and Treatment. *Proc. of IEEE International Conference on Robotics & Automation*, pp. 2505-2510, New Orleans, USA.

Takahashi, N.; Tada, M.; Ueda, J. & et al. (2003). An optical 6-axis force sensor for brain function analysis using fMRI. *Proc. of IEEE Sensors*, pp. 253–258, Toronto, Canada.

Taylor, R.H.; Jensen P.; Whitcomb, L.L. & et al. (1999). A Steady-Hand robotic system for microsurgical augmentation. *International Journal of Robotics Research*. Vol.18(12), 1201–1210

Yu, N.; Murr, W; Blickenstorfer, A. & et al. (2007). An fMRI compatible haptic interface with pneumatic actuation. *Proc. of IEEE International Conference on Rehabilitation Robotics*, pp. 714–720, Noordwijk, Netherlands

Zhai, S. (1995). Human Performance in Six Degree of Freedom Input Control. *Ph.D. dissertation, University of Toronto*

On the Optimal Singularity-Free Trajectory Planning of Parallel Robot Manipulators

Chun-Ta Chen and Te-Tan Liao
Da Yeh University,Far East University
Taiwan

1. Introduction

Parallel robot manipulators comprise a mobile platform connected to a fixed base through three or more articulated links and are used extensively throughout industry for such diverse applications as high-precision positioning systems, fiber alignment, welding, robotic manipulators, automatic inspection systems, and so forth. Therefore, planning a trajectory to perform a specific task is one of the most important class of problems in the applications of the parallel robot manipulators.
However, while moving along a specified trajectory, due to the limits on the workspace and existence of the force singularities, the parallel robot manipulators may not oppose forces or moments at some configurations. As a consequence, the manipulator gains some degrees of reedom, and **becomes** uncontrollable. Even the manipulator is very near to a singular manifold, the leg forces will increase violently to reach their allowable limits. Therefore, it is meaningful to plan a path without crossing a singular manifold on any operation for the parallel robot manipulators.
Compared to the vast researches on the path-programming of serial manipulators, studies on the singularity-free path programming of the parallel robot manipulators are relatively few. (Bhattacharya et al., 1998) developed an exact and an approximate on-line singularity avoidance method to restructure a path in the vicinity of a singular manifold for platform type parallel manipulators. (Dasgupta & Mruthyunjaya, 1998) proposed an algorithm to obtain a singularity-free trajectory for given two end-poses. The continuous paths are constructed through well-conditioned via points by examining the condition number at discrete steps on the corresponding straight line segment. (Sen et al., 2003) used a variational approach based on a Lagrangian to plan singularity-free paths with the actuators lengths remaining within their allowable limits. (Dash et al., 2005) used local routing method based on Grassmann’s line geometry to avoid isolated singularities inside the reachable workspace of parallel manipulators.
However, at present, most researches on the singularity-free path programming only deal with the kinematics of the parallel robot manipulators. When repetitive tasks in industrial applications are considered, some of physical operating costs, such as actuating forces or energy consumption, will become significantly important. Consequently, these effects should be further taken into account for a singularity-free path programming.

In this chapter, the dynamics formulation of a general parallel robot manipulator with an arbitrary geometry and inertia distribution according to the Boltzmann-Hamel-d'Alembert formulation will be first described. Then, the developed dynamics model in terms of the task-space coordinates are used to implement the singularity-free path programming for given two end poses of the parallel robot manipulator by minimizing some cost functions such as actuating forces, travel time and energy expenditure.

2. The Boltzmann-Hamel-d'Alembert formulation

2.1 Dynamic equations in terms of hybrid space coordinates

When planning a singularity avoidance path with the minimization of a cost function, *e.g.* actuating forces, travel time and energy consumption, *etc.*, the dynamic equations of the parallel robot manipulators should be developed for this purpose. Several methods such as the Newton-Euler formulation (Do & Yang, 1998), (Dasgupta & Mruthyunjaya, 1998), (Dasgupta & Mruthyunjaya, 1998), (Nakamura & Ghodoussi, 1989), (Nakamura & Yamane, 2000), virtual work principle (Zhang & Song, 1993), (Wang & Gosselin, 1998), (Tsai, 2000), Kane's method (Liu et al., 2000), kinematic influence coefficient theory (Wang et al., 2003) and the traditional Lagrangian formulation in generalized coordinates (Lebret et al., 1993), (Pang & Shahinpoor, 1994) are proposed for modeling and simulation of the dynamics of parallel manipulators. Comparing these approaches each other, it is obvious that the Lagrangian formulation has more physical insight because it utilizes the kinetic and potential energies to generate a well-structured form of equations of motion. However, if generalized coordinates are selected to express rotations of a structure in space, the partial derivatives of the Lagrangian by Lagrangian formulation in terms of generalized coordinates are quite tedious due to more intensive symbolic computations on time-varying inertia matrix. Therefore, such a direct formulation in all generalized coordinates is so arduous and cumbersome that the dynamic equations were developed most only under some simplifying assumptions regarding the geometry, configuration, structure and inertia distribution of manipulators. Rather than using all of generalized coordinates for the dynamics formulation, here the angular velocities as the time derivatives of a set of quasi-coordinates instead of generalized coordinates are utilized. Such angular velocities expressions are so-called quasi-velocities. The formulated kinetic and potential energies are expressed in matrix forms as function of quasi-velocities and rotation matrices, and then the Boltzmann-Hamel-d'Alembert formulation is employed to derive closed-form dynamic equations of the parallel robot manipulators with a completely general architecture and inertia distribution.

Fig. 1 illustrates the geometric structure of the general parallel robot manipulators considered in the present analysis. As shown, the mechanism has a fixed base and a moving platform connected to the base through six extensible legs with a spherical joint B_i at the top of the leg *i* (*i*= 1,…,6) and a universal joint A_i at the base end. Each leg is composed of two segments, the lower segment 1 is a fixed part and the upper segment 2 is a moving part, both are joined together by a prismatic joint, and each prismatic joint is driven by a linear actuator.

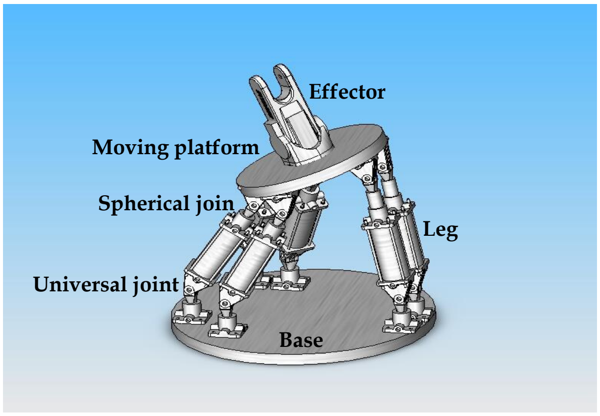


Fig. 1. Schematic diagram of a general 6-UPS parallel robot manipulator

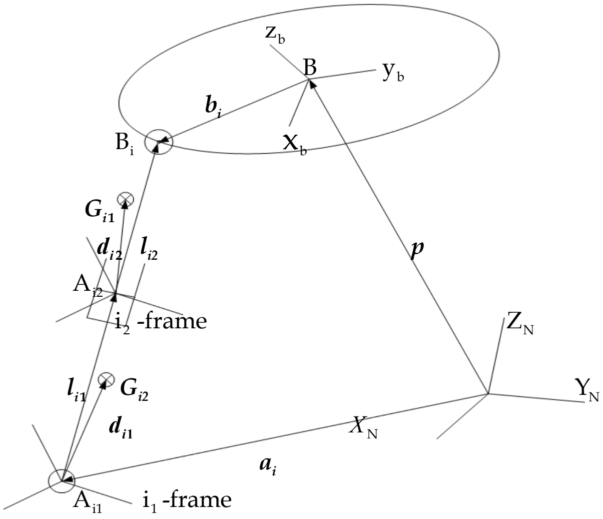


Fig. 2. Coordinate systems for one leg and moving platform

Consider one of the six legs and the moving platform for a dynamic analysis as shown in Fig. 2, an inertial coordinate system N-frame, $X_N Y_N Z_N$, is fixed at the base and another coordinate system B-frame, $x_b y_b z_b$, is attached to the moving platform with its origin at the gravity center B of the moving platform. Two parallel local coordinate systems, i_1-frame and i_2-frame with their corresponding origins A_{i1} and A_{i2}, are attached to the respective lower segment 1 and upper segment 2 of the leg *i*. With the definitions of coordinate systems, the position vector of the gravity center of the moving platform with respect to the N-frame is denoted as $\boldsymbol{p}$, the position vector of A_{i1} with respect to the N-frame is represented as $\boldsymbol{a}_i$, the position vector of the ball joint B_i with respect to the B-frame is defined as $\boldsymbol{b}_i$, and the length vector of leg i from the universal joint A_{i1} to the ball joint B_i

with respect to the i_1-frame is expressed as $\boldsymbol{l}_i$. Because each leg's length vector is equal to a total of the length vector, $\boldsymbol{l}_{i1}$, of the lower segment 1 from A_{i1} to A_{i2} and the constant length vector, $\boldsymbol{l}_{i2}$, of the upper segment 2 from A_{i2} to B_{i}, the length vector of leg *i* can be written as

$$\boldsymbol{l}_i = l_i \boldsymbol{s}_i = \boldsymbol{l}_{i1} + \boldsymbol{l}_{i2} = \left(l_{i1} + l_{i2}\right)\boldsymbol{s}_i \tag{1}$$

where $\boldsymbol{s}_i$ is a unit vector along the *i*th leg axis and expressed in the i_1-frame; $\boldsymbol{l}_i$ is the length magnitude of the leg *i*; $\boldsymbol{l}_{i1}$ and $\boldsymbol{l}_{i2}$ are the respective length magnitudes of the lower segment 1 and the upper segment 2 of the leg *i*.
With multiple closed-loop chain mechanisms in a parallel robot manipulator, the geometric constraint equation on the basis of a vector loop relation can be written as

$$\boldsymbol{p} + {}_B^N\boldsymbol{R}\boldsymbol{b}_i = \boldsymbol{a}_i + {}_i^N\boldsymbol{R}\boldsymbol{l}_i \tag{2}$$

In Eq. (2), the rotation matrix ${}_L^N\boldsymbol{R}$ with the subscript $L \in \{B, i(i = 1,...,6)\}$ represents a transformation from a local L-frame to the inertial N-frame. The time derivative of a rotation matrix can be obtained by

$${}_L^N\dot{\boldsymbol{R}} = {}_L^N\boldsymbol{R}\tilde{\boldsymbol{\omega}}_L \tag{3}$$

in which $\tilde{\boldsymbol{\omega}}_L$ denotes a skew symmetric matrix formed from the angular velocity $\boldsymbol{\omega}_L = \begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^T$ with respect to a local body-fixed L-frame such as

$$\tilde{\boldsymbol{\omega}}_L = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{4}$$

Moreover, the angular velocity expressed in a body-fixed frame can be defined in a set of quasi-coordinates $\boldsymbol{\beta}_L$ that are only defined meaningfully in angular quasi-velocities such as

$$\boldsymbol{\omega}_L = \frac{d\boldsymbol{\beta}_L}{dt} \tag{5}$$

Differentiating the constraint equation, Eq. (2), yields the constraint equation in terms of velocities, the results is

$$\dot{\boldsymbol{p}} + {}_B^N\boldsymbol{R}\tilde{\omega}_B\boldsymbol{b}_i = {}_i^N\boldsymbol{R}\dot{\boldsymbol{l}}_i + {}_i^N\boldsymbol{R}\tilde{\omega}_i\boldsymbol{l}_i \tag{6}$$

where $\dot{\boldsymbol{l}}_i = \dot{\boldsymbol{l}}_{i1}$ implies that each leg's sliding velocity is equivalent to the relative sliding velocity between these two segments of the leg.

Since each leg is connected to the base with an universal joint; therefore, no spin is allowed about its longitudinal axis, it gives

$$\boldsymbol{\omega}_i^T \boldsymbol{s}_i = 0 \tag{7}$$

By pre-multiplying Eq. (6) respectively by $\left({}^N_i\boldsymbol{R}\boldsymbol{s}_i\right)^T$ and by $\tilde{s}_i$, the magnitude of the sliding velocity as well as the angular velocity of leg *i* with respect to the i_1-frame can be calculated as (Chen, 2003)

$$\dot{l}_i = \dot{l}_{i1} = \left({}^N_i\boldsymbol{R}\boldsymbol{s}_i\right)^T \dot{\boldsymbol{p}} + \left({}^N_i\boldsymbol{R}\boldsymbol{s}_i\right)^T {}^N_B\boldsymbol{R}\tilde{\boldsymbol{\omega}}_B\boldsymbol{b}_i \tag{8}$$

$$\boldsymbol{\omega}_i = \frac{1}{l_i}\tilde{\boldsymbol{s}}_i\, {}^N_i\boldsymbol{R}^T\left(\dot{\boldsymbol{p}} + {}^N_B\boldsymbol{R}\tilde{\boldsymbol{\omega}}_B\boldsymbol{b}_i\right) \tag{9}$$

Because Eq. (8) relates the velocities of the *i*th active joint to the linear velocities and angular velocities of the moving platform, the Jacobian matrix can be formulated by writing six times for each *i*=1 to 6 and assembling in a matrix form as

$$\begin{bmatrix} \dot{l}_1 \\ \vdots \\ \dot{l}_6 \end{bmatrix} = \boldsymbol{Jacob}\begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{\omega}_B \end{bmatrix} \tag{10}$$

where

$$\boldsymbol{Jacob} = \begin{bmatrix} ({}^N_1\boldsymbol{R}\boldsymbol{s}_1)^T & ({}^N_1\boldsymbol{R}\boldsymbol{s}_1)^T\, {}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_1^T \\ \vdots & \vdots \\ ({}^N_6\boldsymbol{R}\boldsymbol{s}_6)^T & ({}^N_6\boldsymbol{R}\boldsymbol{s}_6)^T\, {}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_6^T \end{bmatrix} \tag{11}$$

Using the Boltzmann-Hamel-d'Alembert formulism (Chen & Chi, 2008), (Chen & Liao, 2008), the dynamic equation of the parallel robot manipulator can be formulated in terms of the task space coordinates (i.e. $\dot{\boldsymbol{x}} = \left[\dot{\boldsymbol{p}}^T \quad \dot{\boldsymbol{\gamma}}_B^T\right]^T$, where $\boldsymbol{\gamma}_B$ is defined by a set of Euler angles specifying the orientations of the moving platform) as follows:

$$\bar{\boldsymbol{M}}_p\ddot{\boldsymbol{x}} + \bar{\boldsymbol{D}}_p\dot{\boldsymbol{x}} + \bar{\boldsymbol{G}}_p = \bar{\boldsymbol{f}}_p\,, \tag{12}$$

where $\bar{\boldsymbol{f}}_p$ is a function of the output forces of the linear actuators, $\boldsymbol{f} = \left[f_1 \quad \cdots \quad f_6\right]^T$. The matrices and vectors in Eq. (12) are fully derived in the Appendix.

For the forward dynamics analysis, the trajectories of the manipulator can be solved numerically as a function of the actuating forces and initial states. For the inverse dynamics, the actuating forces are given as a function of the positions, velocities and accelerations of the moving platform. I.e.,

$$\boldsymbol{f} = \bar{\boldsymbol{B}}^{-1}(\bar{\boldsymbol{M}}_p\ddot{\boldsymbol{x}} + \bar{\boldsymbol{D}}_p\dot{\boldsymbol{x}} + \bar{\boldsymbol{G}}_p) \tag{13}$$

where $\overline{\boldsymbol{B}} = \boldsymbol{C}_1^T (\boldsymbol{Jacob})^T$

It can be seen from Eq. (13) that a very large actuating force is required if the Jacobian matrix is rank-deficient or the value of the determinant of the Jacobian matrix approaches a singular manifold.

2.2 Fundamental structure properties

Although some fundamental structure properties of dynamic equations have been derived for open-chain manipulators in robotics, these derivations are of general nature based on the direct Lagrangian generalized coordinates formulation in an index form. In the sequel, these fundamental structural properties for the dynamics of the general parallel manipulators will be validated in a straight proof.

Lemma

For the task space dynamic equations, Eq. (12),

1. The inertia matrix $\overline{\boldsymbol{M}}_p$ is symmetric and positive definite.
2. $\dot{\overline{\boldsymbol{M}}}_p - 2\overline{\boldsymbol{C}}_p$ is a skew-symmetric matrix.

Proof:

Since the sub-matrices $\boldsymbol{M}_1$, $\boldsymbol{M}_2$ and $\boldsymbol{M}_p$ are symmetric, (see Appendix), the symmetry property of $\overline{\boldsymbol{M}}_p$ is easily seen by taking transpose on $\overline{\boldsymbol{M}}_p$ such that $\overline{\boldsymbol{M}}_p^T = \overline{\boldsymbol{M}}_p$. The kinetic energy of the parallel robot manipulator is equal to a summation of the kinetic energy of the moving platform and the six legs. Thus,

$$\text{K.E.} = \frac{1}{2}\dot{\boldsymbol{x}}^T \overline{\boldsymbol{M}}_p \dot{\boldsymbol{x}} = \frac{1}{2}\dot{\boldsymbol{v}}_1^{\mathrm{T}} \boldsymbol{M}_1 \dot{\boldsymbol{v}}_1 + \frac{1}{2}\dot{\boldsymbol{v}}_2^{\mathrm{T}} \boldsymbol{M}_2 \dot{\boldsymbol{v}}_2 \tag{14}$$

Therefore, the positive definiteness of $\overline{\boldsymbol{M}}_p$ is guaranteed by the definition that the kinetic energy of the system is zero only if it has a zero velocity.

To prove that $\dot{\overline{\boldsymbol{M}}}_p - 2\overline{\boldsymbol{C}}_p$ is a skew-symmetric matrix, we first show that $\boldsymbol{D}_1 = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_{22} \end{bmatrix}$ and $\dot{\boldsymbol{M}}_2 - 2\boldsymbol{D}_2 = \begin{bmatrix} \boldsymbol{0} & -2\boldsymbol{D}_{34} \\ -2\boldsymbol{D}_{43} & \dot{\boldsymbol{M}}_{44} - 2\boldsymbol{D}_{44} \end{bmatrix}$ are all skew-symmetric. It implies that $\boldsymbol{D}_{22}$ and $\dot{\boldsymbol{M}}_{44} - 2\boldsymbol{D}_{44}$ are required to be skew-symmetric matrices as well as $\boldsymbol{D}_{34}^T = -\boldsymbol{D}_{43}$. The properties that $\boldsymbol{D}_{22}^T = -\boldsymbol{D}_{22}$ and $\boldsymbol{D}_{34}^T = -\boldsymbol{D}_{43}$ are quite clear from the notation of these sub-matrices, (see Appendix). We then calculate

$$\dot{\boldsymbol{M}}_{44} - 2\boldsymbol{D}_{44} = \boldsymbol{diag}\left[\boldsymbol{E}_1 \quad ,\dots, \quad \boldsymbol{E}_6\right] \tag{15}$$

where $\boldsymbol{E}_i = m_{i2}\dot{l}_{i1}(\tilde{\boldsymbol{s}}_i^T \tilde{\boldsymbol{d}}_{i2} - \tilde{\boldsymbol{d}}_{i2}^T \tilde{\boldsymbol{s}}_i) - 2\overline{(\boldsymbol{I}_{i1} + \boldsymbol{I}_{i2})\boldsymbol{\omega}_i}^T - 2m_{i2}l_{i1}(\boldsymbol{\omega}_i^T \boldsymbol{d}_{i2})\tilde{\boldsymbol{s}}_i$, $\qquad i = 1,\dots,6$

After taking transpose it is shown that $(\dot{\boldsymbol{M}}_{44} - 2\boldsymbol{D}_{44})^T = -(\dot{\boldsymbol{M}}_{44} - 2\boldsymbol{D}_{44})$. So it is concluded that both $\boldsymbol{D}_1$ and $\dot{\boldsymbol{M}}_2 - 2\boldsymbol{D}_2$ are symmetric matrices.

Next, it will be shown that $\dot{\boldsymbol{M}}_p - 2\boldsymbol{D}_p$ is skew-symmetric. We obtain

$$\dot{\boldsymbol{M}}_p - 2\boldsymbol{D}_p = -2\boldsymbol{D}_1 + \boldsymbol{J}_1^T\boldsymbol{J}_2^{-T}(\dot{\boldsymbol{M}}_2 - \boldsymbol{D}_2)\boldsymbol{J}_2^{-1}\boldsymbol{J}_1 + \frac{d}{dt}(\boldsymbol{J}_1^T\boldsymbol{J}_2^{-T})\boldsymbol{M}_2\left(\boldsymbol{J}_2^{-1}\boldsymbol{J}_1\right) - \boldsymbol{J}_1^T\boldsymbol{J}_2^{-T}\boldsymbol{M}_2\frac{d}{dt}\left(\boldsymbol{J}_2^{-1}\boldsymbol{J}_1\right) \quad (16)$$

Now that the first two terms of the right hand side are the sum of skew-symmetric matrices, by taking transpose and using symmetry of $\boldsymbol{M}_2$, $(\dot{\boldsymbol{M}}_p - 2\boldsymbol{D}_p)^T = -(\dot{\boldsymbol{M}}_p - 2\boldsymbol{D}_p)$ is proved.

Finally, $\dot{\bar{\boldsymbol{M}}}_p - 2\bar{\boldsymbol{C}}_p$ is calculated as

$$\dot{\bar{\boldsymbol{M}}}_p - 2\bar{\boldsymbol{D}}_p = \boldsymbol{C}_1^T(\dot{\boldsymbol{M}}_p - 2\boldsymbol{D}_p)\boldsymbol{C}_1 + \dot{\boldsymbol{C}}_1\boldsymbol{M}_p\boldsymbol{C}_1^T - \boldsymbol{C}_1^T\boldsymbol{M}_p\dot{\boldsymbol{C}}_1 \quad (17)$$

Because the first term of the right hand side has been proven as a skew-symmetric matrix, after taking transpose and inverting the sign, the skew-symmetric property of $\dot{\bar{\boldsymbol{M}}}_p - 2\bar{\boldsymbol{C}}_p$ for the task space dynamic equations is verified.

3. Singularity-Free Paths Programming

3.1 Cost function and constraints

The objective here is to determine a path completely within the workspace with the following cost functions to be minimized while moving from the configuration $\boldsymbol{x}_0$ to the final configuration $\boldsymbol{x}_f$ during the time interval from t_0 to t_f:

(1) Minimum actuating force

$$G = \int_{t_0}^{t_f} \left(\sum_{l=1}^{6} |f_l|\right) dt \quad (18)$$

(2) Time optimum

$$G = \Delta t = t_f - t_0 \quad (19)$$

(3) Energy efficiency

$$G = \int_{t_0}^{t_f} \left|\boldsymbol{f}^T\dot{\boldsymbol{l}}\right| dt \quad (20)$$

(4) Mixed cost function

A cost function can be formulated by mixing Eqs.(18) & (20), or Eqs. (19) & (20) using a weighting coefficient whose value is in the range of zero and one. The weighting coefficient weights these two functions according to the relative importance.

In addition, the considered trajectories in the task space must satisfy the constraints and conditions as follows:

(1)Initial and final conditions:

$\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ with the associated boundary conditions

$$\dot{x}(t_o) = \dot{x}(t_f) = \ddot{x}(t_o) = \ddot{x}(t_f) = 0 \tag{21}$$

(2)Leg length constraints

$$l_{min} \le l_i(x) \le l_{max} \ \text{ for } i\text{=1,…,6 and } \ t_o \le t \le t_f \tag{22}$$

(3)Leg's linear velocity constraints

$$\left|\dot{l}(x,\dot{x})\right| \le \dot{l}_{max} \quad \text{ for } i\text{=1,…,6 and } t_o \le t \le t_f \tag{23}$$

(4)Leg's linear acceleration constraints

$$\left|\ddot{l}_i(\mathbf{x},\dot{\mathbf{x}},\ddot{\mathbf{x}})\right| \le \ddot{l}_{max} \quad \text{ for } i\text{=1,…,6 and } t_o \le t \le t_f \tag{24}$$

(5)Actuating force constraints

$$\left|f_i(\mathbf{x},\dot{\mathbf{x}},\ddot{\mathbf{x}})\right| \le f_{max} \tag{25}$$

Note that Eq. (25) restricts the output forces so that the planned path is implicitly singularity-free.
Typically, the singularity-free path programming with the minimum cost function is a constrained optimization problem. A suitable path should be selected so that the considered cost function subject to the conditions and constraints, Eqs. (21)-(25), is minimized.

3.2 Geometric path representation

A smooth spatial path can be generated by control points of path function. Because the B-spline function provides a simple method to create curves between the defined points, therefore, it is used for the path function. The trajectory x at each time sub-interval $t_i \le t \le t_{i+1}$ (i=0, 1,…,n-1) can be interpolated as

$$x\{t(u)\} = \boldsymbol{UN}\hat{\boldsymbol{X}}_i \tag{26}$$

where $t(u) = t_i + u\Delta t\,, \quad 0 \le u \le 1$

$$\Delta t = (t_f - t_0)/n$$

$$\boldsymbol{U} = \begin{bmatrix} u^3 & u^2 & u^1 & 1 \end{bmatrix}$$

$$\hat{\boldsymbol{X}}_i = [\ \hat{x}_{i-1}^T \ \ \hat{x}_i^T \ \ \hat{x}_{i+1}^T \ \ \hat{x}_{i+2}^T \ ]^T$$

and $\hat{x}_k^T$ (k=-1, 0, …, n, n+1) are called the control points and total (n+3) points. u is the independent spline parameter, $\boldsymbol{N}$ defines the shape of spline, being expressed for the cubic B-spline as

$$\boldsymbol{N}=\frac{1}{6}\begin{bmatrix}-1 & 3 & -3 & 1\\ 3 & 6 & 3 & 0\\ -3 & 0 & 3 & 0\\ 1 & 4 & 1 & 0\end{bmatrix}$$

The associated velocities and accelerations along the trajectory are obtained by subsequent derivatives of $\boldsymbol{x}$ as

$$\dot{x}\{t(u)\}=\boldsymbol{U}'\boldsymbol{N}\hat{\boldsymbol{X}}_i\,/\,\Delta t \tag{27}$$

$$\ddot{x}\{t(u)\}=\boldsymbol{U}''\boldsymbol{N}\hat{\boldsymbol{X}}_i\,/(\Delta t)^2\,,\quad t_i\le t\le t_{i+1} \tag{28}$$

where the prime is denoted as the derivative with respect to the slpine parameter u. Also, from the boundary condition, Eq. (21), the first three control points and the last three are fixed as

$$\hat{\boldsymbol{x}}_{-1}=\hat{\boldsymbol{x}}_0=\hat{\boldsymbol{x}}_1=\boldsymbol{x}_0 \tag{29}$$

$$\hat{\boldsymbol{x}}_{n-1}=\hat{\boldsymbol{x}}_n=\hat{\boldsymbol{x}}_{n+1}=\boldsymbol{x}_f \tag{30}$$

Thus, the undetermined control points are reduced to $\hat{x}_2,\ldots,\ \hat{x}_{n-2}$. By substituting the terms x, $\dot{x}$ and $\ddot{x}$ into Eq. (13), then into the cost function as well as the constraint Eqs. (22)-(25), the constraint optimization problem can be recast into an unconstrained problem by adding weighted penalty functions to form a pseudo cost function, the pseudo cost function can be expressed as

$$\overline{G}=\sum_{i=0}^{n-1}\Delta t\int_0^1 F(u,\hat{\boldsymbol{X}}_i)du+\sum_{j=1}^{6}\{w_1TRUE(l_{min}-l_{jmin}(\hat{\boldsymbol{x}},u))+\ w_2TRUE(l_{jmax}(\hat{\boldsymbol{x}},u)-l_{max})$$
$$+\,w_3TRUE(\left|\dot{l}_{jmax}(\hat{\boldsymbol{x}},u)\right|-\dot{l}_{max}))+w_4TRUE(\left|\ddot{l}_{jmax}(\hat{\boldsymbol{x}},u)\right|-\ddot{l}_{max})+w_5TRUE(\left|f_{jmax}(\hat{\boldsymbol{x}},u)\right|-f_{max})\} \tag{31}$$

in which the kernel function $F(u,\hat{\boldsymbol{X}}_i)$ is associated with the cost function. In the chapter, $F=\sum_{l=1}^{6}\left|f_l\right|$, $F=1$ and $F=\left|\boldsymbol{f}^T\dot{\boldsymbol{l}}\right|$ are respectively for the minimum actuating force problem, the time optimal problem, and the energy efficiency problem. The constraints are also mapped into the function of the spline parameter u and the set of control points $\hat{x}_k$ (k=-1, 0, …, n, n+1). In addition, $TRUE$ is a logical operational function with the value one if the statement is true and zero if the statement is false; $w_1,\ldots,w_5$ are the weighting factors that determine the magnitude of the penalty and should be taken large enough so that any constraint violation will be penalized with large cost.

By the transformation of the constrained optimization problem, a singularity-free path programming is equivalent to the determination of a set of control points for the minimization of the pseudo cost function. Although sufficient control points can generate any smooth spatial curve, the required computational load for the optimization will be increased dramatically.

3.3 Global search based on PSOA

In order to solve the highly nonlinear optimization problem described by the pseudo cost function, an evolutionary computation based optimization technique, PSOA mimicking the food-searching behavior of birds is employed (Kennedy & Eberhart, 1995).
The process of PSOA is described as following:
(1) Randomly initialize position $\hat{\boldsymbol{X}}_p = [\ \hat{\boldsymbol{x}}_{p2}^T\ ,\ldots,\ \hat{\boldsymbol{x}}_{p,n-2}^T\ ]^T$, namely, undetermined control points, and velocity $\hat{\boldsymbol{V}}_p = [\ \hat{\boldsymbol{v}}_{p2}^T\ ,\ldots,\ \hat{\boldsymbol{v}}_{p,n-2}^T\ ]^T$, p=1,…,P, in which P=40 particles is set.
(2) Formulate the actuating forces through solving the inverse dynamics Eq. (13), and then a numerical integral based on the Simpson's rule is used to calculate the integral of the pseudo cost function $\overline{G}_p$ in Eq. (31) for each particle.
(3) Compare the current fitness value $\overline{G}_p$ with the particle's previous best value $\overline{G}_{pb}$, if $\overline{G}_p < \overline{G}_{pb}$, then $\boldsymbol{Pb}_p = \hat{\boldsymbol{X}}_p$, p=1,…, P, in which $\boldsymbol{pb}_p = [\boldsymbol{pb}_{p2}^T \ldots \boldsymbol{pb}_{p,n-2}^T]^T$ is the position of each particle with its personal best fitness value $\overline{G}_p$.
(4) Compare the current fitness value $\overline{G}_p$ with the group's previous best value $\overline{G}_{gb}$, if $\overline{G}_p < \overline{G}_{gb}$, then $\boldsymbol{gb} = \hat{\boldsymbol{X}}_p$, p=1,…, P, in which $\boldsymbol{gb} = [\boldsymbol{gb}_2^T \ldots \boldsymbol{gb}_{n-2}^T]^T$ is the only one that has the global best fitness value $\overline{G}_{gb}$ thus far.
(5) Update each particle according to PSOA,

$$v_{pj} = v_{pj} + c_1 rand(\ )\left(pb_{pj} - \hat{x}_{pj}\right) + c_2 Rand(\ )\left(gb_j - \hat{x}_{pj}\right),\ j=1,\ldots,6*(n-3) \qquad (32)$$

$$\hat{x}_{pj} = \hat{x}_{pj} + v_{pj} \qquad (33)$$

where v_{pj} is the rate of the position change for particle p with respect to the jth dimension and clamped to the range $[-v_{max}, v_{max}]$ with $v_{\max}$ =0.8 to prevent the likelihood of the particle from leaving the search space; c_1 and c_2 are two positive constants, in which $c_1 = c_2$ =1.5 are set to control the step size that a particle will move in the direction of best position; *rand*( ) and *Rand*( ) are two normal distribution functions. $\hat{x}_{pj}$ is the position of the pth particle with respect to the jth dimension.
(6) If a convergence criterion is met, it means that all the particles have been toward the global best and its own individual best. Otherwise, repeat the steps starting from step (2) until it has reached the maximum allowable iteration number. The maximum iteration number is 350. Furthermore, the relative change in the group's best pseudo objective function $\overline{G}_{gb}$ for two consecutive iterations being less than 0.01 is defined as the convergence criterion for the optimization process.

4. Numerical Simulations

4.1 Kinematic Parameters

The optimal singularity-free path programming with the minimum actuating force for the parallel robot manipulator is implemented in MATLAB routines. The kinematic parameters

used for the simulations are given below,
The platform points in the B-frame:

$$\begin{bmatrix} B_1 & B_2 & B_3 & B_4 & B_5 & B_6 \end{bmatrix} = \begin{bmatrix} 0.3 & 0.3 & 0.0 & -0.2 & -0.15 & 0.15 \\ 0.0 & 0.2 & 0.3 & 0.1 & -0.2 & 0.15 \\ 0.1 & 0.0 & 0.0 & -0.1 & -0.05 & -0.05 \end{bmatrix}$$

The base points in the N-frame:

$$\begin{bmatrix} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.1 & -0.3 & -0.3 & 0.2 & 0.5 \\ 0.2 & 0.5 & 0.3 & -0.4 & -0.3 & -0.2 \\ 0.0 & 0.1 & 0.0 & 0.0 & -0.05 & 0.0 \end{bmatrix}$$

All legs are considered to be identical. The masses of lower segment 1 and upper segment 2 of each leg are respective $m_{i1} = 2\text{kg}$ and $m_{i2} = 1\text{kg}$, and the mass of the moving platform is $m_B = 32\text{kg}$. The unit vector $\boldsymbol{s}_i$ expressed in the i_1-frame is given by $\boldsymbol{s}_i = [1 \quad 0 \quad 0]^T$. The gravity centers of part 1 and part 2 of each leg $\boldsymbol{d}_{i1} = \boldsymbol{d}_{i2} = [0.05 \quad 0.001 \quad 0.001]^T$ m. The constant length of the upper segment 2 is $\boldsymbol{l}_{i2}$ =1.0 m. The moments of inertia of these two segments of each leg as well as the moving platform about their respective gravity centers in local frame are ${}^C\boldsymbol{I}_{i1} = \boldsymbol{diag}\ [0.001,\ 1,\ 1]\ \text{kg}\cdot\text{m}^2$, ${}^C\boldsymbol{I}_{i2} = \boldsymbol{diag}\ [0.0001,\ 0.4,\ 0.4]\ \text{kg}\cdot\text{m}^2$, $\boldsymbol{I}_{\mathbf{B}} = \boldsymbol{diag}\ [2, 2, 4]\ \text{kg}\cdot\text{m}^2$. The moments of inertia of the segments 1 and 2 of each leg about the respective joint points A_{i1} and A_{i2} in local frame can be calculated using parallel axes theorem as $\boldsymbol{I}_{i1} = {}^C\boldsymbol{I}_{i1} + m_{i1}\tilde{\boldsymbol{d}}_{i1}^T\tilde{\boldsymbol{d}}_{i1}$, and $\boldsymbol{I}_{i2} = {}^C\boldsymbol{I}_{i2} + m_{i2}\tilde{\boldsymbol{d}}_{i2}^T\tilde{\boldsymbol{d}}_{i2}$. The vector of gravity is $\boldsymbol{g} = [0 \quad 0 \quad -9.81]^T\ \text{m}/\text{sec}^2$.

The Euler angles vector $\boldsymbol{\gamma} = [\psi \quad \theta \quad \phi]^T$ is used to specify the orientation of the platform and each leg, which means that the inertial frame is rotated about the Z axis with ψ radians first, resulting in the primed frame; then about the y′ axis with θ radians, resulting in the double primed frame; and finally about the x″ axis with ϕ radians, resulting in the local frame. Therefore, the rotation matrix of the local frame relative to the inertial frame is written by

$$\boldsymbol{R} = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi s\theta c\phi \\ s\psi c\theta & c\psi c\phi + s\psi s\theta s\phi & -c\psi s\phi + s\psi s\theta c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{34}$$

where $c(\bullet)$ is a shorthand notation for $cos(\bullet)$ and $s(\bullet)$ for $sin(\bullet)$. The velocity transformation matrix $\boldsymbol{C}_L$ converting the angular velocity in terms of the time derivative of the Euler angles to the one with respect to a body-fixed frame is given as

$$\boldsymbol{C}_L = \begin{bmatrix} -s\theta & 0 & 1 \\ c\theta s\phi & c\phi & 0 \\ c\theta c\phi & -s\phi & 0 \end{bmatrix} \tag{35}$$

where the subscript $L \in \{B, i(i=1,...,6)\}$.
The path planning algorithm presented in the above steps is demonstrated by the following simulations. The desired initial pose x_0 and final pose x_f for the parallel robot manipulator are $\boldsymbol{x}_0 = \begin{bmatrix} 1.6 & -0.2 & -0.6 & 0.0 & 0.0 & 0.0 \end{bmatrix}^T$, $\boldsymbol{x}_f = \begin{bmatrix} -0.8 & 0.1 & -0.4 & 0.0 & 0.0 & 0.0 \end{bmatrix}^T$.
The straight path connecting these two end-poses and keeping constant orientation is through a singular manifold as is shown by the zero determinant of the Jacobian in Fig 3. Then, optimal singularity-free paths will be sought between x_0 and x_f during the time interval subject to each leg's length constraints, $l_{min} = 0.2\text{m}$, $l_{max} = 3.0\text{m}$, $\dot{l}_{max} = 0.8\text{m/sec}$, $\ddot{l}_{max}$ =1.0 m/sec^2 and f_{max} =2500N.

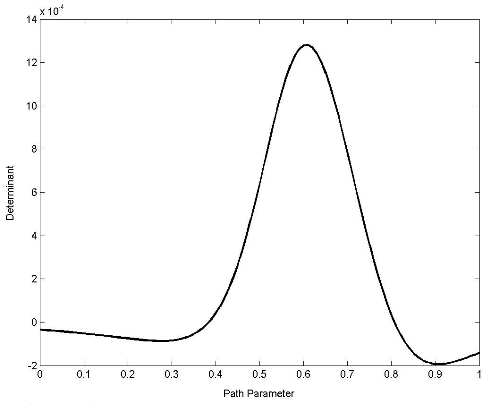


Fig. 3. Determinant in straight path under constant orientation

Also, the weighting factors $w_1,...,w_5$ serving as the magnitude of the penalty should be chosen appropriately only when the constraints are violated; because a small value may yield major constraint violations, but a large value will result in a very poor optimization problem. Thus, w_1, w_2 and w_3 are taken as 5×10^5 and $w_4 = w_5 = 5\times10^4$ by a few trial simulations.
Finally, singularity-free paths are programmed by eight control points, i.e., two undetermined control points among them. More control points could imply a more flexible search for the optimal singularity-free paths, but with increased control points, it will cost more computation time for convergence.

4.2 Results and Discussions

According to the aforementioned parameters, the following examples will be illustrated to determine the optimal singularity-free paths :

(1)Minimum actuating force

The paths programmed for the minimum actuating forces considering the constant orientations and varied orientations of the moving platform during the time interval t_f =20sec are shown in Fig. 4. Fig. 5 is the orientation variations of the moving platform while traveling along the planned path with varied orientations. It is seen that these two planned paths are quite distinct.

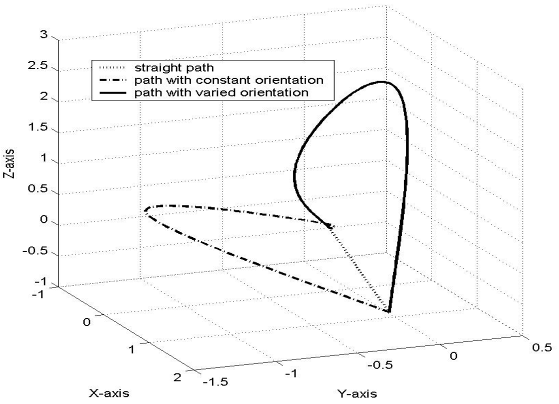


Fig. 4. Singularity-free planned paths for minimum actuating forces

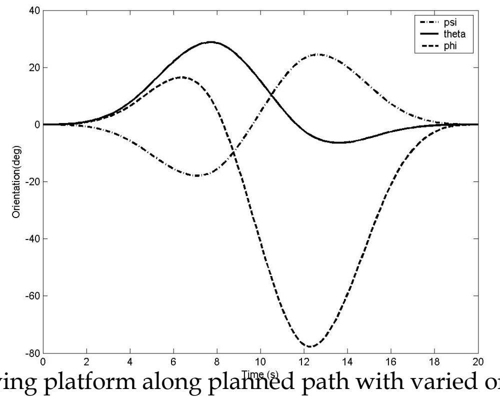


Fig. 5. Orientation of moving platform along planned path with varied orientations

Fig. 6 depicts the variations of the determinant along the corresponding paths. As is shown, the determinants are never equal to zero for the paths, the obtained paths based on PSOA successfully avoid singularities. Moreover, the difference of the evaluated values along the two obtained paths is of one order. The result is reflected on the time history plot of each leg's actuating forces as shown in Figs. 7(a), (b). It is observed that the actuating forces for the actuators 2 and 3 reach their largest values at t=6 sec along the path with a constant orientation, this implies that the robot manipulator is the nearest to a singular manifold for the pose.

Although the actuating forces can be decreased by varying the orientations of the moving platform, the required maximum leg lengths increase as indicated in Figs. 8(a), (b). It means that a larger task space is necessary to accommodate the planned singularity-free path.

(2) Time optimum

For this problem, the travel time t_f is to be determined. Based on the singularity-free path planning algorithm, the planned trajectory is shown in Fig.9 (i.e. the line for μ =1) with the corresponding minimal travel time t_f =5.85 sec.

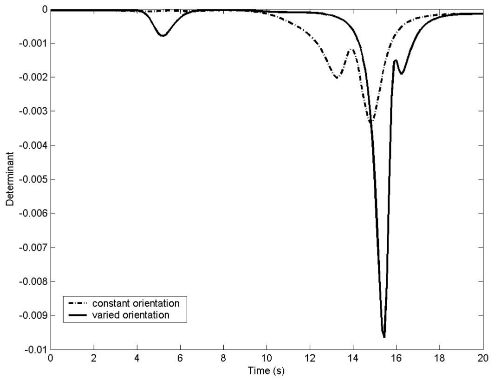


Fig. 6. Variations of determinant along corresponding planned paths

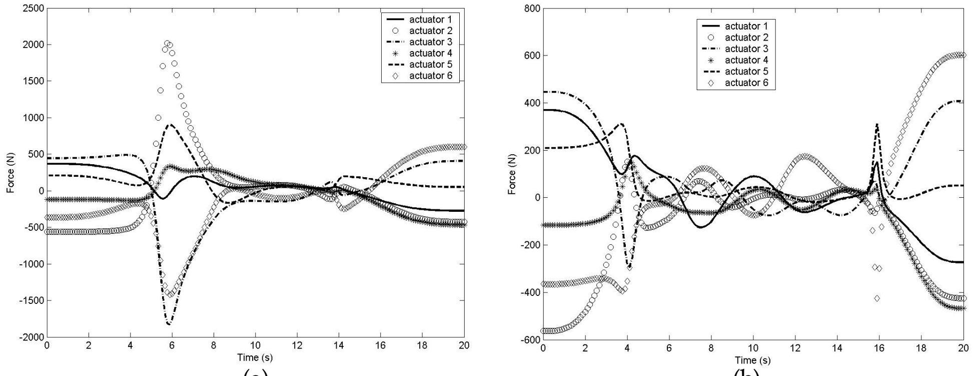


Fig. 7. Actuating forces along planned paths with (a) constant orientation and (b) varied orientations

(3) Energy efficiency

Fig. 9 also shows the minimal-energy trajectory with the corresponding travel time t_f =20.01 sec (i.e. the line for μ =0). Compared with the time optimal trajectory planning, reduction in the travel time is at the expense of a greater consumed energy, a poorer fitness value, and a larger force.

(4) Mixed cost function

The cost function is defined as

$$G=\mu\lambda(\Delta t)+(1-\mu)\int_{t_0}^{t_f}\left|f^T\dot{l}\right|dt \tag{36}$$

The optimal singular free trajectories for μ =0.3, 0.6 and 0.8 with the corresponding determined travel time t_f =7.758, 6.083 and 6.075 sec are also respectively shown in Fig. 9.

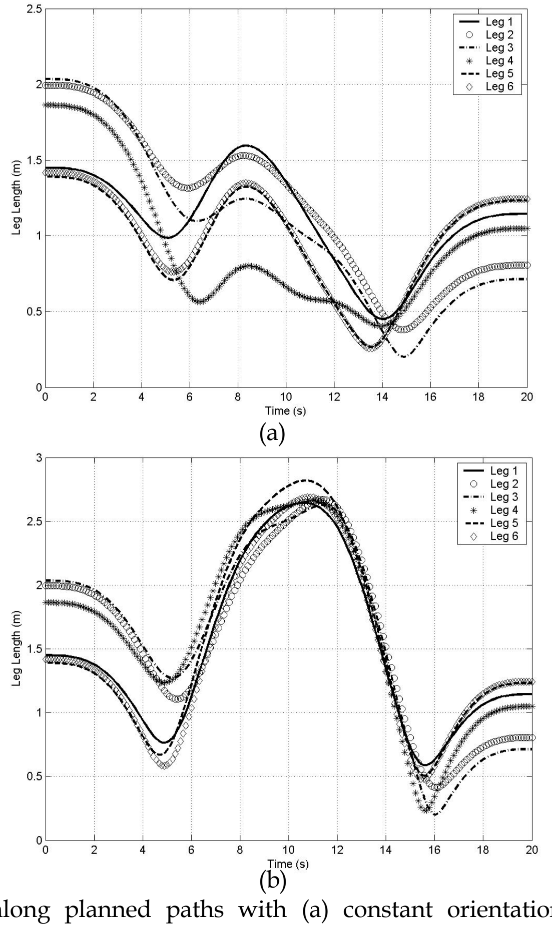


Fig. 8. Leg lengths along planned paths with (a) constant orientation and (b) varied orientations

5. Conclusions

In this chapter, a numerical technique is presented to determine the singularity-free trajectories of a parallel robot manipulator. The required closed-form dynamic equations for the parallel manipulator with a completely general architecture and inertia distribution are developed systematically using the new structured Boltzmann-Hamel-d'Alembert approach, and some fundamental structural properties of the dynamics of parallel manipulators are validated in a straight proof.

In order to plan a singularity-free trajectory subject to some kinematic and dynamic constraints, the parametric path representation is used to convert a planned trajectory into the determination of a set of undetermined control points in the workspace. With a highly nonlinear expression for the constrained optimal problem, the PSOA needing no differentiation is applied to solve for the optimal control points, and then the corresponding trajectories are generated. The numerical results have confirmed that the obtained singularity-free trajectories are feasible for the minimum actuating force problem, time optimal problem, energy efficient problem and mixed optimization problem. The generic nature of the solution strategy presented in this chapter makes it suitable for the trajectory planning of many other configurations in the parallel robot manipulator domain and suggests its viability as a problem solver for optimization problems in a wide variety of research and application fields.

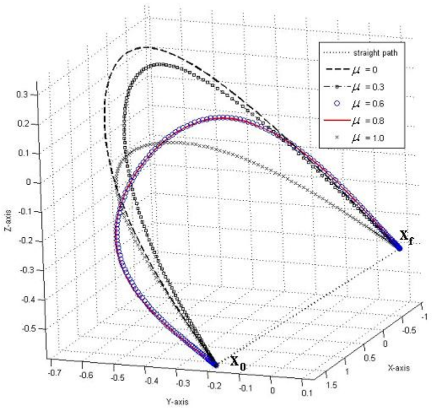


Fig. 9. Planned paths for time optimum, energy efficiency and mixed cost function

6. References

Bhattacharya, S.; Hatwal, H. & Ghosh, A. (1998). Comparison of an exact and an approximate method of singularity avoidance in platform type parallel manipulator. *Mech. Mach. Theory*, **33**, 965-974.

Chen, C.T. (2003). A Lagrangian formulation in terms of quasi-coordinates for the inverse dynamics of the general 6-6 Stewart platform manipulator. *JSME International J. Series C,* **46**, 1084-1090.

Chen, C.T. & Chi, H.W. (2008). Singularity-free trajectory planning of platform-type parallel manipulators for minimum actuating effort and reactions. *Robotica,* **26(3)**, 357-370.

Chen, C.T. & Liao, T.T. (2008). Optimal path programming of the SPM using the Boltzmann-Hamel-d'Alembert dynamics formulation model. Adv Robot, **22(6)**, 705–730.

Dasgupta, B. & Mruthyunjaya, T.S. (1998). Singularity-free planning for the Stewart platform manipulator. *Mech. Mach. Theory,* **33**, 771-725.

Dasgupta, B. & Mruthyunjaya, T.S. (1998). Closed-form dynamic equations of the general Stewart platform through the Newton-Euler approach. *Mech. Mach. Theory,* **33**, 993-1012.

Dasgupta, B. & Mruthyunjaya, T.S. (1998). A Newton-Euler formulation for the Inverse dynamics of the Stewart platform manipulator. *Mech. Mach. Theory,* **33**, 1135-1152.

Do, W. & Yang, D. (1998). Inverse dynamic analysis and simulation of a platform type of robot. *J. Robot. Syst.*, **5**, 209-227.

Dash, A.K.; Chen, I.; Yeo, S. & Yang, G. (2005). Workspace generation and planning singularity-free path for parallel manipulators. *Mech. Mach. Theory,* **40**, 776-805.

Kennedy, J. & Eberhart, R. (1995). Particle swarm optimization, in *Proc. of IEEE Int. Conf. on Neural Network,* pp. 1942-1948, Australia.

Lebret, G.; Liu, K. & Lewis, F.L. (1993). Dynamic analysis and control of a Stewart platform manipulator. *J. Robot. Syst.*, **10**, 629-655.

Liu, M.J.; Li, C.X. & Li, C.N. (2000). Dynamics analysis of the Gough-Stewart platform manipulator. *IEEE Trans. Robot. Automat.,* **16**, 94-98.

Nakamura, Y. & Ghodoussi, M. (1989). Dynamics computation of closed-link robot mechanisms with nonredundant and redundant actuators, *IEEE Transactions on Robotics and Automation,* **5**, 294-302.

Nakamura, Y. & Yamane, K. (2000). Dynamics computation of structure-varying kinematic chains and its application to human figures. *IEEE Trans. Robotics and Automation,* 16, 124-134.

Pang, H. & Shahinpoor, M. (1994). Inverse dynamics of a parallel manipulator. *J. Robot. Syst.,* **11**, 693-702.

Sen, S.; Dasgupta, B. & Bhattacharya, S. (2003). Variational approach for singularity-free path-planning of parallel manipulators. *Mech. Mach. Theory,* **38**, 1165-1183.

Tsai, L.W. (2000). Solving the inverse dynamics of a Stewart-Gough manipulator by the principle of virtual work. *Trans. ASME J. Mech. Design,* **122,** 3-9.

Wang, S.C.; Hikita, H.; Kubo, H.; Zhao, Y.S.; Huang, Z. & Ifukube, T. (2003). Kinematics and dynamics of a 6 degree-of-freedom fully parallel manipulator with elastic joints. *Mech. Mach. Theory,* **38**, 439-461.

Wang, J. & Gosselin, C.M. (1998). A new approach for the dynamic analysis of parallel manipulators. *Multibody System Dynamics,* **2**, 317-334.

Zhang, C. & Song, S. (1993). An efficient method for inverse dynamics of manipulators based on the virtual work principle. *J. Robot. Syst.,* **10**, 605-627.

Acknowledgments

This work is supported by the National Science Council of the ROC under the Grant NSC 98-2221-E-212 -026 and NSC 98-2221-E-269-015.

Appendix

■ Dynamic equation of general parallel robot manipulators

$$\bar{\boldsymbol{M}}_p\ddot{\boldsymbol{x}} + \bar{\boldsymbol{D}}_p\dot{\boldsymbol{x}} + \bar{\boldsymbol{G}}_p = \bar{\boldsymbol{f}}_p \tag{A1}$$

where $\bar{\boldsymbol{M}}_p = \boldsymbol{C}_1^T(\boldsymbol{M}_1 + \boldsymbol{J}_1^T\boldsymbol{M}_2\boldsymbol{J}_1)\boldsymbol{C}_1 = \boldsymbol{C}_1^T\boldsymbol{M}_p\boldsymbol{C}_1$

$$\bar{\boldsymbol{D}}_p = \boldsymbol{C}_1^T\left((\boldsymbol{M}_1 + \boldsymbol{J}_1^T\boldsymbol{M}_2\boldsymbol{J}_1)\dot{\boldsymbol{C}}_1 + (\boldsymbol{D}_1 + \boldsymbol{J}_1^T\boldsymbol{D}_2\boldsymbol{J}_1 + \boldsymbol{J}_1^T\boldsymbol{M}_2\dot{\boldsymbol{J}}_1)\boldsymbol{C}_1\right) = \boldsymbol{C}_1^T\left((\boldsymbol{M}_1 + \boldsymbol{J}_1^T\boldsymbol{M}_2\boldsymbol{J}_1)\dot{\boldsymbol{C}}_1 + \boldsymbol{D}_p\boldsymbol{C}_1\right)$$

$$\bar{\boldsymbol{G}}_p = \boldsymbol{C}_1^T(\boldsymbol{G}_1 + \boldsymbol{J}_1^T\boldsymbol{G}_2)$$

$$\bar{\boldsymbol{f}}_p = \boldsymbol{C}_1^T(\boldsymbol{Jacob})^T\boldsymbol{f}$$

The actuating forces vector $\boldsymbol{f} = \begin{bmatrix} f_1 & \dots & f_6 \end{bmatrix}^T$

In (A1), the velocity transformation matrix, $\boldsymbol{C}_1$, is defined as

$$\boldsymbol{C}_1 = \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_B \end{bmatrix} \tag{A2}$$

where $\boldsymbol{C}_B$ is the angular velocity transformation of the moving platform. In addition, $\boldsymbol{J}_1$ and $\boldsymbol{J}_2$ are the sub-matrices appropriately partitioned while developing the equations of motion, and are defined as

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_1 \mid \boldsymbol{J}_2 \end{bmatrix} = \left[\begin{array}{cc:c} ({}^N_1\boldsymbol{R}\boldsymbol{s}_1)^T & ({}^N_1\boldsymbol{R}\boldsymbol{s}_1)^T\,{}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_1^T & \\ \vdots & \vdots & \\ ({}^N_6\boldsymbol{R}\boldsymbol{s}_6)^T & ({}^N_6\boldsymbol{R}\boldsymbol{s}_6)^T\,{}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_6^T & \\ \frac{1}{l_1}\tilde{\boldsymbol{s}}_1\,{}^N_1\boldsymbol{R}^T & (\frac{1}{l_1}\tilde{\boldsymbol{s}}_1\,{}^N_1\boldsymbol{R}^T)\,{}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_1^T & -\boldsymbol{I}_{24\times24} \\ \vdots & \vdots & \\ \frac{1}{l_6}\tilde{\boldsymbol{s}}_6\,{}^N_6\boldsymbol{R}^T & (\frac{1}{l_6}\tilde{\boldsymbol{s}}_6\,{}^N_6\boldsymbol{R}^T)\,{}^N_B\boldsymbol{R}\tilde{\boldsymbol{b}}_6^T & \end{array}\right] \tag{A3}$$

in which $\boldsymbol{I}_{n\times n}$ is an $\mathrm{n}\times\mathrm{n}$ unitary matrix such that $\boldsymbol{J}_2 = -\boldsymbol{I}_{24\times24}$.

■ Inertia matrix, Coriolis and centrifugal terms, gravity vector

$$\boldsymbol{M}_1 = \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M}_{22} \end{bmatrix}, \quad \boldsymbol{M}_2 = \begin{bmatrix} \boldsymbol{M}_{33} & \boldsymbol{M}_{34} \\ \boldsymbol{M}_{43} & \boldsymbol{M}_{44} \end{bmatrix} \tag{A4}$$

where $\boldsymbol{M}_{11} = m_B\boldsymbol{I}_{3\times3}$

$$\boldsymbol{M}_{22} = \boldsymbol{I}_B$$

$$\boldsymbol{M}_{33} = \boldsymbol{diag}\left[m_{12},\dots,m_{62}\right]$$

$$\boldsymbol{M}_{34} = \boldsymbol{diag}\left[m_{12}\boldsymbol{s}_1^T\tilde{\boldsymbol{d}}_{12}^T \quad ,\dots, \quad m_{62}\boldsymbol{s}_6^T\tilde{\boldsymbol{d}}_{62}^T\right]$$

$$\boldsymbol{M}_{43} = \boldsymbol{diag}\left[m_{12}\tilde{\boldsymbol{d}}_{12}\boldsymbol{s}_1 \quad ,\dots, \quad m_{62}\tilde{\boldsymbol{d}}_{62}\boldsymbol{s}_6\right]$$

$$\boldsymbol{M}_{44} = \boldsymbol{diag}\left[\boldsymbol{I}_1 \quad ,\dots, \quad \boldsymbol{I}_6\right]$$

and $\boldsymbol{I}_i = \boldsymbol{I}_{i1} + \boldsymbol{I}_{i2} - m_{i2}l_{i1}^2\tilde{\boldsymbol{s}}_i^2 + m_{i2}l_{i1}\left(\tilde{\boldsymbol{s}}_i^T\tilde{\boldsymbol{d}}_{i2} + \tilde{\boldsymbol{d}}_{i2}^T\tilde{\boldsymbol{s}}_i\right), \qquad i = 1,\dots,6$

$$\boldsymbol{D}_1 = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_{22} \end{bmatrix}, \boldsymbol{D}_2 = \begin{bmatrix} \mathbf{0} & \boldsymbol{D}_{34} \\ \boldsymbol{D}_{43} & \boldsymbol{D}_{44} \end{bmatrix} \tag{A5}$$

where $\boldsymbol{D}_{22} = \widetilde{\boldsymbol{I}_B \boldsymbol{\omega}_B}^T$

$$\boldsymbol{D}_{34} = \boldsymbol{diag}\left[m_{12}\boldsymbol{\omega}_1^T \tilde{\mathbf{s}}_1 (l_{11}\tilde{\mathbf{s}}_1 + \tilde{\boldsymbol{d}}_{12}) \quad ,\ldots, \quad m_{62}\boldsymbol{\omega}_6^T \tilde{\mathbf{s}}_6 (l_{61}\tilde{\mathbf{s}}_6 + \tilde{\boldsymbol{d}}_{62}) \right]$$

$$\boldsymbol{D}_{43} = \boldsymbol{diag}\left[m_{12}(l_{11}\tilde{\mathbf{s}}_1^T + \tilde{\boldsymbol{d}}_{12}^T)\tilde{\mathbf{s}}_1 \boldsymbol{\omega}_1 \quad ,\ldots, \quad m_{62}(l_{61}\tilde{\mathbf{s}}_6^T + \tilde{\boldsymbol{d}}_{62}^T)\tilde{\mathbf{s}}_6 \boldsymbol{\omega}_6 \right]$$

$$\boldsymbol{D}_{44} = \boldsymbol{diag}\left[\boldsymbol{h}_1 \quad ,\ldots, \quad \boldsymbol{h}_6 \right]$$

and $\boldsymbol{h}_i = m_{i2}\dot{l}_{i1}(l_{i1}\tilde{\mathbf{s}}_i^T + \tilde{\boldsymbol{d}}_{i2}{}^T)\tilde{\mathbf{s}}_i + \widetilde{(\boldsymbol{I}_{i1} + \boldsymbol{I}_{i2})\boldsymbol{\omega}_i}^T + m_{i2}l_{i1}(\boldsymbol{\omega}_i^T \boldsymbol{d}_{i2})\tilde{\mathbf{s}}_i^T \,, \qquad i = 1,\ldots,6$

The tildes over the matrix-vector products in $\boldsymbol{D}_{22}$ and $\boldsymbol{h}_i$ denote a skew-symmetric matrix formed from the matrix-vector product.

$$\boldsymbol{G}_1 = \begin{bmatrix} \boldsymbol{G}_{11} \\ \boldsymbol{G}_{21} \end{bmatrix}, \ \boldsymbol{G}_2 = \begin{bmatrix} \boldsymbol{G}_{31} \\ \boldsymbol{G}_{41} \end{bmatrix} \tag{A6}$$

where $\boldsymbol{G}_{11} = -m_B \boldsymbol{g}$

$$\boldsymbol{G}_{21} = \mathbf{0}_{3\times 1}$$

$$\boldsymbol{G}_{31} = \left[-m_{12}\boldsymbol{g}^T \, {}^N_1\boldsymbol{R}\boldsymbol{s}_1 \quad \ldots \quad -m_{62}\boldsymbol{g}^T \, {}^N_6\boldsymbol{R}\boldsymbol{s}_6 \right]^T$$

$$\boldsymbol{G}_{41} = \left[-\boldsymbol{g}^T \, {}^N_1\boldsymbol{R}(m_{11}\tilde{\boldsymbol{d}}_{11} + m_{12}l_{11}\tilde{\mathbf{s}}_{\mathbf{i}} + m_{12}\tilde{\boldsymbol{d}}_{12})^T \quad \ldots \quad -\boldsymbol{g}^T \, {}^N_6\boldsymbol{R}(m_{61}\tilde{\boldsymbol{d}}_{61} + m_{62}l_{61}\tilde{\mathbf{s}}_{\mathbf{i}} + m_{62}\tilde{\boldsymbol{d}}_{62})^T \right]^T$$

24

Programming-by-Demonstration of Reaching Motions using a Next-State-Planner

Alexander Skoglund, Boyko Iliev
and Rainer Palm
Örebro University
Sweden

1. Introduction

Programming-by-Demonstration (PbD) is a central research topic in robotics since it is an important part of human-robot interaction. A key scientific challenge in PbD is to make robots capable of imitating a human. PbD means to instruct a robot how to perform a novel task by observing a human demonstrator performing it. Current research has demonstrated that PbD is a promising approach for effective task learning which greatly simplifies the programming process (Calinon et al., 2007), (Pardowitz et al., 2007), (Skoglund et al., 2007) and (Takamatsu et al., 2007). In this chapter a method for imitation learning is presented, based on fuzzy modeling and a next-state-planner in a PbD framework. For recent and comprehensive overviews of PbD, (also called "Imitation Learning" or "Learning from Demostration") see (Argall et al., 2009), (Billard et al., 2008) or (Bandera et al., 2007).
What might occur as a straightforward idea to copy human motion trajectories using a simple teaching-playback method, it turns out to be unrealistic for several reasons. As pointed out by Nehaniv & Dautenhahn (2002), there is significant difference in morphology between body of the robot and the robot, in imitation learning known as the correspondence problem. Further complicating the picture, the initial location of the human demonstrator and the robot in relation to task (i.e., object) might force the robot, into unreachable sections of the workspace or singular arm configurations. Moreover, in a grasping scenario it will not be possible to reproduce the motions of the human hand since there so far do not exist any robotic hand that can match the human hand in terms of functionality and sensing. In this chapter we will demonstrate that the robot can generate an appropriate reaching motion towards the target object, provided that a robotic hand with autonomous grasping capabilities is used to execute the grasp.
In the approach we present here the robot observes a human first demonstrating the environment of the task (i.e., objects of interest) and the the actual task. This knowledge, i.e., grasp-related object properties, hand-object relational trajectories, and coordination of reach-and-grasp motions is encoded and generalized in terms of *hand-state space* trajectories. The hand-state components are defined such that they are invariant with respect to perception, and includes the mapping between the human and robot hand, i.e., the correspondence. To

enable a *next-state-planner* (NSP) to perform reaching motions from an arbitrary robot configuration to the target object, the hand-state representation of the task is then embedded into the planner.
An NSP plans only one step ahead from its current state, which contrasts to traditional robotic approaches where the entire trajectory is planned in advance. In this chapter we use the term "next-state-planner", as defined by Shadmehr & Wise (2005), for two reasons. Firstly, because it emphasizes on the next–state planning ability, the alternative term being "dynamic system" which is very general. Secondly, the NSP also act as a *controller* which is an appropriate name, but "next-state-planner" is chosen because the controller in the context of an industrial manipulator refers to the low level control system. In this chapter the term planner is more appropriate. Ijspeert et al. (2002) were one of the first researchers to use an NSP approach in imitation learning. A humanoid robot learned to imitate a demonstrated motion pattern by encoding the trajectory in an autonomous dynamical system with internal dynamic variables that shaped a "landscape" used for both point attractors and limit cycle attractors. To address the above mention problem of singular arm configurations Hersch & Billard (2008) considered a combined controller with two controllers running in parallel; one controller acts in joint space, while the other one acts in Cartesian space. This was applied in a reaching task for controlling a humanoid's reaching motion, where the robot performed smooth motion while avoiding configurations near the joint limits. In a reaching while avoiding an obstacle task, Iossifidis & Schöner (2006) used attractor dynamics, where the target object acts as a point attractor on the end effctor. Both the end-effector and the redundant elbow joint avoided the obstacle as the arm reaches for an object.
The way to combine the demonstrated path with the robots own plan distinguishes our use of the NSP from from previous work (Hersch & Billard, 2008), (Ijspeert et al., 2002) and (Iossifidis & Schöner, 2006). Another difference is the use of the hand state space for PbD; most approaches for motion planning in the literature uses joint space (Ijspeert et al., 2002) while some other approaches use the Cartesian space.
To illustrate the approach we describe two scenarios where human demonstrations of goal-directed reach-to-grasp motions are reproduced by a robot. Specifically, the generation of reaching and grasping motions in pick-and-place tasks is addressed as well as the ability to learn from self observation. In the experiments we test how well the skills perform the demonstrated task, how well they generalize over the workspace and how skills can be adapted from self execution. The contributions of the work are as follows:

1. We introduce a novel next-state-planner based on a *fuzzy modeling* approach to encode human and robot trajectories.
2. We apply the *hand-state concept* (Oztop & Arbib, 2002) to encode motions in hand-state trajectories and apply this in PbD.
3. The combination of the NSP and the hand-state approach provides a tool to address the *correspondence problem* resulting from the different morphology of the human and the robot. The experiments shows how the robot can generalize and use the demonstration despite the fundamentally difference in morphology.
4. We present a performance metric for the NSP, which enables the robot to evaluate its performance and to adapt its actions to fit its own morphology instead of following the demonstration.

2. Learning from Human Demonstration

In PbD the idea is that the robot programmer (here called demonstrator) shows the robot what to do and from this demonstration an executable robot program is created. We assume the demonstrator to be aware of the particular restrictions of the robot. Given this assumption, the demonstrator shows the task by performing it in a way that seems to be feasible for the robot. In this work the approach is entirely based on proprioceptive information, i.e., we consider only the body language of the demonstrator. Furthermore, interpretation of human demonstrations is done under two assumptions: firstly, the type of tasks and grasps that can be demonstrated are *a priori* known by the robot; secondly, we consider only demonstrations of power grasps (e.g., cylindrical and spherical grasps) which can be mapped to–and executed by–the robotic hand.

2.1 Interpretation of Demonstrations in Hand-State Space

To create the associations between human and robot reaching/grasping we employ the hand-state hypothesis from the Mirror Neuron System (MNS) model of (Oztop & Arbib, 2002). The aim is to resemble the functionality of the MNS to enable a robot to interpret human goal-directed reaching motions in the same way as its own motions. Following the ideas behind the MNS-model, both human and robot motions are represented in hand-state space. A hand-state trajectory encodes a goal-directed motion of the hand during reaching and grasping. Thus, the hand-state space is common for the demonstrator and the robot and preserves the necessary execution information. Hence, a particular demonstration can be converted into the corresponding robot trajectory and experience from multiple demonstrations is used to control/improve the execution of new skills. Thus, when the robot execute the encoded hand-state trajectory of a reach and grasp motion, it has to move its own end-effector so that it follows a hand-state trajectory similar to the demonstrated one. If such a motion is successfully executed by the robot, a new robot skill is acquired.

Seen from a robot perspective, human demonstrations are interpreted as follows. If hand motions with respect to a potential target object are associated with a particular grasp type denoted G_i, it is assumed that there must be a target object that matches this observed grasp type. In other words, the object has certain grasp-related features which *affords* this particular type of grasp (Oztop & Arbib, 2002). The position of the object can either be retrieved by a vision system, or it can be estimated from the grasp type and the hand pose, given some other motion capturing device (e.g., magnetic trackers). A subset of suitable object affordances is identified a priori and learned from a set of training data for each grasp type G_i. In this way, the robot can associate observed grasp types G_i with their respective affordances A_i.

According to Oztop & Arbib (2002), the hand-state must contain components describing both the hand configuration and its spatial relation with respect to the affordances of the target object. Thus, the general definition of the hand-state is in the form:

$$H = \{h_1, h_2, \dots h_{k-1}, h_k, \dots h_p\} \tag{1}$$

where $h_1 \dots h_{k-1}$ are *hand-specific components* which describe the motion of the fingers during a reach-to-grasp motion. The remaining components $h_k \dots h_p$ describe the motion of the hand in relation to the target object. Thus, a hand-state trajectory contains a record of both the reaching and the grasping motions as well as their synchronization in time and space.

The hand-state representation in Eqn. 1 is invariant with respect to the actual location and orientation of the target object. Thus, demonstrations of object-reaching motions at different locations and initial conditions can be represented in a common domain. This is both the strength and weakness of the hand-state approach. Since the origin of the hand-state space is in the target object, a displacement of the object will not affect the hand-state trajectory. However, when an object is firmly grasped then, the hand-state become fixated and will not capture a motion relative to the base coordinate system. This implies that for object handling and manipulation the use of a single hand-state trajectory is insufficient.

2.2 Skill Encoding Using Fuzzy Modeling

Once the hand-state trajectory of the demonstrator is determined, it has to be modeled for several reasons. Five important and desirable properties for encoding movements have been identified, and Ijspeert et al. (2001) enumerates them as follows:

1. The representation and learning of a goal trajectory should be simple.
2. The representation should be compact (preferably parameterized).
3. The representation should be reusable for similar settings without a new time consuming learning process.
4. For recognition purpose, it should be easy to categorize the movement.
5. The representation should be able to act in a dynamic environment and be robust to perturbations.

A number of methods for encoding human motions have previously been proposed including splines (Ude, 1993); Hidden Markov Models (HMM) (Billard et al., 2004); HMM combined with Non-Uniform Rational B-Splines (Aleotti & Caselli, 2006); Gaussian Mixture Models (Calinon et al., 2007); dynamical systems with a set of Gaussian kernel functions (Ijspeert et al., 2001). The method we propose is based on fuzzy logic which deals with the above properties in a sufficient manner (Palm et al., 2009).

Let us examine the properties of fuzzy modeling with respect to the above enumerated desired properties. Fuzzy modeling is simple to use for trajectory learning and is a compact representation in form of a set of weights, gains and offsets (i.e., they fulfill property 1 and 2) (Palm & Iliev, 2006). To change a learned trajectory into a new one for a similar task with preserved characteristics of a motion, we proposed a modification of the fuzzy time modeling algorithm (Iliev et al., 2007), thus addressing property 3. The method also satisfies property 4, as it was successfully used for grasp recognition by (Palm et al., 2009). In (Skoglund, Iliev & Palm, 2009) we show that our fuzzy modeling based NSP is robust to short perturbations, like NSPs in general are known to be robust to perturbations (Ijspeert et al., 2002) and (Hersch & Billard, 2008).

Here it follows a description of the fuzzy time modeling algorithm for motion trajectories. Takagi and Sugeno proposed a structure for fuzzy modeling of input-output data of dynamical systems (Takagi & Sugeno, 1985). Let $\mathbf{X}$ be the input data set and $\mathbf{Y}$ be the output data set of the system with their elements $x \in \mathbf{X}$ and $y \in \mathbf{Y}$. The fuzzy model is composed of a set of c rules R from which rule R_i reads:

$$\text{Rule } i: \text{ IF } x \text{ IS } \mathbf{X}_i \text{ THEN } y = A_i x + B_i \tag{2}$$

$\mathbf{X}_i$ denotes the i:th fuzzy region in the fuzzy state space. Each fuzzy region $\mathbf{X}_i$ is defuzzified by a fuzzy set $\int w_{x_i}(x)|x$ of a standard triangular, trapezoidal, or bell-shaped type. $W_i \in \mathbf{X}_i$ denotes the fuzzy value that x takes in the i:th fuzzy region $\mathbf{X}_i$. A_i and B_i are fixed parameters of the local linear equation on the right hand side of Eqn. 2.

