SELF-MOTION CONTROL OF KINEMATICALLY REDUNDANT ROBOT MANIPULATORS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Mechanical Engineering

by
Omar Waleed Najm MAAROOF

July 2012
İZMİR

We approve the thesis of **Omar Waleed Najm MAAROOF**

Examining Committee Members:

Assist. Prof. Dr. Mehmet İsmet Can DEDE
Department of Mechanical Engineering
İzmir Institute of Technology

Assoc. Prof. Dr. Enver TATLICIOĞLU
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

Prof. Dr. Nuri Süha BAYINDIR
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

03 July 2012

Assist. Prof. Dr. Mehmet İsmet Can DEDE
Supervisor, Department of Mechanical Engineering
İzmir Institute of Technology

Prof. Dr. Metin TANOĞLU
Head of the Department of
Mechanical Engineering

Prof. Dr. R. Tuğrul SENGER
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Asst. Prof. Dr. Mehmet İsmet Can Dede for his invaluable advice, guidance, and encouragement.

I would like to thank all of my professors for their help during my study and research.

I would like to thank my wife for her support and patience for my long hours of working and her accompany during living abroad. I am also grateful to my parents for their endless support during my thesis and all of my life.

ABSTRACT

SELF-MOTION CONTROL OF KINEMATICALLY REDUNDANT ROBOT MANIPULATORS

Redundancy in general provides space for optimization in robotics. Redundancy can be defined as sensor/actuator redundancy or kinematic redundancy. The redundancy considered in this thesis is the kinematic redundancy where the total degrees-of-freedom of the robot is more than the total degrees-of-freedom required for the task to be executed. This provides infinite number of solutions to perform the same task, thus, various subtasks can be carried out during the main-task execution.

This work utilizes the property of self-motion for kinematically redundant robot manipulators by designing the general subtask controller that controls the joint motion in the null-space of the Jacobian matrix. The general subtask controller is implemented for various subtasks in this thesis. Minimizing the total joint motion, singularity avoidance, posture optimization for static impact force objectives, which include maximizing/minimizing the static impact force magnitude, and static and moving obstacle (point to point) collision avoidance are the subtasks considered in this thesis.

New control architecture is developed to accomplish both the main-task and the previously mentioned subtasks. In this architecture, objective function for each subtask is formed. Then, the gradient of the objective function is used in the subtask controller to execute *subtask objective* while tracking a given end-effector trajectory. The tracking of the end-effector is called *main-task*.

The SCHUNK LWA4-Arm robot arm with seven degrees-of-freedom is developed first in SolidWorks® as a computer-aided-design (CAD) model. Then, the CAD model is converted to MATLAB® Simulink model using SimMechanics CAD translator to be used in the simulation tests of the controller. Kinematics and dynamics equations of the robot are derived to be used in the controllers. Simulation test results are presented for the kinematically redundant robot manipulator operating in 3D space carrying out the main-task and the selected subtasks for this study. The simulation test results indicate that the developed controller's performance is successful for all the main-task and subtask objectives.

ÖZET

KİNEMATİK OLARAK ARTIK ROBOT KOLLARININ İÇ-HAREKET DENETİMİ

Robotbiliminde artıklık genellikle eniyileştirme için bir alan sağlar. Artıklık algılayıcı/eyletici veya kinematik artıklık olarak tanımlanabilir. Bu tezde ele alınan artıklık, robotun toplam serbestlik derecesinin gerçekleştirilecek görevin gerektirdiği toplam serbestlik derecesinden fazla olduğu kinematik artıklıktır. Bu durum, aynı görevi yerine getirmek için sonsuz sayıda çözüm imkanı sağlar, dolayısı ile, değişik alt-görevlerin ana görev gerçekleştirilirken devam ettirilmesine imkan verir.

Bu çalışmada, Jacobi matrisinin sıfır uzayında, eklem hareketlerini denetleyen genel alt-görev denetleyicisi tasarlayarak, kinematik olarak artık robot kollarının iç-hareketi özelliği kullanılmaktadır. Bu tezde genel alt-görev denetleyicisi değişik al-görevler için uygulanmıştır. Bu alt-görevler, toplam eklem hareketinin en aza indirgenmesi, tekillikten kaçınma, statik darbe büyüklüğünü arttıran/azaltan statik darbe kuvveti amaçlarını içeren duruş eniyileştirmesi, ve statik ve dinamik (noktadan-noktaya) çarpışmadan kaçınma bu tezde ele alınan alt-görevlerdir.

Hem ana görevi hem de daha önce bahsi geçen alt-görevleri gerçekleştirmek için yeni denetleme mimarisi geliştirildi. Bu mimaride, amaç fonksiyonu her alt-görev için oluşturuldu. Sonrasında, robotun uç noktasının belirlenmiş yörüngesi takip edilirken, amaç fonksiyonunun gradyanı, alt-görev denetleyicisinde alt-görev hedefini gerçekleştirmek için kullanıldı. Robotun uç noktasının yörüngesinin takip edilmesi ana görev olarak adlandırılmıştır.

Yedi serbestlik dereceli SHUNCK LWA4-Arm robot kolu ilk önce SolidWorks® ortamında bilgisayar-destekli-tasarım (BDT) modeli olarak oluşturuldu. Sonrasında, denetleyicisinin benzetim testlerinde kullanılması için BDT modeli MATLAB® Simulink modeline SimMechanics BDT çeviricisi ile dönüştürüldü. Denetleyicide kullanılması için robotun kinematik ve dinamik denklemleri türetildi. Bu çalışma için, benzetim test sonuçları, ana görev ve seçilmiş alt-görevleri yerine getiren 3B uzayda çalışan, kinematik olarak artık robot kol için sunuldu. Benzetim testi sonuçları oluşturulan denetleyici peformansının bütün ana görev ve alt-görev hedefleri için başarılı olduğunu göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF SYMBOLS

Abbreviations

CAD	Computer Aided Design.
DH	Denavit Hartenberg convention
DOF	Degrees-Of-Freedom
GPM	Gradient Projection Method
SOI	Surface Of Influence
SVD	Singular Value Decomposition
HTM	Homogenous Transformation Matrix

Lowercase

a_i	Link length
a_k	Effective length $A_k O_k$ of link k along $\vec{u}_1^{(k)}$.
d_{ij}	Minimum distance between i^{th} link and j^{th} obstacle
d_i	Link offset
d_k	Constant offset $O_{k-1}A_k$ of link k with respect to link k-1 along $\vec{u}_3^{(k-1)}$ if joint k is revolute
e	Main-task tracking error for end-effector position
$\dot{e}_N$	Subtask velocity tracking error
$e^{\phi\tilde{u}}$	3x3 rotation matrix representing an orientation ϕ about the axis $\bar{u}$
$f(q)$	Subtask objective function
g	Gradient of subtask objective function
$\bar{g}$	Gravitational effect vector
J	Jacobian matrix
J_{ci}	Jacobian matrix at point i
J_p	Jacobian matrix at end-effector
J^+	Psuedo-inverse of Jacobian matrix
k_σ	Condition number of the matrix

l	Total number of links will avoid obstacle
m	Dimension of the end-effector space
n	Dimension of the joint space
o	Total number of obstacle
$\bar{p}$	Tip point position vector with the elements $p_x, p_y, and\ p_z$
p_m	Point on the manipulator link m
p_o	Point on the obstacle
q	Vector of joint variables
$\dot{q}$	Vector of joint velocities
r	Dimension of the task space
$\bar{r}$	Position vector.
$\bar{r}_{ci}^{(0)}$	Position vector of the center of mass of link i with respect to base frame
$\bar{r}$	Wrist position vector with the elements $d_x, d_y, and\ d_z$
v	Linear velocity vector $\Re^3$of the end-effector
v_{cij}	Escape velocity gain along the direction away from the critical point
v_{ci}	Escape velocity for multiple obstacles
$\dot{q}_c$	Vector of joint velocities as a gradient of objective function to avoid obstacles
v_m	Maximum escape velocity gain
$\tilde{u}$	Cross-product matrix corresponding to the unit column vector $\bar{u}$
$\vec{u}_i$	i^{th} unit vector
u_{ij}	Unit vector from the critical point C_i on the i^{th} link to the j^{th} obstacle
$\vec{u}_i^{(k)}$	Unit basis vectors of $\mathcal{F}_k$; i=1,2,3.
w_m	Manipulability measure
w_f	A scalar of manipulating force measure
x	End-effector pose
$\dot{x}$	End-effector velocity

Uppercase

C	Centripetal and Coriolis matrix
$\hat{C}^{(k-1,\ k)}$	3x3 transformation matrix representing the orientation of the frame *k* with respect to the frame *k*-1
C_i	Critical point at i^{th} link
$F(\dot{q})$	Friction effects vector $\Re^n$
F	Static force/ force and torque vector
G	Gravitational vector
$G_{\bar{r}}$	Representing of $\bar{r}$ vector with respect to frame G
${}^{G}R_B$	Rotation matrix representing rotation from frame B to frame G
$\hat{H}^{(k-1,\ k)}$	4x4 Homogeneous transformation matrix
I	Identity matrix
K	Kinetic energy
K_i	Integral gain.
K_N	Positive definite feedback gain matrix
K_p	Constant feedback gain matrix
K_v	Constant feedback gain matrix
L	Lagrangian of the system
M	Inertia Matrix
N	Calculated nonlinear terms that appear in the dynamics equation of the robot
O_k	Origin of the reference frame $\mathcal{F}_k$ attached to link *k.*
P	Potential energy
P_e	Length magnitude from wrist point to end-effector tool
Ri	The i^{th} revolute joint attached on link *i*
$\hat{R}_{\bar{u},\phi}$	3x3 rotation matrix representing an orientation ϕ about the axis $\bar{u}$
S_k	Variable offset $O_{k-1}A_k$ of link *k* with respect to link *k*-1 along $\vec{u}_3^{(k-1)}$ if joint k is prismatic

Greek

ϕ	Rotation angle
θ_i	Joint angle
$\dot{\theta}_N$	Vectors of joints velocity in the null space
$\ddot{\theta}_N$	Vectors of joints acceleration in the null space
θ_k	Rotation angle of link *k* with respect to link *k*-1 about $\vec{u}_3^{(k-1)}$.
α_k	Twist angle of joint *k*+1 with respect to joint *k* about $\vec{u}_1^{(k)}$.
λ_i	Eigenvalues of a matrix
σ_i	Singular values of a matrix
φ	Force transmission ratio at end-effector and along a specific direction
ω	Angular velocity magnitude
$\bar{\omega}$	Angular velocity column vector
λ	Eigenvalue
σ	Singular value
∇	Gradient
$\tau(t)$	Torque vector $\Re^n$
ξ_d	Disturbance effects vector $\Re^n$

Script

$\mathcal{F}_i$	i^{th} coordinate frame attached on link *i*
$\mathcal{F}_{k-1}^{(O_{k-1})}$	Reference frame fixed to link *k-1*
$\mathcal{F}_k^{(O_k)}$	Reference frame fixed to link *k*

Note: Whenever ($\widehat{\ }$) and ($\bar{\ }$) come over a symbol, this symbol will be considered as a non-column matrix or a vector, respectively. Otherwise, the symbol will be defined.

CHAPTER 1

OVERVIEW FOR REDUNDANT ROBOT MANIPULATORS

Robotics is concerned with the study of machines that are expected to replace human beings in the execution of a task, as regards both physical activity and decision making. Robot makers tend to imitate the characteristics in nature/human. Redundancy is present in human or in any living creature. Human body redundancy can be observed in different levels such as; sensor/actuator level and mechanism level. Sensor/actuator level redundancy includes having two eyes, ears, some interior organs, etc. The mechanism level redundancy can be observed in multiple fingers, and two arms. Furthermore, the arm itself is redundant and this is called kinematic redundancy. Kinematic redundancy provides extra capabilities for the human arm, which will be explored for robot manipulators in this thesis. The aim of this introductory chapter is to define the redundancy concept in robot manipulators, kinematic redundancy, self-motion and the applications of redundant robot manipulators. The objective of this work, contributions of the thesis to the literature, and the contents of this thesis as an outline are presented at the end of the chapter.

1.1. Redundant Robot Manipulator

The motivation for introducing redundancy in robot manipulators goes beyond that for using redundancy in traditional engineering design, namely, increasing robustness to faults so as to improve reliability (e.g., redundant processors or sensors or actuators). The term redundancy used in this thesis, means kinematic redundancy.

1.1.1. Kinematic Redundancy

Redundancy is described in various papers and books which discuss several different issues related to redundant robots. The definitions are very similar to each

other but the most general and clarified definition is given by (Conkur and Buckingham 1997). This definition gives a clear and simple method of describing all potential cases for robot manipulators, relating to the dimensions of the spaces, the manipulator end-effector and the task space. The manipulator can be described as having *n* axes of motion. Similarly, the space defined by the achievable motion of the end-effector will have a dimension *m*. The task space will have dimension *r*.

- **Case 1: $n = m$**

  This is the standard non-redundant robot.

- **Case 2: $n > m$**

  When *n* is designed to be greater than *m*, then the device is redundant. In such situations, the self-shape of a device can be varied without changing the end-effector pose, since the joints do not produce independent motion in end-effector space.

- **Case 3: $m > r$**

  When the task space *r* is completely within the end-effector space *m*, and the dimension of the end-effector space *m*, irrespective of the dimension of the joint space *n*, is greater than the task space, then this describes *task redundancy*.

- **Case 4:**

  As a distinct case in any of these situations there can be examples where for a particular configuration, a mirror configuration exists (such as elbow-up and elbow-down case in Two DOF Revolute Arm). This gives rise to multiple solutions where there is a finite and well defined set of solutions for the end-effector pose.

These mathematical expressions can now be converted into definitions:

Definition 1: If the number of solutions to the inverse kinematics of a manipulator is not unique but finite, the manipulator is said to have *multiple solutions*.

Definition 2: If the dimension of joint space is greater than the dimension of end-effector space then the device is *kinematically redundant*.

Definition 3: If the task-space is completely contained by, and has a lower dimensionality than the end-effector space, the manipulator is said to be *task redundant.*

In standard, it is not necessary that a manipulator is fundamentally redundant; rather, there are some tasks with respect to those make it redundant. Since it is usually recognized that a general task space consists of following an end-effector motion trajectory requiring six degrees-of-freedom (DOF), a robot arm with seven or more DOFs is considered as the typical example of inherently redundant manipulator, for

example 7-DOF LWA4-Arm by SCHUNK. However, even in robot arms with fewer DOF, redundancy can be seen as implying extra DOF which are not needed for tasks in well-organized environments. For instance, using a six axis machine for welding or for stereotactic surgery is not kinematically required, since five axes will place a uniform cross section tool at the appropriate position with the required orientation to complete the task, no final rotation about the tool axis is required. Another example is a planar robotic arm manipulator with three or more joints that acquires to track end-effector position in a Cartesian plane.

In fact, providing robot manipulators with kinematic redundancy is mainly aimed at increasing dexterity and using arm motion in the null space for subtask objective, during executing the main-task in the task space. A simple example given in Figure 1.1.

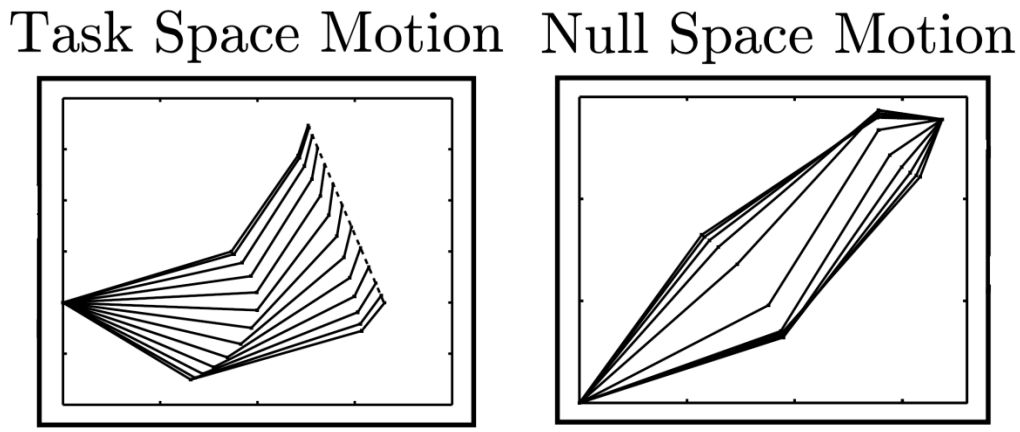


Figure 1.1. Task space and null space motion

In addition to the distinction between non-redundant and redundant robot manipulators, it is also necessary to specify the number of redundancy. Hence the redundant manipulator has more DOF than is required to perform a task in the task space. These extra DOFs allow the robot manipulator to perform more dexterous manipulation and/or provide the robot manipulator with increased flexibility for the execution of sophisticated tasks. For this reason, highly or hyper-redundant manipulators present to improve manipulator performance in complex and unstructured environment. Highly or hyper redundant manipulators are often used in conjunction with trunk or snake robots. The intent is clear, such devices either being planar or fully spatial should have a joint space dimension n that is much greater than the dimension of

the end-effector space *m* (*i. e.*, $n >> m$). Since the maximum value of *m* is 6, any value of *n* greater than 10 fulfills this criterion (Reznik *et al*. 1995). Often the implication of such designs is the need to consider non-Jacobian based inverse kinematics techniques for controlling the self-motion of the device.

1.2. Self-motion

Early manipulator designs, which are characterized by designing the manipulator with the minimum number of joints required to execute its task resulted in a serious limitation in real-world applications. The limitations are due to the singularity problem, joint limits, and workspace obstacles. These limitations increase the regions to be avoided in the joint and task space during operation, thus requiring a carefully planned task space of the manipulator. This is the case of work cells in traditional industrial applications.

On the other hand, the presence of additional DOFs, besides those strictly required to execute the task, brings the flexibility to avoid the limitations stated above. This flexibility is due to having infinite joint configurations for the same end-effector pose since there are an infinite number of possible solutions for the inverse kinematics analysis of redundant robot manipulators. As a result, there exists joint motion which can be propagated in the null space of the Jacobian matrix of a robot manipulator without affecting the end-effector pose. A phenomenon commonly referred to as *self-motion* (Nakamura 1991). This phenomenon implies that the same task at the end-effector level can be executed in several ways at the joint level, giving the possibility of avoiding limitations and ultimately resulting in a more versatile mechanism.

Such a feature is a key to allowing operation in unstructured or dynamically varying environments that characterize advanced industrial applications and service robotics scenarios. In practice, if properly managed, the increased dexterity of kinematically redundant manipulators may allow the robot manipulator to avoid singularities, joint limits, and workspace obstacles, but also to minimize torque/energy over a given task, ultimately meaning that the robotic manipulator can achieve a higher degree of autonomy.

1.3. Applications of the 7-DOF Robot Manipulator

The biological archetype of kinematically redundant manipulator is the human arm, which, not unexpectedly, also inspires the terminology used to characterize the structure of serial-chain manipulators. In fact, the human arm has three DOF at the shoulder, one DOF at the elbow and three DOF at the wrist. The available redundancy can easily be verified, e.g., by laying one's palm on a table and moving the elbow while keeping the shoulder motionless. The kinematic arrangement of the human arm has been replicated in a number of robots often termed as human-arm-like manipulators. This family of 7-DOF manipulators is used for many applications such as; the care-providing robotic system FRIEND presented in Figure 1.2, which is a semi-autonomous robot designed to support disabled and elderly people in their daily life activities (Schunk 2011). Another example is the usage of Care-o-bot, a next generation of household robots, presented in Figure 1.3.

Figure 1.2. The care-providing robotic system FRIEND
(Source: Schunk 2011)

Figure 1.3. Care-o-bot® Household robot
(Source: Fraunhofer 2010)

The new application of an advanced brain-machine interface is shown in Figure 1.4. This brain-controlled DLR redundant robot arm is used for assisting disabled people. Another application shown in Figure 1.5 is the MODICAS assistance robot for total hip arthroplasty implantation using a 7-DOF Mitsubishi PA-10 robot arm.

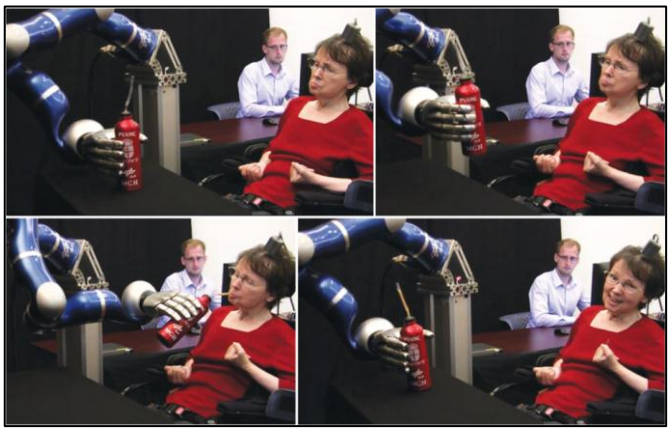


Figure 1.4. Paralyzed woman drinking from a bottle using the DLR Lightweight Robot.
(Source: Leigh *et al*. 2012)

Figure 1.5. Mitsubishi PA-10 Robot arm
(Source: Hans-Christian *et al.* 2010)

The use of two or more robot arms to execute a task, which is the case in the humanoid robot arms that use cooperating multi-fingered hands, is another important application for the 7-DOF arms. The dual usage of 7-DOF redundant arms can be observed in Figure 1.6.

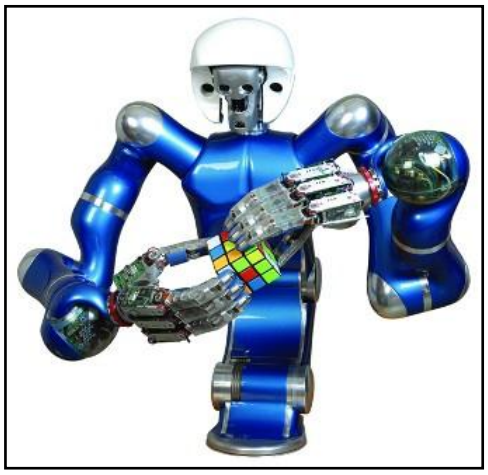


Figure 1.6. Justin the robot
(Source: DLR 2012)

Robot redundancy also has been recognized of major importance for manipulators in space applications. The Special-Purpose Dexterous Manipulator (SPDM), shown in Figure 1.7, consists of two 7-DOF arms which can be interpreted as enhancing the importance of kinematic redundancy in critical tasks.

Figure 1.7. The Special-Purpose Dexterous Manipulator (SPDM)
(Source: Canadian Space Agency 2012)

1.4. Objective of the Thesis

The main objective is to design the control torque input signal, such that the robot end-effector can follow a desired end-effector position trajectory with minimum error. The control signal that will be fed into the actuation system should also include enough information to execute subtasks defined by at least one motion optimization measure. The secondary aim of this study is to integrate a number of different subtasks including minimizing the total joint motion, singularity avoidance, posture optimization for the static impact force objectives, which include maximizing/minimizing the static impact force magnitude, and static and moving obstacle (point to point) collision avoidance.

The controllers developed in this thesis are to be employed for a simulation model of a 7-DOF robot arm. The performance of the controllers is to be evaluated through the simulation tests for a selected main-task and the subtask objective.

1.5. Outline

This thesis comprises of seven chapters. The chapters and their contents are organized as;

- Chapter 2: Literature survey provides a survey for previous work and up-to-date approaches for the control of redundant robot manipulators.
- Chapter 3: Fundamentals of robotics and control, methodology and notation used throughout the thesis are explained in this chapter. It is possible to begin the modeling and control of the redundant robot manipulator with this background knowledge.
- Chapter 4: This chapter focuses on modeling the robot arm in SolidWorks® and MATLAB® Simulink, deriving the kinematics equations, and using the Euler-Lagrange formulation for dynamic modeling.
- Chapter 5: Control design for the main-task and the subtask along with the stability analysis, and the subtask objectives that will be used in this thesis are presented in this chapter.
- Chapter 6: In this chapter an explanation of the main-task objective and the simulation test scenario is followed with the presentation of the results achieved for all the subtasks.
- Chapter 7: This chapter presents the concluding summary of the thesis as well as the discussions on the results and recommendations for future work.

CHAPTER 2

LITERATURE SURVEY

As explained in the previous chapter, redundant robot manipulators are used in wide range of applications. Redundancy introduced to robot manipulators gives advantage upon non-redundant robots because of the self-motion property. Self-Motion property enables the manipulator to achieve the same task in infinite different configurations, and this property is the main motivation behind the thesis. The aim of this chapter is to present a survey on control of redundant robot manipulators.

2.1. Survey on Control of Redundant Robot Manipulators

Redundant robot manipulators provide increased flexibility for the execution of complex tasks, where the redundancy of such manipulators can be effectively used. Therefore, it has been the subject of significant research in the last decades. Various applications of manipulators in space, underwater, and hazardous material handling have led to considerable activity in many research areas especially for those applications using redundant robot manipulator arms.

In an industrial environment, the redundant robots are expected to reduce the manufacturing costs, increase the productivity, and possibly improve the safety for human co-operators. Redundancy of such manipulators should be effectively used in the near future in medical, assistive and rehabilitation robotics applications to improve the safety by answering specific configuration needs during the manipulation.

Self-Motion control of kinematically redundant manipulators has been the subject of intensive research in many directions. This motion is referred to *self-motion* by (Nakamura 1991) since it is not observed in the task space. The extra DOFs have been used to satisfy specific additional tasks. Kinematic redundancy has been promoted as a useful tool in robotics and investigated by many researchers. Robotics researchers have discussed how to specify such joint velocities while performing the main-task controller (end-effector trajectory tracking). In some studies, the redundancy resolution at velocity level (Rajiv *et al.* 1991) is performed to control the main-task integrating the

velocity of the null space into the control signal in task space control. In others, redundancy resolution at acceleration level is used (Wang *et al.* 2010). These techniques might be termed kinematic control.

Some researchers worked on other techniques of kinematic control by defining inverse kinematic not using pseudo inverse. In (Seraji 1989), the configuration control approach in which the end-effector motion in task space is augmented by any n-m dimensional additional tasks, such as optimization of kinematic and dynamic objectives or posture control, was proposed. (Yangmin *et al.* 2001) used neural networks (Cerebellar Model Articulation Controller) for solving the inverse kinematics problems in real time. Another study was conducted for decentralized kinematic control of multiple redundant manipulators for the cooperative task execution problem via recurrent neural networks (Shuai *et al.* 2012).

In (Rajiv *et al.* 1991), kinematic optimal control scheme was presented for 7-DOF manipulators. This scheme used the gradient projection optimization method in the framework of resolved motion rate control, and it did not require calculation of the pseudo-inverse of the Jacobian matrix.

Other works on dynamic control laws for redundant manipulators include approaches for redundancy resolution through torque optimization (Baillieul *et al.* 1984, Hollerbach *et al.* 1985 and Hollerbach *et al.* 1987).

In the work of (Khatib 1983), a control scheme based on the dynamic model of a manipulator in Cartesian space was proposed and this result was extended for redundant manipulators by using the pseudo-inverse of the Jacobian matrix. A dynamic feedback linearizing control law that guarantees asymptotic tracking of a desired end-effector trajectory and the desirable subtask objective was proposed in (Hsu *et al.* 1989). This controller required that the exact dynamic model of the robot manipulator should be known. In (Peng *et al.* 1993), a compliant motion control for kinematically redundant manipulators using an extended task-space formulation was explained.

A robust adaptive controller that ensures globally ultimately bounded Cartesian tracking was proposed in (Colbaugh *et al.* 1995), provided that no external disturbances are present in the robot dynamics and some sufficient conditions on control gains were satisfied. In (Zergeroğlu *et al.* 2004), an adaptive controller to compensate for the parametric uncertainty in the dynamic model was developed. They designed a dynamic feedback linearizing control law that guarantees the tracking of the redundant joint velocities while providing end-effector tracking. In (Tatlıcıoğlu *et al.* 2009), the

feedback linearizing controller presented in (Hsu *et al.* 1989) was redesigned to compensate for parametric uncertainties that are present in the dynamic model. The proposed design used a least-squares algorithm instead of gradient-type algorithms for parameter estimation.

In (Özbay *et al.* 2009), a controller that ensured uniformly ultimately bounded end-effector and subtask tracking is formulated despite the parametric uncertainty associated with the dynamic model. In (Oh *et al.* 1999), a disturbance observer-based robust controller that controls both the motion of the end-effector and the null space motion of the redundant manipulator was proposed. The proposed controller used an extended task space formulation to express both task space and null space dynamics.

A model-based controller that achieves exponential convergence of end-effector and subtask tracking errors was presented by (Zergeroğlu *et al.* 2004). The extensions for adaptive and output feedback type of controllers were also presented in this work.

As it can be observed from the previous research on control of kinematically redundant manipulators, the studies are mainly focused on two major approaches. One approach is the extended or augmented task space formulation. The dimension of the task space is extended by incorporating as many additional constraints as the degree of the redundancy in this approach. Therefore, the resulting system becomes non-redundant (Özbay *et al.* 2008). Another approach is the generalized/pseudo-inverse based control formulations that use the pseudo-inverse of the manipulator's Jacobian matrix in the control formulation. However, this approach cannot guarantee that the control remains bounded when the manipulator is near singularities associated with the Jacobian matrix.

Redundancy has been recognized as a major characteristic in performing tasks that require dexterity comparable to that of the human arm. Most methods for resolving redundancy in manipulation involve defining an objective function to satisfy specific additional tasks. The extra DOFs have been used for obstacle avoidance in (Guo *et al.* 1990, Wang *et al.* 1992, Kemeny 1999 and Chen *et al.* 2002), and in (Kwang-Kyu *et al.* 2007) where the transpose of the Jacobian matrix was used instead of its inverse. The kinematic redundancy is also used for other sub objectives such as mechanical joint-limit avoidance (Tatlıcıoğlu *et al.* 2009). Another use of the kinematic redundancy was observed for optimization of user-defined objective functions such as minimization of joint velocities and accelerations (Seraji 1991). Gertz (Gertz *et al.* 1991), Walker (Walker 1994) and Lin (Lin *et al.* 1995) have used a generalized inertia-weighted

inverse of the Jacobian matrix to resolve redundancy in order to reduce impact forces. In (Yoshikawa 1984), the manipulability measure was used, which has a minimum or maximum value at a desirable configuration. When using the objective function for subtask control while tracking a given end-effector pose, the gradient (or its negative) of objective function for a specific or multi subtasks was used to control joint velocities in the redundant directions.

2.2. Conclusion

In this chapter, the research areas identified for control of redundant manipulators were addressed. In this context, existing schemes in the areas of main-task and subtask control were reviewed. Based on the results of this review, a redundancy resolution scheme at the acceleration level is selected to be implemented in this thesis study to perform dynamic control. Hence dynamic control can deal with external forces and torques and the applied torque signals to the actuators can be bounded as a result end-effector force and torques can be bounded.

The subtask objectives selected for this work comply with the studies that are conducted on rehabilitation and assistive robotics in IZTECH Robotics Laboratory (IRL). Robot manipulators working with the human requires improved level of safety. Dexterity achieved as a result of kinematic redundancy is envisaged to be used for minimizing the total joint motion, avoiding of singularity, bounding the end-effector static force and collision avoidance for static and moving objects.

In spite of the challenging nature of solving the kinematics problem and defining the dynamics model for 7-DOF robot manipulator, the redundant spatial 7-DOF case is taken as a benchmark to gain insight toward a more general and realistic situation for the control study carried out in this work.

The property of self-motion for redundant robot manipulators is utilized to design a general subtask controller. This controller uses the joint motion in the null-space of the Jacobian matrix to perform assigned subtasks. Specifically, objective function for each subtask is formed dependently. The only drawback of the designed controller was the need for the exact model knowledge of the dimensions and masses, but the ability of modeling using CAD programs such as SolidWorks®, model knowledge problem is restricted. Furthermore MATLAB® Simulink and embedded

SimMechanics CAD translator can be utilized for the simulation tests of the controller in the virtual environment before testing in real world. The kinematics and dynamics equations are required to be derived to be used in the controller equation and the next chapter gives fundamental background knowledge to begin the modeling and control of the redundant robot manipulator.

CHAPTER 3

BACKGROUND THEORY AND NOTATION

Fundamental background theory on robot kinematics and control, and notation that are used throughout the thesis are explained in this chapter to facilitate the understanding of the content of the latter chapters. Understanding the concept of rotation matrices is an essential part of modeling robot manipulators. This chapter describes rotational transformations in exponential form to define relative rotations of coordinate systems with respect to each other. Then, the parameters to define a robot mechanism are explained through Denavit-Hartenberg (DH) convention. Formulation of Homogenous Transformation Matrix (HTM) is explained by making use of the DH parameters to represent the position and orientation, which in total is called the pose, of the robot manipulator's end-effector with respect to the task coordinate system. The orientation of the end-effector is also expressed by an Euler's angles sequence; Roll, Pitch, and Yaw angles Representation. The rest of the chapter is dedicated to the concepts of feedback controllers, positive and negative definiteness, Pseudo-Inverse and *Singular Value Decomposition* (SVD), and presenting the philosophy of the *Lyapunov's direct method* for analyzing the stability of the dynamic system.

3.1. Kinematics Background

The next subsections give fundamental background needed to derive the kinematics and dynamics equations.

3.1.1. Exponential Form of Rotation Matrix

Consider a point *P* on the rigid body frame *B*, which its unit vectors are parallel with frame *G* unit vectors at the beginning. The location of point *P* is defined with a position vector $\bar{r}$ from the origin of frame *G*. If the rigid body has an angular velocity $\bar{\omega}$, then the velocity of *P* in the global coordinate frame is

$$\dot{\bar{r}} = \bar{\omega} \times \bar{r} = \tilde{\omega}\bar{r} \quad . \tag{3.1}$$

In Equation 3.1, $\tilde{\omega}$, represents the cross-product matrix corresponding to the unit column vector$\bar{\omega}$. This is a first-order linear differential equation that may be integrated to give

$$\bar{r}(t) = e^{\tilde{\omega}t}\bar{r}(0) \quad , \tag{3.2}$$

Where, $\bar{r}(0)$, is the initial position vector of *P*, and $e^{\tilde{\omega}t}$ is a matrix exponential as shown in Equation 3.3.

$$e^{\tilde{\omega}t} = \hat{I} + \tilde{\omega}t + \frac{(\tilde{\omega}t)^2}{2!} + \frac{(\tilde{\omega}t)^3}{3!} + \cdots \tag{3.3}$$

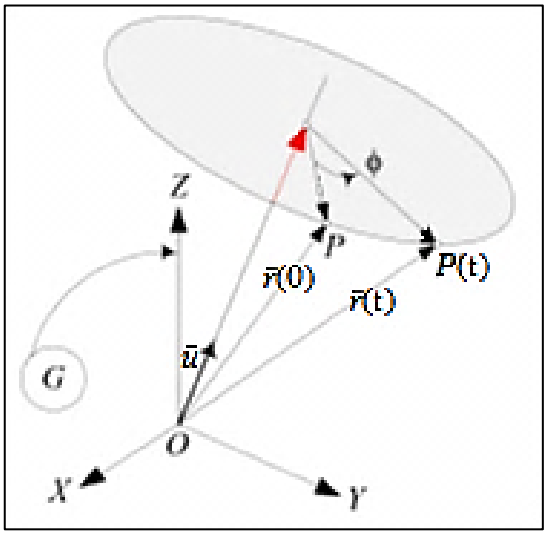


Figure 3.1. Axis and angle of rotation

The angular velocity $\bar{\omega}$, has a magnitude ω and direction indicated by a unit vector $\bar{u}$. Therefore,

$$\bar{\omega} = \omega\bar{u}$$
$$\tilde{\omega} = \omega\tilde{u}$$
$$\tilde{\omega}t = \omega t\tilde{u} = \phi\tilde{u} \tag{3.4}$$

and hence:

$$e^{\tilde{\omega}t} = e^{\phi\tilde{u}} = \hat{I} + \left(\phi - \frac{\phi^3}{3!} + \frac{\phi^5}{5!} - \cdots\right)\tilde{u} + \left(\frac{\phi^2}{2!} - \frac{\phi^4}{4!} + \frac{\phi^6}{6!}\cdots\right)\tilde{u}^2 \quad (3.5)$$

or equivalently

$$e^{\phi\tilde{u}} = \hat{I} + \tilde{u}sin\phi + \tilde{u}^2(1 - cos\phi) \quad (3.6)$$

is derived to represent a rotation in exponential form, where ϕ is the rotation angle, $\hat{I}$ is identity matrix and $\tilde{u}$ is the cross-product matrix corresponding to the unit column vector $\bar{u}$, which describes the axis of rotation. $\tilde{u}$ is generated from the components of $\bar{u}$ as follows:

$$\bar{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \rightarrow \tilde{u} = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \quad . \quad (3.7)$$

Equation 3.6 is an alternative form of the Rodriguez formula showing that $e^{\phi\tilde{u}}$ is the rotation transformation to map $B_{\bar{r}} = \bar{r}(0)\ to\ G_{\bar{r}} = \bar{r}(t)$. Therefore, $e^{\phi\tilde{u}}$ is equivalent to the rotation matrix ${}^{G}R_B = \hat{R}_{\bar{u},\phi}$. For proofs and more details on Rodriguez formula, one can read (Reza 2010).

3.1.2. Denavit-Hartenberg Convention in Exponential Form

DH convention parameters are shown in Figure 3.2. The symbols that appear in the figure are explained below:

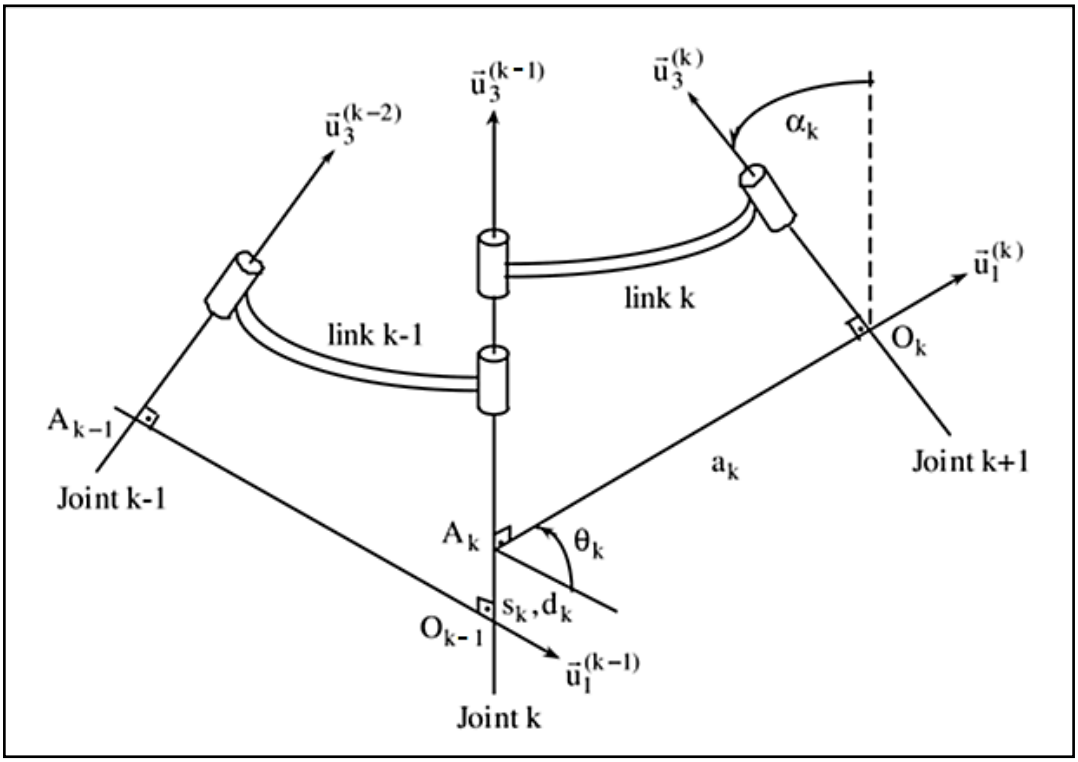


Figure 3.2. Sketch with Link Parameters of D-H Convention
(Source: Balkan *et al.* 2001)

O_k : Origin of the reference frame $\mathcal{F}_k$ attached to link *k.*

$\vec{u}_i^{(k)}$: Unit basis vectors of $\mathcal{F}_k$; *i*=1,2,3.

a_k : Effective length $A_k O_k$ of link *k* along $\vec{u}_1^{(k)}$.

S_k: Variable offset $O_{k-1}A_k$ of link *k* with respect to link *k*-1 along $\vec{u}_3^{(k-1)}$ if joint *k* is prismatic.

d_k: Constant offset $O_{k-1}A_k$ of link *k* with respect to link *k*-1 along $\vec{u}_3^{(k-1)}$ if joint *k* is revolute.

θ_k: Rotation angle of link *k* with respect to link *k*-1 about $\vec{u}_3^{(k-1)}$ if joint *k* is revolute.

δ_k: Rotation angle of link *k* with respect to link *k*-1 about $\vec{u}_3^{(k-1)}$ if joint *k* is prismatic.

α_k: twist angle of joint *k*+1 with respect to joint *k* about $\vec{u}_1^{(k)}$.

$\mathcal{F}_{k-1}^{(O_{k-1})}$: Reference frame fixed to link *k-1*

$\mathcal{F}_k^{(O_k)}$: Reference frame fixed to link *k*

To express position vector of point O_k with respect to point O_{k-1} on reference frame $\mathcal{F}_{k-1}^{(O_{k-1})}$

$$\begin{aligned}
\vec{r}_{k-1\,k} &= \vec{r}_{O_{k-1}O_k} \\
\bar{r}_{k-1\,k}^{(k-1)} &= \{\vec{r}_{k-1\,k}\}|_{\mathcal{F}_{k-1}^{(O_{k-1})}} \\
\bar{r}_{k-1\,k}^{(k-1)} &= s_k \bar{u}_3^{(k-1/k-1)} + a_k \bar{u}_1^{(k/k-1)} \\
\bar{r}_{k-1\,k}^{(k-1)} &= s_k \bar{u}_3 + a_k \hat{C}^{(k-1,\ k)} \bar{u}_1
\end{aligned} \tag{3.8}$$

A general exponential rotation matrix is expressed as in *Rodriguez formula*

$$e^{\tilde{u}\theta} = \hat{I}\, cos\,\theta + \tilde{u}\sin\theta + \bar{u}\bar{u}^T(1 - cos\,\theta). \tag{3.9}$$

The position equation can be simplified as shown in Equation 3.10 and 3.11.

$$\hat{C}^{(k-1,\ k)}\bar{u}_1 = e^{\tilde{u}_3\theta_k}e^{\tilde{u}_1\alpha_k}\bar{u}_1 = e^{\tilde{u}_3\theta_k}\bar{u}_1 = a_k\cos\theta_k\,\bar{u}_1 + a_k\sin\theta_k\,\bar{u}_2 \tag{3.10}$$

$$\bar{r}_{k-1\,k}^{(k-1)} = a_k\cos\theta_k\,\bar{u}_1 + a_k\sin\theta_k\,\bar{u}_2 + s_k\bar{u}_3. \tag{3.11}$$

According to the DH convention, the rotation matrix between two successive link frames can be expressed using exponential rotation matrices as;

$$\hat{C}^{(k-1,\ k)} = e^{\tilde{u}_3\theta_k}e^{\tilde{u}_1\alpha_k} = \begin{bmatrix} \cos\theta_k & -\sin\theta_k\cos\alpha_k & \sin\theta_k\sin\alpha_k \\ \sin\theta_k & \cos\theta_k\cos\alpha_k & -\cos\theta_k\sin\alpha_k \\ 0 & \sin\alpha_k & \cos\alpha_k \end{bmatrix} \quad . \tag{3.12}$$

If the homogeneous transformation matrix is represented as follows for a single link transformation;

$$\hat{H}^{(k-1,\ k)} = \hat{H}_{O_{k-1}O_k}^{(k-1,\ k)} = \begin{bmatrix} \hat{C}^{(k-1,\ k)} & \bar{r}_{k-1\,k}^{(k-1)} \\ 0\quad 0\quad 0 & 1 \end{bmatrix} \tag{3.13}$$

then the open form of the HTM can be derived as;

$$\widehat{H}^{(k-1,\ k)} = \begin{bmatrix} \cos\theta_k & -\sin\theta_k \cos\alpha_k & \sin\theta_k \sin\alpha_k & a_k \cos\theta_k \\ \sin\theta_k & \cos\theta_k \cos\alpha_k & -\cos\theta_k \sin\alpha_k & a_k \sin\theta_k \\ 0 & \sin\alpha_k & \cos\alpha_k & s_k \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (3.14)$$

This matrix expresses the pose of link *k* with respect to link *k*-1. This process can be extended for multi-link mechanism as a sequence of HTM multiplications which is represented in Equation 3.15 for a *k*-link manipulator.

$$\widehat{H}^{(0,k)} = \widehat{H}^{(0,1)}\widehat{H}^{(1,2)} \dots \widehat{H}^{(k-1,k)} \quad (3.15)$$

3.1.3. Roll, Pitch, and Yaw Angles Representation

A rotation matrix *R* can also be described as a product of successive rotations about the principal coordinate axes $\vec{u}_1, \vec{u}_2, and\ \vec{u}_3$ taken in a specific order. These rotations define the roll, pitch, and yaw angles, which are denoted by ϕ, θ, ψ as shown in Figure 3.3. The order of rotation are specified as $\vec{u}_1 - \vec{u}_2 - \vec{u}_3$, in other words, first a rotation about $\vec{u}_1$ by an angle ϕ, which is called yaw angle, then a rotation about the $\vec{u}_2$ by an angle θ, which is called pitch angle, and finally rotation about the $\vec{u}_3$ by an angle ψ, which is called roll angle. Since the successive rotations are relative to the fixed frame, successive rotation sequence is taken in the reverse order (Spong *et al*. 2005).

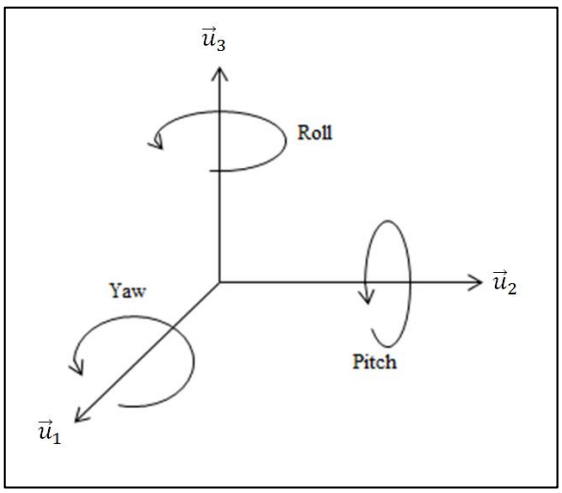


Figure 3.3. Roll, Pitch, and Yaw Angles.

Resulting transformation matrix is given by

$$
\begin{aligned}
R &= e^{\tilde{u}_3\phi} e^{\tilde{u}_2\theta} e^{\tilde{u}_1\psi} \\
&= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix} \\
&= \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix} \quad . \qquad (3.16)
\end{aligned}
$$

Instead of yaw-pitch-roll sequence relative to the fixed frames, the above transformation can be interpreted as roll-pitch-yaw sequence.

The three angles, $\phi, \theta \; and \; \psi$, can be obtained for a calculated rotation matrix as a result of successive rotation of each link of the mechanism. The method to calculate for these three angles is similar to deriving the Euler angles (Spong *et al*. 2005).

3.2. Linear System Theory and Control Background

The next subsections express the concepts of feedback controllers, positive and negative definiteness, Pseudo-Inverse and *Singular Value Decomposition* (SVD), and the philosophy of the *Lyapunov's direct method* for analyzing the stability of the dynamic system is presented.

3.2.1. Feedback Controllers

A control system can be configured as an open-loop or closed-loop control system. With an open-loop controller, the output of the controller is computed without observing the output of the controlled system. This type of a control strategy requires exact knowledge of the behavior of the system and assumes that there will not be any external disturbances. However, by configuring feedback controllers or closed-loop control systems, it might be possible to reject the external disturbances dynamically. A feedback controller observes the system output and calculates the error between this output and a reference value. Then, the controller output is computed based on this error

such that the output approaches the reference value. To achieve a desired behavior of the output, one conventional method is configuring controllers as a combination of the following control actions:

- **P** - *Proportional term*: The input is proportional to the error between the reference value and the current output. K_p is the proportional gain.
- **I** - *Integral term*: Integrates the error over time and multiplies with the integral gain K_i. The term eliminates steady state error.
- **D** - *Derivative term*: Determines the slope of the error over time and multiplies with the derivative gain K_v. The term has a damping effect.

The controller that is configured as a combination of these control elements needs to be tuned in order to guarantee stability or required tracking performance. Lyapunov's direct method is explained in this chapter for this purpose.

3.2.2. Positive and Negative Definiteness

In order to carry out stability analysis using Lyapunov's direct method, the positive and negative definiteness concepts are explained in the subsection. A function $V(x)$ satisfying the conditions $V(0) = 0$ and $V(x) > 0$ for $x \neq 0$ is said to be positive definite. If it satisfies $V(0) = 0$ and the weaker second condition $V(x) \geq 0$ for $x \neq 0$ it is said to be positive semidefinite. The function $V(x)$ is said to be negative definite if $-V(x)$ is positive definite, and negative semidefinite if $-V(x)$ is positive semidefinite. Function of the quadratic form is represented as;

$$V(x) = x^T P\, x \tag{3.17}$$

where P is a real symmetric matrix, and sign definiteness can easily be checked by inspecting P. If all eigenvalues of P are positive (nonnegative), which is true if and only if all the leading principal minors of P are positive (nonnegative), then $V(x)$ is a positive definite (positive semidefinite) function. If $V(x)$ is a positive definite (positive semidefinite) function, the matrix P is also said to be positive definite (positive semidefinite). Finally, P is considered to be negative definite if $-P$ is positive definite, and negative semidefinite if $-P$ is positive semidefinite (Siciliano *et al*. 2009).

It is common practice to write $P > 0$ if P is positive definite, and $P \geq 0$ if P is positive semidefinite.

3.2.3. Lyapunov Direct Method

The philosophy of the *Lyapunov's direct method* is the same as that of most methods used in control engineering to study stability, namely, testing for stability without solving the differential equations describing the dynamic system.

"This method can be presented in short on the basis of the following reasoning. If it is possible to associate an energy-based description with a (linear or nonlinear) autonomous dynamic system and, for each system state with the exception of the equilibrium state, the time rate of such energy is negative" (Siciliano *et al*. 2009).

Then energy decreases along any system trajectory until it attains its minimum at the equilibrium state; this argument justifies an intuitive concept of stability.

For $\dot{e} = f(e)$ by setting $f(0) = 0$, the *equilibrium state* is $e = 0$. A scalar function $V(e)$ of the system state, continuous together with its first derivative, is defined as a *Lyapunov function* if the following properties hold:

$V(e) > 0 \ \forall e \neq 0$;

$V(e) = 0 \quad e = 0$;

$\dot{V}(e) < 0 \ \ \forall e \neq 0$;

$V(e) \rightarrow \infty, ||e|| \rightarrow \infty$.

The existence of such a function ensures *global asymptotic stability* of the equilibrium $e = 0$. In practice, the equilibrium $e = 0$ is globally asymptotically stable if a positive definite, radially unbounded function $V(e)$ is found so that its time derivative along the system trajectories is negative definite. If positive definiteness of $V(e)$ is realized by the adoption of a *quadratic form, i.e.*,

$$V(e) = e^T Q e \tag{3.18}$$

with Q a symmetric positive definite matrix, then in view of $\dot{e} = f(e)$ it follows

$$\dot{V}(e) = 2e^T Q f(e). \tag{3.19}$$

If $f(e)$ is so as to render the function $\dot{V}(e)$ negative definite, the function $V(e)$ is a *Lyapunov function*, since the choice in Equation 3.18 allows system global asymptotic stability to be proved.

If $\dot{V}(e)$ in Equation 3.19 is not negative definite for the given $V(e)$, nothing can be inferred on the stability of the system, since the Lyapunov method gives only a *sufficient* condition. In such cases, one should resort to different choices of $V(e)$ in order to find, if possible, a negative definite $\dot{V}(e)$. In the case when the property of negative definiteness does not hold, but $\dot{V}(e)$ is only *negative semi-definite* $\dot{V}(e) \leq 0$, global asymptotic stability of the equilibrium state is ensured if the only system trajectory, for which $\dot{V}(e)$ is *identically* null ($\dot{V}(e) \equiv 0$), is the equilibrium trajectory $e \equiv 0$. This is a consequence of *La Salle theorem* (Siciliano *et al.* 2009).

3.2.4. Pseudo-Inverse

The inverse of a matrix can be defined only when the matrix is square and nonsingular. The inverse operation can be extended to the case of non-square matrices. Consider a matrix A of dimensions $m \times n$. If $m < n$, a *right inverse* of A can be defined as the matrix A_r of dimensions $n \times m$ so that

$$AA_r = I_m \ . \tag{3.20}$$

If instead $m > n$, a *left inverse* of A can be defined as the matrix A_l of dimensions $n \times m$ so that

$$A_l A = I_n \ . \tag{3.21}$$

it can be noted that I_m and I_n are both identity matrices with dimensions $m \times m$ and $n \times n$ respectively.

If A has more columns than rows, $m < n$, and has rank m, a special right inverse is the matrix

$$A_r^+ = A^T(AA^T)^{-1} \tag{3.22}$$

which is termed *right pseudo-inverse*, since $AA_r^+ = I_m$. If W_r is an $n \times n$ *positive definite* matrix, a *weighted* right pseudo-inverse is given by

$$A_r^+ = W_r^{-1}A^T(AW_r^{-1}A^T)^{-1} \quad . \tag{3.23}$$

If A has more rows than columns $m > n$ and has rank n, a special left inverse is the matrix

$$A_l^+ = (A^TA)^{-1}A^T \tag{3.24}$$

which is termed *left pseudo-inverse*, since $A_l^+A = I_n$. If W_l is an $m \times m$ *positive definite* matrix, a *weighted* left pseudo-inverse is given by

$$A_l^+ = (A^TW_lA)^{-1}A^TW_l \quad . \tag{3.25}$$

The pseudo-inverse is very useful to invert a linear transformation $y = Ax$ when A is a full-rank matrix. If A is a square nonsingular matrix, then obviously $x = A^{-1}y$ and then $A_l^+ = A_r^+ = A^{-1}$. If A has more columns than rows, $m < n$, and has rank m, then the solution x for a given y is not unique. This fact can be shown by the expression presented in Equation 3.26.

$$x = A^+y + (I - A^+A)k \; . \tag{3.26}$$

In Equation 3.26, k is an arbitrary, $n \times 1$ vector and A^+ is the inverse matrix defined in Equation 3.22. Therefore Equation 3.25 can be a solution to the system of linear equation established by:

$$y = Ax \; . \tag{3.27}$$

The term $A^+y \in \mathcal{N}^{\perp}(A) \equiv R(A^T)$ minimizes the norm of the solution $||x||$. The term $(I - A^+A)k$ is the projection of k in $\mathcal{N}(A)$ and is termed *homogeneous solution*. As k varies, all the solutions to the homogeneous equation system, $Ax = 0$, associated with Equation 3.27 are generated.

On the other hand, if A has more rows than columns, $m > n$, then no solution exist for Equation 3.27. An *approximate* solution is given by

$$x = A^{+}y \tag{3.28}$$

where A^{+} as in Equation 3.22 minimizes $||y - Ax||$. If instead $y \in R(A)$, then Equation. 3.28 is a real solution.

Notice that the use of the weighted (left or right) pseudo-inverses in the solution for the linear equation systems leads to analogous results where the minimized norms are weighted according to the metrics defined by matrices W_r and W_l, respectively (Siciliano *et al*. 2009).