The variable $w_i(x)$ is also called degree of membership of x in $\mathbf{X}_i$. The output from rule i is then computed by:

$$y = w_i(x)(A_i x + B_i) \tag{3}$$

A composition of all rules $R_1 \ldots R_c$ results in a summation over all outputs from Eqn. 3:

$$y = \sum_{i=1}^{c} w_i(x)(A_i x + B_i) \tag{4}$$

where $w_i(x) \in [0, 1]$ and $\sum_{i=1}^{c} w_i(x) = 1$.
The fuzzy region $\mathbf{X}_i$ and the membership function w_i can be determined in advance by design or by an appropriate clustering method for the input-output data. In our case we used a clustering method to cope with the different nonlinear characteristics of input-output datasets (see (Gustafson & Kessel, 1979) and (Palm & Stutz, 2003)). For more details about fuzzy systems see (Palm et al., 1997).
In order to model time dependent trajectories $x(t)$ using fuzzy modeling, the time instants t take the place of the input variable and the corresponding points $\mathbf{x(t)}$ in the state space become the outputs of the model.

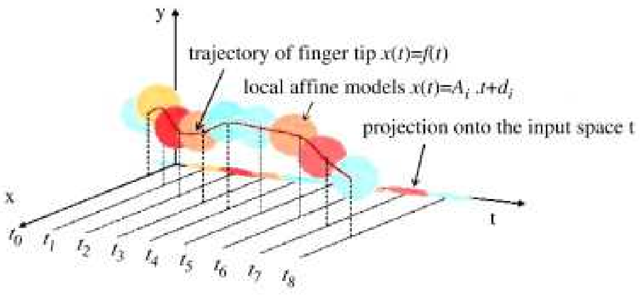

y
trajectory of finger tip $x(t)=f(t)$
local affine models $x(t)=A_i \cdot t+d_i$
projection onto the input space t
x
t
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8

Fig. 1. Time-clustering principle.

The Takagi-Sugeno (TS) fuzzy model is constructed from captured data from the end-effector trajectory described by the nonlinear function:

$$\mathbf{x}(t) = \mathbf{f}(t) \tag{5}$$

where $\mathbf{x}(t) \in \mathbb{R}^3, \mathbf{f} \in \mathbb{R}^3$, and $t \in \mathbb{R}^+$.
Equation (5) is linearized at selected time points t_i with

$$\mathbf{x}(t) = \mathbf{x}(t_i) + \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \cdot (t - t_i) \tag{6}$$

resulting in a locally linear equation in t.

$$\mathbf{x}(t) = \mathbf{A}_i \cdot t + \mathbf{d}_i \tag{7}$$

where $\mathbf{A}_i = \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \in \mathbb{R}^3$ and $\mathbf{d}_i = \mathbf{x}(t_i) - \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \cdot t_i \in \mathbb{R}^3$. Using Eqn. 7 as a local linear model one can express Eqn. 5 in terms of an interpolation between several local linear models by applying Takagi-Sugeno fuzzy modeling (Takagi & Sugeno, 1985) (see Fig. 1)

$$\mathbf{x}(t) = \sum_{i=1}^{c} w_i(t) \cdot (\mathbf{A}_i \cdot t + \mathbf{d}_i) \tag{8}$$

$w_i(t) \in [0, 1]$ is the degree of membership of the time point t to a cluster with the cluster center t_i, c is number of clusters, and $\sum_{i=1}^{c} w_i(t) = 1$.
The degree of membership $w_i(t)$ of an input data point t to an input cluster C_i is determined by

$$w_i(t) = \frac{1}{\sum_{j=1}^{c} \left(\frac{(t-t_i)^T M_{i\,pro}(t-t_i)}{(t-t_j)^T M_{j_{pro}}(t-t_j} \right)^{\frac{1}{\tilde{m}_{proj}-1}}} \tag{9}$$

The projected cluster centers t_i and the induced matrices $M_{i\,pro}$ define the input clusters C_i ($i = 1 \ldots c$). The parameter $\tilde{m}_{pro} > 1$ determines the fuzziness of an individual cluster (Gustafson & Kessel, 1979).

2.3 Model Selection using Q-learning

The actions of the robot have to be evaluated to enable the robot to improve its performance. Therefore, we use the state-action value concept from reinforcement learning to evaluate each action (skill) in a point of the state space (joint configuration). The objective is to assign a metric to each skill to determine its performance, and include this is a reinforcement learning framework.
In contrast to most other reinforcement learning applications, we only deal with one state-action transition, meaning that from a given position only one action is performed and then judged upon. A further distinction to classical reinforcement learning is to ensure that all actions initially are taken so that all existing state-action transitions are tested. Further improvement after the initial learning and adaption is possible by implementing a continuous learning scheme. Then, the robot will receive the reward after each action and continues to improve the performance over a longer time. However, this is beyond the scope of this chapter, and is a topic of future work.
The trajectory executed by the robot is evaluated based on three criteria:

- Deviation between the demonstrated and executed trajectories.
- Smoothness of the motion, less jerk is preferred.
- Successful or unsuccessful grasp.

In a Q-learning framework, the reward function can be formulated as:

$$r = -\frac{1}{t_f} \sum_{t=0}^{t=t_f} |H^r(t) - H(t)| - \frac{1}{t_f} \sum_{t=0}^{t=t_f} \dddot{x}^2(t) + r_g \tag{10}$$

where

$$r_g = \begin{cases} -100 & \text{if} \quad \text{Failure} \\ 0 & \text{if} \quad \text{Success, but overshoot} \\ +100 & \text{if} \quad \text{Success} \end{cases} \tag{11}$$

where $H^r(t)$ is the hand-state trajectory of the robot, $H(t)$ is the hand-state of the demonstration. The second term of the Eqn. 10 is proportional to the jerk of the motion, where t_f

is the duration of the motion, t_0 is the starting time and $\dddot{x}$ is the third derivative of the motion, i.e., jerk. The third term in Eqn. 10 r_g, is the reward from grasping as defined in Eqn. 11 where "failure" meaning a failed grasp and "success" means that the object was successfully grasped. In some cases the end-effector performs a successful grasp but with a slight overshoot, which is an undesired property since it might displace the target. An overshoot means that the target is sightly missed, but the end-effector then returns to the target.
When the robot has executed the trajectories and received the subsequent rewards and the accumulated rewards, it determines what models to employ, i.e., action selection. The actions that received the highest positive reward are re-modeled, but this time as robot trajectories using the hand-state trajectory of the robot as input to the learning algorithm. This will give less discrepancies between the modeled and the executed trajectory, thus resulting in a higher reward. In Q-learning a value is assigned for each state-action pair by the rule:

$$Q(s,a) = Q(s,a) + \alpha * r \tag{12}$$

where s is the state, in our case the joint angles of the manipulator, a is the action, i.e., each model from the demonstration, and α is a step size parameter. The reason for using the joint space as state space is the highly non-linear relationship between joint space and hand-state space (i.e., a Cartesian space): two neighboring points in joint space are neighboring in Cartesian space but not the other way around. This means that action selection is better done in joint space since the same action is more likely to be suitable for two neighboring points than in hand-state space.
To approximate the Q-function we used Locally Weighted Projection Regression (LWRP)[1] as suggested in (Vijayakumar et al., 2005), see their paper for details.

3. Generation and Execution of Robotic Trajectories based on Human Demonstration

This section covers generation and execution of trajectories on the actual robot manipulator. We start with a description of how we achieve the mapping from human to robot hand and how to define the hand-state components. Then, section 3.3 describes the next-state-planner, which produces the actual robot trajectories.

3.1 Mapping between Human and Robot Hand States

The definition of H is *perception invariant* and must be able to update from any type of sensory information. The hand-sate components $h_1, \ldots h_p$ are such that they can be recovered both from human demonstration and from robot perception. Fig. 2 shows the definition of the hand-state in this article.
Let the human hand be at some initial state H_1. The hand then moves along the demonstrated path until the final state H_f is reached where the target object is grasped by the hand (Iliev et al., 2007). That is, the recorded motion trajectory can be seen as a sequence of states, i.e.,

$$H(t) : H_1(t_1) \rightarrow H_2(t_2) \rightarrow \ldots \rightarrow H_f(t_f) \tag{13}$$

Since the hand state representation is defined in relation to the target object, the robot must have access to the complete trajectory of hand of the demonstrator. Therefore the hand-state trajectory can only be computed *during* a motion if the target object is known in advance.

[1] Available at: http://www-clmc.usc.edu/software/lwpr

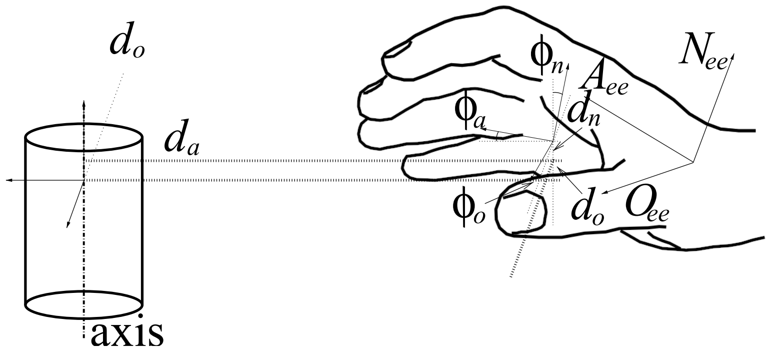


Fig. 2. The hand-state describes the relation between the hand pose and the object affordances. N_{ee} is the the normal vector, O_{ee} the side (orthogonal) vector and A_{ee} is the approach vector. The vector Q_{ee} is the position of the point. The same definition is also valid for boxes, but with the restriction that the hand-state frame is completely fixed, it cannot be rotated around the symmetry axis.

Let $H_{des}(t)$ be the desired hand-state trajectory recorded from a demonstration. In the general case the desired hand-state $H_{des}(t)$ cannot be executed by the robot without modification. Hence, a *robotic* version of $H_{des}(t)$ have to be constructed, denoted by $H^r(t)$, see Fig. 3 for an illustration.
One advantage of using only one demonstrated trajectory as the desired trajectory over trajectory averaging (e.g., (Calinon et al., 2007) or (Delson & West, 1996)) is that the average might contain two essentially different trajectories (Aleotti & Caselli, 2006). By capturing a human demonstration of the task, the synchronization between reach and grasp is also captured, demonstrated in (Skoglund et al., 2008). Other ways of capturing the human demonstration, such as kinesthetics (Calinon et al., 2007) or by a teach pendant (a joystick), cannot capture this synchronization easily.
To find $H^r(t)$ a mapping from the human grasp to the robot grasp a transformation is needed, denoted T^r_h. This transformation can be obtained as a static mapping between the pose of the demonstrator hand and the robot hand while they are holding the same object at a fixed position. Thus, the target state H^r_f will be derived from the demonstration by mapping the goal configuration of the human hand H_f into a goal configuration for the robot hand H^r_f, using the transformation T^r_h:

$$H^r_f = T^r_h H_f \tag{14}$$

The pose of the robot hand at the start of a motion defines the initial state H^r_1. Since H^r_f represents the robot hand holding the object , it has to correspond to a stable grasp. For a known object, suitable H^r_f can either be obtained by simulation (Tegin et al., 2009), grasp planning or by learning from experimental data. Thus, having a human hand state H_f and their corresponding robot hand state H^r_f, T^r_h is obtained as:

$$T^r_h = H^r_f H^{-1}_f \tag{15}$$

It should be noted that this method is only suitable for power grasps. In the general case it might produce ambiguous results or rather inaccuarate mappings.

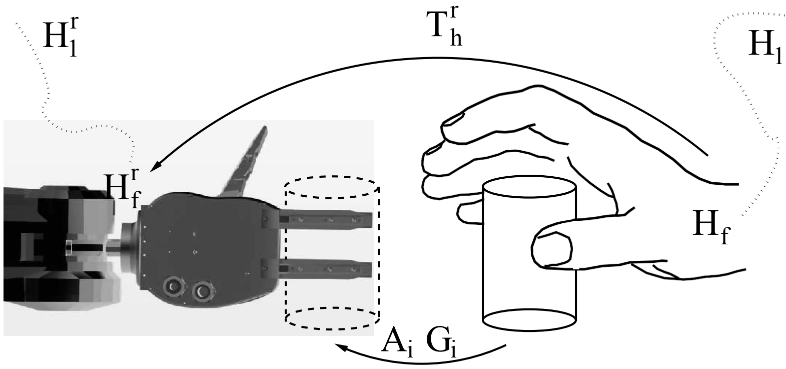


Fig. 3. Mapping from human hand to robotic gripper.

3.2 Definition of Hand-States for Specific Robot Hands

hen the initial state H_1^r and the target state H_f^r are defined, we have to generate a trajectory between the two states. In principle, it is possible to use Eqn. 14 to $H_{des}(t)$ such that it has its final state in H_f^r. The robot then starts at H_1^r and approaches the displaced demonstrated trajectory and then track this desired trajectory until the target state is reached. However, such an approach would not take trajectory constraints into account. Thus, it is also necessary to specify exactly how to approach $H_{des}(t)$ and what segments must be tracked accurately.
The end-effector trajectory is reconstructed from the recorded demonstration, and is represented by a time dependent homogeneous matrix $T_{ee}(t)$. Each element is represented by the matrix

$$T_{ee} = \begin{pmatrix} N_{ee} & O_{ee} & A_{ee} & Q_{ee} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{16}$$

where N_{ee}, O_{ee} and A_{ee} are the normal vector, the side vector, and the approach vector, respectively of the end effector. The position is represented by the vector Q_{ee}. It is important to note that the matrix T_{ee} is defined differently for different end-effectors, for example, the human hand is defined as in Fig. 2.
From the demonstrated trajectory, a handstate trajectory can be obtained as afunction of time. We formulate the hand-state as:

$$H(t) = [d_n(t)\ d_o(t)\ d_a(t)\ \phi_n(t)\ \phi_o(t)\ \phi_a(t)] \tag{17}$$

The individual components denote the position and orientation of the end-effector. The first three components, $d_n(t)$, $d_o(t)$ and $d_a(t)$, describe the distance from the object to the hand along the three axes n, o and a with the object as the base frame. The next three components, $\phi_n(t)$, $\phi_o(t)$ and $\phi_a(t)$, describe the rotation of the hand in relation to the object around the three axes n, o and a. The notion of the hand-state used in this section is illustrated in Fig. 2.

Note that by omitting the finger specific components of the hand-state we get a simplified definition of H, but cannot determine the type of human grasp. In (Skoglund et al., 2008) we give an account of how the finger specific components and object relation components can be used to synchronize reaching with grasping. Another reason for omitting finger specific components is that grasp classification is out of scope of this chapter; only power grasps are used the subsequent experiments. Thus, the grasp type is assumed to be known $G = \{cylindrical, spherical, plane\}$; the affordances are: position, size, and cylinder axis $A = \{width, axis\}$ or box $A = \{width, length, N\text{-}axis, O\text{-}axis, A\text{-}axis\}$. See (Palm & Iliev, 2007) for grasp taxonomy.

3.3 Next-State-Planners for Trajectory Generation

In this section we present the next-state-planner (NSP) that balances its actions between *following a demonstrated trajectory* and *approaching the target*, first presented in (Skoglund et al., 2008). The NSP we use is inspired by the Vector Integration To Endpoint (VITE) planner propsed by Bullock & Grossberg (1989) as a model for human control of reaching motions. The NSP-approach requires a control policy, i.e., a set of equations describing the next action from the current state and some desired behavior.

The NSP generates a hand-state trajectory using the TS fuzzy-model of a demonstration. Since the resulting hand-state trajectory $H^r(t)$ can easily be converted into Cartesian space, the inverse kinematics provided by the arm controller can be used directly.

The TS fuzzy-model serves as a motion primitive for the arm's reaching motion. The initial hand-state of the robot is determined from its current configuration and the position and orientation of the target object, since these are known at the end of the demonstration. Then, the desired hand-state H^r_{des} is computed from the TS fuzzy time-model (Eqn. 8). The desired hand-state H_{des} is fed to the NSP. Instead of using only one goal attractor as in VITE (Bullock & Grossberg, 1989), and additional attractor–the desired hand-state trajectory–is used at each state. The system has the following dynamics:

$$\ddot{H} = \alpha(-\dot{H} + \beta(H_g - H) + \gamma(H_{des} - H)) \tag{18}$$

where H_g is the hand-state goal, H_{des} the desired state, H is the current hand-state, $\dot{H}$ and $\ddot{H}$ are the velocity and acceleration respectively. α is a positive constant (not to be confused with the step size parameter α in Eqn. 12) and β, γ are positive weights for the goal and tracking point, respectively.

If the last term $\gamma(H_{des} - H)$ in Eqn. 18 is omitted, i.e., $\gamma = 0$, then the dynamics is *exactly* as the VITE planner Bullock & Grossberg (1989). Indeed, if no demonstration is available the planner can still produce a motion if the target is known. Similarly, if the term $\beta(H_g - H)$ is omitted, the planner becomes a trajectory following controller. To determine β and γ, which controls the behavior of the NSP, we use a time dependent weighting mechanism. The weighting is a function of time left t_l to reach the goal at t_f; $\gamma = K(t_l/t_f)^2$, where K is 2; $\beta = 1 - \gamma$. Since the environment demonstration provides better accuracy than the task demonstration, it is reasonable to give the target a hight weight at the end of the trajectory (i.e., $\beta = 1$), Spherical linear interpolation is used. To interpolate between initial and final orientation along the trajectory spherical linear interpolation is used (Shoemake, 1985).

It is also possible to determine β and γ by the variance across multiple demonstrations (Skoglund, Tegin, Iliev & Palm, 2009).

Analytically, the poles in Eqn. 18 are:

$$p_1, p_2 = -\frac{\alpha}{2} \pm \sqrt{\frac{\alpha^2}{4} - \alpha\gamma}. \tag{19}$$

The real part of p_1 and p_2 will be ≤ 0, which will result in a stable system (Levine, 1996). Moreover, $\alpha \not\lesssim 4\gamma$ and $\alpha \geq 0$, $\gamma \geq 0$ will contribute to a critically damped system, which is fast and has small overshoot. Fig. 4 shows how different values γ affect the dynamics of the planner.

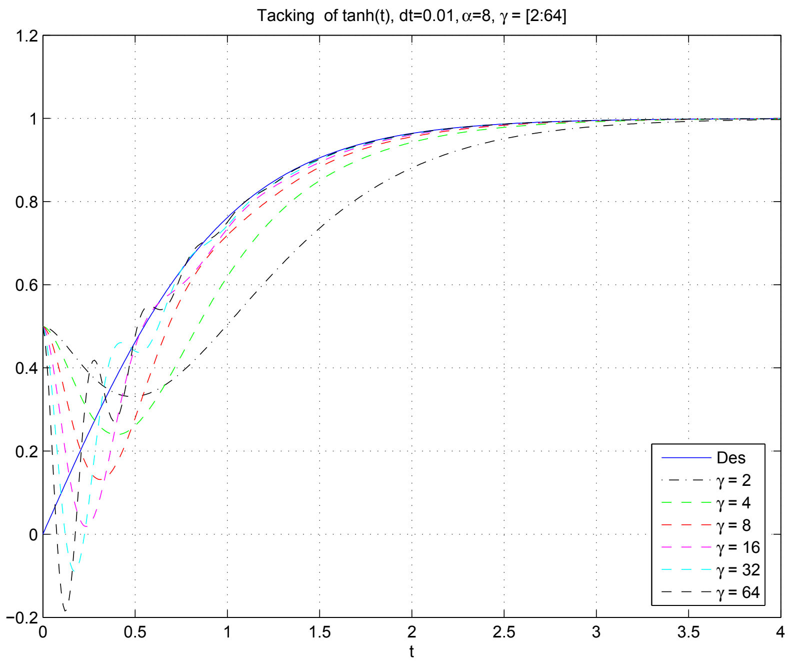


Fig. 4. The dynamics of the planner for six different values of γ. The tracking point is $tanh(t)$, with $dt = 0.01$ and α is fixed at 8. A low value on $\gamma = 2$ produces slow dynamics (black dot-dashed line), while a high value $\gamma = 64$ is fast but overshoots the tracking point (black dashed line).

The controller has a feedforward structure as in Fig. 5. The reason for this structure is that a commercial manipulator usually has a closed architecture, where the controller is embedded in the system. For this type of manipulators, a trajectory is usually pre-loaded and then executed. Therefore, we generate the trajectories in batch mode for the ABB140 manipulator. Since our approach is general, for a given different robot platform with hetroceptive sensors (e.g., vision) our method can be implemented in a feedback mode, but this requires that the hand-state $H(t)$ can be measured during execution.

3.4 Demonstrations of Pick-and-Place Tasks

In our setup a demonstration is done in two stages: environment- and task demonstration. During the first stage, the *environment demonstration*, the target objects in the workspace are

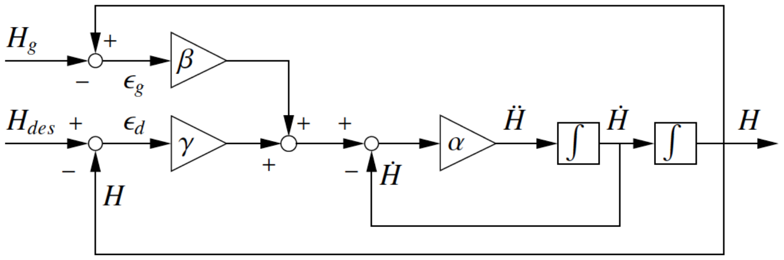


Fig. 5. Hand-state planner architecture. H_g is the desired hand-state goal, H_{des} is the desired hand-state at the current distance to target.

shown to the robot. The environment demonstration can provide a more accurate workspace model than the task demonstration, since this is a separated stage and the demonstrator can focus on the objects and don't consider the dynamics of the task. In the experiment in section 5.2, we use a simplistic modeling of the environment where all objects are modeled as box shaped objects. The result from an environment demonstration is shown in Fig. 6, where the bounding boxes were created from information on hand/fingerpose and tactile data. A bounding box represents each object with position of the center and length, width and height, which are used to compute the orientation of the object. A more sophisticated modeling–based on point clouds–could be used if a better granularity of the workspace is needed (Charusta et al., 2009).
In the *task demonstration*, i.e., a pick-and-place of an object, only the task is shown. Once the workspace model is available, only the demonstrated trajectory is used. If an environment demonstration is unavailable the target object can be determined from task demonstration, where the center point of the grasp can be estimated from the grasp type. However, this not as accurate as using data form an environment demonstration. The task demonstration contains the trajectories which the robot should execute to perform the task. Trajectories recorded from a task demonstration are shown in Fig. 7.

4. Experimental Platform

For these experiments human demonstrations of a pick-and-place task are recorded with two different subjects, using the PhaseSpace Impulse motion capturing system. The Impulse system consists of four cameras (in experiment 1), and five (in Experiment 2) mounted around the operator to register the position of the LEDs. Each LED has a unique ID by which it is identified. Each camera can process data at 480 Hz and have 12 Mega pixel resolution resulting in sub-millimeter precision. One of the cameras can be seen in the right picture of Fig. 8. The operator wears a glove with LEDs attached to it, left picture see Fig. 8. Thus, each point on the glove can be associated with a finger, the back of the hand or the wrist. To compute the orientation of the wrist, three LEDs must be visible during the motion. The back of the hand is the best choice since three LEDs are mounted there and they are most of the time visible to at least three cameras. One LED is mounted on each finger tip, and the thumb has one additional LED in the proximal joint. One LED is also mounted on the target object. Moreover, we have

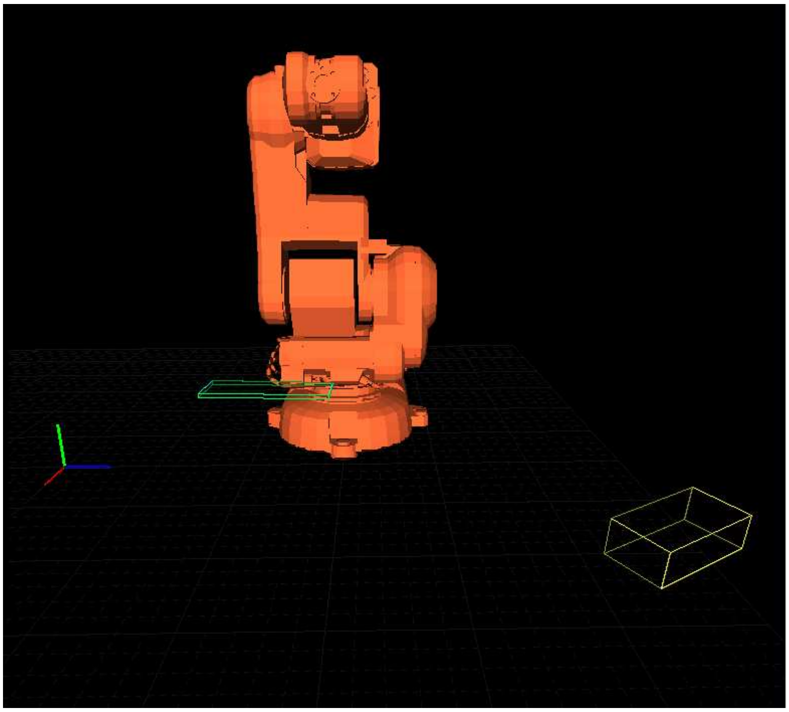


Fig. 6. The result of an environment demonstration.

mounted tactile sensors (force sensing resistors) to detect contact with objects. The sensors are mounted on the fingertips of the glove, shown in the middle of Fig. 8. We define a grasp as when contact is detected at the thumb sensor and one additional finger. This means that only grasps which include the thumb and one other finger can be detected. No grasp recognition is necessary since the gripper only allows one grasp type. When a grasp is detected the distance to each object in the workspace is measured and the nearest object, if below some distance threshold, is identified as the target object.

The motions are automatically segmented into reach and retract motions using the velocity profile and distance to the object. The robot used in the experiments is the industrial manipulator ABB IRB140. In this experiment we use the anthropomorphic gripper KTHand (Fig. 9), which can perform power grasps (i.e., cylindrical and spherical grasps) using a hybrid position/force controller. For details on the KTHand, see (Tegin et al., 2008).

The demonstrations were performed with the teacher standing in front of the robot. First, the environment is demonstrated by tactile exploration of the workspace. The demonstrator touches the objects of interest; with special care so the boundaries of each object are correctly captured. Second, the task is demonstrated where the teacher starts with the hand in a position similar the robot's home position, i.e., $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$. The results from the environment and task demonstrations are shown in Fig. 6 and 7 respectively, with the base frame of the

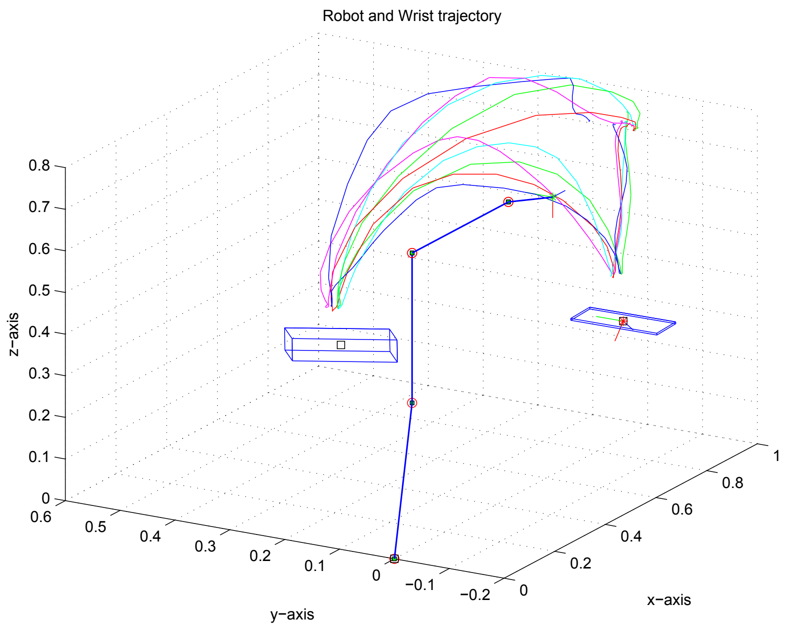


Fig. 7. Five sample demonstrated trajectories (the wrist is plotted), the robot and the two objets of interest. The object to the right is marked with a red star to indicate that this object was the nearest when a grasp was detected in the task demonstration.

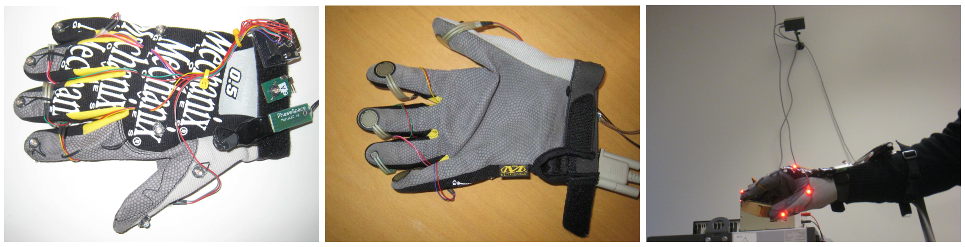


Fig. 8. **Left:** The glove used in the Impulse motion capturing system from PhaseSpace. The glove from the top showing the LEDs. **Middle:** The glove with the tactile sensors mounted on each finger tip. **Right:** The system in use showing one of the cameras and the LED on the glove.

robot as the reference frame. The object is determined from the environment demonstration, since it is more exact compared to the task demonstration.

5. Experimental Evaluation

In this section we provide an experimental evaluation of the presented method.

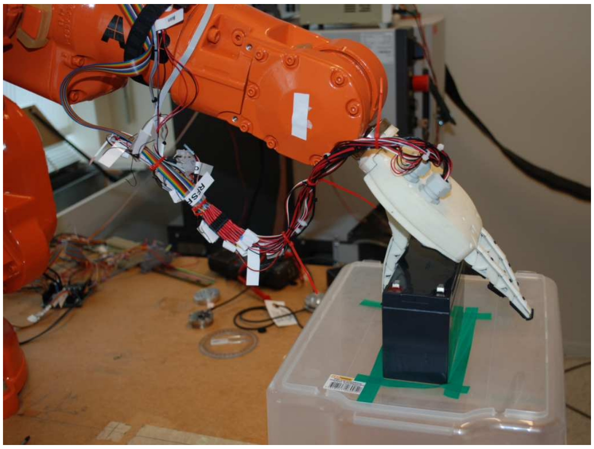


Fig. 9. The anthropomorphic gripper KTHand used in the second experiment.

5.1 Experiment 1 – A Complete Pick-and-Place Task

To test the approach on an integrated system the KTHand is mounted on the ABB manipulator and a pick-and-place task is executed, guided by a demonstration showing pick-and-place task of a box (110 $\times$ 56 $\times$ 72 mm). The synchronization between reach and grasp is performed by a simple finite state machine. After the grasp is executed, the motion to the placing point is performed by following the demonstrated trajectory (see section 2.2). Since the robot grasp pose corresponds approximately to the human grasp pose it is possible for the planner to reproduce the human trajectory almost exactly. This does not mean that the robot actually can execute the trajectory, due to workspace constraints. The retraction phase follows the same strategy as the reaching motion, but in reverse. Fig. 10 shows the complete task learned from demonstration.

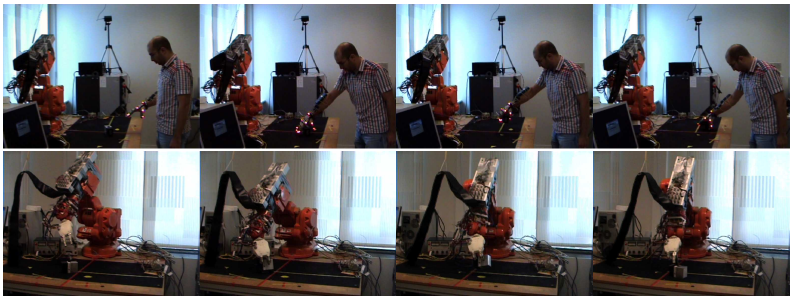


Fig. 10. Industrial manipulator programmed using a demonstration. A movie of the sequence is available at: http://www.aass.oru.se/Research/Learning/arsd.html.

5.2 Experiment 2 – Skill Improvement

In this experiment the robot uses the self executed trajectories for skill modeling and compare the performance to the skills models from human demonstration. The aim is to enable to robot to improve its performance. In this experiment 20 task demonstrations of the same pick-and-place task were performed, where only the reaching part of the task was used to train a reaching skill. Five of these demonstrations are shown in Fig. 7. In addition, one environment demonstration was also recorded, shown in Fig. 6. All demonstrations were modeled using fuzzy time-modeling, where the position of the index finger and orientation of the wrist were used to record the trajectory. Three of the modeled trajectories are shown in figure 11 in the top graphs as dashed lines. The reason for using the index finger instead of–what would be more appropriate–the center point between the index finger and the tip of the thumb is that the LED on the thumb was often hidden resulting from occlusions during the motion. Thus, the number of occlusions is minimized.

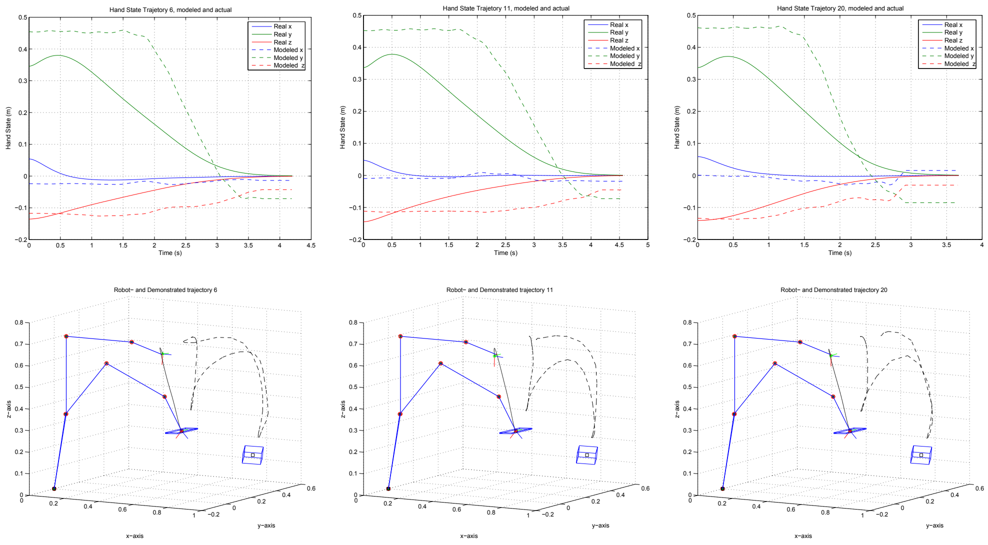


Fig. 11. Demonstrations number H_6, H_{11} and H_{20}, of which the two former were the two best performing models, and the latter was the worst performing model in generating the reaching trajectory for the robot to execute. **Top:** The dashed lines in the top graphs are the modeled trajectories from the demonstration H_{des}. The solid lines are the trajectory executed by the robot, H^r. **Bottom:** Both as H_{des} (dashed) as the human demonstrated in, and H^r (solid) as the robot executed it, Cartesian space.

5.2.1 Evaluation of Hand-State Trajectories

All the recorded demonstrations were modeled and executed by the robot in the same manner as in Exp. 1. In addition the performance of each model was evaluated, the criteria in Eqn. 10. Three of the executed trajectories are shown in Fig. 11 in the top graph as solid lines and in the bottom graph as Cartesian trajectories. At the end of the demonstrations (around 3.5 to 4.0 sec) in Fig. 11 the difference between the model and the actual trajectory is up to 8 cm (in

y-direction). Since the model is created from the trajectory of the fingertip of the index finger, the end of the trajectory will be displaced from the center of the target object. This is due to the fact that the index finger will be at the side of the box shaped object during the grasp. Recall from the definition of the hand-state space (section 2.1) that the target object is the frame in which all motions are in relation to. This means that the origin of the hand-state frame is the top center of the target object. The point attractor of Eqn. 18 for the trajectory generation is switching from the task demonstration to the target object, as described below. Therefore, the point $[0,0,0]$ becomes the point attractor at the end of the motion.
The hand-state error and minimum jerk can be evaluated already in the generation state in simulation, i.e., before the actual execution on the real robot. The grasp success is evaluated from real execution. In the real execution the robot starts in the home position $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$ deg, with a small perturbation added to the last three joints since the home position is a singular configuration. All models succeeded in generating a trajectory which positioned the end-effector in such a way that a successful grasp can be performed, resulting in a positive reward of $+100$. This reward is then decreased by adding a negative reward from hand-state error and jerk, as described by equations 10-11.

Human Actions	H_1	H_2	H_3	H_4	H_5	H_6	H_7
Q-values	0.7572	0.7854	0.8120	0.8596	0.6864	0.9521	0.7410
Human Actions	H_8	H_9	H_{10}	H_{11}	H_{12}	H_{13}	H_{14}
Q-values	0.8966	0.7738	0.8659	0.9964	0.8555	0.8349	0.8334
Human Actions	H_{15}	H_{16}	H_{17}	H_{18}	H_{19}	H_{20}	
Q-values	0.8319	0.8481	0.7656	0.7572	-1.9459	0.6745	
Robot Actions	R_1	R_2					
Q-values	1.0769	1.0437					

Table 1. The Q-values at the home configuration, $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$ deg. When the robot imitated human actions H_6 and H_{11} (light gray) from its home configuration, it received the highest Q-value and used these two to create the two new action models R_1 and R_2. To compare the two original motions H_6 and H_{11} were selected since they performed best and H_{20} which had the lowest positive Q-value.

In Table 1 this is shown in the rows with joint configuration $\Theta = [0\,0\,0\,0\,0\,0]^T$ deg. The models from human demonstration H_6 and H_{11} performed best when the robot executed them. Hence, these values were selected to be remodeled into robot skills: R_1 and R_2, meaning that the robot trajectory is used to create the fuzzy models. Worst performance was obtained in H_{19} and H_{20}, where H_{19} received a negative Q-value. Finally, R_1 and R_2 were tested from the home configuration and evaluated using the same criteria, shown in Fig. 12. As expected, their performance was better than all others, since the hand-state error is reduced.

5.2.2 Generalization of Robot Skills

Generalization over the workspace is used as a test on how well the skills modeled from robot execution perform in comparison to the skills modeled from human demonstration. This is done by positioning the manipulator in twelve starting configurations, different from the home configuration $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$, while keeping the target object at the same position as in the demonstration. The models H_6 and H_{11}, acquired from human demonstrations, received the highest Q-values (see Table 1) and are therefor selected for comparison. In addition we also selected one of the worst scoring models of the remaining models, namely H_{20}. A set of twelve joint configurations were chosen as the initial starting positions. The configurations were chosen to approximately correspond to 1/4 of the full joint range at each joint, except last joint where 1/8 of the range was used. Two exceptions from this were the configurations $\Theta = [0\ 20\ 40\ 1\ 1\ 1]^T$ and $\Theta = [0\ 0\ 25\ 1\ 1\ 1]^T$ due to workspace restrictions. The resulting Q-values after executing a reaching motion form these joint configurations can be seen in Table 2. In the table the highest Q-values are highlighted. From two of the starting configurations none of the models could execute a successful reaching motion, namely $\Theta = [0\ -45\ 0\ 1\ 1\ 1]^T$ deg and $\Theta = [0\ 0\ 0\ 1\ -55\ 1]^T$ deg (dark rows in Table 2). This means that new demonstrations should be done from these positions. From the other joint configurations neither H_6 nor H_{11} performed best. Surprisingly, the worst model H_{20} was the only one to perform a successful reaching motion from configuration $\Theta = [0\ 0\ -50\ 1\ 1\ 1]^T$ deg, resulting in the highest Q-value. From the rest of the configurations robot skill R_1 or R_2 performed best, thus confirming our hypothesis that skills modeled from own experience are better adapted to the robot than skills directly modeled from observation. Fig. 13 shows four sample configurations where the robot skill R_1 and R_2 are used to generate reaching actions from configurations different from the trained position. In Fig. 14 two sequences are shown where the real robot executes the reaching skills from the trained position, and from the tested position.

	Q-values				
	Human Actions			Robot Actions	
Joint configuration, **q**	H_6	H_{11}	H_{20}	R_1	R_2
$[0\ 0\ 0\ 0\ 0\ 0]^T$	0.9521	0.9964	0.6745	1.0769	1.0437
$[-45\ 0\ 0\ 1\ 1\ 1]^T$	1.6079	1.7029	1.1207	−0.0670	1.7847
$[0\ -45\ 0\ 1\ 1\ 1]^T$	−0.3890	−0.2960	−0.8784	−0.0670	−0.2143
$[0\ 0\ -50\ 1\ 1\ 1]^T$	−0.3895	−0.2968	1.1208	−0.0676	−0.2150
$[0\ 0\ 0\ -50\ 1\ 1]^T$	1.6104	1.7032	1.1208	−0.0675	1.7850
$[0\ 0\ 0\ 1\ -55\ 1]^T$	−0.3895	−0.2968	−0.8792	−0.0676	−0.2150
$[0\ 0\ 0\ 1\ 1\ -50]^T$	1.6102	1.7031	1.1208	−0.0675	1.7850
$[45\ 0\ 0\ 1\ 1\ 1]^T$	1.6102	1.7030	1.1206	1.9320	1.7847
$[0\ 20\ 40\ 1\ 1\ 1]^T$	1.6104	1.7032	1.1207	1.9324	1.7850
$[0\ 0\ 25\ 1\ 1\ 1]^T$	1.5848	1.6515	1.0994	1.9017	1.7275
$[0\ 0\ 0\ 50\ 1\ 1]^T$	1.6105	1.7031	1.1207	1.9324	1.7850
$[0\ 0\ 0\ 1\ 55\ 1]^T$	1.6105	1.7032	1.1208	1.9324	1.7850
$[0\ 0\ 0\ 1\ 1\ 50]^T$	1.6105	1.7032	1.1208	1.9324	1.7850

Table 2. The Q-values for selected joint configurations. Twelve other configurations than the home configuration were selected to test how well the robot skills R_1 and R_2 performed. As comparison, the two original motions H_6 and H_{11} were selected since they performed best. In addition H_{20} was selected, because it had the lowest positive Q-value.

6. Conclusions and Future Work

In this article, we present a method for programming-by-demonstration of reaching motions which enables robotic grasping. To allow the robot to interpret both the human motions and

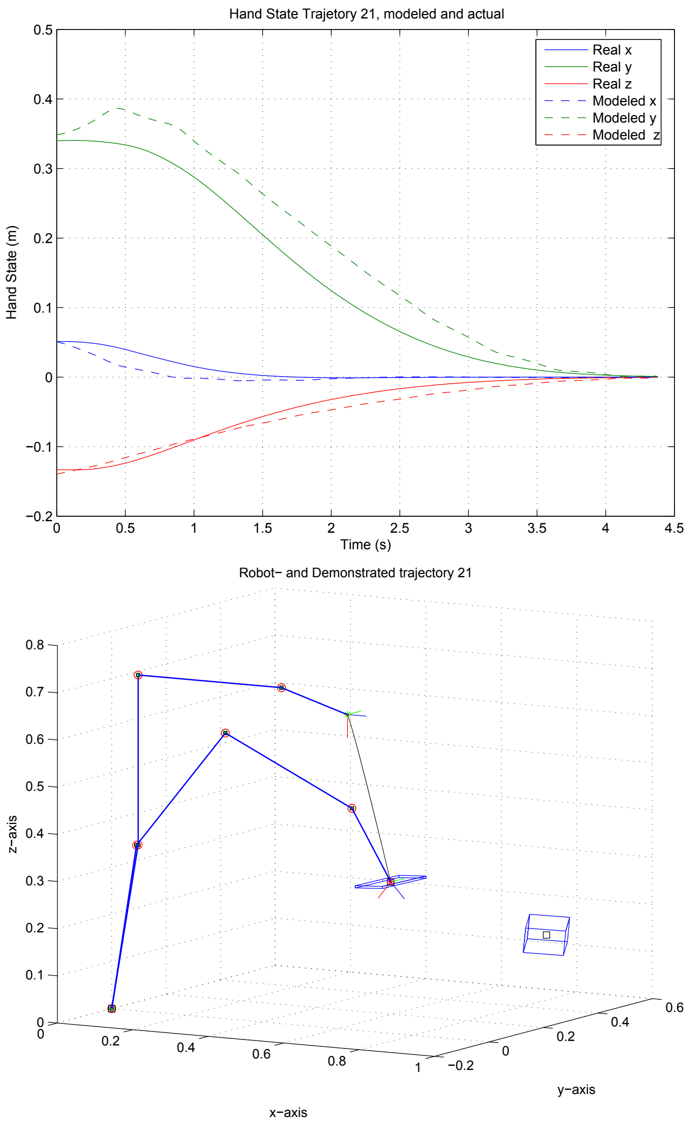


Fig. 12. Robot model R_1 and the generated reaching trajectory the robot executes.

its own in similar ways, we employ a hand-state space representation as a common basis between the human and the robot.
We presented the design of a NSP, which includes the advantages of fuzzy modeling and executes the motion in hand-state space.

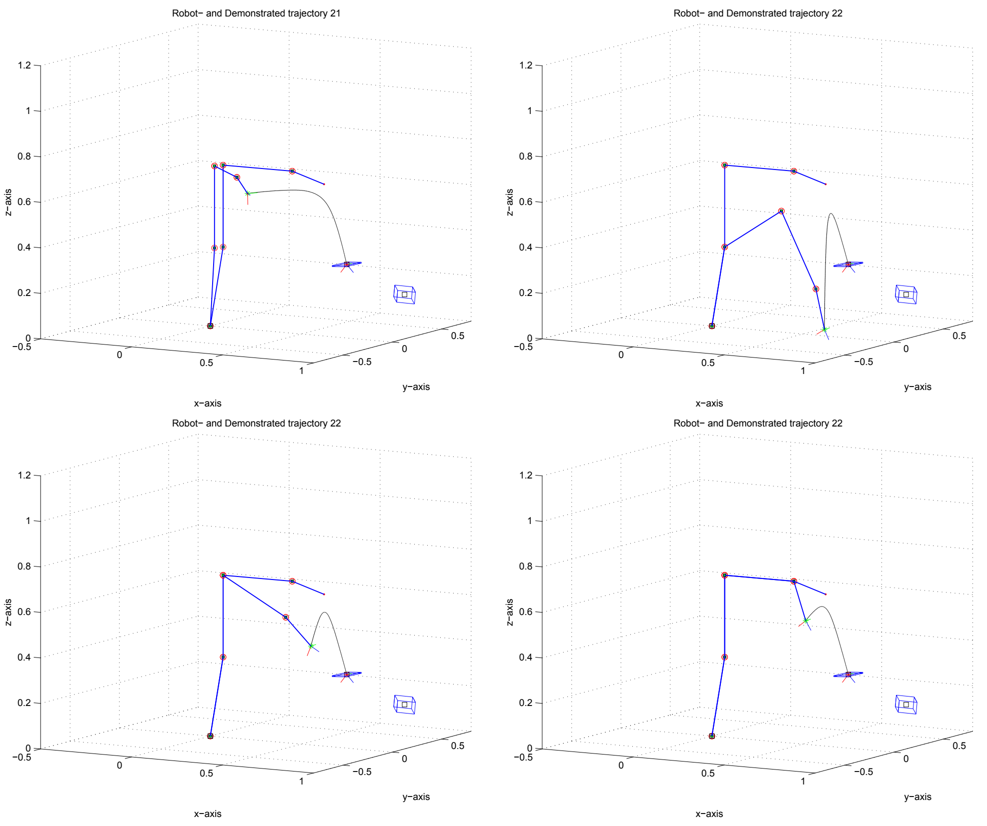
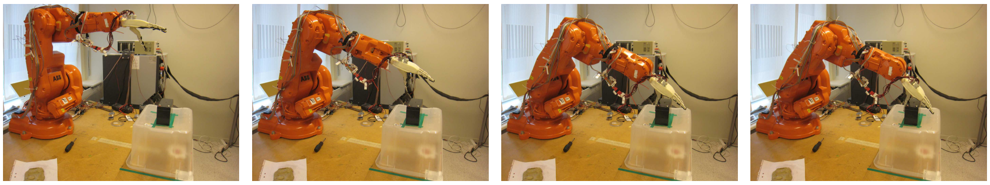


Fig. 13. Reaching from different initial positions. Four sample configurations, $\Theta = [-45\ 0\ 0\ 1\ 1\ 1]^T$, $\Theta = [0\ 20\ 40\ 1\ 1\ 1]^T$, $\Theta = [0\ 0\ 25\ 1\ 1\ 1]^T$ and $\Theta = [0\ 0\ 0\ 1\ 55\ 1]^T$.

Human demonstrations have been shown to provide sufficient knowledge to produce skill models good enough for the robot to use as its own skills.
It is shown that the suggested method can generate executable robot trajectories based on current and past human demonstrations despite morphological differences. The robot gains experience from human demonstration, and we have shown how the robot can improve when the selfexecuted motions are performed. The generalization abilities of the trajectory planner are illustrated by several experiments where an industrial robot arm executes various reaching motions and performs power grasping with a three-fingered hand.
To validate the performance of the skills learned from demonstration we included a metric which can be used in reinforcement learning. The performances of the initially learned skills were compared to the skills learned from self execution (robot skills). The result showed that robot skills indeed performed better than skills from human demonstration. In reinforcement learning the performance metric is used to formulate the reward function.
In our future work we plan to extend the theoretical and experimental work to include all feasible grasp types of the KTHand. To remedy the effect of the small workspace of the robot

R_2 from configuration $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$



R_2 from configuration $\Theta = [0\ 20\ 40\ 1\ 1\ 1]^T$

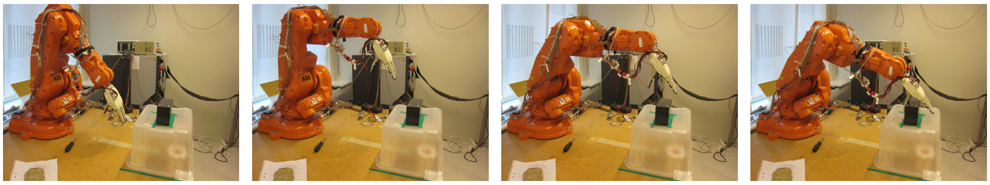


Fig. 14. The top series show the robot grasping the box shaped object from configuration $\Theta = [0\ 0\ 0\ 0\ 0\ 0]^T$ using robot skill R_2 . The bottom graph show the same skill R_2 but from a different initial position than the trained one: $\Theta = [0\ 20\ 40\ 1\ 1\ 1]^T$.

a different workspace configuration will be used. Furthermore, the robot's own perception will be incorporated into the loop to enable the robot to learn from its own experience.

Acknowledgment: Johan Tegin at Mechatronics Laboratory, at the Royal Institute of Technology, Stockholm, should be acknowledged for providing access to the KTHand. Krzysztof Charusta and Per Sporrong at the Center for Applied Autonomous Sensor Systems (AASS) at Örebro University, should be acknowledged for their help with the experiments.

7. References

Aleotti, J. & Caselli, S. (2006). Robust trajectory learning and approximation for robot programming, *Robotics and Autonomous Systems* **54**(5): 409–413.

Argall, B. D., Chernova, S., Veloso, M. & Browning, B. (2009). A survey of robot learning from demonstration, *Robotics and Autonomous Systems* **57**(5): 469–483.

Bandera, J. P., Marfil, R., Molina-Tanco, L., Rodríguez, J. A., Bandera, A. & Sandoval, F. (2007). *Humanoid Robots: Human-like Machines*, Itech, Vienna, Austria, chapter Robot Learning by Active Imitation, pp. 519–535.

Billard, A., Calinon, S., Dillmann, R. & Schaal, S. (2008). *Springer handbook of Robotics*, Springer, chapter Robot Programming by Demonstration, pp. 1371–1394.

Billard, A., Epars, Y., Calinon, S., Schaal, S. & Cheng, G. (2004). Discovering optimal imitation strategies, *Robotics and Autonomous Systems* **47**(2–3): 69–77.

Bullock, D. & Grossberg, S. (1989). VITE and FLETE: Neural modules for trajectory formation and postural control, *in* W. A. Hershberger (ed.), *Volitonal Action*, Elsevier Science Publishing B.V. (North-Holland), pp. 253–297.

Calinon, S., Guenter, F. & Billard, A. (2007). On learning, representing, and generalizing a task in a humanoid robot, *IEEE Transactions on Systems, Man and Cybernetics, Part B* **37**(2): 286–298.

Charusta, K., Dimitrov, D., Lilienthal, A. J. & Iliev, B. (2009). Grasp-related features extraction by human dual-hand object exploration, *Proceedings of the 14:th International Conference on Advanced Robotics*.

Delson, N. & West, H. (1996). Robot programming by human demonstration: Adaptation and inconsistency in constrained motion, *IEEE International Conference on Robotics and Automation*, pp. 30–36.

Gustafson, D. & Kessel, W. (1979). Fuzzy clustering with a fuzzy covariance matrix., *Proceedings of the 1979 IEEE CDC* pp. 761–766.

Hersch, M. & Billard, A. G. (2008). Reaching with multi-referential dynamical systems, *Autonomous Robots* **25**(1–2): 71–83.

Ijspeert, A. J., Nakanishi, J. & Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots, *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 1398–1403.

Ijspeert, A., Nakanishi, J. & Schaal, S. (2001). Trajectory formation for imitation with nonlinear dynamical systems, *IEEE International Conference on Intelligent Robots and Systems (IROS 2001)*, Vol. 2, pp. 752–757.

Iliev, B., Kadmiry, B. & Palm, R. (2007). Interpretation of human demonstrations using mirror neuron system principles, *Proceedings of the 6th IEEE International Conference on Development and Learning*, Imperial College London, pp. 128–133.

Iossifidis, I. & Schöner, G. (2006). Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for an redundant robot arm, *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, pp. 580–585.

Levine, W. S. (1996). The root locus plot, *in* W. S. Levine (ed.), *The Control Handbook*, CRC Press, chapter 10.4, pp. 192–198.

Nehaniv, C. L. & Dautenhahn, K. (2002). The correspondence problem, *in* K. Dautenhahn & C. Nehaniv (eds), *Imitation in Animals and Artifacts*, The MIT Press, Cambridge, MA, pp. 41–61.

Oztop, E. & Arbib, M. A. (2002). Schema design and implementation of the grasp-related mirror neurons, *Biological Cybernetics* **87**(2): 116–140.

Palm, R., Driankov, D. & Hellendoorn, H. (1997). *Model Based Fuzzy Control*, Springer.

Palm, R. & Iliev, B. (2006). Learning of grasp behaviors for an artificial hand by time clustering and Takagi-Sugeno modeling, *Proceedings of the IEEE International Conference on Fuzzy Systems*, Vancouver, BC, Canada, pp. 291–298.

Palm, R. & Iliev, B. (2007). Segmentation and recognition of human grasps for programming-by-demonstration using time-clustering and fuzzy modeling, *Proceedings of the IEEE International Conference on Fuzzy Systems*, London, UK.

Palm, R., Iliev, B. & Kadmiry, B. (2009). Recognition of human grasps by time-clustering and fuzzy modeling, *Robotics and Autonomous Systems* **57**(5): 484–495.

Palm, R. & Stutz, C. (2003). Generation of control sequences for a fuzzy gain scheduler, *International Journal of Fuzzy Systems* **5**(1): 1–10.

Pardowitz, M., Knoop, S., Dillmann, R. & Zöllner, R. D. (2007). Incremental learning of tasks from user demonstrations, past experiences, and vocal comments, *IEEE Transactions on Systems, Man and Cybernetics, Part B* **37**(2): 322–332.

Shadmehr, R. & Wise, S. P. (2005). *Computational Neurobiology of Reaching and Pointing - A Foundation for Motor Learning*, Computational Neuroscience, The MIT Press.

Shoemake, K. (1985). Animating rotation with quaternion curves, *ACM SIGGRAPH Computer Graphics* **19**(3): 245–254.

Skoglund, A., Iliev, B., Kadmiry, B. & Palm, R. (2007). Programming by demonstration of pick-and-place tasks for industrial manipulators using task primitives, *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Jacksonville, Florida, pp. 368–373.

Skoglund, A., Iliev, B. & Palm, R. (2008). A hand state approach to imitation with a next-state-planner for industrial manipulators, *Proceedings of the 2008 International Conference on Cognitive Systems*, University of Karlsruhe, Karlsruhe, Germany, pp. 130–137.

Skoglund, A., Iliev, B. & Palm, R. (2009). Programming-by-demonstration of reaching motions – a next-state-planner approach, *Robotics and Autonomous Systems* **In press.**

Skoglund, A., Tegin, J., Iliev, B. & Palm, R. (2009). Programming-by-demonstration of reach to grasp tasks in hand-state space, *Proceedings of the 14:th International Conference on Advanced Robotics*, Munich, Germany.

Takagi, T. & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control., *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-15**(1): 116–132.

Takamatsu, J., Ogawara, K., Kimura, H. & Ikeuchi, K. (2007). Recognizing assembly tasks through human demonstration, *International Journal of Robotics Research* **26**(7): 641–659.

Tegin, J., Ekvall, S., Kragic, D., Wikander, J. & Iliev, B. (2009). Demonstration based learning and control for automatic grasping, *Journal of Intelligent Service Robotics* **2**(1): 23–30.

Tegin, J., Wikander, J. & Iliev, B. (2008). A sub €1000 robot hand for grasping – design, simulation and evaluation, *International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, Coimbra, Portugal.

Ude, A. (1993). Trajectory generation from noisy positions of object features for teaching robot paths, *Robotics and Autonomous Systems* **11**(2): 113–127.

Vijayakumar, S., D'Souza, A. & Schaal, S. (2005). Incremental online learning in high dimensions, *Neural Computation* **17**(12): 2602–2634.

Robot Arms with 3D Vision Capabilities

Theodor Borangiu and Alexandru Dumitrache
Politehnica University of Bucharest
Romania

1. Introduction

The use of industrial robots in production started a rapid expansion in the 1980s, since they had the possibility of improving productivity, being able to work for extended periods of time with good repeatability, therefore making the quality of products stable. However, since the first robots worked "blindly", on a pre-programmed trajectory, dedicated equipment had to be prepared only for supplying the workpieces to the robots (Inaba & Sakakibara, 2009). Also, human operators had to manually align the workpieces before the robot was able to manipulate them.
The *intelligent robots* appeared later in 2001 in order to solve this problem. An intelligent industrial robot is not a humanoid robot that walks and talks like a human, but one that performs complex tasks, similar to a skilled worker. This is achieved with sensors (vision, force, temperature etc) and artificial intelligence techniques.
Today, solutions to problems like picking parts placed randomly in a bin (*bin picking*), which were considered difficult a few years ago, are now considered mature: (Hardin, 2008) and (Iversen, 2006). A similar problem is *auto racking*, where robots have to pick parts which are presented one at a time, although the exact location and 3D orientation varies. These applications are made possible using 3D vision sensors.

1.1 3D vision sensors

Two types of vision sensors are used on the factory floor: two-dimensional (2D) and three-dimensional (3D). The 2D sensors are usually similar to a digital photographic camera, being able to capture an image of the workpiece and obtain the position and rotation angle of the object. This works well for parts that can sit on a flat surface, and enables them to be picked by a SCARA robot, for example.
There are two major methods for 3D vision sensors:

- Structured light
- Stereoscopic vision

1.2 Structured light sensors

Structured light sensors have a common principle: projecting a narrow band of light onto a three-dimensionally shaped surface produces a line of illumination that appears distorted when viewed from other perspectives than that of the projector. The shape of the line of illumination can be captured by a 2D camera, allowing the exact geometric reconstruction of the surface shape using the *triangulation* method.

The simplest sensor using the triangulation principle is the *range finder*, which measures the distance to the closest reflective object (Fig. 1). A pulse of light (laser, regular visible light or infrared) is emmited and then reflected back on a linear CCD array. The position of the reflected light on the CCD array can be used to compute the distance to the closest object. This kind of sensor is also affordable to robot hobbists (Palmisano, 2007).

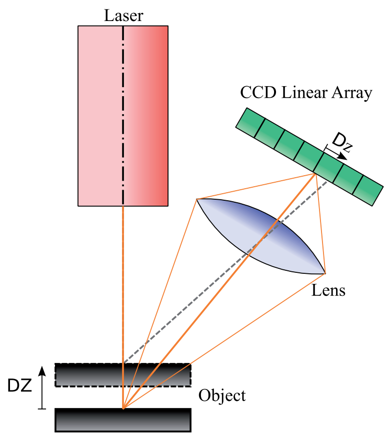


Fig. 1. Triangulation-based laser range finder

A more complex sensor is the *profile scanner*, which projects a narrow stripe of laser light onto the surface being digitized. A 2D camera, placed at a known angle with respect to the laser plane, records the image of the laser stripe and computes the local geometrical shape of the surface. For reconstructing a full 3D model of the workpiece, the profile scanner has to be swept around the part. The most precise way is to use a coordinate measuring machine (CMM). The sensors can also be mounted on industrial robots, which are more flexible in positioning and orienting the sensor, but also less accurate. Laser-based vision systems can generate very accurate 3D surface maps of the digitized parts, depending on the quality of the components used, but can be slow, since the sensor has to be moved continuously around the workpiece.
A faster method is projecting a pattern consisting of more light stripes. A typical setup for 3D measuring has a stripe projector, which is similar to a video projector, and at least one camera. Common setups include two cameras in opposite sides of the projector.
The depth can be reconstructed by analyzing the stripe patterns recorded by the camera, and several algorithms are available. The displacement of any single stripe can be converted into 3D coordinates; this involves identifying the stripes, either by tracing or counting stripes; ambiguities may appear when the workpiece contains sharp vertical walls. Another method involves alternating stripe patterns, resulting in binary sequences. Depth may also be computed by variations in the stripe width along the surface, and also by frequency / phase analysis of the stripe pattern by means of Fourier or Wavelet transforms. Practical implementation combine these methods in order to reconstruct a complete and unambiguous model of the surface.

1.3 Stereoscopic Vision

Stereoscopic vision attempts to compute the third dimension (the depth) in a way similar to the human brain. A 3D binocular stereo vision system uses two cameras which take images of the same scene from different positions (Fig. 2), and then computes the 3D coordinates for each pixel by comparing the parallax shifts between the two images.

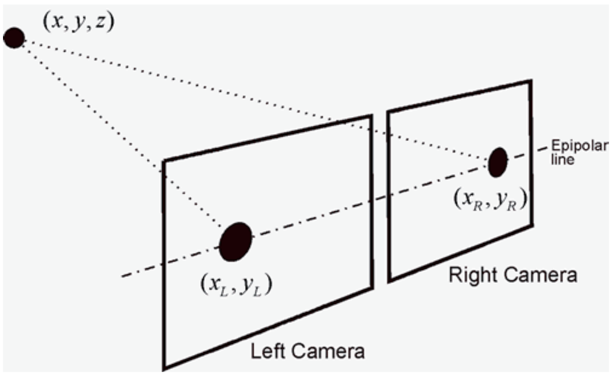


Fig. 2. Stereo vision principle: two cameras, which view the same scene, detect a common 3D point on different 2D locations

The main process in stereoscopic vision is the *stereo correspondence* between the two images, which is used to estimate *disparities* (differences in image locations of an object recorded by the two cameras) Calin & Roda (2007). The disparity is negatively correlated with the distance from the cameras: as the distance from the camera increases, the disparity decreases. Using geometry and algebra, the points that appear in the 2D stereo images can be mapped as coordinates in 3D space.
The stereo matching problem was, and continues to be, one of the most active research areas in computer vision. Several algorithms were developed; a classification and comparative benchmark for dense two-frame correspondence algorithms is presented in (Scharstein & Szeliski, 2002).
Aside from depth perception, stereo matching is also used in mobile robotics, where comparing two succesive images taken with the same camera leads to estimation of the motion of the robot.
For robotic applications, which require 6-DOF part localization in 3D space, stereo vision is considered also a mature technology: (Hardin, 2008) and (Iversen, 2006). 3D part localization is also possible even with a single 2D camera, using a trained model of the part, and the approach is already used in production (Iversen, 2006). An experiment showing the integration of a binocular stereo vision system with an industrial robot is presented by Cheng & Chen (2008).
The main advantage of stereo vision techniques is that they do not require additional light sources, and therefore, these techniques are non-invasive with respect to the surrounding environment.

2. 3D reconstruction system

An application involving a 6-DOF robot arm and a profile scanning device is presented in this section. The structure of the system can be seen in Fig. 3(a). The main components are:

- Short range laser probe, able to measure distances between 100 and 200 mm with 30 μm accuracy, using one laser beam and two CCD cameras;

- 6-DOF vertical robot arm, with 650 mm reach and 20 μm repeatability;
- 1-DOF rotary table, for holding the workpiece being scanned;
- 4-axis CNC milling machine, for reproduction of the scanned parts.

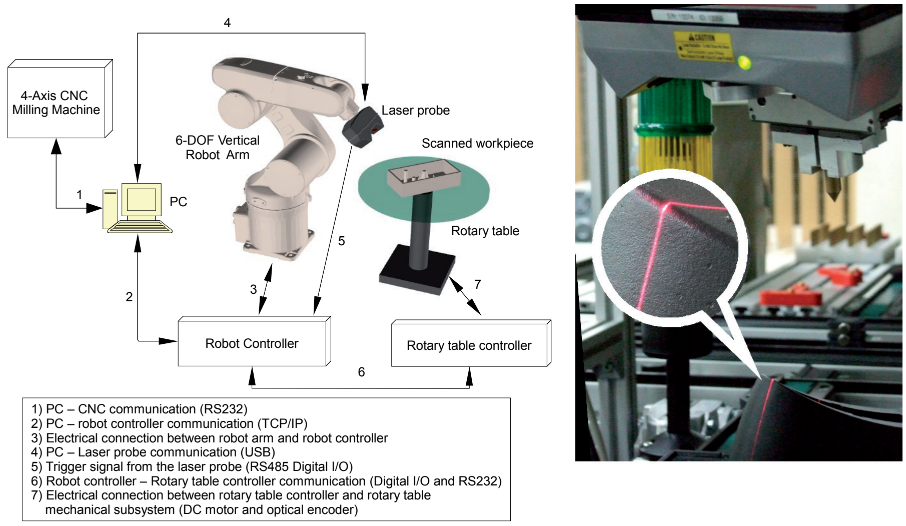

4
4-Axis CNC Milling Machine
6-DOF Vertical Robot Arm
Laser probe
Scanned workpiece
1
PC
Rotary table
5
2
3
7
Robot Controller
Rotary table controller
6
1) PC – CNC communication (RS232)
2) PC – robot controller communication (TCP/IP)
3) Electrical connection between robot arm and robot controller
4) PC – Laser probe communication (USB)
5) Trigger signal from the laser probe (RS485 Digital I/O)
6) Robot controller – Rotary table controller communication (Digital I/O and RS232)
7) Electrical connection between rotary table controller and rotary table mechanical subsystem (DC motor and optical encoder)

Fig. 3. (a) Overview of the 3D scanning system; (b) Laser sensor scanning a dark surface

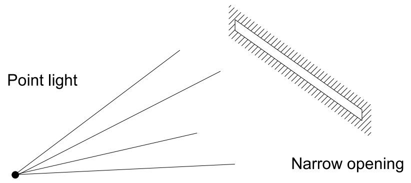
2.1 Simulation platform

Before installing the physical hardware, a software simulation platform was developed in order to test the scanning strategies, develop the motion planning algorithms and analyze the laser sensor behavior in controlled situations.
The simulator, presented in detail in (Borangiu et al., 2008a), has two components:

- Robot motion simulation, which uses a 7-DOF kinematic model based on Denavit-Hartenberg convention (Spong et al., 2005) and renders individual rigid meshes for each DOF of the system;
- Optical simulation of the laser sensor in a virtual 3D world, using raytracing.

The simulator has two modes of operation: static and dynamic simulation. In the static mode, the robot maintains the position of the laser sensor fixed, and the two CCD image sensors show a simulated image. In dynamic mode, the user can specify complex scanning trajectories which will be followed. The result from the laser sensor is analyzed and a point cloud model, representing the scanned virtual part, is created. The simulator is also capable of exporting animations with the scanning system following a predefined program.
The profile scanner emits a laser beam focused into a plane, which, projected on a surface, is reflected on the image sensor as a line or a curve. This laser beam may be modelled as a point light source, which is constrained to pass to a narrow opening (Fig. 4). In POV-Ray, a freeware raytracing software package (POV-Ray, 2003), the laser beam can be simulated with the code from Fig. 4 (b). Here, the laser rays start from origin, are projected in the positive direction of

the Z axis and the laser rays are going to be focused in YZ plane. The narrow opening has the dimensions L (large edge) and W (small edge) and is located at a distance D from the origin.



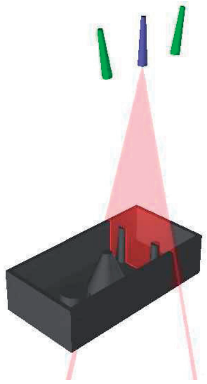
(a) Laser beam modelling principle

```
light_source {
    <0, 0, 0>
    color rgb <1, 0, 0>
    projected_through {
        box {
            <-W/2, -L/2, D>,
            < W/2,  L/2, D + eps>
        }
    }
}
```

(b) POV-Ray example

Fig. 4. Laser beam modelling using POV-Ray

The cameras used in the laser probe are be modeled as two standard perspective cameras, which may be implemented in POV-Ray by entering their parameters such as position, orientation and focal length. In the following text, only one of the two cameras will be described, as the other one is identical and symmetrical to the first one. For converting the image data into 3D coordinates, a pinhole camera model (Peng & Gupta, 2007) is used.


(a)

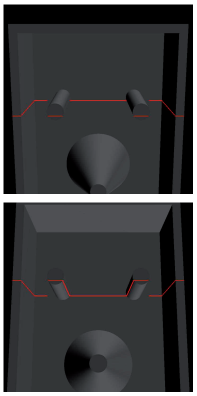
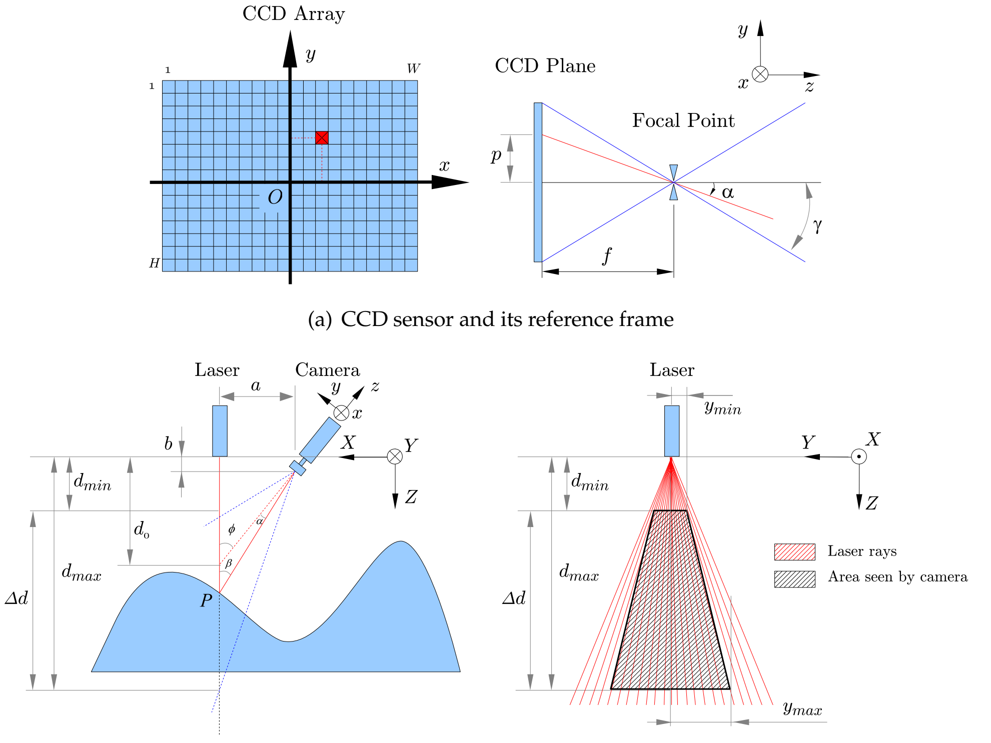

(b)

Fig. 5. Laser sensor simulation: (a) planar laser beam, two cameras and a virtual workpiece; (b) Simulated images obtained from the two cameras using raytracing

Let XYZ be the reference frame of the laser probe (Fig. 6(b) and 6(c)), and let xyz be the reference frame of the CCD array from the camera (Fig. 6(a) and 6(b)). Referring to Fig. 6(b), the camera position and orientation with respect to the laser device is given by three scalar parameters: a, b and ϕ.