3.2.5. Singular Value Decomposition (SVD)

It is not possible to define eigenvalues for a non-square matrix. An extension of the eigenvalue concept can be obtained by singular values of a non-square matrix. Given a matrix A of dimensions, $m \times n$, the matrix $A^{\mathrm{T}}A$ has n nonnegative eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ (ordered from the largest to the smallest) which can be expressed in the form

$$\lambda_i = \sigma_i^2, \qquad \sigma_i \geq 0. \tag{3.29}$$

The scalars $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$ are said to be the *singular values* of matrix A. The SVD of matrix A is given by

$$A = U\Sigma V^{T} \tag{3.30}$$

where U is an $m \times m$ orthogonal matrix

$$U = [u_1\ u_2 \ldots u_m], \tag{3.31}$$

V is an $n \times n$ orthogonal matrix

$$V = [v_1\ v_2\ \ldots\ v_n\ ] \tag{3.32}$$

and Σ is an $m \times n$ matrix

$$\Sigma = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \qquad D = diag\{\sigma 1, \sigma 2, \ldots, \sigma r\} \tag{3.33}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. The number of non-null singular values is equal to the rank r of matrix A.

The columns of U are the eigenvectors of the matrix AA^T, whereas the columns of V are the eigenvectors of the matrix A^TA. In view of the partitions of U and V in Equations 3.31 and 3.32, it is $Av_i = \sigma_i u_i$, for $i = 1, \ldots, r$ and $Av_i = 0$, for $i = r + 1, \ldots, n$.

SVD is useful for the analysis of the linear transformation as will be seen in Chapter 5. According to a geometric interpretation, the matrix A transforms the unit sphere in $\mathbb{R}^n$ defined by $||x|| = 1$ into the set of vectors $y = Ax$ which define an *ellipsoid* of dimension r in $\mathbb{R}^m$. The singular values are the lengths of the various axes of the ellipsoid. The *condition number* of the matrix

$$k_\sigma = \frac{\sigma_1}{\sigma_r} \tag{3.34}$$

is related to the eccentricity of the ellipsoid and provides a measure of ill-conditioning ($\kappa >> 1$) for numerical solution of the system established by Equation 3.27, (Siciliano *et al.* 2009).

It is worth noticing that the numerical procedure of SVD is commonly adopted to compute the (right or left) pseudo inverse A^+, even in the case of a matrix A not having full rank. In fact, from Equations 3.30-3.32 it is

$$A^+ = V\Sigma^+U^T \tag{3.35}$$

with

$$\Sigma^+ = \begin{bmatrix} D^+ & 0 \\ 0 & 0 \end{bmatrix} \qquad D^+ = diag\left\{\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \cdots, \frac{1}{\sigma_r}\right\} \quad . \tag{3.36}$$

CHAPTER 4

MODELING OF THE ROBOT MANIPULATOR

SCHUNK 7-DOF LWA4-Arm shown in Figure 4.1 is chosen to be modeled for simulation purposes for this thesis study. This robot manipulator arm has a modular configuration which makes it flexible for use in different robot applications such as industrial and service robotics. This robot arm is a lightweight manipulator that has seven DOF where the joint rotation axes are arranged to intersect at a shoulder point, an elbow, and a wrist point.

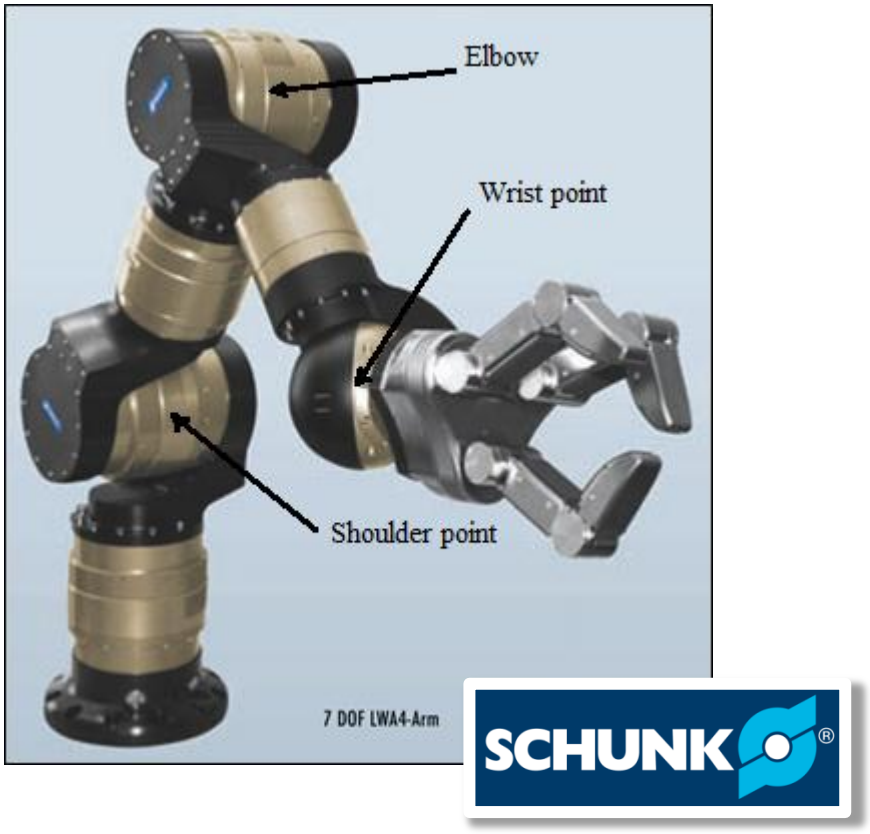


Figure 4.1. SCHUNK 7-DOF LWA4-Arm

4.1. Computer-Aided Modeling of the Robot Arm

Computer-aided modeling is done using the virtual prototyping of robot controllers' method (Dede 2010) in two stages. First, robot arm is modeled and reassembled in SolidWorks computer-aided-design (CAD) software with respect to the CAD data provided in (Schunk modular robotics 2011) as shown in Figure 4.2. COSMOSMotion module of SolidWorks is then used to develop the mechanism by configuring the joint structures in CAD environment. Then, the CAD model is exported in 3D XML format by using the plug-in, SimMechanics Link, to MATLAB® Simulink. MATLAB Simulink, which is a block diagram modeling environment for the engineering design and simulation of rigid multibody machines and their motions, using the standard Newtonian dynamics of forces and torques, is used for simulation studies.

Figure 4.2. Robot manipulator arm CAD model

The robot manipulator arm is modeled with a suite of tools to specify various issues like; bodies and their mass properties, their possible motions, kinematic constraints, and coordinate systems. Furthermore to initiate and measure body motions

with respect to the physical criteria. Like other Simulink models, the system is represented by a set of connected block diagram as illustrated in Figure 4.3. As a result of the transfer of the model from CAD environment to SimMechanics software, the model could be used for simulation studies that are conducted to test the controller.

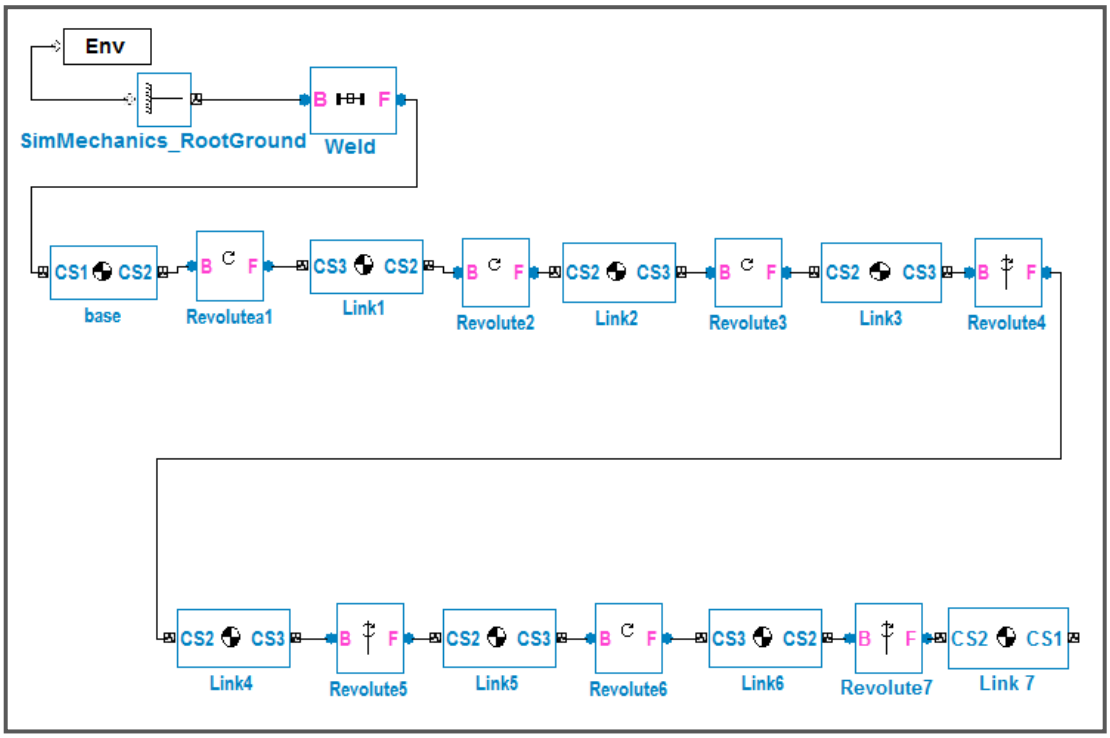


Figure 4.3. Block diagram of LWA4

The second stage includes the modeling of the control system and development of the necessary kinematics and dynamics equations for the robot using MATLAB® Simulink blocks. The subsystems created for the necessary calculations are shown in Figure 4.4. The subsystems represent the calculation of the Jacobian matrix for the end-effector and its inverse. The calculations of inertia matrix and gravity vector needed for the equation of motion are also built in Simulink which are denoted with "Inertia Matrix" and "Gravity Matrix" subsystem titles in Figure 4.4. The necessary feedback signals are built in sensor reading block which is denoted with "sensor reading". The main-task controller, subtask controller and end-effector trajectory are also created as will be shown throughout this work. The visualization tool of MATLAB Simulink is also used to display and animate 3D machine geometries, before and during simulation.

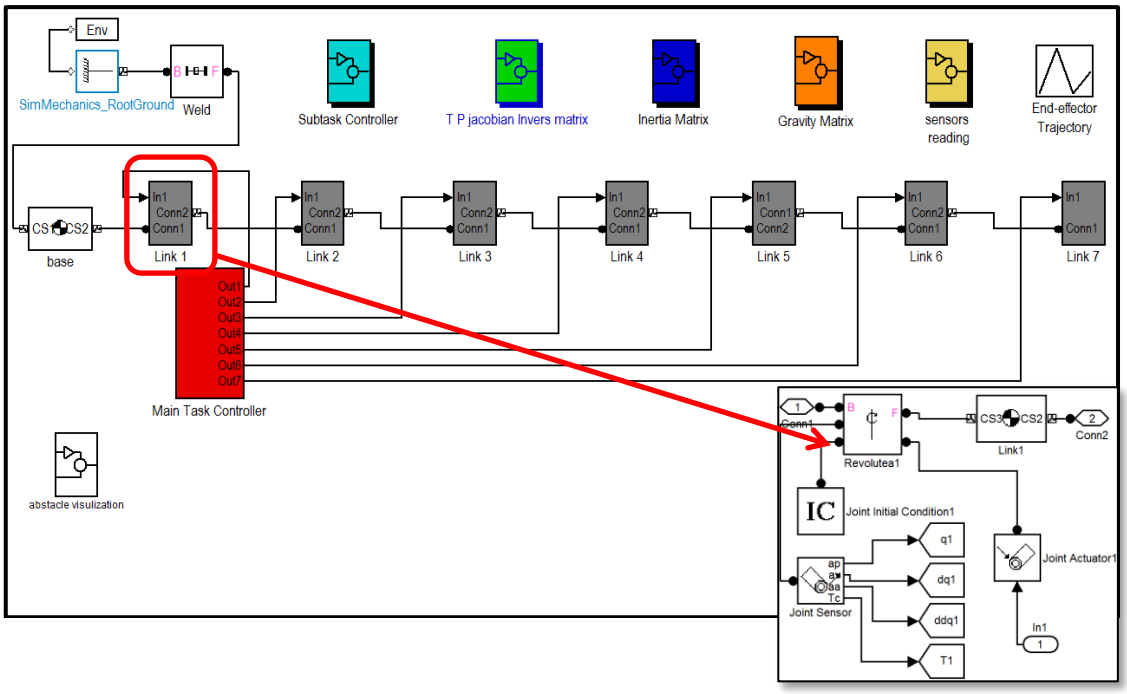


Figure 4.4. Block diagram of LWA4 and control system

4.2. Deriving the Robot Arm Equations

In this subsection, kinematic and dynamic analyses are conducted to derive the robot arm equations. The first part contains the derivation of the robot arm kinematics and the next part is reserved for the dynamics equations of the robot arm.

4.2.1. Robot Arm Kinematics

The problem of kinematics is to describe the motion of the manipulator without consideration of the forces and torques causing the motion. The kinematic description is therefore a geometric one. A large part of robot kinematics is concerned with the establishment of various coordinate systems to represent the positions and orientations of rigid objects, and with transformations among these coordinate systems. Indeed, the geometry of three-dimensional space and of rigid motions plays a central role in all aspects of robotic manipulation. The proposed robot manipulator arm is composed of a set of links connected to each other by revolute joints. In this section, *Forward Kinematic Analysis* is carried out according to DH convention. Then, velocity equations

are derived for wrist point, tip point of the manipulator (end-effector) and mass center point for each link.

4.2.1.1. Forward Kinematics Analysis

The objective of forward kinematic analysis is to determine the cumulative effect of the entire set of joint variables, that is, to determine the position and orientation of the end-effector given the values of these joint variables. To perform the kinematic analysis, we attach a coordinate frame rigidly to each link. In particular, we attach $\mathcal{F}_i - \vec{u}_1^{(i)}\vec{u}_2^{(i)}\vec{u}_3^{(i)}$, to link i. This means that whatever motion the robot executes, the coordinates of each point on link i, are constant when expressed in the i^{th} coordinate frame $\mathcal{F}_i$. Furthermore, when revolute joint R_i is actuated, link i and its attached frame, $\mathcal{F}_i - \vec{u}_1^{(i)}\vec{u}_2^{(i)}\vec{u}_3^{(i)}$, experience a resulting motion. The frame $\mathcal{F}_0 - \vec{u}_1^{(0)}\vec{u}_2^{(0)}\vec{u}_3^{(0)}$, which is attached to the robot base, is referred to as the base frame as shown in Figure 4.5. The gravitational effect vector is $\bar{g} = [0 \;\; 0 \;\; -9.81]^T \; (m/s^2)$.

It is possible to carry out forward kinematics analysis without using the DH convention. However, the kinematic analysis of an n-link manipulator can be extremely complex, and the DH convention introduced in Sub-section 3.2.2 simplifies the analysis considerably.

The DH convention is a commonly used convention, for selecting reference frames in robotic applications that eases the communication between robot engineers. After assigning the coordinate frames, HTM, $\widehat{H}$, for each link will be configured. DH parameters will be used to formulate the HTM for each link. DH parameters, for the proposed redundant arm manipulator demonstrated in Figure 4.5, are represented in Table 4.1.

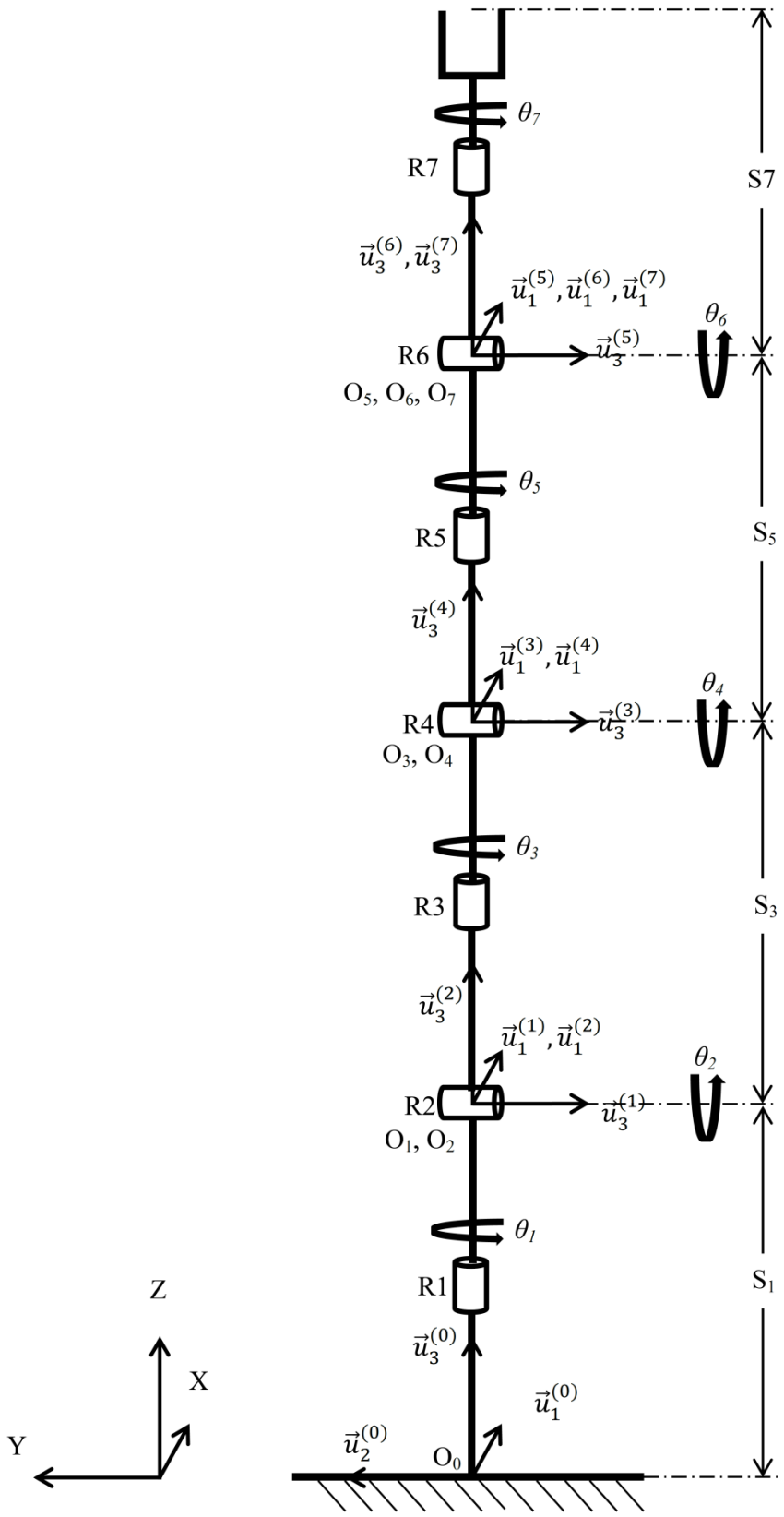


Figure 4.5. LWA4 Robot arm joint schematic

Table 4.1. Denavit-Hartenberg convention

i	θ_i	d_i (m)	a_i (m)	α_i (rad)
1	θ_1	0.3	0	π/2
2	θ_2	0	0	- π/2
3	θ_3	0.328	0	π/2
4	θ_4	0	0	- π/2
5	θ_5	0.317248	0	π/2
6	θ_6	0	0	- π/2
7	θ_7	0	0	0

In the DH convention the only variable is θ_i, therefore the notation simplified by writing *ci* for *cos θi* and *si* for *sin θi*. The distance from the end-effector to the wrist point is selected to be P_e = 0.0757 m.

According to DH convention, the components of HTM for each link will be calculated as follows:

$$\begin{aligned}
&\hat{C}^{(0,1)} = e^{\tilde{u}_3\theta_1}e^{\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{01}^{(0)} = d_1\bar{u}_3^{(0)},\\
&\hat{C}^{(1,2)} = e^{\tilde{u}_3\theta_2}e^{-\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{12}^{(1)} = \bar{0},\\
&\hat{C}^{(2,3)} = e^{\tilde{u}_3\theta_3}e^{\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{23}^{(2)} = d_3\bar{u}_3^{(2)},\\
&\hat{C}^{(3,4)} = e^{\tilde{u}_3\theta_4}e^{-\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{34}^{(3)} = \bar{0},\\
&\hat{C}^{(4,5)} = e^{\tilde{u}_3\theta_5}e^{\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{45}^{(4)} = d_5\bar{u}_3^{(4)},\\
&\hat{C}^{(5,6)} = e^{\tilde{u}_3\theta_6}e^{-\tilde{u}_1\frac{\pi}{2}}, \quad \bar{r}_{56}^{(5)} = \bar{0},\\
&\hat{C}^{(6,7)} = e^{\tilde{u}_3\theta_7}, \quad \bar{r}_{23}^{(2)} = \bar{0},
\end{aligned} \tag{4.1}$$

where the column representations of the unit basis vectors are

$$\bar{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad \bar{u}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \qquad \bar{u}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

The base-to-hand rotation matrix, $\hat{C}^{(0,7)}$, is obtained as a successive multiplication of seven matrices after the necessary simplification are conducted;

$$\hat{C}^{(0,7)} = \hat{C}^{(0,1)}\hat{C}^{(1,2)} \dots \hat{C}^{(6,7)} = e^{\tilde{u}_3\theta_1}e^{-\tilde{u}_2\theta_2}e^{\tilde{u}_3\theta_3}e^{-\tilde{u}_2\theta_4}e^{\tilde{u}_3\theta_5}e^{-\tilde{u}_2\theta_6}e^{\tilde{u}_3\theta_7}. \qquad (4.2)$$

HTM representation for each link can be expressed as;

$$\left.\begin{aligned}
\widehat{H}^{(0,1)} &= \begin{bmatrix} \hat{C}^{(0,1)} & \bar{r}_{01}^{(0)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(1,2)} &= \begin{bmatrix} \hat{C}^{(1,2)} & \bar{r}_{12}^{(1)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(2,3)} &= \begin{bmatrix} \hat{C}^{(2,3)} & \bar{r}_{23}^{(2)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(3,4)} &= \begin{bmatrix} \hat{C}^{(3,4)} & \bar{r}_{34}^{(3)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(4,5)} &= \begin{bmatrix} \hat{C}^{(4,5)} & \bar{r}_{45}^{(4)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(5,6)} &= \begin{bmatrix} \hat{C}^{(5,6)} & \bar{r}_{56}^{(5)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \\
\widehat{H}^{(6,7)} &= \begin{bmatrix} \hat{C}^{(6,7)} & \bar{r}_{67}^{(6)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}.
\end{aligned}\right\} \qquad (4.3)$$

HTM between the base frame and the last link frame can be represented as a successive multiplication of seven HTMs; $\widehat{H}^{(0,7)} = \widehat{H}^{(0,1)}\widehat{H}^{(1,2)} \dots \widehat{H}^{(6,7)}$, and $\widehat{H}^{(0,7)}$ can be represented as;

$$\widehat{H}^{(0,7)} = \begin{bmatrix} & & & d_x \\ & \hat{C}^{(0,7)} & & d_y \\ & & & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad . \qquad (4.4)$$

Wrist-point position is shown in the open form in Equation 4.5.

$$\bar{r} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} = d_1\bar{u}_3^{(0)} + d_3\hat{C}^{(0,2)}\bar{u}_3^{(2)} + d_5\hat{C}^{(0,4)}\bar{u}_3^{(4)}$$
$$= d_1\bar{u}_3 + e^{\tilde{u}_3\theta_1}[(-d_3s2 - d_5c4s2 - d_5s4c3c2)\bar{u}_1 + (-d_5s4s3)\bar{u}_2$$
$$+(d_3c2 + d_5c4c2 - d_5s4c3s2)\bar{u}_3] \quad (4.5)$$

If the tip point of the end-effector is selected to be in the direction of the approach vector, $\bar{u}_3^{(7)} = \bar{u}_a$, then the tip-point position can be calculated as,

$$\bar{P} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \bar{r} + P_e\hat{C}^{(0,7)}\bar{u}_3^{(7)} \quad , \quad (4.6)$$

or

$$\bar{P} = \hat{H}^{(0,7)}\begin{bmatrix} 0 \\ 0 \\ P_e \\ 1 \end{bmatrix} \quad . \quad (4.7)$$

4.2.1.2. Derivation of the Jacobian Matrix

The LWA4 manipulator with 7-DOF is considered with joint variables $q1, q2, \ldots, q7$. Let $\hat{H}^{(0,7)}$ denote the transformation from the base frame to the end-effector frame, where $q = [q1 \ldots q7]^T$ is the vector of joint variables. As the robot is in motion, the joint variables, q_i, the end-effector position, $\bar{P}^{(0)}$, and orientation of the end-effector, $\hat{C}^{(0,7)}$, will be functions of time. The objective of this section is to relate the linear and angular velocity of the end-effector to the vector of joint velocities $\dot{q}(t)$. Let

$$v = J_v\dot{q} \quad (4.8)$$

and

$$\omega = J_\omega \dot{q} \quad , \tag{4.9}$$

where $v \in \Re^3$ denotes the linear velocity of the end-effector and $\omega \in \Re^3$ defines the angular velocity vector of the end-effector where the portions of the Jacobian matrix for the linear and angular velocity are denoted respectively as J_v and $J_\omega \in \Re^{3\times7}$. The Equations 4.8 and 4.9 combined as:

$$\dot{x} = J\dot{q} \ , \tag{4.10}$$

in which both $\dot{x}$ and J are given by

$$\dot{x} = \begin{bmatrix} v \\ \omega \end{bmatrix} \qquad , \qquad J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix} \quad . \tag{4.11}$$

The above formulas make the determination of the Jacobian matrix of any manipulator simple since all of the quantities needed are available once the forward kinematics analysis is carried out. The end-effector position is calculated from Equation 4.7. However, the position will be the first three rows of the vector as $\bar{P}^{(0)} \in \Re^3$. J_p will be denoted as the Jacobian matrix of the end-effector and it will be calculated in linear and angular portions as:

$$\hat{J}_{pv} = \begin{bmatrix} \frac{\partial \bar{P}^{(0)}}{\partial \theta_1} & \frac{\partial \bar{P}^{(0)}}{\partial \theta_2} & \cdots & \frac{\partial \bar{P}^{(0)}}{\partial \theta_7} \end{bmatrix} \tag{4.12}$$

and

$$\bar{J}_{p\omega i} = col[\frac{\partial \hat{C}}{\partial \theta_i} \hat{C}^T], \tag{4.13}$$

$$\hat{J}_{p\omega} = \begin{bmatrix} \bar{J}_{p\omega 1} & \bar{J}_{p\omega 2} & \cdots & \bar{J}_{p\omega 7} \end{bmatrix}. \tag{4.14}$$

The portions of the Jacobian matrix calculated for the end-effector can then be combined as shown in Equation 4.15.

$$\hat{J}_p = \begin{bmatrix} \hat{J}_{pv} \\ \hat{J}_{p\omega} \end{bmatrix}. \tag{4.15}$$

The above procedure works not only for computing the velocity of the end-effector but also for computing the velocity of any point on the manipulator by calculating for J_{ci}, the Jacobian matrix at mass center point at link *i*. This information will be required in Sub-section 4.3.2.1 when the velocities of the mass centers of various links are to be calculated in order to derive the dynamic equations of motion. This information will also be used for obstacle avoidance subtask.

4.2.2. Derivation of the Dynamic Equation of Motion

The dynamic equations explicitly describe the relationship between force and motion whereas the kinematic equations describe the motion of the robot without consideration of the forces and torques producing the motion. The equation of motion is important to consider in the design of robots, in simulation and animation of robot motion, and in the design of control algorithms.

Euler-Lagrange equation to derive the equation of motion is often used for numerical calculation. The general form of Euler-Lagrange equations of motion is represented in Equation 4.16. The forces/torques required for the actuation of the system is denoted with τ_i in this equation

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \quad i = 1, \ldots, n \ . \tag{4.16}$$

The order, *n*, of the system is determined by the number of so-called **generalized coordinates** (Spong *et al.* 2005) that are required to describe the evolution of the system. In robotics, the order of the system is determined by the DOF of the robot arm.

$$L = K - P \tag{4.17}$$

The function *L*, which is the difference of the kinetic and potential energy, is called the Lagrangian term for the system. The Euler-Lagrange equations provide a formulation of the dynamic equations of motion equivalent to those derived using Newton's Second Law.

4.2.2.1. Kinetic Energy for an n-Link Robot

The kinetic energy of the rigid body is given as;

$$K = \frac{1}{2} m v^T v + \frac{1}{2} \omega^T I \omega \ , \tag{4.18}$$

where *m* is a mass and *I* is inertia matrix.

The linear and angular velocities of any point on any link can be expressed in terms of the Jacobian matrix and the derivative of the joint variables as explained in Sub-section 4.3.1.2.

In the analyses of robot arms, the joint variables are indeed the generalized coordinates. Therefore, for Jacobian sub-matrices calculated for each link, J_{vi} and $J_{\omega i}$, being functions of joint variables and time, the linear and angular velocity equations are formulated as

$$v_i = J_{v_i}(q)\dot{q} \quad , \quad \omega_i = J_{\omega_i}(q)\dot{q} \quad . \tag{4.19}$$

It is supposed that the mass of link *i* is m_i and the inertia matrix of link *i* evaluated around the coordinate frame parallel to frame *i* with the origin at the center of mass is I_i. Making use of Equation 4.19, the overall kinetic energy of the manipulator can be calculated as

$$K = \frac{1}{2} \dot{q}^T \sum_{i=1}^{n} [m_i J_{v_i}(q)^T J_{v_i}(q) + J_{\omega_i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega_i}(q)] \, \dot{q} \ . \tag{4.20}$$

In other words, the kinetic energy of the manipulator is formulated in the form

$$K = \frac{1}{2} \sum_{i,j}^{n} M_{ij}(q) \dot{q}_i \dot{q}_j = \frac{1}{2} \dot{q}^T M(q) \dot{q} \tag{4.21}$$

In Equation 4.21, $M(q)$ is a symmetric positive definite matrix that is in general configuration dependent. The matrix M is called the inertia matrix.

4.2.2.2. Potential Energy for an n-Link Robot

In the case of rigid dynamics, the only source of potential energy is gravity. The potential energy of the i^{th} link can be computed by assuming that the mass of the entire object is concentrated at its center of mass and is given by

$$P_i = \bar{g}^T . \bar{r}_{ci} m_i \quad . \tag{4.22}$$

In Equation 4.22, $\bar{g}$ vector contains the direction and magnitude information of gravity in the base frame and the vector $\bar{r}_{ci}^{(0)}$ provides the coordinates of the center of mass of link i with respect to base frame. The total potential energy of the n-link robot is therefore

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} \bar{g}^T . \bar{r}_{ci}^{(0)} m_i. \tag{4.23}$$

In the case that the robot contains elasticity, for example, flexible joints, then the potential energy will include terms containing the energy stored in the elastic elements. It should be noted that the potential energy is a function only of the generalized coordinates and not their derivatives, *i.e.* the potential energy depends on the configuration of the robot but not on its velocity.

4.2.2.3. Euler-Lagrange Equation

The Euler-Lagrange equations for such a system can be derived as follows. Since

$$L = K - P = \frac{1}{2} \sum_{i,j} M_{ij}(q) \dot{q}_i \dot{q}_j - P(q), \tag{4.24}$$

we have that :

$$\frac{\partial L}{\partial \dot{q}_k} = \sum_j M_{kj}\dot{q}_j, \tag{4.25}$$

and

$$\begin{aligned}\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} &= \sum_j M_{kj}\ddot{q}_j + \sum_j \frac{d}{dt}M_{kj}\dot{q}_j \\ &= \sum_j M_{kj}\ddot{q}_j + \sum_{i,j}\frac{\partial M_{kj}}{\partial q_i}\dot{q}_i\,\dot{q}_j,\end{aligned} \tag{4.26}$$

also

$$\frac{\partial L}{\partial q_k} = \frac{1}{2}\sum_{i,j}\frac{\partial M_{ij}}{\partial q_k}\dot{q}_i\,\dot{q}_j - \frac{\partial P}{\partial q_k}\,. \tag{4.27}$$

Thus, the Euler-Lagrange equations can be written

$$\sum_j M_{kj}\ddot{q}_j + \sum_{i,j}\left\{\frac{\partial M_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial M_{ij}}{\partial q_k}\right\}\dot{q}_i\,\dot{q}_j - \frac{\partial P}{\partial q_k} = \tau_k. \tag{4.28}$$

By interchanging the order of summation and taking advantage of symmetry, we can show that

$$\sum_{i,j}\left\{\frac{\partial M_{kj}}{\partial q_i}\right\}\dot{q}_i\,\dot{q}_j = \frac{1}{2}\sum_{i,j}\left\{\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j}\right\}\dot{q}_i\,\dot{q}_j, \tag{4.29}$$

hence:

$$\sum_{i,j}\left\{\frac{\partial M_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial M_{ij}}{\partial q_k}\right\}\dot{q}_i\,\dot{q}_j = \sum_{i,j}\frac{1}{2}\left\{\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j} - \frac{\partial M_{ij}}{\partial q_k}\right\}\dot{q}_i\,\dot{q}_j, \tag{4.30}$$

Christoffel symbols of the First Kind:

$$c_{ijk} = \frac{1}{2}\sum_{i,j}\left\{\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j} - \frac{\partial M_{ij}}{\partial q_k}\right\}. \tag{4.31}$$

The terms c_{ijk} in Equation 4.31 are known as Christoffel symbols (of the first kind) see in (Spong *et al.* 2005). Note that, for a fixed *k*, we have $c_{ijk} = c_{jik}$, which reduces the effort involved in computing these symbols by a factor of about one half. Finally, for the gravitational term

$$G_k = \frac{\partial P}{\partial q_k} \quad . \tag{4.32}$$

Then we can write the Euler-Lagrange equations as:

$$\sum_i M_{kj}(q)\ddot{q}_j + \sum_{i,j} c_{ijk}(q)\dot{q}_i\dot{q}_j + G_k(q) = \tau_k \quad , k = 1, \cdots, n. \tag{4.33}$$

In the above equation, there are three types of terms. The first involve the second derivative of the generalized coordinates. The second are quadratic terms in the first derivatives of *q*, where the coefficients may depend on $\dot{q}$. These are further classified into two types as terms involving a product of the type ${\dot{q}_i}^2$ which are called centrifugal and terms involving a product of the type $\dot{q}_i\dot{q}_j$ where $i \neq j$ which are called Coriolis terms. The third type of terms is involving only *q* but not its derivatives. Clearly the latter arise from differentiating the potential energy. It is common to write Equation 4.33 in matrix form, from Euler-Lagrange equations:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \quad , \tag{4.34}$$

where the elements of $C(q,\dot{q})$ matrix is defined as:

$$c_{kj} = \sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i = \sum_{i=1}^{n} \frac{1}{2}\left\{\frac{\partial M_{kj}}{\partial q_j} + \frac{\partial M_{ki}}{\partial q_j} - \frac{\partial M_{ij}}{\partial q_k}\right\}\dot{q}_i \quad . \tag{4.35}$$

CHAPTER 5

REDUNDANT ROBOT MANIPULATOR CONTROL

In this chapter, the inverse kinematic of redundant robot manipulator is calculated using pseudo-inverse. Redundancy resolution at acceleration level is presented. Computed-torque control output signal needed to perform the main-task and subtask objective is designed. Designing process and derivation of control signals are revealed. Furthermore the techniques of selecting the objective functions for the subtasks considered in this thesis are illustrated. the subtasks considered for this study are; Minimizing the total joint motion, singularity avoidance, posture optimization for the static impact force objectives, which include maximizing/minimizing the static impact force magnitude, and static and moving obstacle (point to point) collision avoidance.

5.1. Kinematics Model of Redundant Robot Manipulator

Redundant manipulators have a larger number of DOF, *n*, than the dimension of the workspace *m*. The end-effector position and orientation in the task space, denoted by $x(t) \in \Re^m$, is defined as a function of joint position vector as:

$$x = k(q) = \begin{bmatrix} p(q) \\ \phi(q) \end{bmatrix}, \tag{5.1}$$

Where $k(q) \in \Re^m \;\; m \in \aleph$ is the forward kinematics calculation, $q(t) \in \Re^n$ denote the link position vector of an *n*-link manipulator, $p(q) \in \Re^l$ and $\phi(q) \in \Re^{(m-l)}$ are the vectors representing the end-effector position, and orientation respectively and $l \in \aleph$ is the size of the task space for positioning.

Based on Equation 5.1, relationships between the end-effector velocity and acceleration and the joint variables' velocities and accelerations are obtained as follows:

$$\dot{x} = J(q)\dot{q} \tag{5.2}$$

$$\ddot{x} = \dot{J}(q)\dot{q} + J(q)\ddot{q} \quad , \tag{5.3}$$

where $J(q) = \partial k(q)/\partial q \in \Re^{m\times n}$, is the Jacobian matrix of the manipulator and d $\dot{q}(t), \ddot{q}(t) \in \Re^n$ denote the link velocities and acceleration vectors, respectively. Since *J* is not a square matrix for redundant manipulators, where *m* < *n*, the *psuedo-inverse*, J^+ defined in Equation 5.7, can be used to obtain the inverse relations as can be depicted in Equations 5.4 and 5.5.

$$\dot{q} = J^+\dot{x} + \dot{\theta}_N \tag{5.4}$$

$$\ddot{q} = J^+\left(\ddot{x} - \dot{J}\dot{q}\right) + \ddot{\theta}_N \tag{5.5}$$

In the above equations, both $\dot{\theta}_N$ and $\ddot{\theta}_N$ are vectors of joints' velocity and acceleration in the null space of *J* respectively. The psuedo-inverse J^+, is defined as the unique matrix such that (Golub 1983, Yoshikawa 1984)

$$\begin{aligned} JJ^+J = J \quad &, \; J^+JJ^+ = J^+, \\ (J^+J)^T = J^+J \quad &, (JJ^+)^T = JJ^+ \; . \end{aligned} \tag{5.6}$$

When *J* has full rank (the manipulator is not in a singular configuration), the pseudo-inverse can be calculated as;

$$J^+ = J^T(JJ^T)^{-1} \quad , \tag{5.7}$$

so that J^+ satisfies $JJ^+ = I_m$, where I_m is *m* x *m* identity matrix.

5.2. Dynamic Model of Redundant Robot Manipulator

The dynamic model for an *n*-link, all revolute-joint robot manipulator is developed in the following form (Spong *et al.* 1989).

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \xi_d = \tau \tag{5.8}$$

Equation 5.8 is the extended version of Equation 4.35 with the addition of external effects where $M(q) \in \Re^{n\times n}$ represents the inertia matrix, $C(q,\dot{q}) \in \Re^{n\times n}$ represents the Centripetal-Coriolis matrix, $G(q) \in \Re^{n}$ is the gravity vector, $F(\dot{q}) \in \Re^{n}$ represents the friction effects vector, $\xi_d \in \Re^{n}$ is a vector containing the unknown but bounded, additive disturbance effects and $\tau(t) \in \Re^{n}$ is the torque input vector.

5.3. The Control Problem

Various techniques can be employed for controlling a robot manipulator. The technique may have a major influence on the manipulator performance and the possible range of applications, depending on the way it is followed and implemented. For instance, the need for trajectory tracking control in the task space may lead to software/hardware implementations with respect those allowing point-to-point controls, where only the final position is of concern.

Independent of the type of mechanical manipulator, it is worth stating that task specification (end-effector motion and forces) is usually performed in the task space, whereas control actions (joint actuator generalized forces) are carried out in the joint space. This fact leads to considering two kinds of general control schemes (Siciliano *et al.* 2009), namely, a *joint space control* scheme Figure 5.1 and *task space control* scheme Figure 5.2. In both schemes, the control structure has closed-loops to exploit the needed features provided by feedback to control the robot arm and execute the required tasks. In general terms, the following considerations are stated in the next paragraph should be made.

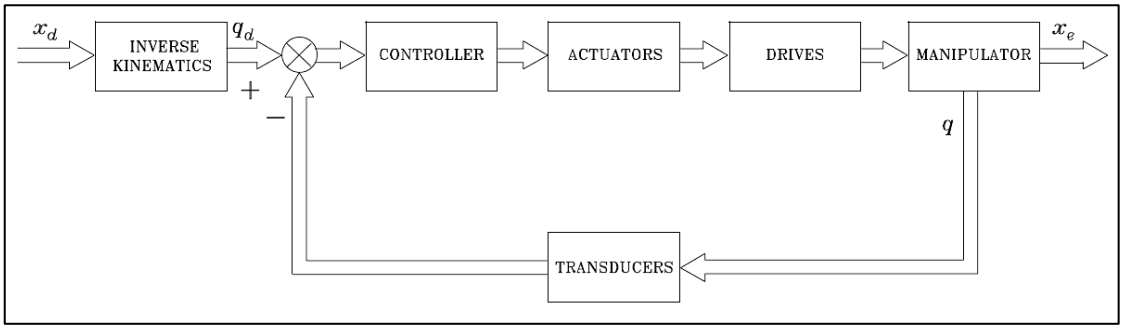


Figure. 5.1. General scheme of joint space control
(Source: Siciliano *et al.* 2009)

The *joint space control* problem first makes use of the manipulator inverse kinematics solution to transform the motion requirements, x_d, from the task space into the corresponding motion, q_d, in the joint space. Then, a joint space control scheme is designed that allows the actual motion, q, to track the reference inputs. However, this solution has two drawbacks. First, in redundant robot manipulators for the desired motion in the task space, x_d, there are an infinity number of the corresponding motion possibilities in the joint space, q_d. However, by using pseudo-inverse we can find minimum joint velocities $\dot{q}_d$, when the null space motion is zero, but q_d is also needed to apply PD control which is hard to find, especially when the motion in the null space is included. Second, the joint space control scheme does not influence the task space variables x, which are controlled in an open-loop fashion through the manipulator mechanical structure.

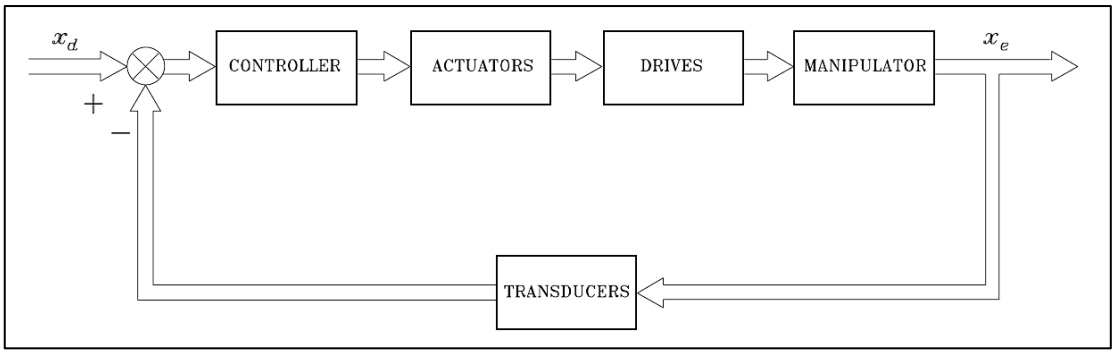


Figure. 5.2. General scheme of task space control
(Source: Siciliano *et al.* 2009)

It is then clear that any uncertainty of the structure (construction tolerance, lack of calibration, gear backlash, elasticity) or any imprecision in the knowledge of the end-effector pose relative to a desired trajectory, causes a loss of precision on the task space.

The *task space control* problem follows a global approach that requires a greater algorithmic complexity when taking into account that inverse kinematics is now embedded into the feedback control loop. Its advantage is the possibility of acting directly on task space variables. However, this is somewhat only a potential advantage, since measurement of task space variables is often performed not directly, but through the evaluation of direct kinematics functions starting from measured joint space variables. Evaluating the facts listed above, task space control technique is selected to be implemented in this thesis study.

5.4. Dynamic Control Objective

The dynamic control objective is to design the control torque input signal τ (t) to be fed into the actuators of the robot to make the end-effector follow a desired end-effector position trajectory as closely as possible. The control signal should also include enough information to execute subtasks defined by at least one motion optimization measure. In this thesis study one motion optimization measure is used in the simulation tests. From now on, the task-space tracking will be referred as ***main-task objective*** and enabling the use of manipulators redundancy in optimization as secondary or ***subtask objective***.

5.4.1. Main-task Control Objective

Dynamic control of redundant manipulators in task space requires the computation of joint accelerations. Hence, redundancy resolution should be performed at the acceleration level. By using the second-order differential kinematics given in Equation 5.5 and applying PD control a version of computed-torque control is developed to determine the control output τ (t) as shown in Equation 5.9.

$$\tau = M_c\{J^+(\ddot{x}_d + K_v\dot{e} + K_p e - \dot{J}\dot{q}) + \ddot{\theta}_N\} + N_c \tag{5.9}$$

In Equation 5.9, x_d is the desired position defined in task space, $e = x_d - x$ is the tracking error, K_v and K_p are constant feedback gain matrices, M_c is the calculated inertia matrix, N_c is the calculated nonlinear terms that appear in the dynamics equation of the robot and $\ddot{\theta}_N$ is designed joint acceleration vector in the null space of J. If the manipulator does not go through a singularity, then the control law in Equation 5.9 guarantees that the tracking error converges to zero exponentially.

Proof: The closed loop system is given by

$$M\ddot{q} + N = M_c\{J^+(\ddot{x}_d + K_v\dot{e} + K_p e - \dot{J}\dot{q}) + \ddot{\theta}_N\} + N_c \ . \tag{5.10}$$

In this equation, the manipulator dynamics is assumed to be known with high precision so that $\widehat{M}^{-1}\widehat{M}_c \triangleq \hat{I}$ and $\overline{N} - \overline{N}_c \triangleq \overline{0}$. Making use of this assumption, Equation 5.10 is simplified to

$$\ddot{q} = J^{+}\left(\ddot{x}_d + K_v\dot{e} + K_p e - \dot{J}\dot{q}\right) + \ddot{\theta}_N \quad . \qquad (5.11)$$

Since M is uniformly positive definite, Equation 5.5 can be combined with Equation 5.11 to formulate Equation 5.12 considering that $JJ^{+} = I$ when J has full rank.

$$\ddot{e} + K_v\dot{e} + K_p e = 0, \qquad (5.12)$$

The proper choice of K_v and K_p (e.g. $K_v = k_v I$ and $K_p = k_p I$ with $s^2 + k_v s + k_p$ a *Hunwitz polynomial*) in Equation 5.12 implies that e goes to zero exponentially. Figure 5.3 shows main-task controller modeled in Simulink environment with respect to Equation 5.9.

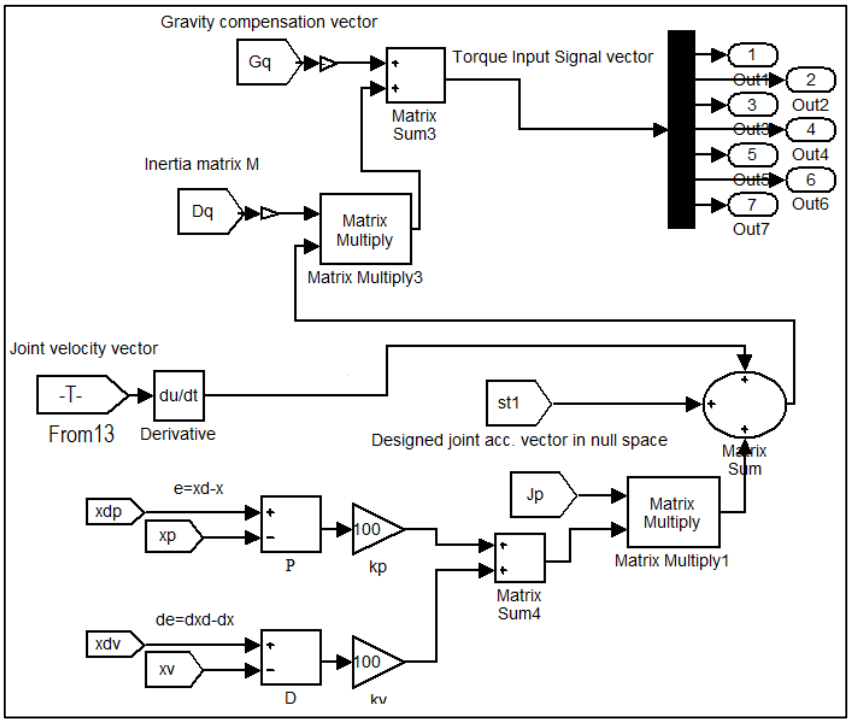


Figure 5.3. Main-task controller modeled in Simulink

5.4.2. Subtask Control Objective

We consider that a vector function $g(.) \in \Re^n$ is calculated as a gradient of the objective function *f*(*q*) for a specific subtask (which may be a function of time, the current state, etc.), and the null space joint velocities will be needed to track the projection of *g* onto the null space of *J*. Since (I_n - J^+J) projects vectors onto the null space of *J*, this can be formulated in an error signal calculation,

$$\dot{e}_N = (I - J^+J)g - \dot{\theta}_N \tag{5.13}$$

which converges to zero.

Assuming the manipulator does not go through a singularity condition, it is needed to design $\ddot{\theta}_N$ to get the desired result for subtask objective. Let missing part of the control that was given in Equation 5.9, $\ddot{\theta}_N$, be determined as;

$$\ddot{\theta}_N = (I - J^+J)\dot{g} - \left(J^+\dot{J}J^+ + \dot{J}^+\right)Jg + K_N\dot{e}_N \ . \tag{5.14}$$

In Equation 5.14, K_N is a positive definite feedback matrix. Then, the joint velocities in the null space converge to $(I - J^+J)g$, *i.e.*, $\dot{e}_N$, and the tracking error *e* (as defined in Section 5.1) converges to zero as given in the proof.

Proof: First note that $\ddot{\theta}_N$ given by Equation 5.11 belongs to the *null space* of *J* (when *J* has full rank) since

$$J(I - J^+J) = J - J = 0 \tag{5.15}$$

$$J\left(J^+\dot{J}J^+ + \dot{J}^+\right) = \dot{J}J^+ + J\dot{J}^+ = \frac{d}{dt}(JJ^+) = \frac{d}{dt}I = 0 \tag{5.16}$$

$\dot{e}_N$: is in the *null space* of *J* as well, and therefore,

$$JK_N\dot{e}_N = 0 \tag{5.17}$$

Equations 5.15 - 5.17 prove that the designed equation, Equation 5.14, in null space will not affect the main-task objective.

Now, let the second derivative of the error, calculated by taking the derivative of Equation 5.13 with respect to time, be considered as;

$$\ddot{e}_N = (I - J^+J)\dot{g} - \left(\dot{J}^+J + J^+\dot{J}\right)g - \ddot{\theta}_N \ . \tag{5.18}$$

Adding and subtracting $J^+\dot{J}J^+Jg$, to Equation 5.18, and substituting $\ddot{\theta}_N$ from Equation 5.14, Equation 5.19 is derived.

$$\ddot{e}_N = -J^+\dot{J}g + J^+\dot{J}J^+Jg - K_N\dot{e}_N \ . \tag{5.19}$$

By defining the following *non-negative* scalar Lyapunov function, the designed Equation 5.14 can be proven to be stable as follow:

$$v = \frac{1}{2}\dot{e}_N^2 \tag{5.20}$$

Then

$$\begin{aligned} \dot{v} &= \dot{e}_N^T\ddot{e}_N, \\ &= \dot{e}_N^T\left(-J^+\dot{J}g + J^+\dot{J}J^+Jg - K_N\dot{e}_N\right) \\ &= -\dot{e}_N^TK_N\dot{e}_N \ . \end{aligned} \tag{5.21}$$

In Equation 5.21, the following equality in Equation 5.22 is used that are formed by using Equation 5.13 and $\dot{\theta}_N$ from Equation 5.4;

$$\dot{e}_N^T = (g - \dot{q})^T(I - J^+J) \ . \tag{5.22}$$

The equalities represented below are the mathematical proofs that were used to calculate for Equation 5.21.

$$(I - J^+J)^T = (I - J^+J), \tag{5.23}$$

$$(I - J^+J)(I - J^+J) = (I - J^+J), \tag{5.24}$$

$$(I - J^{+}J)J^{+} = 0 \tag{5.25}$$

Since v is positive definite and $\dot{v}$ is negative definite, $\|\dot{e}_N\|$, goes to zero, exponentially fast.

Figure 5.4 shows subtask control scheme modeled in Simulink environment.

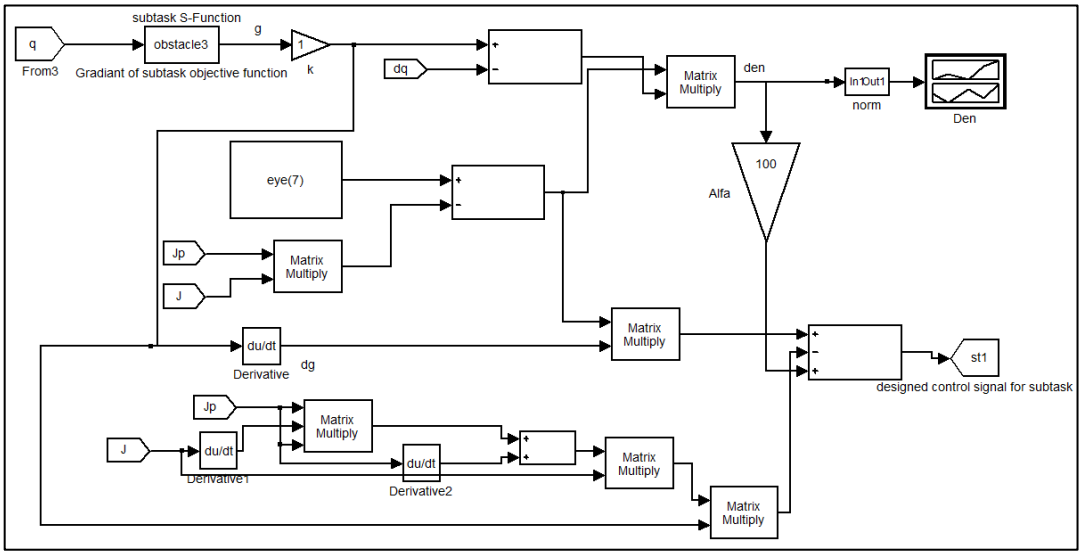


Figure 5.4. Subtask controller modeled in Simulink

5.5. Subtasks

The projection of the gradient function g onto the null space of J can be considered as the desired null space joint velocities that are needed to accomplish a given subtask. To control self-motion of the joint velocities, the gradient g (or its negative) of the objective function *f(q)* can be used as,

$$g = \nabla f = \frac{\partial f}{\partial q} = \left[\frac{\partial f}{\partial q_1} \cdots \frac{\partial f}{\partial q_i} \cdots \frac{\partial f}{\partial q_n}\right]. \tag{5.26}$$

Several researchers have worked on selecting the null space joint velocities for the purpose of avoiding singularity, joint limit avoidance, obstacle avoidance, minimizing potential energy, impact force configuration and achieving other subtasks. In the next subsections, the subtasks that are used in this thesis are presented.

5.5.1. Joints Motion Minimization

By using the stated principles above, the first subtask objective is based on minimizing joint motions for a 7-DOF redundant manipulator where the norm (or length) of joint velocities vector, $\|\dot{q}\|$, will minimized, in addition to the Main-task objective, which is tracking control.

To achieve this subtask, *f(q)* is chosen to be zero then, and as long as the manipulator is not in a singularity configuration, the null space velocity, $(I - J^{+}J)g$ will go to zero. However, this choice of control makes no provision for avoiding singularities (Baillieul 1984). If the manipulator approaches a singularity configuration, the control law defined by Equation 5.9 and 5.14 is no longer defined since J^{+} is discontinuous.

5.5.2. Singularity Avoidance

The objective function is selected according to *the manipulability measure* presented in (Yoshikawa 1984), which is a scalar value w_m defined as

$$w_m = \sqrt{det(JJ^T)} = \sigma_1\sigma_2 \ \dots \sigma_m \tag{5.27}$$

where σ_i is the i^{th} singular value of Jacobian matrix *J*. Depending on SVD explained in Sub-section 3.3.4, the manipulability ellipsoid is defined to be

$$u \in \Re^m \ : u^T(JJ^T)^{-1}u \ \leq 1 \qquad . \tag{5.28}$$

This definition forms an ellipsoid in *m* dimension (note that *J* has full rank equals to *m*), with principal axes in the directions of the columns of *u*, and their magnitudes are the values of the corresponding singular values in w_m. The volume of the ellipsoid is equal to $\frac{4}{3}\pi\sqrt{det(JJ^T)}$, so it is proportional to w_m. Since *J* maps joint velocities to end-effector velocities, the interpretation of the ellipsoid Figure 5.5 is as follows: large (small) magnitudes for the ellipsoid axes are corresponding to directions in end-effector space in which large (small) end-effector velocities can be generated.