(a) CCD sensor and its reference frame

(b) Side view of the laser probe

(c) Front view of the laser probe

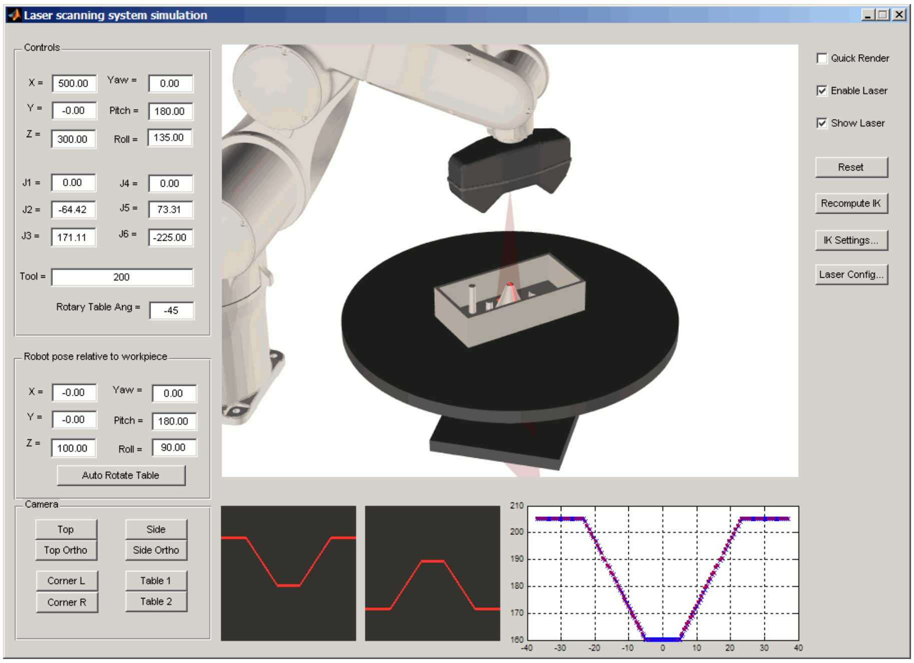
Fig. 6. Triangulation

Using these notations, let $P = (P_X; P_Y; P_Z)$ the point of reflection of a laser ray, in the XYZ reference frame, and let $p = (p_x; p_y)$ be the coordinate of the pixel at which the ray was detected on the CCD matrix, in xy reference frame. Knowing the 2D pixel coordinates p, the location of the 2D point P can be expressed using the triangulation equations (1):

$$P_X = 0 \qquad P_Y = \frac{a}{f\ \sin\left(\phi - \arctan\frac{p_y}{f}\right)}\ p_x \qquad P_Z = \frac{a}{f\ \tan\left(\phi - \arctan\frac{p_y}{f}\right)} + b \qquad (1)$$

where $f = \dfrac{H}{2\tan\gamma}$ if the unit length is considered to be 1 pixel, i.e. the distance between two adjacent pixels on the CCD array.

These equations are valid only under ideal conditions, i.e. when the camera and the laser sensor are perfectly aligned and there are no optical distortions from the camera lens. In practice, the transformation for converting the 2D pixel coordinates into 3D data expressed in milimetres is obtained using a calibration procedure. A look-up table model accounts for any nonlinear errors, especially lens distortion, and physical alignment between the camera and the laser is tuned using scalar offset parameters.


Laser scanning system simulation

Controls

X = 500.00 Yaw = 0.00
Y = -0.00 Pitch = 180.00
Z = 300.00 Roll = 135.00

J1 = 0.00 J4 = 0.00
J2 = -64.42 J5 = 73.31
J3 = 171.11 J6 = -225.00

Tool = 200

Rotary Table Ang = -45

Robot pose relative to workpiece

X = -0.00 Yaw = 0.00
Y = -0.00 Pitch = 180.00
Z = 100.00 Roll = 90.00

☐ Quick Render
☑ Enable Laser
☑ Show Laser

Fig. 7. Screenshot of the laser scanning system simulator

2.2 Integration Issues

A 3D point cloud model of the scanned part can be obtained by combining the measurements from the sensor with the instantaneous position of the robot. Aside from mechanical and electrical connections between the sensor and the laser probe, the two devices have to be synchronized. There are two operating modes supported by the sensor:

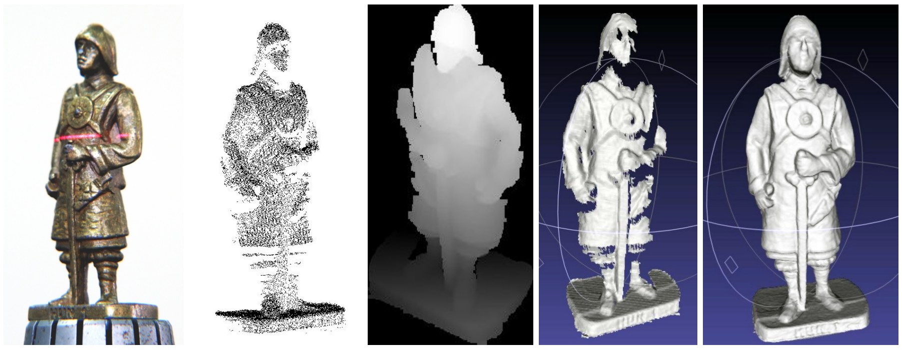
- Stop and look
- Buffered synchronization

With the first method, the measurements from the sensor are read only when the robot is not moving, and has reached its programmed destination. The method is the easiest to implement, does not need any synchronization signals between the vision sensor and the robot, but it is also the slowest, being limited at around 1 or 2 sensor readings per second. It is used only for debugging purposes.
The second method requires a trigger signal, which in the current implementation is sent from the vision sensor in the middle of the exposure period, from the sensor to the robot. When receiving the signal, the robot latches its instantaneous position, and stores it in a buffer. The trigger signal may be reversed, so the robot activates the vision sensor. The data from the robot and the sensor is collected on the PC and processed at a later time. This method allows sensor readings to be taken while the robot is still in motion, and close-spaced measurements can be taken at much higher rates, e.g. 50 readings/second. However, there may be a significant delay from the of data acquisition until the data is processed by the PC. In the system used here, the bottleneck is the Ethernet link between the robot and the PC, and the delay is usually

0.2 ... 0.3 seconds, and could reach 1 second. The buffers ensure that the data is matched properly even when high delays occur in communication.
Since the system also has a 7^{th} degree of freedom, the rotary table, its controller has to be also synchronized with the robot. If the table does not rotate while the sensor is actually scanning, there is no need for additional synchronization. If the table would have to rotate while scanning takes place, the rotary table controller would have to also listen to the trigger signal.
Another issue in integrating the three components (sensor, robot and turntable) is the calibration. For accurate 3D reconstruction, the system has to know, at every moment, the position of the table and the position of the sensor, both relative to robot base. Calibration issues are discussed in detail in (Borangiu et al., 2008b) and (Borangiu et al., 2009a).

2.3 Surface Reconstruction from Point Cloud

The raw output from the laser scanning system is a *point cloud* model, consisting of a huge and disorganised set of 3D points (X, Y and Z coordinates). This format is rarely used in practice; other models can be derived from it, such as the depth map or the polygon mesh.
An example of 3D reconstruction is given in Fig. 8, when the scanned part was a small decorative object having 40 mm height. The scanning procedure was done in 16 passes, i.e. 8 passes looking at the part from above and 8 passes from below. In each scan pass, the laser sensor was moved only in translation. Between two scan passes, the turntable was rotated in 45 degree increments in order to get a complete 3D representation of the part surface.
From each scan pass, the point cloud was transformed into a depth map using a straightforward approach, mapping the farthest point to black and the closest point to white. From the depth map it was possible to obtain a mesh by taking 4 adjacent pixels and forming a quadrilateral. The meshes from the 16 scans were stitched in MeshLab, an open source package for processing and editing large and unstructured 3D triangular meshes Cignoni (2008).



(a) (b) (c) (d) (e)

Fig. 8. 3D reconstruction example: (a) Photography of a decorative object, along with the laser stripe; (b) Point cloud from one scan pass; (c) Depth map computed from the point cloud; (d) Mesh model obtained from the depth map; (e) Complete 3D model, postprocessed in MeshLab

3. Automatic 3D Contour Following

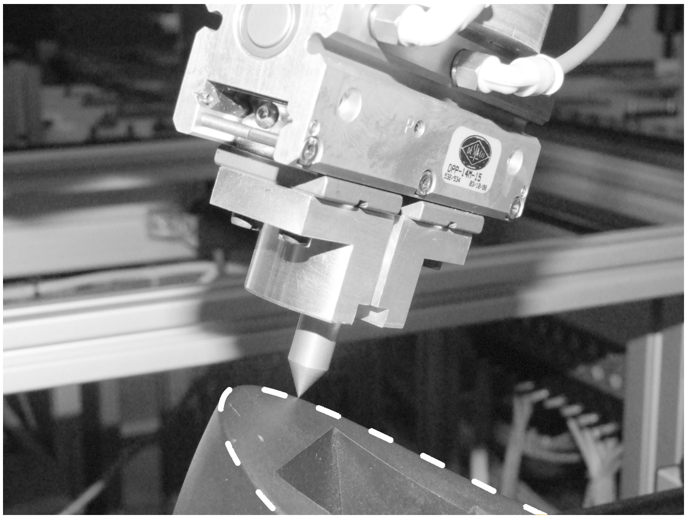
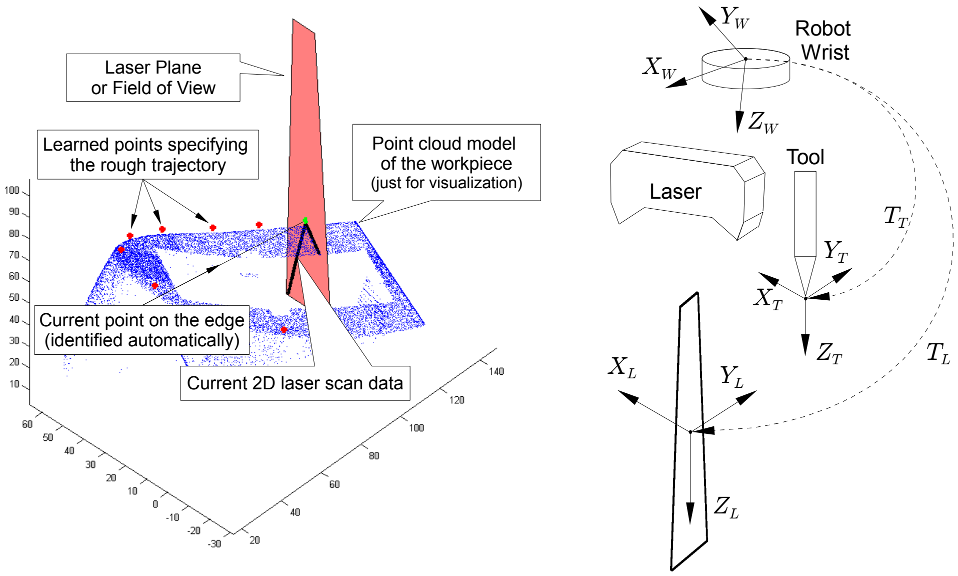


Fig. 9. 3D contour following using a sharp tool tip


Laser Plane or Field of View
Learned points specifying the rough trajectory
Point cloud model of the workpiece (just for visualization)
Current point on the edge (identified automatically)
Current 2D laser scan data
Y_W
X_W
Z_W
Robot Wrist
Laser
Tool
T_T
Y_T
X_T
Z_T
T_L
X_L
Y_L
Z_L
Fig. 10. (a) Trajectory learning assisted by automatic edge recognition; (b) The two tool transformations: one for trajectory teaching, other for following it using the physical tool

A second application, described in (Borangiu et al., 2009b), uses the same profile sensor for teaching a complex 3D path which follows an edge of an workpiece, without the need to have a CAD model of the respective part. The 3D contour is identified by its 2D profile, and the robot is able to learn a sequence of points along the edge of the part. After teaching, the robot is able to follow the same path using a physical tool, in order to perform various technological operations, for example, edge deburring or sealant dispensing. For the experiment, a sharp tool was used, and the robot had to follow the contour as precisely as possible. Using the laser sensor, the robot was able to teach and follow the 3D path with a tracking error of less than 0.1 milimetres.
The method requires two tool transformations to be learned on the robot arm (Fig. 10(b)). The first one, T_L, sets the robot tool center point in the middle of the field of view of the laser sensor, and also aligns the coordinate systems between the sensor and the robot arm.
Using this transform, any homogeneous 3D point $P_{sensor} = (X, Y, Z, 1)$ detected by the laser sensor can be expressed in the robot reference frame (World) using:

$$P_{world} = T_{robot}^{DK}\ T_L\ P_{sensor} \tag{2}$$

where T_{robot}^{DK} represents the position of the robot arm at the moment of data acquisition from the sensor. The robot position is computed using direct kinematics.
The second transformation, T_T, moves the tool center point on the tip of the physical tool. These two transformations, combined, allow the system to learn a trajectory using the 3D vision sensor, having T_L active, and then following the same trajectory with the physical instrument by switching the tool transformation to T_T.
The learning procedure has two stages:

- Learning the coarse, low resolution trajectory (manually or automatically)
- Refining the accuracy by computing a fine, high resolution trajectory (automatically)

The coarse learning step can be either interactive or automatic. In the interactive mode, the user positions the sensor by manually jogging the robot until the edge to be tracked arrives in the field of view of the sensor, as in Fig. 10(a). The edge is located automatically in the laser plane by a 2D vision component. In the automatic mode, the user only teaches the edge model, the starting point and the scanning direction, and the system will advance automatically the sensor in fixed increments, acquiring new points. For non-straight contours, the curvature is automatically detected by estimating the tangent (first derivative) at each point on the edge.
The main advantage of the automatic mode is that it can run with very little user interaction, while the manual mode provides more flexibility and is advantageous when the task is more difficult and the user wants to have full control over the learning procedure.
A related contour following method, which also uses a laser-based optical sensor, is described in (Pashkevich, 2009). Here, the sensor is mounted on the welding torch, ahead of the welding direction, and it is used in order to accurately track the position of the seam.

4. Conclusions

This chapter presented two applications of 3D vision in industrial robotics. The first one allows 3D reconstruction of decorative objects using a laser-based profile scanner mounted on a 6-DOF industrial robot arm, while the scanned part is placed on a rotary table. The second application uses the same profile scanner for 3D robot guidance along a complex path, which is learned automatically using the laser sensor and then followed using a physical tool. While the laser sensor is an expensive device, it can obtain very good accuracies and is suitable for precise robot guidance.

5. References

Borangiu, Th., Dogar, Anamaria and A. Dumitrache (2008a), Modelling and Simulation of Short Range 3D Triangulation-Based Laser Scanning System, *Proceedings of ICCCC'08*, Oradea, Romania

Borangiu, Th., Dogar, Anamaria and A. Dumitrache (2008b), Integrating a Short Range Laser Probe with a 6-DOF Vertical Robot Arm and a Rotary Table, *Proceedings of RAAD 2008*, Ancona, Italy

Borangiu, Th., Dogar, Anamaria and A. Dumitrache (2009a), Calibration of Wrist-Mounted Profile Laser Scanning Probe using a Tool Transformation Approach, *Proceedings of RAAD 2009*, Brasov, Romania

Borangiu, Th., Dogar, Anamaria and A. Dumitrache, (2009b) Flexible 3D Trajectory Teaching and Following for Various Robotic Applications, *Proceedings of SYROCO 2009*, Gifu, Japan

Calin, G. & Roda, V.O. (2007) Real-time disparity map extraction in a dual head stereo vision system, *Latin American Applied Research*, v.37 n.1, Jan-Mar 2007, ISSN 0327-0793

Cheng, F. & Chen, X. (2008). Integration of 3D Stereo Vision Measurements in Industrial Robot Applications, *International Conference on Engineering & Technology*, November 17-19, 2008 – Music City Sheraton, Nashville, TN, USA, ISBN 978-1-60643-379-9, Paper 34

Cignoni, P. et. al., MeshLab: an Open-Source Mesh Processing Tool *Sixth Eurographics Italian Chapter Conference*, pp. 129-136, 2008.

Hardin, W. (2008). 3D Vision Guided Robotics: When Scanning Just WonâĂŹt Do, *Machine Vision Online*. Retrieved from `https://www.machinevisiononline.org/public/articles/archivedetails.cfm?id=3507`

Inaba, Y. & Sakakibara, S. (2009). Industrial Intelligent Robots, In: *Springer Handbook of Automation*, Shimon I. Nof (Ed.), pp. 349-363, ISBN: 978-3-540-78830-0, StÃijrz GmbH, WÃijrzburg

Iversen, W. (2006). Vision-guided Robotics: In Search of the Holy Grail, *Automation World*. Retrieved from `http://www.automationworld.com/feature-1878`

Palmisano, J. (2007). How to Build a Robot Tutorial, *Society of Robots*. Retrieved from `http://www.societyofrobots.com/sensors_sharpirrange.shtml`

Pashkevich, A. (2009). Welding Automation, In: *Springer Handbook of Automation*, Shimon I. Nof (Ed.), pp. 1034, ISBN: 978-3-540-78830-0, StÃijrz GmbH, WÃijrzburg

Peng, T. & Gupta, S.K. (2007) Model and algorithms for point cloud construction using digital projection patterns. *ASME Journal of Computing and Information Science in Engineering*, 7(4): 372-381, 2007.

Persistence of Vision Raystracer Pty. Ltd., *POV-Ray Online Documentation*

Scharstein, D. & Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1/2/3):7-42, April-June 2002.

Spong, M. W., Hutchinson, S., Vidyasagar, M. (2005). Robot Modeling and Control, *John Wiley and Sons, Inc.*, pp. 71-83, 2005

Robot assisted 3D shape acquisition by optical systems

Cesare Rossi, Vincenzo Niola, Sergio Savino and Salvatore Strano
University of Naples "Federico II"
ITALY

1. Introduction

In this chapter, a short description of the basic concepts about optical methods for the acquisition of three-dimensional shapes is first presented. Then two applications of the surface reconstruction are presented: the passive technique Shape from Silhouettes and the active technique Laser Triangolation. With both these techniques the sensors (telecameras and laser beam) were moved and oriented by means of a robot arm. In fact, for complex objects, it is important that the measuring device can move along arbitrary paths and make its measurements from suitable directions. This chapter shows how a standard industrial robot with a laser profile scanner can be used to achieve the desired d-o-f.
Finally some experimental results of shape acquisition by means of the Laser Triangolation technique are reported.

2. Methods for the acquisition of three-dimensional shapes

In this paragraph the computational techniques are described to estimate the geometric property (the structure) of the three-dimensional world (3D), starting from ist bidimensional projections (2D): the images. The shape acquisition problem (shape/model acquisition, image-based modeling, 3D photography) is introduced and all steps that are necessary to obtain true tridimensional models of the objects, are synthetized [1].
Many methods for the automatic acquisition of the shape object exist. One possible classification of the methods for shape acquisition is illustrated in figure 1.
In this chapter optical methods will be analyze. The principal advantages of this kind of techniques are the absence of contact, the rapidity and the economization. The limitations include the possibility of being able to acquire only the visible part of the surfaces and the sensibility to the property of the surfaces like transparency, brilliance and color.
The problem of image-based modeling or 3D photography, can be described in this way: the objects irradiate visible light; the camera capture this "light", whose characteristics depend on the lighting system of the scene, surface geometry, reflecting surface; the computer elaborates the light by means of opportune algorithms to reconstruct the 3D structure of the objects.

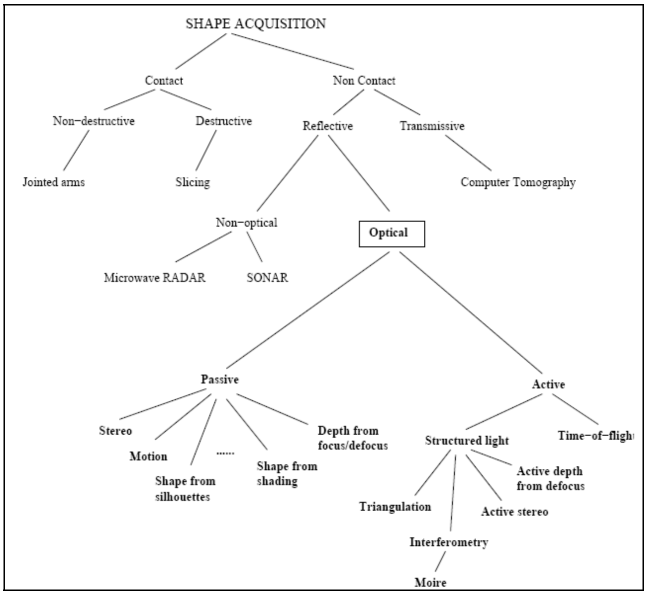


Fig. 1. Classification of the methods for shape acquisition [1]

In the figure 2 is shown an equipment for the shape acquisition by means two images.

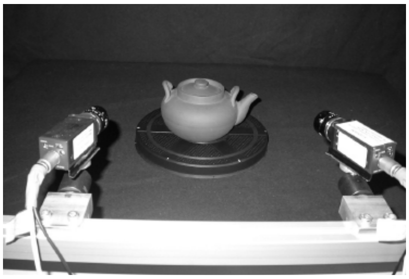


Fig. 2. Stereo acquisition

The fundamental distinction between the optical techniques for shape acquisition, regards the use of special lighting sources. In particular, it is possible to distinguish two kinds of optical methods: active methods, that modify the images of scene by means of opportune luminous pattern, laser lights, infrared radiations, etc., and passive methods, that analyze the images of the scene without to modify it. The active methods have the advantage to concur high resolutions, but they are more expensive and not always applicable. The passive methods are economic, they have fewer constraints obligatory, but they are characterized by lower resolutions.
Many of the optical methods for the shape acquisition have like result an image range, that is an image in which every pixel contains the distance from the sensor, of a visible point of the scene, instead of its brightness (figure 3). An image range is constituted by measures

(discrete) of a 3D surface respect to a 2D plan (usual the plane image sensor) and therefore it is also called: 2.5D image. The surface can be always expressed in the form Z = f(X, Y), if the reference plane is XY. A sensor range is a device that produces an image range.

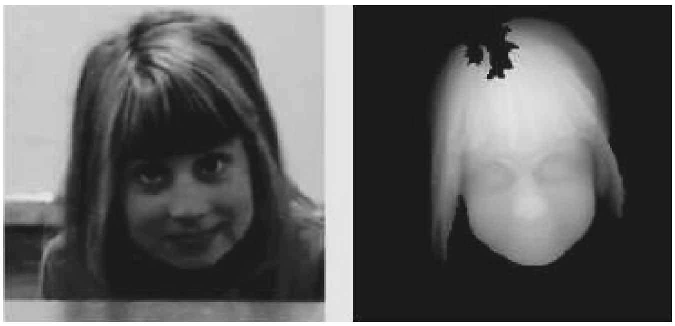


Fig. 3. Brightness reconstruction of an image [1]

Below optical sensor range is any optical system of shape acquisition, active or passive, that is composed of equipment and softwares and that gives back an image range of the scene. The main characteristics of a sensor range are:

- **resolution**: the smallest change of depth that the sensor can find;
- **accuracy**: diffrence between measured value (average of repeated measures) and true value (it measures the systematic error);
- **precision**: statistic variation (standard deviation) of repeated measures of a same quantity (dispersion of the measures around the average);
- **velocity**: number of measures in a second.

2.2 From the measure to the 3D model

The recovery of 3D information, however, does not exhaust the process of shape acquisition, even if it is the fundamental step. In order to obtain a complete model of an object, or of a scene, many images range are necessary, and they che they must be aligned and merged with each other to obtain a 3D surface (like poligonal mesh).

The reconstruction of the model of the object starting from images range, previews three steps:

- **adjustment**: (or alignment) in order to transform the measures supplied from the several images range in a one common reference system;
- **geometric fusion**: in order to obtain a single 3D surface (typically a poligonal mesh) starting various image range;
- **mesh simplification**: the points given back by a sensor range are too many to have a manageable model and the mesh must be simplified.

Below the first phase will be described above all, the second will be summarily and the third will be omitted.

An image range Z(X,Y) defines a set of 3D points (X,Y,Z(X,Y)), figure 4a. In order to obtain a surface in the 3D space (surface range) it is sufficient connect between their nearest points with triangular surfaces (figure 4b).

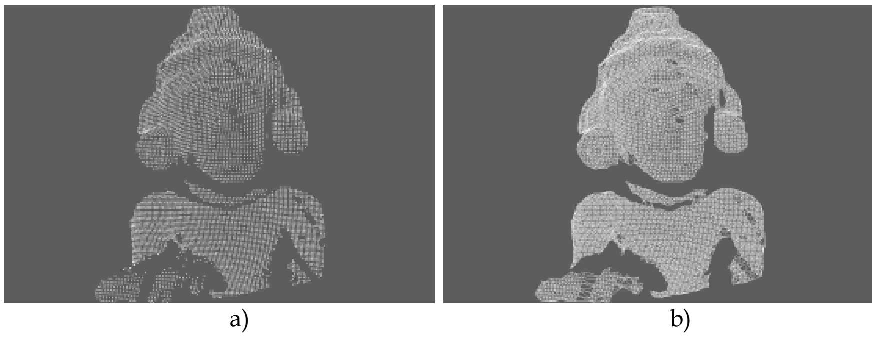


Fig. 4. Image range result (a) and its surface range (b) [1].

In many cases depth discontinuities can not be covered with triangles in order to avoid making assumptions that are unjustified on the shape of the surface. For this reason it is desirable to eliminate triangles with sides too long and those with excessively acute angles.

2.3 Adjustment

The sensors range don't capture the shape of an object with a single image, many images are needed, each of which captures a part of the object surface. The portions of the surface of the object are obtained by different images range, and each of them is made in its own reference system (that depends on sensor position).
The aim of adjustment is to expres all images in the same reference system, by means of an opportune rigid transformation (rotation and translation).
If the position and orientation of the sensor are known, the problem is resolved banally. However in many cases, the sensor position in the space is unknown and the transformations can be calculated using only images data, by means of opportune algorithms, one of these is ICP (Iterated Closest Point).
In the figure 5, on the left, eight images range of an object are shown, each in its own reference system; on the right, all images are were superimposed with adjustment operation.

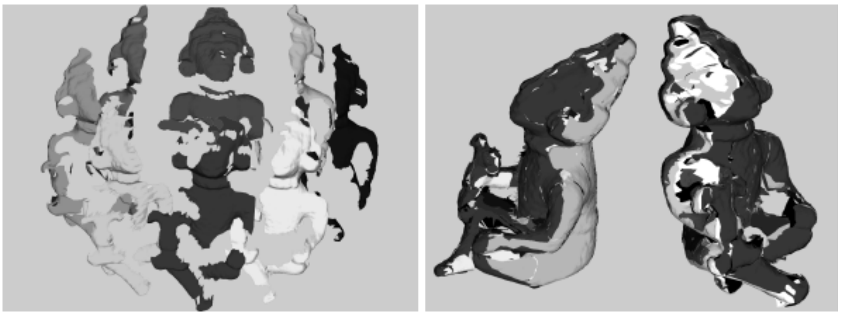


Fig. 5. Images range and result of adjustment operation [1]

2.4 Geometric fusion

After all images range data are adjusted in one reference system, they be united in a single shape, represented, as an example, by triangular mesh. This problem of surface reconstruction, can be formulated like an estimation of the bidimensional variety which approximates the surface of the unknown object by starting to a set of 3D points. The methods of geometric fusion can be divided in two categories:

- **Integration of meshes**: the triangular meshes of the single surfaces range, are joined.
- **Volumetric fusion**: all data are joined in a volumetric representation, from which a triangular mesh is extracted.

2.4.1 Integration of meshes

The techniques of integration of meshes aim to merge several 3D overlapped triangular meshes into a single triangular mesh (using the representation in terms of surface range).
The method of Turk and Levoy (1994) merges overlapped triangular meshes by means of a technique named "zippering". The overlapping meshes are eroded to eliminate the overlap and then it is possible to use a 2D triangolation to sew up the edges. To make this the points of the two 3D surface close to edges, must be projected onto a plane 2D.
In the figure 6, on the left, two aligned surface are shown, and on the right, the zippering result is shown.

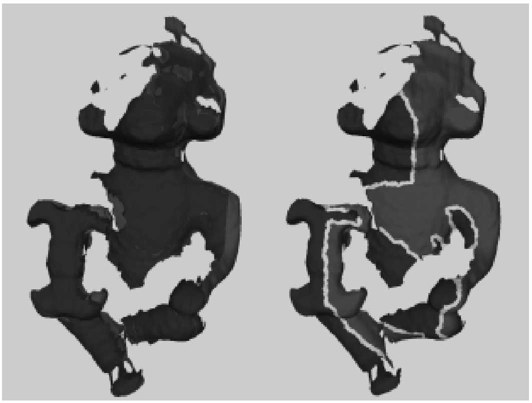


Fig. 6. Aligned surface and zippering result

The techniques of integration of meshes allow the fusion of several images range without losing accuracy, since the vertices of the final mesh coincide with the points of the measured data. But, for the same reason, the results of these techniques are sensitive to erroneous measurements, that may cause problems in the surface reconstruction.

2.4.2 Volumetric Fusion

The volumetric fusion of surface measurements constructs an intermediate implicit surface that combines the measurements overlaid in a single representation. The implicit

representation of the surface is an iso-surface of a scalar field f(x,y,z). As an example, if the function of field is defined as the distance of the nearest point on the surface of the object, then the implicit surface is represented by f(x,y,z) = 0. This representation allows modeling of the shape of unknown objects with arbitrary topology and geometry.
To switch from implicit representation of the surface to a triangular mesh, it is possible to use the algorithm Marching Cubes, developed by Lorensen e Cline (1987) for the triangulation of iso-surfaces from the discrete representation of a scalar field (as the 3D images in the medical field). The same algorithm is useful for obtaining a triangulated surface from volumetric reconstructions of the scene (shape from silhouette and photo consistency).
The method of Hoppe and others (1992) neglects the structure of the data (surface range) and calculates a surface from the unstructured "cloud" of points.
Curless and Levoy (1996) instead, take advantage of the information contained in the images range in order to assign the voxel that lie along the sight line that, starting from a point of the surface range, arrives to the sensor.
An obvious limitation of all geometric fusion algorithms based on an intermediate structure of discrete volumetric data is a reduction of accuracy, resulting in the loss of details of the surface. Moreover the space required for the volumetric representation grows quickly when resolution grows.

2.5 Optical methods for the shapes acquisition

All computational techniques use some indications in order to calculate the shape of the objects starting from the images. Below the main methods divided between active and passive, are listed.

Passive optical methods:

- depth from focus/defocus
- shape from texture
- shape from shading
- stereo-photometric
- stereopsis
- shape from silhouette
- shape from photo-consistency
- structure from motion

Active optical methods:

- active defocus
- active stereo
- active triangolation
- interferometry
- flight time

All the active methods, except the last one, employ one or two cameras and a source of special light, and fall in the wider class of the methods with **structured lighting system**.

3. Three-dimensional reconstruction with technique: Shape from Silhouettes

In this paragraph one of the passive optical techniques of 3D reconstruction will be introduced in detail, with some obtained results.

3.1 Principle of volumetric reconstruction from shapes

The aim of volumetric reconstruction is to create a representation that describes not only the surface of a region, but also the space that it encloses. The hypothesis is that there is a known and limited volume, in which the objects of interest lie. A 3D box is modelled to be an initial volume model that contains the object. This box is divided in discrete elements called voxels, that are three-dimensional equivalent of bidimensional pixel. The reconstruction coincides with the assignment of a label of occupation (or color) to each element of volume. The label of occupation is usually binary (transparent or opaque).
Volumetric reconstruction offers some advantages compared to traditional stereopsi techniques: it avoids the difficult problem of finding correspondences, it allows the explicit handling of occlusions and it allows to obtain directly a three-dimensional model of the object (it is not necessary to align parts of the model) integrating simultaneously all sights (which are the order of magnitude of ten).
Like in the stereopsi, the cameras are calibrated.
Below one of the many algorithms developed for the volumetric reconstruction from silhouettes of an object of interest is described: shape from silhouettes .
Shape From Silhouettes is well-known technique for estimating 3D shape from its multiple 2D images.
Intuitively the silhouette is the profile of an object, comprehensive of its inside part. In the “Shape from Silhouette” technique silhouette is defined like a binary image, which value in a certain point (x, y) underlines if the optical ray that passes for the pixel (x, y) intersects or not the object surface in the scene. In this way, Every point of the silhouette, respectively of value “1” or “0”, identifies an optical ray that intersects or not the object.
To store the labels of the voxel it is possible to use the octree data structure.The octree are trees with eight ways in which each node represents a part of space and the children nodes represents the eight divisions of that part of space (octants).

3.1.1 Szeliski algorithm

The Szeliski algorithm allows to build the volumetric model by means of octree structure.
The octree structure is constructed subdividing each cube in eight part (octants), starting from the root node that represents the initial volume. Each cube has associated a color:

- black: it represents a occupied volume;
- white: it represents an empty volume;
- gray: it represents an inner node whose classification is still uncertain.

For each octant, it is necessary to verify if its projection in image i, is entire contained in the black region. If that happens for all N shapes, the octant is labeled as black. If instead, the octant projection is entire contained in the background (white), also for a single camera, the octant is labeled as white. If one of these two cases happens, the octant becomes a leaf of the octree and it is not more tried, otherwise, it is labeled as gray and it is subdivided in eight parts. In order to limit the dimension of the tree, gray octants with minimal dimension, are

labeled as black. At the end of process it is possible to obtain an octree that represents 3D object structure.
An example of volumetric model construction with Szeliski algorithm is shown in figure 7.

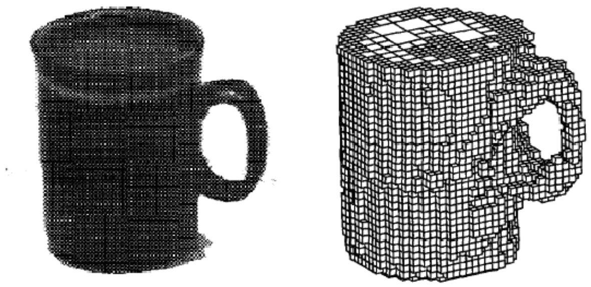


Fig. 7. Model reconstruction with Szeliski algorithm

3.1.2 Proposed method

The analysis method is to define a voxel box that contains the object in three dimensional space and to discard subsequently those points of the initial volume that have an empty intersection with at least one of the cones of the shapes obtained from the acquired images.
The voxel will have only binary values, black or white.
The algorithm is performed by projecting the center of each voxel into each image plane, by means of the known intrinsic and extrinsic camera parameters. If the projected point is not contained in the silhouette region, the voxel is removed from the object volume model.

3.2 Images elaboration

Starting from an RGB image (figure 8), it is analized and, by means of segmentation procedure, it is possible to obtain object silhouette reconstruction.

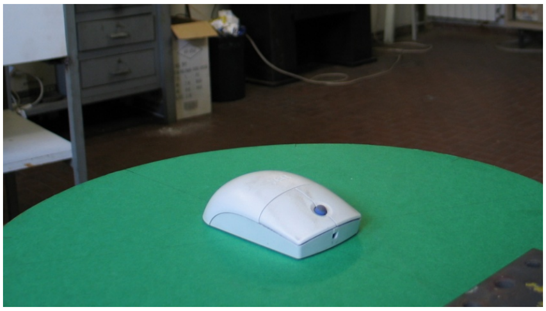


Fig. 8. RGB image

Segmentation is the process by means of which the image is subdivided in characteristics of interest. This operation is based on strong intensity discontinuities or regions that introduce homogenous intensity on the base of established criteria. There are four kinds of discontinuities: points, lines, edge, or, in a generalized manner, interest points.

In order to separate an object from the image background, the method of the threshold s is used: each point (u,v) with f(u,v) > s (f(u,v) < s) is identified like object, otherwise like background. The described elaboration is used to identify and characterize the various regions that are present in each image; in particular, by means of this technique it is possible to separate the objects from the background.
The first step is to transform RGB image in gray scale image (figure 9), in thi way the intensity value becomes a threshold to identify the object in the image.

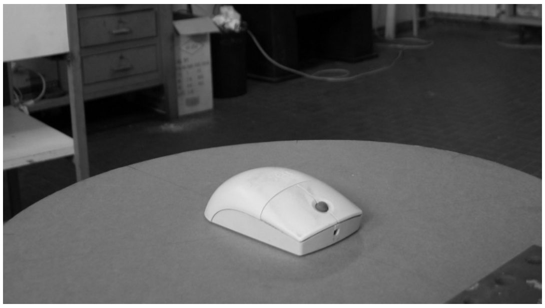


Fig. 9. Gray scale image

The gray scale image is a matrix, whose dimensions correspond to number of pixel along the two directions of the sensor, and whose elements have values in range [0,255]. Subsequently, a limited set of pixel that contains the projection of the object in the image plane, is selected, so it is possible to facilitate the segmentation procedure (figure 10).

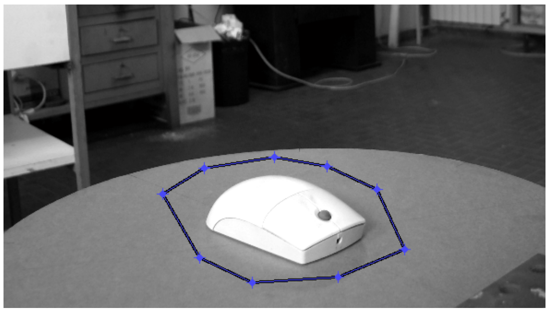


Fig. 10. Object identification

In gray scale image, pixel with intensity f(u,v) different from that of points that are not representative of object (background T), are chosen. This tecnique is very effective, if in the image there is a real difference between object and background.
For each pixel (u, v) unit value or zero, respectively, is assigned, if it is an optical beam passing through the object or not (figure 11).

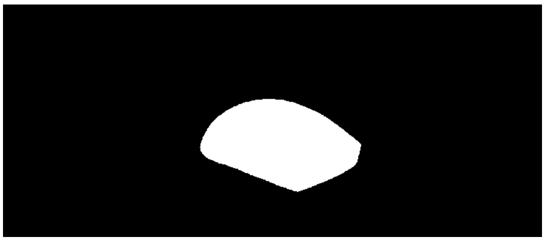


Fig. 11. The computed object silhouette region

3.3 Camera calibration

It is necessary to identify all model parameters in order to obtain a good 3-D reconstruction. Calibration is a basic procedure for the data analysis.
There are many kind of procedure to calibrate a camera system, but in this paragraph the study of the calibration procedures will not be discussed in detail.
The aim of calibration procedure is to obtain all intrinsic and estrinsic parameters of the camera system. Calibration procedure is based on a set of images, taken with a target placed in different positions that are known in a base reference system. A least square optimization allows to identify all parameters.

3.4 The proposed algorithm

The result of image elaboration is the object silhouette region for each image with pixel coordinates and centroid coordinates of these region in image reference system, (Fig. 11).
By means of calibration parameters (intrinsic and estrinsic), it is possible to evaluate, for each image, an homogeneous transformation matrix, between the image reference system of each image and a base reference system.
The first step is to discretize a portion of work space by mean an opportune box divided in voxels (figure 12). In this operation, it is necessary to choose the number of voxels, the dimension of box and its position in a base reference system. The position of voxel is chosen evaluating the intersections in base reference system, of the camera optical axis that pass through silhouette centroids of at least, two images. Subsequently it is possible to divide the initial volume model in a number of voxels according to the established precision, and it is possible to evaluate the centers of voxels in base reference.

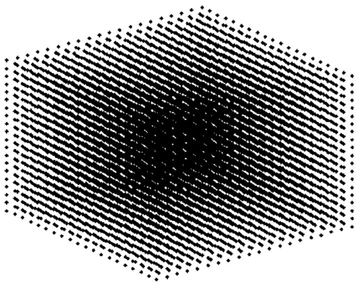


Fig. 12. Voxel discretization of workspace.

For each of the silhouettes, the projection of the centre of the voxels in the image plane can be obtained as follows:

$$\{\tilde{\phi}\}_j = [M]_i \cdot \{\tilde{w}\}_j \qquad i = 1,\dots,p \qquad j = 1,\dots,q \tag{1}$$

Where p is the number of silhouettes, q is the number of the voxels and $[M]_i$ is the transformation matrix between the base frame and the image frame.
The algorithm is performed by projecting the center of each voxel into each image plane, by means of the known intrinsic and extrinsic camera parameters. If the projected point is not contained in the silhouette region, the voxel is removed from the object volume model. This yields the following relation:

$$\begin{Bmatrix} u \\ v \\ 0 \\ 1 \end{Bmatrix}_j = [\tilde{\phi}]_j = [M]_p \cdot \{\tilde{w}\}_j \qquad j = 1,\dots,q \qquad 1 \le u \le n \qquad 1 \le v \le m \tag{2}$$

Where n and m are the number of pixels along the directions u and v, respectively. So a matrix [A] having dimension [n,m] is written; the values of the elements of this matrix is one if corresponds to a couple of coordinates (u,v) obtained by eq. (2) that belongs to the object silhouette in the image; otherwise the value is zero:

$$[A] := a_{hk} = 1 \qquad h = u_j, k = v_j \tag{3}$$

A set of pixels having coordinates $(\overline{u}, \overline{v})$, belonging to the first silhouette, is defined as follows:

$$\Omega_1 := (\overline{u}, \overline{v})_1 \tag{4}$$

The points of the image that belong to the silhouette are defined as follows:

$$I_1(u,v) = \begin{cases} 1 & (u,v) \subseteq \Omega_1 \\ 0 & (u,v) \not\subset \Omega_1 \end{cases} \qquad 1 \le u \le n \qquad 1 \le v \le m \tag{5}$$

By the product among the matrix defined in eq. (3) with the matrix in eq. (5), the following set of indexes is obtained:

$$\overline{j} : [A] \cdot [I_1] = 1 \qquad 1 \le u \le n \qquad 1 \le v \le m \tag{6}$$

Now the points (in the base frame) which indexes are integer numbers belonging to the set $\overline{j}$ are considered. These points are used as starting points to repeat the same operations, described by eq. (2), for all the other images. This procedure is recursive and is called space carving technique.

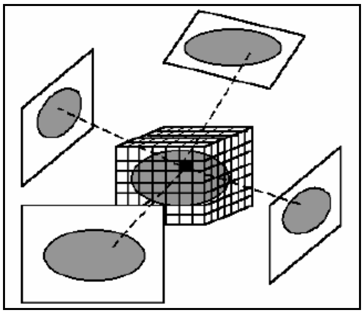


Fig. 13. Scheme of space carving technique.

3.5 Evaluation of the resoluction

The choosen number of voxels defines the resoluction of the reconstructed object.
Consider a volume which dimensions are l_x, l_y and l_z and a discretization along the three directions: Δ_x, Δ_y e Δ_z, the object resolutions that can be obtained in the three directions are:

$$a_x = \frac{l_x}{\Delta_x}; a_y = \frac{l_y}{\Delta_y}; a_z = \frac{l_z}{\Delta_z} \qquad (7)$$

In figure 14 two examples of reconstruction with different resolution choices are shown.

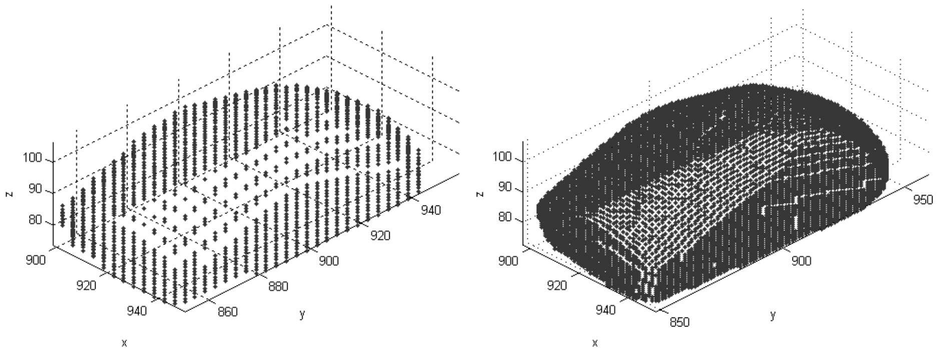


Fig. 14. Examples of reconstructions with different resolutions.

It must be pointed out that the choice of the initial box (resolution) depends on the optical sensor adopted. It is clear that an object near to the sensor will appear bigger than a far one, hence the error achieved for the same unit of discretization is increased with the distance for a given focal length.

$$a^c{}_u = \delta_u \cdot \frac{w_\varsigma}{f} ; a^c{}_v = \delta_v \cdot \frac{w_\varsigma}{f} \tag{8}$$

Obviously it must be checked that the choosen initial resolution is not bigger than the resolution that the optical sensor can achieve. To this pourpose, once the object has been reconstructed, the minimum dimension that can be recorded by the camera is computed by eq. (8); the latter is compared with the resolution initially stated.
This techique is rater slow because it is necessary to project each voxel of the initial box in the image plane of each of the photos. Moreover, a big number of phots is required and the procedure can not be made in real –time. It must also be noted, however, that this procedure can be used in a rater simple way in order to obtain a rough evaluation of the volume and of the shape of an object.
This techinque is very suitable to be assisted by a robot arm. Infact the accuracy of the reconstruction obtained depends on the number of images used, on the positions of each viewpoint considered, on the camera's calibration quality and on the complexity of the object shape. By positioning the camera on the robot, it is possible to know, exactly, not only the characteristics of the camera, but also the position of the camera reference frame in the robot work space. Therefore the camera intrinsic and extrinsic parameters are known without a vision system calibration and it's easy to make an elevated number of photos. That is to say, it could be possible to obtain vision system calibration, robot arm mechanical calibration and trajectories recording and planning.

4. Three-dimensional reconstruction by means of Laser Triangolation

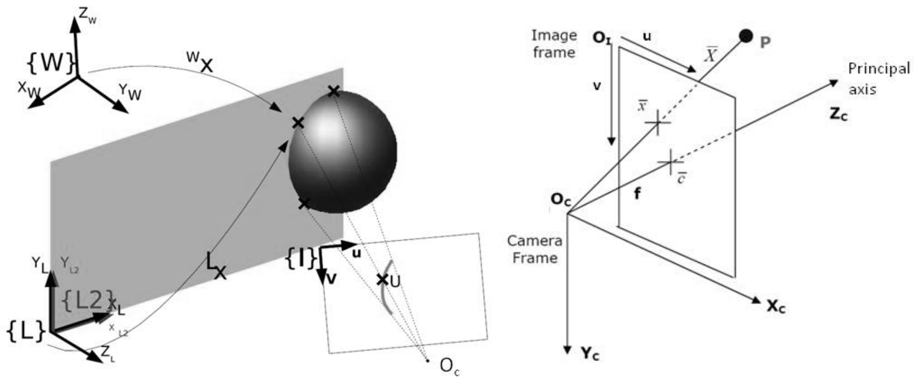
The laser triangulation technique maily permits higher operating speed with a satisfacting quality of reconstruction.

4.1 Principle of laser triangolation

In this paragraph is described a method for surface reconstruction, that uses of a linear laser emitter and a webcam, and uses triangulation principle applied to a scanning belt on object surface.
Camera observes the intersection between laser and object: laser line points in image frame, are the intersections between image plane and optical rays that pass through the intersection points between laser and object. By means of a transformation matrix, it is possible to express the image frame coordinates, in pixel, in a local reference frame. In figure 15 a) is shown a scheme of scanning system: {W} is local reference frame, {I} is image frame with coordinates system {u,v}, and {L} is laser frame. {L2} is laser plane that contains laser knife and scanning belt on the surface of object, and it coincides with (x,y) plane of laser frame {L}. Starting from the coordinates in pixel (u,v), in image frame, it is possible to write the coordinates of the scanning belt on the object surface, in camera frame by means of equation (9). Camera frame is located in camera focal point figure 15 b).

$$\begin{Bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{Bmatrix} = \begin{bmatrix} \delta_x & 0 & 0 & -\delta_x u_0 \\ 0 & \delta_y & 0 & -\delta_y v_0 \\ 0 & 0 & 0 & f \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{Bmatrix} u \\ v \\ 0 \\ 1 \end{Bmatrix} \tag{9}$$

with:
(u_0,v_0): image frame coordinates of focal point projection in image plane;
(δ_x, δ_y): physical dimension of sensor pixel along direction u and v;
f: focal length.



a) b)

Fig. 15. Scheme of scanning system

It is possible to write the expression of the optical beam of a generic point in the image frame, that can be identified by means of parameter t.

$$\begin{cases} x_c = (u - u_0)\delta_u t \\ y_c = (v - v_0)\delta_v t \\ z_c = ft \end{cases} \tag{10}$$

Laser frame {L} is rotated and translated respect to camera frame, the steps and their sequence are:
Translation Δx_{lc} along axis x_c;
Translation Δy_{lc} along axis y_c;
Translation Δz_{lc} along axis z_c;
Rotation φ_{lc} around axis z_c;
Rotation θ_{lc} around axis y_c;
Rotation ψ_{lc} around axis x_c ;
Hence in equation (11) the transformation matrix between laser frame and camera frame is:

$$\left[{}^{c}T_{l}\right]=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & \cos(\psi_{lc}) & -\sin(\psi_{lc}) & 0\\ 0 & \sin(\psi_{lc}) & \cos(\psi_{lc}) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}\cos(\theta_{lc}) & 0 & \sin(\theta_{lc}) & 0\\ 0 & 1 & 0 & 0\\ -\sin(\theta_{lc}) & 0 & \cos(\theta_{lc}) & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\cdot$$
$$\begin{bmatrix}\cos(\varphi_{lc}) & -\sin(\varphi_{lc}) & 0 & 0\\ \sin(\varphi_{lc}) & \cos(\varphi_{lc}) & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & \Delta z_{lc}\\ 0 & 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & \Delta y_{lc}\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & 0 & \Delta x_{lc}\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad (11)$$

Laser plane {L2} coincides with (x,y) plane of laser frame {L}, so it contains three points:

$$\{p\}_l=\{0,0,0,1\}^T;\{q\}_l=\{1,0,0,1\}^T;\{r\}_l=\{0,0,1,1\}^T \quad (12)$$

In camera frame:

$$\{p_x,p_y,p_z,1\}_c^T=\left[{}^{c}T_l\right]^{-1}\{p\}_l;\{q_x,q_y,q_z,1\}_c^T=$$
$$\left[{}^{c}T_l\right]^{-1}\{q\}_l;\{r_x,r_y,r_z,1\}_c^T=\left[{}^{c}T_l\right]^{-1}\{r\}_l \quad (13)$$

It is possible to obtain laser plane equation in camera frame, solving equation (14) in xc, yc and zc.

$$\det\begin{bmatrix}x_c-p_x & y_c-p_y & z_c-p_z\\ q_x-p_x & q_y-p_y & q_z-p_z\\ r_x-p_x & r_y-p_y & r_z-p_z\end{bmatrix}=0 \quad (14)$$

If it is:

$$M_x=\det\begin{bmatrix}q_y-p_y & q_z-p_z\\ r_y-p_y & r_z-p_z\end{bmatrix};M_y=\det\begin{bmatrix}q_x-p_x & q_z-p_z\\ r_x-p_x & r_z-p_z\end{bmatrix};$$
$$M_z=\det\begin{bmatrix}q_x-p_x & q_y-p_y\\ r_x-p_x & r_y-p_y\end{bmatrix}$$

equation in camera frame, is:

$$(x_c-p_x)M_x-(y_c-p_y)M_y+(z_c-p_z)M_z=0 \quad (15)$$

It is possible to evaluate coordinates xc, yc e zc, in camera frame, solving system (16) with unknown t:

$$\begin{cases} x_c = (u - u_0)\delta_x t \\ y_c = (v - v_0)\delta_y t \\ z_c = ft \\ (x_c - p_x)M_x - (y_c - p_y)M_y + (z_c - p_z)M_z = 0 \end{cases} \tag{16}$$

The solution is:

$$t = \frac{p_x M_x - p_y M_y + p_z M_z}{(u - u_o)\delta_x M_x - (v - v_o)\delta_y M_y + fM_z} \tag{17}$$

Equation (17) permits to compute in the camera frame, the points coordinates of the scanning belt on the object surfaces, starting to its image coordinates (u,v). In this way it is possible to carry out a 3-D objects reconstruction by means of a laser knife.

4.2 Detection of the laser path

A very important step for 3-D reconstruction is image elaboration for the laser path on the target, [5, 6, 7] , the latter is shown in figure 16.

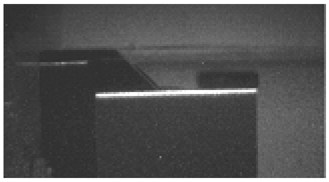
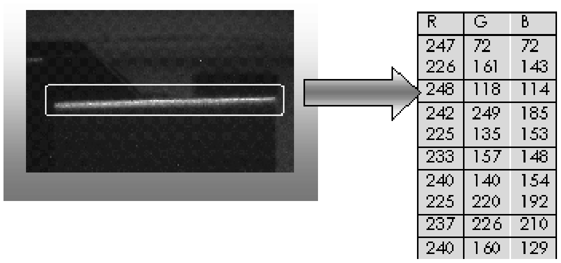


Fig.16. Laser path image.

Image elaboration procedure permits to user to choose some image points of laser line, in order to identify three principal colours (Red, Green, and Blue) of laser line, figure 17.



R	G	B
247	72	72
226	161	143
248	118	114
242	249	185
225	135	153
233	157	148
240	140	154
225	220	192
237	226	210
240	160	129

Fig. 17. Image matrix representation.

With mean values of scanning belt principal colours, it is possible to define a brightness coefficient of the laser line, according to relation (18):

$$s = \frac{\max(\text{mean}(R), \text{mean}(G), \text{mean}(B)) + \min(\text{mean}(R), \text{mean}(G), \text{mean}(B))}{2} \tag{18}$$

By means of relation (19), an intensity analysis is carried out on RGB image.

$$L(u, v) = \frac{\max(R, G, B) + \min(R, G, B)}{2} \tag{19}$$

By equation (19), the matrix that contains the three layers of RGB image, is transformed in a matrix L, that represents image intensity. This matrix represents same initial image, but it gives information only about luminous intensity in each image pixel, and so it is a grayscale expression of initial RGB image, figure 18 a) and b).

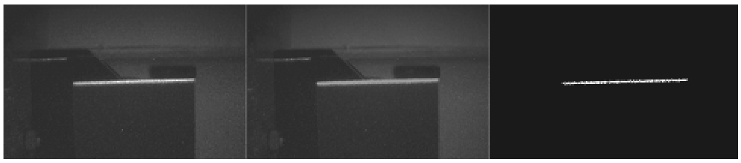
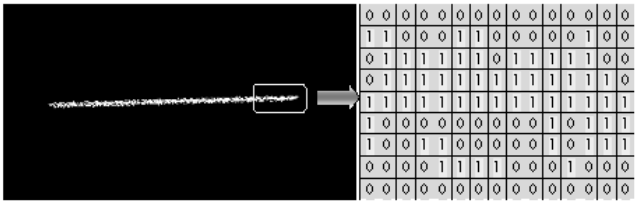

a) b) c)

Fig. 18. a)RGB initial image ; b)grayscale initial image L ; c) matrix Ib.

With relation (20), it is possible to define a logical matrix Ib. Matrix Ib indicates pixels of matrix L with a brightness in a range of 15% brightness coefficient s.

$$I_b(u, v) = \begin{cases} 1 & \text{se} \quad 0.85 \cdot s \le L(u, v) \le 1.15 \cdot s \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

In figure 18 c), matrix Ib is shown.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	1	0	0	0	0	0	1	0	0
0	1	1	1	1	1	1	0	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	1	0	1	1	1
1	0	0	1	1	0	0	0	0	0	1	0	1	1	1
0	0	0	0	1	1	1	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 19. I_b representation.

In matrix Ib, the scanning belt on the object surface (see fig. 19) is represented by means of the pixel value one. The area of the laser path in the image plane depends on the real dimension of laser beam and on external factors such as: reflection phenomena, inclination of object surfaces.
3-D reconstruction procedure is based on triangulation principle and it doesn't consider the laser beam thickness, so it is necessary to associate a line to the image of scanning belt.

Since the laser path in the image plane is rather horizontal, a geometrical mean is computed on, on the columns of the matrix Ib , that is to say: in the opposite direction to wider extension of the laser path in the image. This is shown in equation (21).

$$\begin{gathered}\Omega := \{u,v\}; \overline{\Omega} := \{\overline{u},\overline{v}\} \Rightarrow \\ \{\overline{u} = \frac{\sum\limits_{u,v\in\Omega} u I_b(u,v)}{\sum\limits_{u,v\in\Omega} I_b(u,v)} \neq \infty, \overline{v} = v : \sum_{u,v\in\Omega} I_b(u,v) \neq 0\}\end{gathered} \tag{21}$$

Then a matrix hp is also defined as follows:

$$h_b(u,v) = \begin{cases} 1 & \text{se} \quad \{u,v\} \in \overline{\Omega} \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

Matrix hb is hence a logical matrix with same dimension of Ib in which that represents laser image like a line, figure 20. This line is centre line of scanning belt on object surface in image.

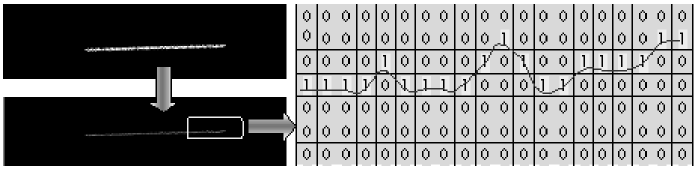


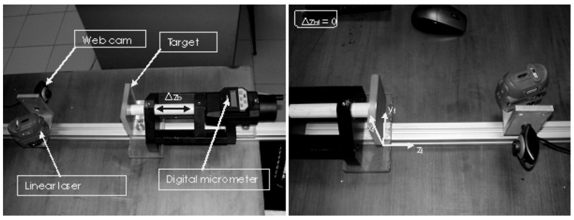
Fig. 20. h_b representation.

The set of transformations (19), (20) and (21) represents the image elaboration, that is necessary to identify a laser line in image. The points of this line are used in 3-D reconstruction procedure.

4.3 Calibration procedure

It is necessary to **i**dentify all model parameters in order to obtain a good 3-D reconstruction. Calibration is a basic procedure for the data analysis, [4, 6, 8]. For this reason, it was developed a calibration procedure for a classic laser scanner module.

The laser scanner module is composed by a web-cam with resolution 640x480 pixel and a linear laser. The calibration test rig is realized with a guide on which is fixed the laser scanner module and a digital micrometer with a target, figure 21 a).



a) b)

Fig. 21. a) Laser scanner module calibration test rig; b) fixed frame $O_f x_f y_f z_f$.

A fixed frame $O_f x_f y_f z_f$ has origin in a vertex of the rectangular target with dimension a=68 mm and b=74.5 mm, like it is shown in figure 21 b). Target movements are indicated with Δzbf.
In figure 22 some steps of calibration procedure are shown.

Fig. 22. Calibration procedure steps.

Calibration procedure is based on a set of images, taken with the target placed in different positions respect to the laser scanner module. A least square optimization allows to identify all parameters. In figure 23 are shown 10 images used for calibration, of scanning belt, with 10 different Δzb_f.

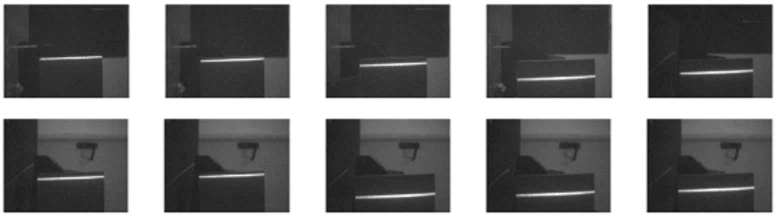


Fig. 23. Images used for calibration.

The aim of calibration procedure is to obtain the parameters that define the relation between laser frame LxLyLzL and camera frame $O_c x_c y_c z_c$, figure 15.
In every calibration step, it is possible to define transformation matrix between fixed frame $O_f x_f y_f z_f$ and target frame:

$$\left[{}^{f}T_b\right]_i = [1 \quad 0 \quad 0 \quad 0; 0 \quad 1 \quad 0 \quad 0; 0 \quad 0 \quad 1 \quad -\Delta z_{fb}^{i}; 0 \quad 0 \quad 0 \quad 1] \qquad (23)$$

The transformation matrix between fixed frame $O_f x_f y_f z_f$ and camera frame $O_c x_c y_c z_c$ can be optained analogous to matrix [^cT_l], and it is a function of six parameters:

$$\left[{}^fT_c\right] = f(\Delta x_{cf}, \Delta y_{cf}, \Delta z_{cf}, \psi_{cf}, \theta_{cf}, \varphi_{cf}) = f(\pi_{cf}) \tag{24}$$

with пcf set of parameter of transformation [^fT_c].
Equation (11) can be written as:

$$\left[{}^cT_l\right] = g(\Delta x_{lc}, \Delta y_{lc}, \Delta z_{lc}, \psi_{lc}, \theta_{lc}, \varphi_{lc}) = f(\pi_{lc}) \tag{25}$$

with пlc set of parameter of transformation [^cT_l].
For each step of the calibration procedure it is possible to evaluate the coordinates of the points of the laser line in the camera frame, according to equations (13) and (21):

$$\{\overline{x}_c, \overline{y}_c, \overline{z}_c, 1\}_i^T = f(\pi_{lc}, f) \cdot \{\overline{u}, \overline{v}, 0, 1\}_i^T \tag{26}$$

By means of equation (22), it is possible to write:

$$\begin{aligned} &\{\overline{x}_f, \overline{y}_f, \overline{z}_f, 1\}_i^T = \left[{}^fT_c\right]^{-1} \cdot \{\overline{x}_c, \overline{y}_c, \overline{z}_c, 1\}_i^T = \\ &\left[{}^fT_c\right]^{-1} \cdot f(\pi_{lc}, f) \cdot \{\overline{u}, \overline{v}, 0, 1\}_i^T = \omega(\pi_{cf}, \pi_{lc}, f) \cdot \{\overline{u}, \overline{v}, 0, 1\}_i^T \end{aligned} \tag{27}$$

The optimization problem can be defined as:

$$\min_{\rho \in R^{13}} F(\rho)^2 \tag{28}$$

F(ρ) is a vectorial function defined as:

$$\begin{aligned} &F(\rho) = \{(\overline{z}_f^{i+1} - \overline{z}_f^i) - (\Delta\overline{z}_{bf}^{i+1} - \Delta\overline{z}_{bf}^{i+1}), \min(\overline{x}_f^i), \\ &\max(\overline{x}_f^i) - a, \max(\overline{z}_f^i) - \min(\overline{z}_f^i), \\ &\max(\overline{y}_f^i) - \min(\overline{y}_f^i), \min(\overline{y}_f^1), \min(\overline{z}_f^1)\}^T \end{aligned} \tag{29}$$

With a least square optimization, the unknown parameters of set [пcf, пlc, f] are identified.
An interactive GUI was developed to allow the user to acquire images, to execute optimization and to verify the results.
In figure 24, a graphical result of calibration is shown: it is possible to observe the camera frame position, the laser beam (represented by the quadrilateral on the left), and the 3-D reconstructions of scanning beam on target surface, for each image used in the calibration procedure.

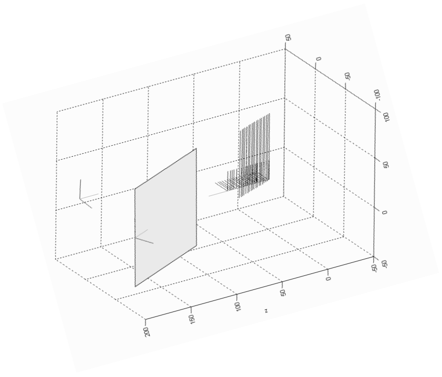


Fig. 24. Calibration results

4.4 System accuracy

4.4.1 Camera error

An obvious source of camera calibration error arises from the spatial quantization of the image. However, many times the image event of interest spans many pixels. It is then possible to calculate the centroid of this event to sub-pixel accuracy, thereby reducing the spatial quantization error. Limiting the sub-pixel accuracy is the fact that in general the perspective projection of an object's centroid does not equal the centroid of the object's perspective projection. However, often times the error incurred from assuming the centroid of the projection is the projection of the centroid is quite small [14].

Camera accuracy is a function of its distance from observed object.

If an unitary variation of pixels in the directions u and v is considered, from the equation (10) is gotten:

$$\begin{aligned} x_c &= (u+1-u_0)\delta_x \frac{z_c}{f} \\ y_c &= (v+1-v_0)\delta_y \frac{z_c}{f} \end{aligned} \tag{30}$$

Calculating the difference respectively between x_c and y_c after and before the variation is gotten the accuracy of the system, that is the variation of x_c and y_c that it can be gotten for a minimum variation in terms of pixels in the image frame:

$$\begin{aligned} \Delta x_c &= x_c(u+1) - x_c(u) \\ \Delta y_c &= y_c(v+1) - y_c(v) \end{aligned}$$

The image accuracies in directions x_c and y_c are:

$$a^c{}_x = \delta_u \cdot \frac{z_c}{f}$$

$$a^c{}_y = \delta_v \cdot \frac{z_c}{f} \tag{31}$$

4.4.2 Scanner module error

It is possible to define an accuracy expression for 3D-reconstruction that can be obtained by means of laser scanner module, according to equations (16) and (17).
A variation (α, β) of image coordinates (u,v), generates a variation of parameter t of equation (17):

$$\Delta t = \frac{-(p_x M_x - p_y M_y + p_z M_z)\cdot(\alpha\delta_x M_x - \beta\delta_y M_y)}{(u\delta_x M_x + \alpha\delta_x M_x - v\delta_y M_y - \beta\delta_y M_y + fM_z)\cdot(u\delta_x M_x - v\delta_y M_y + fM_z)} \tag{32}$$

where:
- α pixel variation in u direction;
- β pixel variation in v direction;

The variation of parameter t, allows to define an expression of accuracy of 3D reconstruction. In fact, by means of equation (16), it is possible to obtain the variation of coordinates in camera frame in function of variation of image coordinates:

$$\begin{cases} \Delta x_c = \alpha\delta_u \cdot \dfrac{p_x M_x - p_y M_y + p_z M_z}{(u - u_o + \alpha)\delta_x M_x - (v - v_o + \beta)\delta_y M_y + fM_z} + (u - u_0)\delta_x \cdot \Delta t \\ \Delta y_c = \beta\delta_v \cdot \dfrac{p_x M_x - p_y M_y + p_z M_z}{(u - u_o + \alpha)\delta_x M_x - (v - v_o + \beta)\delta_y M_y + fM_z} + (v - v_0)\delta_y \cdot \Delta t \\ \Delta z_c = \Delta t \cdot f \end{cases} \tag{33}$$

An unitary variation of image coordinates (α=1 and β=0, or α=0 and β=1, or α=1 and β=1), allows to define three accuracy parameters of scanner laser 3D reconstruction:

$$\begin{cases} a^{SL}{}_x = \Delta x_c \\ a^{SL}{}_y = \Delta y_c \\ a^{SL}{}_z = \Delta z_c \end{cases} \Rightarrow \text{with} \Rightarrow \begin{cases} (\alpha,\beta) = (0,1) \\ \text{or} \\ (\alpha,\beta) = (1,0) \\ \text{or} \\ (\alpha,\beta) = (1,1) \end{cases} \tag{34}$$

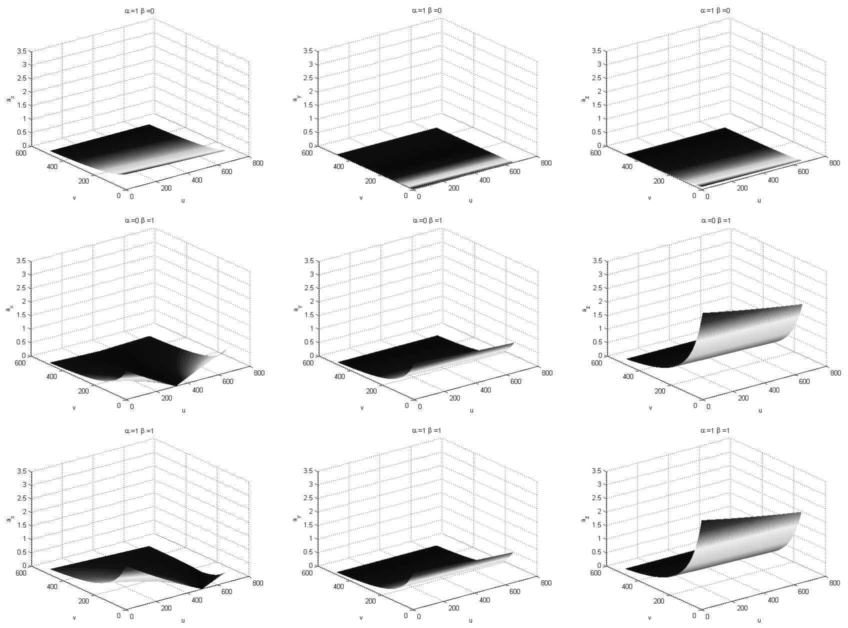


Fig. 25. Error of the accuracy

As it has shown in the figure 25, the worst accuracy of the laser scanner for a pixel variation is in the direction z_c. Besides it can be seen that az has minimum value for values of v = dv and it grows up to v = 1 with a non-linear law.

4.4.3 Laser precision

Most of the outlier and other erroneous points are caused by reflections. In these cases, the high energy laser beam is reflected from mirroring surfaces such as metal or glass. Therefore, too much light hits the sensor of the camera and so-called blooming effects occur. In other cases, a direct reflection may miss the camera. In addition, a part of the object may lie in the path from the projected laser line to the camera causing a shadowing effect. All these effects are responsible for gaps and holes. At sharp edges of some objects, partial reflections appear. In addition, craggy surfaces cause multiple reflections and, therefore, indefinite point correlations.

Furthermore, aliasing effects in the 2D image processing of laser beam, lead to high frequent noise in the generated 3D data [15].

The laser beam thickness in the image, can vary because of the above described effects, but 3-D reconstruction procedure is based on triangulation principle and it doesn't consider this phenomenon. In fact, the detection of laser path allows to identify a line in image with a thickness of a pixel.

The real laser beam thickness and its path thickness in the image, must be considered to evaluate the precision of 3D reconstruction.

The accuracy a^c_z becomes worse, if a thickness parameter is considered. This parameter is a generalized measure of laser beam thickness in pixel, and it can be expressed with two

components: thu thickness measure along direction u in image frame and thv thickness measure along direction v in image frame.
An expression of 3D reconstruction accuracy in direction z of camera frame, can be obtained by means of equation (11), in which parameters (α, β) are the generalized laser beam thickness (thu, thv):

$$\begin{cases} a_x = \Delta x_c \\ a_y = \Delta y_c \Rightarrow \text{with} \Rightarrow (\alpha,\beta) = (th_u/2, th_v/2) \\ a_z = \Delta z_c \end{cases} \tag{35}$$

The equations (32) define the resolution of the laser scanner 3-D reconstruction, and they allows to evaluate the accuracy of each point coordinates that is obtained with laser beam image elaboration.

4.5 Scanner range

Another characteristic of a 3D laser scanner is the minimum and the maximum distance between a generic point of a surface and the image plane. These parameters define the range of the scanning procedure. Decreasing the angle θ of inclination of the laser plane respect to the plane $x_c z_c$ of the camera frame at a respective fixed distance s the range of scanning increases (figure 26).

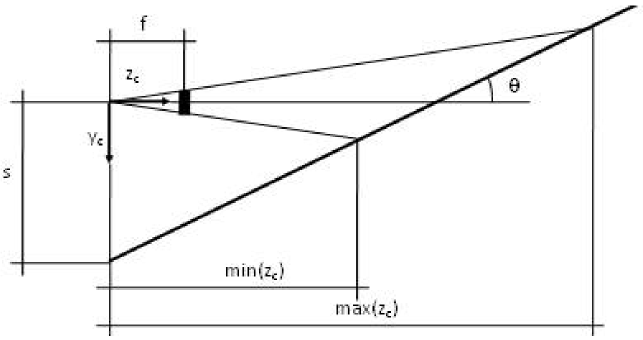


Fig. 26. The minimum and the maximum distance between a generic point of a surface and the image plane

This is not a good solution since to decrease of this angle the accuracy worsens notably as is shown in figure 27.

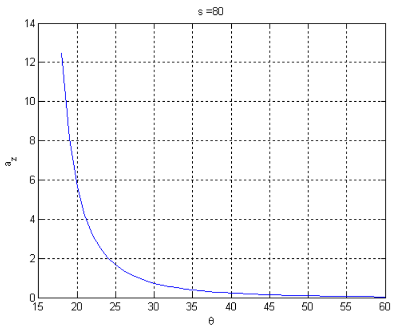


Fig. 27. Accuracy diagram.