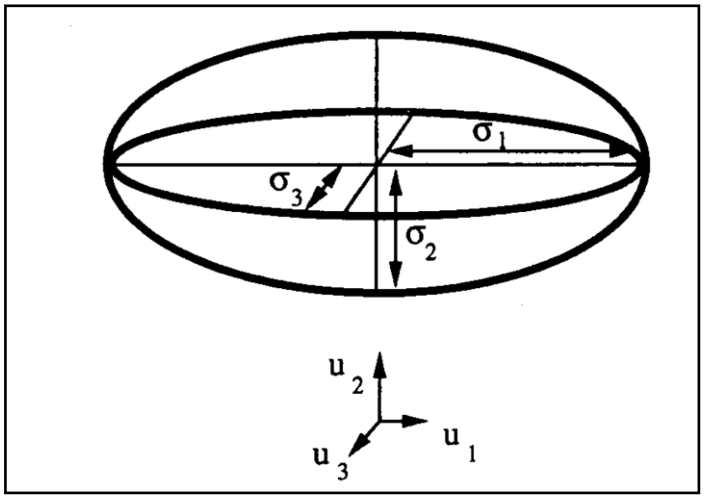


Figure 5.5. 3D Manipulability ellipsoid
(Source: Walker 1990)

The objective function is therefore chosen as,

$$f(q) = k \cdot (w_m)^2 = kdet(JJ^T) \tag{5.29}$$

where k is the self-motion control parameter gain, *det*(.) denotes the determinant of matrix, $J(q)$ is the manipulator Jacobian matrix. This objective function is based on purely robot kinematics. When the manipulator approaches its singularities, $f(q)$ decreases to zero. In order to maximize the manipulability of the manipulator, choosing the gradient as $g = \nabla f$ would keep the manipulator away from singularities.

5.5.3. Posture Optimization for the Static Impact Force Magnitude Measure Subtask Objectives

For robotic manipulators that are used in interaction tasks, the control engineer often requires the ability to specify the impact force that the end-effector can exert to the environment. For hammering or chiseling applications, the impact force may be required to be maximized, while in a medical application, a reduced collision force may be necessary. For these reasons, an appealing subtask is to position the arm in a posture which requires minimum or maximum torque for a desired force in a certain direction (Patel *et al.* 2005).

The manipulating force measure is a scalar, w_f, based on the static force/torque vector, F, relationship given by (Walker 1990);

$$\tau = J^T F, \tag{5.30}$$

and

$$w_f = 1/w_m = 1/(\sigma_1\, \sigma_2\, \ldots\, \sigma_m) \quad . \tag{5.31}$$

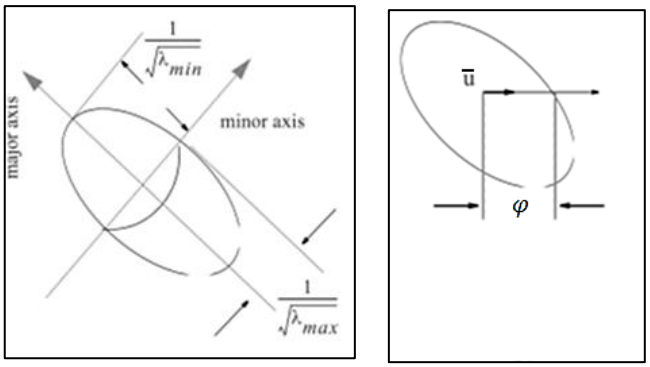
A manipulating force ellipsoid is defined similar to the manipulability ellipsoid in Section 5.2. In this case, large (small) principal axis directions are associated with directions in which large (small) static forces can be generated. The ellipsoid is perpendicular to the manipulability ellipsoid in the sense that their magnitudes in each principal axis direction are inversely proportional. Force ellipsoid can be defined by; $F^T(JJ^T)F$ where, F is the environment reaction force, which could also be selected as the forces exerted by the manipulator to the environment. The optimal direction for exerting the maximum force is along the major axis of the force ellipsoid which coincides with the eigenvector of the matrix JJ^T *c*orresponding to its largest eigenvalue λ_{max} as shown in Figure 5.6(a). The force transfer ratio along a certain direction is equal to the distance from the center to the surface of the force ellipsoid along this vector as it can be observed in Figure 5.6(b). In Figure 5.6(b), $\bar{u}$ is the unit vector along the desired direction and φ is the force transmission ratio along $\bar{u}$. Since $\varphi\bar{u}$ is a point on the surface of the ellipsoid, it should satisfy the following equation:

$$(\varphi\bar{u})^T(JJ^T)(\varphi\bar{u}) = 1 \tag{5.32}$$

Re-arranging Equation 5.32, an impact force magnitude measure can be written as $\varphi = [\bar{u}^T(JJ^T)(\bar{u})]^{-1/2}$. Hence, Chiu (Chiu 1988) proposed to maximize the following kinematic function presented in Equation 5.33 (task compatibility index) for maximum force configuration.

$$f(q) = \varphi^2 = \frac{1}{\bar{u}^T(JJ^T)\bar{u}} \tag{5.33}$$

In this thesis, utilizing Equation 5.33, impact force subtask objectives is defined to bound the impact force with the environment at the highest or lowest level (Tatlıcıoğlu *et al.* 2009).



(a) (b)

Figure 5.6. Force Ellipsoid (a) Ellipsoid axes, (b) Force Transfer Ratio in direction $\bar{u}$

5.5.3.1. Maximizing the Static Impact Force Magnitude

The objective for this subtask is to keep the robot manipulator in a posture that *maximizes* the ability to withstand external static impact force with the environment at a given end-effector position and in specific direction at the point of contact. *f*(*q*) is defined as the denominator of Equation 5.33, and can be written as follows:

$$f(q) = k\,\bar{u}^T(JJ^T)\bar{u}, \tag{5.34}$$

small values of *f*(*q*) indicate postures with high impact forces at the end-effector; therefore, after setting, $g = -\nabla f$, and specifying force unit vector at the point of contact for the end-effector, by minimizing *f*(*q*), subtask objective can be reached.

5.5.3.2. Minimizing the Static Impact Force Magnitude

The objective for this subtask is to keep the robot manipulator in a posture that *minimizes* the ability to withstand external static impact force with the environment, for a given end-effector position and in specific direction at the point of contact. The goal of this subtask is to force the manipulator into postures that result in *larger* values of *f*(*q*) which is defined in Equation 5.34. After setting, $g = \nabla f$, and specifying force unit vector at the point of contact for the end-effector, the subtask objective will be reached.

5.5.4. Static and Moving Obstacle (point to point) Collision Avoidance

The purpose of this subtask is to select an objective function that keeps the closest point on the links away from the selected obstacles. Varies studies have been reported on the obstacle avoidance issues for redundant manipulators. One of the common approaches is to optimize an objective function for obstacle avoidance using self-motion while completing the main-task. Typical forms for such an objective function employ the minimum distance between manipulator and obstacles:

$$d = min\{\|p_m - p_o\|\} \tag{5.35}$$

Where $p_m \in \Re^3$and $p_o \in \Re^3$ denote a point on the manipulator and the obstacle, respectively, and ||.|| is the Euclidean norm. Once the minimum distance is determined, there are several different ways to establish the objective function by using *d* for the purpose of collision avoidance. The simplest objective function is directly obtained by setting *f*(*q*) = *d* to be maximized with positive *k* (Guo *et al.* 1990), (Wang *et al.* 1992). Assuming multiple obstacles in the workspace, the objective function *f*(*q*) can be modified as a sum of minimum distances (Kemeny 1999):

$$f(q) = \sum_{i=1}^{l} \sum_{j=1}^{o} d_{ij}(q) \tag{5.36}$$

Where d_{ij} is the minimum distance between i^{th} link and j^{th} obstacle, *l* and *o* are the total number of links and obstacle respectively. While in (Chen *et al.* 2002) squared minimum distance is used as an objective function as follows.

$$f(q) = \sum_{i=1}^{l} \sum_{j=1}^{o} {d_{ij}}^2(q) \tag{5.37}$$

The other type of common objective function for collision avoidance has been formulated using reciprocal of the minimum distance:

$$f(q) = \sum_{i=1}^{l} \sum_{j=1}^{o} {d_{ij}}^{-1}(q) \tag{5.38}$$

this should be minimized with negative *k*.

For all above objective functions, there are several shortcomings of using minimum distance as in the review of (Kwang-Kyu *et al.* 2007). Let us consider the gradient of several distances between point obstacles and a robot link, see Figure 5.7. The points P_1, P_2, and P_3 represent point obstacles which lie on the same line perpendicular to the link at C, d_1, d_2, and d_3 is the distance from the link to P_1, P_2, and P_3, respectively. For infinitesimal rotation δq_1 of the joint variable q_1, δd_1 and δd_2 can be rewritten as

$$\frac{\partial d_1}{\partial q_1} = \frac{\partial d_2}{\partial q_1} = \|OC\| \tag{5.39}$$

From Equation 5.39, it should be noted that all the obstacles along the same line perpendicular to the i^{th} link have the same norm of distance gradient with respect to q_i regardless of the distance. Furthermore, the norm of the gradient is equal to the projection of the obstacles to the link, in this case $\|OC\|$. Therefore, the following relationship holds in this case:

$$\frac{\partial d_1}{\partial q_1} = \frac{\partial d_2}{\partial q_1} = -\frac{\partial d_3}{\partial q_1} \tag{5.40}$$

When taking multiple obstacles P_1 and P_2 illustrated in Figure 5.8.a. into account and assuming that both d_1 and d_2 are the smallest distances from obstacles, the objective function in Equation 5.36 with gradient projection method (GPM) cannot provide a decision for obstacle avoidance priority.

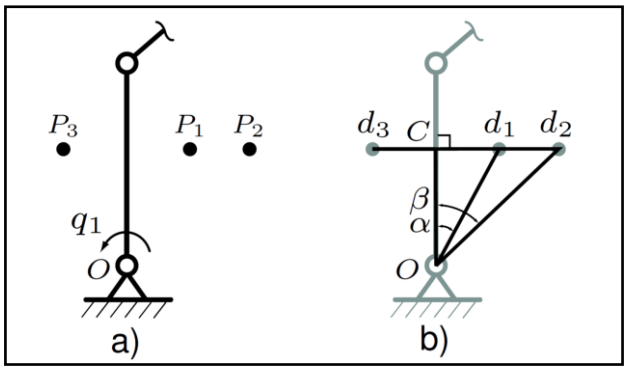


Figure 5.7. Manipulator with point obstacles: P1, P2, and P3 lie on the same line (Source: Kwang-Kyu *et al.* 2007)

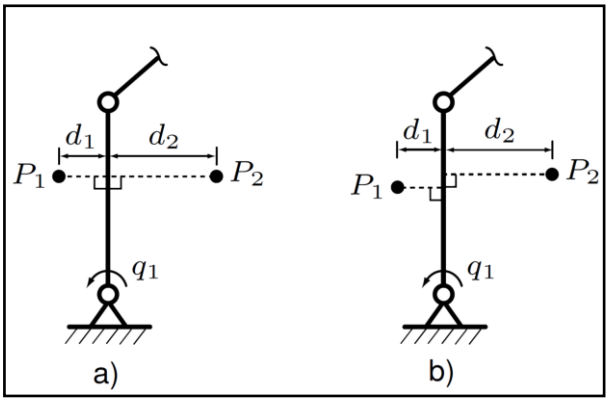


Figure 5.8. Manipulator with point obstacles: P1, P2, and P3
(Source: Kwang-Kyu *et al.* 2007)

Because both gradients are of the same norm and opposite direction, the obstacle P_1 is more critical for collision. Even wrong decision for collision avoidance can also be made, for instance, the links shown in Figure. 5.8.b. In this configuration, clearly P_1 is potentially more dangerous for the collision to the manipulator. However, the objective function Equation 5.36 with GPM will drive the first link counter clock wise to increase d_2 since the gradient of d_2 is greater than that of d_1. This problem is getting even worse when the objective function Equation 5.37 is applied to Figure. 5.8.a, since its gradient is now weighted by each distance such that the obstacle P_2 has dominant gradient ($d_2 \nabla d_2$) over P_2 ($d_1 \nabla d_1$).

The Jacobian matrix transpose enhanced inverse kinematics proposed in (Wolovich *et al.* 1984) only differs from the resolved motion rate control in that the transpose of the Jacobian matrix is used instead of its inverse. This method extended to redundant manipulators (Sciavicco 1988) with extended task space scheme and its stability was discussed in (Das *et al.* 1988).

Although the convergence properties of the Jacobian matrix transpose method is poor, its low computational burden is still very attractive for such applications in which exact trajectory tracking for the critical point with respect to obstacles is not in question. Hence, a computationally simple obstacle avoidance scheme can be achieved by means of the Jacobian matrix transpose method, in which the escape velocity gain v_{cij} can be defined as a function of the minimum distance d_{ij} along the direction away from the critical point X_{Cij}.

$$v_{cij} = v_m e^{-d_{ij}} \bar{u}_{ij} \tag{5.41}$$

Where v_m is the maximum escape velocity gain, and $\bar{u}_{ij}$ is the unit vector from the critical point X_{Cij} on the i^{th} link to the j^{th} obstacle. A cleaver way to select the gain for each obstacle avoidance objective will give a decision to the robot to select the best self-motion to reach the objective. A set of calculations will be needed to calculate for $\bar{u}_{ij}$. This set of calculation will be started by defining the unit vector direction of the link *i* as follows:

$$e_i = \frac{(X_{i+1} - X_i)}{l_i} \tag{5.42}$$

where, e_i represents the unit vector in frame *i* and along link *i*. X_i and X_{i+1} are the positions of both ends for link *i* of length l_i . Then to find the position of point $X_{C_{ij}}$, Equation 5.43and 5.44 are used.

$$\beta_{ij} = e_i^T \left(X_{oj} - X_i\right) \tag{5.43}$$

$$X_{C_{ij}} = X_i + \beta_{ij} e_i \tag{5.44}$$

Then the minimum distance between i^{th} link and j^{th} obstacle is calculated as shown in Equation 5.45. Finally, the unit vector direction is calculated by Equation 5.46.

$$d_{ij} = \left\| X_{C_{ij}} - X_{oj} \right\| \tag{5.45}$$

$$u_{ij} = \frac{\left(X_{C_{ij}} - X_{oj}\right)}{d_{ij}} \tag{5.46}$$

Furthermore, in case of multiple obstacles, the escape velocity v_{ci} can be obtained as:

$$v_{ci} = \sum_{j=1}^{o} v_{cij} \quad . \tag{5.47}$$

When d_{ij} is sufficiently large, then no collision danger exists, which results in v_{cij} to be small. On the contrary, when d_{ij} is getting small (equal to zero), then additional

safety action should be taken while holding v_{cij} with its maximum value ($v_{cij} = v_m$). Once the obstacle escape velocity is determined, it can be transformed to the joint space using Jacobian matrix transpose method (Sciavicco *et al.* 1988, Das *et al.* 1988):

$$\dot{q}_c = \sum_{i=1}^{k} J_{ci}^{T}(q) v_{ci} \quad . \qquad (5.48)$$

In Equation 5.48, $\dot{q}_c$, is a vector of joint velocities. This vector will be used as a gradient g of objective function to avoid obstacles.

To determine the minimum distance d_{ij}, links are modeled by straight lines and the objects are assumed to be circles for planar case and spheres for spatial case. Each object is enclosed in a fictitious protection shield (represented by a circle or sphere) called the Surface of Influence (SOI). This method can provide increased safety for avoiding collusions.

The first step involves distance calculation to find the location of the point X_{Cij} (called the critical point) on each link that is nearest to the obstacle by the procedure indicated in Equation 5.42 to Equation 5.46. This algorithm is executed for each link and each obstacle.

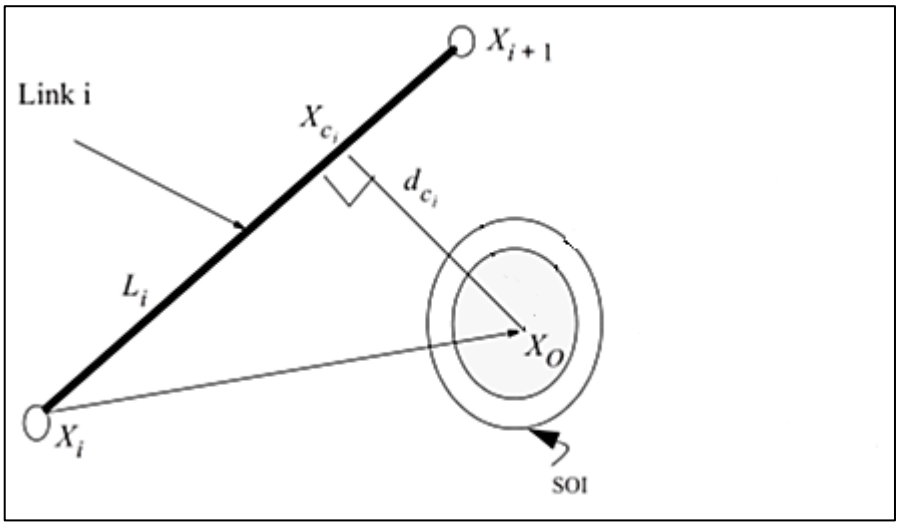


Figure 5.9. Critical Distance scheme from obstacle one

CHAPTER 6

SIMULATION AND RESULTS

In this chapter simulation test setup is revealed. The criteria for selecting the main-task objective, end-effector trajectory in the task space, are discussed. The Simulink model defined in Chapter 4 and the designed control presented in Chapter 5 are implemented in MATLAB® Simulink to conduct tests for evaluating the controller developed in this study. In this chapter, first, the simulation test scenario is explained and it is followed by the presentation of the results achieved for all the subtask simulation tests.

6.1. Simulation Test Setup

To illustrate the performance of the proposed general subtask controller presented in this work, a set of simulation results are presented in this section. In these simulations, the aim is to utilize the virtual model of 7-DOF LWA4-Arm produced by SCHUNK GmbH. The manipulator's CAD model is presented in Figure 6.1. The manipulator model is then transferred to the simulation environment as explained in Chapter 4.

Figure 6.1. The CAD Model of 7-DOF LWA4-Arm by SCHUNK GmbH

The simulations tests are carried out in MATLAB® Simulink simulation environment Version 7.11.0.584 (R2010b), 64-bit (win64) with a third-order ordinary differential equation solver (Bogacki-Shamine) and fixed-step sample time of 0.1 kHz.

The manipulator is initially at rest in the following joint positions q= [0 -25 0 -35 0 0 0]^T, in degrees.

In the main-task controller presented in Equation 5.9, the nonlinear terms that include centripetal and Coriolis, $C(q,\dot{q})\dot{q}$, frictional, $F(\dot{q})$ and disturbance, ξ_d, are neglected since the robot moves in slow motion. However, the effect of gravitational forces is calculated to form the *N* nonlinear effect cancellation term in Equation 5.9.

6.2. Task Space Trajectory

The selected main-task objective is to track positions in *x*-*y*-*z* space where the end-effector orientation is not specified. In this case, the redundant manipulator had more DOF than is required to perform a task in its task space. Hence, these extra DOFs allowed the robot manipulator to perform more dexterous manipulation and/or provided the robot manipulator system with increased flexibility for the execution of sophisticated tasks as illustrated in Figure 6.2 by making use of increase self-motion capability. Since the dimension *n* of the link position variables is seven and the number of the task-space variables *m* is three, the null space of Jacobian matrix has a minimum dimension of, *n* – *m*, four, which is the extra DOF. As shown in Figure 6.2.b. the self-motion can be represented as a spherical movement of the elbow around shoulder point center and spherical movement for the wrist around the end-effector point accompanied with relative movement of the joints in between.

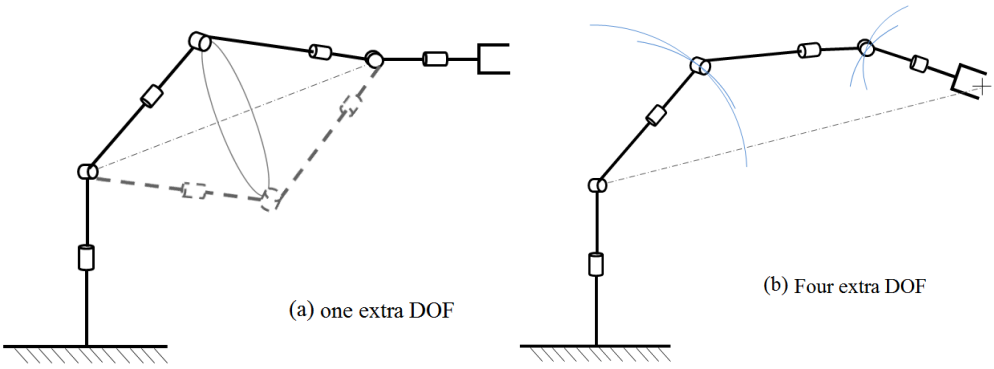
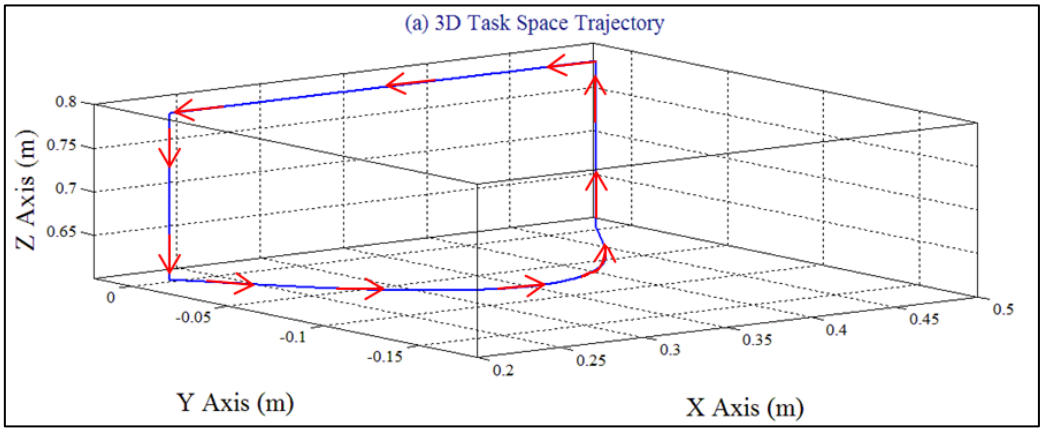
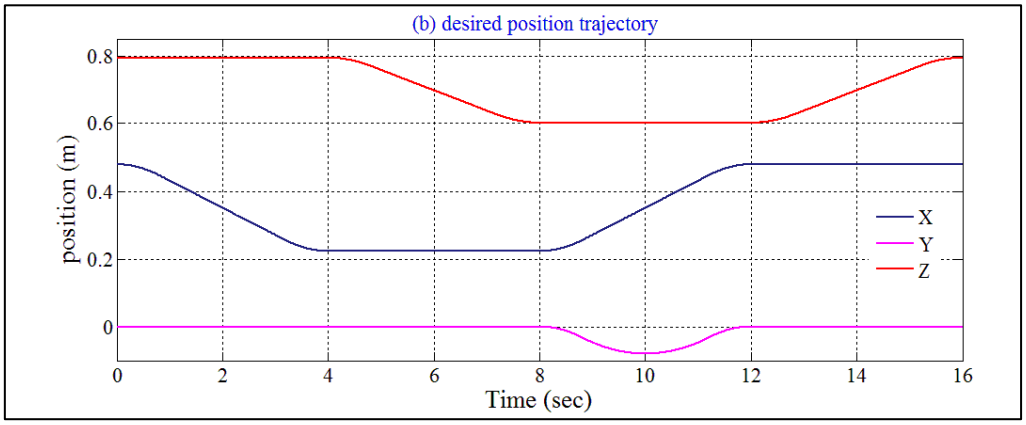


Figure 6.2. Self-Motion of Spatial 7-DOF. (a) With one extra DOF, (b) With four extra DOF

Figure 6.3 shows the desired task-space trajectories for all simulations. The trajectory is selected to be in 3D space following linear and curved motions in the directions as shown in Figure 6.3.a. Initial position of the end-effector can be depicted in Figure 6.3.b. At each corner of the path, the end-effector motion velocity goes to zero as it can be noticed in Figure 6.3.c.



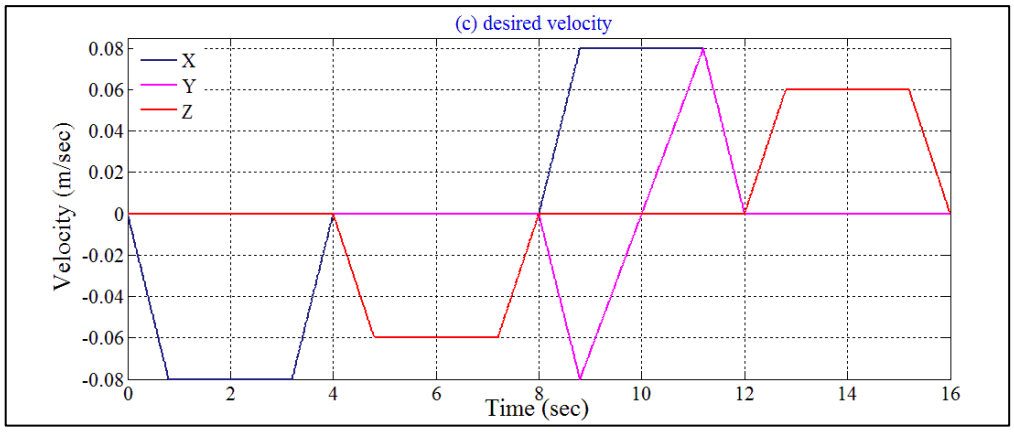
(a)



(b)

Figure 6.3. Desired task-space trajectories: (a) 3D task space trajectory, (b) Desired position trajectory, (c) Desired velocity

(cont. on next page)



(c)

Figure 6.3. (cont.)

6.3. Simulation Results

Simulations are conducted to test and illustrate the performance of the proposed controller. The main-task and subtask controller parameters are tuned to the following values after some iterative experimental tests to provide better performance:

$k_v = 100, k_p = 100,$

$K_N = diag\{100, 100, 100, 100, 100, 100, 100, \}.$

6.3.1. Free Self-Motion Simulation Result

The first simulation presented here is carried out to provide a comparison for the evaluation of the next two subtask simulations. In this simulation, the subtask input $\ddot{\theta}_N$ is set to zero, where no restriction on the self-motion of the robot is set and only the end-effector tracking objective is enforced. Figure 6.4 shows tracking error for the end-effector position. It can be observed that the end-effector position tracking error is bounded within 0.5 mm per axis for simulation test.

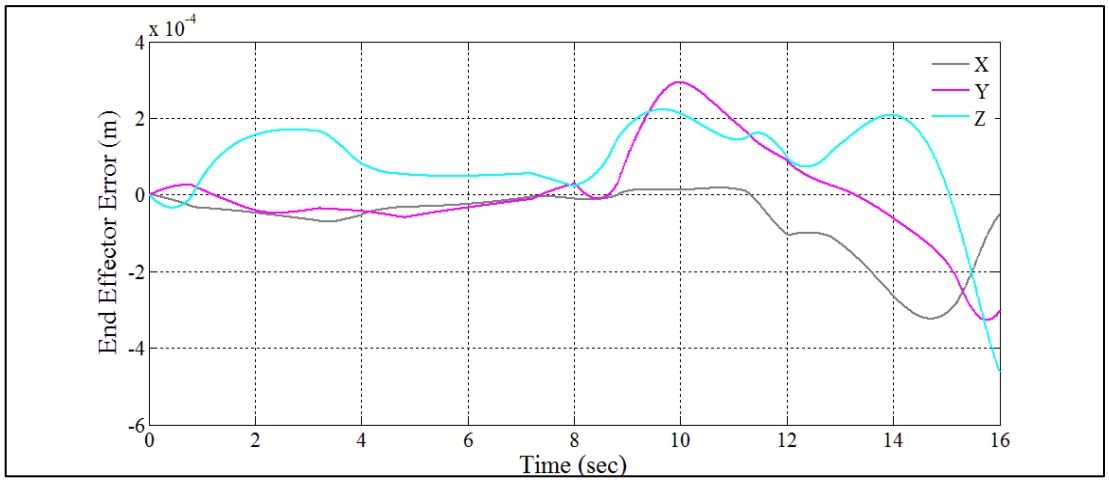


Figure 6.4. End-effector position error for Free Self-Motion

Figure 6.5 shows the joint velocities trajectories for the free self-motion test. It can be noticed that the joint velocities can reach up to 1.2 rad/sec and this high amount of velocities will be shown to be unnecessary motion for some joints in the next sub-section.

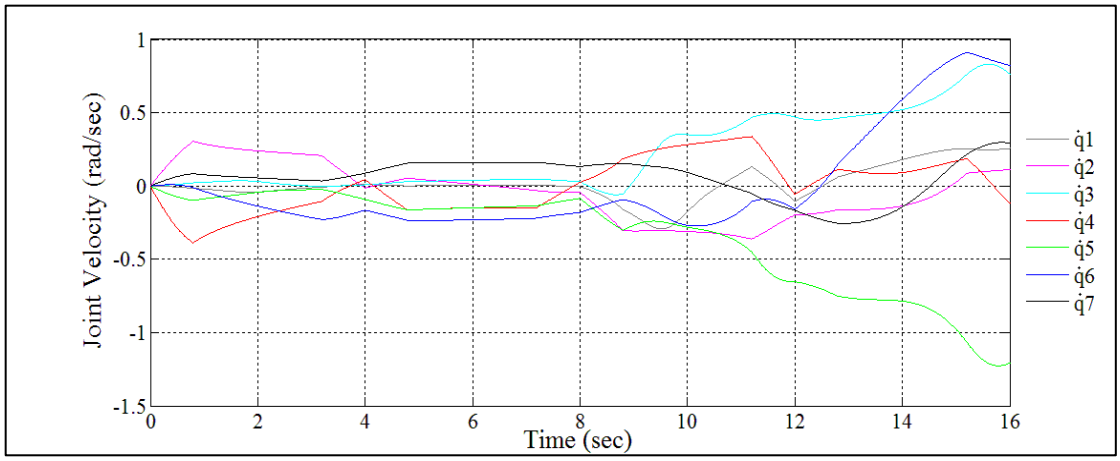


Figure 6.5. Joint velocities for Free Self-Motion

Figure 6.6 shows the controller's output torque signals as a result of the main-task and free self-motion, which appear to be up to 15 Nm resulting from the uncontrolled motion in the null space. In this figure and in the Torque vs. Time plots from this point on, torque applied to the i^{th} joint is indicated with T_i symbol in the legends.

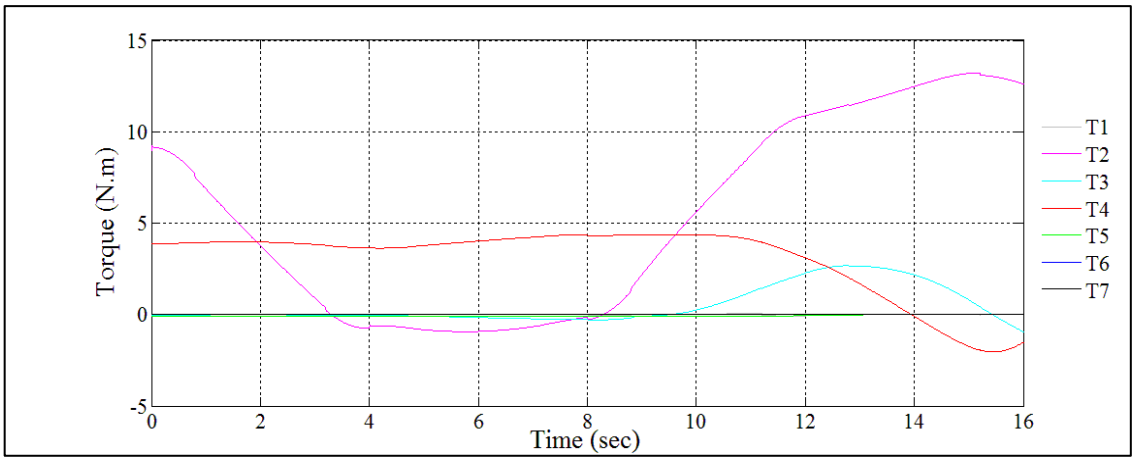


Figure 6.6. Controller output torque signals for Free Self-Motion

6.3.2. Joint Motion Minimization Simulation Result

In this subtask simulation $f(q)$, is set to zero to minimize total joint motion and to realize the subtask objective presented in Sub-section 5.5.1. Subtask's objective function control gain magnitude k is tuned to 50 for better result. Figure 6.7 shows tracking error for the end-effector position. It can be observed that error magnitude is less than the Free Self-Motion test result. Figure 6.8 shows the link velocity trajectories with Joint Motion Minimization Subtask. It shows that the main-task is executed with less joint motion than observed in the Free Self-Motion test results.

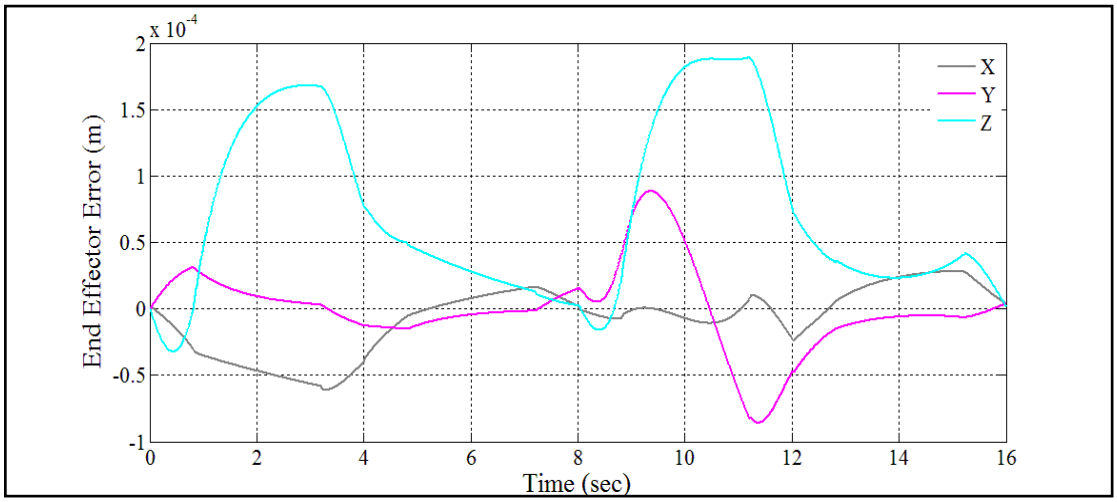


Figure 6.7. End-effector position error for Joint Motion Minimization subtask

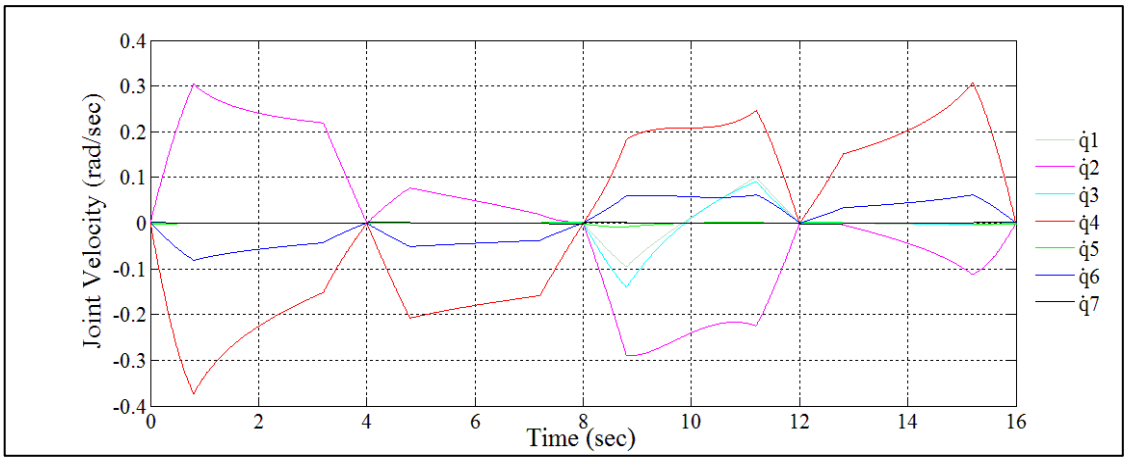


Figure 6.8. Joint velocities for Joint Motion Minimization subtask

The required torque signals for executing both main-task and subtask are presented in Figure 6.9. It can be observed that the applied torque magnitudes are less than the ones in the Free Self-Motion test. Figure 6.10 clearly indicates the performance of this subtask by showing joint velocities vector's norm (or length) and comparing it with the previous simulation. Figure 6.11 verifies the validity of subtask controller by showing that the subtask error signal is bounded.

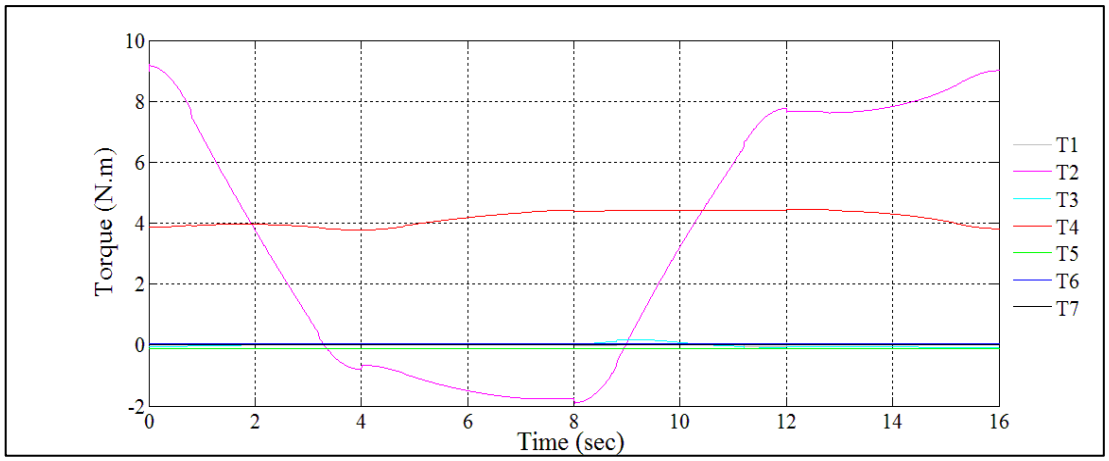


Figure 6.9. Controller output torque signals for Joint Motion Minimization subtask

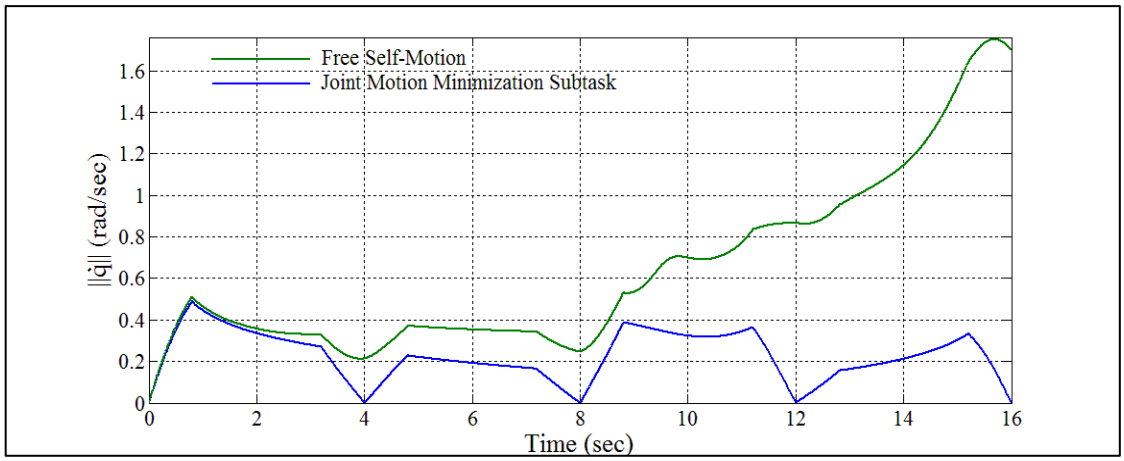


Figure 6.10. Joint velocities vector norm

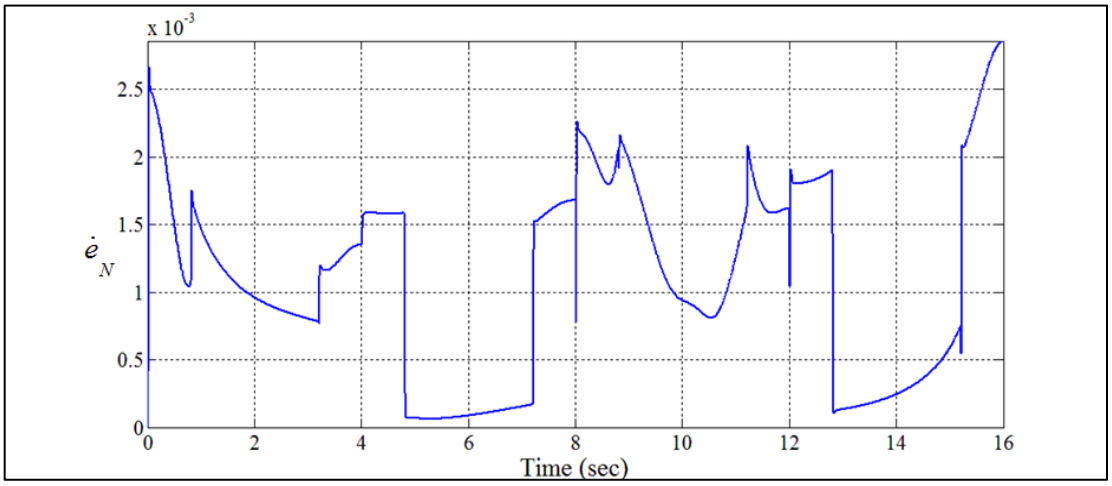


Figure 6.11. Subtask error norm magnitude for Joint Motion Minimization subtask

6.3.3. Singularity Avoidance Simulation Result

In this simulation test, $f(q)$ is selected to maximize the manipulability and to realize the subtask objective presented in Sub-section 5.5.2. Subtask's objective function control gain magnitude k is tuned to be 50 for better result. Figure 6.12 shows tracking error for the end-effector position. As the error is bounded within 0.2 mm, it can be stated that the main-task objective is reached. Figure 6.13 shows joint velocities for singularity avoidance subtask. It can be observed that the main-task is executed with less joint motion than the Free Self-Motion results as it was the case for the Joint Motion Minimization subtask simulations. The torque signals required to execute both

main-task and subtask together is presented in Figure 6.14, which are also smaller in magnitude than for the Free Self-Motion simulation.

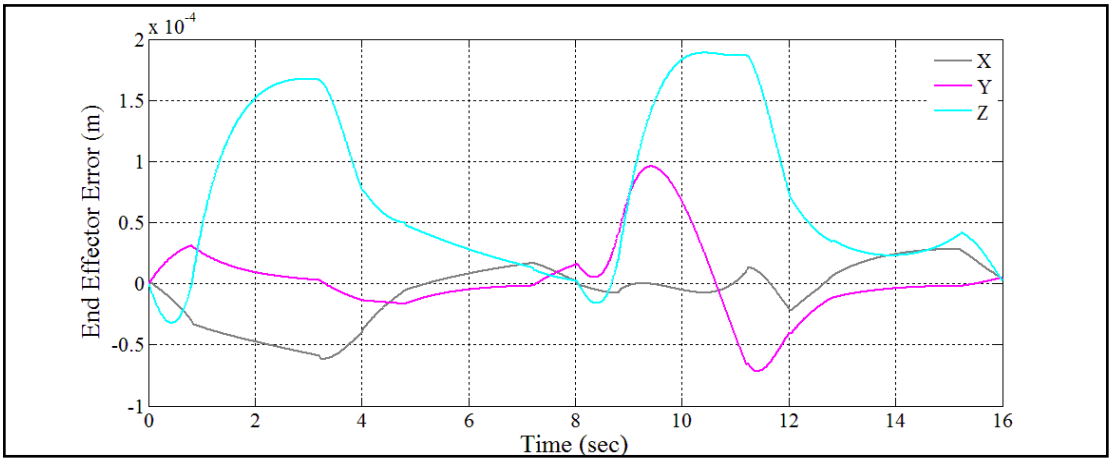


Figure 6.12. End-effector position error for Singularity Avoidance subtask

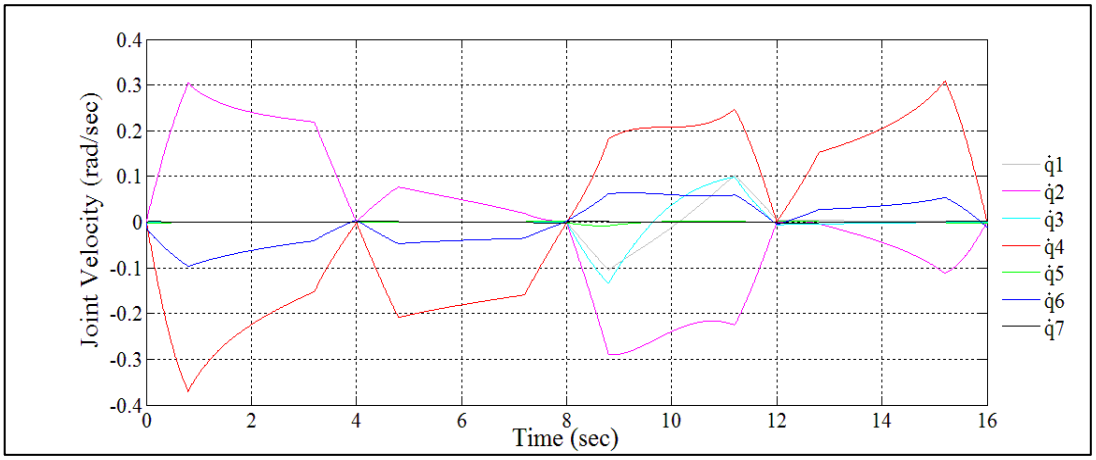


Figure 6.13. Joint velocities for Singularity Avoidance subtask

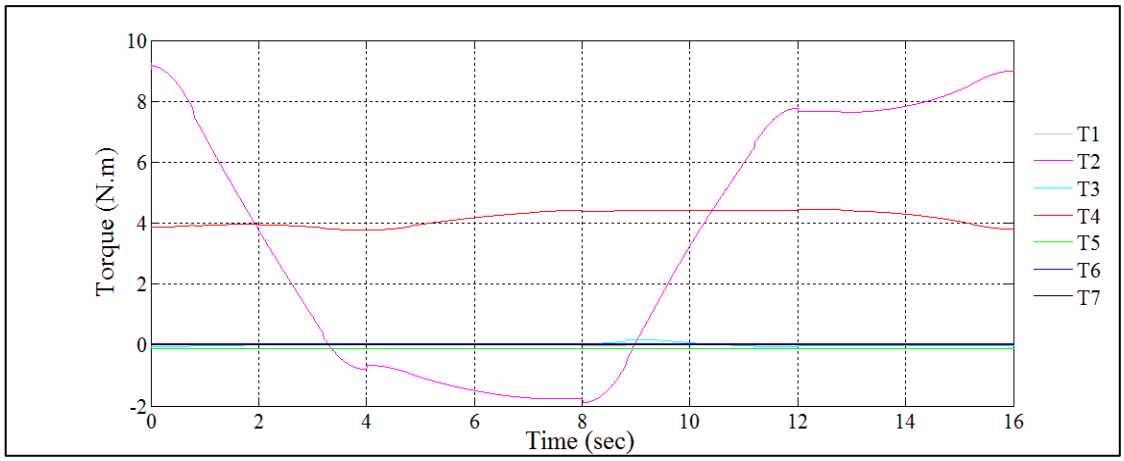


Figure 6.14. Controller output torque signals for Singularity Avoidance subtask

The effect of singularity avoidance subtask objective on the system can be observed in Figure 6.15. The difference in manipulability measures between the results of Free Self-Motion simulation and Singularity Avoidance subtask can be noticed in this figure. As the manipulability approaches zero after the 12th second for the Free Self-Motion simulation, which indicates that the manipulator approaches a singularity condition, the manipulability measure is maximized in the test results of the Singularity Avoidance subtask. Figure 6.16 verifies the validity of subtask controller by showing subtask error signal is bounded.

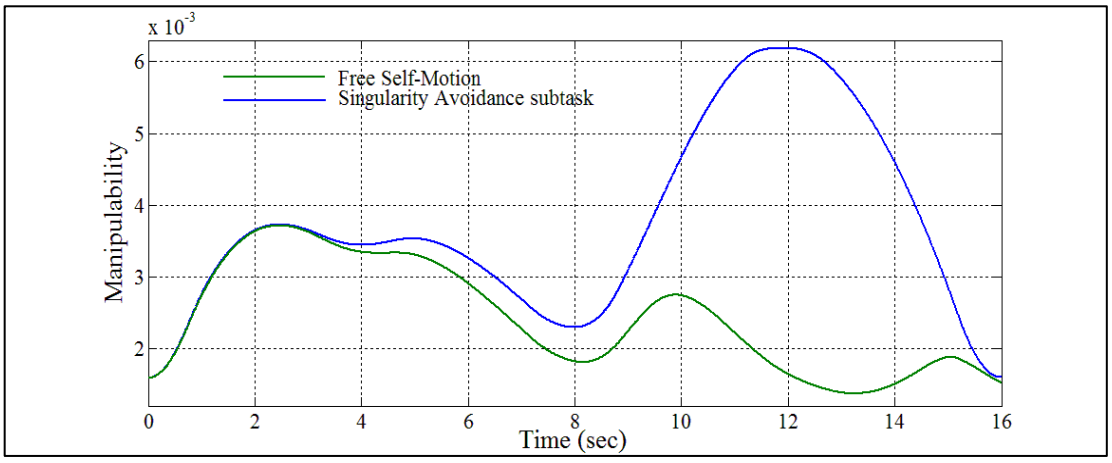


Figure 6.15. Manipulability measure for Free Self-Motion and Singularity Avoidance subtask

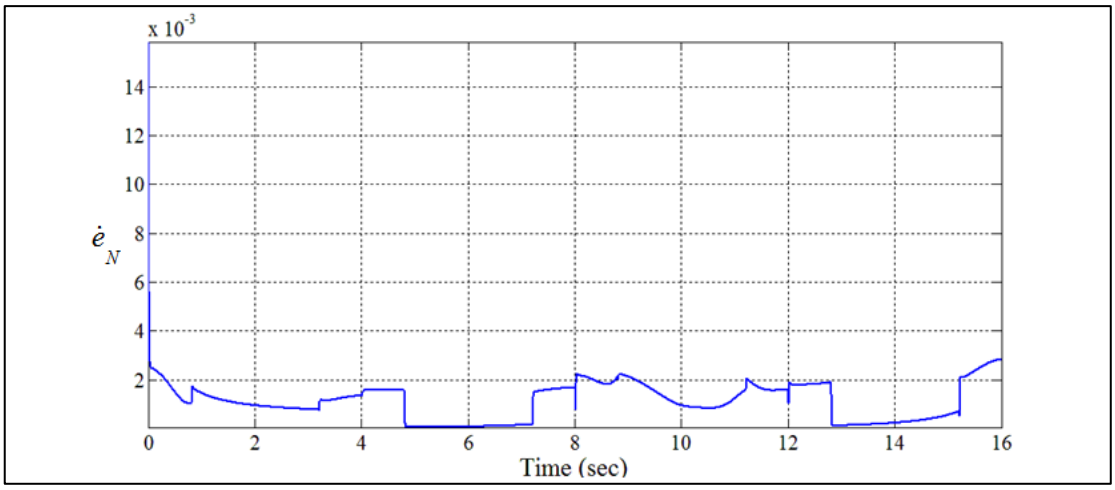


Figure 6.16. Subtask error signal norm for Singularity Avoidance subtask

6.3.4. Posture Optimization for the Static Impact Force Magnitude Measure Subtask Objectives Simulation Results

In this simulation test, the posture optimization for the static impact force subtask presented in Sub-section 5.5.3 is used. Two sets of simulations are performed for maximization and minimization of static impact force magnitudes.

6.3.4.1. Maximizing the Static Impact Force Magnitude

In this simulation, posture optimization presented in Sub-section 5.5.3.1 for the static impact force magnitude maximizing subtask objective is executed.

Subtask's objective function controller parameter is tuned to be $k = 30$ for providing better results. Force unit vector direction is selected to be in x direction, $\bar{u} = [1\ 0\ 0]^T$.

Figure 6.17 shows tracking error for the end-effector position. The error for the main-task is bounded within 1.5 mm and this indicates that the main-task objective is satisfied.

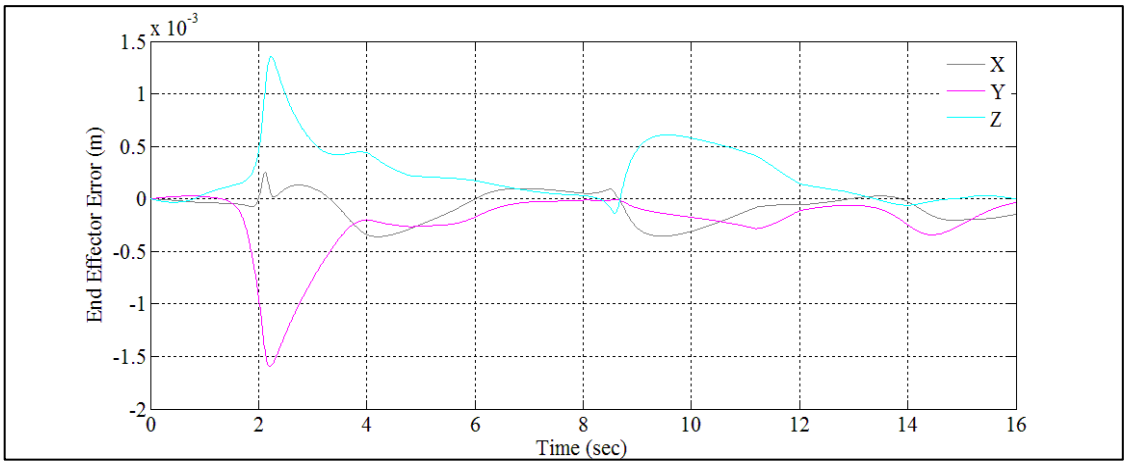


Figure 6.17. End-effector position error for Static Impact Force Maximization subtask

Acquired joint velocities from the joint sensors throughout this simulation are presented in Figure 6.18. At the 2nd second of the simulation test, the second joint axis is rotated to be parallel to *x*-axis as a result of the motion of the first and third joint and this can be observed by the large amount of joint velocities for the first and the third joint in Figure 6.18. By this configuration of the robot arm, the manipulability in *x* direction is minimized and subtask objective is reached.