For the considered system with values s = 90 mm and θ = 23° it have been gotten: max(zc) = 525 mm and min(zc) = 124 mm;

5. Experimental results

To evaluate the accuracy of the laser scanner system, the latter was fixed on a robot arm; in this way it was possible to capture a lot of shape information of the object from different views.

5.1 The test rig

5.1.1 Laser Scanner Module

Our rig, is based on a laser profile, that essentially consists in a line laser and a camera. The laser beam defines a "laser plane" and the part of the laser plane that lies in the image view of the camera is denoted the "scanning window", figure 28.

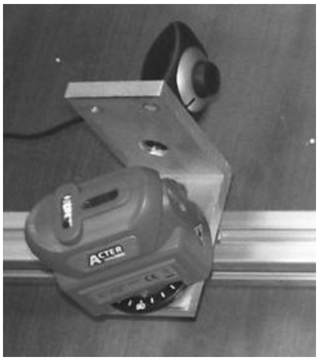


Fig. 28. Scanner module

The laser scanner device was realized by assembling a commercial linear laser and a common web-cam.

5.1.2 The Robot

In order to optimize the accuracy of the reconstruction resulting model, scanning should be adapted to the shape of the object. One way to do that is to use an industrial robot to move a laser profile scanner along curved paths.
The scanner laser module was mounted on a revolute robot with three d.o.f., designed and assembled in our Department, figure 29.

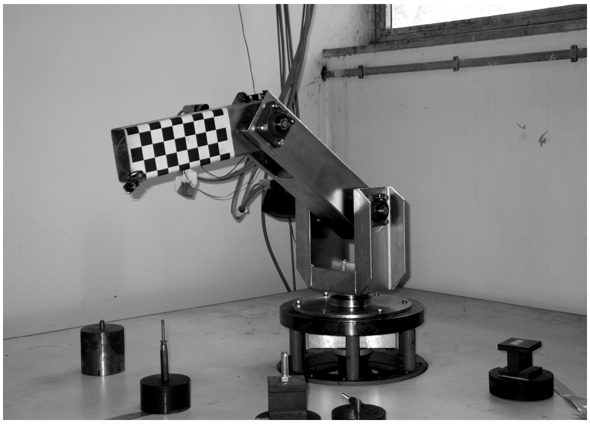


Fig. 29. Revolute robot

The robot serves as a measuring device to determine the scanning window position and orientation in 3D for each camera picture, with a great precision. All scan profiles captured during a scan sequence must be mapped to a common 3D coordinate system, and to do this, positional information from the robot were used [11]. Figure 30 shows the equipment at work.

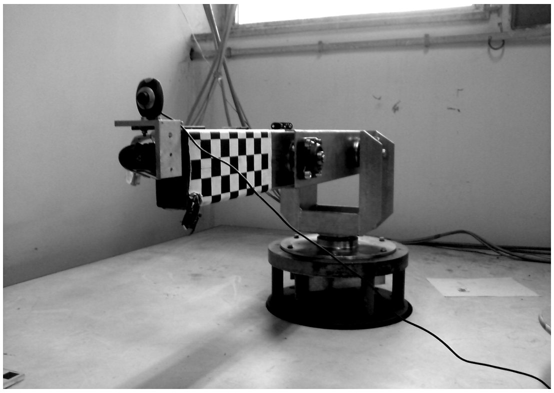


Fig. 30. The robot scanning system

The authors have developed a solution where the robot controller and scanner software work separately during a scan sequence that will be described in the following paragraphs.

5.2 The laser scanner on the robot model

When the laser scanner module is installed on the robot, figure 30, it is possible to use positional information from robot to determine the scanning window position and orientation in 3D.

Defining [DH] as the transformation matrix between coordinates in the robot base frame 0 (the fixed one) and those in frame 3 (the one of the last link), figure 31, for the coordinates of a generic point P exists this relationship:

$$\{P\}_0 = [DH] \cdot \{P\}_3 \tag{36}$$

The matrix [DH] depends on 9 constant kinematic structure parameters, that are known, and 3 variable joints position parameters that are measurable by means of robot control system.

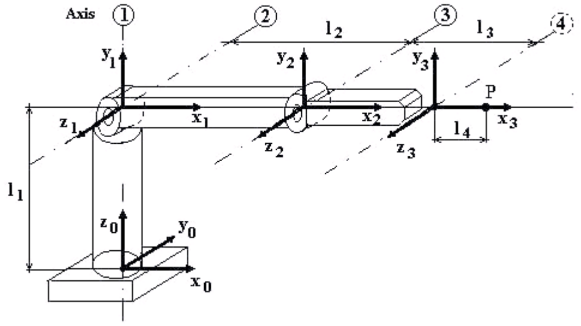


Fig. 31. Revolute robot scheme

Knowing the transformation matrix $[^cT_3]$ between the camera frame and the frame of the robot last link, it's possible to obtain a transformation matrix between the camera frame and the frame 0, figure 32.

$$\{P\}_c = [^cT_3] \cdot [DH]^{-1} \cdot \{P\}_0 \tag{37}$$

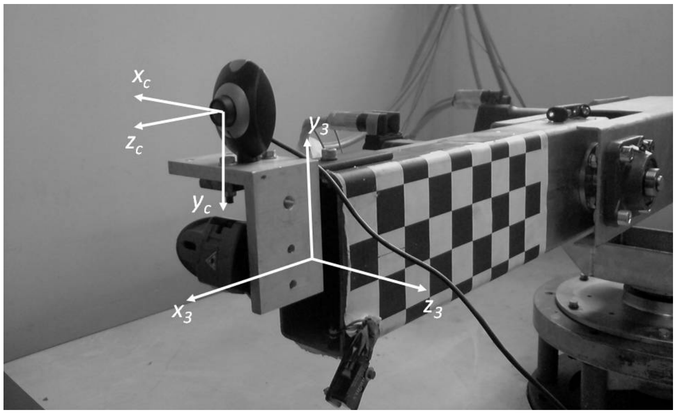
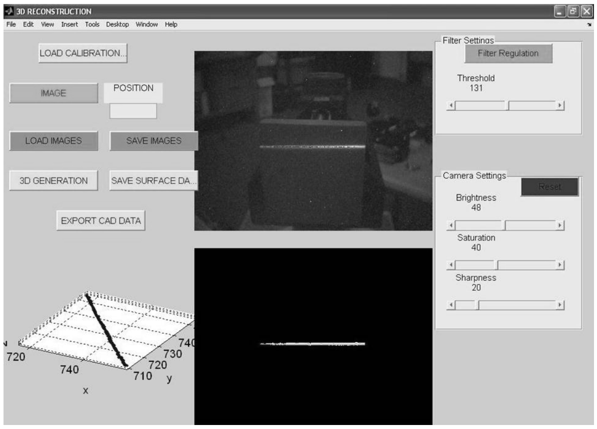


Fig. 32. Camera reference system.

By means of the equations (16), (17) and (29), the relationship between image coordinates (u,v) of the laser path and its coordinates in the robot base frame 0, is defined. By means of these equations, it is possible to reconstruct the 3D points in the robot base frame, of the intersection between the laser line and the object.
Robot positioning errors do not influence the 3D reconstruction, because each image is acquired and elaborated in a real robot position, that is known by means of robot encoders [12].

5.3 The data capture and the registration

The scanner video camera captures profiles from the surface of the object as 2D coordinates in the laser plane. During a scan sequence the laser scanner module is moved in order to capture object images from different sides and with different angles-shot according to the shape of the object.
In Matlab, an interactive GUI was developed in order to allows users to acquire and to elaborate data, figure 33. For each camera picture, along the scan path, the scanner derives a scan profile built up of point coordinates, in real-time.
The first step present in the GUI is the load of the calibration parameters that are composed by the laser scanner parameters and the matrix [${}^{c}T_3$]. The second step is to filter the pixels of the laser path from the image, to do this, there are some regulations: the identification of the intensity of selected pixels, the calculus of the threshold and other regulations of the camera settings. The third step is to write the 3 joint position parameters of the robot in the window "position" , after this, clicking on the button "Image" the software save all the information, necessary for the reconstruction, in the workspace of the Matlab. Clicking on the button "3D generation" the software calculates the 3D positions of the laser path in the robot base frame, and the result is shown on the GUI window.



Threshold 131

Brightness 48

Saturation 40

Sharpness 20

Fig. 33. Developed software.

When the scanning procedure is completed, the user can save images and relative robot position information in a file, save the cloud of points represent the surface of the test object and export the surface information in a file format that permits to load the data from the CAD software "CATIA".

Besides it’s possible to load image information from a preview scanning procedure, this is useful for reconstruct the same laser path information using different calibration parameters.

5.4 The surfaces reconstruction

The system has been tested before in a fixed robot position, to verify calibration and reconstruction procedures, then the shape of some components, was defined using robot to move laser scanner module. The test objects are shown in the figure 34.

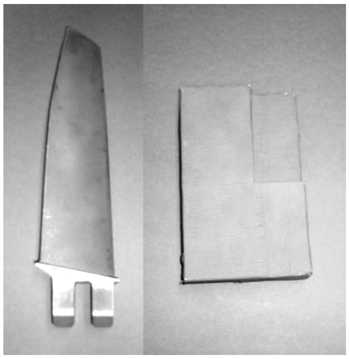


Fig. 34. Test specimens.

In the figure 35 and 36 it’s possible to see a step of the procedure with the final results for the two test specimens.

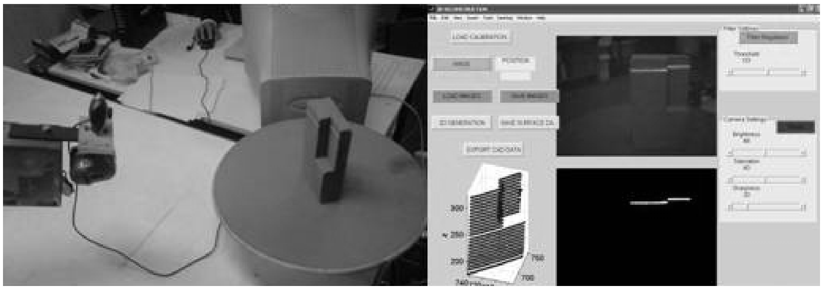


Fig. 35. Elaboration procedure of the first test specimen

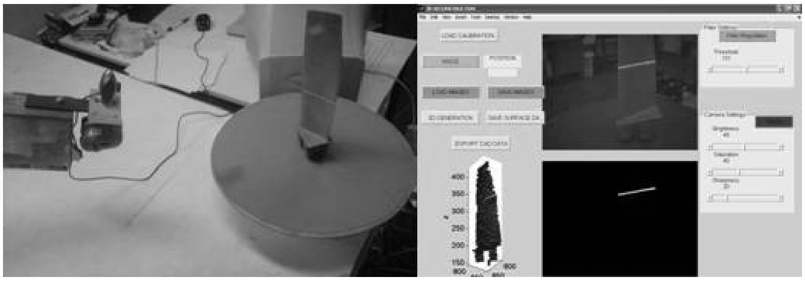


Fig. 36. Elaboration procedure of the second test specimen

By using the software CATIA it was possible to build the surface of the two test objects, in this way it was obtained the CAD model, this step of the 3D reconstruction method is a real reverse engineering application. The routine "Digitized shape editor" of the "CATIA" addresses digitalized data import, clean up, tessellation, cross sections, character line, shape and quality checking. In the figure 37 and 38 are shown the comparisons between the clouds of points and the respective surfaces for each object.

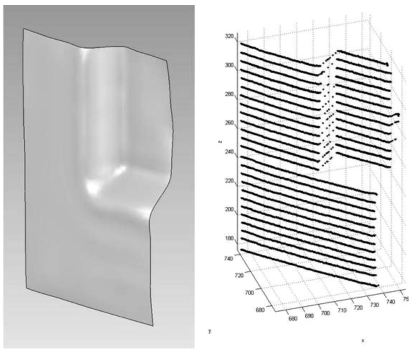


Fig. 37. First test specimen results

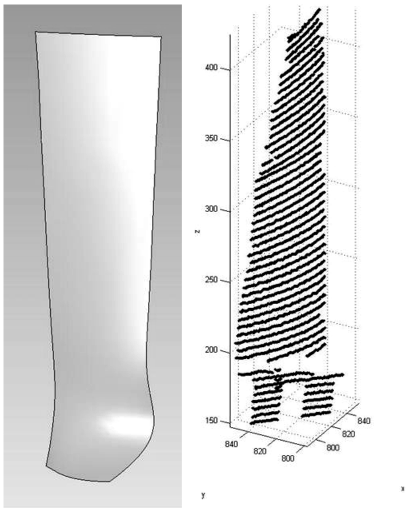


Fig. 38. Second test specimen results

In the figures 39 and 40, an evaluation of 3D-reconstruction accuracy is shown for the two analyzed test specimen. It is possible to observe, that these first results have the worst accuracy along direction z of camera frame, according to observations of paragraph 5.3.

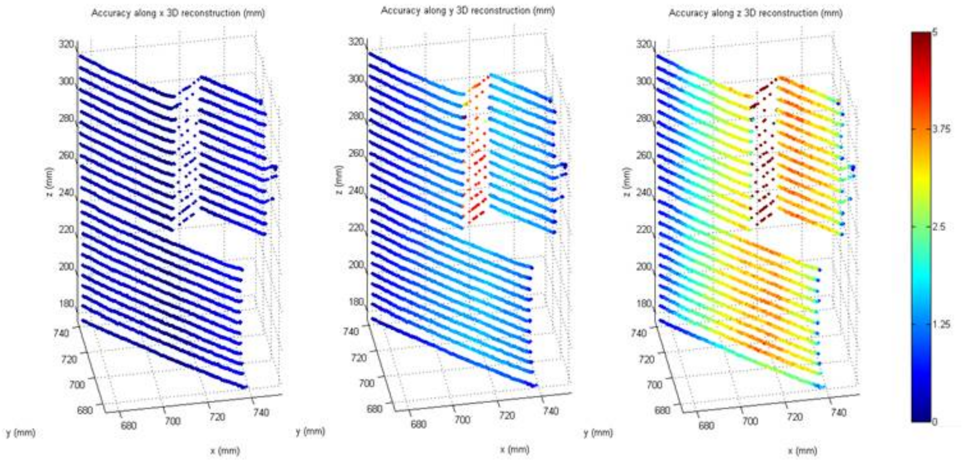


Fig. 39. First test specimen results

The results of the 3D reconstruction obtained by means of the rig that was designed and developed at authors' laboratory were compared with the ones obtained by means of a commercial 3D laser scanner. In the figures 41 and 42 the clouds of points obtained with the two different rigs are compared.

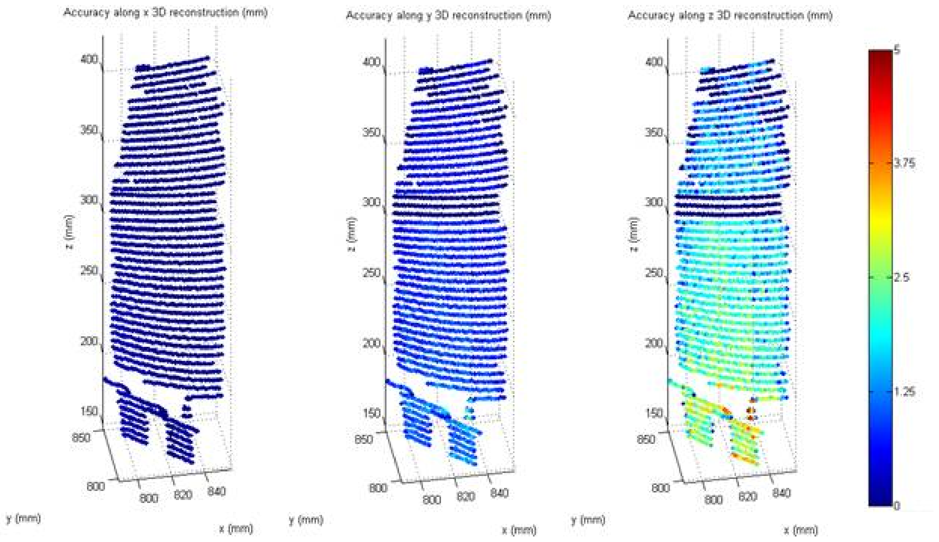


Fig. 40. Second test specimen results

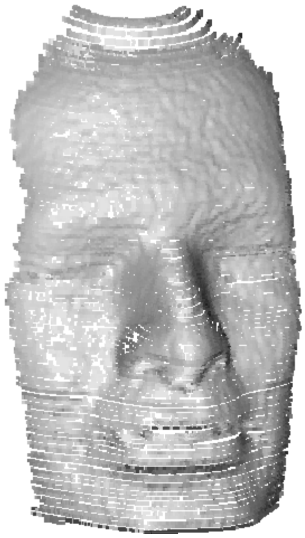


Fig. 41. Cloud of points obtained by the aouthors' robot assisted rig.

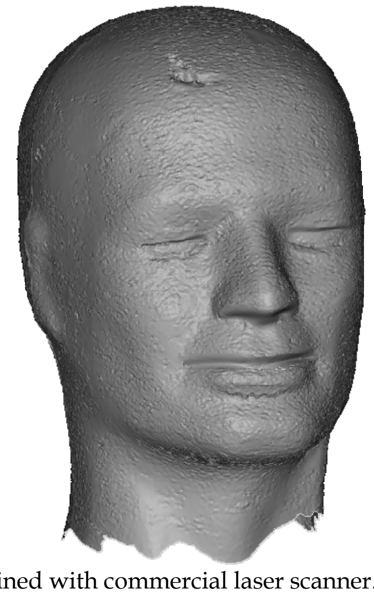


Fig. 42. Clouds of point obtained with commercial laser scanner.

In figure 43 is reported a comparison between the points obtained by the authors' rig and the commercial laser scanner. In figure 44 is reported a comparison between the surafces fobtaine by the above mentioned rigs.

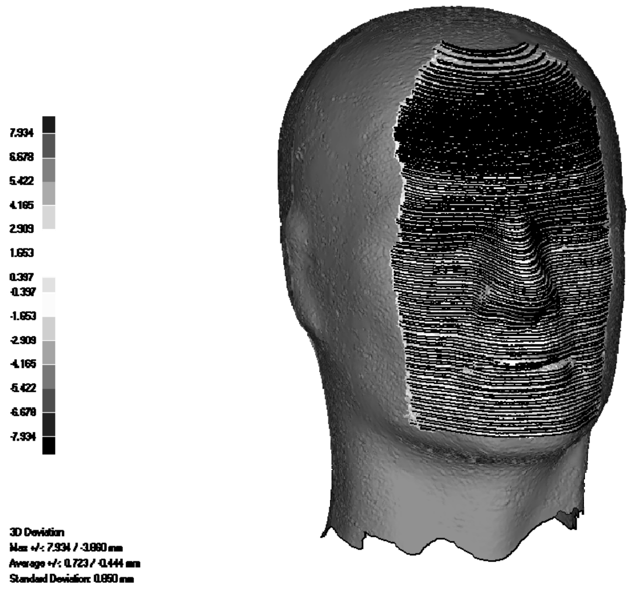


Fig. 43. Comparison between results obtained with two different rig.

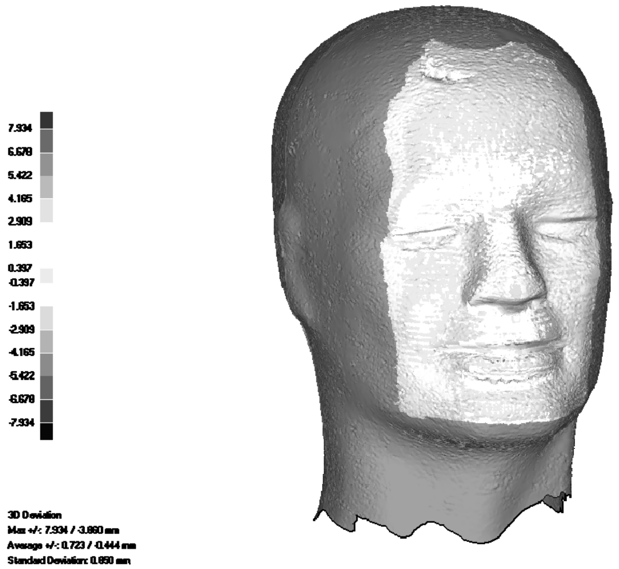


Fig. 44. Comparison between results obtained with two different rig.

It was observed that in most cases the differences are no more than ±1.5 mm; just in few areas the differences can reach 5 mm. A more detailed analisys showed that these differences concern single points, so it is possible to presume that a preliminary analisyss of the cloud of points could permit a general increase of the reconstruction accuracy.

6. Conclusions

The proposed procedures are absolutely non-invasive since they do not involve any modification of the scene; in fact no markers with features visible by both the camera and the laser, or any other device, are required.
As for the first results of the new method for real time shape acquisition by a laser scanner , it must be said that, although the test rig has been conceived just to validate the method (hence no high resolution cameras were adopted), the tests have showed encouraging results. These results can be summarized as follows.

1. It is possible to calibrate the intrinsic parameters of the video system, the position of the image plane and the laser plane in a given frame, all in the same time.
2. The surface shapes can be recognized and recorded with an appreciable accuracy.
3. The proposed method can be used for robotic applications such as robotic kinematic calibration and 3D surfaces recognition and recording. For this last purpose the test rig was fitted on a robot arm that permitted to the scanner device to 'observe' the 3D object from different and known position.

A detailed analysis of the sources of errors and the verification of the accuracy has been also carried on. As far as the latter aspect is concerned, the authors believe that a better system for tracking the position of the robot arm could enhance accuracy.
Finally, the authors would like to point out that the solution proposed is relatively low cost, scalable and flexible. It is also suitable for applications other than RE, like robot control or inspection.

7. References

[1] Fusiello, A. (2005). Visione Computazionale: appunti delle lezioni, *Informatic Department, University of Verona*, 3 March 2005.

[2] F. Blais (2004). Review of 20 years of range sensor development, *Journal of Electronic Imaging* , Vol. 13, No. 1, pp. 231–240.

[3] D. Acosta, O. García and J. Aponte (2006). Laser Triangulation for shape acquisition in a 3D Scanner Plus Scanner, *Proc. of the Electronics Robotics and Automotive Mechanics Conference*, 2006.

[4] J. Forest (2004). New methods for triangulation-based shape acquisition using laser scanners., *PhD thesis*, University of Girona, 2004.

[5] C. Colombo, D. Comanducci and A. Del Bimbo (2006). Low-Cost 3D Scanning by Exploiting Virtual Image Symmetries, *Journal of Multimedia*, Vol. 1, No. 7.

[6] N. Koller (2005). Fully Automated Repair of Surface Flaws using an Artificial Vision Guided Robotic Grinder, *PhD thesis*, University of Leoben, 2007.

[7] C. Matabosch (2007).Hand-held 3D-scanner for large surface registration, *PhD thesis*, University of Girona, 2007.

[8] M. Ritter, M. Hemmleb, O. Sinram, J. Albertz and H. Hohenberg (2004). A Versatile 3D Calibration Object for Various Micro-range Measurement Methods, *Proc. of ISPRS*, pp. 696-701, Istanbul, 2004.

[9] L.A. Albuquerque and J.M.S.T. Motta (2006). Implementation of 3D Shape Reconstruction from Range Images for Object Digital Modeling, *ABCM Symposium Series in Mechatronics*, Vol. 2, pp.81-88.

[10] Sören Larsson and J.A.P. Kjellander (2006). Motion control and data capturing for laser scanning with an industrial robot, *Robotics and Autonomous Systems*, Vol.54, No.6, pp. 453-460, 30 June 2006.
[11] R.A. Jarvis and Y.L. Chiu (1996). Robotic Replication of 3D Solids, *Proceedings of IROS*, 1996.
[12] V. Niola, C. Rossi and S. Savino (2007). Vision System for Industrial Robots Path Planning, *Journal of Mechanics and Control*.
[13] Cesare Rossi, Sergio Savino and Salvatore Strano (2008). 3D object reconstruction using a robot arm, *Proc. of 2-nd European Conference on Mechanism Science*, Cassino, Italy, 2008.
[14] J. Russell Noseworthy, Arthur M. Ryan, Lester A. Gerhardt (1991). Camera and Laser Scanner Calibration with Imprecise Data, *Proc. of Third Annual Conference on Intelligent Robotic Systems for Space Exploration*, pp. 99-111, 1991.
[15] C. Teutsch, T. Isenberg, E. Trostmann, M. Weber (2004). Evaluation and Optimization of Laser Scan Data, *15th Simulation and Visualisation*, March 4—5, 2004.

ROBUST CONTROL DESIGN FOR TWO-LINK NONLINEAR ROBOTIC SYSTEM

Shield B. Lin[+] and Sheng-Guo Wang[*]
[+] *Prairie View A&M University, Prairie View, Texas*
USA
[*] *University of North Carolina at Charlotte, Charlotte, North Carolina*
USA

1. Introduction

A good performance of robot control requires the consideration of efficient dynamic models and sophisticated control approaches. Traditionally, control law is designed based on a good understanding of system model and parameters. Thus, a detailed and correct model of a robot manipulator is needed for this approach [1, 2].
A two-link planar nonlinear robotic system is a well-used robotic system, e.g., for welding in manufacturing and so on. Generally, a dynamic model can be derived from the general Lagrange equation method. The modeling of a two-link planar nonlinear robotic system with assumption of only masses in the two joints can be found in the literature, e.g., [3, 4]. Here, the authors revise centrifugal and Coriolis force matrix in the literature [3, 4] as pointed out in the next section. Furthermore, in practice, the robot arms have their mass distributed along their arms, not only masses in the joints as assumed. Thus, it is desired to develop a detailed model for two-link planar robotic systems with the mass distributed along the arms. Distributed mass along robot arms was discussed by inertia in SCARA robot [5]. Here, we present a new detailed consideration of any mass distributions along robot arms in addition to the joint mass.
Moreover, it is also necessary to consider numerous uncertainties in parameters and modeling. Thus, robust control, robust adaptive control and learning control become important when knowledge of the system is limited. We need robust stabilization of uncertain robotic systems and furthermore robust performance of these uncertain robotic systems. Robust stabilization problem of uncertain robotic control systems has been discussed in [1-3, 5-6] and many others. Also, adaptive control methods have been discussed in [1, 7] and many others. Because the closed-loop control system pole locations determine internal stability and dominate system performance, such as time responses for initial conditions, papers [6, 8] consider a robust pole clustering in vertical strip on the left half s-plane to consider robust stability degree and degree of coupling effects of a slow subsystem (dominant model) and the other fast subsystem (non-dominant model) in a two-time-scale system. A control design method to place the system poles robustly within a vertical strip has been discussed in [6, 8-10], especially [6] for robotic systems. However, as mentioned

above, there is a need of a detailed and practical two-link planar robotic system modeling with the practically distributed robotic arm mass for control.

Therefore, this chapter develops a practical and detailed two-link planar robotic systems modeling and a robust control design for this kind of nonlinear robotic systems with uncertainties via the authors' developing robust control approach with both H∞ disturbance rejection and robust pole clustering in a vertical strip. The design approach is based on the new developing two-link planar robotic system models, nonlinear control compensation, a linear quadratic regulator theory and Lyapunov stability theory.

2. Modeling of Two-Link Robotic Systems

The dynamics of a rigid revolute robot manipulator can be described as the following nonlinear differential equation [1, 2, 6, 10]:

$$F_c = M(q)\ddot{q} + V(q,\dot{q})\dot{q} + N(q,\dot{q}) \tag{1.a}$$

$$N(q,\dot{q}) = G(q) + F_d\dot{q} + F_s(\dot{q}) \tag{1.b}$$

where $M(q)$ is an $n \times n$ inertial matrix, $V(q,\dot{q})$ an $n \times n$ matrix containing centrifugal and coriolis terms, $G(q)$ an $n \times 1$ vector containing gravity terms, $q(t)$ an $n \times 1$ joint variable vector, F_c an $n \times 1$ vector of control input functions (torques, generalized forces), F_d an $n \times n$ diagonal matrix of dynamic friction coefficients, and $F_s(\dot{q})$ an $n \times 1$ Nixon static friction vector.

However, the dynamics of the robotic system (1) in detail is needed for designing the control force, i.e., especially, what matrices $M(q)$, $V(q,\dot{q})$ and $G(q)$ are.

Consider a general two-link planar robotic system in Fig. 1, where the system has its joint mass m_1 and m_2 of joints 1 and 2, respectively, robot arms mass m_{1r} and m_{2r} distributed along arms 1 and 2 with their lengths l_1 and l_2, generalized coordinates q_1 and q_2, i.e., their rotation angles, $q = [q_1\ q_2]'$, control torques (generalized forces) f_1 and f_2, $F_c = [f_1\ f_2]'$.

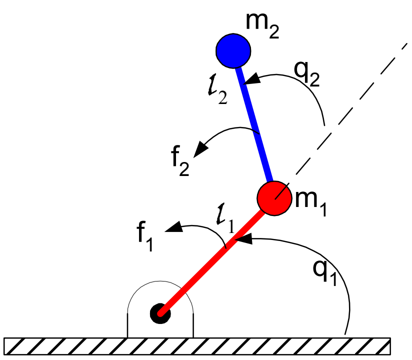


Fig. 1. A two-link manipulator

Theorem 1. A general two-link planar robotic system has its dynamic model as in (1) with

$$M(q)=\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \tag{2}$$

$$\begin{aligned} M_{11} &= (m_1+\xi_1 m_{r1}+m_2+m_{r2})l_1^2 \\ &+(m_2+\xi_2 m_{r2})l_2^2+2(m_2+\zeta_2 m_{r2})l_1 l_2\cos q_2 \end{aligned} \tag{3.a}$$

$$\begin{aligned} M_{12}=M_{21} &= (m_2+\xi_2 m_{r2})l_2^2 \\ &+(m_2+\zeta_2 m_{r2})l_1 l_2\cos q_2 \end{aligned} \tag{3.b}$$

$$M_{22}=(m_2+\xi_2 m_{r2})l_2^2 \tag{3.c}$$

$$V(q,\dot q)=-(m_2+\zeta_2 m_{r2})l_1 l_2 \dot q_2 \sin q_2 \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \tag{4}$$

$$G(q)=g\begin{bmatrix} (m_1+\zeta_1 m_{r1}+m_2+m_{r2})l_1\cos q_1+(m_2+\zeta_2 m_{r2})l_2\cos(q_1+q_2) \\ (m_2+\zeta_2 m_{r2})l_2\cos(q_1+q_2) \end{bmatrix} \tag{5}$$

where g is the gravity acceleration, m_1 and m_2 are joints 1 and 2 mass, respectively, m_{r1} and m_{r2} are total mass of arms 1 and 2, which are distributed along their arm lengths of l_1 and l_2, the scaling coefficients ξ_1, ξ_2, ζ_1 and ζ_2 are defined as follows:

$$\xi_i=\int_0^{l_i}\rho_i(l)S_i(l)l^2dl/m_{ri}l_i^2\,,\ \zeta_i=\int_0^{l_i}\rho_i(s)S_i(l)ldl/m_{ri}l_i\,,$$
$$m_{ri}=\int_0^{l_i}\rho_i(l)S_i(l)dl\,,\ i=1,2 \tag{6}$$

where $\rho_1(l)$ and $\rho_2(l)$ are the arm mass density functions along their length l, $S_1(l)$ and $S_2(l)$ are the arm cross-sectional area functions along the length l.

Proof: The proof is via Lagrange method and dynamic motion equations. The mass distribution can be various by introducing the above new scaling coefficients. Due to the page limit, detail of the proof is omitted.

Remark 1. From (2)–(4) in Theorem 1, $\dot M(q)=V(q,\dot q)$. Theorem 1 is also different from the result in [3-6]. Especially, there are no corresponding items of ζ_i in [3-6].

Corollary 1. A two-link planar robotic system with consideration of only joint mass has its dynamic model as in (1) and Theorem 1, but with

$$m_{ri}=0\,,\ \xi_i=0\,,\ \zeta_i=0\,,\ i=1,2 \tag{7}$$

It means that its inertia matrix $M(q)$ in (2), and

$$M_{11}=(m_1+m_2)l_1^2+m_2 l_2^2+2m_2 l_1 l_2\cos q_2\,,$$
$$M_{12}=M_{21}=m_2(l_2^2+l_1 l_2\cos q_2)\,,\ M_{22}=m_2 l_2^2 \tag{8}$$

$$V(q,\dot{q}) = -m_2 l_1 l_2 \dot{q}_2 \sin q_2 \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \tag{9}$$

$$G(q) = g \begin{bmatrix} (m_1 + m_2) l_1 \cos q_1 + m_2 l_2 \cos(q_1 + q_2) \\ m_2 l_2 \cos(q_1 + q_2) \end{bmatrix} \tag{10}$$

Remark 2. It is noticed that centrifugal and Coriolis matrix $V(q,\dot{q})$ in (26) is equivalent to:

$$V(q,\dot{q}) = m_2 l_1 l_2 \sin q_2 \begin{bmatrix} -\dot{q}_2 & -\dot{q}_1 - \dot{q}_2 \\ -\dot{q}_2 & 0 \end{bmatrix} \tag{11}$$

in (1). Note that the Coriolis matrix is different from some earlier literatures in [3, 4].
Theorem 2. Consider a two-link planar robotic system having its robot arms with uniform mass distribution along the arm length. Thus, its dynamic model is as (1) – (6) of Theorem 1 with its scaling factors as follows:

$$\xi_1 = \xi_2 = 1/3 \text{, and } \zeta_1 = \zeta_2 = 1/2 \tag{12}$$

Proof: It can be proved by Theorem 1 and the uniform mass distribution in (6).
Theorem 3. Consider a two-link planar robotic system having its robot arms with linear tapered-shapes respectively along the arm lengths as:

$$r_i(l) = r_{0i} - k_i l \text{, } 0 \le l \le l_i \text{, } S_i(l) = \mu_i r_i^2(l) \text{, } i = 1,2 \tag{13}$$

where $r_i(l)$ in length is a general measure of the arm cross-section at the arm length *l*, e.g., as a radius for a disk, a side length for a square, $\sqrt{ab}$ for a rectangular with sides a and b, etc., $S_i(l)$ is the cross-sectional area of arm i at its length position *l*, μ_i is a constant, e.g., as π for a circle and 1 for a square. Assume that arm 1 and arm 2 respectively have their two end cross-sectional areas as:

$$S_{01} = S_1(0),\ S_{t1} = S_1(l_1),\ S_{02} = S_2(0),\ S_{t2} = S_2(l_2) \tag{14}$$

where $S_{0i} > S_{ti}$, $i = 1,2$. Their density functions are constants as $\rho_i(l) = \rho_i$, $i = 1,2$. Then, its dynamic model is as in (1) – (6) of Theorem 1 with its scaling factors:

$$\xi_i = \frac{S_{0i} + 6S_{ti} + 3\sqrt{S_{0i}S_{ti}}}{10(S_{0i} + S_{ti} + \sqrt{S_{0i}S_{ti}})},\ \zeta_i = \frac{S_{0i} + 3S_{ti} + 2\sqrt{S_{0i}S_{ti}}}{4(S_{0i} + S_{ti} + \sqrt{S_{0i}S_{ti}})} \tag{15}$$

$i = 1,2$, and its arm mass:

$$m_{ri} = \rho_i l_i (S_{0i} + S_{ti} + \sqrt{S_{0i}S_{ti}})/3 \text{, } i = 1,2 \tag{16}$$

Proof: It is proved by substituting (13) and (14) into (6) in Theorem 1 and further derivations.

Remark 3. The scaling factors (15) and the arm mass (16) in Theorem 3 may have other equivalent formulas, not listed here due to the page limit. Here, we choose (15) and (16) because the two-end cross-sectional areas of each arm are easily found from the design parameters or measured in practice. The arm cross-sectional shapes can be general in (13) in Theorem 3.

3. Robust Control

In view of possible uncertainties, the terms in (1) can be decomposed without loss of any generality into two parts, i.e., one is known parts and another is unknown perturbed parts as follows [2, 6]:

$$M = M_0 + \Delta M\,, \quad N = N_0 + \Delta N\,, \quad V = V_0 + \Delta V \tag{17}$$

where M_0, N_0, V_0 are known parts, ΔM, ΔN, ΔV are unknown parts. Then, the models in Section 2 can be used not only for the total uncertain robotic systems with uncertain parameters, but also for a known part with their nominal parameters of the systems.

Following our [6], we develop the torque control law as two parts as follows:

$$F_c = M_0(q)\ddot{q}_d + V_0(q,\dot{q})\dot{q} + N_0(q,\dot{q}) - M_0(q)u \tag{18}$$

where the first part consists of the first three terms in the right side of (18), the second part is the term of u that is to be designed for the desired disturbance rejection and pole clustering, q_d is the desired trajectory of q, however, the coefficient matrices are as (2) – (6) in Theorem 1 with all nominal parameters of the system. Define an error between the desired q_d and the actual q as:

$$e = q_d - q\,. \tag{19}$$

From (1) and (17)–(19), it yields:

$$\begin{aligned}\ddot{e} &= M^{-1}(q)[\Delta M(q)\ddot{q}_d + \Delta V(q,\dot{q})\dot{q} + \Delta N(q,\dot{q}) + M_0(q)u] \\ &= w + E\dot{e} + Fu + u \end{aligned}\tag{20}$$

$$E = -M^{-1}(q)\Delta V(q,\dot{q})\,, \quad F = -M^{-1}(q)\Delta M(q)$$

$$w = -F\ddot{q}_d - E\dot{q}_d + M^{-1}\Delta N \tag{21}$$

From [6], we can have the fact that their norms are bounded:

$$\|w\| < \delta_w\,, \quad \|E\| < \delta_e\,, \quad \|F\| < \delta_f \tag{22}$$

Then, it leads to the state space equation as:

$$\dot{x} = Ax + Bu + B[0 \quad E]x + BFu + Bw \tag{23}$$

$$x = \begin{bmatrix} e \\ \dot{e} \end{bmatrix} = [e_1\ e_2\ \dot{e}_1\ \dot{e}_2]',\ A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{24}$$

The last three terms denote the total uncertainties in the system. The desired trajectory q_d for manipulators to follow is to be bounded functions of time. Its corresponding velocity $\dot{q}_d$ and acceleration $\ddot{q}_d$, as well as itself q_d, are assumed to be within the physical and kinematic limits of manipulators. They may be conveniently generated by a model of the type:

$$\ddot{q}_d(t) + K_v\dot{q}_d(t) + K_p q_d(t) = r(t) \tag{25}$$

where $r(t)$ is a 2-dimensional driving signal and the matrices K_v and K_p are stable.
The design objective is to develop a state feedback control law for control u in (18) as

$$u(t) = -Kx(t) \tag{26}$$

such that the closed-loop system:

$$\dot{x} = (A - BK + B\begin{bmatrix}0 & E\end{bmatrix} - BFK)x + Bw \tag{27}$$

has its poles robustly lie within a vertical strip Ω:

$$\lambda(A_c) \in \Omega = \{s = x + jy \mid -\alpha_2 < x < -\alpha_1 \le 0\} \tag{28}$$

and a δ-degree disturbance rejection from the disturbance w to the state x, i.e.,

$$\|T_{xw}(s)\|_\infty = \left\|(sI - A_c)^{-1}B\right\|_\infty \le \delta \tag{29}$$

$$A_c = A - BK + B\begin{bmatrix}0 & E\end{bmatrix} - BFK \tag{30}$$

From [6], we derive the following robust control law to achieve this objective.
Theorem 4. Consider a given robotic manipulator uncertain system (27) with (1)–(6), (17)-(22), (24), where the unstructured perturbations in (21) with the norm bounds in (22), the disturbance rejection index $\delta > 0$ in (29), the vertical strip Ω in (28) and a matrix Q>0. With the selection of the adjustable scalars ε_1 and ε_2, i.e.,

$$(1 - \delta_f)/\delta_e > \varepsilon_1 > 0,\ (1 - \delta_f - \varepsilon_1\delta_e)\delta > \varepsilon_2 > 0 \tag{31}$$

there always exists a matrix $P > 0$ satisfying the following Riccati equation:

$$\begin{aligned} &A'_{\alpha 1}P + PA_{\alpha 1} - (1-\delta_f - \varepsilon_1\delta_e - \varepsilon_2/\delta)PBB'P \\ &+ (\delta_e/\varepsilon_1)I + (1/\varepsilon_2\delta)I + Q = 0 \end{aligned} \tag{32}$$

where

$$A_{\alpha 1} = A + \alpha_1 I = \begin{bmatrix} \alpha_1 I_2 & I_2 \\ 0 & \alpha_1 I_2 \end{bmatrix} \tag{33}$$

Then, a robust pole-clustering and disturbance rejection control law in (18) and (26) to satisfy (29) and (30) for all admissible perturbations *E* and *F* in (22) is as:

$$u = -Kx = -rB'Px \tag{34}$$

if the gain parameter *r* satisfies the following two conditions:

$$\text{(i)} \quad r \geq 0.5 \text{ and} \tag{35}$$

$$\text{(ii)} \quad \begin{aligned} &2\alpha_2 P + A'P + PA - (\delta_e/\varepsilon_1)I \\ &-[2r(1+\delta_f) + \varepsilon_1\delta_e]PBB'P > 0 \end{aligned} \tag{36}$$

Proof: Please refer to the approach developed in [6, 8] and utilizing the new model in Section 2.
It is also noticed that:

$$BB' = \begin{bmatrix} 0 & 0 \\ 0 & I_2 \end{bmatrix} \tag{37}$$

It is evident that condition (i) is for the α_1-degree stability and δ-degree disturbance rejection, and condition (ii) is for the α_2-degree decay, i.e., the left vertical bound of the robust pole-clustering.

Remark 4. There is always a solution for relative stability and disturbance rejection in the sense of above discussion. It is because the Riccati equation (32) guarantees a positive definite solution matrix *P*, and thus there exists a Lyapunov function to guarantee the robust stability of the closed loop uncertain robotic systems. The nonlinear compensation part in (18) has a similar function to a feedback linearization. The feature differences of the proposed method from other methods are the new nominal model, and the robust pole-clustering and disturbance attenuation for the whole uncertain system family. It is further noticed that the robustly controlled system may have a good Bode plot for the whole frequency range in view of Theorem 4, inequality (29) and its H-infinity norm upper bound.

Remark 5. The tighter robust pole-clustering vertical strip $-\alpha_1 > \operatorname{Re} \lambda(A_c) \geq -\underline{\alpha}_2$ has

$$\begin{aligned} \underline{\alpha}_2 = \alpha_1 + 0.5\overline{\lambda}\{&P^{-1/2}[-A'P - PA_{\alpha 1} + (\delta_e/\varepsilon_1)I \\ &+ (2r(1+\delta_f) + \varepsilon_1\delta_e)PBB'P]P^{-1/2}\} \end{aligned} \tag{38}$$

4. Examples

Example 1. Consider a two-link planar manipulator example (Fig. 1). First, only joint link masses are considered for simplicity, as the one in [3, 6]. However, we take the correct model in Corollary 1 and Remark 2 into account. The system parameters are: link mass $m_1 = 2kg$, $m_2 = 10kg$, lengths $l_1 = 1m$, $l_2 = 1m$, angular positions q_1, q_2 (rad), applied torques f_1, f_2 (Nm). Thus, the nominal values of coefficient matrices for the dynamic equation (1) in Corollary 1 are:

$$M_0(q) = \begin{bmatrix} 22+20C_2 & 10(1+C_2) \\ 10(1+C_2) & 10 \end{bmatrix}, V_0(q,\dot{q}) = -10S_2\dot{q}_2\begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}$$

$$N_0(q,\dot{q}) = g\begin{bmatrix} 12C_1+10C_{12} \\ 10C_{12} \end{bmatrix} \tag{39}$$

where $C_i = \cos q_i$, $i = 1,2$, $C_{12} = \cos(q_1+q_2)$, $S_2 = \sin q_2$, and g is the gravity acceleration.

The desired trajectory is $q_d(t)$ in (25) with $K_v = 0$, and

$K_p = I$, the signal $r(t) = [0.5 \quad 1]'$, the initial values of the desired states $q_d(0) = [2 \quad 2]'$, $\dot{q}_d(0) = 0$, i.e.,

$$q_{d1}(t) = 1.5\cos t + 0.5, \quad \text{and} \quad q_{d2}(t) = \cos t + 1 \tag{40}$$

The initial states are set as $q_1(0) = q_2(0) = -2$, and $\dot{q}_1(0) = \dot{q}_2(0) = 0$, i.e., $e_1(0) = 4$, $e_2(0) = 4$, $\dot{e}_1(0) = 0$, and $\dot{e}_2(0) = 0$. The state variable is $x = [e' \;\; \dot{e}']'$ where $e = q_d - q$. The parametric uncertainties are assumed to satisfy (22) with $\delta_f = 0.5$, $\delta_e = 40$, $\delta_N = 10$. Select the adjustable parameters $\varepsilon_1 = 0.012$, $\varepsilon_2 = 0.0015$ from (31), disturbance rejection index $\delta = 0.1$, the relative stability index $\alpha_1 = 0.1$, and the left bound of vertical strip $\alpha_2 = 2000$ since we want a fast response. By Theorem 4, we solve the Riccati equation (32) to get the solution matrix P and the gain matrix as:

$$P = \begin{bmatrix} 12693I_2 & 1584I_2 \\ 1584I_2 & 1643I_2 \end{bmatrix}, K = rB'P = [950.1823I_2 \; 985.7863I_2]$$

with $r = 0.6$. The eigenvalues of the closed-loop main system matrix $A - BK$ are {-0.9648, -0.9648, -984.8215, -984.8215}. Remark 5 gives the result $-\underline{\alpha}_2 = -1873$. The uncertain closed-loop system has its $-\alpha_2 < \text{Re}[\lambda(A_c)] < -\alpha_1$ robustly.

The total control input (law) is

$$F_c = F_T = M_0\ddot{q}_d + V_0(q,\dot{q})\dot{q} + N_0 - M_0u$$

$$
\begin{aligned}
&= \begin{bmatrix} 22+20C_2 & 10(1+C_2) \\ 10(1+C_2) & 10 \end{bmatrix} \begin{bmatrix} -1.5\cos t \\ -\cos t \end{bmatrix} \\
&+ 10S_2\dot{q}_2 \begin{bmatrix} -2 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + g \begin{bmatrix} 12C_1 + 10C_{12} \\ 10C_{12} \end{bmatrix} + \\
&\begin{bmatrix} 22+20C_2 & 10(1+C_2) \\ 10(1+C_2) & 10 \end{bmatrix} \begin{bmatrix} 950.1823I_2 & 985.7863I_2 \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \end{bmatrix} \qquad (41)
\end{aligned}
$$

A simulation for this example is taken with $\Delta M(q) = 0.4M_0(q)$, i.e., 40% disturbance, $\left\| M^{-1}(q)\Delta M(q) \right\| = 0.2857 < \delta_f = 0.5$, $\Delta V_m(q,\dot{q}) = 0.2V_{m0}(q,\dot{q})$ with 20% disturbance, and $\Delta N(q) = 0.2N_0(q)$ with 20% disturbance. The simulation results by MATLAB and Simulink are shown in Figs. 2-3.

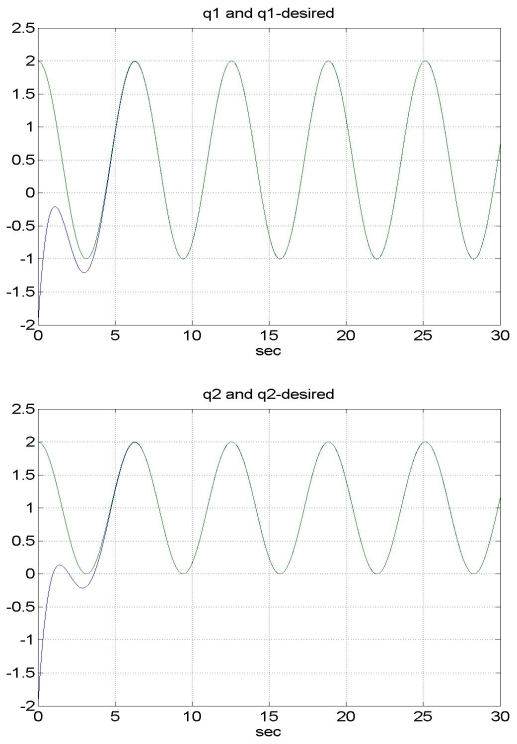


Fig. 2. States and their desired states: (a) $q_1(t)$ & $q_{1d}(t)$, (b) $q_2(t)$ & $q_{2d}(t)$ in Example 1

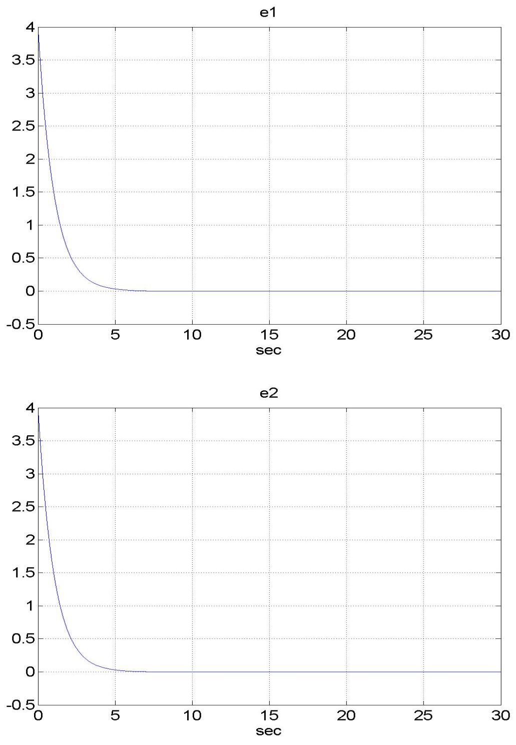


Fig. 3. Error signals: (a) $e_1(t)$, (b) $e_2(t)$ in Example 1

Example 2. Consider a two-link planar robotic system example with the joint mass and the arm mass along the arm length. The mass of joint 1 is $m_1 = 1$ kg, and the mass of joint 2 is $m_2 = 0.5$ kg. The dimensions of two robot arms are linearly reduced round rods. The two terminal radii of the arm rod 1 are $r_{01} = 3$ cm and $r_{t1} = 2$ cm. The two terminal radii of the arm rod 2 are $r_{02} = 2$ cm and $r_{t1} = 1$ cm. Their end cross-sectional areas are $S_{01} = 9\pi$ cm^2, $S_{t1} = 4\pi$ cm^2, $S_{02} = 4\pi$ cm^2, and $S_{t2} = 1\pi$ cm^2. The arm length and mass are $l_1 = 1$ m, $l_2 = 1$ m, $m_{r1} = 5.2$ kg, and $m_{r2} = 1.9$ kg. By Theorem 3, it leads to:

$$\xi_1 = 0.2684\,,\ \xi_2 = 0.2286\,,\ \zeta_1 = 0.4342\,,\ \zeta_2 = 0.3929 \tag{42}$$

By Theorem 3, the nominal model parameters are:

$$M_0(q) = \begin{bmatrix} 5.7301 + 2.4929C_2 & 0.9343 + 1.2464C_2 \\ 0.9343 + 1.2464C_2 & 0.9343 \end{bmatrix} V_0(q,\dot{q}) = -1.2464S_2\dot{q}_2\begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}$$

$$N_0(q,\dot{q}) = g\begin{bmatrix} 5.6579C_1 + 1.2464C_{12} \\ 1.2464C_{12} \end{bmatrix} \tag{43}$$

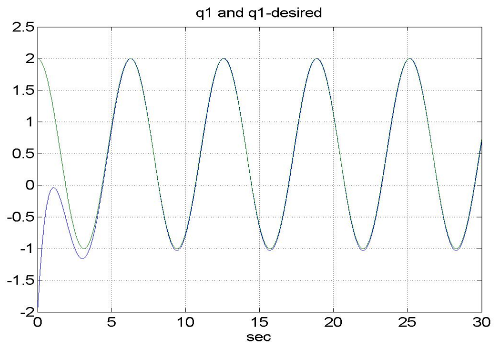
The desired trajectory $q_d(t)$ is the same as in Example 1. The initial states are set as $q_1(0) = -2$, $q_2(0) = 0$, $\dot{q}_1(0) = \dot{q}_2(0) = 0$, i.e., $e_1(0) = 4$, $e_2(0) = 2$, $\dot{e}_1(0) = 0$, $\dot{e}_2(0) = 0$.
The parametric uncertainties in practice are assumed to satisfy (22) with $\delta_f = 0.25$, $\delta_e = 10$, $\delta_N = 10$. Select the adjustable parameters $\varepsilon_1 = 0.0375$ and $\varepsilon_2 = 0.0188$ from (31), the disturbance rejection index $\delta = 0.1$, the relative stability index $\alpha_1 = 0.1$, and the left bound of vertical strip $\alpha_2 = 100$. By Theorem 4, the solution matrix P to (32) and the gain matrix K are

$$P = \begin{bmatrix} 898.87I_2 & 13.55I_2 \\ 13.55I_2 & 12.47I_2 \end{bmatrix}, K = rB'P = [65.0589I_2 \; 59.8659I_2]$$

with $r = 4.8$. The eigenvalues of the closed-loop main system matrix $A - BK$ are $\{-1.1072, -58.7587, -1.1072, -58.7587\}$. The uncertain system has $-\alpha_2 < \mathrm{Re}[\lambda(A_c)] < -\alpha_1$ robustly.
The total control input (law) is :

$$\begin{aligned} f &= M_0\ddot{q}_d + V_0(q,\dot{q})\dot{q} + N_0 - M_0u \\ &= \begin{bmatrix} 5.7301 + 2.4929C_2 & 0.9343 + 1.2464C_2 \\ 0.9343 + 1.2464C_2 & 0.9343 \end{bmatrix} \cdot \begin{bmatrix} -1.5\cos t \\ -\cos t \end{bmatrix} \\ &\quad - 1.2464S_2\dot{q}_2 \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + g \begin{bmatrix} 5.6579C_1 + 1.2464C_{12} \\ 1.2464C_{12} \end{bmatrix} \\ &\quad + \begin{bmatrix} 5.7301 + 2.4929C_2 & 0.9343 + 1.2464C_2 \\ 0.9343 + 1.2464C_2 & 0.9343 \end{bmatrix} \\ &\quad \cdot \left[65.0589I_2 \quad 59.8659I_2\right] \cdot \left[e' \quad \dot{e}'\right]' \end{aligned} \tag{44}$$

The simulation is taken with $\Delta M(q) = 0.25M_0(q)$, $\Delta V_m(q,\dot{q}) = 0.1V_{m0}(q,\dot{q})$, and $\Delta N(q) = 0.1N_0(q)$. The results are shown in Figs. 4-5. It is noticed that the error may be reduced when the gain parameter r is set large.



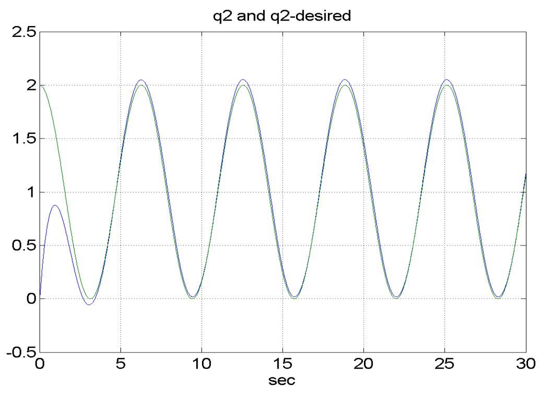


Fig. 4. States and their desired states: (a) $q_1(t)$ & $q_{1d}(t)$, (b) $q_2(t)$ & $q_{2d}(t)$

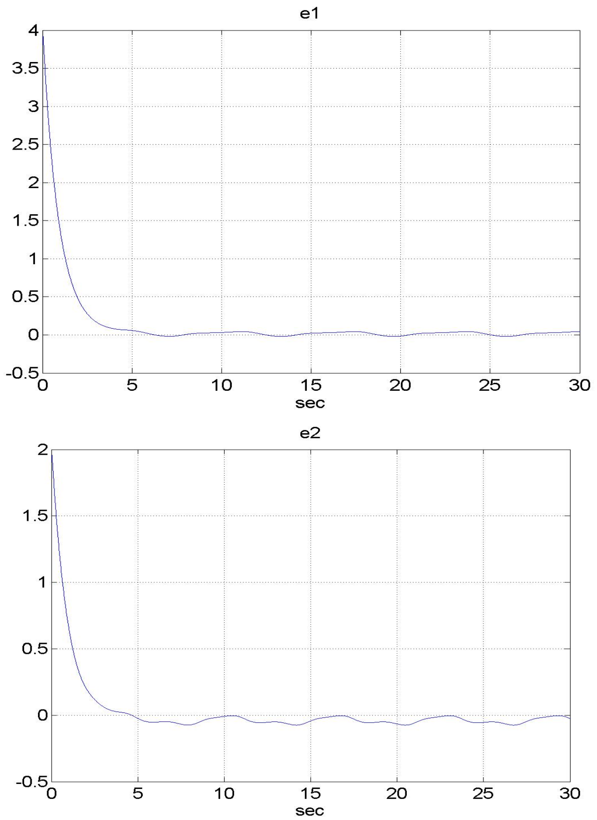


Fig. 5. Error signals: (a) $e_1(t)$, & (b) $e_2(t)$

5. Conclusion

The chapter develops the practical models of two-link planar nonlinear robotic systems with their arm distributed mass in addition to the joint-end mass. The new scaling coefficients are introduced for solving this problem with the distributed mass along the arms. In addition, Theorems 2 and 3 respectively present two special cases: a uniform arm shape (i.e., uniform distributed mass) and a linear reduction of arm shape along the arm length.
Based on the presented new models, an approach to design a continuous nonlinear control law with a linear state-feedback control for the two-link planar robotic uncertain nonlinear systems is presented in Theorem 4. The designed closed-loop systems possess the properties of robust pole-clustering within a vertical strip on the left half s-plane and disturbance rejection with an H_∞-norm constraint. The suggested robust control for the uncertain nonlinear robotic systems can guarantee the required robust stability and performance in face of parameter errors, state-dependent perturbations, unknown parameters, frictions, load variation and disturbances for all allowed uncertainties in (22). The presented robust control does always exist as pointed out in Remark 4. The adjustable scalars ε_i, i=1, 2, provide some flexibility in finding a solution of the algebraic Riccati equation. The designed uncertain system has α_1-degree robust stabilization and δ-degree disturbance rejection. The controller gain parameter *r* is selected such that the designed uncertain linear system achieves robust pole-clustering within a vertical strip. The examples illustrate excellent results. This design procedure may be used for designing other control systems, modeling, and simulation.

6. References

[1]J.J.Craig, *Adaptive control of mechanical manipulators*, Addison-Wesley (Publishing Company, Inc., New York, 1988).

[2]J.H. Kaloust, & Z. Qu, Robust guaranteed cost control of uncertain nonlinear robotic system using mixed minimum time and quadratic performance index, *Proc. 32nd IEEE Conf. on Decision and Control*, 1993, 1634-1635.

[3]J. Kaneko, A robust motion control of manipulators with parametric uncertainties and random disturbances, *Proc. 34rd IEEE Conf. on Decision and Control*, 1995, 1609-1610.

[4]R.L. Tummala, Dynamics and Control – Robotics, in *The Electrical Engineering Handbook*, Ed. by R.C. Dorf, (2nd ed., CRC Press with IEEE Press, Boca Raton, FL, 1997, 2347).

[5]M. Garcia-Sanz, L. Egana, & J. Villanueva, "Interval Modelling of a SCARA Robot for Robust Control", *Proc. 10th Mediterranean Conf. on Control and Automation,* 2002.

[6]S. Lin, & S.-G. Wang, Robust Control with Pole Clustering for Uncertain Robotic Systems, *International Journal of Control and Intelligent Systems*, *28*(2), 2000, 72-79.

[7]S.B. Lin, and O. Masory, Gains selection of a variable gain adaptive control system for turning, *ASME Journal of Engineering for Industry*, *109*, 1987, 399-403.

[8]S.-G. Wang, L.S. Shieh, & J.W. Sunkel, Robust optimal pole-clustering in a vertical strip and disturbance rejection for Lagrange's systems, *Int. J. Dynamics and Control, 5*(3), 1995, 295-312.

[9]S.-G. Wang, L.S. Shieh, & J.W. Sunkel, Robust optimal pole-placement in a vertical strip and disturbance rejection, *Proc. 32nd IEEE Conf. on Decision and Control*, 1993, 1134-1139. *Int. J. Systems Science*, *26*(10), 1995, 1839-1853.
[10]S.-G. Wang, S.B. Lin, L.S. Shieh, & J.W. Sunkel, Observer-based controller for robust pole clustering in a vertical strip and disturbance rejection in structured uncertain systems, *Int. J. Robust & Nonlinear Control*, *8*(3), 1998, 1073-1084.
[11]J.J. Craig, *Introduction to Robotics: Mechanics and Control* (2nd ed., Addison-Weeley Publishing Company, Inc., New York, 1988).

Role of Finite Element Analysis in Designing Multi-axes Positioning for Robotic Manipulators

T.T. Mon, F.R. Mohd Romlay and M.N. Tamin
Universiti Malaysia Pahang, Universiti Teknology Malaysia
Malaysia

1. Introduction

Simulation of robot manipulator in Matlab/Simulink or any other mechanism simulator is very common for robot design. However, all these approaches are mainly concerned with design configuration having little analysis meaning that the robot model is formed by linking the kinematics and solid description, and simulated for alternative configuration of movements (Cleery & Mathur, 2008). Indeed comprehensive design should have analysis at different computational levels. Finite element method (FEM) has been a major tool to develop a computational model in various fields of studies because of its modelling and simulation capability close to reality. Subsequently, modelling and analysis with FEM has become the most convenient way to economically design and analyze real world problems, either in static or dynamic. As a result, huge amount of reports on this topic can be found in the literature (Mackerle, 1999). Unfortunately however, this technique has not comprehensively applied in designing in designing a robot while choosing the best components for the design is as important as having good performance and no environmental impact of the machines over its lifetime. Building block of a robot manipulator is electromechanical system in which mechanical systems are controlled by sophisticated electric motor drives. Since energy saving everywhere is a major challenge now and in future, getting electromechanical design right will significantly contribute to energy saving.

This chapter is dedicated to the application of Finite Element Method (FEM) in designing multi-axes positioning for robot manipulators. Computational model that can predict physical behaviour of dynamic robot manipulators constructed using FE codes is presented, and this is major contribution of the chapter. FEM tools necessary for modelling and analysis of multi-axes positioning are presented in large part. Rather than a FEM discourse, FEM is presented by highlighting mathematics behind and an application example as they relates to practical robotic manipulation. It is, however, assumed that the reader has acquired some basic knowledge of FEM consistent with the expected level of mathematics. Hence, the chapter is organized as follows.

In the early part of the chapter, the important terminologies used in robotics are defined in the background. The material is presented using a number of examples as evidenced in the

published reports. Then the chapter will go to mathematics behind finite element modelling and analysis of kinematics of a structure in 3D space as robotics involves tracking moving objects in 3D space. This will also include mathematical tools essential for the study of robotics, particularly matrix transforms, mathematical models of robot manipulators, direct kinematic equations, inverse kinematic technique and Jacobian matrix needed to control position and motion of a robot manipulator. More emphasis will be on how these mathematical tools can be linked to and incorporated into FEM to carry out design analysis of robot structure.
The rest of the chapter will present application of FEM in practical robot design, detailed development of FE model computable for multi-axes positioning using a particular FE code (ALGOR, 2008), and useful results predicted by the computational model. The chapter will be closed by concluding remarks to choice of FE codes and its impact on the computational model and finally the usefulness of computational model.

2. Background in Robotics

Multi-axis positioning meant here is different movements of a point, or a structure in different directions. This term is drawn from the term usually come with computer numerical controlled machines just as 3-axis, 5-axis and so on, where the 3-axis machine, for instance, implies that it can make a maximum of three different positioning of the controlled elements. Each axis is alternatively referred to as degree of freedom (DOF) that is something to do with motion in a system or a structure. Since the term 'axis' is adopted to represent an element that creates motion, 3-axis positioning means three DOF's, for example (Rahman, 2004). In relation to these definitions, one manipulator of a robot can represent one axis or DOF as the manipulator is the robot's arm, a movable mechanical unit comprising of segments or links jointed together with axes capable of motion in various directions allowing the robot to perform tasks. Typically, the body, arm and wrist are components of manipulators. Movements between the various components of the manipulator are provided by series of joints.
The points that a manipulator bends, slides or rotates are called joints or position axes. Position axes are also called the world coordinates. The world coordinate system is usually identified as being a fixed location within the manipulator that serves as an absolute frame of reference.
In general, the manipulator's motion can be divided into two categories: translation and rotation. Although one can further categorize it in specific term such as a pitch (up-and-down motion); a yaw (side-to-side motion); and a roll (rotating motion), any of these is fall into either translation or rotation. The individual joint motion associated with either of these two categories is referred to degree of freedom. Subsequently, one degree of freedom is equal to one axis. The industrial robots are typically equipped with 4-6 axes.
The power supply provides the energy required for a robot to be operated. Electricity is the most common source of power and is used extensively with industrial robots. Payload is the weight that the robot is designed to lift, hold, and position repeatedly with the same accuracy. Hence, the power supply has direct relation to the payload rating of a robot.
Among the important dynamic properties of a robot that properly regulates its motion are: stability, control resolution, spatial resolution, accuracy, repeatability and compliance. To take these factors into account in the design of a robot is a complex issue. Lack of stability

occurs very often due to wear of manipulator components, movement longer than the intended, longer time to reach and overshooting of position.

Control resolution is all about position control. It is a function of the design of robot control system and specifies the smallest increment of motion by which the system can divide its working space. It is the smallest incremental change in position that its control system can measure. In other words, it is the controller's ability to divide the total range of movements for the particular joint into individual increments that can be addressed in the controller. This depends on the bit storage capacity in the control memory. For example, a robot with 8 bits of storage can divide the range into 256 discrete positions. The control resolution would be also defined as the total motion range divided by the number of increments. For example, a robot has one sliding joint with a full range of 1.0 m. The robot control memory has 12-bit storage capacity. The control resolution for this axis of motion is 0.244mm. The spatial resolution of a robot is the smallest increment of movement into which the robot can divide its work volume.

Mechanical inaccuracies in the robot's links and joint components and its feedback measurements system (if it is a servo-controlled robot) constitute the other factor that contributes to spatial resolution. Mechanical inaccuracies come from elastic deflection in the structural members, gear backlash, stretching of pulley cords and other imperfections in the mechanical system. These inaccuracies tend to be worse for large robot simply because the errors are magnified by the large components. The spatial resolution is degraded by these mechanical inaccuracies.

Rigidity of the structure also affects the repeatability of the robot. Compliance is a quality that gives a manipulator of a robot the ability to tolerate misalignment of mating parts. It is essential for assembly of close-fitting parts. In an electric manipulator, the motors generally connect to mechanical coupling. The sticking and sliding friction in such a coupling can cause a strange effect on the compliance, in particular, being back-drivable.

An Off-line programming system includes a spatial representation of solids and their graphical interpretation, automatic collision detection, incorporation of kinematic, path planning and dynamic simulation and concurrent programming. The off-line programming will grow more in the future because of graphical computer simulation used to validate program development. It is important both as aids in programming industrial automation and as platforms for robotic research (Billingsley, 1985; Keramas, 1999; Angeles, 2003).

3. Mathematical Foundation

Forward and reverse kinematics methods are the principal mathematics behind typical modeling, computation and analysis of robot manipulators. Since the latter is deduced from the former, broader review will focus on some related mathematics of the former. Forward kinematic equation relates a pose element to the joint variables. The pose matrix is computed from the joint variables. The position and orientation of end-manipulator (also-called the end-effector, the last joint that directly touches and handles the object) is computed from all joint variables. The position and orientation of the end manipulator are computed from a set of joint variable values which are already known or specified. The computation follows the arrow directions starting from joint 1 as illustrated in Fig. 1. It should be noted that in kinematics analysis, the manipulators are assumed to be rigid.

3.1 Defining the location of an object in space

As a general case, the object location in 3D space is considered. A matrix representation is widely used to represent the object location as it is convenient and easy to handle especially when the location is changed. Two parameters are needed to define the object location: position and orientation. Basically, a homogenous vector 'v' is represented as

$$\mathrm{v} = \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} \tag{1}$$

if v is a free vector or

$$\mathrm{v} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2}$$

if v represents the position of a particular point in the usual coordinates system (x, y, and z). This coordinates system is referred to a frame. These frames are used to track an object location in space. As shown in Fig. 2, the frame $\mathbf{F_o}$ is attached to a fixed point while another $\mathbf{F_A}$ to an object. The object position is described by the vector pA of the origin A of the frame $\mathbf{F_A}$. The orientation of the object is given by the homogeneous vectors of each unit vectors x_A, y_A and z_A of $\mathbf{F_A}$ with respect to $\mathbf{F_o}$.
Then the object location is mathematically represented by a post matrix 'P' as:

$$\mathrm{P} = [\mathrm{x_A}\ \mathrm{y_A}\ \mathrm{z_A}\ \mathrm{p_A}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where the matrix

$$\mathrm{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{4}$$

and

$$\mathbf{p_A} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{5}$$

formed by the coordinates of the vectors xA, yA and zA is a rotation matrix that holds the orientation of the object while the $\mathbf{p_A}$ holds the position of the object. In a compact form, the pose matrix can be written as:

$$\mathbf{P}=\begin{bmatrix} \mathrm{R} & \mathbf{p} \\ \mathrm{O} & 1 \end{bmatrix} \tag{6}$$

where O refers to a 1x3 vector of zeros.

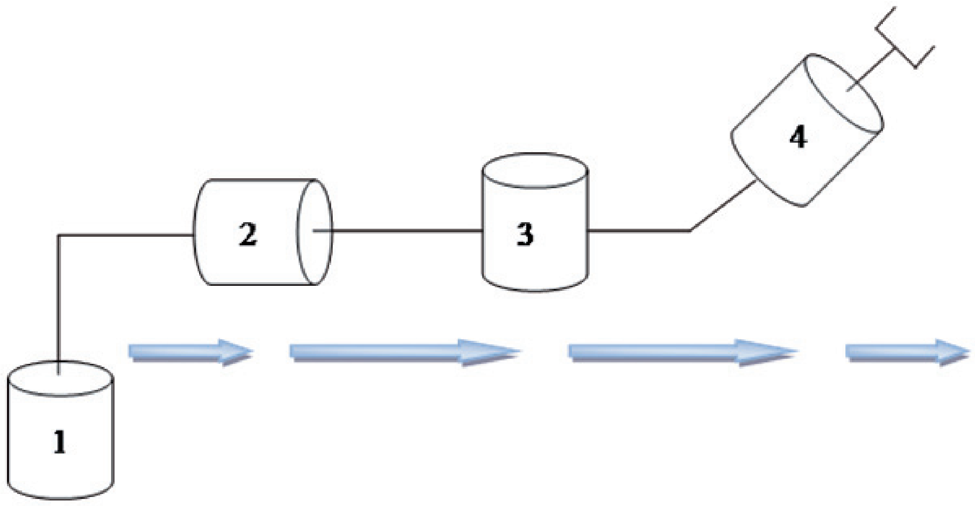


Fig. 1. Forward kinematics

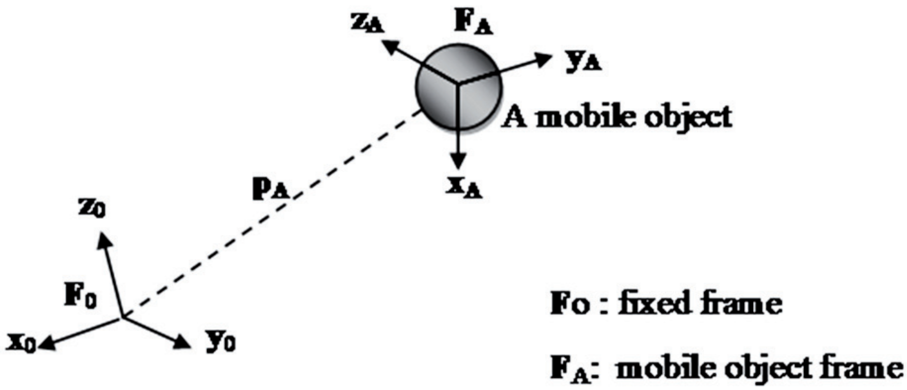


Fig. 2. The object with respect to frames.