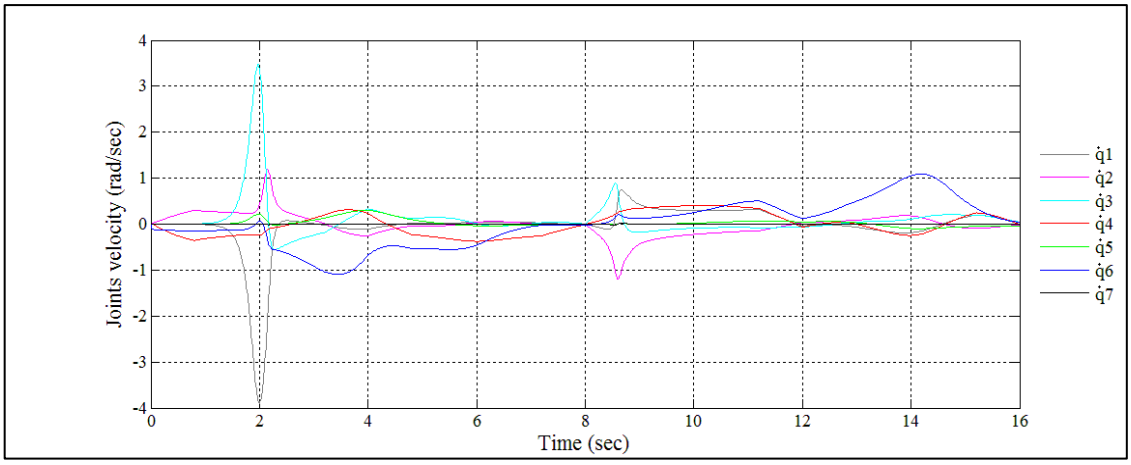


Figure 6.18. Joint velocities for Static Impact Force Maximization subtask

Figure 6.19 shows the controller output torque signals calculated by the subtask and main-task controller. It can be noticed that both joint velocities and control input torque signals are bounded. The maximum torque signal is within 13 Nm, which is smaller than the Free Self-Motion simulation result, however, it is larger than the Joint Motion Minimization subtask as expected.

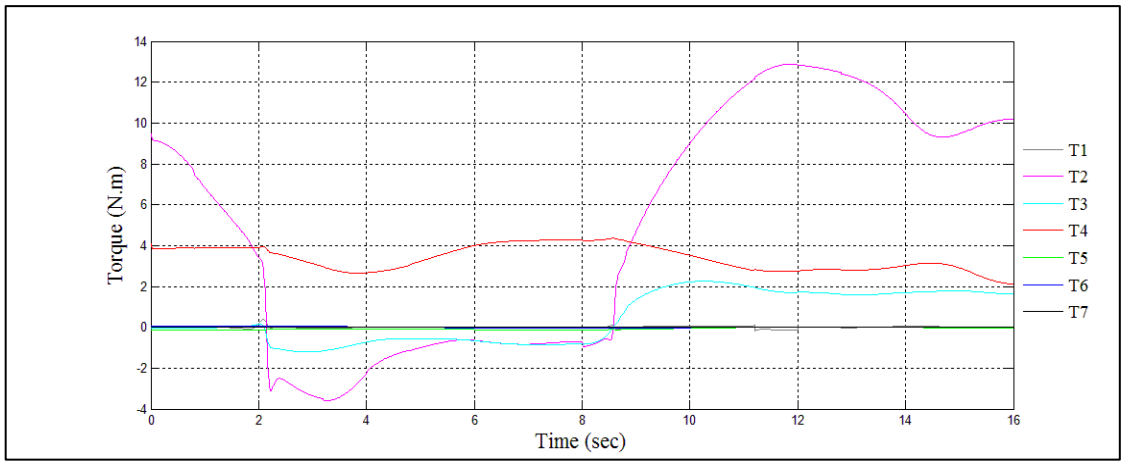


Figure 6.19. Controller output torque signals for Static Impact Force Maximization subtask

It can be observed from Figure 6.20 that the objective function magnitude was minimized throughout the simulation. The sudden decrease of the manipulability at the 2nd second indicates the change in the configuration that was previously mentioned for joint velocities, which is needed for the purpose of maximizing the static impact force.

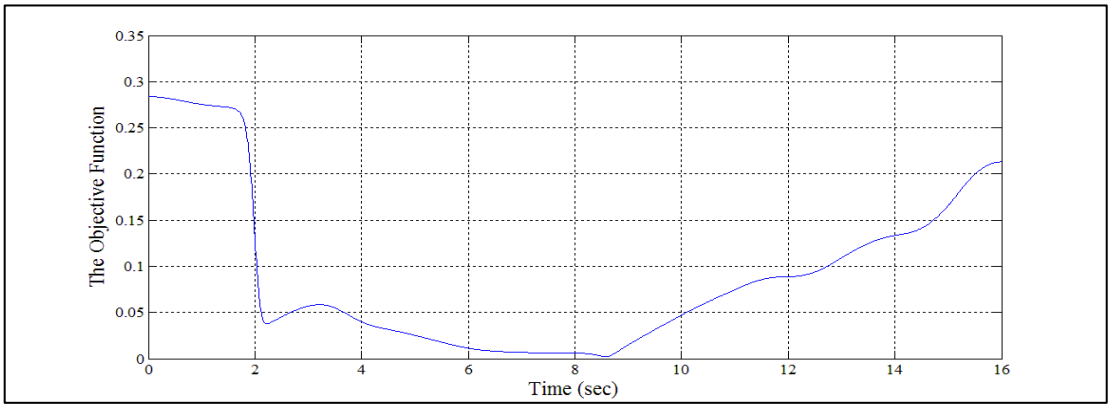


Figure 6.20. Objective function magnitude for Static Impact Force Maximization subtask

The configuration of the robot during the simulation is illustrated in Figure 6.21 where the link positions are shown at each second of simulation test. Figure 6.22 verifies the validity of subtask controller by showing subtask error signal is bounded. The relatively large error observed at 2nd second is due to the higher velocities achieved in the first and third joints, which are needed to change the configuration.

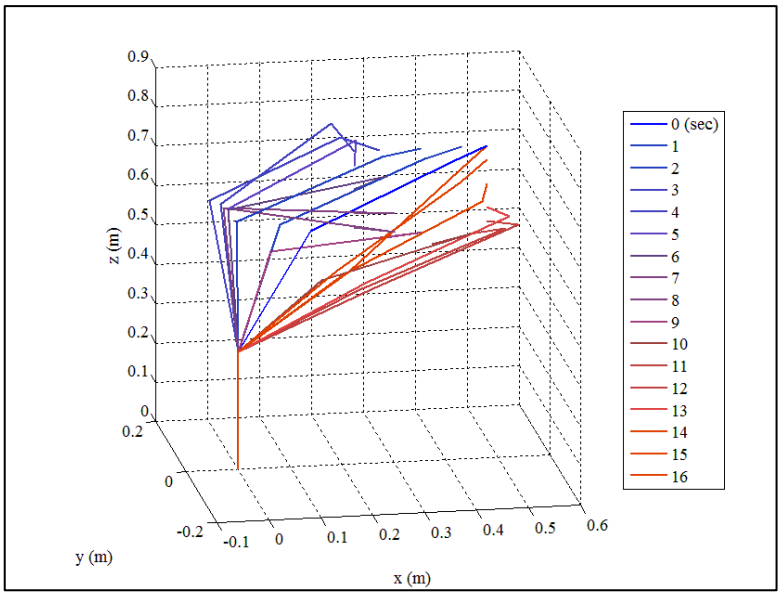


Figure 6.21. Robot arm motion during simulation for Static Impact Force Maximization subtask

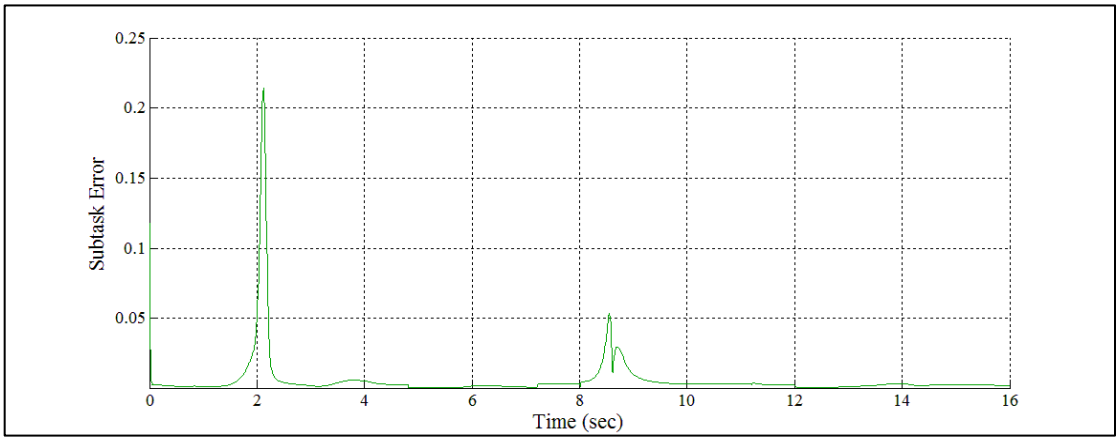


Figure 6.22. Subtask error signal norm for Static Impact Force Maximization subtask

6.3.4.2. Minimizing the Static Impact Force Magnitude

In this simulation, posture optimization presented in Sub-section 5.5.3.2 for the static impact force magnitude minimizing subtask objective is executed.

Subtask's objective function controller parameter is tuned to be $k = 100$ for achieving better results. The forces the manipulator can exert are selected to be minimized in the *x-axis* direction, $\bar{u} = [1\ 0\ 0]^T$.

Figure 6.23 shows tracking error for the end-effector position. The small increase in error shown here between the 2nd and 4th seconds is due to the change in configuration. This change in configuration is needed to achieve this subtask objective.

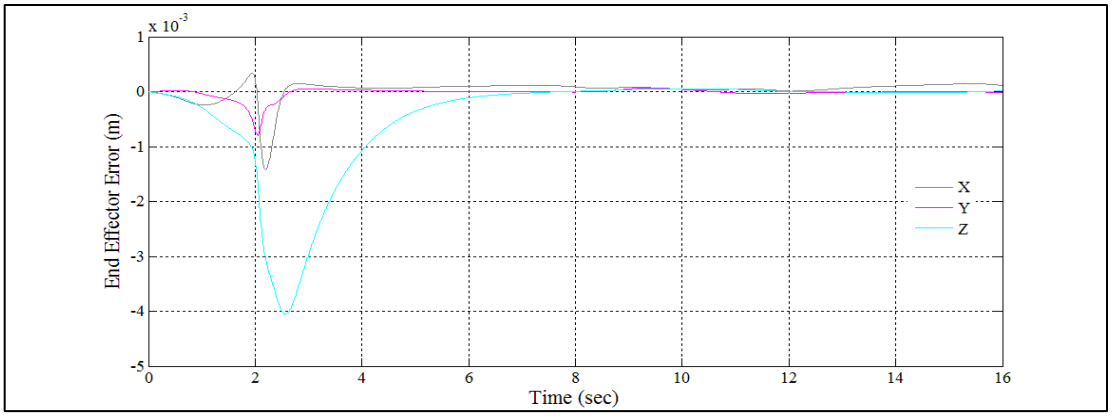


Figure 6.23. End-effector position error for Static Impact Force Minimization subtask

The joint velocities signals vector measured and recorded throughout this simulation is presented in Figure 6.24. The increase in the velocity of joints 2, 4 and 6 is due to the change in configuration. Furthermore, this configuration was near singularity because the wrist point is near its workspace limits, which is also called extended singularity.

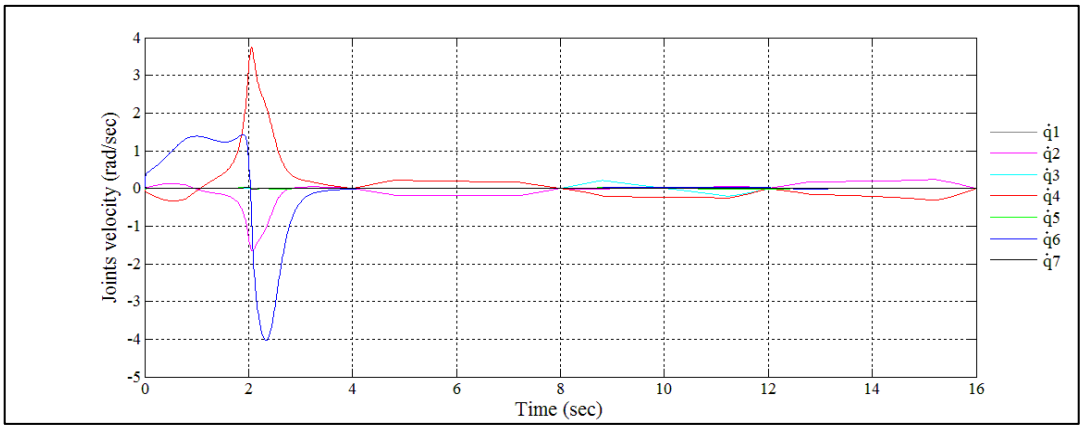


Figure 6.24. Joint velocities for Static Impact Force Minimization subtask

Figure 6.25 shows the controller output torque signals calculated as a result of the main-task and subtask controller. The torque signal of joint 2 is higher than the

others since it carries the robot arm weight and the mass center of the links are arranged to be in the furthest possible configuration from joint 2 to realize this subtask objective.

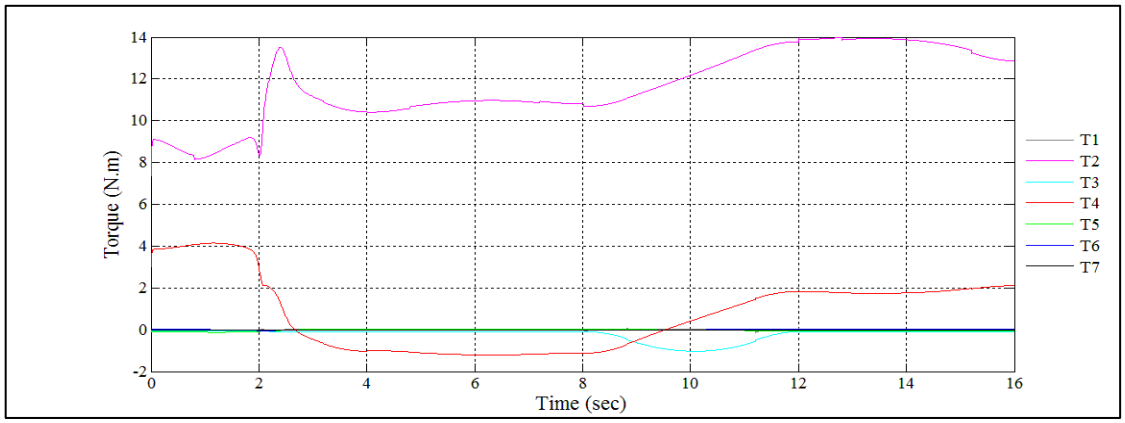


Figure 6.25. Controller output torque signals for Static Impact Force Minimization subtask

It can be observed from Figure 6.26 that the objective function magnitude was maximized throughout the simulation, which is needed for the purpose of minimizing the static impact force. The effect of the change in configuration between the 2nd and 4th seconds can be observed in this figure. The increase in magnitude as expected can also be observed with respect to the objective function of the previous simulation shown in Figure 6.20.

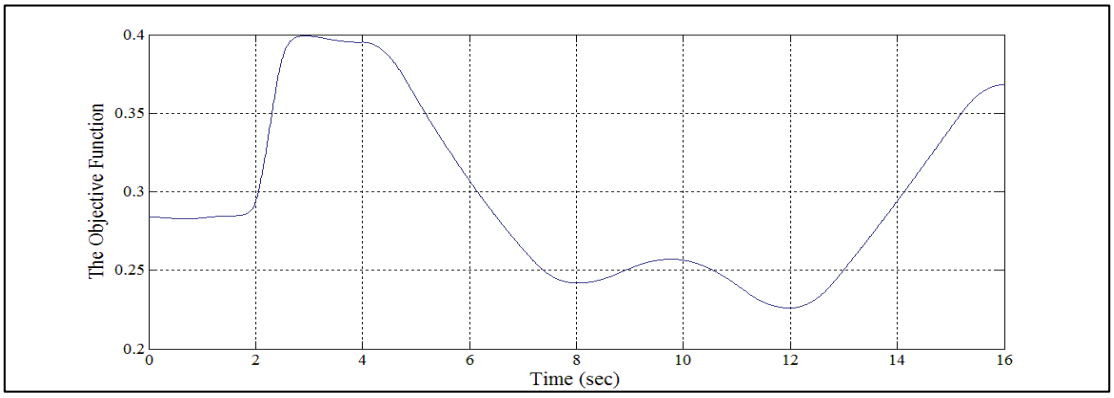


Figure 6.26. Objective function magnitude for Static Impact Force Minimization subtask

The configuration of the robot during the simulation is illustrated in Figure 6.27 and link positions are shown for each second of simulation. Figure 6.28 verifies the

validity of subtask controller by showing subtask error signal is bounded. However, some increase in the error value can be observed since the arm approaches the extended singularity between the 2nd and 4th seconds of the simulation.

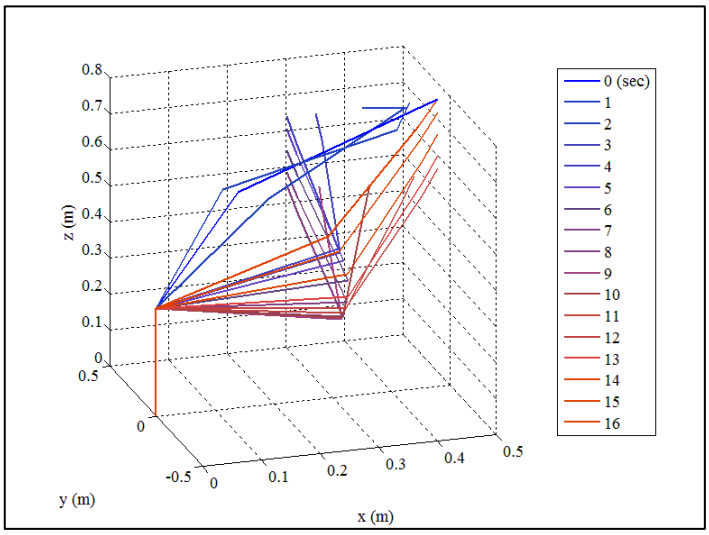


Figure 6.27. Robot arm motion during simulation for Static Impact Force Minimization subtask

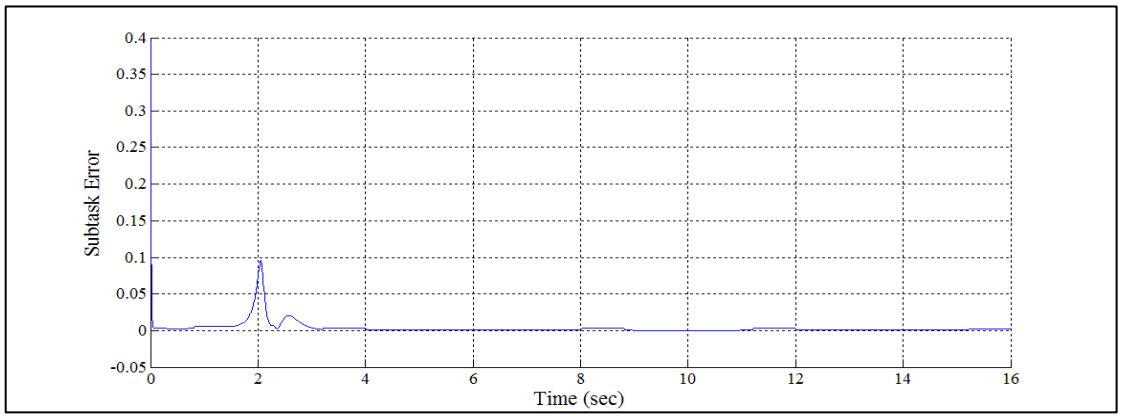


Figure 6.28. Subtask error signal for Static Impact Force Minimization subtask

6.3.5. Static and Moving Obstacle (point to point) Collision Avoidance Simulation Results

In this simulation test, static and moving obstacle (point to point) collision avoidance as presented in Sub-section 5.5.4 is utilized. Two simulations are executed separately; first for one static obstacle avoidance and second for two static obstacles avoidance subtask objectives.

Subtask's objective function controller parameter is tuned to be $v_m = 50$ for the case of one obstacle located at $\mathrm{X_O} = [0 \ -0.05 \ 0.6]^{\mathrm{T}}$ and $v_m = 30$ for the case of two obstacles located at $\mathrm{X_{O1}} = [0.15 \ 0.15 \ 0.55]^{\mathrm{T}}$ and $\mathrm{X_{O2}} = [-0.05 \ -0.15 \ 0.65]^{\mathrm{T}}$ respectively (It should be noted that the obstacles in both simulation results are considered to be static). Figure 6.29 shows the obstacles along with the robot arm.

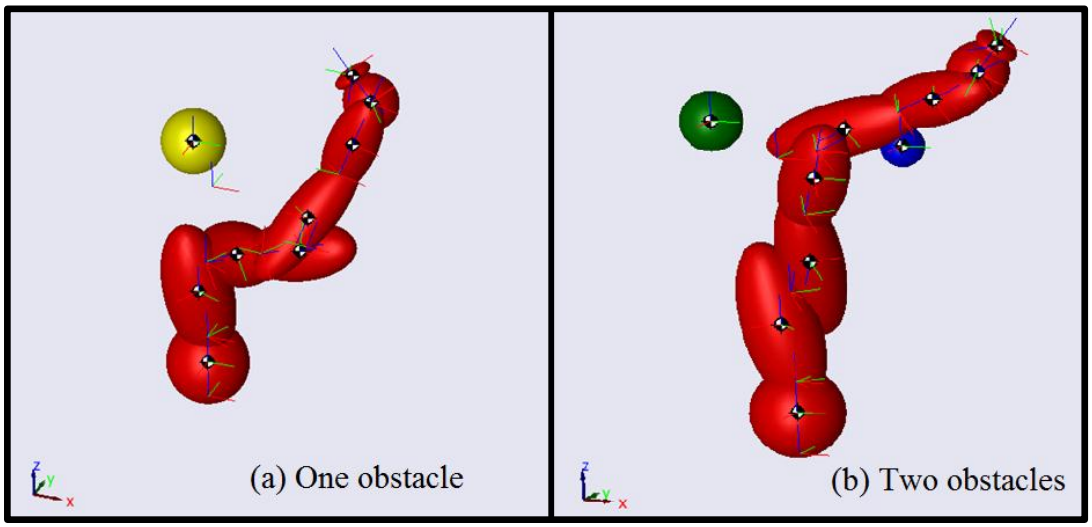
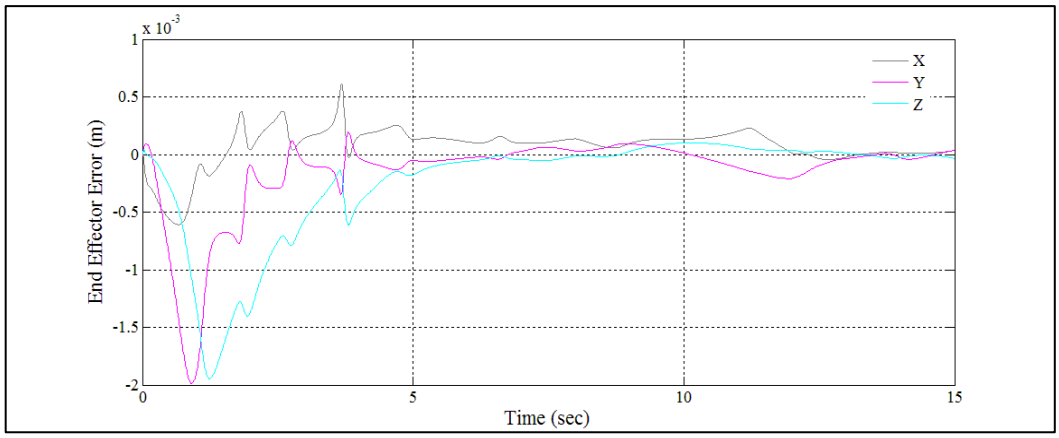
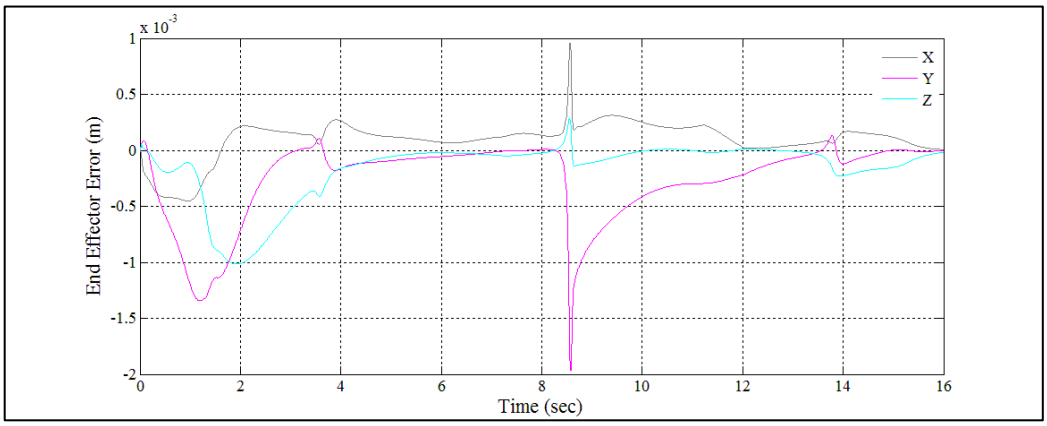


Figure 6.29. Obstacle avoidance; (a) For One obstacle, (b) For Two obstacles.

Figure 6.30 shows the tracking error for the end-effector position for each simulation; for one obstacle and for two obstacles avoidance. Magnitude of the errors are bounded which indicate that main-task objective is achieved for each simulation test.



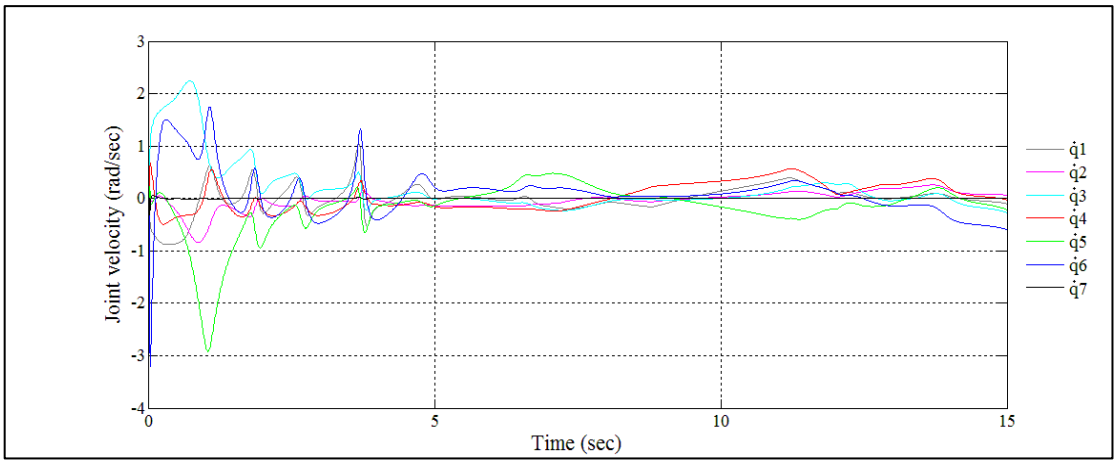
(a)



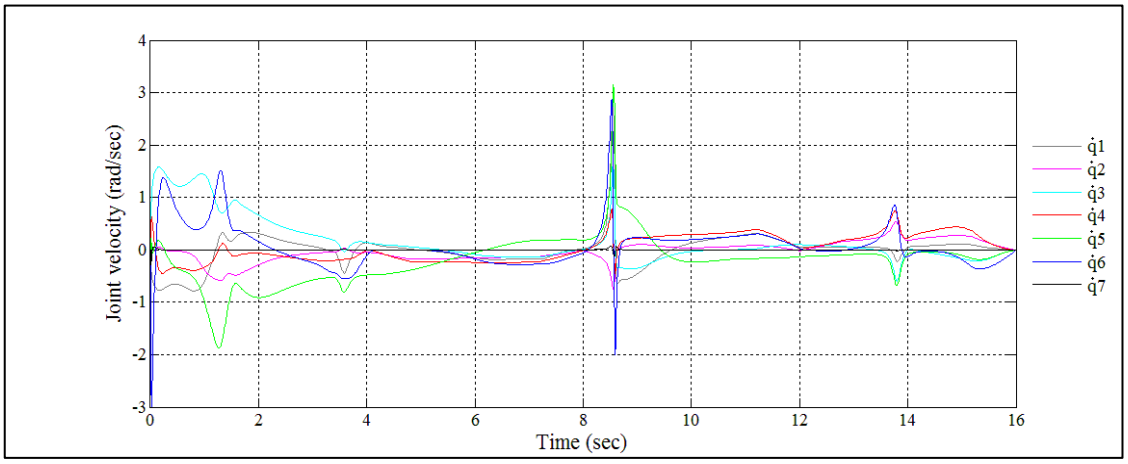
(b)

Figure. 6.30. End-effector position error; (a) One Obstacle Avoidance subtask, (b) Two Obstacles Avoidance subtask

Figure 6.31 shows joint velocities measured in one obstacle and two obstacles avoidance subtasks simulation tests. The joint velocities are observed to be relatively larger than all of the previous simulations. The escape velocity is minimized but does not go to zero as the manipulator is away from the obstacles, thus the joint velocities receive higher magnitudes in this subtask.



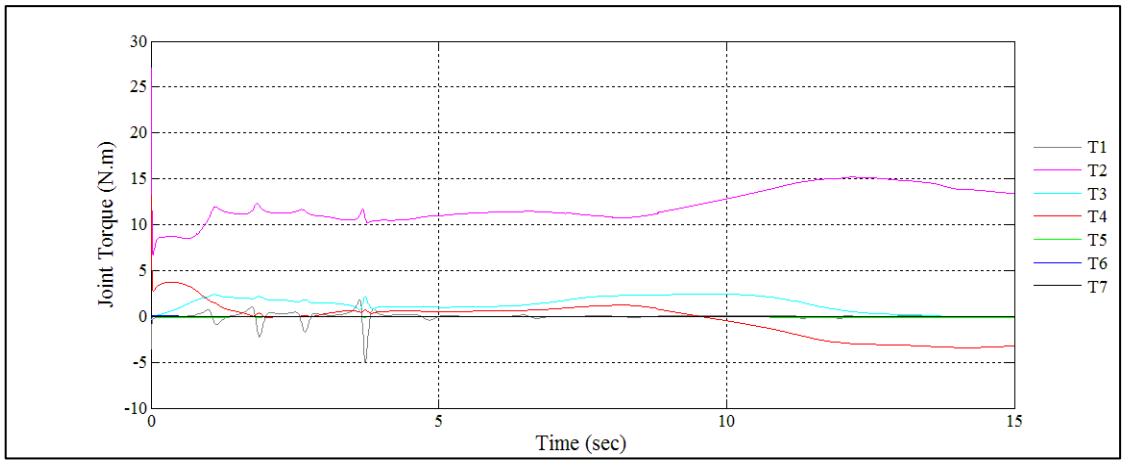
(a)



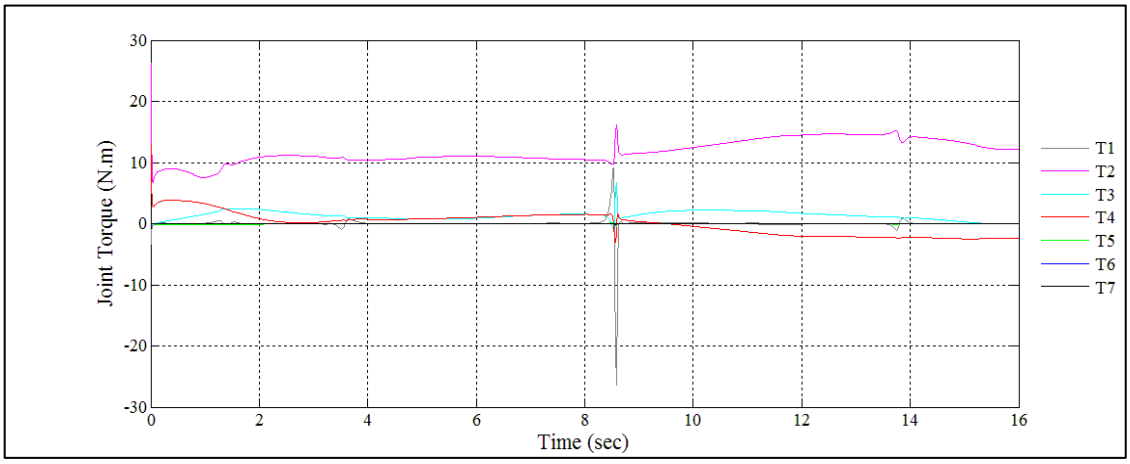
(b)

Figure 6.31. Joint velocities; (a) One Obstacle Avoidance subtask, (b) Two Obstacles Avoidance subtask

The torque signals required to execute both main-task and avoiding obstacles are presented in Figure 6.32. It can be noticed that joint velocities and input torque signals are bounded for both simulation tests.



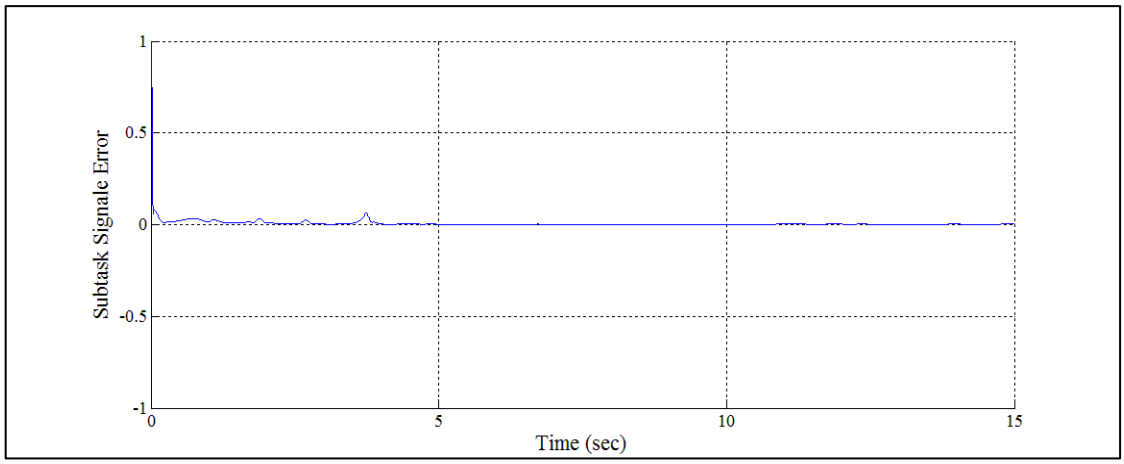
(a)



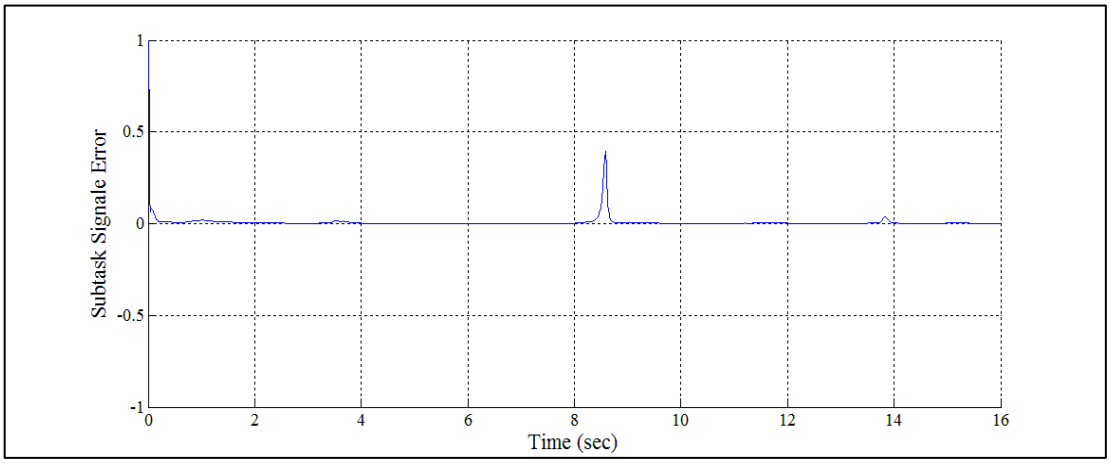
(b)

Figure 6.32. Controller output torque signals; (a) One Obstacle Avoidance subtask, (b) Two Obstacles Avoidance subtask

Figure 6.33 shows the validity of avoiding obstacles subtasks by showing that the subtask error signals are bounded. It can be noticed that this error is small relative to the previous simulation results.



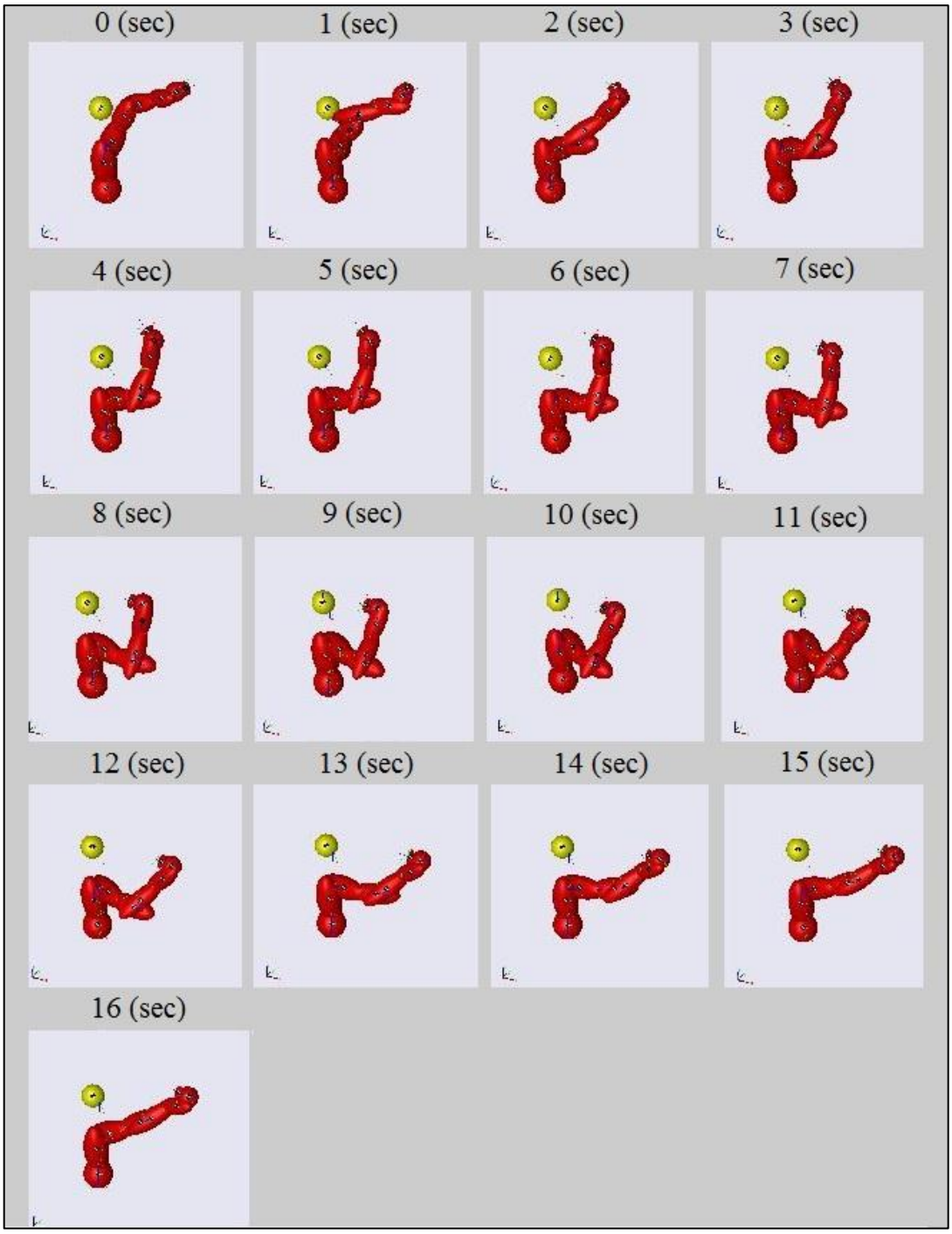
(a)



(b)

Figure 6.33. Subtask signal error norm magnitude; (a) One Obstacle Avoidance subtask, (b) Two Obstacles Avoidance subtask

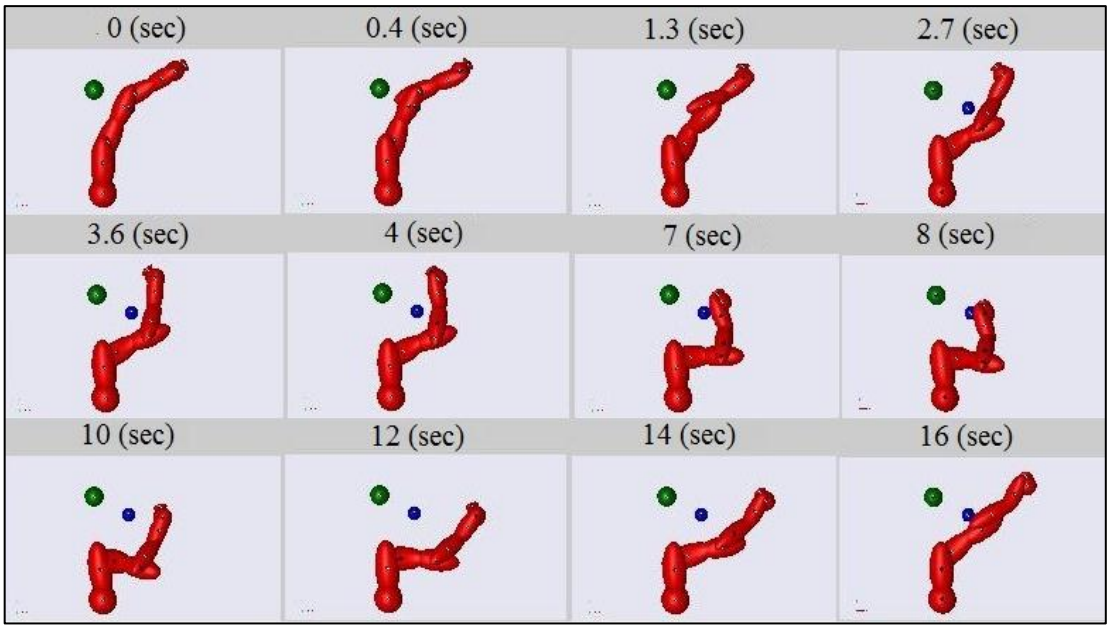
Obstacle avoidance sequences througout both of the two simulation test executions can be depicted in Figure 6.34. In both simulation tests, no collision happens between the robot body and the obstacles.



(a)

Figure 6.34. Obstacle avoidance sequences; (a) One Obstacle Avoidance subtask, (b) Two Obstacles Avoidance subtask.

(cont. on next page)



(b)

Figure 6.34. (cont.)

CHAPTER 7

CONCLUSIONS

7.1. Conclusions

New general subtask controller is designed by utilizing self-motion property of redundant robot manipulator in this study. The controller does not place any restriction on the self-motion of the manipulator; thus, the extra degrees of freedom are available for subtasks like maintaining manipulability, avoidance of mechanical limits and obstacle avoidance, etc.

The 7-DOF LWA4-Arm by SCHUNK GmbH is modeled for simulation test studies to validate the designed controller. Four subtasks, defined as minimization of the total joint motion and singularity avoidance, static impact force minimization / maximization, single- and multi-obstacle avoidance, are implemented separately to test the designed controller.

In all simulations, the main task is designated to be the tracking of the end-effector position trajectory. The end-effector position trajectory is followed with bounded errors in all simulation tests. Therefore, it is valid to state that the main objective of the proposed controller for end-effector position trajectory tracking is achieved regardless of the subtask. However, a slight difference in the error magnitude can be observed in some of the simulation test. The reason for that is when joint velocities are increased relatively for the same main-task but different subtasks, the disregarded Coriolis and centripetal forces in the controller have an increased effect on the error signal.

First subtask objective studied in this thesis is the minimization of joint motion. Results indicated that the total motion of the joints is minimized with respect to the uncontrolled self-motion case study. Motion in the null space is forced to be zero as a result of the designed subtask objective function. The second subtask was the singularity avoidance where the manipulability of the end-effector is maximized. However, the extended singularity cannot be avoided because the end-effector reaches the boundaries of its workspace. It can be observed from the results that determinant of

JJ^T is maximized. Minimizing or maximizing the static impact force of the manipulator was studied as the third subtask. The first set of simulation tests were conducted for the maximizing the static impact force and the second set were conducted for the minimization. The manipulability of the end-effector for maximizing the static impact forces is designated to be minimized along the desired direction (which can be the normal of the surface to be contacted). On the other hand, the manipulability of the end-effector for minimizing the static impact forces is designated to be maximized along the desired direction. In both cases, the effectiveness of the subtask controller can be observed from the manipulability along the desired direction (along the x-axis) plots which the objective function. Robot arm motion sketches for each second of the simulation are also presented to observe the configuration of the robot arm. The last subtask was selected to be obstacle avoidance. In the simulations static obstacle avoidance is studied, however, the idea can be extended for moving obstacle case by making the obstacle position as a function of time. The results indicate that the links move away from the obstacle relatively faster as they are near the obstacle. The reason for this phenomenon is that the objective function equation is selected to be an exponential function with negative power of the distance. This function can be modified if the motion of the links going away from the obstacle needs to be zero after some predefined distance. It should be also noted that the obstacle placement is taken as the mass center and a surface of influence (SOI) is not defined.

It can also be noticed that the uncontrolled self-motion of the robot (first simulation) results in undesired motion of the links. Therefore, it can be concluded that the self-motion definitely should be controlled in redundant manipulators. Furthermore it can be employed to execute another subtask beside the main-task. Overall, it can be stated that, in this thesis study, the stability and effectiveness of the designed controller is verified first mathematically and then by simulation results.

7.2. Future Work

Designed controller can be tested for other subtasks like; Joint Limit Avoidance, Robot Body Collision Avoidance, Minimizing Potential Energy, Whole Arm Grasping Control, etc. in the future studies.

Virtual model can be also improved by using Virtual Reality Sink block in Simulink to visualize the associated model (.wrl) file drawn in Solidwork during simulation instead of using SimMechanics visualization used in this thesis, which shows the model as an equivalent ellipsoids for each link.

This controller can also be tested by defining a 6-DOF task space but it is advisable to use more than 7-DOF redundant robot arm to perform the required main-task and subtask with more flexibility. Also, in this situation, it can be advised to select other methods than using pseudo-inverse method to find the inverse kinematics because, using pseudo inverse will lead to increased mathematics calculations due to the need of calculating the determinant of a larger Jacobian matrix.

In application specific case, this study can be extended for use in robot controllers designed for working with human. The use of redundancy along with minimizing the static force and obstacle avoidance can increase the level of safety in assistive and rehabilitation robotics. Furthermore the simulation results can be verified by carrying out real-world experimental study of the control algorithm developed in this thesis.

REFERENCES

Baillieul, J.; J. Hollerbach and R. W. Brockett (1984). Programming and Control of Kinematically Redundant Manipulators, Proc. 23rd Conf. on Decision and Control, Las Vegas, NV, U.S.A, pp. 768-774.

Balkan, T.; M. Kemal Özgören; M. Sahir Arıkan and H. Murat Baykurt (2001). A kinematic structure-based classification and compact kinematic equations for six-dof industrial robotic manipulators, Mechanism and Machine Theory, Vol. 36, Issue 7, pp. 817-832.

Canadian, Space Agency (2012). Website Accession date: 6-6-2012 , http:\\www.asc-csa.gc.ca/eng/default.asp.

Chen, J. L. ; J. S. Liu; W. C. Lee and T. C. Liang (2002). On-line multi-criteria based collision-free posture generation of redundant manipulator in constrained workspace, Robotica, Vol. 20, pp. 625-636.

Chiu, S. L. (1988). Task compatibility of manipulator postures, Int. Journal of Robotics Research, Vol. 7, No. 5, pp 13-21.

Colbaugh, R. and K. Glass (1995). Robust adaptive control of redundant manipulators, J. Intell. Robot. Syst., Vol. 14, pp. 68–88.

Conkur, E . Sahin and R. Buckingham (1997). Clarifying the definition of redundancy as used in robotics, Robotica, Vol. 15 Issue 5, pp. 583–586.

Das, H.; J-J. E. Slotine and T. B. Sheridan (1988). Inverse kinematic algorithms for redundant systems, IEEE Inter. Conf. on Robotics and Automation, Philadelphia, PA, U.S.A, pp. 43-48.

Dede, M. I. C. (2010). Virtual Prototyping of Robot Controllers, International Journal of Design Engineering, vol. 3 (3), pp. 276-288.

DLR (2012). Institute of Robotics and Mechatronics**,** German center. Website Accession http: www.dlr.de/rm/en.

Fraunhofer (2010). Institute for Manufacturing Engineering and Automation IPA website Accession : http://www.care-o-bot-research.org.

Gertz, M.W.; J. Kim and P. Khosla (1991). Exploiting redundancy to reduce impact force, IEEE/RSJ Workshop on Intel. Rob. and Sys., Osaka, Japan, pp. 179-184.

Golub, G. H. and C. F. Van Loan (1983). Matrix Computations; Johs Hopkins Studies in Mathematical Sciences. The Johns Hopkins University Press. Baltimore and London.

Guo, Z. Y. and T. C. Hsia (1990). Joint trajectory generation for redundant robots in an environment with obstacles. IEEE Inter. Conf. on Robotics and Automation, Cincinnati, OH, U.S.A, pp. 157-162.

Schneider, Hans-Christian and J. Wahrburg (2010). Simulation Model for the Dynamics Analysis of a Surgical Assistance Robot, Robot Surgery, Seung Hyuk Baik (Ed.), ISBN: 978-953-7619-77-0, InTech

Hollerbach, J. M. and K. C. Suh (1985). Redundancy Resolution of Manipulators through Torque Optimization, Proc. of IEEE Int. Conf. on Robotics and Automation, St. Louis, Missouri, U.S.A, pp. 1016-1021.

Hollerbach, J. M. and K. C. Suh (1987). Local versus Global Optimization of Redundant Manipulators, Proc. of IEEE Int. Conf. on Robotics and Automation, Raleigh, NC, U.S.A, pp. 619-624.

Hsu, P.; J. Hauser and S. Sastry (1989). Dynamic Control of Redundant Manipulators, Journal of Robotic Systems, Vol. 6, pp. 133-148.

Kemeny, Z. (1999). Design and evaluation environment for collision-free motion planning of cooperating redundant robots, Periodica Polytechnica Ser. El. Eng, vol. 43, no. 3, pp. 189-198.

Khatib, O. (1983). Dynamic control of manipulators in operational space, in Proc. 6th IFTOMM Congr. Theory of Machines and Mechanisms, New Delhi, India, pp. 1–10.

Kwang-Kyu, L. and B. Martin (2007). Obstacle Avoidance for Redundant Robots Using Jacobian Transpose Method, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, U.S.A, pp. 3509-3514.

Leigh, R. H.; B. Daniel and J. Beata (2012). Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. Nature international weekly journal of science, vol. 485, Issue 7398, pp. 372–375.

Lin, Z.; R.V. Patel and C. A. Balafoutis (1995). Augmented impedance control: An approach to impact reduction for kinematically redundant manipulators, Journal of Robotic Systems, vol. 12, Issue 5, pp. 301- 313.

Nakamura, Y. (1991). Advanced Robotics Redundancy and Optimization, Addison-Wesley Pub. Co., MA, U.S.A.

Oh, Y. and W. K. Chung (1999). Disturbance-observer-based motion control of redundant manipulators using inertially decoupled dynamics, IEEE/ASME Trans. Mechatronics Vol. 4, Issue 2, pp. 133–146.

Özbay, U.; H. Türker Şahin and E. Zergeroğlu (2008). Robust tracking control of kinematically redundant robot manipulators subject to multiple self-motion criteria, Robotica, Vol. 26, pp. 711–728.

Patel, R. V. and F. Shadpey (2005). Control of Redundant Robot Manipulators: Theory and Experiments, 1st ed., Springer-Verlag Berlin Heidelberg, Germany.

Peng, Z. X. And N. Adachi (1993). Compliant motion control of kinematically redundant manipulators, IEEE Trans. Robot. Automat., vol. 9, Issue 6, pp. 831-837.

Rajiv, V. D.; A. E. James and M. Scott (1991). Real-Time Implementation of an Optimization Scheme for Seven-Degree-of-Freedom Redundant Manipulators, IEEE Transactions On Robotics And Automation, Vol. 1, No. 5, pp. 579-588.

Reza, N. J. (2010). Theory of Applied Robotics: Kinematics, Dynamics, and Control. 2nd ed., Springer New York, U.S.A.

Reznik, D. and V. Lumelsky (1995). Sensor-based motion planning in three dimensions for a highly redundant snake robot. Advanced Robotics, Vol. 9, No. 3, pp. 255–280.

Schunk, (2011). Website Accession date: 24-11-2011 http://www.schunk.com/schunk_files/attachments/ModularRobotics_2010-06_EN.pdf.

Schunk modular robotics, (2011). Website Accession date: 24-11-2011 http://www.schunk-modular- robotics.com/left-navigation/service- robotics/ servicedownload/simulationcad/cad-data.html.

Sciavicco, L. and B. Siciliano (1988). A solution algorithm to the inverse kinematic problem for redundant manipulators, IEEE Transactions on Robotics and Automation, Vol. 4, Issue 4, pp. 403-410.

Seraji, H. (1989). Configuration control of redundant manipulators: Theory and implementation, IEEE Trans. Robotics and Automation, Vol. 5, Issue 4, pp. 472–490.

Seraji, H. (1991). Task options for redundancy resolution using configuration control, IEEE Conf. on Decision and Control, Brighton, England, pp. 2793- 2798 vol. 3.

Shuai, Li; C. Sanfeng; B. Liu; L. Yangming and L. Yongsheng (2012). Decentralized kinematic control of a class of collaborative redundant manipulators via recurrent neural networks, Neurocomputing, Vol. 91, pp. 1–10.

Siciliano, B.; S. Lorenzo; V. Luigi and O. Giuseppe (2009). Robotics Modeling, Planning and Control, Advanced Textbooks in Control and Signal Processing, Springer-Verlag London.

Spong, M. W. and M. Vidyasagar (1989). Robot Dynamics and Control, NY, U.S.A, John Wiley and Sons, Inc.

Spong, M. W.; S. Hutchinson and M. Vidyasagar (2005). Robot Modeling and Control, John Wiley & Sons, Inc., NY, U.S.A.

Tatlıcıoğlu, E.; D. Braganza; T. C. Burg and D. M. Dawson (2009). Adaptive control of redundant robot manipulators with subtask objectives, Robotica, Vol. 27, Issue 6, pp. 873–881.

Walker, I. D. (1990). The use of kinematic redundancy in reducing impact and contact effects in manipulation, Proc. IEEE International Conf. Robotics and Automation, Cincinnati, OH, U.S.A, pp. 434-439.

Walker, I. D. (1994). Impact Configurations and Measures for Kinematically Redundant and Multiple Armed Robot Sysytems, IEEE transactions on robotics and automation, Vol. 10. No. 5. pp. 670-683.

Wang, J.; Y. Li and X. Zhao (2010). Inverse Kinematics and Control of a 7-DOF Redundant Manipulator Based on the Closed-Loop Algorithm, International Journal of Advanced Robotic Systems, Vol. 7, No.4, pp. 1-9.

Wang, D. and Y. Hamam (1992). Optimal trajectory planning of manipulators with collision detection and avoidance, Int. J. Robotics Research, Vol. 11, No. 5, pp. 460-468.

Wolovich, W. A. and H. Elliott (1984). A computational technique for inverse kinematics, Proc. of Conference on Decision and Control, Las Vegas, NV, U.S.A Vol. 23, pp. 1359-1362.

Yangmin, L. and H. L. Sio (2001). Kinematics Control of Redundant Manipulators Using CMAC Neural Network, The 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI2001), Orlando, FL, U.S.A, vol. IX, pp.274-279.

Yoshikawa, T. (1984). Analysis and Control of Robot Manipulators with Redundancy, Proc. First Int. Symp.on Robotics Research, Cambridge, MIT Press, pp. 735-748.

Zergeroğlu, E.; D. M. Dawson; I. D. Walker and P. Setlur (2004). Nonlinear tracking control of kinematically redundant robot manipulators, IEEE/ASME Trans. Mechatronics, vol. 9, No.1, pp. 129– 132.