When a point, say Q given by its coordinate vector ${}^{A}q = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ with respect to frame $\mathbf{F}_A$, is transformed to the frame $\mathbf{F}_O$, the transformed vector say ${}^{o}q$ can be expressed as:

$$^{o}q = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} + \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = R\ {}^{A}q + p \qquad (7)$$

In compact form

$$^{o}q = {}^{o}T_A\ {}^{A}q \qquad (8)$$

where $^{o}T_A = P$

Similarly in alternative representation of frame transforms, simple space translation and rotation can be conveniently represented as a compact-form matrix as:

$$^{T}\mathbf{q} = \mathbf{T}_u\ \mathbf{q} \qquad (9)$$

$$^{R}\mathbf{q} = \mathbf{R}_{v,\theta}\ \mathbf{q} \qquad (10)$$

where $\mathbf{q} = \begin{bmatrix} x_q \\ y_q \\ z_q \end{bmatrix}$ is coordinates vector of a point Q, $\mathbf{T}_u$ is a translation vector, $\mathbf{R}_{v,\theta}$ is a rotation vector, $^{T}\mathbf{q}$ is coordinates vector of a point Q' where Q is translated by $\mathbf{T}_u$ and $^{R}\mathbf{q}$ coordinates vector of a point Q' where Q is rotated around $\mathbf{v}$ by an angle θ.

Furthermore, the translations and rotations along the reference axes, called canonical translations/rotations, have the homogeneous matrix of the following form respectively as:

$$T_x(d) = \begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11)$$

$$T_y(d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12)$$

$$T_z(d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (13)$$

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha \\ 0 & s_\alpha & c_\alpha \end{bmatrix} \qquad (14)$$

$$R_x(\beta) = \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \tag{15}$$

$$R_z(\gamma) = \begin{bmatrix} c_\gamma & -s_\gamma & 0 \\ s_\gamma & c_\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

Where $T_x(d), T_x(d), \text{and } T_x(d)$ are translations of a distance, 'd' in x, y and z directions respectively, and $R_x(\alpha), R_y(\beta), \text{and } R_z(\gamma)$ are rotations about the respective axis by the respective angle with $c_{()} = \cos\,()$ and $s_{()} = \sin()$. These can be expanded to arbitrary transformations. Refer to Manseur (2006) for details.

3.2 DH parameters

In the conventional analysis of motion of robot manipulators, the Denavit-Hartenberg (DH) modelling technique is commonly used as a standard technique. Reference frames are assigned to each link based on DH parameters, starting from the fixed link all the way to the last link. The DH model is obtained by describing each link frame with respect to the preceding link frame. The original representation of one frame with respect to another using pose matrix requires a minimum of six parameters. The DH modelling technique reduces these parameters to four, routinely noted as:

di, the link offset,
ai, the link length,
θi, the link angle and
αi, the link twist as illustrated in Fig. 3.

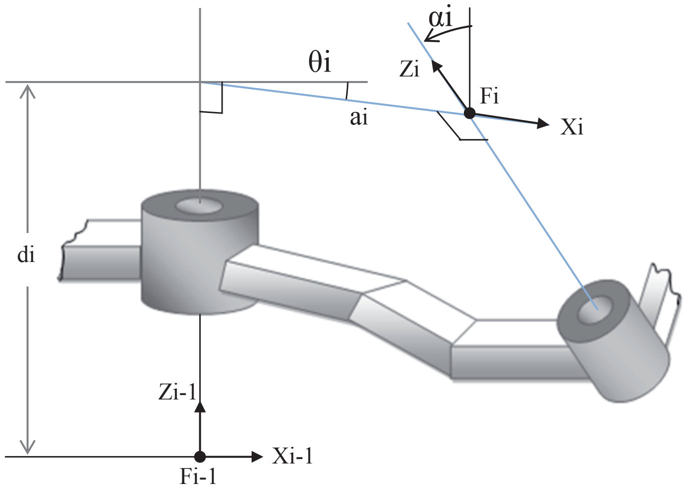


Fig. 3. Illustration of DH parameters

Referring to Fig. 4, in relation to the link frames and DH parameters, homogenous frame transform from the frame (i) to (i-1) can be generally written as

$$ {}^{i-1}\mathbf{A}_i = T(z, d_i)\ R(z, \theta_i)\ T(x, a_i)\ R(x, \alpha_i) \tag{17}$$

where T(z, di) is translation from Fi-1 to Fd, R(z, θi) the rotation transform from Fd to Fθ, T(x, ai) the translation transform from Fθ to Fa, R(x, αi) the rotation transform from Fa to Fi.

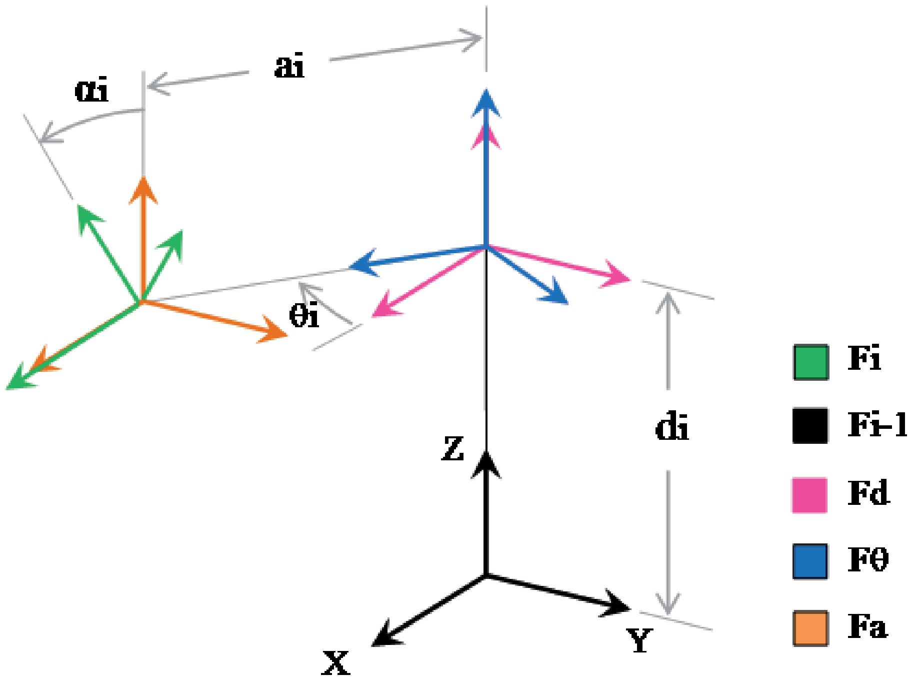


Fig. 4. Frame transformation of the DH parameters.

The matrix ${}^{i-1}\mathbf{A}_i$ is then used to express the transform of the end frame to the base frame for 'n-axis' or 'n-joint' robot manipulators. That is

$$ {}^{o}u = A_1 A_2 \ldots \ldots A_{n-1} A_n \quad {}^{n}\mathbf{u} \tag{18}$$

or

$$ {}^{o}u = T_n \ {}^{n}\mathbf{u} \tag{19}$$

where $T_n = A_1 A_2 \ldots \ldots A_{n-1} A_n = \mathbf{P}$ = pose matrix of the end-effector. This process is called forward kinematic in modelling, computation and analysis of robotic manipulator.
In the reverse kinematics method, as it means, one or more sets of joint variables are computed from the known end-effector pose matrix. As such the direction of computation is

revered from that of the forward kinematics as illustrated in Fig.5. Nevertheless, the same basic concept of forward kinematics is applied except the fact that the mathematical manipulation becomes different and sophisticated. To be brief, the equations derived from the forward kinematics method are reduced to a set of equations involving as few unknown joint variables as possible using a kinematic function that depends on only certain joint variables. This results simplified set of equation that can be solved analytically. Details can be found elsewhere (Manseur, 2006).

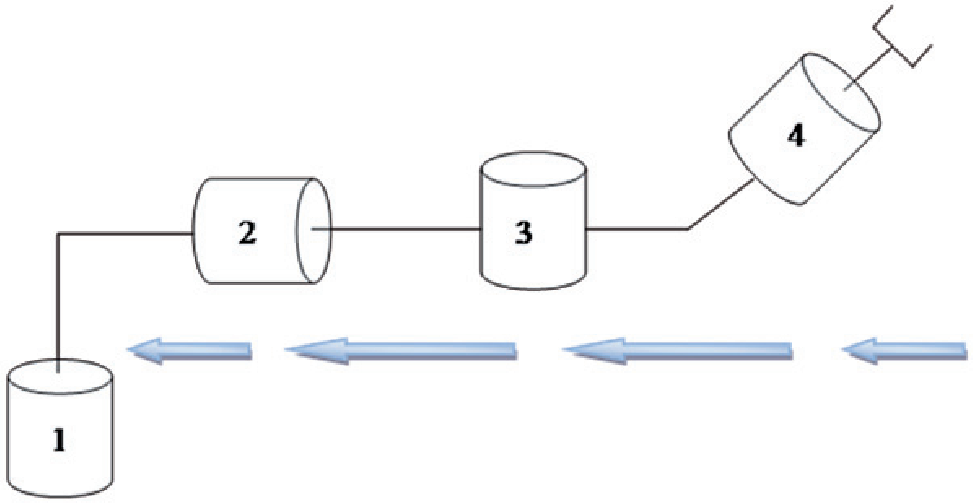


Fig. 5. Reverse kinematics

3.3 Velocity Kinematics

When the time factor is taken into account in the movement of a manipulator, *Jacobian* matrix is used to manipulate the transformation of manipulator. The generalized velocity vector **V** of the end-effector with respect to the base frame can be a form of a linear velocity vector v and an angular velocity vector ω:

$$\mathbf{V} = \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{20}$$

where

$$v = \begin{bmatrix} dp_x/dt \\ dp_y/dt \\ dp_z/dt \end{bmatrix} \tag{21}$$

and

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{22}$$

For the case of n-joint robot, the relationship between the **V** and the joint velocities can generally be expressed as

$$V = J(q)\,\dot{q} = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix} \dot{q} \tag{23}$$

Where **q** = $[q_1 \quad q_2 \quad \cdots\cdots \quad q_n]^T$ is the joint variable and $\dot{q}$ = $[\dot{q}_1 \quad \dot{q}_2 \quad \cdots\cdots \quad \dot{q}_n]^T$ is corresponding velocity vector. The matrix J(q) is called the Jacobian matrix. It is the mathematical relationship between joint motion and end-effector motion. It plays an important role in the analysis and control of robotic motion owing to the fact that it establishes a linear relation between velocity vectors in Cartesian space and in joint space. Details of derivation of J and its computation can be found in (Manseur, 2006).

3.4 Typical Robot Simulators

Nowadays softwares are readily available to generate and solve the kinematic equations. BUILD, SKEG and SIMULINK are some of those. Typical robot simulators allow the programming of a manipulator positioning in Cartesian space and relate a Cartesian co-ordinate set to the robot model's joint angles known as the inverse kinematics transformation (Zlajpah, 2008).

Kinematic description is made up of joint rotation and position information. The inverse kinematic links the kinematics and solid descriptions. During the linking process, a sequence of procedure is called to solve a particular position of the robot. For example, the method of solving 6 DOFs robot as shown in Fig. 6 requires the following steps:

i. From tool position and orientation, calculate wrist position
ii. Calculate angle of rotation of axis 1
iii. Calculate angle 2 and 3 from the two link mechanism connecting the shoulder and the wrist
iv. Calculate the remaining joint angles using spherical trigonometry [1]

Robot simulators do not allow dynamic effects or systematic errors in either trajectory calculation or inverse kinematic processing. The implementation of a full dynamic model would be out of question for a general simulator (e.g. BUILD simulator).

Robot errors can be categorized into two: static error related to end points and dynamic error related to path. The formers result from deviation of the achieved position from the demand position. The latter comes from mismatch between a desired path and the true path. The computation of such information is very appropriate for a solid modelling system in which methods for analyzing space are well developed. However in these simulations, there will be differences between the actual robot performance and the modelled robot performance. The difference will depend on the assumptions made when designing robot model.

A particular robot's physical kinematics will deviate from the idealized model due to manufacturing tolerances and this will lead to a reduction in the accuracy of the robot's positioning with respect to a Cartesian space. The weight affects inaccuracies due to compliance of robot manipulators (Billingsley, 1985).

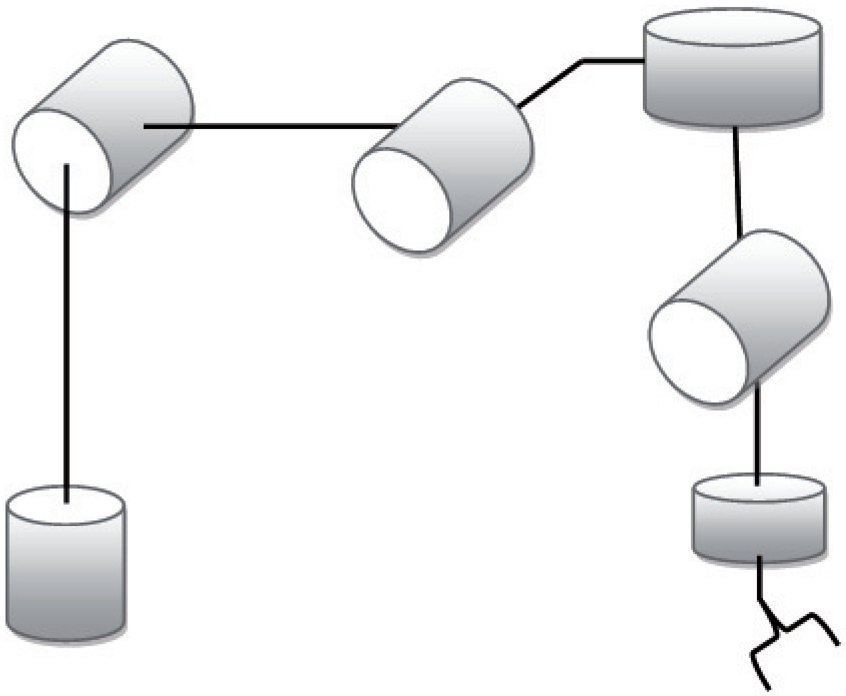


Fig. 6. Robot with 6 DOFs.

To sum up, in conventional modelling technique, the robot manipulators are modelled as a kinematic model. The accuracy of simulated motion in this technique is questionable. Additionally, the model is inadequate for the analysis of the dynamics laws of motion in which inertial parameters, link masses and shapes, and applied forces and torques must be considered. This is where FEM comes into play.

4. Finite Element Method

It is hard to completely define finite element method (FEM) in one sentence or even a paragraph. One may define it as a numerical method for solving engineering problem and physics, or a method to computationally model reality in a mathematical form; either one is acceptable indeed. However, for more complete definition of FEM, the following paragraph may suit it better.
"A continuum is discretized into simple geometric shapes called finite elements; constitutive relations, loading and constraints are defined over these elements; assembly of elements results set of equations; solution of these equations gives the approximate behaviour of the continuum."
The procedure of using FEM to analyze the system behaviour generally includes:

- Discretize the system and select element type.
- Select displacement function.
- Define strain-displacement and stress-strain relation.
- Derive stiffness matrix and equations.
- Assemble global equations.
- Introduce boundary condition (BC)
- Solve for unknown variables.

Finite element method (FEM) is now considered matured to develop a computational model to predict solution domain accurately without the need of expensive experimental work. Some advantages of using FEM are being economically-viable technique, flexibility to analyze various physical problems, and capability of modelling complex boundary conditions and material behaviour. Most importantly, the aspects of FEM applications are easily adaptable to engineers' needs (Chandrupatla, & Belegundu, 2002; Hutton, 2004). .

4.1 Multi-axes positioning with FEM

Multi-axis positioning in FEM can be modelled with manipulation of parameters that can be found under the category of boundary conditions. The concept of DOF here can be illustrated in a simple spring-mass system. Fig. 7(a) shows a system consisting of one spring and one mass where one DOF is imposed on the mass as it displaces due to the weight or applied force. When there is more than one spring or mass, DOF also becomes more than one as shown in Fig. 7(b). In both cases, DOF is in translation. In general structural dynamics, motion can be in translation or rotation defined in standard coordinate axes, x, y or z. These motions are illustrated in Fig. 7(c) where Ux, Uy, and Uz are translations in x, y and z directions while Rx, Ry, and Rz are rotations about x, y and z axes respectively. Hence, in any general-purpose finite element (FE) code, maximum of six DOFs can be defined at a node point for structural analysis.

4.2 Structural analysis with FEM

Basically, computation in FEM involves developing matrix equations for the system and manipulation of these equations. Therefore, one may say that the heart of FEM is rooted in matrix manipulation. For simplest static problem shown in Fig. 8 the system is discretized into two spring elements and three nodes. The FE equation for this system with reference to the coordinate axes can be derived as:

$$\begin{bmatrix} k_1 & -k_1 & 0 \\ -k_1 & k_1+k_2 & -k_2 \\ 0 & -k_2 & k_2 \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \end{Bmatrix} \tag{24}$$

Where $k_1$and $k_2$are spring constants, $U_1, U_2,$ and U_3 are displacements or DOFs at nodes, and $F_1, F_2,$ and F_3 are applied forces.

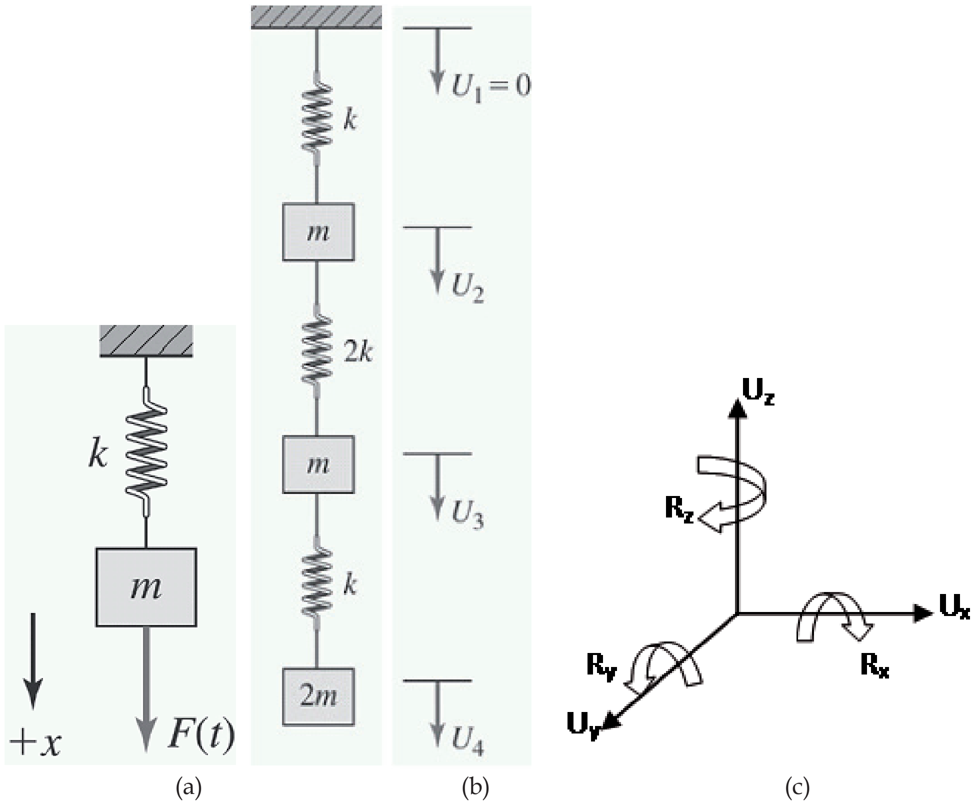


Fig. 7. Motion in standard coordinate axes

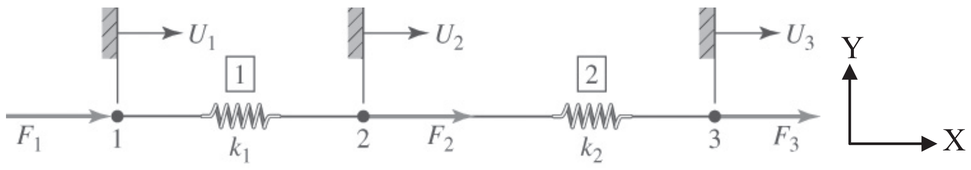


Fig. 8. FE model with spring elements

Detail of derivation procedure is available in (Hutton, 2004). In order to solve for any unknown in equation 24, all possibly-known BC's and forces are imposed. The matrix equations are then solved using an appropriate solver. Thus in FEM, the coordinate axes can be defined up to the user, the kinematics and physics of the system are taken care of in BC and material model respectiely, and all necessary computations are performed by the solver. The FE equations are becoming complicated and the more advanced mathematics is required to solve when the system is in dynamics and constructed of complicated geometry in 3D where the element geometry is not so simple as the above example, and also number of elements can go to thousands. For dynamics analysis, finite element formulation is usually based on Lagrangian mechanics which states:

$$L = T \tag{25}$$

where L is Lagrangian term and T is total kinetic energy of the bod. In dynamic problem, mass and acceleration effects come into picture. For 3D body, T can be expressed as:

$$T = \frac{1}{2}\int\int\int(\dot{u}^2 + \dot{v}^2 + \dot{w}^3)\rho\ dx\ dy\ dz \tag{26}$$

Where ρ is mass density of the body, and u˙,v˙,"and" w˙ are velocity components of differential mass associated with displacements (u, v, w) in the coordinate directions. If computational domain is discretized by

$$u(x,\ y,\ z,\ t) = \sum_{i=1}^{M} N_i(x,\ y,\ z)u_i(t) \tag{27}$$

$$v(x,\ y,\ z,\ t) = \sum_{i=1}^{M} N_i(x,\ y,\ z)v_i(t) \tag{28}$$

$$w(x,\ y,\ z,\ t) = \sum_{i=1}^{M} N_i(x,\ y,\ z)w_i(t) \tag{29}$$

the element kinetic energy can then be expressed as

$$T^{(e)} = \frac{1}{2}\int\int\int\begin{pmatrix}\{\dot{u}\}^T[N]^T[N]\{\dot{u}\} + \{\dot{v}\}^T\{N\}^T[N]\{\dot{v}\} \\ + \{\dot{w}\}^T[N]^T[N]\{\dot{w}\}\rho\ dV^{(e)}\end{pmatrix} \tag{30}$$

In compact form,

$$T^{(e)} = \frac{1}{2}\{\dot{\delta}\}^T[m^{(e)}]\{\dot{q}\} \tag{31}$$

being nodal velocities as

$$\{\dot{q}\} = \begin{Bmatrix}\{\dot{u}\} \\ \{\dot{v}\} \\ \{\dot{w}\}\end{Bmatrix} \tag{32}$$

and the element mass matrix as

$$[m^{(e)}] = \iiint \begin{bmatrix} [N]^T[N] & 0 & 0 \\ 0 & [N]^T[N] & 0 \\ 0 & 0 & [N]^T[N] \end{bmatrix} \rho \, dV^{(e)} \tag{33}$$

Then based on the Lagrangian approach, the global equations for the FE model of a structure subjected to dynamic loading can be expressed in the form

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0, \qquad i = 1, \; P \tag{34}$$

where P is the total number of DOFs. Solving these equations yields a general system of ordinary equation:

$$[M]\{\ddot{q}\} + [K]\{q\} = \{F\} \tag{35}$$

Finite element solver will solve equation (35) for displacements thereby strains and stresses.

5. Application Example

Application of FEM in modelling and analysis of multi-axis positioning is demonstrated in this section. The real robot manipulators (model: Fanuc M-6iB) shown in Fig. 9 is used as a reference for realistic manipulator geometry. Fig. 10 shows the number of axes of the reference robot. The axes were assigned as J1, J2, and so on. All axes are in rotation. This Fig. was used for the demonstration purpose of the axes. The actual rotation of each axis can be found in (Fanuc Robotics, 2008). Modelled robot geometry was simplified and designed in Solidworks as shown in Fig. 11 keeping the important features of the reference robot manipulators. Despite simplification to manipulator geometries for physical analysis, motions of each manipulator were identical to the real robot, except the fact that virtual robot would have one DOF .i.e. J6 less. The dimensions were based on approximate scale of the reference robot. Small holes were created in each component and used as reference points for assembly into complete robot geometry. These holes would also be useful for defining joints in finite element analysis.

As shown in Fig. 11, the virtual robot has five axes. Each axis would be called J1, J2, and so on just as assigning in the reference robot specification. Solidworks model was saved as the IGES file format that can be exported to finite element package for simulation of physical behaviour.

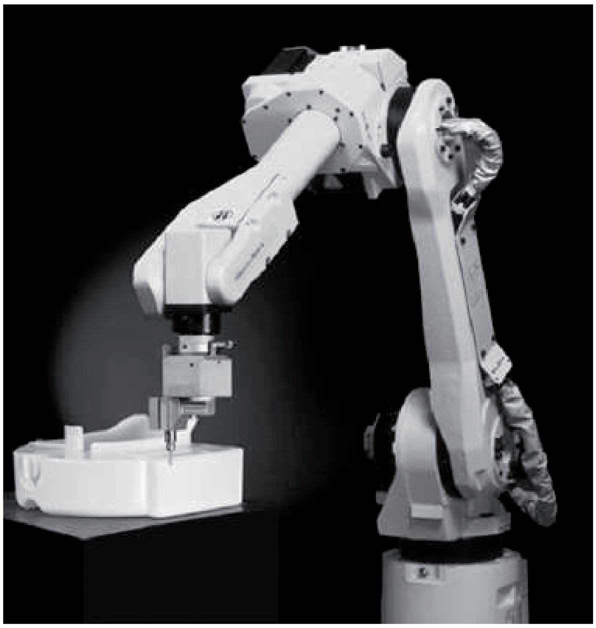


Fig. 9. Reference robot – model FANUC M-6iB (Fanuc Robotics, 2008).

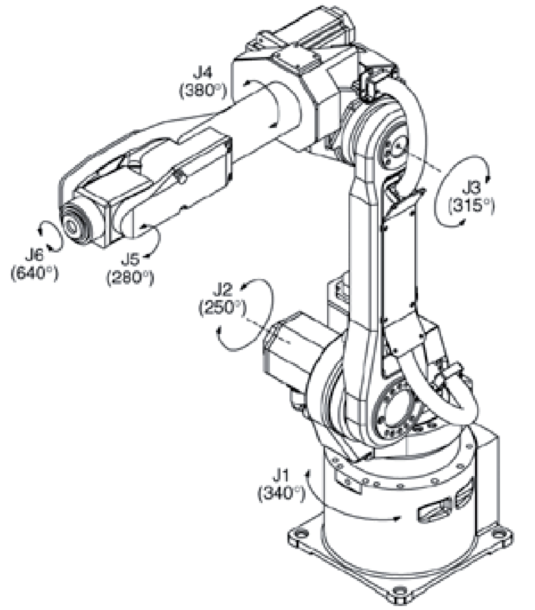


Fig. 10. The axes of the reference robot manipulators (Fanuc Robotics, 2008)

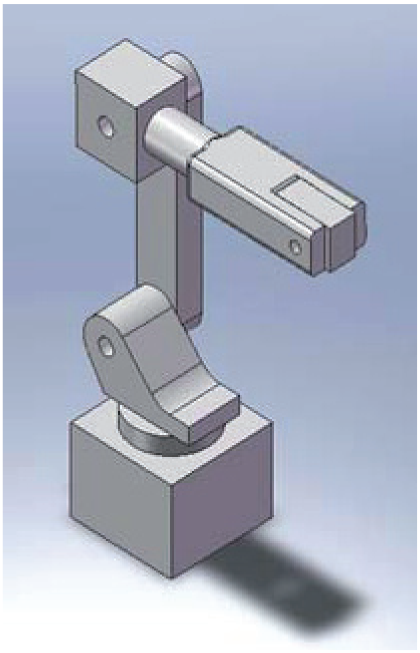


Fig. 11. Simplified robot model geometry

5.1 Finite element analysis

The model geometry of robot manipulators was imported from Solidworks into finite element environment for physical analysis. General-purpose FE codes (ALGOR version 21.2) were tested for the analysis. Fig. 12 shows the discretized model where 8-node brick elements were used as element type for all components. Total number of elements used was 20875 brick elements in manipulators (J1 – J5) and 736 beam elements at joints. According to the available information, steel material (AISI 4130) was used for J5 and the base and the remaining parts were made of aluminum 2024-T3. Linear material model was assumed and the properties of these materials were modelled with data available in FE package material library. Element formulations were solved for large displacement using total Lagrangian approach.

In order to define five axes in the manipulators, joint mesh method and beam elements were used. Pin joint type was chosen for relative movement of each manipulator. Beam elements were selected for joint mesh as these elements can withstand the moment induced. Boundary conditions were then defined at each joint of the manipulator using local coordinate system assigned at joints to create motion being consistent with the reference robot. As this type of robot has to be fixed on the floor, at ceiling or on the table when in use, all nodes at the base were constrained in all directions. To rotate manipulator J1 relative to J2, nodal rotation about Z-axis was prescribed at its joint. At the same time, J2 is rotating about its own axis X as well as swinging about Z. The former DOF is defined by assigning initial velocity at corresponding joint and the latter by using rotational DOF at the node joining with J4. Similarly, the remaining DOFs were defined to create motions of other manipulators. To eliminate the unknowns in solving FE equations, additional nodal and surface boundary conditions were applied on all joint node and surfaces of manipulators respectively in such a way that translation or rotation in unnecessary directions were constrained.

Mechanical event simulation (MES) with linear material model was used to analyze the kinematics as well as physical response due to dynamic effects. MES can simultaneously analyze mechanical behaviour involving large deformations, nonlinear response, kinematic motion and forces caused by that motion and predict resulting stresses and strains. Details can be found in (ALGOR, 2008).

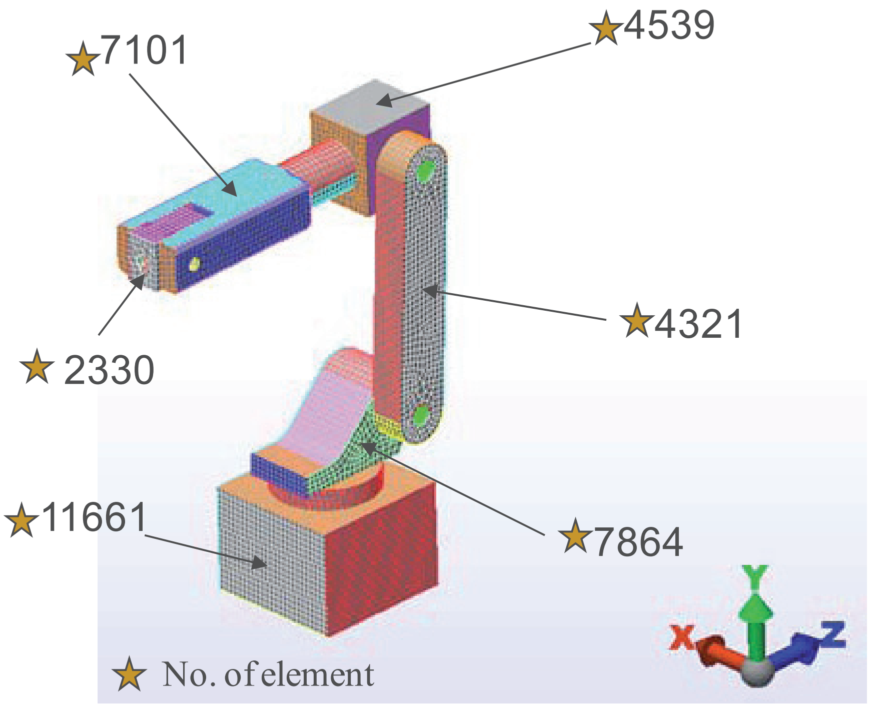


Fig. 12. Finite element model of robot manipulators.

5.2 Discussion

The idea here is the creation of multi-axis positioning with consideration of physical behaviour through FEM. This was done by manipulating available FE codes constructed for boundary conditions. Furthermore, a lot of information can be extracted from the FE analysis. The useful information on dynamic structural behaviour includes stress, strain, displacements and moment/torque. The stress-strain and displacement plots, for instance, are very important to evaluate the design. The information on torque is useful to choose a proper motor size for the real robot. Inertia effect, weight effect and can also be taken into account in the model. In this simple example, these factors were ignored.

Motions of manipulators were analyzed only for 0.1 s due to computational limitation of FE code applied although the motions of the manipulators in reference robot were specified for 1s in the specification. Even that the computing time took 16 hrs to complete the analysis. Fig. 13 demonstrates the positioning of robot manipulators (axes) as defined by boundary conditions. It also depicts the deformation of each manipulator due to dynamic loads. Fig. 14 – 18 shows the predicted displacements of corresponding manipulators. Table 1 shows comparison between the rotation of the reference robot manipulators and that of virtual robot manipulators. The computational results are consistent with the motion of the reference robot manipulators from view point of motion speed.

Axes	Ref robot spec.		Computed rotation	% error
	1 s	0.18 s	0.18 s	
J1	150°	27°	26.5°	2
J2	160	28.8	28.87	0.02
J3	170	30.6	30.8	0.06
J4	400	72	72	0
J5	400	72	71.93	0

Table 1. Comparison between the actual and computed rotations

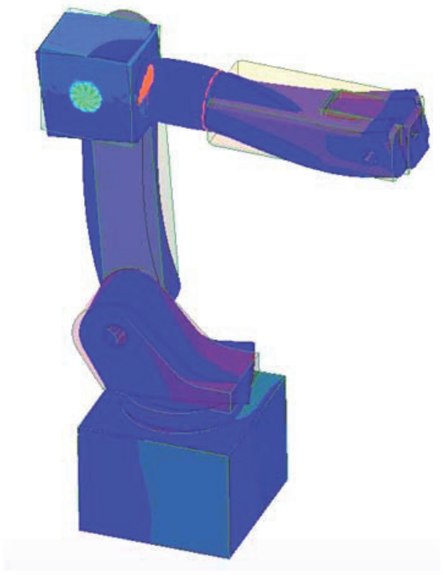


Fig. 13. The deformed plot of robot manipulators.

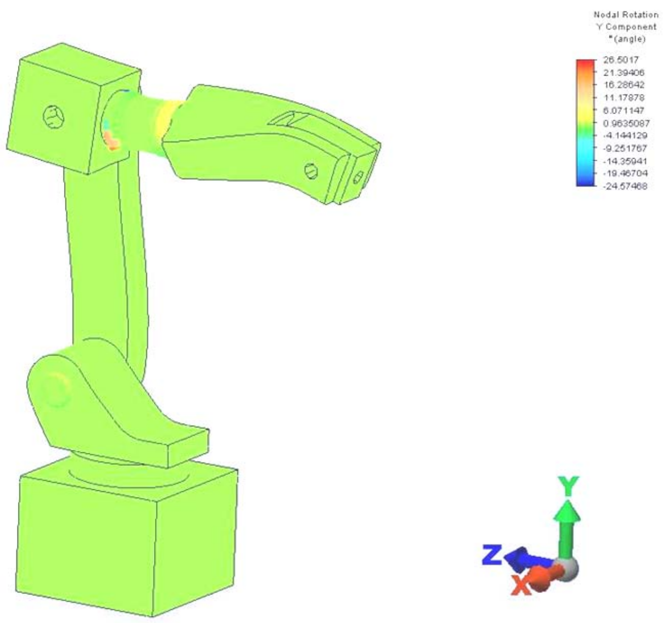


Fig. 14. Computed Rotational displacement of J1

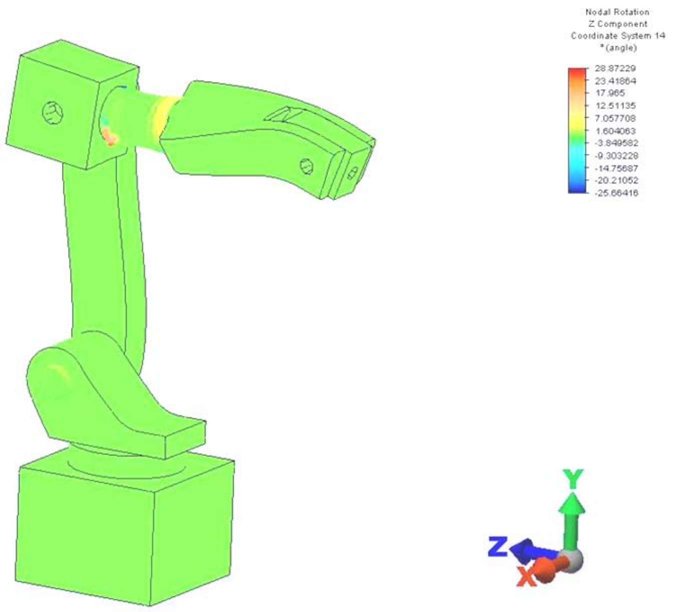


Fig. 15. Computed Rotational displacement of J2

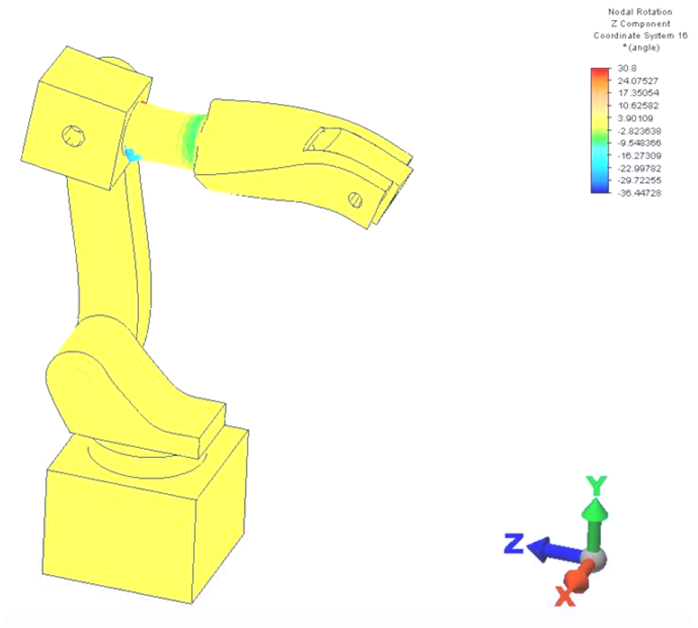


Fig. 16. Computed Rotational displacement of J3

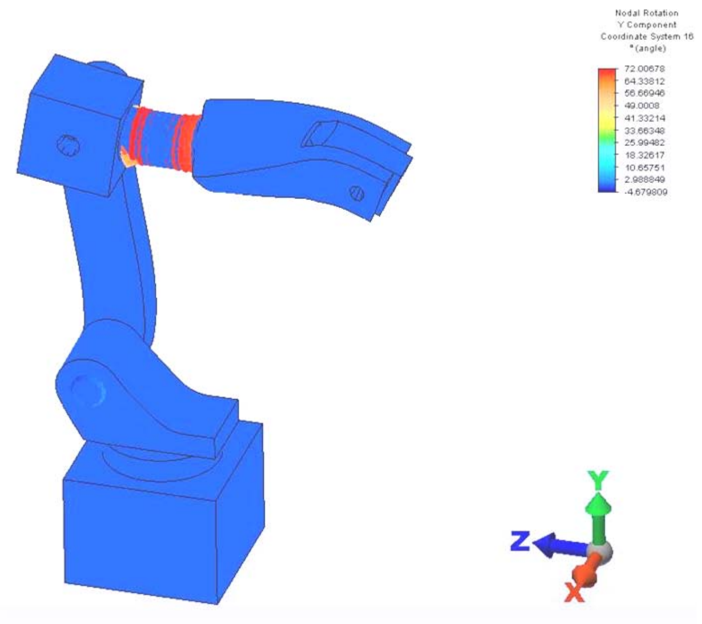


Fig. 17. Computed Rotational displacement of J4

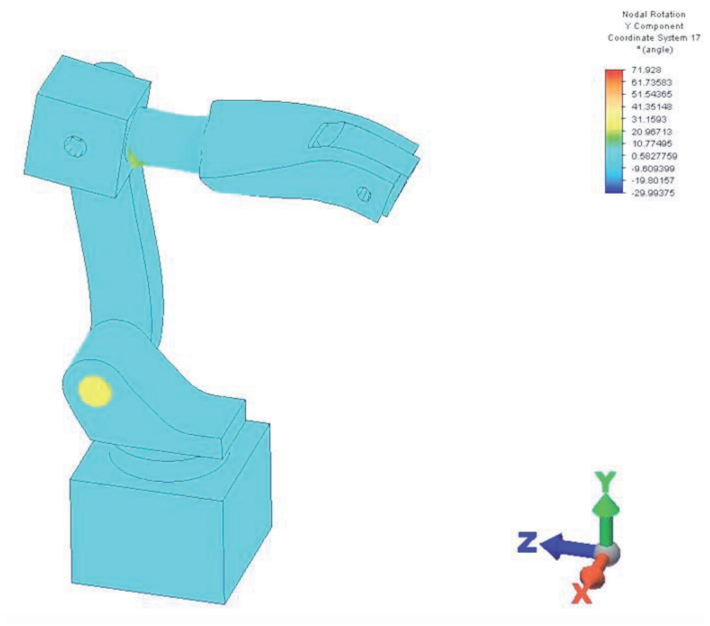


Fig. 18. Computed Rotational displacement of J5

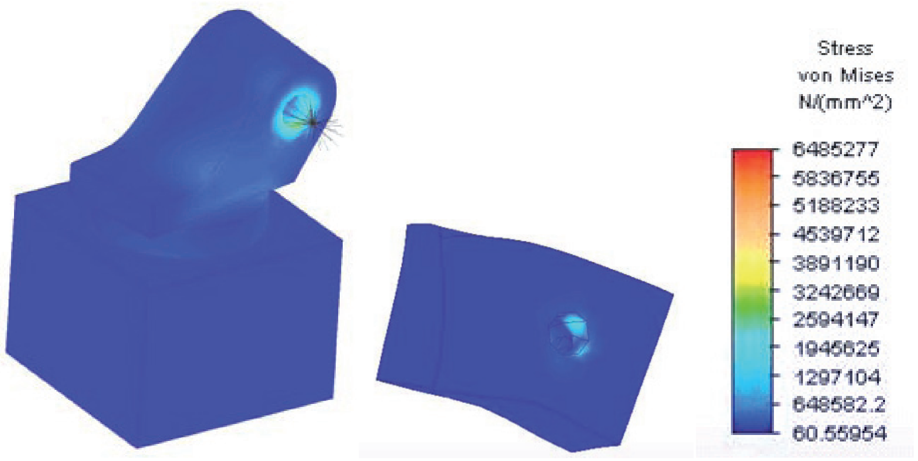


Fig. 19. Computed effective stress.

Fig. 19 shows contour plots of dynamic stress at joints. Contour plots show the deformation of each component clearly. The twisting and bending of the manipulator can be clearly seen. The main reason is that in current model the friction effect at the contact and clearance were ignored. The sizes of holes in mating parts for assembly were the same as if zero clearance. Presence of bearing was not there. The size and geometry of manipulators were merely approximation which was not based on proper design calculations. Consequently, severe twisting would occur especially at the base where it supports all manipulators and endure the dynamically-exerted forces. The computed effective stress is useful to determine the failure of each manipulator due to dynamic loading. Fig. 20 shows the computed maximum torque required to rotate J5. Computed torque can be used to estimate motor power required for robot manipulators.

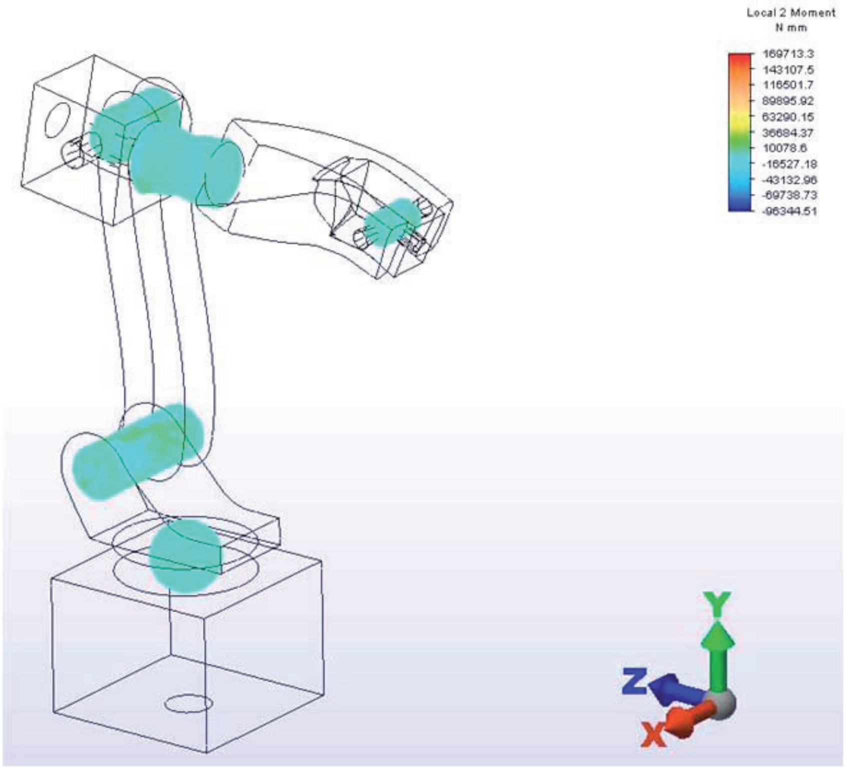


Fig. 20. Computed torque.

6. Conclusion

The FEM is capable of analyzing multi-axes positioning as well as physical response to dynamic load. The current FE code used for dynamic analysis is flexible to analyze physical response of the system due to multi-axes positioning involving large displacement.
Capability of FE code may be different from that of another i.e. FE analysis greatly depends on the FE code. Some FE codes have limitations. Such limitations are usually model size (i.e. numbers of elements and nodes), meshing, material models, analysis type and solver type. In addition, the hardware capacity affects computational power. In future work, weight factor, mesh design and an apparopriate material model will be takne into account in the finite element modeling and analysis.

7. References

ALGOR, *User's Guide*, 2008.

Angeles, J. (2003). *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, Second Edition, Springer-Verlag New York, Inc.

Billingsley, J. (1985). *Robots and automated manufacture*, IEEE Control Engineering Series 28, Peter Peregrinus Ltd, London, pp. 11-23, 29-33.

Chandrupatla, T.R & A.D. Belegundu, A.D. (2002). *Introduction to Finite Elements in Engineering*, Third Edition, Prentice Hall, ISBN : 0-13-207036-7.

Cleery, B.M. & Mathur, N. (2008). Mechanical Engineering Magazine, ASME 130 (6) June 2008.pp.30-33.

Fanuc Robotics (2008). http://www.robots.com/robots.php.

Hutton, D.V. (2004). *Fundamentals of Finite Element Analysis*, International Edition, McGraw Hill (Asia), Singapore, ISBN : 007-124160-4

Keramas, J. G. (1999). *Robot Technology Fundamentals*, Delmar Publishers, pp.24-25.

Lau, H. Y. K. ; Mak, K. L. & Lu, M. T. H. (2003). A Virtual Design Platform for Interactive Product Design and Visualization, *Journal of Materials Processing Technology* 139, 2003, pp. 402–407.

Mackerle, J. (1999). Finite Element Analysis and Simulation of Machining. *Journal of Materials Processing Technology* 86: 17-44.

Manseur, R. (2006). *Robot Modeling and Kinematics*, Da Vinci Engineering Press, Massachusetts, ISBN : 1-58450-851-5.

Rahman, M. (2004). Modeling and Measurement of Multi-axis Machine Tools to Improve Positioning Accuracy in a Software Way, Academic Dissertation presented with the assent of the Faculty of Technology, University of Oulu, for public discussion in Kuusamonsali (Auditorium YB210), Linnanmaa, June 4th, 2004.

Zlajpah, L. (2008). Simulation in Robotics, *Mathematics and Computers in Simulation* (2008), Available online at www.sciencedirect.com.

Statistical Imitation Learning in Sequential Object Manipulation Tasks

Komei Sugiura, Naoto Iwahashi, Hideki Kashioka
and Satoshi Nakamura
National Institute of Information and Communications Technology
Japan

1. Introduction

Imitation learning research explores various methods for teaching robots new motions by user-friendly means of interaction (Kuniyoshi et al., 1994) (Breazeal & Scassellati, 2002). Imitation learning has garnered considerable interest from robotics and artificial intelligence communities.

For robots aimed at household environments, motions such as "to put the dishes in the cupboard" are fundamental, but difficult to program beforehand. The reason is that the desired motion depends on the size and shape of the dishes, as well as those of the cupboard, and also on whether the cupboard has a door. Furthermore, the functional capability of natural communication with users is crucial for such assistive robots. However, it is difficult to map words or symbols to motions because of the same reason mentioned above. In (Krüger et al., 2007), the difficulties involved in mapping symbols to motions are discussed in detail.

There have been studies which try to solve the problems of mapping symbols to motions in the framework of imitation learning. A motion learning/generation method based on hidden Markov models (HMMs) is proposed in (Inamura et al., 2004). (Ogawara et al., 2002a) and (Ogawara et al., 2002b) present a method in which the relative trajectories between two objects are modeled by hidden Markov models (HMMs). Furthermore, we have proposed a motion learning and generation method that is based on reference-point-dependent HMMs, which enabled the learning of motions such as rotating an object, drawing a spiral, and placing a puppet on a box (Sugiura & Iwahashi, 2007) (Haoka & Iwahashi, 2000). In (Takano et al., 2007), mocap data is learned by using HMMs, and those HMMs are converted for the retrieval of sequential motions. A method based on recurrent neural networks is proposed in (Sugita & Tani, 2005). This method is extended to deal with sequential motions in (Ogata et al., 2007).

In this chapter, we present a novel method that generates and recognizes sequential motions for object manipulation such as placing an object on another ("place-on") and moving it away ("move-away"). In this method, motions are learned using reference-point-dependent probabilistic models, which are then transformed and combined. These composite probabilistic models are used for the recognition of sequential motions performed by a user.

Moreover, motions can be generated from the composite probabilistic models in accordance with user instructions, which can then be performed by a robot arm. Fig. 1 shows the hardware platform used in this study. The system has multimodal interfaces such as a stereo vision camera and a microphone.
The main advantage of mapping symbols to motions and combing them is as follows. The machine can first decompose the given task into learned motions. It can then present a planned motion as a sequence of symbols or words. The sequence is grounded on the motions taught by the user, and so the user can understand the meaning. This is a significant safety feature because if the machine can inform the user of the planned motions before executing them, the user can determine in advance whether they are safe or not.
The rest of this chapter is organized as follows. Section 2 first states the problem we try to solve and briefly reviews related work. It then describes the proposed method in Section 3. Section 4 shows the results of simulation experiments in which the proposed method generated motions by combining learned probabilistic models. The results of physical experiments are described in Section 5 in detail. Section 6 discusses problems and possible applications with the proposed method. Finally, Section 7 concludes the chapter.

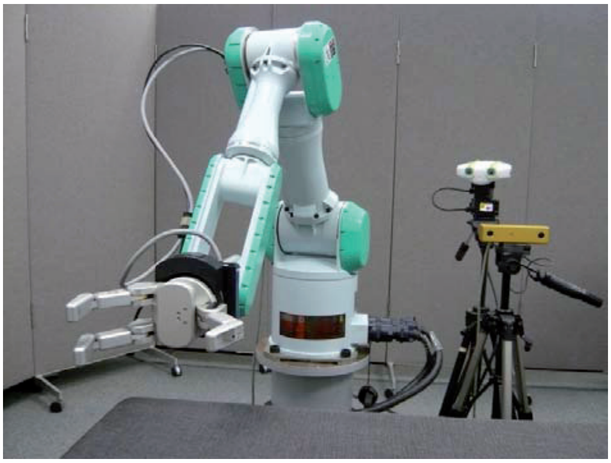


Fig. 1. Experimental platform used in this study.

2. Modeling Object-Manipulation

2.1 Reference-Point-Dependent Motions

Here, we consider the problem of learning reference-point-dependent motions in the framework of imitation learning (Kuniyoshi et al., 1994) (Breazeal & Scassellati, 2002) by a robot.
Clustering manipulation trajectories and mapping them to a verb are not sufficient for verb learning if the trajectories are considered only in the camera coordinate system. For simplicity, we assume that the mapping between the camera coordinate system and the world coordinate system is given, and that the user’s utterance is accurately recognized. Let us consider the example shown in the left-hand side image of Fig. 2. The figure depicts a camera image, in which a user is placing a green puppet on the box. The trajectory itself is

meaningless since it depends on the position of the box. In the case of Fig. 2, clustering manipulation trajectories in the camera coordinate system works only if the position of the does not change.

On the other hand, if we consider a coordinate system with its center at the box, we are able to cluster the trajectories in the coordinate system and map them to a verb. We call such a coordinate system as an *intrinsic coordinate system*. The origin of an intrinsic coordinate system is called the *reference point*. It is to be noted that the origin of the intrinsic coordinate system changes with the position of the box.

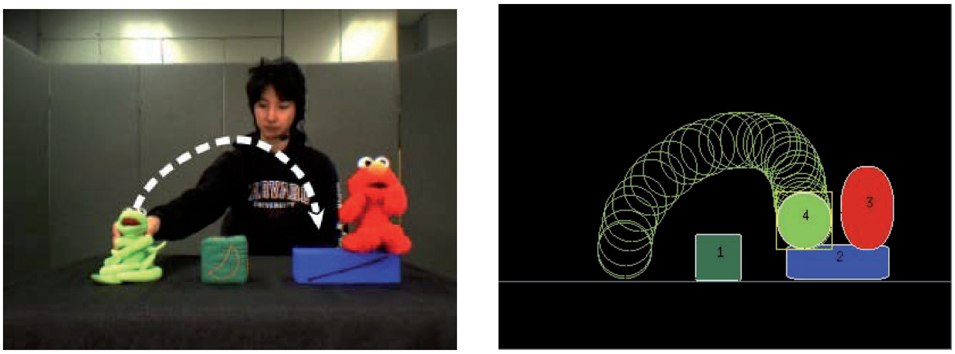


Fig. 2. Left: Example shot of camera images. The user is manipulating the green puppet. The dotted line represents the trajectory of manipulation. Right: Preprocessed visual features obtained from the image stream.

Regier investigated a model describing the spatial relationship between two objects (Regier, 1996). He proposed to model verbs as the time evolution of the spatial relationship between a trajector and a landmark. Here, a *trajector* is defined as a participant (object) that is focused on. A *landmark* has a secondary focus and a trajector is characterized with respect to a landmark. In cognitive linguistics, words representing spatial relationships such as "away" and "left of" are described as the relationship between a trajector and a landmark (Langacker, 1987). In (Ogawara et al., 2002b), the relative trajectories between two objects were modeled by using probabilistic models. The probabilistic models are used for the generation of manipulation trajectories.

In contrast, the proposed method estimates four components, which are necessary for learning object-manipulation verbs, from camera images. The components are as follows: (1) the trajector and landmark, (2) the reference point, (3) the intrinsic coordinate system, and (4) the parameters of the motion's probabilistic model. Fig. 3 shows examples, the verbs "raise" and "move-closer". We can reasonably assume that the reference point of "raise" is the trajector's center of gravity. The intrinsic coordinate system can be a Cartesian coordinate system as shown in the left-hand figure. In the case of "move-closer", another type of intrinsic coordinate system is necessary. In this case, the x-axis of the coordinate system passes through the centers of gravity of the trajector and the landmark. As explained in Section 4, we assume that there are several types of intrinsic coordinate systems.

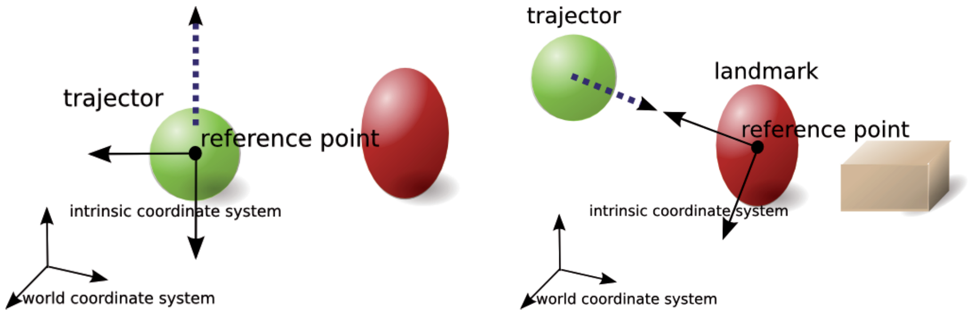


Fig. 3. Relationship between trajector/landmark, a reference point, and an intrinsic coordinate system. The spheres, ellipsoids, and box represent objects, and the arrows represent the axes of the intrinsic coordinate systems. Left: “raise”. The small sphere is the trajector, and the reference point is its center. The x-axis of the intrinsic coordinate system is horizontal. Right: “move-closer”. The direction of the x-axis is toward the trajector from the landmark.

2.2 Motion Learning by Reference-Point-Dependent Probabilistic Models

Consider that L training samples are given for a verb. Let V_l denote the lth training sample. V_l consists of the motion information of the trajector, ξ_l, and the candidate set of reference points, $\mathbf{R}_l$, as follows:

$$V_l = (\xi_l, \mathbf{R}_l), \tag{1}$$

$$\xi_l = \{\mathbf{y}_l(t) \mid t = 0, 1, \ldots, T_l\}, \tag{2}$$

$$\mathbf{y}_l(t) = \left[\mathbf{x}_l(t)^{\mathrm{T}}, \dot{\mathbf{x}}_l(t)^{\mathrm{T}}, \ddot{\mathbf{x}}_l(t)^{\mathrm{T}}\right]^{\mathrm{T}}, \tag{3}$$

$$\mathbf{R}_l = \{\mathbf{O}_l, \mathbf{x}_l(0), \mathbf{x}_{\text{center}}\} \equiv \left\{\mathbf{x}_{r_l} \mid r_l = 1, 2, \ldots, |\mathbf{R}_l|\right\} \tag{4}$$

where $\mathbf{x}_l(t)$, $\dot{\mathbf{x}}_l(t)$, and $\ddot{\mathbf{x}}_l(t)$ denote the position, velocity, and acceleration of the trajector, respectively; T_l denotes the duration of the trajectory; and $\mathbf{O}_l$ denotes the set of the static objects’ centers of gravity. The operator $|\bullet|$ represents the size of a set. The reason why $\mathbf{O}_l$ is included in $\mathbf{R}$ is that the static objects are candidate landmarks. We also include the first position of the trajector, $\mathbf{x}_l(0)$, in $\mathbf{R}$ so that we can describe a motion concept that is dependent only on the object's trajectory. Additionally, the center of the camera image, $\mathbf{x}_{\text{center}}$, is added to $\mathbf{R}$ to describe motion concepts that are independent of the positions of the objects.

We assume that there are K types of intrinsic coordinate systems, and these types are provided by the designer. We denote the type of the intrinsic coordinate system by k. k corresponds to a verb, and the reference point corresponds to each V_l. We obtain the estimated intrinsic coordinate system for the lth data from the estimation of k and the reference point $\mathbf{x}_{r_l}$.

Let ${}^{C_k(\mathbf{x}_{r_l})}Y_l$ denote the trajectory in the intrinsic coordinate system $C_k(\mathbf{x}_{r_l})$. Henceforth, parameters in a particular coordinate system are written in a similar manner. Now, the index series of reference points, $\mathbf{r} = \{r_l \mid l = 1, 2, \ldots, L\}$, the type of the intrinsic coordinate system, *k*, and the parameters of a probabilistic model regarding trajectories, λ , are searched for using the following maximum likelihood criterion:

$$\begin{aligned}\left(\hat{\mathbf{r}}, \hat{k}, \hat{\lambda}\right) &= \underset{\mathbf{r},k,\lambda}{\operatorname{argmax}} \sum_{l=1}^{L} \log P\left(Y_l \mid r_l, k, \lambda\right) \\ &= \underset{\mathbf{r},k,\lambda}{\operatorname{argmax}} \sum_{l=1}^{L} \log P\left({}^{C_k(x_{rl})}\xi_l ; \lambda\right)\end{aligned} \tag{5}$$

where $\hat{\bullet}$ represents estimation. In (Sugiura & Iwahashi, 2007) and (Haoka & Iwahashi, 2000), the solution to Equation (5) is explained in detail.

3. Combination of Reference-Point-Dependent HMMs

3.1 Transformation of HMMs

Here, we consider the problem of the recognition and generation of sequential motions based on composite reference-point-dependent HMMs.

In speech recognition, HMMs are transformed for speaker adaptation by using transformation matrices. Here, the transformation matrices are independent of the order of HMMs. However, we cannot combine two reference-point-dependent HMMs in this manner. This is because the *j*th HMM parameters are dependent on the (*j* - 1)th HMM parameters (Fig. 4).

Fig. 5 illustrates an example of the process of combining two reference-point-dependent HMMs. To combine HMMs corresponding to “raise” and “move-closer,” the output probability distributions of each HMM must be transformed since they represent the distributions on different coordinate systems.

An advantage of transforming intrinsic coordinate systems is the smoothness of the composite trajectories. In the proposed method, velocity and acceleration data as well as position data are used for learning. For safety reasons, changes in the velocity and acceleration data should be continuous. It is therefore important to obtain smooth trajectories of $\dot{\mathbf{x}}_l(t)$ and $\ddot{\mathbf{x}}_l(t)$ when combining two HMMs. Let us consider a case in which verbs dependent only on velocity information, e.g., “throw,” are to be combined. If two HMMs were simply aligned to generate the composite trajectory, the velocity changes might be discontinuous in this case. In contrast, the proposed method, which is described in detail below, generates a smooth trajectory.

Now we consider the problem of obtaining a composite HMM from the transformation and combination of reference-point-dependent HMMs. Let $\lambda^{\langle j \rangle}$ and $C^{\langle j \rangle}$ denote the parameters and the intrinsic coordinate system, respectively, of the *j*th HMM. Those HMMs are modeled as left-to-right HMMs. The output probability density function of each state is modeled by a single Gaussian. The mean position vector at state *s*, ${}^{C^{\langle j \rangle}}\boldsymbol{\mu}_x(s)$, is transformed by the following homogeneous transformation matrix:

$$\begin{bmatrix} {}^{W}\boldsymbol{\mu}_x(s) \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{W}_{C^{\langle j \rangle}}R & {}^{W}\boldsymbol{\mu}_x^{\langle j-1 \rangle}\left(S_{j-1}\right) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{C^{\langle j \rangle}}\boldsymbol{\mu}_x(s) - {}^{C^{\langle j \rangle}}\boldsymbol{\mu}_x(1) \\ 1 \end{bmatrix} \qquad \left(\mathrm{j} = 1,2,\ldots, \mathrm{D}, \mathrm{s} = 1,2,\ldots,\mathrm{S_j}\right), \tag{6}$$

where ${}^{W}_{C\langle j\rangle}R$ denotes the rotation matrix from $C^{\langle j\rangle}$ to the world coordinate system *W*. Furthermore, s = 0 and $s = S_j+1$ are defined as the initial and final states of the *j*th HMM, respectively. The mean vector of velocity, ${}^{C\langle j\rangle}\boldsymbol{\mu}_{\dot{x}}(s)$, and the mean vector of acceleration, ${}^{C\langle j\rangle}\boldsymbol{\mu}_{\ddot{x}}(s)$, are rotated as follows:

$$ {}^{W}\boldsymbol{\mu}_{\dot{x}}(s) = {}^{W}_{C\langle j\rangle}R\; {}^{C\langle j\rangle}\boldsymbol{\mu}_{\dot{x}}(s) \tag{7} $$

$$ {}^{W}\boldsymbol{\mu}_{\ddot{x}}(s) = {}^{W}_{C\langle j\rangle}R\; {}^{C\langle j\rangle}\boldsymbol{\mu}_{\ddot{x}}(s) \tag{8} $$

In contrast, the diagonal items of covariance matrices for position are approximated as follows:

$$ \operatorname{diag} {}^{W}\Sigma_x(s) = \operatorname{diag} {}^{C\langle j\rangle}\Sigma_x(s) \tag{9} $$

where ${}^{W}\Sigma_x(s)$ and ${}^{C\langle j\rangle}\Sigma_x(s)$ denote the covariance matrices at state s in coordinate systems *W* and $C^{\langle j\rangle}$, respectively. The non-diagonal items of the matrices are equal to zero. The matrices for velocity and acceleration are transformed by the same simple approximation. We do not perform a rotation of the covariance matrix because the HMM-based trajectory generation method (Tokuda et al., 1995) that we use does not deal with full covariance matrices.

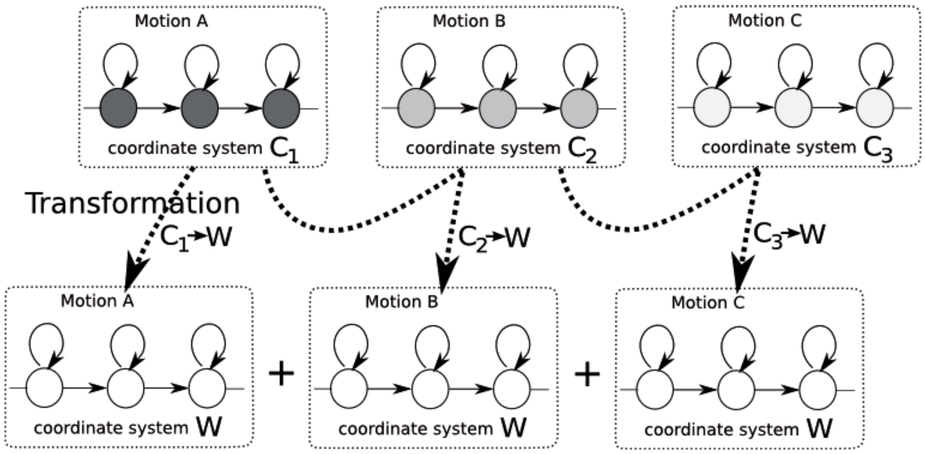


Fig. 4. Schematic of the combination of two reference-point-dependent HMMs. *W* represents the world coordinate system.

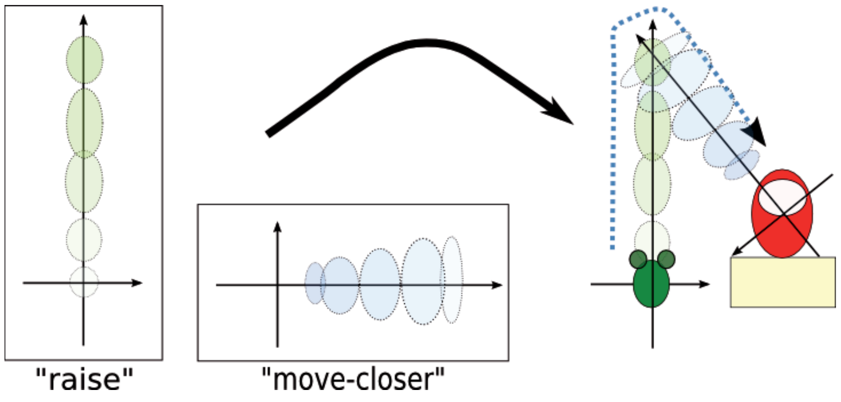


Fig. 5. Example of transformation in the combination of two HMMs, "raise" and "move-closer". Each dotted circle represents the variation of output probability distributions at each state of a left-to-right HMM. The direction of state transition is indicated by the color darkness. The intrinsic coordinate system of "move-closer" is transformed so that the its x-axis passes through both the landmark (the reference point of "move-closer") and the last position of the HMM regarding "raise". The dotted line represents the composite trajectory.

3.2 Generation of Motion Sequences by Composite HMMs

Here, we consider the problem of generating trajectories of sequential motions from composite HMMs. Suppose that a static image and the index of the trajector are given. As given in Section 2.2, we extract the candidate set of reference points, **R**. Our proposed method deals with two types of motion generation: (1) explicit instruction and (2) target instruction.

3.2.1 Explicit Instruction

The user requests the robot to move an object according to his/her instruction, which consists of a sequence of motions. Inputs from the user are the object ID and a sequence of verb-landmark pairs. The proposed method outputs the maximum likelihood trajectory that accomplishes the sequence.

Suppose a set of verbs, the intrinsic coordinate systems corresponding to verbs, and the HMM parameters corresponding to the verbs are given. Let $V = \{v_i \mid i = 1,2,\ldots,|V|\}$ denote a set of verbs, λ_i denote HMM parameters corresponding to verb v_i, and k_i denote the index of intrinsic coordinate system corresponding to verb v_i. Let $(\mathbf{i},\mathbf{r})$ denote a D-tuple of verb-landmark pairs as follows:

$$(\mathbf{i},\mathbf{r}) = \left(i^{\langle 1\rangle}, i^{\langle 2\rangle}, \ldots, i^{\langle D\rangle}, r^{\langle 1\rangle}, r^{\langle 2\rangle}, \ldots, r^{\langle D\rangle}\right) \tag{10}$$

The candidate set of reference points, **R**, can be obtained from an image stream (c.f. Section 2.2). The method explained in Section 3.1 provides a composite HMM $\Lambda_D(\mathbf{i},\mathbf{r})$.

The trajectory of Object r_{traj} corresponding to the verb-landmark pairs, (**i**, **r**), is obtained as follows:

$$\begin{aligned}\hat{\xi} &= \underset{\xi}{\operatorname{argmax}}\, P\left(\xi \mid r_{\mathrm{traj}}, Q_D(\mathbf{i}), \mathbf{r}, \mathbf{R}\right) \\ &= \underset{\xi}{\operatorname{argmax}}\, P\left(\xi \mid \mathbf{x}_{\mathrm{traj}}, Q_D(\mathbf{i}), \Lambda_D(\mathbf{i}, \mathbf{r})\right)\end{aligned} \tag{11}$$

where $Q_D(\mathbf{i})$ denotes the state sequence of the HMM corresponding to verb v_i, and $\mathbf{x}_{\mathrm{traj}}$ denotes the initial position of the trajector. The method explained in (Tokuda et al., 1995) provides the maximum likelihood trajectory from the unknown state sequence $Q_D(\mathbf{i})$.

3.2.2 Target Instruction

Next, we consider the problem of obtaining the optimal index sequence of verb-landmark pairs, $(\hat{\mathbf{i}}, \hat{\mathbf{r}})$, which affords the maximum likelihood trajectory $\hat{\xi}$ from initial position $\mathbf{x}_{\mathrm{traj}}$ to the goal position x_G. Most motion planning methods (see e.g., (Latombe, 1991)) do not provide linguistic expressions that explain generated trajectories. In contrast, the proposed method can generate trajectories consisting of learned motions labeled by verb indices. Therefore, the proposed method allows a robot to explain generated trajectories by natural language expressions if it has a speech interface. For example, the robot can generate confirmation utterances such as “The robot will put Object A on Object B, then raise Object A, is it OK?” Generating such confirmation utterances is desirable from the viewpoint of safety since the user can judge whether the motion is appropriate or not before the planned motion is performed.

In Target Instruction mode, the user requests that the robot move an object to a goal. Inputs from the user are the object ID and goal position x_G. The proposed method outputs the maximum likelihood sequence of verb-landmark pairs, $(\hat{\mathbf{i}}, \hat{\mathbf{r}})$, and the maximum likelihood trajectory $\hat{\xi}$. We obtain $(\hat{\xi}, \hat{\mathbf{i}}, \hat{\mathbf{r}})$ by conditioning the right side of Equation (11) with x_G and then adding $(\mathbf{i}, \mathbf{r})$ to the search arguments:

$$\left(\hat{\xi}, \hat{\mathbf{i}}, \hat{\mathbf{r}}\right) = \underset{\xi, \mathbf{i}, \mathbf{r}, D}{\operatorname{argmax}}\, P\left(\xi \mid \mathbf{x}_{\mathrm{traj}}, \mathbf{x}_G, Q_D(\mathbf{i}), \Lambda_D(\mathbf{i}, \mathbf{r})\right),$$

where the number of combined HMMs, D, is a search depth parameter.

3.3 Recognition of Motion Sequences by Composite HMMs

The recognition of sequential motions by reference-point-dependent HMMs can be formalized as the problem for obtaining the maximum likelihood probabilistic model for trajectory ξ under the condition where a lexicon of verbs $L_v = \{v_i, \lambda_i, k_i \mid i = 1, 2, \ldots, |V|\}$ is given. The maximum likelihood index sequence of the verb-landmark pairs, $(\hat{\mathbf{i}}, \hat{\mathbf{r}})$, is searched through the following equation:

$$
\begin{aligned}
(\hat{\mathbf{i}}, \hat{\mathbf{r}}) &= \operatorname*{argmax}_{\mathbf{i},\mathbf{r},D} P(\xi \mid \mathbf{i}, \mathbf{r}, D, \mathbf{R}) \\
&= \operatorname*{argmax}_{\mathbf{i},\mathbf{r},D} P(\xi \mid \Lambda_D(\mathbf{i}, \mathbf{r}))
\end{aligned}
$$

4. Simulation Experiments

4.1 Experimental Setup

We first conducted simulation experiments for evaluating the proposed method. The simulator consists of a graphical interface and a mouse. Virtual objects shown on the screen can be manipulated by dragging them with a mouse.

In the learning phase, a user was asked to teach motions. The trajectories were recorded and used for training probabilistic models. The trajectories for the following seven verbs were collected.

raise, move-closer, move-away, rotate, place-on, put-down, jump-over

For each verb, the number of training samples was 15. Those motions were taught by the user in the learning phase beforehand, and they were constant throughout the motion generation experiments.

The verbs were successfully learned in the experiment. Fig. 6 shows the examples of the training samples. In this figure, the thick arrows represent the trajectories of an object manipulated by the user. The thin arrows represent the x- and y-axes of the estimated type of the intrinsic coordinate system. The type name is shown at the lower right of each illustration. The types of the intrinsic coordinate system are defined as follows:

C_1: A coordinate system with its origin at the landmark position. C_1 is a translated camera coordinate system. The x-axis is inverted in case the x-coordinate of the original position of the trajector is negative after translation.

C_2: An orthogonal coordinate system with its origin at the landmark position. The direction of the x-axis is from the landmark towards the trajector.

C_3: A translated camera coordinate system with its origin at the original position of the trajector.

C_4: A translated camera coordinate system with its origin at the center of the image.

After probabilistic models were trained, we carried out motion generation simulation experiments. Two types of motion experiments were performed: for target instruction and for explicit instruction.

In the target instruction experiment, the user requested the robot to move an object in a camera image. The inputs were object ID and a goal point, and the outputs from the robot were both a sequence of verb-landmark pairs and a trajectory to the goal. We used 64 grid points as goal points. The search depth parameter D was set as D = 3. Therefore, the estimated motion sequence consisted of up to three HMMs.

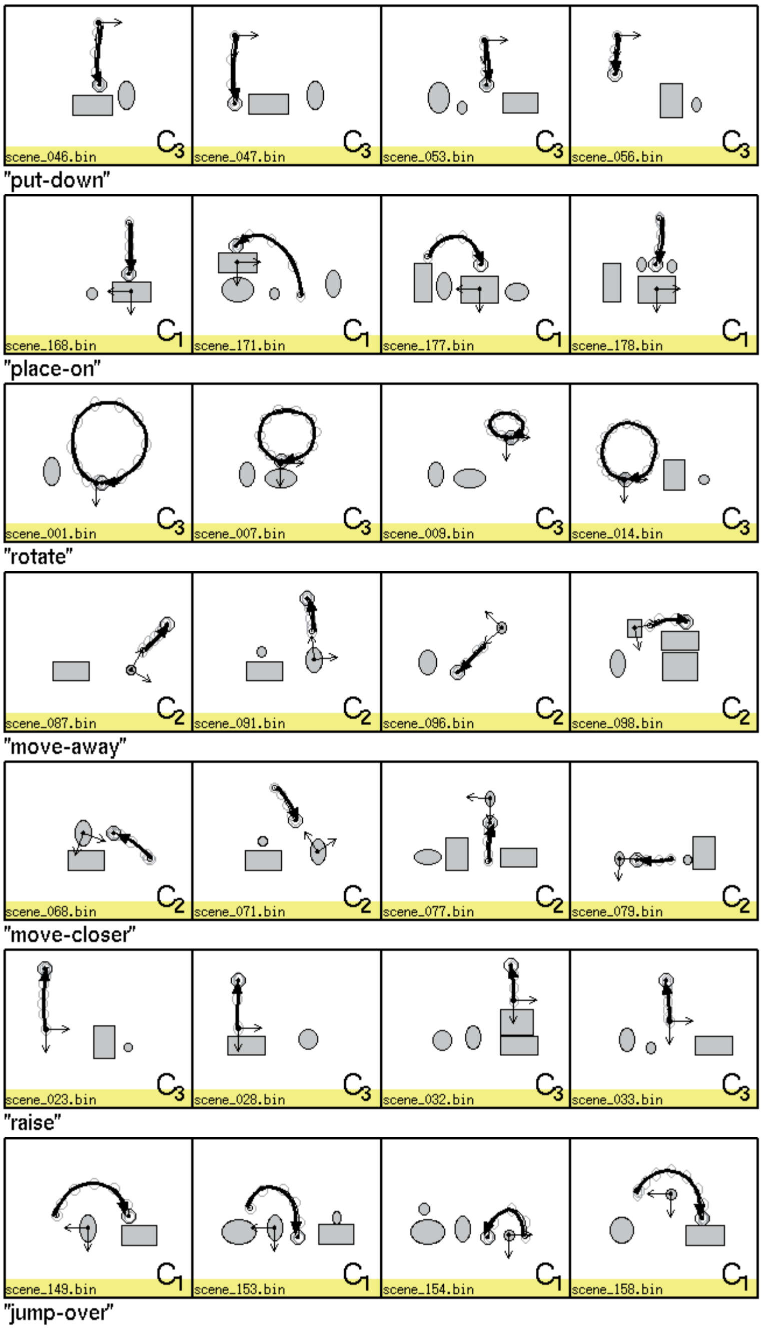


Fig. 6 Examples of training samples.

4.2 Result (1): Motion Generation

4.2.1 Explicit Instruction

The explicit instruction experiment was carried out in the same simulation environment. Fig. 7 shows two examples of motion generation: "jump object 1 over object 2, then put object 1 down, and move object 1 closer to object 4" and "jump object 2 over object 1, jump object 2 over object 1 again, and then place object 2 on object 5". The inputs for the two cases were as follows:

	Trajector	Motion sequence
(a)	Object 1	<jump-over, 2><put-down, no landmark><move-closer, 4>
(b)	Object 2	<jump-over, 1><jump-over, 1> <place-on, 4>

Fig. 7 shows that the three motions were combined smoothly. To examine this result quantitatively, we plotted the evolution of position, velocity, and acceleration in case (a) in Fig. 8. Fig. 8 verifies that the composite trajectory of velocity and that of acceleration were continuous.

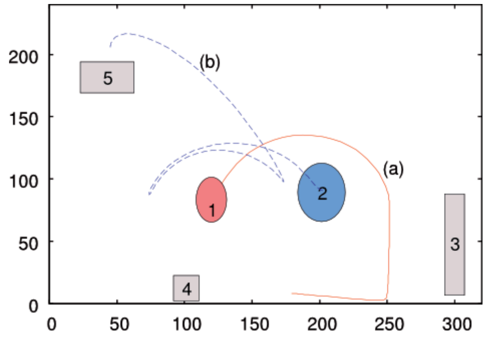


Fig. 7. Explicit instruction

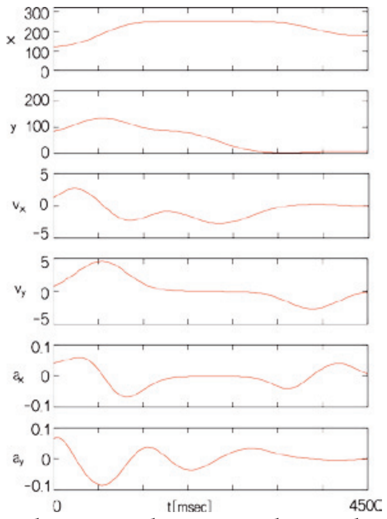


Fig. 8. Evolution of position, velocity, acceleration under condition (a) in Fig. 7.

4.2.2 Target Instruction

The simulation environment is shown in Fig. 9. There were five objects in the environment (depicted by numbered boxes and circles). Object 1 was used as the trajector. Fig. 9 shows the examples of maximum likelihood trajectories output by the proposed method. In the figure, bracketed pairs represent the estimated sequences of verb-landmark pairs. For example, <place-on, 2><move-closer, 5> signifies "place object 1 on object 2, and move object 1 closer to object 5."

From Fig. 9, we can see that the proposed method combined two motions smoothly rather than independently. In particular, although <move-away, 2> and <move away, 2><jump-over, 4> share <move-away, 2>, the trajectories do not overlap with each other. This is probably due to the large variation for the last position in the learned probabilistic model of "move-away." In other words, a part of the trajectory of <move-away, 2> was curved to smoothly combine <move away, 2> and <jump-over, 4>, but the likelihood of the resultant trajectory was still high because of its large variance.

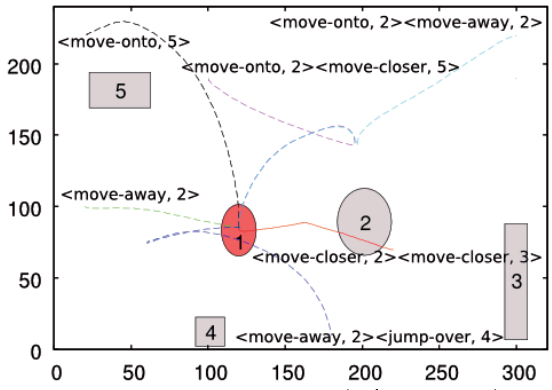


Fig. 9. Generated motions in Target Instruction mode for manipulating object 1. Numbered boxes and circles represent objects. Bracketed pairs represent the estimated sequence of verb-landmark pairs. Specifically, <place-on, 2><move-closer, 5> means "place object 1 on object 2, and move object 1 closer to object 5".

5. Physical Experiments

5.1 Experimental Setup

The experiments were conducted by using a PA-10 manipulator manufactured by Mitsubishi Heavy Industries with seven degrees of freedom (DOFs). The manipulator was equipped with a BarrettHand, a four-DOF multifingered grasper. The user's movements were recorded by a Bumblebee 2 stereo vision camera at a rate of 30 [frame/s]. The size of each camera image was 320 × 240 pixels. The left-hand side image of Fig. 2 shows an example shot of an image stream, and the right-hand side image of the figure shows the internal representation of the image stream. All the motion data used for learning and recognition were obtained from physical devices. In addition, motion generation results were examined in an environment using the manipulator and physical objects such as puppets and toys.

The difference of parameter setup between the simulation and physical experiments was the number of training samples, *L*. In physical experiments, *L* was set to 9 for each motion.

5.2 Motion Generation

5.2.1 Explicit Instruction

Fig. 10 shows an example trajectory generated by the proposed method. The solid line represents the trajectory generated in the explicit instruction mode. The input for the explicit instruction mode was as follows:

Trajector	Motion sequence
Object 2	<move-away, 1><jump-over, 4><move-closer, 4>

From **Fig. 10**, we can see that the proposed method generated an appropriate trajectory. To support this, the manipulator is shown performing the generated trajectory, as shown in **Fig. 11**.

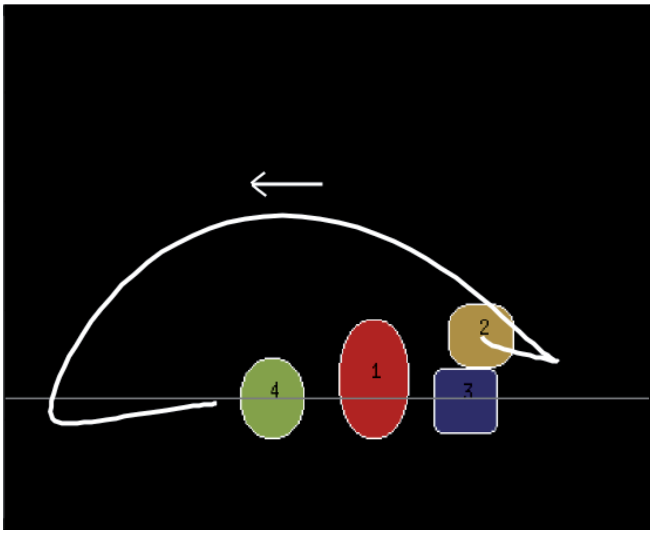


Fig. 10. Generated trajectory in the explicit instruction mode.

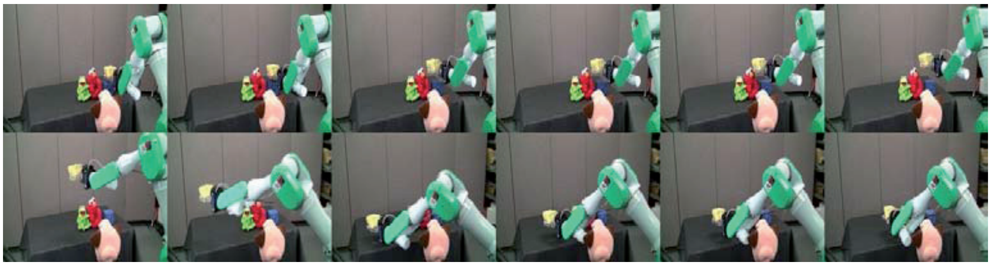
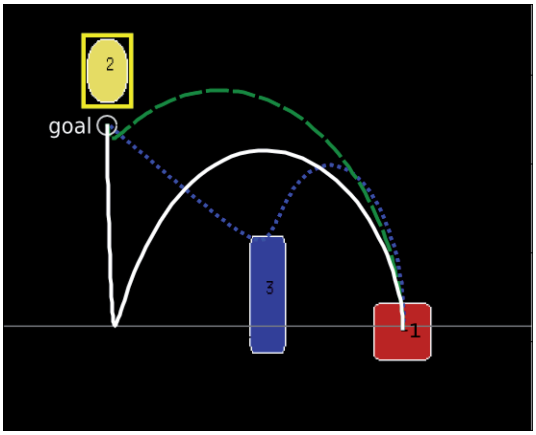


Fig. 11. Sequential photographs of the manipulator executing the trajectory shown in Fig. 10.

5.2.2 Target Instruction

For the target instruction mode, the top three trajectories are shown in **Fig. 12**. The trajector ID was set to 1, and the goal position used is indicated in the figure. In the figure, the solid, broken, and dotted lines represent the best, second-best, and third-best trajectories, respectively. In addition, the top three verb-landmark pairs and the log likelihood are shown in the figure.



1.	(solid line)	<jump-over, 3> <move-closer, 2>	-16.45
2.	(broken line)	<jump-over, 3>	-18.66
3.	(dotted line)	<place-on, 3> <move-closer, 2>	-25.06

Fig. 12. Generated trajectories in the target instruction mode.

5.3 Motion Recognition

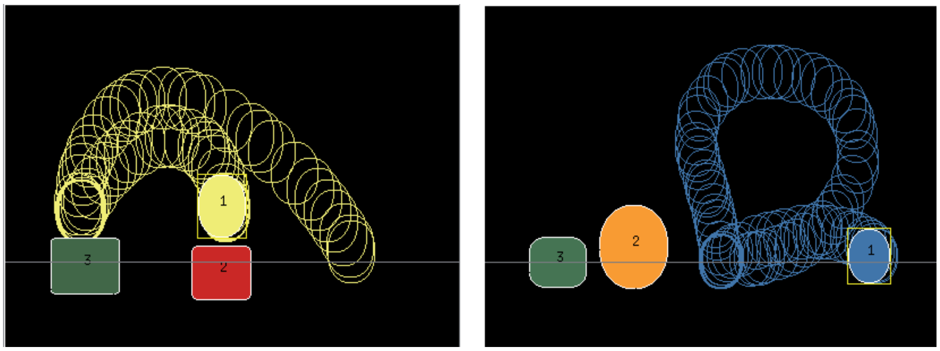
The user was presented with six pairs of randomly chosen verbs, and performed the motions sequentially. The manipulation trajectories and the positions of the static objects were recorded to obtain a test set. For each pair, five different object settings were given. Therefore, the size of the test set was 30.

Fig. 13 illustrates example trajectories in the test set. In the figures, the top three recognition results for each scene are shown. The bracketed pairs and numbers represent the estimated sequences of verb-landmark pairs and the log likelihood, respectively.

We can see that a correct recognition result was obtained for the left-hand figure of Fig. 13. On the other hand, the correct recognition result for the right-hand figure does not have the maximum likelihood. This is considered to be due to the fact that the trajectory in the training set always starts from pause ($\dot{\mathbf{x}}_l(0)=\mathbf{0}$), and therefore, the composite HMMs contain states representing pauses between motions. However, the two motions are consecutively performed. Therefore, the likelihood of such trajectory given the combined HMMs was smaller than the likelihood of the trajectory given an HMM.

Table 1 shows the number of correctly recognized samples. The column labeled "*n*-best" stands for the number of correct answers contained in the top *n* recognition results. The accuracy of 1-best, 2-best, and 3-best recognition results are 63%, 83%, and 87%, respectively.

In the table, we obtain an accuracy of 80% (12/15) for sequences (1), (3), and (4). This is reasonable since we have obtained an accuracy of 90% for the recognition of single motions in preliminary experiments. However, we obtain an accuracy of 47% (7/15) for sequences (2), (5), and (6), which contains at least one C_2 verb. This result also supports the fact that the approximation (Equation (9)) deteriorated the recognition accuracy.



1. <place-on, 3> <place-on, 2> :	-22.04	1. <move-away, 2> :	-22.18
2. <jump-over, 2> <place-on, 2> :	-23.79	2. <rotate> <move-away, 2> :	-22.65
3. <place-on, 3> <move-away, 3> :	-28.79	3. <rotate> :	-25.06

Fig. 13. Examples of test set. The recognition results for each scene are shown below the corresponding figure. Left: "place Object 1 on Object 3, then place Object 1 on Object 2." Right: "rotate Object 1, and then move Object 1 away from Object 2."

Test set	1-best	2-best	3-best
(1) rotate + rotate	5	5	5
(2) move-away + move-closer	3	3	3
(3) place-on + place-on	3	5	5
(4) rotate + jump-over	4	4	4
(5) rotate + move-away	2	4	4
(6) move-closer + place-on	2	4	5
Total	19/30 (63%)	25/30 (83%)	26/30 (87%)

Table 1. Number of correctly recognized samples.

5. Discussion

5.1 Recognition Accuracy

Here, we discuss two causes for the deterioration in the recognition accuracy.
The first one is that sequential motions performed by users tend to be smoothly combined. However, the likelihood for such motions is not always high. This is because the trajectory

in the training set always starts from pause ($\dot{\mathbf{x}}_l(0)=\mathbf{0}$), and therefore, the composite HMMs contain states representing pauses between motions.
We assume that this problem can be solved by using pause HMMs. In this case, a sequence of HMMs comprising a motion HMM sandwiched between pause HMMs are trained. Furthermore, we can obtain a composite HMM by aligning the HMMs of pause, motion A, motion B, and pause, and thereby combining two motions.
Another problem is that Equation (9) does not consider the full covariance matrices, and so the rotation of coordinate systems is ignored. As stated above, this approximation deteriorated the recognition accuracy for the C_2 verbs. In the future work, we will perform the rotation of covariance matrices.

5.2 Collision Avoidance
The proposed method has a possibility of generating inappropriate trajectories leading to object collision. For instance, Fig. 9 shows that the trajectory of <move-closer, 2><move-closer, 3> runs through Object 2.
There are at least three solutions to this problem. The first is to select the maximum likelihood sequence of verb-landmark pairs among which no collision occurs. This is the simplest solution since the positions of all static objects are evident in camera images. Another solution is to slightly change the maximum trajectory so as to avoid collisions. The third one is to modify the output probability density functions of HMMs by setting them to 0 near the position of obstacles. The third method, however, will not work if there are many obstacles in the camera image.

5.3 Future Work
The proposed method can be applied for motion planning rather than manipulating objects. One example is a mobile robot that generates a path to a goal set by the user by combining learned motions and informs the user about it. Suppose a camera is placed on the ceiling of an office and a camera image, as shown in Fig. 9, is obtained. In this case, the robot can decompose an instruction such as “go to the president’s room” into learned motions such as “move-forward” and “move-along.”
Another application would be the prediction of motions. In this chapter, partial trajectories were not considered. However, if the motion recognition is performed with a part of the trajectory, the system can predict the landmark of the motion and help the user. For example, this could be used in a technique to unlock a door before the user grasps the door knob.

6. Conclusion

It is important for robots to be able to report their internal states to humans in a comprehensive manner in environments shared by both. For example, it is critical for the safety of people that the robots are able to communicate their next move.
In this chapter, we have presented a method to combine reference-point-dependent probabilistic models. The experimental results of the Target Instruction mode revealed that the proposed method successfully decomposed goal-oriented motions into learned motions. This indicates that the robot could decompose the given task into learned motions and then present the planned motions, in a manner easy for the user to understand. This is a

significant safety feature because if the machine can inform the user of the planned motions before executing them, the user can decide in advance whether they are safe or not.
Furthermore, the proposed method enables the recognition of sequential motions. One of the contributions of this work is the recognition of sequential object manipulation. In the future work, the proposed method would be applied to problems for the retrieval of motions from video data and action mining in ubiquitous computing.

7. References

Breazeal, C. & Scassellati, B. (2002). Robots that imitate humans. *Trends in Cognitive Science*, Vol. 6, No. 11, pp. 481–487.

Haoka, T. & Iwahashi, N. (2000). Learning of the reference-point-dependent concepts on movement for language acquisition, *Proceedings of Pattern Recognition and Media Understanding*, Vol. 2000-105, pp. 39–46.

Inamura, T.; Toshima, I.; Tanie, H. & Nakamura, Y. (2004). Embodied symbol emergence based on mimesis theory, *International Journal of Robotics Research*, Vol. 23, No. 4, pp. 363–377.

Krüger V.; Kragic, D.; Ude, A. & Geib, C. (2007). The meaning of action: a review on action recognition and mapping, *Advanced Robotics*, Vol. 21, No. 13, pp. 1473–1501.

Kuniyoshi, Y.; Inaba, M. & Inoue, H. (1994). Learning by watching: extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation, Vol.* 10, No. 6, pp. 799–822.

Langacker, R.W. (1987). *Foundations of Cognitive Grammar: Theoretical Prerequisites*, Stanford University Press.

Latombe, J.C. (1991). *Robot motion planning (Kluwer International Series in Engineering and Computer Science, 124), S*pringer.

Ogata, T.; Murase, M.; Tani, J.; Komatani, K. & Okuno, H.G. (2007). Two-way translation of compound sentences and arm motions by recurrent neural networks. *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1858–1863.

Ogawara, K.; Takamatsu, J.; Kimura, H. & Ikeuchi, K. (2002a). Generation of a task model by integrating multiple observations of human demonstrations. *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*,

Ogawara, K.; Takamatsu, J.; Kimura, H. & Ikeuchi, K. (2002b). Modeling manipulation interactions by hidden Markov models, *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robos and Systems*, pp. 1096-1101.

Regier, T. (1996). *The Human Semantic Potential: Spatial Language and Constrained* Connectionism (Bradford Books), The MIT Press.

Sugita, Y. & Tani, J. (2005). Learning combinatoriality from the interaction between linguistic and behavioral processes, *Adaptive Behavior*, Vol. 13, No. 1, pp. 33-52.

Sugiura, K. & Iwahashi, N. (2007). Learning object-manipulation verbs for human-robot interaction, *Proceedings of the 2007 International Workshop on Multimodal Interfaces in Semantic Interaction*, pp. 32-38.

Takano, W.; Yamane, K. & Nakamura, Y. (2007). Capture database through symbolization, recognition, and generation of motion patterns, *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pp. 3092-3097.

Tokuda, K.; Kobayashi, T. & Imai, S. (1995). Speech parameter generation from HMM using dynamic features. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 660–663.

Tangible interfaces for tangible robots

Andrew Cyrus Smith
CSIR's Meraka Institute
South Africa

1. Introduction

Various modes of tangible interfaces have been explored and researched. In this chapter we limit our look at tangible user interfaces to a subset of these. The subset is characterised by portability and no attached tethers, be they mechanical links or electrical wires. The subset does include tangible objects that are connected to a larger system for the purpose of relative position and orientation detection, if relevant. Such detection mechanisms include optical, magnetic, and radio means. Examples of optical detection are the use of fibre optics and a video camera. Magnetic detection utilises either the presence of a magnetic field, or the changes in such a field. Radio detection mechanisms include the use of the Global Positioning System (GPS) and radio frequency identification (RFID). Using electrically conductive pins provides for another untethered system.
Electrical field sensing and the use of acoustic waves are also covered in this chapter.
In our discussion we assume open-loop control of robot manipulators, that is, the user interface does not receive feedback from sensing subsystems. The user interface relies on other subsystems to check the inputs provided by the user interface with the actual position of the manipulator.

2. A Short Introduction to Tangible User Interfaces

It this section we introduce the novice to this exiting mode of interfacing to technologies. We look at the properties of known Tangible User Interfaces (TUI's) and how they have been applied in the real-world.
What are Tangible User Interfaces (TUI's)? The term TUI has been coined by Ullmer and Ishii in 1997 (Ullmer, 1997). This definition is somewhat restrictive in that the output is also reflected in the input device. An example of such a device is Tobopo. Tobogo is a physical device that will record the actions the user has taken on its various components. For example, if the user constructs a model dog and moves the various legs, the system will record the motions and replay them. It is quite possible to let the system modify the behaviour after being recorded, or show a response even during the recording.
In this chapter we look at a broader definition of TUI, similar to the relaxation of TUI's by others (Fisken, 2004). In the definition we address cubic objects that provide an input to

some system. The output is not manifested in the cubes as per the strict definition of TUI's. As applied to robot effectors, this implies that the output is visible through the change in the effectors' state. For the purpose of this chapter we prefer the broad script of TUI's as given in Fishkin2004. In this broad script we are concerned with an "input event", some system that "senses" the event and somehow responds to it, and some form of feedback initiated by the system which is called an "output event".
We base our interaction roles on those described elsewhere (Yanco, 2004). In the taxonomy of Yanco, five interaction roles are given. These are "a supervisory role", "an operator", "a teammate", "a mechanic", and "a bystander". The tangible interfaces we describe are best suited in the role of a supervisor. The supervisor constructs the series of actions that should be executed by the robot and then activates the programme represented by the cubes. The underlying system does not simply execute a number of steps, but has the ability to change the execution sequence based on inputs received from the actuators, the environment, or another system (Fig. 1.).
Some three dimensional TUI's are manipulated in a two-dimensional plane. Other TUI's have been developed that also work in three dimensional spaces, such as Tobopo, ActiveCubes, and SystemBlocks. Tobopo can be used as an autonomous system. ActiveCubes are used to sense and interface with other systems.

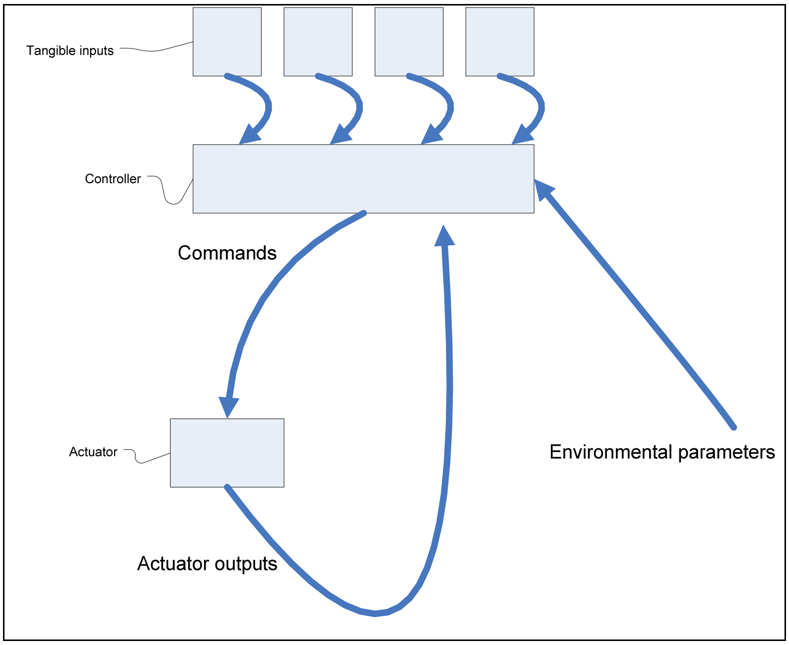


Fig. 1. Generic tangible system diagram

3. Why TUI and not GUI?

Ever since the electronic computer became a research tool the operator had to take care of the delicate input mechanisms available to interact with the computer. At first the

mechanisms available were switches and paper tape. These were followed by magnetic tape and paper punch cards. At this time output mechanisms evolved from paper tape and lights, to the two-dimensional cathode ray tube (CRT) display. Yet these output mechanisms are still two-dimensional. The information displayed on these displays has progressed from only textual to the incorporation of graphical elements. Over time the Graphical User Interface (GUI) has become familiar to all computer users. Yet some users still insist that the textual interface suits them the best. They claim that they are the most productive with such an interface. At the same time these users also make extensive use of the QWERTY keyboard to interact with the computer. They are professional computer system developers and make little use of the computer mouse, claiming that the keyboard shortcuts they are accustomed to empowers them more than using a mouse and the GUI. For the majority of computer users the GUI and mouse remains the most prominent interface to the electronic computer.
There exists, however, a relatively new research field in which the manipulation of physical artefacts are considered as an alternative interface to the electronic computer. It can be argued that making use of tangible interaction with the computer, in addition to the GUI, increases the 'bandwidth' available to a user for interacting with the computer. An increased bandwidth allows for faster interaction. The use of gross motor skills, in addition to the fine motor skills required for operating a computer mouse, might be more 'natural' for some users.

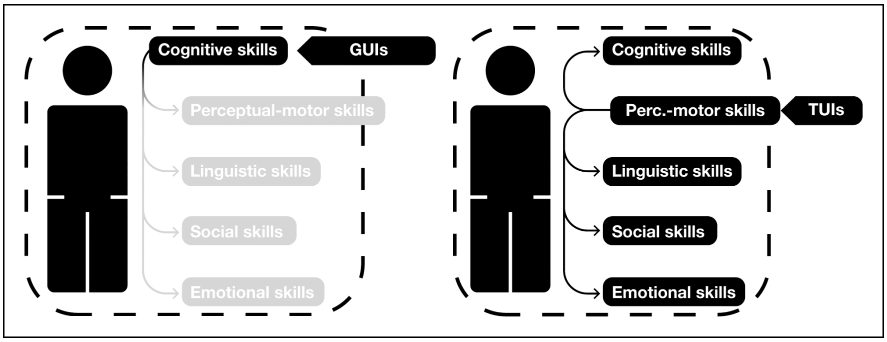


Fig. 2. Graphical User Interfaces address the user's cognitive skills. Tangible User Interfaces also incorporates the user's motor skills (Hegeveld 2009)

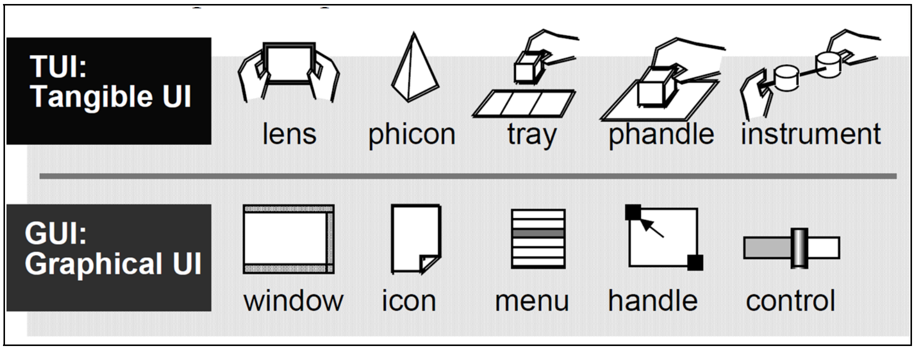


Fig. 3. TUI instantiations of GUI elements (Ullmer 1997)

In the physical world in which we use Tangible User Interfaces (TUI's), we can find similarities between the TUI's and the GUI's by extrapolating the two dimensional screen to the three dimensional physical world.

4. Limitations and Advantages of TUI's

TUI's have certain advantages and limitations compared to other technology interfaces. These advantaged and limitations are discussed in this section.

4.1 Limitations

Tangible User Interfaces have a number of disadvantages over conventional Graphical User Interfaces. These include storage of the constructed sequence, the space required for the sequence, how to make it persistent (as one would save a file to a hard disk), how to document it and transporting the constructed sequence.

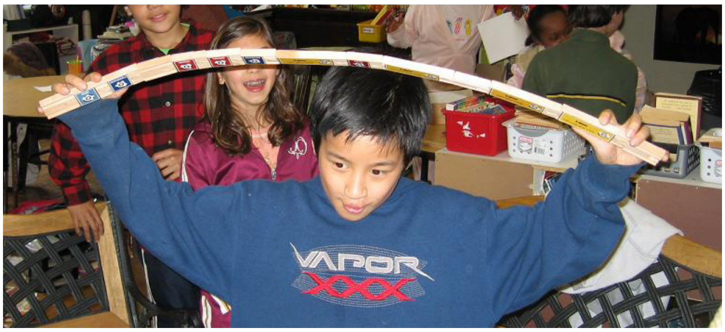


Fig. 4. Transporting TUI sequences can be difficult (Horn 2009)

4.2 Advantages

Tangible User Interfaces can potentially be designed to be intuitive for the novice user (Fig. 5.), but potentially frustrating for an advanced user. A textual interface or an iconic interface could be presented to the advanced user as a possible solution.

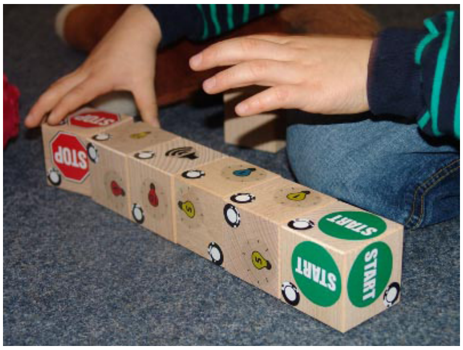


Fig. 5. Tangicons (Scharf et al., 2008)

5. Data Coupling Mechanisms

5.1 Magnetic

Magnetic detection is possible using either mechanical switches or solid state detection circuitry. Figures 6 and 7 illustrate a system called GameBlocks which makes use of mechanical "reed" switches.

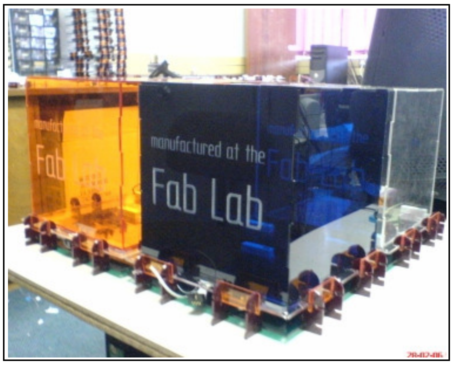


Fig. 6. GameBlocks (Smith 2007)

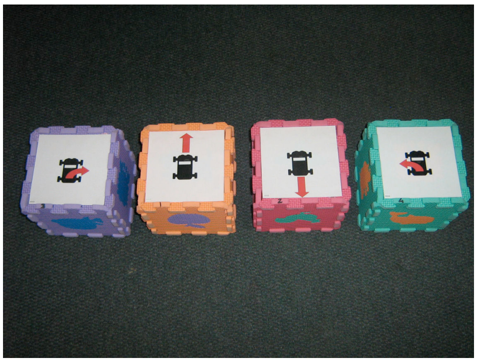


Fig. 7. GameBlocks (Smith 2009)

5.2 Electrical contact

Electrical contacts rely on direct physical contact between two or more electrically conductive components. A few examples follow.

AlgoBlocks

AlgoBlocks makes use of wide electrical connectors to distribute the data through the system.

Fig. 8. AlgoBlocks (Suzuki 1995)

FlowBlocks

FlowBlocks distributes data using the same magnets that are used to keep the various components together.

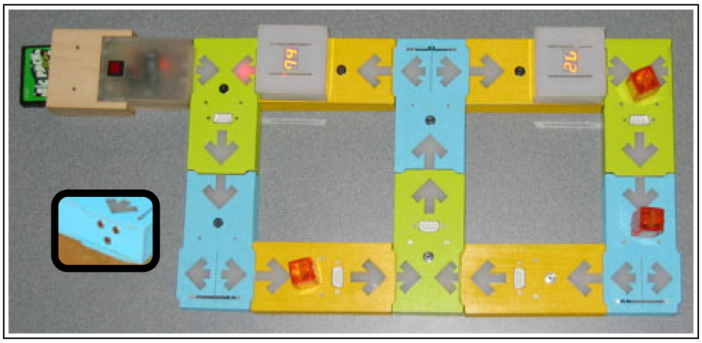


Fig. 9. Flowblocks are connected using magnets. The same magnets are also used to transfer data and power between the blocks. The insert shows three magnets at the end of one of the blocks. Magnets assist is aligning the blocks properly (Zuckerman 2005)

VIO controls

VIO controls make use of pins which consist of two parts each. One part runs along the inside of the other and is slightly longer than the outer sleeve. The longer length allows penetration to a second conductive layer which is located below the upper conductive layer. The sleeve makes contact with the upper layer only.

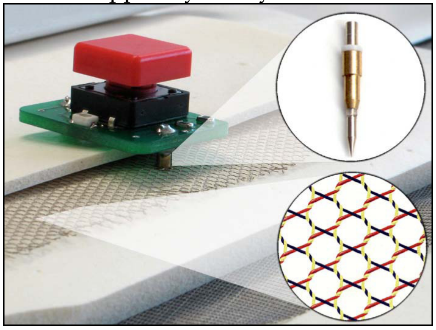


Fig. 10. VIO controls (Villar and Gellersen)

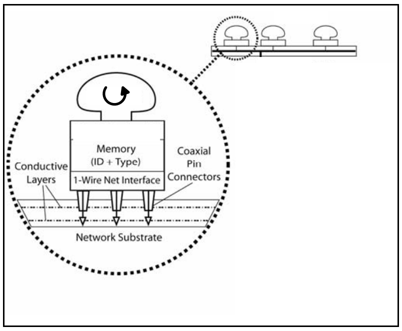


Fig. 11. The electrical configuration of VIO controls. (Adapted from Villar and Gellersen)

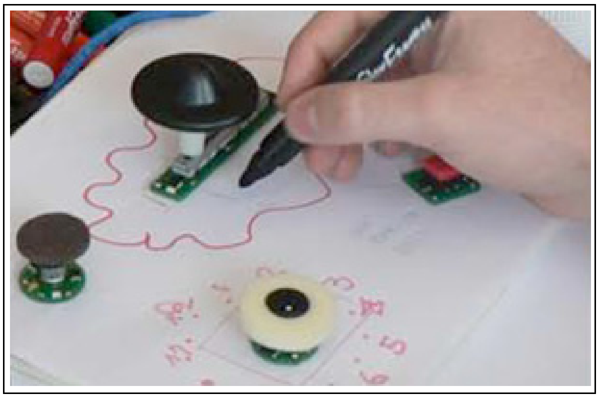


Fig. 12. An example of the VIO controls being applied

5.3 Optical: video camera from below
This approach makes use of bottom projection (Fig. 13.) with the video camera placed below the work surface. A configuration like this is convenient as it eliminates obscuration of both the projection and video recordings (Fig. 14.).

5.4 Optical: video camera from above
In the previous section an example in given of fiducial markers (Fig. 15.) placed at the bottom of the object being tracked. Another configuration is with the fiducial markers placed on top of the object to be tracked (Fig. 16.). Optional images are also projected from above the interaction surface.

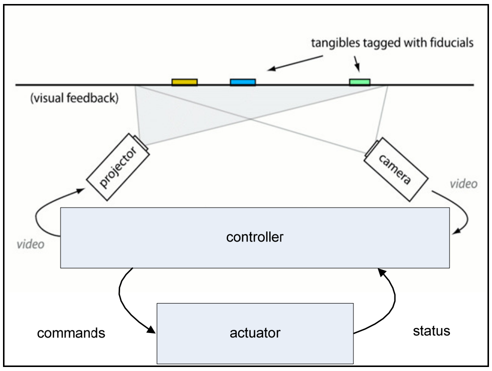


Fig. 13. Using tangibles tagged with fiducials to control and actuator. Visual feedback is provided by the projection below the transparent work surface (Adapted from Kaltenbrunner and Bencina)

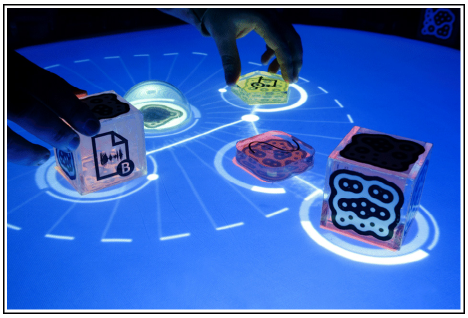


Fig. 14. Tangibles with fiducials and bottom projection are used to control a music synthesiser in a system called "reacTable"

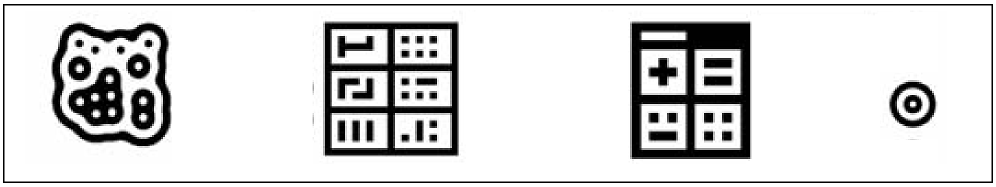


Fig. 15. Examples of fiducial marker types. The fiducial on the left is very compact. (Adapted from Kaltenbrunner and Bencina)

When using Illuminating Light (Fig. 17.), a software programme identifies the coloured dots and their patterns on the optical elements. A projector then adds additional information onto the work surface, such as the path of reflected light.
Tern (Fig. 18.) consists of a collection of interlocking pieces. Each piece has a unique optical pattern imprinted on the top which identifies the function of that piece.

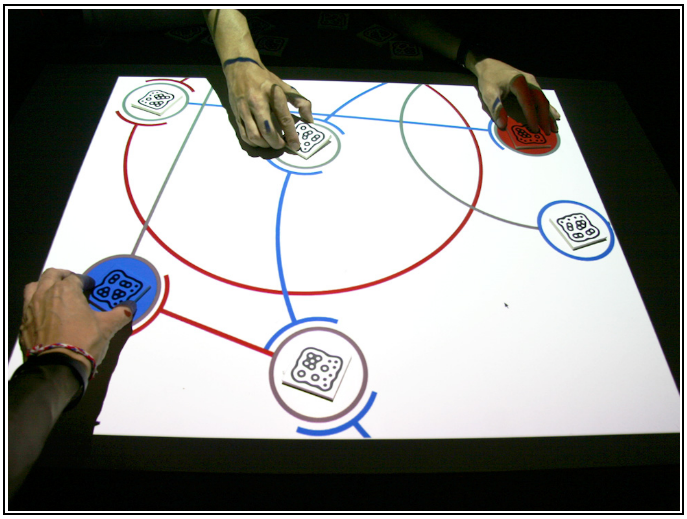


Fig. 16. Top camera and top projection (Kirton 2008)

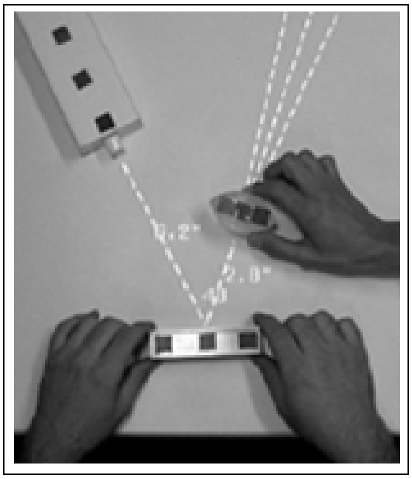


Fig. 17. Illuminating Light (Underkoffler 1999)

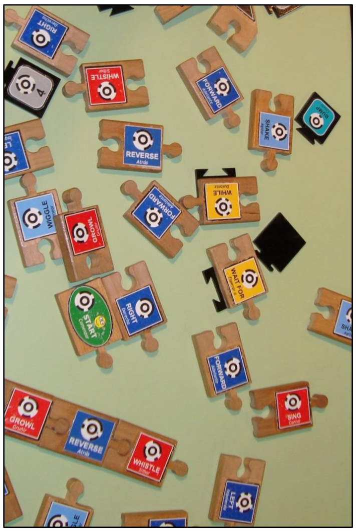


Fig. 18. This tangible interface consists of wooden blocks shaped like jigsaw puzzle pieces (Horn2009)

5.5 Optical: one dimensional

In contrast to the two-dimensional video camera communication mechanism described earlier in this chapter, we here present two examples of TUI systems that make use of a single light source that communicates between two blocks (Fig. 19, 20.).

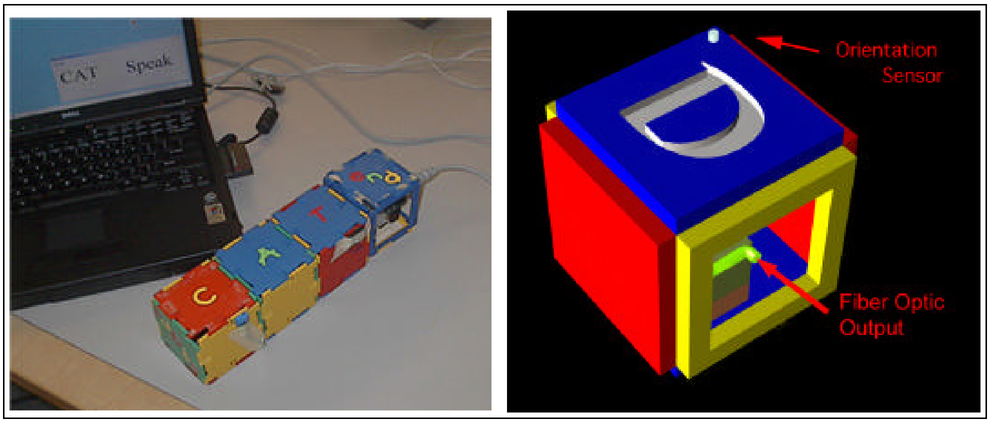


Fig. 19. Computational alphabet block (Eisenberg 2002)

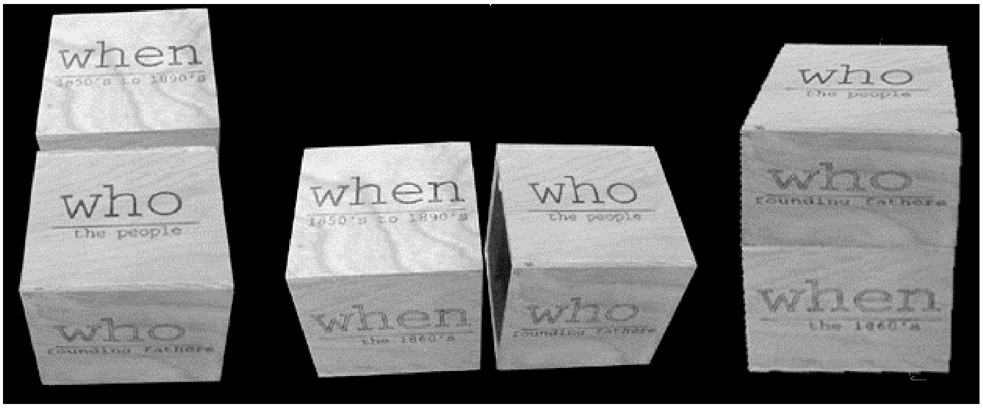


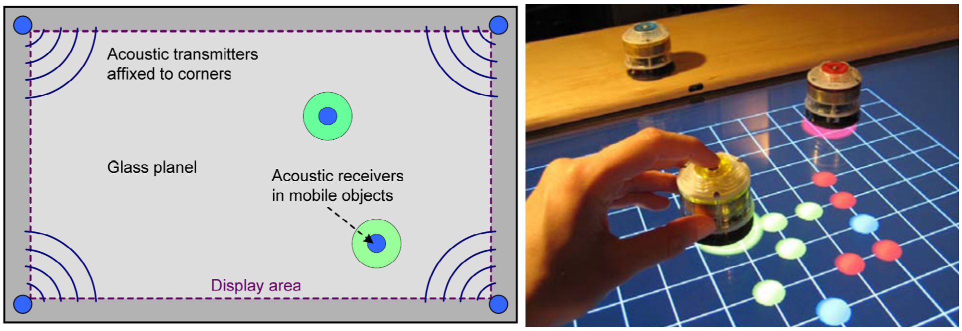
Fig. 20. Using the Navigation Blocks to construct disjunctions and negations. Left: "or" query. Middle: "and" query. Right: "not" query (Camarata 2002)

5.6 Acoustic sensing

The acoustic table (Fig. 21.) consists of a number of acoustic transmitters which are used to 'illuminate' the surface of the interaction area. The objects to be detected contain circuitry that responds to the "illumination" by transmitting infrared signals to a set of infrared detectors around the table.

5.7 Induction sensing

Induction sensing systems (Fig. 22.) make use of low frequency alternating current flowing through a wire grid. The objects to be sensed contain their own inductive and capacitive circuits which resonates at a pre-determined frequency. If the sensing surface is stimulated at the same frequency at which the object to be sensed has been tuned, the object will be detected.


Acoustic transmitters affixed to corners
Glass planel
Acoustic receivers in mobile objects
Display area
Fig. 21. Acoustic table (Mazalek)

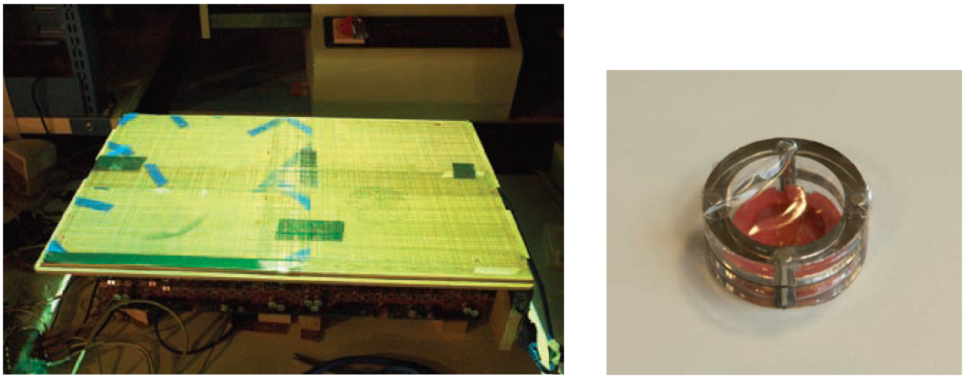


Fig. 22. Several modified sensing antennas and LC tag (Patten 2005)

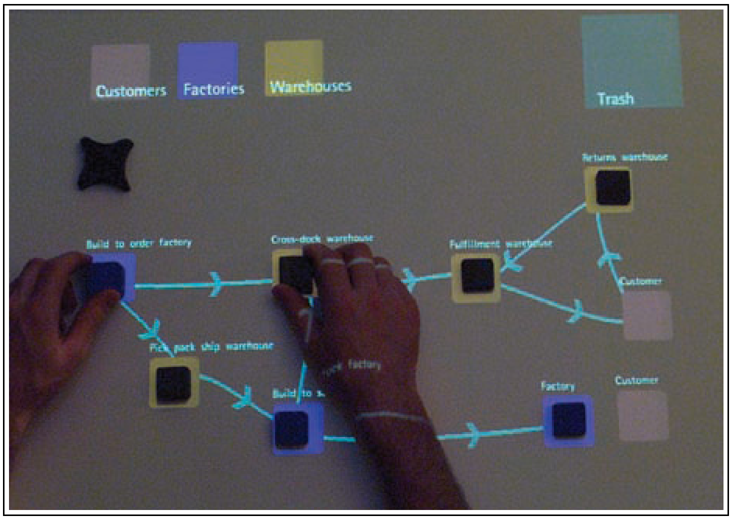


Fig. 23. Resonant table in use with overhead projection (Patten 2005)

6. Other TUI's

We have not covered the multitude of possible sensing and construction mechanisms. In this section we simply provide a few more interesting examples.

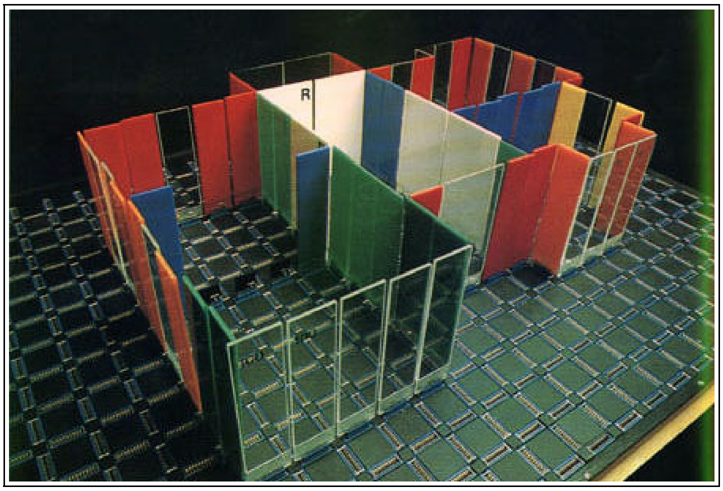


Fig. 24. Grid-restricted tangibles (Frazer 1995)

The following TUI's are not restricted to a surface for assembly. They operate independently of a surface and can be manipulated in the hand of the user while in operation.

6.1 Tobopo

Topobo (Fig. 25.) consists of a number of building elements, most of which contain electrical circuits.
Some elements serve as sensors, others as actuators. What makes Tobopo unique is that some building elements contain both a sensing circuitry and actuators. As an example, if the user rotates the shaft of a motor element, the Tobopo system can record that action and on command 'replay' the action.

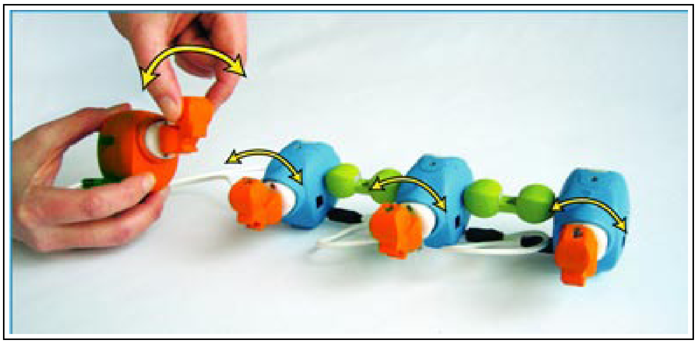


Fig. 25. Tobopo programming and replay (Raffle 2008)

6.2 SystemBlocks

SystemsBlocks (Fig. 26.) consists of a number of objects with embedded electronic circuitry. These augmented objects are interconnected using electrical wires through which data flows.

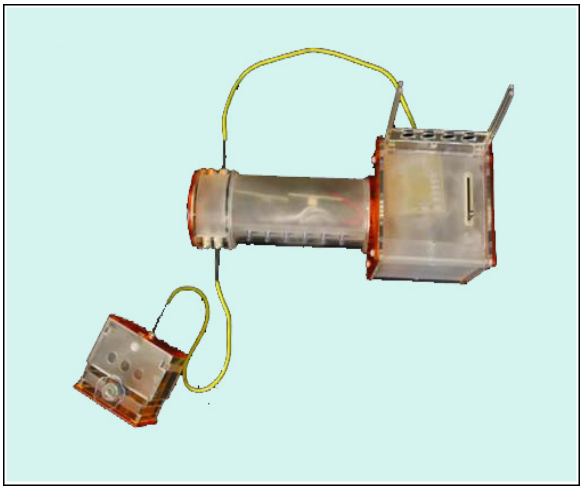


Fig. 26. SystemBlocks are interconnected using electrical wires (Zuckerman 2004)

6.3 ActiveCube

ActiveCube (Fig. 27.) is a system comprising of various cubes, each containing an electric circuit specific to the cube's intended function. The cubes are custom designed to serve as either sensors or actuators. Examples of sensor cubes are a sound processor, an infrared sensor, a gyroscopic sensor, a tactile sensor, and an ultrasonic sensor. Examples of actuator cubes are a motor, a buzzer, a vibrator, and a light. Activecubes are snapped together using the four clothing fasteners on each of the six cube surfaces. These fasteners are also used for transferring data between the cubes.

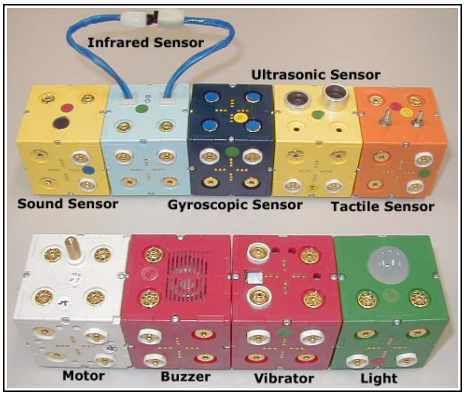


Fig. 27. ActiveCube (WATANABE 2004)

7. Tangible Interfaces Research at the CSIR's Meraka Institute

In the research described in the above sections, little or no consideration has been given to the costs involved in creating the tangible interfaces. A different approach is followed at the CSIR's Meraka Institute.

The Institute is located in the developing region of Southern Africa where access to funding is limited, perhaps more so than in developed regions where the research covered above is taking place. As a result the cost of technology is considered a very important system component. To achieve the objective of affordable Tangible Interfaces for developing regions, the Institute explores various materials and technologies. In all its research to date, the Institute has made use of low cost electronic components for interfacing the tangible objects to toy robotic devices (Smith, 2008).

The approach followed at the Institute, which distinguishes it from the others mentioned, is that of leveraging communal knowledge and the use of low-cost technologies. To this end, one of the research objectives is to develop a modular system in which various community members can collaborate in the co-creation of a robotic system using tangible interfaces. When realised, one team member will assemble a simple, low cost electronic circuit. In turn, another team member will design and craft Tangible Interface objects. The electronic circuit and the crafted object will then be integrated to form a Tangible Interface. This Tangible Interface can then be manipulated by the end user. The purpose of the electronic circuit is to sense the position and orientation of the tangible object and then send commands to a robot. Examples of robots used in the research include humanoid robots and LEGO cars (Fig. 28.).

7.1 Technology

The sensing mechanism is common to all the prototypes described in this section. In these prototypes the sensing of a Tangible Interface object is accomplished through a combination of low-cost reed switches and permanent magnets. A number of reed switches are mounted on a sensing platform and magnets are embedded inside Tangible Interface objects. When a Tangible Interface object is placed on top of the sensing surface, a pre-determined combination of reed switches close. At the same time an electronic circuit senses the state of the reed switches and sends appropriate instructions to a robot for execution.

7.2 Prototypes

Cubic - and rotational Tangible Interfaces prototypes are described in the following sections.

Cubic Interfaces

Initial research at the Meraka Institute made use of acrylic sheets. These were cut according to a profile which allows assembly into a cube without the need for adhesives (Fig. 29.). Low power laser facilities at a local FabLab (Gershenfeld, 2005) were of immense value in completing this task (Smith, 2006). As a side it can be noted that FabLab is a concept which originated at the MIT Media Lab with the objective of making advanced prototyping technologies available to communities in developing regions.

The second prototype was constructed from commercially available closed-cell foam squares. These squares are manufactured in large quantities for use in baby and toddler rooms (Fig. 30.). The bright colours and soft texture afforded by these foam squares are ideal for young users of the Tangible Interfaces (Smith 2009a).

Both the acrylic- and foam cube- designs make use of a sensor matrix to detect the tangible object. This cubic configuration has been dubbed "GameBlocks".

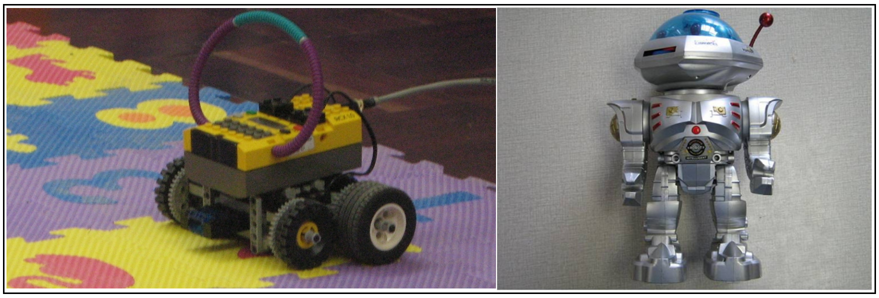


Fig. 28. Two toy robots used in the Tangible Interfaces research

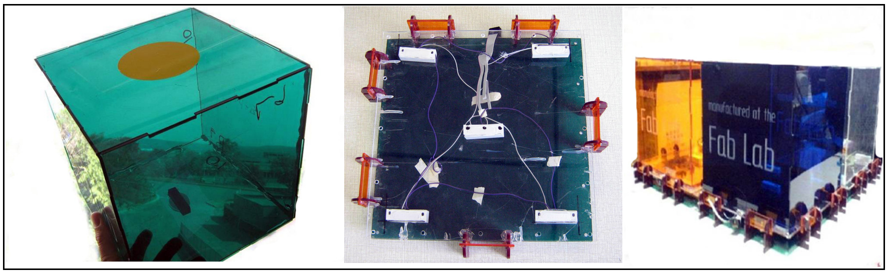


Fig. 29. The acrylic GameBlocks consists of cubes (left) and sensing trays (center)

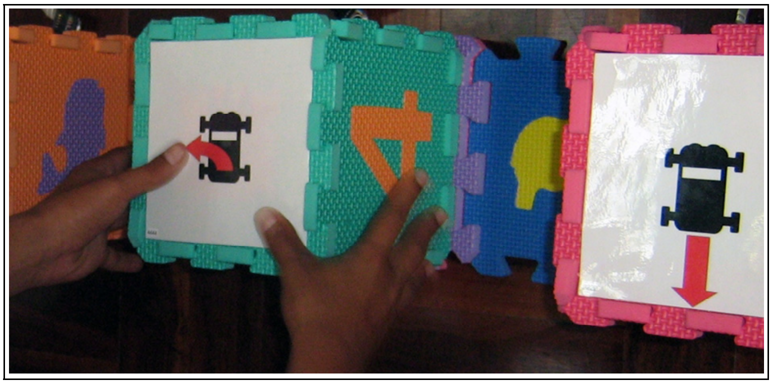


Fig. 30. Closed-cell foam GameBlocks

Rotational interfaces

A different sensor configuration was tested in the third and fourth prototype designs. In this configuration all tangible objects are identical in both shape and function, the difference being the spatial orientation of the tangible being manipulated. By changing the configuration of sensors inside the sensing surface as well as that of the embedded magnets inside the tangible, a configuration for sensing rotation was realised.

In the third prototype the properties of soft rock was explored. Using hand tools, the end user can easily shape the soft rock to create a personalized tangible (Fig. 31.). This prototype has been dubbed "RockBlocks" (Smith, 2009b).
"Dialando" is, similar to RockBlocks, a tangible interface based on rotational information. This fourth prototype demonstrates the use of recycled materials in its construction. A tangible object is constructed by sandwiching low cost magnets between two discarded CD/DVDs and finishing the construction off with a section of discarded electrical cord.

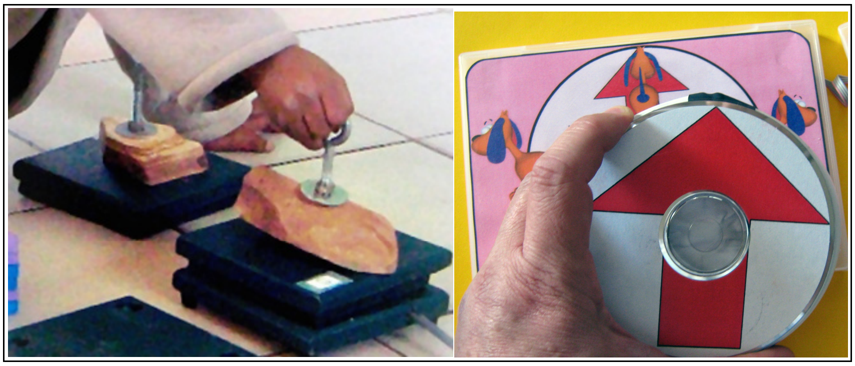


Fig. 31. RockBlocks and Dialando

7.3 Problems and solutions

A common problem experienced in various degrees is that of aligning the tangible object with the sensing surface. If the alignment is slightly out, the tangible object will either not be sensed or will be sensed incorrectly. The fourth prototype described above is an attempt to address this problem.
In an effort to reduce alignment problems a neodymium magnet was incorporated at the centre of the tangible object (Fig. 31.). A matching magnet was also positioned in the center of the sensing surface. Being of opposite polarity, the two magnets pull the tangible object into place when approaching the sensing surface, thus eliminating most of the misalignment problems experienced in the other designs.

7.4 Future work

In the design of the Dialando prototype, most of the alignment problems have been addressed through the addition of magnet-pairs. What still needs addressing is how to limit rotation of the tangible object to discrete angles. It is anticipated that this can be accomplished using a similar mechanism as that implemented for solving the misalignment between the sensing surface and the tangible object.

8. References

Camarata, K.; Do, E. Y.; Johnson, B. R. & Gross, M. D. (2002). Navigational Blocks: Navigating Information Space with Tangible Media, *IUI'02*, 2002

Eisenberg, M.; Eisenberg, E.; & Gross, M. (2002). Khomkrit Kaowthumrong, Nathaniel Lee, and Will Lovett , Computationally-Enhanced Construction Kits for Children: Prototype and Principles, *ICLS 2002*, 2002

Fishkin, K. P. (2004). A taxonomy for and analysis of tangible interfaces, *Personal Ubiquitous Comput.*, 8, 347-358, 2004, Springer-Verlag

Frazer J. H. (1995). *An Evolutionary Architecture*, 1995, Architectural Association

Gershenfeld, N.(2005); *FAB: the coming revolution on your desktop – from personal computers to personal fabrication*, Basic Books, ISBN 046-5-027466

Hengeveld, B.; Hummels, C. & Overbeeke, K. (2009). Tangibles for Toddlers Learning Language, *Proceedings of the Third International Conference on Tangible and Embedded Interaction*, 2009, ACM

Horn, M. S. (2009). *Tangible Computer Programming: Exploring the Use of Emerging Technology in Classrooms and Science Museums*, PhD thesis, 2009, Tufts University

Ishii, H. (1997). *Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms*, 1997.

Kaltenbrunner, M. & Bencina, R. (2007). reacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction, *Proceedings of the 1st international conference on Tangible and embedded interaction*, 2007, ACM

Kirton, T.; Ogawa, H.; Sommerer, C. & Mignonneau, L. (2008). PINS: A Prototype Model Towards the Definition of Surface Games, *Proceeding of the 16th ACM international conference on Multimedia*, 2008, ACM

Mazalek, A. (2005). *Media Tables: An extensible method for developing multi-user media interaction platforms for shared spaces*, PhD dissertation, 2005, MIT

Patten, J. M. (2005). *Mechanical Constraints as Common Ground between People and Computers*, PhD dissertation, 2005, MIT

Raffle, H. S. (2008). *Sculpting Behavior: A tangible language for hands-on play and learning*, PhD dissertation, 2008, MIT

Suzuki, H. & Kato, H. (1995). Interaction-level support for collaborative learning: AlgoBlock—an open programming language, *Proceedings Computer support for collaborative learning*, pp. 349-355, 1995, Lawrence Erlbaum Associates, Inc.

Smith, A. C.; (2006). Tangible Cubes as Programming Objects, *Artificial Reality and Telexistence: 16th international conference*, pp.157-161, Hangzhou, November 2006, IEEE Computer Society.

Smith, A. C. (2007). Using Magnets in Physical Blocks That Behave As Programming Objects, *Proceedings of the Conference on Tangible and Embedded Interaction*, 2007, ACM

Smith, A. C.; (2008). A Low-cost, Low-energy Tangible Programming System for Computer Illiterates in Developing Regions, *Proceedings of TEDC: Technology for Innovation and Education in Developing Countries*, ISBN 978-0-620-43087-6, Kampala, August 2008, Smith, Pretoria

Smith, A.C.; (2009a). *Symbols* for Children's Tangible Programming Cubes: an Explorative Study, *Southern African Computer Lecturers' Association Conference*, ISBN 978-1-60558-683-0, South Africa, pp. 105-109, June 2009, ACM.

Smith, A.C.; (2009b). Hand-Crafted Programming Objects and Visual Perception, *IST-Africa 2009 Conference*, ISBN 978-1-905824-11-3, Kampala, May 2009, IIMC International Information Management Corporation.

Spiessl, W.; Villar, N.; Gellersen, H. & Schmidt, A. (2007). VoodooFlash: Authoring across Physical and Digital Form, 2007, ACM Ullmer, B. & Ishii, H. (1997). The metaDESK: Models and Prototypes for Tangible User Interfaces, *Proceedings of UIST '97*, 1997, ACM

Underkoffler, J. & Ishii, H. (1998). Illuminating light: an optical design tool with a luminoustangible interface CHI '98: *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 542-549, 1998, ACM Press/Addison-Wesley Publishing Co.

Underkoffler, J. & Ishii, H. (1999). Illuminating Light: A Casual Optics Workbench, *CHI 99*, 1999, ACM

Villar, N. & Gellersen, H. (2007). A Malleable Control Structure for Soft wired User Interfaces, *Proceedings of the 1st international conference on Tangible and embedded interaction*, 2007, ACM

Watanabe, R.; Itoh, Y.; Asai, M.; Kitamura, Y. & Kishino, F. (2004). Osaka University, and Hideo Kikuchi, The Soul of ActiveCube — Implementing a Flexible, Multimodal, Three-Dimensional Spatial Tangible Interface, *Computers in Entertainment*, Vol. 2, No. 4, Article 6b, 2004, ACM

Yanco, H. A. & Drury, J. (2004). Classifying Human-Robot Interaction: An Updated Taxonomy, *Conference on Systems, Man and Cybernetics*, 2004, IEEE

Zuckerman, O. (2004). *System Blocks: Learning about Systems Concepts through Hands-on Modeling and Simulation*, MSc dissertation, 2004, MIT

Zuckerman, O.; Arida, S. & Resnick, M. (2005). Extending Tangible Interfaces or Education: Digital Montessori-inspired Manipulatives, *CHI 2005*, 2005, ACM

Timoshenko Beam Theory based Dynamic Modeling of Lightweight Flexible Link Robotic Manipulators

Malik Loudini
École Nationale Supérieure d'Informatique
Algeria

1. Introduction

In recent years, establishing more and more explicit, complete and accurate dynamic models for the special category of flexible link manipulators has been a formidable challenging and still open problem in robotics research.

This chapter is devoted to a methodological presentation of the application of Timoshenko beam (TB) theory (TBT) concepts to the mathematical description of flexible link robotic manipulators dynamics, as a more refined modeling approach compared to the classical Euler-Bernoulli (EB) theory (EBT) which is the conventionally adopted one.

Compared with the conventional heavy and bulky rigid robots, the flexible link manipulators have their special potential advantages of larger work volume, higher operation speed, greater payload-to-manipulator weight ratio, lower energy consumption, better manoeuvrability and better transportability. However, their utilization incurs a penalty due to elastic deformation and vibration typically associated with the structural flexibility. As a consequence, the motion planning and dynamics modeling of this class of robotic manipulators are apparently made extremely complicated, as well as their tip position control.

The complexity of modeling and control of lightweight flexible manipulators is widely reported in the literature. Detailed discussions can be found in (Kanoh et al., 1986; Baruh & Taikonda, 1989; Book, 1990; Yurkovich, 1992; Book, 1993; Junkins & Kim, 1993; Canudas de Wit et al., 1996; Moallem et al., 2000; Benosman et al., 2002; Robinett et al., 2002; Wang & Gao, 2003; Benosman & Vey, 2004; Dwivedy & Eberhard, 2006, Tokhi & Azad, 2008).

In order to fully exploit the potential advantages offered by these lightweight robot arms, one must explicitly consider the effects of structural link flexibility and properly deal with (active and/or passive) control of vibrational behavior. In this context, it is highly desirable to have an explicit, complete and accurate dynamic model at disposal.

In this chapter, we aim to present the details of our investigations concerned with deriving accurate equations of motion of a flexible link robot arm by the use of the TBT.

In the first part of this work, a brief review of different beam theories and especially that of Timoshenko is given. Then, based on the TBT, the emphasis is essentially set on a detailed description of the different steps, allowing the obtaining of accurate and complete governing equations of the transversely vibrating motion of an actuated lightweight flexible link robot arm carrying a payload at its free end-point. To display the most relevant aspects of structural properties inherent to the modeled deformable link studied as a beam, important damping mechanisms often ignored, internal structural viscoelasticity (Kelvin-Voigt damping) and external viscous air damping, are included in addition to the TBT effects of cross section shear deformation and rotational inertia.

In the other part of this chapter, an illustrative application case of the above presentation is rigorously highlighted. A new comprehensive dynamic model of a planar single link flexible manipulator considered as a shear deformable TB with internal structural viscoelasticity is proposed. On the basis of the combined Lagrangian-Assumed Modes Method with specific accurate boundary conditions (BCs), the full development details leading to the establishment of a closed form dynamic model, suitable for control purposes, are given.

2. Timoshenko Beam Theory Based Mathematical Modeling

2.1 Brief review of beam theories

A rigorous mathematical model widely used for describing the transverse vibration of beams is based on the TBT (or thick beam theory) (Timoshenko, 1974) developed by Timoshenko in the 1920s. This partial differential equation (PDE) based model is chosen because it is more accurate in predicting the beam's response than the EB beam (EBB) theory (EBBT) (Meirovitch, 1986) one (Aldraihem, 1997; Geist & McLaughlin, 2001; Stephen, 2006; Salarieh & Ghorashi, 2006). Indeed, it has been shown in the literature that the predictions of the TB model are in excellent agreement with the results obtained from the exact elasticity equations and experimental results (Trail-Nash & Collar, 1953; Huang, 1961; Stephen, 1982; Han et al., 1999; Stephen, 2006).

Historically, the first important beam model was the one based on the EBT thin or classical beam theory as a result of the works of the Bernoulli's (Jacob and Daniel) and Euler. This model, established in 1744, includes the strain energy due to the bending and the kinetic energy due to the lateral displacement of the beam. In 1877, Lord Rayleigh improved it by including the effect of rotary inertia in the equations describing the flexural and longitudinal vibrations of beams by showing the importance of this correction especially at high frequency frequencies (Rayleigh, 2003). In 1921 and 1922, Timoshenko proposed another improvement by adding the effect of shear deformation (Timoshenko, 1921; Timoshenko, 1922). He showed, through the example of a simply-supported beam, that the correction due to shear is four times more important than that due to rotary inertia and that the EB and Rayleigh beam equations are special cases of his new result. As a summary, four beam models can be pointed out (Table 1), the EBB and TB models being the most widely used.

As seen above, the TBT accounts for both the effect of rotary inertia and shear deformation, which are neglected when applied to EBBT. The transverse vibration of the beam depends on its geometrical and material properties as well as the external applied torque. The geometrical properties refer mainly to its length ℓ, size and shape of its cross-section such as its area A, moment of inertia I with respect to the central axis of bending, and Timoshenko's shear coefficient k which is a modifying factor ($k < 1$) to account for the distribution of shearing stress such that effective shear area is equal to kA. The material properties refer to its density in mass per unit volume ρ, Young's modulus or modulus of elasticity E and shear modulus or modulus of rigidity G.

Effect **Beam model**	*Lateral displacement*	*Bending moment*	*Rotary inertia*	*Shear deformation*
Euler-Bernoulli	+	+	–	–
Rayleigh	+	+	+	–
Shear	+	+	–	+
Timoshenko	+	+	+	+

Table 1. The four beam models with the corresponding effects

2.2 The flexible robotic system: Definitions and variables

The flexible manipulator physical system under consideration is shown in Fig. 1. It consists of a pinned-free or a clamped-free with tip payload (see Fig. 2) planar moving flexible arm which can bend freely in the horizontal plane. The deflection which is the transverse displacement of the link from the X -axis is denoted by $w(x,t)$.

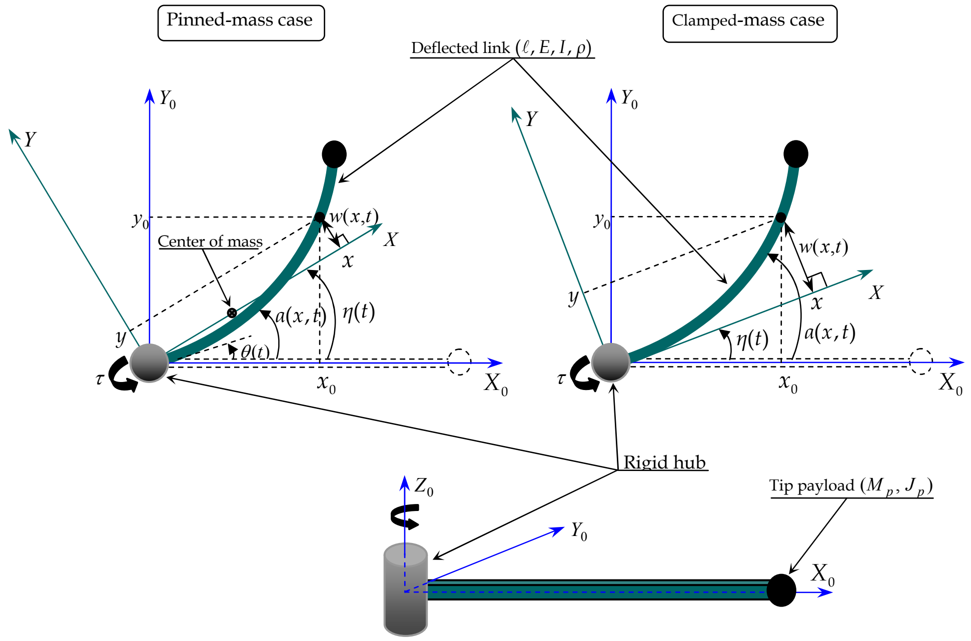


Fig. 1. Pinned and clamped configurations of the considered flexible link manipulator arm

The Fig. 1. is a "top" view of the manipulator in deflection and the axis of rotation of the rigid hub (Z_0) is perpendicular to robot evolution plane. The $X_0 - Y_0$ coordinate frame is the inertial frame of reference. The one indicated by $X - Y$ is a frame of reference that rotates with the overall structure.

For the pinned case (Loudini et al., 2007a; Loudini et al., 2007b), the X -axis is intersecting the center of mass of the whole system. In the clamped case (Loudini et al., 2006), the X -

axis is tangent to the beam at the base (Bellezza et al., 1990).

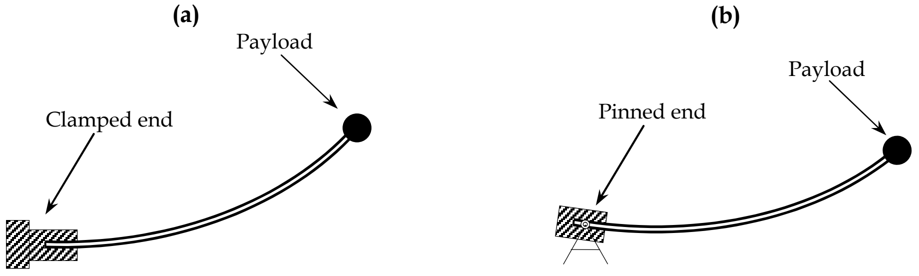


Fig. 2. The two different cases: (a) Clamped-Mass, (b) Pinned-Mass

In Fig. 2., the first case is named Clamped-Mass, meaning that one end is blocked in both angular and vertical direction, and the other end is carrying an inertia load.
The second case is named Pinned-Mass and, as before, it is locked at one end in the vertical direction but free to move in the angular like if it were mounted to a rotary actuator that did not provide a torque, and carrying an inertia load at the other end.
Considering, as usual, the flexible link as a beam, its cross-section height is assumed to be larger than the base. This constrains deflections to occur only in the horizontal plane. Thus, those due to gravity are assumed negligible.
As depicted in Fig. 1, the robot manipulator is essentially composed of a rigid hub, a flexible link and a payload. These three parts are characterized by different physical and mechanical parameters (see the nomenclature at the end of the chapter). In particular, the rotating inertia of the actuating servomotor and the pinning (clamping) rigid hub are modeled as a single hub inertia J_h. The payload is modeled as an end mass M_p with a rotational inertia J_p. $\eta(t)$ being the rotating X-axis angular position, the angular position of the hub, $\theta(t)$ (for the pinned case), and that of a point of the deflected link, $a(x,t)$, are, respectively given, for small deflections, by:

$$\theta(t) = \eta(t) + \left.\frac{\partial w(x,t)}{\partial x}\right|_{x=0} \quad \text{(pinned case)} \tag{1}$$

$$a(x,t) = \eta(t) + \frac{w(x,t)}{x} \tag{2}$$

2.3 Derivation of the governing equations of motion

The kinematics of deformation of an element of the deflected link with width dx at position x are shown in Fig. 3. Due to the effect of shear, the original rectangular element changes its shape to somewhat like a parallelogram with its sides slightly curved.

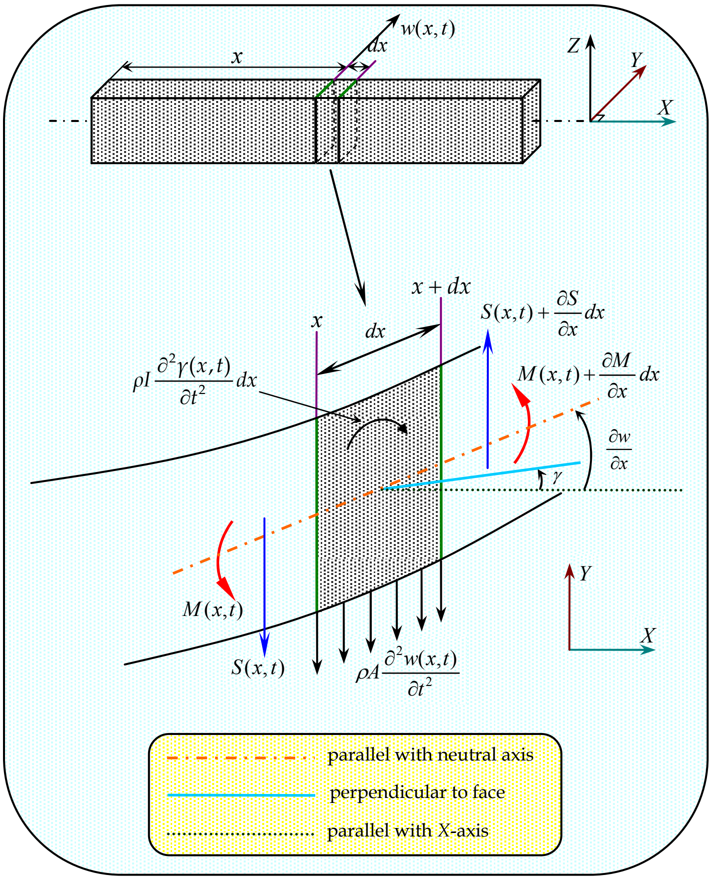

$w(x,t)$
x
dx
Z
Y
X
$x+dx$
x
dx
$S(x,t)+\frac{\partial S}{\partial x}dx$
$\rho I\frac{\partial^2\gamma(x,t)}{\partial t^2}dx$
$M(x,t)+\frac{\partial M}{\partial x}dx$
$\frac{\partial w}{\partial x}$
γ
$M(x,t)$
$S(x,t)$
$\rho A\frac{\partial^2 w(x,t)}{\partial t^2}$
Y
X
parallel with neutral axis
perpendicular to face
parallel with X-axis

Fig. 3. Kinematics of deformation of a bending element

This element undergoes a shearing force $S(x,t)$ and a bending moment $M(x,t)$. On the opposite side, which corresponds to a position $x+dx$, the shearing force ($S+dS$) is

$$S(x+dx,t)=S(x,t)+\frac{\partial S(x,t)}{\partial x}dx \tag{3}$$

Likewise the moment force ($M+dM$) at the position $x+dx$ is

$$M(x+dx,t)=M(x,t)+\frac{\partial M(x,t)}{\partial x}dx \tag{4}$$

Note that the total deflection is due to both bending and shear forces, so that the shear angle $\sigma(x,t)$ (or loss of slope) is equal to the slope of centerline (neutral axis) $\frac{\partial w(x,t)}{\partial x}$ less slope of bending $\gamma(x,\mathrm{t})$:

$$\sigma(x,t)=\frac{\partial w(x,t)}{\partial x}-\gamma(x,t) \tag{5}$$

The shear force S is given by

$$S(x,t)=kAG\sigma(x,t)=kAG\left[\frac{\partial w(x,t)}{\partial x}-\gamma(x,t)\right] \tag{6}$$

By considering the "standard linear solid (SLS) model" or Zener model (Zener, 1965), with the stress-strain law given by

$$\nu+C_D\frac{\partial\nu}{\partial t}=E\varepsilon+K_D\frac{\partial\varepsilon}{\partial t} \tag{7}$$

and assuming linear variations of strain and stress across the beam depth, the total moment obtained by integrating first moment of stress across the beam cross section is (Baker et al., 1967):

$$M(x,t)=\left(1+C_D\frac{\partial}{\partial t}\right)M_0=I\left(E+K_D\frac{\partial}{\partial t}\right)\gamma_x(x,t) \tag{8}$$

The total internal moment (bending and damping) M is then given by (Banks & Inman, 1991; Banks et al., 1994)

$$M(x,t)=EI\frac{\partial\gamma(x,t)}{\partial x}+K_DI\frac{\partial^2\gamma(x,t)}{\partial x\partial t} \tag{9}$$

The equation of motion of the studied single link elastic robot arm can be derived by considering both the equilibrium of the moments and the forces.
Taking moments as positive in the counter-clockwise direction, their summation with disregarding the second order term of dx, yields the relation between the spatial change in the bending moment and the shear force

$$\frac{\partial M(x,t)}{\partial x}=-S(x,t)+\rho I\frac{\partial^2\gamma(x,t)}{\partial t^2} \tag{10}$$

where the term $\rho I\frac{\partial^2\gamma(x,t)}{\partial t^2}$ stands for the distributed rotational inertia given by the product of the mass moment of inertia of the cross section and the angular acceleration.

The relation that fellows balancing forces is

$$\frac{\partial S(x,t)}{\partial x} - A_D \frac{\partial w(x,t)}{\partial t} = \rho A \frac{\partial^2 w(x,t)}{\partial t^2} \tag{11}$$

where the terms $A_D \frac{\partial w(x,t)}{\partial t}$, $\rho A \frac{\partial^2 w(x,t)}{\partial t^2}$ represent, respectively, the air resistance force and the distributed transverse inertial force.

Substitution of (6) and (9) into (10) and likewise (6) into (11) yields the two coupled equations of the damped TB motion:

$$K_D I \frac{\partial^3 \gamma(x,t)}{\partial x^2 \partial t} + EI \frac{\partial^2 \gamma(x,t)}{\partial x^2} + kAG\left(\frac{\partial w(x,t)}{\partial x} - \gamma\right) - \rho I \frac{\partial^2 \gamma(x,t)}{\partial t^2} = 0 \tag{12}$$

$$kAG\left(\frac{\partial^2 w(x,t)}{\partial x^2} - \frac{\partial \gamma(x,t)}{\partial x}\right) - \rho A \frac{\partial^2 w(x,t)}{\partial t^2} - A_D \frac{\partial w(x,t)}{\partial t} = 0 \tag{13}$$

If the damping effects terms are suppressed, the classical set of two coupled PDEs developed by Timoshenko (Timoshenko, 1921; Timoshenko, 1922) arises:

$$EI \frac{\partial^2 \gamma(x,t)}{\partial x^2} + kAG\left(\frac{\partial w(x,t)}{\partial x} - \gamma\right) - \rho I \frac{\partial^2 \gamma(x,t)}{\partial t^2} = 0 \tag{14}$$

$$kAG\left(\frac{\partial^2 w(x,t)}{\partial x^2} - \frac{\partial \gamma(x,t)}{\partial x}\right) - \rho A \frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{15}$$

The modeled beam cross-sectional area and density being uniform, equations (14) and (15) can be easily decoupled as follows:

$$\begin{aligned} &K_D I \frac{\partial^5 w(x,t)}{\partial x^4 \partial t} - \frac{K_D I \rho}{KG} \frac{\partial^5 w(x,t)}{\partial x^2 \partial t^3} + EI \frac{\partial^4 w(x,t)}{\partial x^4} - \rho I\left(1 + \frac{E}{KG} + \frac{K_D A_D}{\rho KAG}\right) \frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \ldots \\ &\quad \ldots \frac{\rho^2 I}{KG} \frac{\partial^4 w(x,t)}{\partial t^4} - \frac{EIA_D}{kAG} \frac{\partial^3 w(x,t)}{\partial x^2 \partial t} + \frac{\rho I A_D}{kAG} \frac{\partial^3 w(x,t)}{\partial t^3} + \rho A \frac{\partial^2 w(x,t)}{\partial t^2} + A_D \frac{\partial w(x,t)}{\partial t} = 0 \end{aligned} \tag{16}$$

$$\begin{aligned} &K_D I \frac{\partial^5 \gamma(x,t)}{\partial x^4 \partial t} - \frac{K_D I \rho}{KG} \frac{\partial^5 \gamma(x,t)}{\partial x^2 \partial t^3} + EI \frac{\partial^4 \gamma(x,t)}{\partial x^4} - \rho I\left(1 + \frac{E}{KG} + \frac{K_D A_D}{\rho KAG}\right) \frac{\partial^4 \gamma(x,t)}{\partial x^2 \partial t^2} + \ldots \\ &\quad \ldots \frac{\rho^2 I}{KG} \frac{\partial^4 \gamma(x,t)}{\partial t^4} - \frac{EIA_D}{kAG} \frac{\partial^3 \gamma(x,t)}{\partial x^2 \partial t} + \frac{\rho I A_D}{kAG} \frac{\partial^3 \gamma(x,t)}{\partial t^3} + \rho A \frac{\partial^2 \gamma(x,t)}{\partial t^2} + A_D \frac{\partial \gamma(x,t)}{\partial t} = 0 \end{aligned} \tag{17}$$

Similar to the those established in (De Silva, 1976; Sooraksa & Chen, 1998), equation (16) is the fifth order TB homogeneous linear PDE with internal and external damping effects expressing the deflection $w(x,t)$.

We have added to this equation the following initial and pinned (clamped)-mass boundary conditions (Loudini et al., 2007a, Loudini et al., 2006):

Initial conditions:
$$w(x,0) = w_0 \,, \quad \left.\frac{\partial w(x,t)}{\partial t}\right|_{t=0} = \dot{w}_0 \tag{18}$$

Pinned end:
$$w(0,t) = 0 \,, \quad \left[M(x,t) - J_h \frac{\partial^3 w(x,t)}{\partial x \partial t^2} \right]_{x=0} = 0 \tag{19}$$

Clamped end:
$$w(0,t) = 0 \,, \quad \left.\frac{\partial w(x,t)}{\partial x}\right|_{x=0} = 0 \tag{20}$$

Free end with payload mass:
$$\begin{cases} \left[\dfrac{\partial M(x,t)}{\partial x} - M_p \dfrac{\partial^2 w(x,t)}{\partial t^2} \right]_{x=\ell} = 0 \\ \\ \left[M(x,t) + J_p \dfrac{\partial^3 w(x,t)}{\partial x \partial t^2} \right]_{x=\ell} = 0 \end{cases} \tag{21}$$

The classical fourth order TB PDE is retrieved if the damping effects terms are suppressed:

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} - \rho I\left(1+\frac{E}{KG}\right)\frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \frac{\rho^2 I}{KG}\frac{\partial^4 w(x,t)}{\partial t^4} + \rho A\frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{22}$$

If the effect due to the rotary inertia is neglected, we are led to the shear beam (SB) model (Morris, 1996; Han et al., 1999):

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} - \frac{\rho IE}{KG}\frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \rho A\frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{23}$$

but, if the one due to shear distortion is the neglected one, the Rayleigh beam equation (Han et al., 1999; Rayleigh, 2003) arises:

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} - \rho I\frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \rho A\frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{24}$$

Moreover, if both the rotary inertia and shear deformation are neglected, then the governing equation of motion reduces to that based on the classical EBT (Meirovitch, 1986) given by

$$EI\frac{\partial^4 w(x,t)}{\partial x^4}+\rho A\frac{\partial^2 w(x,t)}{\partial t^2}=0 \tag{25}$$

If the above included damping effects are associated to the EBB, the corresponding PDE is

$$K_D I\frac{\partial^5 w(x,t)}{\partial x^4 \partial t}+EI\frac{\partial^4 w(x,t)}{\partial x^4}+\rho A\frac{\partial^2 w(x,t)}{\partial t^2}+A_D\frac{\partial w(x,t)}{\partial t}=0 \tag{26}$$

The resolution of the PDE with mixed derivative terms (16) is a complex mathematical problem. Among the few methods existing in the literature, we cite the following approaches with some representative works: the finite element method (Kapur, 1966; Hoa, 1979; Kolberg 1987), the Galerkin method (Wang and Chou, 1998; Dadfarnia et al., 2005), the Rayleigh-Ritz method (Oguamanam and Heppler, 1996), the Laplace transform method resulting in an integral form solution (Boley & Chao, 1955; Wang & Guan, 1994; Ortner & Wagner, 1996), and the eigenfunction expansion method, also referred to as the series or modal expansion method (Anderson, 1953; Dolph, 1954; Huang, 1961; Ekwaro-Osire et al., 2001; Loudini et al. 2006; Loudini et al. 2007a; Loudini et al. 2007b).

In the latter one, $w(x,t)$ can take the following expanded separated form which consists of an infinite sum of products between the chosen transverse deflection eigenfunctions or mode shapes $W_n(x)$, that must satisfy the pinned (clamped)-free (mass) BCs, and the time-dependant modal generalized coordinates $\delta_n(t)$:

$$w(x,t)=\sum_{n=1}^{\infty}W_n(x)\delta_n(t) \tag{27}$$

2.4 Dynamic model deriving procedure

In order to obtain a set of ordinary differential equations (ODEs) of motion to adequately describe the dynamics of the flexible link manipulator, the Hamilton's or Lagrange's approach combined with the Assumed Modes Method (AMM) (Fraser & Daniel, 1991; Loudini et al. 2006; Loudini et al. 2007a; Loudini et al. 2007b; Tokhi & Azad, 2008) can be used.

According to the Lagrange's method, a dynamic system completely located by n generalized coordinates q_i must satisfy n differential equations of the form:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right)-\frac{\partial L}{\partial q_i}+\frac{\partial D}{\partial \dot{q}_i}=F_i\,,\ i=0,1,2,\cdots \tag{28}$$

where L is the so-called Lagrangian given by

$$L=T-U \tag{29}$$

T represents the kinetic energy of the modeled system and U its potential energy. Also, in (28) D is the Rayleigh's dissipation function which allows dissipative effects to be included, and F_i is the generalized external force acting on the corresponding coordinate q_i.
Theoretically there are infinite number of ODEs, but for practical considerations, such as boundedness of actuating energy and limitation of the actuators and the sensors working frequency range, it is more reasonable to truncate this number at a finite one n (Cannon & Schmitz, 1984; Kanoh & Lee, 1985; Qi & Chen, 1992).

The total kinetic energy of the robot flexible link and its potential energy due to the internal bending moment and the shear force are, respectively, given by (Macchelli & Melchiorri, 2004; Loudini et al. 2006; Loudini et al. 2007a; Loudini et al. 2007b):

$$T = \frac{1}{2}\int_0^{\ell} \rho A \left[\frac{\partial w(x,t)}{\partial t}\right]^2 dx + \frac{1}{2}\int_0^{\ell} \rho I \left[\frac{\partial \gamma(x,t)}{\partial t}\right]^2 dx \tag{30}$$

$$U = \frac{1}{2}\int_0^{\ell} EI \left[\frac{\partial \gamma(x,t)}{\partial x}\right]^2 dx + \frac{1}{2}\int_0^{\ell} KAG\left[\sigma(x,t)\right]^2 dx \tag{31}$$

The dissipated energy due to the damping effects can be written as (Krishnan & Vidyasagar, 1988; Loudini et al. 2006; Loudini et al. 2007a; Loudini et al. 2007b):

$$D = \frac{1}{2}\int_0^{\ell} A_D \left[\frac{\partial w(x,t)}{\partial t}\right]^2 dx + \frac{1}{2}\int_0^{\ell} K_D I \left[\frac{\partial^3 w(x,t)}{\partial x^2 \partial t}\right]^2 dx \tag{32}$$

Substituting these energies expressions into (28) accordingly and using the transverse deflection separated form (27), we can derive the desired dynamic equations of motion in the mass (B), damping (H), Coriolis and centrifugal forces (N) and stiffness (K) matrix familiar form:

$$B\frac{d^2q(t)}{dt^2} + H\frac{dq(t)}{dt} + N\big(q(t),\dot{q}(t)\big) + Kq(t) = F(t) \tag{33}$$

with $q(t) = \begin{bmatrix}\theta(t) & \delta_1(t) & \delta_2(t) & \cdots & \delta_n(t)\end{bmatrix}^T$; $F(t) = \begin{bmatrix}\tau & 0 & 0 & \cdots & 0\end{bmatrix}^T$.

If we disregard some high order and nonlinear terms, under reasonable assumptions, the matrix differential equation in (33) could be easily represented in a state-space form as

$$\begin{cases}\dot{z}(t) = A_z z(t) + B_z u(t) \\ y(t) = C_z z(t)\end{cases} \tag{34}$$

with $u(t)=\begin{bmatrix}\tau & 0 & \cdots & 0\end{bmatrix}^T$; $z(t)=\begin{bmatrix}\theta(t) & \delta_1(t) & \cdots & \delta_n(t) & \dot{\theta}(t) & \dot{\delta}_1(t) & \cdots & \dot{\delta}_n(t)\end{bmatrix}^T$.

Solving the state-space matrices gives the vector of states $z(t)$, that is, the angular displacement, the modal amplitudes and their velocities.

3. A Special Case Study: Comprehensive Dynamic Modeling of a Flexible Link Manipulator Considered as a Shear Deformable Timoshenko Beam

In this second part of our work, we present a novel dynamic model of a planar single-link flexible manipulator considered as a tip mass loaded pinned-free shear deformable beam. Using the classical TBT described in section 2 and including the Kelvin-Voigt structural viscoelastic effect (Christensen, 2003), the lightweight robotic manipulator motion governing PDE is derived. Then, based on the Lagrange's principle combined with the AMM, a dynamic model suitable for control purposes is established.

3.1 System description and motion governing equation

The considered physical system is shown in Fig. 4. The basic deriving procedure to obtain the motion governing equation has been described in the previous section, and so only an outline giving the main steps is presented here.
The effect of rotary inertia being neglected in this case study, equation (10) expressing the equilibrium of the moments becomes:

$$\frac{\partial M(x,t)}{\partial x}=-S(x,t) \tag{35}$$

The relation that fellows balancing forces is

$$\frac{\partial S(x,t)}{\partial x}=\rho A\frac{\partial^2 w(x,t)}{\partial t^2} \tag{36}$$

Substitution of 6 and 9 into 35 and likewise 6 into 36 yields the two coupled equations of the damped SB motion:

$$K_D I\frac{\partial^3\gamma(x,t)}{\partial x^2\partial t}+EI\frac{\partial^2\gamma(x,t)}{\partial x^2}+kAG\left[\frac{\partial w(x,t)}{\partial x}-\gamma(x,t)\right]=0 \tag{37}$$

$$kAG\left[\frac{\partial^2 w(x,t)}{\partial x^2}-\frac{\partial\gamma(x,t)}{\partial x}\right]-\rho A\frac{\partial^2 w(x,t)}{\partial t^2}=0 \tag{38}$$

Equations 37 and 38 can be easily decoupled to obtain the fifth order SB homogeneous linear PDEs with internal damping effect expressing the deflection $w(x,t)$ and the slope of bending $\gamma(x,\mathrm{t})$

$$K_D I \frac{\partial^5 w(x,t)}{\partial x^4 \partial t} - \frac{\rho K_D I}{KG} \frac{\partial^5 w(x,t)}{\partial x^2 \partial t^3} + EI \frac{\partial^4 w(x,t)}{\partial x^4} - \frac{\rho EI}{KG} \frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \rho A \frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{39}$$

$$K_D I \frac{\partial^5 \gamma(x,t)}{\partial x^4 \partial t} - \frac{\rho K_D I}{KG} \frac{\partial^5 \gamma(x,t)}{\partial x^2 \partial t^3} + EI \frac{\partial^4 \gamma(x,t)}{\partial x^4} - \frac{\rho EI}{KG} \frac{\partial^4 \gamma(x,t)}{\partial x^2 \partial t^2} + \rho A \frac{\partial^2 \gamma(x,t)}{\partial t^2} = 0 \tag{40}$$

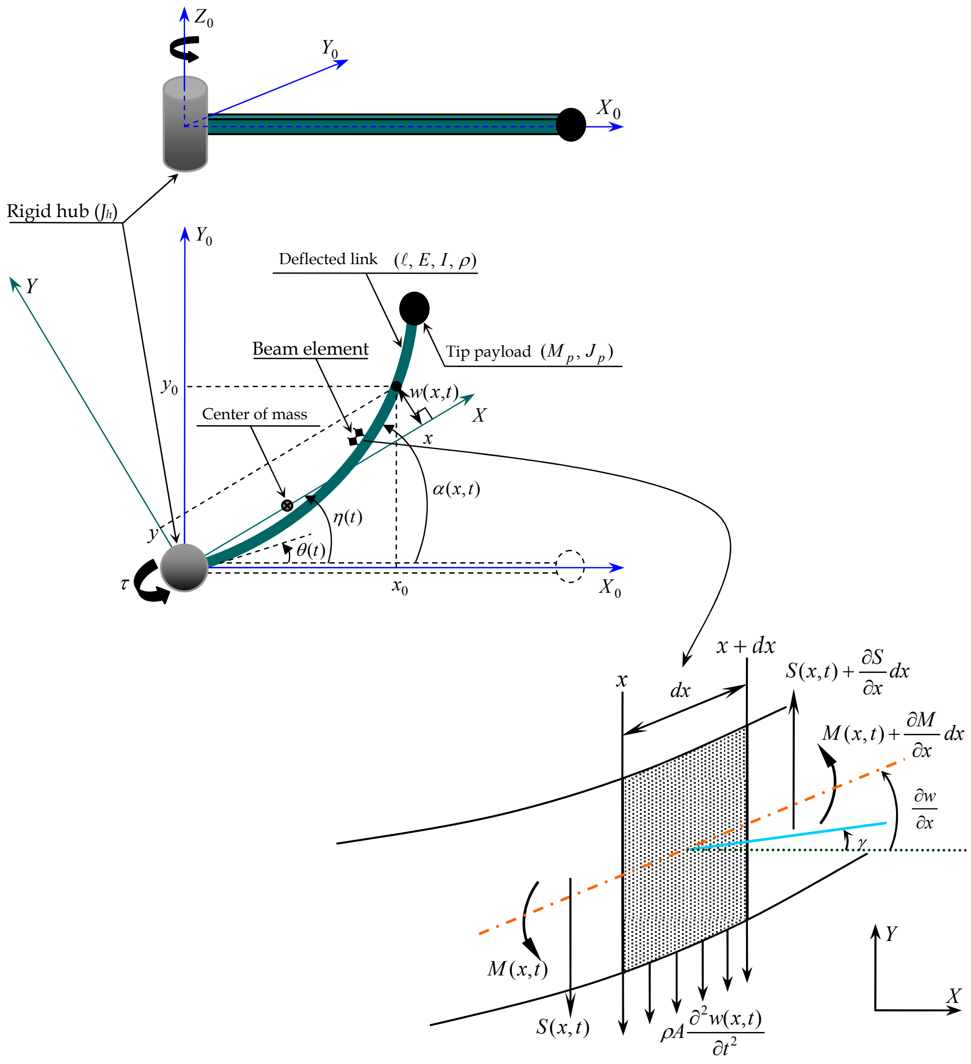

Z_0
Y_0
X_0
Rigid hub (J_h)
Y_0
Y
Deflected link (ℓ, E, I, ρ)
Beam element
Tip payload (M_p, J_p)
y_0
$w(x,t)$
X
Center of mass
x
$\alpha(x,t)$
$\eta(t)$
$\theta(t)$
y
τ
x_0
X_0
$x + dx$
x
dx
$S(x,t) + \frac{\partial S}{\partial x} dx$
$M(x,t) + \frac{\partial M}{\partial x} dx$
$\frac{\partial w}{\partial x}$
γ
Y
X
$M(x,t)$
$S(x,t)$
$\rho A \frac{\partial^2 w(x,t)}{\partial t^2}$

Fig. 4. Physical configuration and kinematics of deformation of a bending element of the studied flexible robot manipulator considered as a shear deformable beam

We affect to the equation (39) the same initial and pinned-mass boundary conditions, given by equations 18, 19, and 21, with taking into account the result established by (Wang & Guan, 1994; Loudini et al., 2007b) about the very small influence of the tip payload inertia on the flexible manipulator dynamics:

Initial conditions: $$w(x,0) = w_0\,,\;\; w_t(x,t)\big|_{t=0} = \dot{w}_0 \tag{41}$$

BCs at the pinned end (root of the link):

$$w(x,t)\big|_{x=0} = 0 \text{ : zero average translational displacement} \tag{42}$$

$$M(x,t)\big|_{x=0} = J_h \left.\frac{\partial^3 w(x,t)}{\partial x \partial t^2}\right|_{x=0} \text{ : balance of bending moments} \tag{43}$$

BCs at the mass loaded free end:

$$M(x,t)\big|_{x=\ell} = 0 \text{ : zero average of bending moments} \tag{44}$$

$$\left.\frac{\partial M(x,t)}{\partial x}\right|_{x=\ell} = M_p \left.\frac{\partial^2 w(x,t)}{\partial t^2}\right|_{x=\ell} \text{ : balance of shearing forces} \tag{45}$$

The classical fourth order SB PDEs are retrieved if the damping effect term is suppressed:

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} - \frac{\rho EI}{KG}\frac{\partial^4 w(x,t)}{\partial x^2 \partial t^2} + \rho A \frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{46}$$

$$EI\frac{\partial^4 \gamma(x,t)}{\partial x^4} - \frac{\rho EI}{KG}\frac{\partial^4 \gamma(x,t)}{\partial x^2 \partial t^2} + \rho A \frac{\partial^2 \gamma(x,t)}{\partial t^2} = 0 \tag{47}$$

Moreover, if shear deformation is neglected, then the governing equation of motion reduces to that based on the classical EBT, given by 25.

If the above included damping effect is associated to the EBB, the corresponding PDE is

$$K_D I \frac{\partial^5 w(x,t)}{\partial x^4 \partial t} + EI\frac{\partial^4 w(x,t)}{\partial x^4} + \rho A \frac{\partial^2 w(x,t)}{\partial t^2} = 0 \tag{48}$$

To solve the PDEs with mixed derivative terms (39) and (40), we have tried to apply the classical AMM which is well known as a computationally efficient scheme that separates the mode functions from the shape functions.
The forms of equations (39) and (40) being identical, $w(x,t)$ and $\gamma(x,t)$ are assumed to

share the same time-dependant modal generalized coordinate $\delta(t)$ under the following separated forms with the respective mode shape functions (eigenfuntions) $\Phi(x)$ and $\Psi(x)$ that must satisfy the pinned-free (mass) BCs:

$$\begin{aligned} w(x,t) &= \Phi(x)\delta(t) \\ \gamma(x,t) &= \Psi(x)\delta(t) \end{aligned} \tag{49}$$

Unfortunately, the application of 49 has not been possible to derive the mode shapes expressions. This is due to the unseparatability of some terms of 39 and 40.
To find a way to solve the problem, we have based our investigations on the result pointed out in (Gürgöze et al., 2007). In this work, it has been established that the characteristic equation of a visco-elastic EBB i.e., a Kelvin-Voigt model (given in our chapter by 48), is formally the same as the frequency equation of the cantilevered elastic beam (the EB modeled by 25). Thus, we can assume that the damping effect affects only the modal function $\delta(t)$. So, the mode shape is that of the SB model (46, 47).

Applying the AMM to the PDEs 46 and 47, we obtain

$$EI\Phi^{iv}(x)\delta(t) - \frac{\rho EI}{KG}\Phi''(x)\ddot{\delta}(t) + \rho A\Phi(x)\ddot{\delta}(t) = 0 \tag{50}$$

$$EI\Psi^{iv}(x)\delta(t) - \frac{\rho EI}{KG}\Psi''(x)\ddot{\delta}(t) + \rho A\Psi(x)\ddot{\delta}(t) = 0 \tag{51}$$

Separating the functions of time, t, and space x:

$$\frac{\ddot{\delta}(t)}{\delta(t)} = -\frac{\Phi^{iv}(x)}{\frac{\rho A}{EI}\Phi(x) - \frac{\rho}{KG}\Phi^{ii}(x)} = -\frac{\Psi^{iv}(x)}{\frac{\rho A}{EI}\Psi(x) - \frac{\rho}{KG}\psi^{ii}(x)} = \text{constant} = -\lambda \tag{52}$$

The differential equation for the temporal modal generalized coordinate is

$$\ddot{\delta}(t) + \lambda\delta(t) = 0 \tag{53}$$

Its general solution is assumed to be in the following forms:

$$\delta(t) = De^{j\omega t} + \bar{D}e^{-j\omega t} = F\cos(\omega t + \varphi) \tag{54}$$

where

$$\lambda = \omega^2 \tag{55}$$

The constants D and its complex conjugate $\bar{D}$ (or F and the phase ϕ) are determined from the initial conditions. The natural frequency ω is determined by solving the spatial problem given by

$$\begin{aligned}
&\Phi^{iv}(x) + \frac{\rho}{KG}\omega^2\Phi^{ii}(x) - \frac{\rho A}{EI}\omega^2\Phi(x) = 0 \\
&\Psi^{iv}(x) + \frac{\rho}{KG}\omega^2\Psi^{ii}(x) - \frac{\rho A}{EI}\omega^2\Psi(x) = 0
\end{aligned} \tag{56}$$

The solutions of 56 can be written in terms of sinusoidal and hyperbolic functions

$$\begin{aligned}
\Phi(x) &= C_1 \sin ax + C_2 \cos ax + C_3 \sinh bx + C_4 \cosh bx \\
\Psi(x) &= D_1 \sin ax + D_2 \cos ax + D_3 \sinh bx + D_4 \cosh bx
\end{aligned} \tag{57}$$

where

$$a = \sqrt{\frac{\rho}{2KG}\omega^2 + \sqrt{\left(\frac{\rho}{2KG}\omega^2\right)^2 + \frac{\rho A}{EI}\omega^2}} \quad ; \quad b = \sqrt{-\frac{\rho}{2KG}\omega^2 + \sqrt{\left(\frac{\rho}{2KG}\omega^2\right)^2 + \frac{\rho A}{EI}\omega^2}} \tag{58}$$

The constants $C_k, D_k; k = \overline{1,4}$ of 57 are determined through the BCs 42-45 rewritten on the basis of 49, 53 and 55 as follows:

$$\Phi(0) = 0 \tag{59}$$

$$\Psi'(0) = -\frac{J_h}{EI}\omega^2\Phi'(0) = -J\Phi'(0) \tag{60}$$

$$\Psi'(\ell) = 0 \tag{61}$$

$$\Phi'(\ell) - \Psi(\ell) = \frac{M_p}{KAG}\omega^2\Phi(\ell) = M\Phi(\ell) \tag{62}$$

By applying 59-62 to 57, we find these relations

$$C_2 = -C_4 \tag{63}$$

$$aD_1 + bD_3 = -aJC_1 - bJC_3 \tag{64}$$

$$aD_1 \cos a\ell - aD_2 \sin a\ell + bD_3 \cosh b\ell + bD_4 \sinh b\ell = 0 \tag{65}$$

$$\begin{aligned}&\left(C_1a - C_2M - D_2\right)\cos a\ell - \left(C_2a + C_1M + D_1\right)\sin a\ell + \left(C_3b - C_4M - D_4\right)\cosh b\ell + \cdots\\ &\qquad\cdots\left(C_4b - C_3M - D_3\right)\sinh b\ell = 0\end{aligned} \tag{66}$$

The relations between the unknown constants C_k and D_k are obtained by substituting 57 into 38:

$$D_1 = \frac{R-a^2}{a}C_2\,;\, D_2 = -\frac{R-a^2}{a}C_1\,;\, D_3 = \frac{R+b^2}{b}C_4\,;\, D_4 = \left(\frac{R+b^2}{b}\right)C_3 \tag{67}$$

or

$$C_1 = -\frac{a}{\left(R-a^2\right)}D_2\,;\, C_2 = \frac{a}{\left(R-a^2\right)}D_1\,;\, C_3 = \frac{b}{\left(R+b^2\right)}D_4\,;\, C_4 = \frac{b}{\left(R+b^2\right)}D_3 \tag{68}$$

where $R = \dfrac{\rho\omega^2}{KG}$.

From 63 and 67, we obtain

$$D_3 = -\frac{a\left(R+b^2\right)}{b\left(R-a^2\right)}D_1 = D_{31}D_1 \tag{69}$$

From some combinations of 63-69, we find the relations

$$C_2 = \left[-\frac{\dfrac{a\left(R+b^2\right)}{b\left(R-a^2\right)}\sinh b\ell - \sin a\ell}{\cos a\ell - \dfrac{R+b^2}{R-a^2}\cosh b\ell + \dfrac{\left(a^2+b^2\right)\left(R+b^2\right)}{bJ\left(R-a^2\right)}\sinh b\ell}\right]C_1 = C_{21}C_1 \tag{70}$$

$$C_3 = \left[\frac{\left(\dfrac{R}{a} - MC_{21}\right)\cos a\ell - \left(\dfrac{R}{a}C_{21} + M\right)\sin a\ell + MC_{21}\cosh b\ell + \dfrac{R}{b}C_{21}\sinh b\ell}{M\sinh b\ell - \dfrac{R}{b}\cosh b\ell}\right]C_1 = C_{31}C_1 \tag{71}$$

$$D_2 = \left[\frac{\cos a\ell - \frac{R+b^2}{R-a^2}\cosh b\ell + \frac{\left(a^2+b^2\right)\left(R+b^2\right)}{bJ\left(R-a^2\right)}\sinh b\ell}{\frac{a\left(R+b^2\right)}{b\left(R-a^2\right)}\sinh b\ell - \sin a\ell} \right] D_1 = D_{21}D_1 \tag{72}$$

$$D_4 = \left[\frac{\frac{aMD_{21}-R}{R-a^2}\sin a\ell - \frac{RD_{21}+aM}{R-a^2}\cos a\ell + \left(\frac{aR}{b\left(R-a^2\right)} + \frac{bM}{R+b^2} \right)\sinh b\ell + \cdots \cdots \frac{aM}{R-a^2}\cosh b\ell}{\frac{R}{R+b^2}\cosh b\ell} \right] D_1 \tag{73}$$

$$= D_{41}D_1$$

Replacing (63) and (69)-(73) into (57), we obtain

$$\begin{aligned} \Phi(x) &= C_1\left[\sin ax + C_{21}\left(\cos ax - \cosh bx\right) + C_{31}\sinh bx\right] \\ \Psi(x) &= D_1\left[\sin ax + D_{21}\cos ax + D_{31}\sinh bx + D_{41}\cosh bx\right] \end{aligned} \tag{74}$$

In order to establish the frequency equation, we rewrite the equations 63-66 as fellows

$$C_2 + C_4 = 0 \tag{75}$$

$$aJC_1 + \left(R-a^2\right)C_2 + bJC_3 + \left(R+b^2\right)C_4 = 0 \tag{76}$$

$$\underbrace{\left(R-a^2\right)\sin a\ell}_{CF_1} C_1 + \underbrace{\left(R-a^2\right)\cos a\ell}_{CF_2} C_2 + \underbrace{\left(R+b^2\right)\sinh b\ell}_{CF_3} C_3 + \underbrace{\left(R+b^2\right)\cosh b\ell}_{CF_4} C_4 = 0 \tag{77}$$

$$\begin{aligned} &\underbrace{\left(\frac{R}{a}\cos a\ell - M\sin a\ell\right)}_{CF_5} C_1 + \underbrace{\left(-M\cos a\ell - \frac{R}{a}\sin a\ell\right)}_{CF_6} C_2 + \underbrace{\left(-\frac{R}{b}\cosh b\ell - M\sinh b\ell\right)}_{CF_7} C_3 + \cdots \\ &\cdots \underbrace{\left(-\frac{R}{b}\sinh b\ell - M\cosh b\ell\right)}_{CF_8} C_4 = 0 \end{aligned} \tag{78}$$

Consider the coefficients of the four equations as a matrix C given by

$$C = \begin{bmatrix} 0 & 1 & 0 & 1 \\ aJ & R-a^2 & bJ & R+b^2 \\ CF_1 & CF_2 & CF_3 & CF_4 \\ CF_5 & CF_6 & CF_7 & CF_8 \end{bmatrix} \tag{79}$$

In order that solutions other than zero may exist, the determinant of C must me null. This leads to the frequency equation

$$\begin{gathered} -\frac{a}{b}RJ\left(R+b^2\right)+\left(a^2+b^2\right)\left[MaJ+\frac{R}{a}\left(R+b^2\right)\right]\cos a\ell \sinh b\ell+\cdots \\ \cdots\left(a^2+b^2\right)\left[MbJ+\frac{R}{b}\left(R-a^2\right)\right]\sin a\ell \cosh b\ell+\cdots \\ \cdots\left[RJ\left(a^2+b^2\right)+\frac{b}{a}RJ\left(R-a^2\right)-M\left(a^2+b^2\right)^2\right]\sin a\ell \sinh b\ell-\cdots \\ \cdots J\left[\frac{a}{b}\left(R-a^2\right)+\frac{b}{a}\left(R+b^2\right)\right]\cos a\ell \cosh b\ell=0 \end{gathered} \tag{80}$$

3.2 Derivation of the dynamic model

As explained before, the energetic Lagrange's principle is adopted.
The total kinetic energy is given by

$$T = T_\ell + T_h + T_p \tag{81}$$

where T_h, T_ℓ and T_p are the kinetic energies associated to, respectively, the rigid hub, the flexible link, and the payload:

$$T_h = \frac{1}{2}J_h\dot{\theta}^2(t) \tag{82}$$

$$T_\ell = \frac{1}{2}\int_0^\ell \rho A\left\{\left[x\dot{\theta}(t)+\frac{\partial w(x,t)}{\partial t}\right]^2+\left[\dot{\theta}(t)w(x,t)\right]^2\right\}dx \tag{83}$$

$$T_p = \frac{1}{2}M_p\left\{\left[\left(x\dot{\theta}(t)+\frac{\partial w(x,t)}{\partial t}\right)_{x=\ell}\right]^2+\left[\dot{\theta}(t)w(\ell,t)\right]^2\right\}+\frac{1}{2}J_p\left\{\left[\dot{\theta}(t)+\frac{\partial}{\partial t}\left(\left.\frac{\partial w(x,t)}{\partial x}\right|_{x=\ell}\right)\right]^2\right\} \tag{84}$$

The potential energy of the system, U, can be written as

$$U=\frac{1}{2}\int_0^\ell EI\left[\frac{\partial\gamma(x,t)}{\partial x}\right]^2 dx+\frac{1}{2}\int_0^\ell KAG\left[\frac{\partial w(x,t)}{\partial x}-\gamma(x,t)\right]^2 dx \tag{85}$$

The dissipated energy D may be written as

$$D=\frac{1}{2}\int_0^\ell K_D I\left[\frac{\partial^3 w(x,t)}{\partial x^2\partial t}\right]^2 dx \tag{86}$$

Using, for ease of manipulation, the following notations and substitutions

$$\Phi_{i\ell}=\Phi_i(\ell)\,;\,\Phi_{i\ell}^{\;2}=\Phi_i^{\,2}(\ell)\,;\,\Phi'_{i\ell}=\Phi'_i(\ell)\,;\,\Phi'^{\,2}_{i\ell}=\Phi'^{2}_i(\ell);\,\Gamma_{1_i}=\int_0^\ell\Phi_i^{\,2}(x)dx\,;\,\Gamma_{1_{12}}=\int_0^\ell\Phi_1(x)\Phi_2(x)dx;$$

$$\Gamma_{2_i}=\int_0^\ell\Psi_i^{\,2}(x)dx\,;\,\Gamma_{2_{12}}=\int_0^\ell\Psi_1(x)\Psi_2(x)dx;\Gamma_{3_i}=\int_0^\ell\Phi'^{2}_i(x)dx\,;\,\Gamma_{3_{12}}=\int_0^\ell\Phi'_1(x)\Phi'_2(x)dx;$$

$$\Gamma_{4_i}=\int_0^\ell\Psi'^{2}_i(x)dx\,;\,\Gamma_{4_{12}}=\int_0^\ell\Psi'_1(x)\Psi'_2(x)dx\,;\,\Gamma_{5_i}=\int_0^\ell\Phi''^{2}_i(x)dx\,;\,\Gamma_{5_{12}}=\int_0^\ell\Phi''_1(x)\Phi''_2(x)dx\,;\,\Gamma_{6_i}=\int_0^\ell x\Phi_i(x)dx;$$

$$\Gamma_{7_i}=\int_0^\ell\Phi'_i(x)\Psi_i(x)dx\,;\,\Gamma_{7_{12}}=\int_0^\ell\Phi'_1(x)\Psi_2(x)dx\,;\,\Gamma_{7_{21}}=\int_0^\ell\Phi'_2(x)\Psi_1(x)dx.$$

we obtain

$$\begin{aligned}
L=&\frac{1}{2}\left(J_h+J_p+M_p\ell^2+\frac{1}{3}\rho A\ell^3\right)\dot\theta^2(t)+\frac{1}{2}\left(M_p\Phi_{1\ell}^{\;2}+\rho A\Gamma_{1_1}\right)\delta_1^{\,2}(t)\dot\theta^2(t)+\cdots\\
&\cdots\frac{1}{2}\left(M_p\Phi_{2\ell}^{\;2}+\rho A\Gamma_{1_2}\right)\delta_2^{\,2}(t)\dot\theta^2(t)+\left(M_p\Phi_{1\ell}\Phi_{2\ell}+\rho A\Gamma_{1_{12}}\right)\delta_1(t)\delta_2(t)\dot\theta^2(t)+\cdots\\
&\cdots\left(\rho A\Gamma_{6_1}+M_p\ell\Phi_{1\ell}+J_p\Phi'_{1\ell}\right)\dot\theta(t)\dot\delta_1(t)+\left(\rho A\Gamma_{6_2}+M_p\ell\Phi_{2\ell}+J_p\Phi'_{2\ell}\right)\dot\theta(t)\dot\delta_2(t)+\cdots\\
&\cdots\frac{1}{2}\left(\rho A\Gamma_{1_1}+M_p\Phi_{1\ell}^{\;2}+J_p\Phi'^{\,2}_{1\ell}\right)\dot\delta_1^{\,2}(t)+\frac{1}{2}\left(\rho A\Gamma_{1_2}+M_p\Phi_{2\ell}^{\;2}+J_p\Phi'^{\,2}_{2\ell}\right)\dot\delta_2^{\,2}(t)+\cdots\\
&\cdots\left(\rho A\Gamma_{1_{12}}+M_p\Phi_{1\ell}\Phi_{2\ell}+J_p\Phi'_{1\ell}\Phi'_{2\ell}\right)\dot\delta_1(t)\dot\delta_2(t)+\cdots\\
&\cdots\left[KAG\Gamma_{7_1}-\frac{1}{2}KAG\left(\Gamma_{3_1}+\Gamma_{2_1}\right)-\frac{1}{2}EI\Gamma_{4_1}\right]\delta_1^{\,2}(t)+\cdots\\
&\cdots\left[KAG\Gamma_{7_2}-\frac{1}{2}KAG\left(\Gamma_{3_2}+\Gamma_{2_2}\right)-\frac{1}{2}EI\Gamma_{4_2}\right]\delta_2^{\,2}(t)+\cdots\\
&\cdots\left[KAG\Gamma_{7_{12}}+KAG\Gamma_{7_{21}}-KAG\left(\Gamma_{3_{12}}+\Gamma_{2_{12}}\right)-EI\Gamma_{4_{12}}\right]\delta_1(t)\delta_2(t)
\end{aligned} \tag{87}$$

$$D = \frac{1}{2} K_D I \dot{\delta}_1^2(t) \Gamma_{5_1} + \frac{1}{2} K_D I \dot{\delta}_2^2(t) \Gamma_{5_2} + K_D I \dot{\delta}_1(t) \dot{\delta}_2(t) \Gamma_{5_{12}} \tag{88}$$

Based on the Lagrange's principle combined with the AMM, and after tedious manipulations of extremely lengthy expressions, the established dynamic equations of motion are obtained in a matrix form by:

$$\underbrace{\begin{bmatrix} B_{11}(q) & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}}_{B(q)} \begin{bmatrix} \ddot{\theta}(t) \\ \ddot{\delta}_1(t) \\ \ddot{\delta}_2(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{bmatrix}}_{H} \begin{bmatrix} \dot{\theta}(t) \\ \dot{\delta}_1(t) \\ \dot{\delta}_2(t) \end{bmatrix} + \underbrace{\begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix}}_{N(q,\dot{q})} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & K_{22} & K_{23} \\ 0 & K_{32} & K_{33} \end{bmatrix}}_{K} \begin{bmatrix} \theta(t) \\ \delta_1(t) \\ \delta_2(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \tau \\ 0 \\ 0 \end{bmatrix}}_{F} \tag{89}$$

where

$$B_{11} = J_h + J_p + M_p \ell^2 + \frac{1}{3} \rho A \ell^3 + \left(M_p \Phi_{1\ell}^2 + \rho A \Gamma_{1_1} \right) \delta_1^2(t) + \left(M_p \Phi_{2\ell}^2 + \rho A \Gamma_{1_2} \right) \delta_2^2(t) + 2\left(M_p \Phi_{1\ell} \Phi_{2\ell} + \rho A \Gamma_{1_{12}} \right) \delta_1(t) \delta_2(t);$$

$$B_{12} = \rho A \Gamma_{6_1} + M_p \ell \Phi_{1\ell} + J_p \Phi'_{1\ell}; \; B_{13} = \rho A \Gamma_{6_2} + M_p \ell \Phi_{2\ell} + J_p \Phi'_{2\ell}; \; B_{21} = \rho A \Gamma_{6_1} + M_p \ell \Phi_{1\ell} + J_p \Phi'_{1\ell};$$

$$B_{22} = \rho A \Gamma_{1_1} + M_p \Phi_{1\ell}^2 + J_p \Phi'^2_{1\ell}; \; B_{23} = \rho A \Gamma_{1_{12}} + M_p \Phi_{1\ell} \Phi_{2\ell} + J_p \Phi'_{1\ell} \Phi'_{2\ell};$$

$$B_{31} = \rho A \Gamma_{6_2} + M_p \ell \Phi_{2\ell} + J_p \Phi'_{2\ell}; \; B_{32} = \rho A \Gamma_{1_{12}} + M_p \Phi_{1\ell} \Phi_{2\ell} + J_p \Phi'_{1\ell} \Phi'_{2\ell}; \; B_{33} = \rho A \Gamma_{1_2} + M_p \Phi_{2\ell}^2 + J_p \Phi'^2_{2\ell};$$

$$H_{22} = K_D I \Gamma_{5_1}; \; H_{23} = K_D I \Gamma_{5_{12}}; \; H_{32} = K_D I \Gamma_{5_{12}}; \; H_{33} = K_D I \Gamma_{5_2};$$

$$N_1 = 2\left(M_p \Phi_{1\ell}^2 + \rho A \Gamma_{1_1} \right) \delta_1(t) \dot{\delta}_1(t) \dot{\theta}(t) + 2\left(M_p \Phi_{2\ell}^2 + \rho A \Gamma_{1_2} \right) \delta_2(t) \dot{\delta}_2(t) \dot{\theta}(t) + 2\left(M_p \Phi_{1\ell} \Phi_{2\ell} + \rho A \Gamma_{1_{12}} \right) \left[\dot{\delta}_1(t) \delta_2(t) \dot{\theta}(t) + \delta_1(t) \dot{\delta}_2(t) \dot{\theta}(t) \right];$$

$$N_2 = -\left(M_p \Phi_{1\ell}^2 + \rho A \Gamma_{1_1} \right) \delta_1(t) \dot{\theta}^2(t) - \left(M_p \Phi_{1\ell} \Phi_{2\ell} + \rho A \Gamma_{1_{12}} \right) \delta_2(t) \dot{\theta}^2(t);$$

$$N_3 = -\left(M_p \Phi_{1\ell} \Phi_{2\ell} + \rho A \Gamma_{1_{12}} \right) \delta_1(t) \dot{\theta}^2(t) - \left(M_p \Phi_{2\ell}^2 + \rho A \Gamma_{1_2} \right) \delta_2(t) \dot{\theta}^2(t);$$

$$K_{22} = KAG\left(\Gamma_{3_1} + \Gamma_{2_1} \right) - 2KAG\Gamma_{7_1} + EI\Gamma_{4_1}; \; K_{23} = KAG\left(\Gamma_{3_{12}} + \Gamma_{2_{12}} \right) - KAG\Gamma_{7_{12}} - KAG\Gamma_{7_{21}} + EI\Gamma_{4_{12}};$$

$$K_{32} = KAG\Gamma_{7_{12}} + KAG\Gamma_{7_{21}} - KAG\left(\Gamma_{3_{12}} + \Gamma_{2_{12}} \right) - EI\Gamma_{4_{12}}; \; K_{33} = 2KAG\Gamma_{7_2} - KAG\left(\Gamma_{3_2} + \Gamma_{2_2} \right) - EI\Gamma_{4_2}$$

4. Conclusion

An investigation into the development of flexible link robot manipulators mathematical models, with a high modeling accuracy, using Timoshenko beam theory concepts has been presented.

The emphasis has been, essentially, set on obtaining accurate and complete equations of motion that display the most relevant aspects of structural properties inherent to the modeled lightweight flexible robotic structure.

In particular, two important damping mechanisms: internal structural viscoelasticity effect (Kelvin-Voigt damping) and external viscous air damping have been included in addition to the classical effects of shearing and rotational inertia of the elastic link cross-section.

To derive a closed-form finite-dimensional dynamic model for the planar lightweight robot arm, the main steps of an energetic deriving procedure based on the Lagrangian approach combined with the assumed modes method has been proposed.

An illustrative application case of the general presentation has been rigorously highlighted. As a contribution, a new comprehensive mathematical model of a planar single link flexible manipulator considered as a shear deformable Timoshenko beam with internal structural viscoelasticity is proposed.

On the basis of the combined Lagrangian-Assumed Modes Method with specific accurate boundary conditions, the full development details leading to the establishment of a closed form dynamic model have been explicitly given.

In a coming work, a digital simulation will be performed in order to reveal the vibrational behavior of the modeled system and the relation between its dynamics and its parameters. It is also planned to do some comparative studies with other dynamic models.

The mathematical model resulting from this work could, certainly, be quite suitable for control purposes. Moreover, an extension to the multi-link case, requiring very high modeling accuracy to avoid the cumulative errors, should be a very good topic for further investigation.

5. References

Aldraheim, O. J.; Wetherhold, R. C. & Singh, T. (1997). Intelligent Beam Structures: Timoshenko Theory vs. Euler-Bernoulli Theory, *Proceedings of the IEEE International Conference on Control Applications*, pp. 976-981, ISBN: 0-7803-2975-9, Dearborn, September 1997, MI, USA.

Anderson, R. A. (1953). Flexural Vibrations in Uniform Beams according to the Timoshenko Theory. *Journal of Applied Mechanics*, Vol. 20, No. 4, (1953) 504-510, ISSN: 0021-8936.

Baker, W. E.; Woolam, W. E. & Young, D. (1967). Air and internal damping of thin cantilever beams. *International Journal of Mechanical Sciences*, Vol. 9, No. 11, (November 1967) 743-766, ISSN: 0020-7403.

Banks, H. T. & Inman, D. J. (1991). On damping mechanisms in beams. *Journal of Applied Mechanics*, Vol. 58, No. 3, (September 1991) 716-723, ISSN: 0021-8936.

Banks, H. T.; Wang, Y. & Inman, D. J. (1994). Bending and shear damping in beams: Frequency domain techniques. *Journal of Vibration and Acoustics*, Vol. 116, No. 2, (April 1994) 188-197, ISSN: 1048-9002.

Baruh, H. & Taikonda, S. S. K. (1989). Issues in the dynamics and control of flexible robot manipulators. *AIAA Journal of Guidance, Control and Dynamics*, Vol. 12, No. 5, (September-October 1989) 659-671, ISSN: 0731-5090.

Bellezza, F.; Lanari, L. & Ulivi, G. (1990). Exact modeling of the slewing flexible link, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 734-739, ISBN: 0-8186-9061-5, Cincinnati, May 1990, OH, USA.

Benosman, M.; Boyer, F.; Vey, G. L. & Primautt, D. (2002). Flexible links manipulators: from modelling to control. *Journal of Intelligent and Robotic Systems*, Vol. 34, No. 4, (August 2002) 381–414, ISSN: 0921-0296.

Benosman, M. & Vey, G. L. (2004). Control of flexible manipulators: A survey. *Robotica*, Vol. 22, No. 5, (October 2004) 533–545, ISSN: 0263-5747.

Boley, B. A. & Chao, C. C. (1955). Some solutions of the Timoshenko beam equations. *Journal of Applied Mechanics*, Vol. 22, No. 4, (December 1955) 579-586, ISSN: 0021-8936.

Book, W. J. (1990). Modeling, design, and control of flexible manipulator arms: A tutorial review, *Proceedings of the IEEE Conference on Decision and Control*, pp. 500–506, Honolulu, December 1990, HI, USA.

Book, W. J. (1993). Controlled motion in an elastic world. *Journal of Dynamic Systems, Measurement and Control*, Vol. 115, No. 2B, (June 1993) 252-261, ISSN: 0022-0434.

Cannon, R. H. Jr & Schmitz, E. (1984). Initial experiments on the end-point control of a flexible one-link robot. *International Journal of Robotics Research*, Vol. 3, No. 3, (September 1984) 62-75, ISSN: 0278-3649.

Canudas de Wit, C.; Siciliano, B. & Bastin, G. (1996). *Theory of Robot Control*, Springer-Verlag, ISBN: 978-3-540-76054-2, London.

Christensen, R. M. (2003). *Theory of Viscoelasticity.* Dover Publications, ISBN: 978-0-486-42880-2, New York.

Dolph, C. (1954). On the Timoshenko theory of transverse beam vibrations. *Quarterly of Applied Mathematics*, Vol. 12, No. 2, (July 1954) 175-187, ISSN: 0033-569X.

Dwivedy, S. K. & Eberhard, P. (2006). Dynamic analysis of flexible manipulators, a literature review. *Mechanism and Machine Theory*, Vol. 41, No. 7, (July 2006) 749–777, ISSN: 0094-114X.

Ekwaro-Osire, S.; Maithripala, D. H. S. & Berg, J. M. (2001). A Series expansion approach to interpreting the spectra of the Timoshenko beam. *Journal of Sound and Vibration*, Vol. 240, No. 4, (March 2001) 667-678, ISSN: 0022-460X.

Fraser, A. R. & Daniel, R. W. (1991). *Perturbation Techniques for Flexible Manipulators*, Kluwer Academic Publishers, ISBN: 0-7923-9162-4, Norwell, MA, USA.

Dadfarnia, M.; Jalili, N. & Esmailzadeh, E. (2005). A Comparative study of the Galerkin approximation utilized in the Timoshenko beam theory. *Journal of Sound and Vibration*, Vol. 280, No. 3-5, (February 2005) 1132-1142, ISSN: 0022-460X.

Geist, B. & McLaughlin, J. R. (2001). Asymptotic formulas for the eigenvalues of the Timoshenko beam. *Journal of Mathematical Analysis and Applications*, Vol. 253, (January 2001) 341-380, ISSN: 0022-247X.

Gürgöze, M.; Doğruoğlu, A. N. & Zeren, S (2007). On the eigencharacteristics of a cantilevered visco-elastic beam carrying a tip mass and its representation by a spring-damper-mass system. *Journal of Sound and Vibrations*, Vol. 1-2, No. 301, (March 2007) 420-426, ISSN: 0022-460X.

Han, S. M.; Benaroya, H.; & Wei T. (1999). Dynamics of transversely vibrating beams using four engineering theories. *Journal of Sound and Vibration*, Vol. 225, No. 5, (September 1999) 935-988, ISSN: 0022-460X.

Hoa, S. V. (1979). Vibration of a rotating beam with tip mass. *Journal of Sound and Vibration*, Vol. 67, No. 3, (December 1979) 369-381, ISSN: 0022-460X.

Huang, T. C. (1961). The effect of rotary inertia and of shear deformation on the frequency and normal mode equations of uniform beams with simple end conditions. *Journal of Applied Mechanics*, Vol. 28, (1961) 579-584, ISSN: 0021-8936.

Junkins, J. L. & Kim, Y. (1993). *Introduction to Dynamics and Control of Flexible Structures*. AIAA Education Series (J. S. Przemieniecki, Editor-in-Chief), ISBN: 978-1-56347-054-3, Washington DC.

Kanoh, H.; Tzafestas, S.; Lee, H. G. & Kalat, J. (1986). Modelling and control of flexible robot arms, *Proceedings of the 25th Conference on Decision and Control*, pp. 1866-1870, Athens, December 1986, Greece.

Kanoh, H. & Lee, H. G. (1985). Vibration control of a one-link flexible arm, *Proceedings of the 24th Conference on Decision and Control*, pp. 1172-1177, Ft. Lauderdale, December 1985, FL, USA.

Kapur, K. K. (1966). Vibrations of a Timoshenko beam, using a finite element approach. *Journal of the Acoustical Society of America*, Vol. 40, No. 5, (November 1966) 1058–1063, ISSN: 0001-4966.

Kolberg, U. A. (1987). General mixed finite element for Timoshenko beams. *Communications in Applied Numerical Methods*, Vol. 3, No. 2, (March-April 1987) 109–114, ISSN: 0748-8025.

Krishnan, H. & Vidyasagar, M. (1988). Control of a single-link flexible beam using a Hankel-norm-based reduced order model, *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 9-14, ISBN: 0-8186-0852-8, Philadelphia, April 1988, PA, USA.

Loudini, M.; Boukhetala, D.; Tadjine, M.; & Boumehdi, M. A. (2006). Application of Timoshenko Beam Theory for Deriving Motion Equations of a Lightweight Elastic Link Robot Manipulator. *International Journal of Automation, Robotics and Autonomous Systems*, Vol. 5, No. 2, (2006) 11-18, ISSN 1687-4811.

Loudini, M.; Boukhetala, D. & Tadjine, M. (2007a). Comprehensive Mathematical Modelling of a Transversely Vibrating Flexible Link Robot Manipulator Carrying a Tip Payload. *International Journal of Applied Mechanics and Engineering*, Vol. 12, No. 1, (2007) 67-83, ISSN 1425-1655.

Loudini, M.; Boukhetala, D. & Tadjine, M. (2007b). Comprehensive mathematical modelling of a lightweight flexible link robot manipulator. *International Journal of Modelling, Identification and Control*, Vol.2, No. 4, (December 2007) 313-321, ISSN: 1746-6172.

Macchelli, A. & Melchiorri, C. (2004). Modeling and control of the Timoshenko beam. The distributed port hamiltonian approach. *SIAM Journal on Control and Optimization*, Vol. 43, No. 2, (March-April 2004) 743–767, ISSN: 0363-0129.

Meirovitch, L. (1986) *Elements of Vibration Analysis*, McGraw-Hill, ISBN: 978-0-070-41342-9, New York, USA.

Moallem, M.; Patel R. V. & Khorasani, K. (2000) *Flexible-link Robot Manipulators : Control Techniques and Structural Design*, Springer-Verlag, ISBN 1-85233-333-2, London.

Morris, A. S. & Madani, A. (1996). Inclusion of shear deformation term to improve accuracy in flexible-link robot modeling. *Mechatronics*, Vol. 6, No. 6, (September 1996) 631-647, ISSN: 0957-4158.

Oguamanam, D. C. D. & Heppler, G. R. (1996). The effect of rotating speed on the flexural vibration of a Timoshenko beam, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2438-2443, ISBN: 0-7803-2988-0, Minneapolis, April 1996, MN, USA.

Ortner, N. & Wagner, P. (1996). Solution of the initial-boundary value problem for the simply supported semi-finite Timoshenko beam. *Journal of Elasticity*, Vol. 42, No. 3, (March 1996) 217-241, ISSN: 0374-3535.

Qi, X. & Chen, G. (1992). Mathematical modeling for kinematics and dynamics of certain single flexible-link robot arms, *Proceedings of the IEEE Conference on Control Applications*, pp. 288-293, ISBN: 0-7803-0047-5, Dayton, September 1992, OH, USA.

Rayleigh, J. W. S. (2003). *The Theory of Sound*, Two volumes, Dover Publications Inc., ISBNs: 978-0-486-60292-9 & 978-0-486-60293-6, New York.

Robinett III, R. D.; Dohrmann, C.; Eisler, G. R.; Feddema, J.; Parker, G. G.; Wilson, D. G. & Stokes, D. (2002). *Flexible Robot Dynamics and Controls*, IFSR International Series on Systems Science and Engineering, Vol. 19, Kluwer Academic/Plenum Publishers, ISBN: 0-306-46724-0, New York, USA.

Salarieh, H. & Ghorashi, M. (2006). Free vibration of Timoshenko beam with finite mass rigid tip load and flexural–torsional coupling. *International Journal of Mechanical Sciences*, Vol. 48, No. 7, (July 2006) 763–779, ISSN: 0020-7403.

De Silva, G. W. (1976). Dynamic beam model with internal damping, rotatory inertia and shear deformation. *AIAA Journal*, Vol. 14, No. 5, (May 1976) 676–680, ISSN: 0001-1452.

Sooraksa, P. & Chen, G. (1998). Mathematical modeling and fuzzy control of a flexible-link robot arm. *Mathematical and Computer Modelling*, Vol. 27, No. 6, (March 1998) 73-93, ISSN: 0895-7177.

Stephen, N. G. (1982). The second frequency spectrum of Timoshenko beams. *Journal of Sound and Vibration*, Vol. 80, No. 4, (February 1982) 578-582, ISSN: 0022-460X.

Stephen, N. G. (2006). The second spectrum of Timoshenko beam theory. *Journal of Sound and Vibration*, Vol. 292, No. 1-2, (April 2006) 372-389, ISSN: 0022-460X.

Timoshenko, S. P. (1921). On the correction for shear of the differential equation for transverse vibrations of prismatic bars. *Philosophical Magazine Series 6*, Vol. 41, No. 245, (1921) 744-746, ISSN: 1941-5982.

Timoshenko, S. P. (1922). On the transverse vibrations of bars of uniform cross section. *Philosophical Magazine Series 6*, Vol. 43, No. 253, (1922) 125-131, ISSN: 1941-5982.

Timoshenko, S.; Young, D. H. & Weaver Jr., W. (1974) *Vibration Problems in Engineering*, Wiley, ISBN: 978-047-187-315-0, New York.

Tokhi, M. O. & Azad, A. K. M. (2008). *Flexible Robot Manipulators: Modeling, Simulation and Control*. Control Engineering Series 68, The Institution of Engineering and Technology (IET), ISBN: 978-0-86341-448-0, London, United Kingdom.

Trail-Nash, P. W. & Collar, A. R. (1953). The effects of shear flexibility and rotary inertia on the bending vibrations of beams. *Quarterly Journal of Mechanics and Applied Mathematics*, Vol. 6, No. 2, (March 1953) 186-222, ISSN: 0033-5614.

Wang, F. –Y. & Guan, G. (1994). Influences of rotary inertia, shear and loading on vibrations of flexible manipulators. *Journal of Sound and Vibration*, Vol. 171, No. 4, (April 1994) 433-452, ISSN: 0022-460X.

Wang, F.-Y. & Gao, Y. (2003). *Advanced Studies of Flexible Robotic Manipulators: Modeling, Design, Control and Applications*. Series in Intelligent Control and Intelligent Automation, Vol. 4, World Scientific, ISBN: 978-981-238-390-5, Singapore.

Wang, R. T. & Chou, T. H. (1998). Non-linear vibration of Timoshenko beam due to a moving force and the weight of beam. *Journal of Sound and Vibration*, Vol. 218, No. 1, (November 1998) pp. 117-131, ISSN: 0022-460X.

Yurkovich, Y. (1992). Flexibility Effects on Performance and Control. In: *Robot Control*, M. W. Spong, F. L. Lewis and C. T. Abdallah (Eds.), Part 8, (August 1992) 321-323, IEEE Press, ISBN: 978-078-030-404-8, New York.

Zener, C. M. (1965). *Elasticity and Anelasticity of Metals*, University of Chicago Press, 1st edition, 5th printing, Chicago, USA.

Nomenclature

A	link cross-section area
B	inertia matrix
C_D	viscoelastic material constant
D	dissipated energy
E	link Young's modulus of elasticity
F	vector of external forces
G	shear modulus
H	damping matrix
I	link moment of inertia
J_h	hub and rotor (actuator) total inertia
J_p	payload inertia
k	shear correction factor
K	stiffness matrix
K_D	Kelvin-Voigt damping coefficient
ℓ	link length
L	Lagrangian
M	bending moment
M_p	payload mass
n	mode number
N	vector of Coriolis and centrifugal forces
q	vector of generalized coordinates
S	shear force
t	time
T	kinetic energy
U	stored potential energy
$w(x,t)$	transverse deflection

x	coordinate along the beam
$\Phi_n(x)$	n th transverse mode shape
$\Psi_n(x)$	n th rotational mode shape
$a(x,t)$	angular position of a point of the deflected link
δ_n	n th modal amplitude
ε	strain
ν	normal stress
$\theta(t)$	angular position of the rotating X -axis
ρ	link uniform linear mass density
σ	shear angle
τ	actuator torque applied at the base of the link
ω_n	n th natural frequency of vibration
ω_{d_n}	n th damped natural frequency of vibration
$\gamma(x,t)$	rotation of cross-section about neutral axis
$\eta(t)$	rotating X -axis angular position
ξ_n	n th damping ratio.

Trajectory Control of RLED Robot Manipulators Using a New Type of Learning Rule

Hüseyin Canbolat
Mersin University
Turkey

1. Introduction

Rigid Link Electrically Driven (RLED) robot manipulators are used extensively in applications. For RLED manipulators, a hybrid adaptive-learning controller, which do not utilize the velocity measurements, is designed and proved that it can be made semi-global asymptotically stable (Canbolat et al., 1996). The learning part in that work (Canbolat et al., 1996) is based on the results given in (Messner et al. 1991). However (Messner et al., 1991) neglected the electrical dynamics and the velocity measurements are available. In (Canbolat et al., 1996), the system is designed through a high pass filter which produces the surrogates of velocity. Later in another work, (Kaneko & Horowitz, 1997) designed a similar controller for a robot manipulator using a velocity observer neglecting the electrical dynamics. The system in (Canbolat et al., 1996) had not been verified through simulation and experiments. Recently, (Uguz & Canbolat, 2006) published the simulation results of the controller proposed in (Canbolat et al., 1996) for a sinusoidal desired position. However, a typical desired position for a robotic application is not generally sinusoidal. Due to this, more general position vectors should be generated. A general task requires a smooth trajectory, which starts from an initial position to a final position and repeats this over and over again. Such a desired trajectory can be generated in several ways (Fu et al., 1987). In the simulation of the system in (Canbolat et al., 1996), desired functions should satisfy the certain specifications. For this purpose, the polynomial method given in (Fu et al., 1987) is slightly modified in order to accommodate with the requirements of the controller. The modifications are necessary due to the continuous third derivative or jerk requirement in (Canbolat et al., 1996). Here, we also proposed other methods, which utilize transcendental functions. Transcendental function methods give a trajectory that can be continuously differentiable up to any order.

Learning control law is usually used for repetitive tasks in which a certain task should be repeated in each cycle. Indeed, the adaptive and learning control schemes are very similar, since both strategies are based on the estimation of unknown system dynamics. However, the learning control philosophy tries to estimate the unknown time functions instead of estimating the unknown constant parameters of the system as in the adaptive control setup. The aim of the learning control is to improve the tracking performance of the manipulator at

each cycle using the error information obtained during the previous cycles. Thus the tracking of a desired trajectory is expected to improve in a period of the specified task comparing the results in the previous period (Arimoto, 1986; Messner et al., 1991). The control law is adjusted using the tracking error obtained at previous trials. The controller is expected to "learn" the unknown dynamics and make the tracking error goes to zero (Messner et al., 1991). The research on the design of adaptive control laws which tracks a desired trajectory asymptotically for rigid link robot manipulators has been conducted for years. The parametric uncertainties for a given system are inevitable for precise control. The uncertainties considerably affect the control performance of the system. Adaptive controllers, which updates the parameter estimates according to an adaptive update rule, tries to achieve the required specifications in the presence of parametric uncertainties (Lewis et al., 1993). In the case of robot manipulators, the control should be nonlinear due to the nonlinear nature of robot manipulator dynamics. Adaptive control law requires the linear parameterization of the system dynamics (Sadegh et al., 1990). However, the learning controller is generally used for periodic desired trajectories (Arimoto et al., 1985; Bondi et al., 1988; Horowitz et al., 1991; Kaneko & Horowitz, 1992; Kawamura et al., 1988; Kuc et al., 1992; Qu et al., 1993). (Messner et al., 1991) proposed a new learning algorithm. The algorithm is based on the selection of a Hilbert-Schmidt kernel. The uncertainties are modeled as an integral equation, which includes the multiplication of the kernel and a function that represents the system uncertainties. The learning update rule is based on the estimation of the system uncertainties via an update rule for the unknown function in the integral equation in terms of the known system variables. The controller makes the system follow the desired trajectory asymptotically (Canbolat et al., 1996).
The simulation of the learning control scheme (Canbolat et al., 1996) could not be achieved due to the partial derivatives of the control law with respect to the second time variable created by the Hilbert-Schmidt kernel. The two-time variables make the system complicated to simulate using traditional simulation packages, such as MATLAB® Simulink and SIMNON. In order to simulate the system in Simulink, the second time variable is considered to be discrete. Therefore, only the samples of the variables at specified locations on the second axis are estimated instead of a continuum of time. However, this process does not result a discrete-time system. Instead, the process results a higher order nonlinear continuous system through the state variables created due to the time-dynamic nature of the control law in both independent time variables, that is, the controller equations include partial derivatives with respect to both time variables. Since time is not discretized the resulting variables on the second axis has still continuous dynamics with respect to the real time.
In this work, the hybrid adaptive/learning controller proposed by (Canbolat et al., 1996) is simulated. The controller does not need the exact parameter values of the robot manipulator. The parameters of the electrical subsystem are updated according to an adaptive rule; while the uncertainties in the mechanical subsystem are compensated via learning term presented by (Messner et al., 1991) and (Canbolat et al., 1996). The controller was designed using a back-stepping technique and follows the desired trajectory asymptotically. The system used in the simulation is a rigid-link electrically driven (RLED) two-link planar robot manipulator, which is actuated by brushed DC (BDC) motors. The controller does not use the link velocities and compensates the electrical subsystem parameter uncertainties using an adaptive update law, while compensating the

uncertainties in the mechanical subsystem via a learning law. The controller is a partial state feedback controller which uses only the link positions and the actuator currents and forces the system follow the desired trajectory asymptotically (Canbolat et al., 1996). The controller is simulated using the MATLAB® SIMULINK software package. The results of the simulation shows that the proposed controller provides the semi-global asymptotic trajectory following.

Robot manipulators are implemented in various types like rectangular, cylindrical, spherical, revolute and horizontal joints to achieve the desired movements. From an industrial point of view, the Selective Compliance Articulated Robotic Arm (SCARA) type manipulators are utilized in the processes such as pick-and-place, painting, brushing, and peg-in-hole. In general, a SCARA manipulator has four degrees of freedom. Shoulder, elbow and wrist arms are controlled by servo motors while the fourth movement is realized pneumatically.

Various types of robot manipulators are designed according to the required movement types but the design of the controller is as important as the design of the mechanical parts. Several studies are available in the literature related to the design of controllers for robot manipulators employing classical proportional-integral-differential (PID) (Das & Dulger, 2005), adaptive (Queiroz et al., 1997; Kaneko & Horowitz, 1997), learning (Canbolat et al., 1996; Horowitz et al., 1991; Messner et al., 1991) artificial intelligence (Golnazarian, 1995; Jungbeck & Madrid, 2001) and fuzzy logic algorithms (Lewis et al., 1993).

Here, we describe the design of the hybrid adaptive repetitive controllers given in (Canbolat et al., 1996) and (Horowitz et al., 1991) and generate desired position functions, which satisfy the specifications given. However, the computation of derivatives requires the manipulation of highly nonlinear transcendental functions. The physical limitations of the robot manipulator are not considered in generation of desired trajectories. For a thorough position function the physical properties should be considered, such as, maximum velocity, acceleration, and jerk. Then a delayed hybrid adaptive repetitive controller (Sahin & Canbolat, 2007) is designed based on the method of (Horowitz et al., 1991). Also, the controllers are applied to a Serpent-1 model SCARA manipulator used in (Das & Dulger, 2005) in a simulation environment for a desired path generated according to the specifications of the hybrid adaptive-learning controller. Then, the performance of the robot with classical PD controller, learning based controller without electrical dynamics and adaptive/learning based hybrid controller are examined by means of simulations. Based on the simulation results, the performance of learning based controllers and classical PD controller is discussed.

2. Control Objective

The objective of this work is to develop a repetitive link position tracking controller for rigid link electrically driven (RLED) robot manipulators driven by brushed DC motors. The controller compensates for the effects of actuator dynamics. Furthermore, it uses only the link position and motor current measurements while compensating for the parametric uncertainty throughout the entire mechanical system and eliminating the link velocity measurements.

To facilitate the control law development, the position tracking error is defined as

$$e=q_d-q. \tag{1}$$

The parametric uncertainties of the mechanical subsystem are included in $c(\gamma)$ of (11) and the unknown electrical subsystem parameters are represented by the following vector

$$\theta_e = \left[\theta_{e1}^T, \theta_{e2}^T, \ \dots \ , \theta_{en}^T\right]^T \in \Re^{3n} \tag{2}$$

where

$$\theta_{ei} = \left[L_i, R_i, K_{bi}\right]^T \in \Re^3 \tag{3}$$

in which L_i, R_i, and K_{bi} are the diagonal elements of electrical subsystem matrices L, R, and K_b, respectively. The true values of these parameters are not known except it is assumed that their upper and lower bounds are known. Whenever these upper and lower bounds are referred in the text, we will denote upper and lower bounds of a parameter matrix with the subscripts upper and lower, respectively. For example, $L_{lower} \le \lambda_{\min}(L)$ denotes the lower bound for the matrix L, where $\lambda_{\min}(L)$ is the minimum eigenvalue of the matrix L.

A dynamic estimate $\hat{\theta}_e \in \Re^{3n}$ is used for θ_e. The parameter estimation error, $\tilde{\theta}_e$ is defined as follows

$$\tilde{\theta}_e = \theta_e - \hat{\theta}_e\,. \tag{4}$$

In the following section, we will give the details of the control design. The controller will be a partial state feedback controller in the sense that it will not utilize link velocity measurements to compensate for parametric uncertainties in the system. It is shown that the designed controller guarantees the semi-global asymptotic link position tracking. The system performance is simulated through a computer code. The code is written for both hybrid adaptive-learning controllers for BDC RLED robot manipulators and for the learning controller designed in (Messner et al., 1991). The results of the simulations show that the controller performs well in terms of error is below some certain value. However, the error does not become zero, but it has some average value. This is because of the complexity of the control law and the minimum information used to achieve the control goal.

3. System Model

3.1 Robot and Actuator Dynamics

The dynamics of an n-link robot manipulator electrically driven by brushed DC (BDC) motors can be expressed as follows:

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F_d\dot{q} = K_\tau\ I \quad \text{(mechanical subsystem)} \tag{5}$$

$$L\dot{I} + RI + K_b\dot{q} = v \qquad \text{(electrical subsystem)} \tag{6}$$

where,

$q, \dot{q}, \ddot{q}$	:nx1 link position, velocity and acceleration vectors, respectively,
$M(q)$	:nxn symmetric, positive definite inertia matrix,
$V_m(q,\dot{q})$	:nxn matrix of centripetal and Coriolis terms,
F_d	:nxn constant, diagonal, dynamic friction matrix,
$G(q)$	:nx1 gravitational effects vector,

τ	:nx1 torque vector,
L	:nxn diagonal inductance matrix,
R	:nxn diagonal resistance matrix,
K_b	:nxn diagonal back-emf matrix,
K_τ	:nxn diagonal torque coefficients matrix, and
v	:nx1 motor input voltages vector.

The periodic desired trajectory $q_d(t)$ and its time derivatives up to 3rd order should be continuous and bounded (Canbolat et al., 1996).
The following properties of robot dynamics were utilized in the stability analysis of the controller:

1. For any given vector, $x(t)$, the inertia matrix, $M(q)$, satisfies the following inequality:

$$M_1 \|x\|^2 \le x^T M(q) x \le M_2 \|x\|^2 \tag{7}$$

where M_1 and M_2 are known positive constants that depend on the mass properties of the specific robot for which the controller is designed.

2. The matrix $\dot{M}(q) - 2V_m(q,\dot{q})$ is skew symmetric, that is, for any given vector x, we have

$$x^T \left(\dot{M}(q) - 2V_m(q,\dot{q}) \right) x = 0 \tag{8}$$

3. The Coriolis-centripetal matrix V_m is bounded as

$$\|V_m(q,\dot{q})\|_{i\infty} \le \zeta_c \|\dot{q}\| \tag{9}$$

where ζ_c is a known positive constant.

4. The left-hand side of (5) can be written in terms of the desired trajectory as

$$w(t) = M(q_d)\ddot{q}_d + V_m(q_{d,}\dot{q}_d)\dot{q}_d + G(q_d) + F_d\dot{q}_d \,. \tag{10}$$

Since the desired trajectories $q_d, \dot{q}_d, \ddot{q}_d$ are periodic with the period T, $w(t)$ of (10) is also periodic. $w(t)$, can be expressed as a linear integral equation as shown by (Horowitz et al., 1991). That is, $w(t)$ can be expressed as follows

$$w(t) = \int_0^T K(t,\gamma) c(\gamma) d\gamma \tag{11}$$

where $K(t,\gamma)$ is a *known* Hilbert-Schmidt kernel and $c(\gamma)$ is an *unknown* influence function. Note that t and γ are independent variables.

5. (Horowitz et al., 1991) used the kernel of the form

$$K(t,\gamma) = f_0 + \sum_{m=1}^{\infty}\left(f_m \cos(2\pi mt/T)\cos(2\pi m\gamma/T) + d_m \sin(2\pi mt/T)\sin(2\pi m\gamma/T)\right), \tag{12}$$

where f_i and d_i's are scalar constants, which satisfy the conditions,

$Dm^{-2} < | f_m |$ and $Dm^{-2} < | d_m |$ for all $m>N$ with D, N are constants.

If the kernel of the form given in (12) is utilized, then

$$\int_0^T \|c(\gamma)\|^2 \, d\gamma < \mu \tag{13}$$

where μ is a positive constant.

Consider the following kernel, which is a Gaussian distribution function, given by

$$K(t,\gamma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(t-\gamma)^2}{2\sigma^2}\right) \tag{14}$$

where σ is a positive design constant. This function satisfies the conditions given in (12) (Horowitz et al., 1991).

If the kernel defined in (13) is used, then the following relations can be shown:

$$\sup_{t\in[0,T]} \int_0^T K^2(t,\gamma)d\gamma = \kappa < \infty \tag{15}$$

$$\sup_{t\in[0,T]} \int_0^T \left(\frac{\partial}{\partial t}K(t,\gamma)\right)^2 d\gamma = \kappa_d < \infty, \tag{16}$$

where κ and κ_d are positive constants.

3.2 Position Tracking Controller

To achieve the control objective, the methods proposed by (Burg et al., 1996) and (Horowitz et al., 1991) are combined. The design procedure can be summarized as: i) We use a pseudo-velocity filter to generate the signals for use as link velocity, ii) since the developed torque is a function of motor currents, a desired current signal is designed to force the link position to track the desired trajectory (backstepping) and iii) the voltage control input is designed to ensure the motor currents tracks the desired current.

Using the position tracking error defined in (1), the following high pass filter is designed to obtain a velocity dependent signal e_f:

$$\begin{aligned} \dot{p} &= -(k+1)p + (k^2+1)e \\ e_f &= -ke + p \\ p(0) &= ke(0) \end{aligned} \tag{17}$$

where k is a positive gain constant and the auxiliary signal p is used to get two implementable equations for the filter.

The filtered tracking error is defined as follows

$$\eta = \dot{e} + e + e_f . \tag{18}$$

Note that, the filtered tracking error, η, cannot be measured, since the link velocities cannot be measured. Based on the dynamics of η, the auxiliary variable $w_1(t)$ is defined as

$$w_1(t) = \int_0^T K(t,\gamma)c_x(\gamma)d\gamma = K_\tau^{-1}\int_0^T K(t,\gamma)c(\gamma)d\gamma \tag{19}$$

Note that the uncertainties in K_τ is now included in $w_1(t)$. The desired current, I_d, is designed to force the filtered tracking error, η, to zero.

The desired current, I_d, is defined as

$$I_d = \hat{w}_1(t) - ke_f + e \tag{20}$$

and the current tracking error, η_I, is defined as

$$\eta_I = I_d - I \tag{21}$$

where $\hat{w}_1(t)$ is the estimate of $w_1(t)$ and is defined as

$$\hat{w}_1(t) = \int_0^T K(t,\gamma)\hat{c}_x(t,\gamma)d\gamma \tag{22}$$

where $\hat{c}_x(t,\gamma) \in \Re^n$ is an estimate of $c_x(\gamma)$. $\hat{c}_x(t,\gamma)$ is updated according to the following rule

$$\begin{aligned}\hat{c}_x(t,\gamma) = &\int_0^t \Big(K(\sigma,\gamma)K_L\big(e(\sigma)+e_f(\sigma)\big)\Big)d\sigma \\ &+K(t,\gamma)K_L e(t) - K(0,\gamma)K_L e(0) \\ &-\int_0^t \left(\frac{\partial}{\partial\sigma}K(\sigma,\gamma)\right)K_L e(\sigma)d\sigma.\end{aligned} \tag{23}$$

where K_L, is an nxn diagonal, positive definite gain matrix.

The electrical parameter regression matrix is defined as

$$Y_e\theta_e = Lw_2 + (k-1)Le_f - kLe + RI + K_b(\dot{q}_d + e + e_f) \tag{24}$$

where

$$w_2 = \int_0^T \frac{\partial K(t,\gamma)}{\partial t} \hat{c}_x(t,\gamma) d\gamma \,. \tag{25}$$

Based on the current tracking error dynamics the voltage control input is designed as

$$v = Y_e \hat{\theta}_e + \left(\|K_L\|_{i2}^2 \kappa^2 k_{n1} + k^4 k_{n2} + k_{n3} + k_{n4} + k_{n5} + 1 \right) \eta_I \tag{26}$$

where $\| K_L \|_{i2}$ denotes the induced-2 norm of the matrix K_L and the k_{ni} (i=1,2,...,5) and κ are positive control gains, and the adaptive electrical parameter update rule is defined as

$$\dot{\hat{\theta}}_e = \Gamma_e Y_e^T \eta_I \tag{27}$$

with $\Gamma_e \in \Re^{3n\text{x}3n}$ is a positive definite, diagonal adaptive gain matrix and the electrical regression matrix $Y_e \in \Re^{3n\text{x}3n}$ is defined as

$$Y_e = blockdiag \left\{ \begin{bmatrix} w_{2i} + (k-1)e_{fi} - (k+1)e_i \\ I_i \\ \dot{q}_{di} + e_i + e_{fi} \end{bmatrix}^T \right\} \tag{28}$$

where w_{2i} is the ith element of w_2 defined in (25).

The following theorem can be stated for the tracking performance of the proposed controller (Canbolat et al., 1996).

Theorem 1: If the control gains k_{ni} and k_n satisfy the following conditions

$$k_{ni} > 3L_{\max}^2 + K_{\tau\max}^2 + K_{b\max}^2 \tag{29}$$

$$k_n \geq \frac{1}{\lambda_3} \left(\frac{\lambda_2}{\lambda_1} \|x(0)\|^2 + 1 \right) \tag{30}$$

where

$$\lambda_1 = \min \left\{ K_{\tau lower}, m_1, L_{lower}, \lambda_{\min}(\Gamma_e^{-1}), K_{\tau lower} \lambda_{\min}(K_L^{-1}) \right\} \tag{31}$$

$$\lambda_2 = \max \left\{ K_{\tau upper}, m_2, L_{upper}, \lambda_{\max}(\Gamma_e^{-1}), K_{\tau upper} \lambda_{\max}(K_L^{-1}) \right\} \tag{32}$$

$$\lambda_3 = \min \left\{ K_{\tau lower}, 1 - L_{upper}^2 \left(\frac{1}{k_{n1}} + \frac{1}{k_{n2}} + \frac{1}{k_{n3}} \right) - \frac{K_{bupper}^2}{k_{n4}} - \frac{K_{\tau upper}^2}{k_{n5}} \right\} \tag{33}$$

$$x = \begin{bmatrix} e^T & e_f^T & \eta^T & \eta_I^T & \tilde{\theta}_e^T & \int_0^T \tilde{c}_x^T(t,\gamma) d\gamma \end{bmatrix}^T \in \Re^{8n}, \tag{34}$$

then

$$\lim_{t\to\infty}\|e(t)\| = 0. \tag{35}$$

In the above equations, $(.)_{lower}$ and $(.)_{upper}$ denotes the known lower and upper bounds for the eigenvalues of the corresponding unknown parameter matrices, respectively. Similarly, $\lambda_{\min}(.)$ and $\lambda_{\max}(.)$ denotes the minimum and maximum eigenvalues of the matrix in parentheses, respectively.

The proof of this theorem can be found in (Canbolat et al., 1996). For the sake of brevity, the proof is omitted here.

3.3 Delayed Learning Rule

We define the following delayed update rule for $\hat{w}_1(t)$ in (22) and $w_2(t)$ in (25) similar to (Messner et al., 1991):

$$\hat{w}_1(t) = \int_0^T K(t,\gamma)\hat{c}_x^k(\gamma)d\gamma \quad kT \le t < (k+1)T \tag{36}$$

$$w_2(t) = \int_0^T \left\{\frac{\partial}{\partial t}K(t,\gamma)\right\}\hat{c}_x^k(\gamma)d\gamma \quad kT \le t < (k+1)T \tag{37}$$

where $\hat{c}_x^k(\gamma)$ is defined as

$$\begin{aligned}\hat{c}_x^k(\gamma) = \hat{c}_x^{k-1}(\gamma) + K(kT,\gamma)K_L e(kT) \\ + \int_{kT-T}^{kT} K(\sigma,\gamma)K_L\left[e(\sigma)+e_f(\sigma)\right]d\sigma \\ \int_{kT-T}^{kT}\left\{\frac{\partial}{\partial\sigma}K(\sigma,\gamma)\right\}K_L e(\sigma)d\sigma,\end{aligned} \tag{38}$$

with $\hat{c}_x^0(\gamma) = -K(0,\gamma)K_L e(0)$. Note that the form of (22) and (25) are not changed except the estimate of $\hat{c}_x(t,\gamma)$ defined in (23) is replaced with $\hat{c}_x^k(\gamma)$. Similarly, all other equations are same. One can use the same equations for the controller by changing (22), (23) and (25) with (36), (38) and (37), respectively. This definition aims the reduction of computational burden for real time applications. One can show that the controller with the delayed learning rule of (38) is asymptotically stable using the arguments used by (Messner et al., 1991) and (Canbolat et al., 1996).

3.4 Generation of Desired Trajectories

A proper desired trajectory should be generated for the proposed controller, for performance evaluation. A position function for a robot manipulator is a smooth function that starts from a certain initial position and ends at a final position. Generally, the acceleration and velocity are required to be continuous and smooth. However, in our case the desired trajectory should be continuously differentiable up to the third order. There are several methods to generate the desired trajectories. One common method is to separate the trajectory into three main parts (initial-lift off (IL), lift off-set down (LS), and set down-final (SF)) and impose the continuity conditions at the boundaries. The coefficients of the polynomials are to be solved according to the imposed conditions. (Fig. 1).

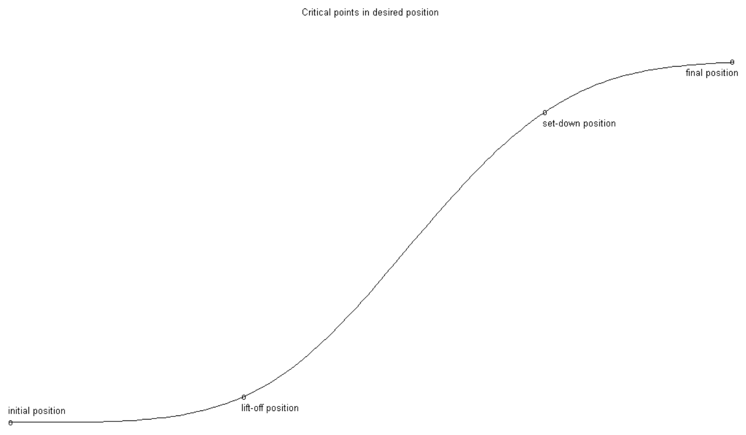


Fig. 1. Critical points for a robot end-effector position

In this section, we propose several methods to determine the desired trajectories. Polynomial methods are the modifications of 3-5-3 and 4-3-4 methods given in (Fu et al., 1987). The naming of the method is based on the degrees of the polynomials, which are valid for the IL, LS and SF parts. Since these methods generate functions continuously differentiable up to the second order, we should have modified the method to generate functions with continuous third time derivatives. The modification is done by increasing the degrees of the polynomials. Following the same convention, the modified methods are named as 4-6-4 and 5-4-5. Detailed formulae can be found in (Fu et al., 1987).
Formulation is carried as in (Fu et al., 1987). Let t_0, t_1, t_2, t_3 be the time boundaries for subtrajectories IL, LS and SF (Fig. 2). Let us assign the numbers 1, 2, 3, 4, 5, and 6 for forward IL, LS, SF and backward IL, LS, SF subintervals, respectively, and define the following dimensionless variable τ for each subinterval as follows

$$\tau = \frac{t - t_{i-1}}{t_i - t_{i-1}} \tag{39}$$

for $t \in [t_{i-1}, t_i]$ and i=1, 2, …, 6. Note that, in a given subinterval $[t_{i-1}, t_i]$ $\tau \in [0,1]$. With the definition given in (39) each subtrajectory can be defined on [0, 1]. Boundary conditions become the conditions at τ=0 and τ=1.

Let h_i be the polynomial in the ith subinterval. The first and kth derivative of h_i can be found as

$$\frac{dh_i}{dt} = \frac{dh_i}{d\tau}\frac{d\tau}{dt} = \frac{1}{t_i - t_{i-1}}\frac{dh_i}{d\tau} = A_i \frac{dh_i}{d\tau}$$
$$\frac{d^k h_i}{dt^k} = A_i^k \frac{d^k h_i}{d\tau^k} \tag{40}$$

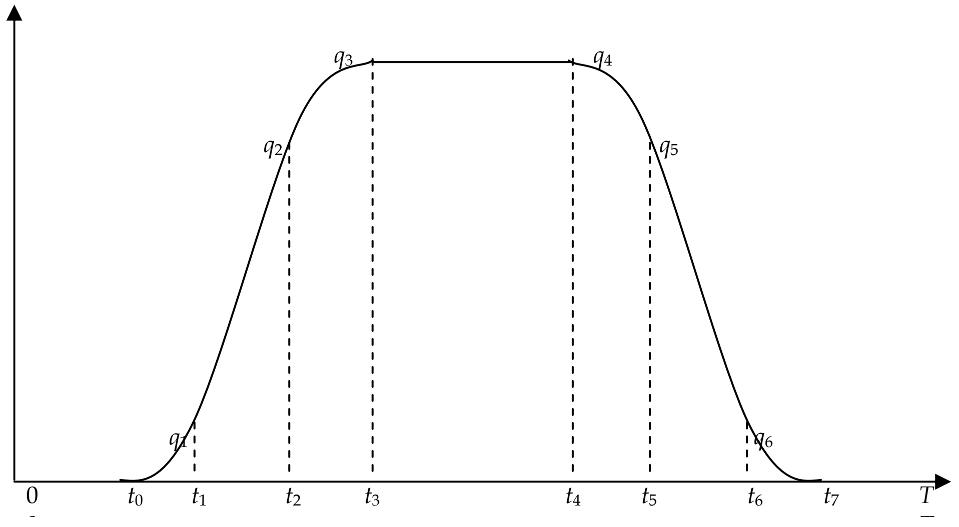


Fig. 2. Critical points of a desired trajectory

where $A_i = \dfrac{1}{t_i - t_{i-1}}$.

Since both methods use polynomials of maximum 6th degree in each subinterval, one can write the following general expressions for h_i and its derivatives:

$$\begin{aligned}
h_i(\tau) &= a_{i6}\tau^6 + a_{i5}\tau^5 + a_{i4}\tau^4 + a_{i3}\tau^3 + a_{i2}\tau^2 + a_{i1}\tau + a_{i0} \\
h_i'(\tau) &= \frac{1}{t_i - t_{i-1}}\left(6a_{i6}\tau^5 + 5a_{i5}\tau^4 + 4a_{i4}\tau^3 + 3a_{i3}\tau^2 + 2a_{i2}\tau + a_{i1}\right) \\
h_i''(\tau) &= \frac{1}{\left(t_i - t_{i-1}\right)^2}\left(30a_{i6}\tau^4 + 20a_{i5}\tau^3 + 12a_{i4}\tau^2 + 6a_{i3}\tau + 2a_{i2}\right) \\
h_i'''(\tau) &= \frac{1}{\left(t_i - t_{i-1}\right)^3}\left(120a_{i6}\tau^3 + 60a_{i5}\tau^2 + 24a_{i4}\tau + 6a_{i3}\right).
\end{aligned} \tag{41}$$

Let q_i, q_l, q_s ve q_f be the positions at the initial, lift-off, set-down and final positions, respectively (Fig. 1). Each polynomial should satisfy the following boundary conditions:

$$\begin{aligned}
h_i(1) &= h_{i+1}(0) \\
h_i'(1) &= h_{i+1}'(0) \\
h_i''(1) &= h_{i+1}''(0) \\
h_i'''(1) &= h_{i+1}'''(0)
\end{aligned} \tag{42}$$

where primes denote the derivative with respect to time. In (42), i should be 1 or 2. (42) creates 8 equations for the coefficients. At the initial and final positions, the following conditions should be satisfied:

$$\begin{aligned}
& h_1(0) = q_b \\
& h_1'(0) = h_1''(0) = h_1'''(0) = 0 \\
& h_3(1) = q_f \\
& h_3'(1) = h_3''(1) = h_3'''(1) = 0
\end{aligned} \tag{43}$$

From (43) we have another set of 8 equations. Thus we have 16 equations for 17 unknown coefficients. In order to have a unique solution, one can use the position at the lift-off or set-down positions. Using the lift-off position, we have the following equation:

$$h_1(1)=q_l \tag{44}$$

Since the desired trajectory is periodic, the manipulator should go back to the initial position at the end of the period. Due to this, the formulation given in (39-44) should be done twice for both reaching the final position and returning to the initial position. The forward and backward trajectories may not be symmetric. That is, we are free to select different time intervals and different lift-off and set-down positions. We can even use different methods in the generation of forward and backward paths. Typically, symmetrical trajectories are easy to use in applications.

There are 8 critical points in a period (Fig. 2). The time instants t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7 and the corresponding positions q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7 are critical points of a desired trajectory. The first 4 points q_0, q_1, q_2, q_3 correspond to initial, lift-off, set-down and final positions of the forward path, and the last 4 points q_4, q_5, q_6, q_7 correspond to initial, lift-off, set-down and final positions of the backward path, respectively. The equalities,

$$q_0=q_7,\ q_3=q_4, \tag{45}$$

should be satisfied for a periodic trajectory. For a symmetrical trajectory, the following constraints in positions,

$$q_1=q_6,\ q_2=q_5, \tag{46}$$

and in time

$$t_0=T-t_7,\ t_1=T-t_6,\ t_2=T-t_5,\ t_3=T-t_4, \tag{47}$$

should be satisfied. For each desired trajectory, the following position values are used:

q_i=0, q_l=0.08, q_s=0.92, q_f=1,
in forward path and
q_i=1, q_l=0.92, q_s=0.08, q_f=0
in backward path. The corresponding time instants are assigned as follows:

t_0=0,1*T*;	t_1=0,15*T*;	t_2=0,25*T*;	t_3=0,3*T*
t_4=0,7*T*;	t_5=0,75*T*;	t_6=0,85*T*;	t_7=0,9*T*

All position values are in radians, since we used a two-link robot with revolute joints. It is possible to select closer lift-off and set-down points. However, in this case the subpolynomials may have maxima and minima inside their own subintervals. Typically, a robot path should be smooth and monotone increasing or decreasing.

<u>4-6-4 Method</u>

In this method, IL and SF polynomials are fourth order. Therefore,
$a_{i6}=a_{i5}=0$
in (41). Furthermore, the initial conditions in (43) requires
$a_{10}=q_b$, $a_{11}=a_{12}=a_{13}=0$
for forward path and
$a_{40}=q_b$, $a_{41}=a_{42}=a_{43}=0$
for backward path. The conditions in (42), (43) and (44) give the following matrix equality for the unknown coefficients:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 4A_1 & 0 & 0 & 0 & 0 & 0 & 0 & -A_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6A_1^2 & 0 & 0 & 0 & 0 & 0 & -A_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4A_1^3 & 0 & 0 & 0 & 0 & -A_2^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 6A_2 & 5A_2 & 4A_2 & 3A_2 & 2A_2 & A_2 & 0 & 0 & 0 & 0 & -A_3 & 0 \\ 0 & 0 & 15A_2^2 & 10A_2^2 & 6A_2^2 & 3A_2^2 & A_2^2 & 0 & 0 & 0 & 0 & -A_3^2 & 0 & 0 \\ 0 & 0 & 20A_2^3 & 10A_2^3 & 4A_2^3 & A_2^3 & 0 & 0 & 0 & 0 & -A_3^3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{14} \\ a_{10} \\ a_{26} \\ a_{25} \\ a_{24} \\ a_{23} \\ a_{22} \\ a_{21} \\ a_{20} \\ a_{34} \\ a_{33} \\ a_{32} \\ a_{31} \\ a_{30} \end{bmatrix} = \begin{bmatrix} q_i \\ q_l \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ q_f \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{48}$$

where A_i's are defined in (40). (48) is valid for both forward and backward paths. Solving (48) for forward and backward paths, we obtained the following solution (Fig. 3):

$$\begin{bmatrix} a_{14} & a_{44} \\ a_{10} & a_{40} \\ a_{26} & a_{56} \\ a_{25} & a_{55} \\ a_{24} & a_{54} \\ a_{23} & a_{53} \\ a_{22} & a_{52} \\ a_{21} & a_{51} \\ a_{20} & a_{50} \\ a_{34} & a_{64} \\ a_{33} & a_{63} \\ a_{32} & a_{62} \\ a_{31} & a_{61} \\ a_{30} & a_{60} \end{bmatrix} = \begin{bmatrix} 0.0800 & -0.0800 \\ 0.0000 & 1.0000 \\ -3.4695 & 3.4695 \\ 11.6716 & -11.6716 \\ -12.4098 & 12.4098 \\ 2.5600 & -2.5600 \\ 1.9200 & -1.9200 \\ 0.6400 & -0.6400 \\ 0.0800 & 0.9200 \\ -0.0077 & 0.0077 \\ 0.0309 & -0.0309 \\ -0.0463 & 0.0463 \\ 0.0309 & -0.0309 \\ 0.9923 & 0.0077 \end{bmatrix} \tag{49}$$

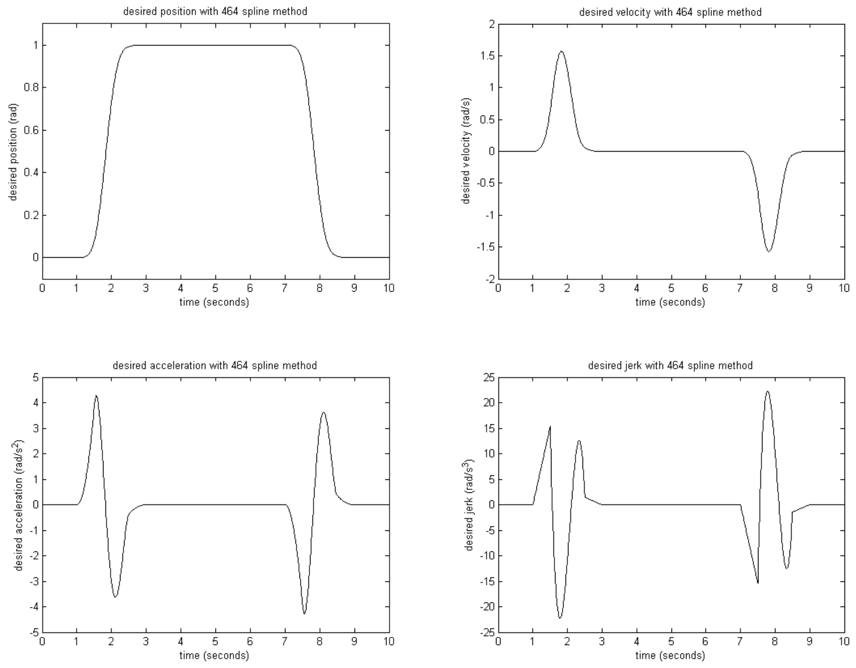


Fig. 3. Desired position and its derivatives with 4-6-4 spline method

5-4-5 Method

In this method, we should have

$a_{i6}=0$
for IL and SF subintervals in (41) and for the LS subinterval
$a_{i6}=a_{i5}=0$.
From the initial conditions in (43), we write
$a_{10}=q_b$, $a_{11}=a_{12}=a_{13}=0$
for forward path and
$a_{40}=q_b$, $a_{41}=a_{42}=a_{43}=0$
for backward path.
The conditions in (42), (43) and (44) give the following matrix equality for the unknown coefficients:

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5A_1 & 4A_1 & 0 & 0 & 0 & 0 & -A_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10A_1^2 & 6A_1^2 & 0 & 0 & 0 & -A_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10A_1^3 & 4A_1^3 & 0 & 0 & -A_2^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 4A_2 & 3A_2 & 2A_2 & A_2 & 0 & 0 & 0 & 0 & 0 & -A_3 & 0 \\ 0 & 0 & 0 & 6A_2^2 & 3A_2^2 & A_2^2 & 0 & 0 & 0 & 0 & 0 & -A_3^2 & 0 & 0 \\ 0 & 0 & 0 & 4A_2^3 & A_2^3 & 0 & 0 & 0 & 0 & 0 & -A_3^3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 6 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 4 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{15} \\ a_{14} \\ a_{10} \\ a_{24} \\ a_{23} \\ a_{22} \\ a_{21} \\ a_{20} \\ a_{35} \\ a_{34} \\ a_{33} \\ a_{32} \\ a_{31} \\ a_{30} \end{bmatrix} = \begin{bmatrix} q_i \\ q_l \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ q_f \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{50}$$

where A_i's are defined in (40). (50) is valid for both forward and backward paths. Solving (50) for forward and backward paths, we obtained the following solution (Fig. 4):

$$\begin{bmatrix} a_{15} & a_{45} \\ a_{14} & a_{44} \\ a_{10} & a_{40} \\ a_{24} & a_{54} \\ a_{23} & a_{53} \\ a_{22} & a_{52} \\ a_{21} & a_{51} \\ a_{20} & a_{50} \\ a_{35} & a_{65} \\ a_{34} & a_{64} \\ a_{33} & a_{63} \\ a_{32} & a_{62} \\ a_{31} & a_{61} \\ a_{30} & a_{60} \end{bmatrix} = \begin{bmatrix} -0.0644 & 0.0644 \\ 0.1444 & -0.1444 \\ 0.0000 & 1.0000 \\ -0.0381 & 0.0381 \\ -0.5299 & 0.5299 \\ 0.8900 & -0.8900 \\ 0.5113 & -0.5113 \\ 0.0800 & 0.9200 \\ -0.0720 & 0.0720 \\ 0.2013 & -0.2013 \\ -0.0853 & 0.0853 \\ -0.2320 & 0.2320 \\ 0.2747 & 0.2747 \\ 0.9133 & 0.0867 \end{bmatrix} \tag{51}$$

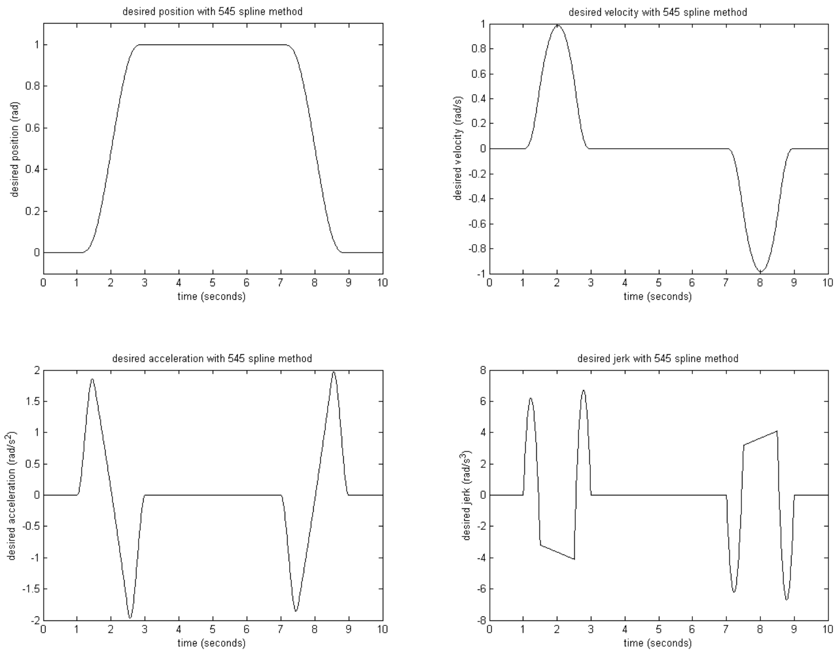


Fig. 4. Desired position and its derivatives with 5-4-5 spline method

Transcendental Function Methods

In these methods, an upper and lower bounded, monotone increasing or decreasing transcendental function, which is continuously differentiable up to the third order, is used. In order to obtain a periodic trajectory, the argument of the transcendental function should be a continuously differentiable periodic function up to the third order. Typically, a sinusoidal function can be selected as the argument. The advantage of using a transcendental function with a periodic argument function is that for each subinterval the same function is used. This reduces the formulation time of the trajectory. Also the trajectory function never has local maxima and minima in the subintervals, since they are monotone increasing or decreasing. However, it is not easy to determine the exact lift-off and set-down points. The derivatives of the function may become very complex, as in the hyperbolic tangent case. Fortunately, we need only the first order derivatives, because the controller uses only the desired trajectory and the desired velocity to compute the input functions, as in (24).

1.Hyberbolic Tangent Method

Let us use the hyperbolic tangent function for the trajectory. In this method, there is no need to consider the forward and backward subintervals of IL, LS, and SF. Instead, we use a hyperbolic tangent function with a continuously differentiable (at least up to the third order) periodic function as the argument. Indeed the method uses the fact that the hyperbolic tangent function can take values in the interval [-1, 1]. Same function is valid for all times and for each subinterval of the trajectory. However, it is not easy to determine the boundaries for lift-off and set-down positions. There is no general method to determine these points. In this method the desired trajectory is defined as:

$$h(t)=b[a+d\tanh(c\cos(\omega t))] \tag{52}$$

Where b is a weighting constant in radians, a is the constant that determines the initial position, c is the constant that determines the lift-off and set-down positions, d is the constant which determines the difference between the initial ($ba\text{–}bd$) and final ($ba+bd$) positions, ω is the angular frequency of the desired trajectory. Note that cosine function in the argument is continuously differentiable of any order. The determination method of the constant c is trial and error. Typically, c should be selected large enough so that the trajectory reaches to its final position and remains there for some time without subjecting excessive velocities and accelerations for a pick and place task (Fig. 5). However, a, b, and d can be determined according to the initial and final desired positions. One can easily find the velocity, acceleration and jerk functions by taking the successive derivatives of (52) as

$$v(t)=\frac{d}{dt}h(t)=-bcd\omega\sin(\omega t)\operatorname{sech}^2\left(c\cos(\omega t)\right) \tag{53}$$

$$a(t)=\frac{d}{dt}v(t)=-bcd\omega^2\operatorname{sech}^2\left(c\cos(\omega t)\right)\left[\cos(\omega t)+2c\tanh(c\cos(\omega t)\sin^2(\omega t)\right] \tag{54}$$

$$\begin{aligned} j(t)=\frac{d}{dt}a(t)=bcd\omega^3\sin(\omega t)\operatorname{sech}^2\left(c\cos(\omega t)\right)\big[&1-6c\cos(\omega t)\tanh(c\cos(\omega t) \\ &+2c^2\sin^2(\omega t)\left\{\operatorname{sech}^2\left(c\cos(\omega t)\right)-2\tanh^2\left(c\cos(\omega t)\right)\right\}\big]. \end{aligned} \tag{55}$$

The jerk expression given in (14) can also be written as follows

$$\begin{aligned} j(t)=\frac{d}{dt}a(t)=bcd\omega^3\sin(\omega t)\operatorname{sech}^2\left(c\cos(\omega t)\right)\big[&1-6c\cos(\omega t)\tanh(c\cos(\omega t) \\ &+2c^2\sin^2(\omega t)\left\{3\operatorname{sech}^2\left(c\cos(\omega t)\right)-2\right\}\big]. \end{aligned} \tag{56}$$

2.Error Function Method:

Here the trajectory is selected as the integral of Gaussian distribution function, which is known as the error function, defined as

$$\operatorname{erf}(t)=\frac{2}{\sigma\sqrt{\pi}}\int_0^t\exp\left(-\frac{\tau^{\,2}}{\sigma^2}\right)d\tau\,. \tag{57}$$

To get a continuous and periodic function, we use a sinusoidal argument as in hyperbolic tangent function. The following function is periodic, continuous and differentiable at least up to the third order:

$$h(t)=B\left[A-D\operatorname{erf}\left(C\cos(\omega t)\right)\right]. \tag{58}$$

The advantage of this function is that the derivatives are in terms of simple exponential and sinusoidal functions. Note that the function given in (58), has the initial value of zero, if one selects $A=D$.

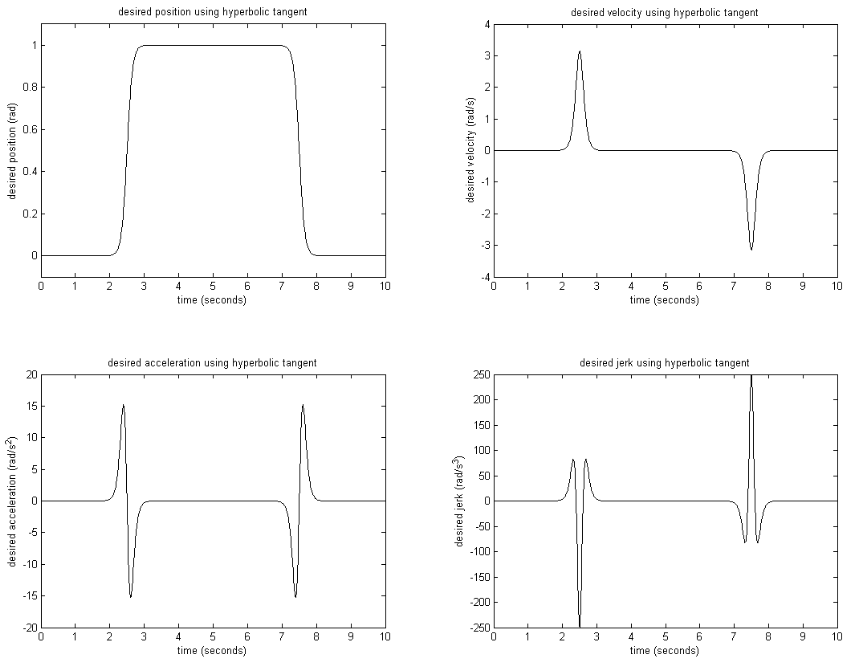


Fig. 5. Desired position with hyperbolic tangent function (c=10)

3.5 PD and Learning Controllers

A frequently used controller in control systems is the classical proportional-derivative (PD) controller (Das & Dulger, 2005). The main advantage of the PD controller is that it can easily be implemented on simple microcontroller architectures. On the other hand, the performance obtained from PD controllers is not satisfying for most of the sensitive applications. In this work, PD and learning controllers (Messner et al., 1991) are simulated along with the hybrid adaptive-learning controller (Canbolat et al. 1996) in order to compare the achieved performance. For this purpose, we repeat the main equations of learning controller below.

First, the main equation of the PD controller is

$$V_k(t) = K_p e(t) + K_d \frac{de(t)}{dt} \tag{59}$$

where $e(t)$ is the error function, K_p is the proportional control coefficient and K_d refers to the derivative control constant.

The learning algorithm proposed by (Messner et al., 1991) has more complex structure than a PD controller. The mechanical part of the robot dynamics can be rearranged as

$$M(q)\ddot{q} = T(t) - V_m(q,\dot{q})\dot{q} - G(q) - F_d(\dot{q}) \tag{60}$$

where $T(t)$ is the control torques applied to the joints and $q(t)=[q_1(t), q_2(t)]^T$ is the position of the manipulator, $\dot{q}$ and $\ddot{q}$ are nx1 vectors of joint velocities, and accelerations, respectively. The other terms are defined in (1). The following function, which includes the mechanical uncertainties, is defined

$$w_r(t) = M(q_d)\ddot{q}_d + V_m(q_d,\dot{q}_d)\dot{q}_d + G(q_d) \tag{61}$$

where $q_d(t)$ denotes the desired periodic trajectory vector of the robot links and the dots denote the differentiation with respect to time, t. Using these two equations, the following error dynamics and the desired compensation control law (DCCL) can be derived

$$\dot{e}(t) = f(t,e) + B(t,e)\left[w_r(t) - \hat{w}_r(t)\right] \tag{62}$$

$$T(t) = \hat{w}_r(t) + F_v e_v(t) + F_p e(t) + d_m(q,\dot{q}) + q_n(e_v) \tag{63}$$

where $e = \begin{bmatrix} e_p^T & e_v^T \end{bmatrix}^T$ and $\hat{w}_r(t)$ is the estimate of the influence function, $w_r(t)$, that compensates for mechanical uncertainties, F_v and F_p are PD gains, $e_v(t)$ is the reference velocity error vector, $e(t)$ is the position error vector, $d_m(q,\dot{q})$ is the friction compensation. $q_n(e_v)$ is the nonlinear compensation function. As it can be seen from (63), the repetitive-learning algorithm utilizes PD control but also uses $\hat{w}_r(t)$ to compensate for the uncertain parameters, thus providing "learning". The position error is defined as

$$e_p(t) = q_d(t) - q(t) \tag{64}$$

where $q_d(t)$ is the desired trajectory. The reference velocity error function is defined as

$$e_v(t) = \dot{e}_p(t) + \lambda e_p(t) \ , \tag{65}$$

where λ is a positive constant and the nonlinear compensation function is given as:

$$q_n(e_v) = \sigma \left|e_p\right|^2 \left|e_v\right| \tag{66}$$

The estimate of the uncertainty function $\hat{w}_r(t)$ is updated with following rules:

$$\hat{w}_r(t) = \int_0^T K(t,\tau)\hat{c}(t,\tau)d\tau \tag{67}$$

$$\frac{\partial \hat{c}(t,\tau)}{\partial t} = K(t,\tau)K_L R^T e(t) \tag{68}$$

where $K(t,\tau)$ is a function that can be selected by the designer as in hybrid controller, $e(t)$ is defined in (63), K_L and R are constant matrices (Messner et al., 1991).

4. SCARA Robot Model

The SCARA manipulator considered in this study is an experimental robot that has DC servo motors for the movements of elbow and shoulder. The third movement is controlled pneumatically. The schematic configuration of the robot is shown in Fig. 6.

The electrical and mechanical dynamical equations of the manipulator are as follows (Das & Dulger, 2005)

$$\begin{bmatrix} L_{a1} & 0 \\ 0 & L_{a2} \end{bmatrix}\begin{bmatrix} \dot{I}_{a1} \\ \dot{I}_{a2} \end{bmatrix} + \begin{bmatrix} R_{a1} & 0 \\ 0 & R_{a1} \end{bmatrix}\begin{bmatrix} I_{a1} \\ I_{a2} \end{bmatrix} + \begin{bmatrix} K_{e1} & 0 \\ 0 & K_{e2} \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} V_{a1} \\ V_{a2} \end{bmatrix} \tag{69}$$

$$\begin{bmatrix} K_{T1} & 0 \\ 0 & K_{T2} \end{bmatrix}\begin{bmatrix} I_{a1} \\ I_{a2} \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{70}$$

$$\begin{bmatrix} A & B \\ E & D \end{bmatrix}\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} C \\ F \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{71}$$

The elements A, B, C, and D in (71) are defined as

$$A = J_{m1} + \frac{(J_1 + J_2)}{N_1^2} + \frac{m_1 r_1^2 + m_2 r_2^2 + 4m_2 r_1^2}{4N_1^2} + \frac{m_2 r_1 r_2}{N_1^2}\cos\left(\frac{\theta_{m2}}{N_2}\right) \tag{72a}$$

$$B = \frac{J_2}{N_1 N_2} + \frac{m_2 r_2^2}{4N_1 N_2} + \frac{m_2 r_1 r_2}{2N_1 N_2}\cos\left(\frac{\theta_{m2}}{N_2}\right) \tag{72b}$$

$$C = \frac{-m_2 r_1 r_2}{N_1 N_2}\left(\frac{\dot{\theta}_{m1}\dot{\theta}_{m2}}{N_1} + \frac{\dot{\theta}_{m2}^2}{2N_2}\right)\sin\left(\frac{\theta_{m2}}{N_2}\right) \tag{72c}$$

$$D = J_{m1} + \frac{J_2}{N_2^{\,2}} + \frac{m_2 r_2^{\,2}}{4N_2^{\,2}} \tag{72d}$$

$$E = \frac{J_2}{N_1 N_2} + \frac{m_2 r_2^{\,2}}{4N_1 N_2} + \frac{m_2 r_1 r_2}{2N_1 N_2}\cos\left(\frac{\theta_{m2}}{N_2}\right) \tag{72e}$$

$$F = \frac{m_2 r_1 r_2}{2N_1 N_2}\left(\frac{\dot{\theta}_{m1}^2}{N_1}\right)\sin\left(\frac{\theta_{m2}}{N_2}\right) \tag{72f}$$

where I_{a1} and I_{a2} are motor currents, V_{a1} and V_{a2} are motor voltages, T_1 and T_2 are motor torques, θ_1 and θ_2 are link angles and θ_{m1} and θ_{m2} are motor angles of link 1 and link 2, respectively. Other physical parameters and their values that appear in (72) for the Serpent-1 model SCARA robot are given in Table 1.

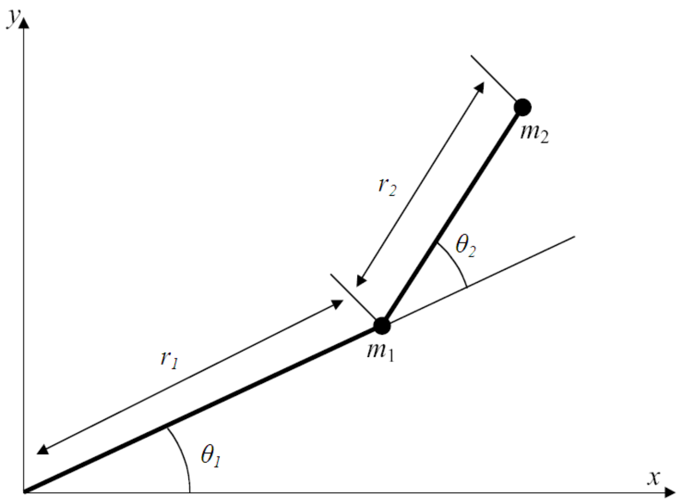


Fig. 6. Upper view of SCARA robot

Parameter	Meaning	Value
L_{a1}, L_{a2}	Armature inductances of motors 1 and 2	1.3mH, 1.3mH
R_{a1}, R_{a2}	Armature resistances of motors 1 and 2	3.5 Ω , 3.5 Ω
K_{e1}, K_{e2}	Inverse emf coefficients of motors 1 and 2	0.047 V.s/rad, 0.047 V.s/rad
K_{T1}, K_{T2}	Torque coefficients of motors 1 and 2	0.047 Nm/A, 0.047 Nm/A

J_1, J_2	Moment of inertias of arms 1 and 2	0.0980kgm^2, 0.0980kgm^2
J_{m1}, J_{m2}	Inertias of motors 1 and 2	3.3.10^{-6} kgm^2, 3.3.10^{-6} kgm^2
m_1, m_2	Masses of arms 1 and 2	1.90 kg, 0.93 kg
r_1, r_2	Lenghts of arms 1 and 2	250 mm, 150 mm
N_1, N_2	Gearbox ratios of motors 1 and 2	90, 220

Table 1. Serpent-1 robot parameters and their values

5. Simulation

Dynamics of the SCARA robot and three types of controllers, namely PD, learning and adaptive/learning controllers are modelled in MATLAB Simulink environment. A general simulation model is given in Fig. 7.
In the first simulation, the SCARA is controlled by PD controller. In this case, the electrical dynamics are neglected and the controller block is replaced with a PD controller (Fig.7). The control coefficients are selected as K_{p1}=300, K_{d1}=50, K_{p2}=30, K_{d2}=15 for link 1 and link 2, respectively (Das & Dulger, 2005).
As the second simulation, SCARA is controlled by learning controller. Here the electrical dynamics are again neglected and the controller block is replaced with the learning controller designed by (Messner et al., 1991). In the learning controller, the parameters are selected as;

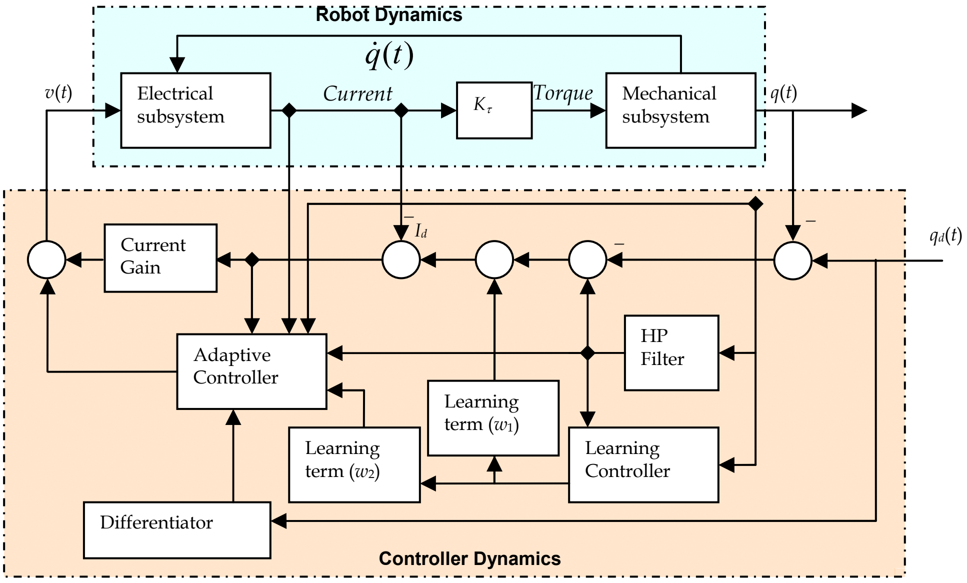


Fig. 7. Detailed Block diagram of robot and controller

$$F_p = \begin{bmatrix} 2000 & 0 \\ 0 & 160 \end{bmatrix} \quad (73)$$

$$F_v = \begin{bmatrix} 200 & 0 \\ 0 & 4 \end{bmatrix} \quad (74)$$

$$K_L = \begin{bmatrix} 2000 & 0 \\ 0 & 175 \end{bmatrix} \quad (75)$$

and λ_p=10, ve σ_n=0, $d_m(x_p)$=0 (Messner et al., 1991). The computation of $\dot{\hat{c}}_x$ and w_r are accomplished by numerical integration with embedded function blocks. The learning controllers have two different independent dynamic (time) variables. The simulation packages do not allow more than one independent simulation variables. To overcome this limitation, the second time variable is defined as a discrete variable and at every discrete point some state variables are introduced according to the dynamics. The differentiation and integration in the second variable are defined through summation and difference equations. The result is a heavy computational burden on the system.

The simulation model of the adaptive/learning hybrid controller is essentially the same as in Fig. 7. The parameters of the adaptive/learning controller are selected as; k=15, σ=12 and

$$K_L = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix} \quad (76)$$

Again, the computation of $\dot{\hat{c}}_x$, $\dot{w}_1$, w_2 are realized with numerical integrator blocks.

The desired link angle function is chosen as

$$q_d(t) = -0.5 + (-1 + \tanh(10\cos(\omega t))), \quad (77)$$

where ω=1 rad/s.

The function given in (77) is a pick-and-place type task that is widely used in industrial applications. This trajectory function satisfies the periodicity and continuous 3rd order derivative requirements of hybrid/learning controller as discussed in section 3.4.

The desired and achieved link angles when PD controller is used and the link angle errors are given in Fig. 8 and Fig. 9, respectively. The maximum angle errors are 0.4 rad for first link and 0.65 rad for the second link.

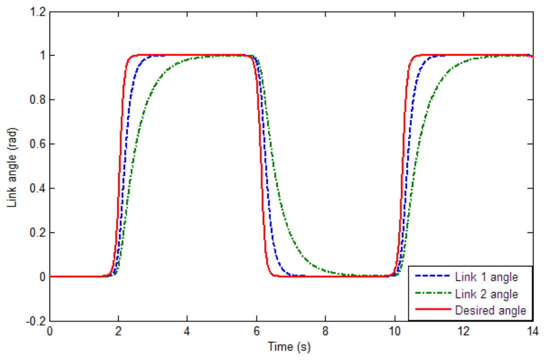


Fig. 8. Desired and simulated link angles when PD controller is utilized

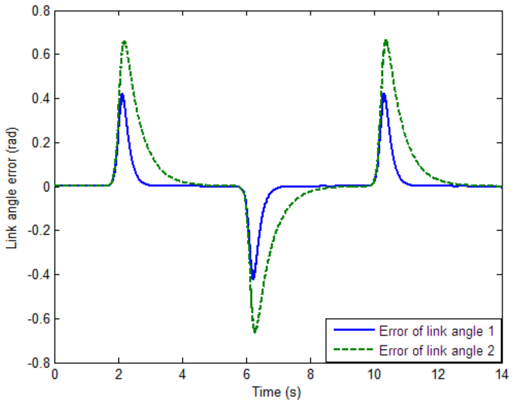


Fig. 9. Link angle errors when PD controller is used

Similarly, the link angle errors for learning controller are plotted in Fig. 10. The maximum angle errors are 0.09 rad for first link and 0.19 rad for the second link. The angle error decreased with respect to PD controller case as it is expected.
The link angle errors are given in Fig. 11 for the hybrid controller. Note that, the maximum link angles are lower compared to learning controller, 0.06 rad for both link 1 and link 2 (the error plots for link 1and 2 are overlapped in Fig. 11). It is worth noting that, the link angle errors have greater average values when hybrid controller is used. We think that the average value is greater for the hybrid controller, since it uses less information for the compensation of the uncertainties comparing with the learning controller given in (63), which uses both link positions and velocities. However the hybrid controller uses the measurements of link positions and motor currents. Furthermore, the learning controller neglects the electrical dynamics and compensates for only mechanical parameter uncertainties. On the other hand, the hybrid controller does not neglect electrical dynamics and compensates for mechanical and electrical parameter uncertainties. That is, the computational burden on the hybrid controller is much more than the learning controller. We think that this fact results more error in the average although the maximum error is less.

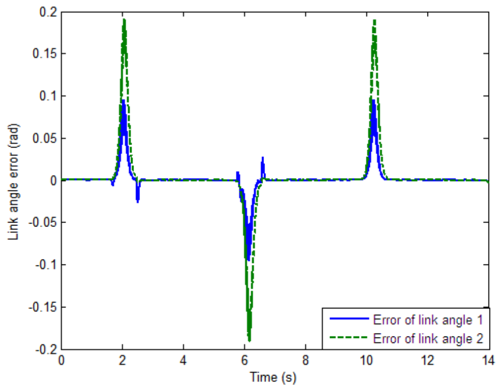


Fig. 10. Link angle errors when learning controller is used

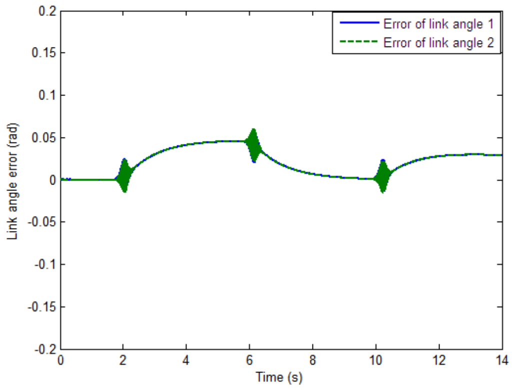


Fig. 11. Link angle errors when adaptive/learning controller is used

6. Conclusion

In this paper, the design of the hybrid adaptive/learning controller is described. Also the design of the learning controller proposed by (Messner et al., 1991) is described shortly along with a classical PD controller. The simulation model of a SCARA robot manipulator is presented and the performance of the controllers are examined through simulation runs. The simulation model and its parameters are based on a physical model of a SCARA robot given in (Das & Dulger, 2005). The simulation model includes the mechanical subsystem, electrical subsystem and the three different types of controllers. The classical PD, learning and adaptive/learning controller schemes are modelled and SCARA robot is simulated with three types of controllers.

The second time variable introduced in learning type controllers results a computational burden in dynamics, since the dynamics of controller is dependent both on the real time variable and the second time variable created via the Hilbert-Schmidt kernel used in learning laws. Moreover, no standard simulation package allows the use of a second

independent time variable in the models. To overcome this difficulty, we discretize the second variable. In order to keep the dynamics with respect to that variable we should have introduced a large number of extra system states at each discrete point of the second variable. Although the simulation is sufficiently fast with a high performance (1.7GHz CPU and 512MB RAM) personal computer, it is not fast enough with a personal computer of lower specifications (667Mhz CPU and 64MB RAM). Considering the much slower computers employed for the single task of controlling industrial robots, a real time application apparently is not possible at this stage. Therefore, the work to reduce the computational burden in the control law is continuing and as soon as this is achieved, an experiment to examine the hybrid controller for a real robot will be performed.
The parameters of a 2-link Serpent-1 model robot are used in simulations and the robot is desired to realize a pick and place type movement. According to the simulation results, the learning and adaptive/learning hybrid controllers provided lower angle errors compared to classical PD controller. Moreover, the maximum angle errors of links when controlled by adaptive/learning controller decreased from 0.09 rad to 0.06 rad for first link and 0.19 rad to 0.06 rad for second link compared to learning controller, which means 33.3% and 63.1% decrement for first link and second link, respectively.
Although the hybrid controller is more complex than PD and learning controllers, its position and velocity errors have smaller maximum values than the learning controller. However its performance is not good in the error averages. We think that the high error averages are due to the fact that the hybrid controller uses partial state information (no link velocities) and compensates for both mechanical and electrical parameter uncertainties, whereas the learning controller uses full state information (both link positions and velocities) though it compensates only for mechanical uncertainties, since it neglects electrical dynamics.
Our work is continuing to develop more powerful computational schemes for the hybrid adaptive/learning controller to reduce the computational burden. Recently, we tried to introduce a low pass filter in the hybrid controller to filter the high frequency components, which effect the tracking performance negatively, in the input voltage. The preliminary results show that the error becomes smoother and its average value reduces.

7. References

Arimoto, S. (1986). Mathematical theory of learning with applications to robot control, In: Adaptive and Learning Systems, K.S. Narendra (Ed.), Plenum Press, ISBN: 0306422638, New York.

Arimoto, S.; Kawamura, S.; Miyazaki, F. & Tamaki, S. (1985). Learning control theory for dynamical systems. Proceedings of IEEE 24th Conference on Decision and Control, 1375-1380, ISBN: 9999269222, Ft. Lauderdale FL, December 1985, IEEE Press, Piscataway NJ.

Bondi, P.; Casalino, G. & Gambardella, L. (1988). On the iterative learning control theory of robotic manipulators. IEEE Journal of Robotics and Automation, Vol. 4, No.1, (February 1988), 14-22, ISSN: 0882-4967.

Burg, T.; Dawson, D. M.; Hu, J. and de Queiroz, M. (1996). An adaptive partial state feedback controller for RLED robot manipulators. IEEE Transactions on Automatic Control, Vol. 41, No. 7, (July 1996), 1024-1030, ISSN:0018-9286.

Canbolat, H.; Hu, J. & Dawson, D.M. (1996). A hybrid learning/adaptive partial state feedback controller for RLED robot manipulators. International Journal of Systems Science, Vol. 27, No. 11, (November 1996), 1123-1132, ISSN:0020 7721.

Das, T. & Dülger, C. (2005). Mathematical Modeling, Simulation and Experimental Verification of a SCARA Robot. Simulation Modelling Practice and Theory, Vol.13, No.3, (April 2005), 257-271, ISSN:1569-190X.

De Queiroz, M.S.; Dawson, D.M. & Canbolat, H. (1997). Adaptive Position/Force Control of BDC-RLED Robots without Velocity Measurements. Proceedings of the IEEE International Conference on Robotics and Automation, 525-530, ISSN:1050-4729, Albuquerque NM, April 1997, IEEE Press, Piscataway NJ.

Fu, K.S.; Gonzalez, R.C. & Lee, C.S.G. (1987). Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill, ISBN:0-07-100421-1, New York.

Golnazarian, W. (1995). Time-Varying Neural Networks for Robot Trajectory Control. Ph.D. Thesis, University Of Cincinnati, U.S.A.

Horowitz, R.; Messner, W. & Moore, J. (1991). Exponential convergence of a learning controller for robot manipulators. IEEE Transactions on Automatic Control, Vol. 36, No. 7, (July 1991), 890-894, ISSN:0018-9286.

Jungbeck, M. & Madrid, M.K. (2001). Optimal Neural Network Output Feedback Control for Robot Manipulators. Proceedings of the Second International Workshop on Robot Motion Control, 85-90, ISBN: 8371435150, Bukowy Dworek Poland, October 2001, Uniwersytet Zielonogorski, Instytut Organizacji i Zarzadzania.

Kaneko, K.& Horowitz, R. (1992). Learning control of robot manipulators with velocity estimation. Proceedings of USA/Japan Symposium on Flexible Automation, 828-836, ISBN: 0791806758, M. Leu (Ed.), San Fransisco CA, July 1992, ASME.

Kaneko, K. & Horowitz, R. (1997). Repetitive and Adaptive Control of Robot Manipulators with Velocity Estimation. IEEE Trans. Robotics and Automation, Vol. 13, No. 2 (April 1997), 204-217, ISSN:1042-296X.

Kawamura, S.; Miyazaki, F. & Arimoto, S. (1988). Realization of robot motion based on a learning method. IEEE Transactions on Systems, Man and Cybernetics, Vol.18, No. 1, (Jan/Feb 1988), 126-134, ISSN:0018-9472.

Kuc, T.; Lee, J. & Nam, K. (1992). An iterative learning control theory for a class of nonlinear dynamic systems. Automatica Vol.28, No.6, (November 1992), 1215-1221, ISSN:0005-1098.

Lewis, F.L.; Abdallah, C.T. & Dawson, D.M. (1993). Control of Robot Manipulators, Macmillan, ISBN: 0023705019, New York.

Messner, W.; Horowitz, R.; Kao, W.W. & Boals M. (1991). A new adaptive learning rule. IEEE Transactions on Automatic Control, Vol. 36, No. 2, (February 1991) 188-197, ISBN:0018-9286.

Qu, Z.; Dorsey, J.; Johnson, R. & Dawson, D.M. (1993). Linear learning control of robot motion. Journal of Robotic Systems Vol.10, No.1, (February 1993), 123-140, ISBN: 0741-2223.

Sadegh, N.; Horowitz, ; Kao, W.W. & Tomizuka, M. (1990). A unified approach to the design of adaptive and repetitive controllers for robotic manipulators. ASME Journal of Dynamic Systems, Measurement and Control, Vol.112, No.4 (December 1990), 618-629, ISSN: 0022-0434.

Sahin, V.D. & Canbolat, H. (2007). DC Motorlarla Sürülen Robot Manipülatörleri için Gecikmeli Öğrenme Denetleyicisi Tasarımı (Design of Delayed Learning Controller for RLED Robot Manipulators Driven by DC Motors). TOK'07 Otomatik Kontrol Milli Toplantısı Bildiriler Kitabı (Proc. of TOK'07 Automatic Control National Meeting), 130-133, Istanbul, Turkey, September 2007, Istanbul (Turkish).

Uğuz, H. & Canbolat, H. (2006). Simulation of a Hybrid Adaptive-Learning Control Law for a Rigid Link Electrically Driven Robot Manipulator. Robotica, vol.24, No.3, (May 2006), 349-354, ISSN: 0263-5747.