نموذج رقم (1)

إقـــــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

Sliding Mode Fuzzy Controller Applied to Robot Manipulator

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وإن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل لنيل درجة أو لقب علمي أو بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.

DECLARATION

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification.

Student's name: **Mohammed Ali Abu Nada**

Signature: [illegible]

Date: 11-5-2014

Islamic University of Gaza
Deanery of Graduate Studies
Faculty of Engineering
Electrical Engineering Department

Master Thesis

Sliding Mode Fuzzy Controller Applied to Robot Manipulator

Mohammed Ali Abu Nada

Advisor
Dr. Basil Hamed
Dr. Iyad Abuhadrous

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Electrical Engineering

March 2014

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza

مكتب نائب الرئيس للبحث العلمي والدراسات العليا   هاتف داخلي 1150

الرقم ج س غ/35/ Ref
التاريخ 2014/03/08 Date

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ **محمد علـي حسين أبو نـدى** لنيل درجة الماجستير في كلية *الهندسة* قسم <u>الهندسة الكهربائية– أنظمة التحكم</u> وموضوعها:

Sliding Mode Fuzzy Controller Applied to Robot Manipulator

وبعد المناقشة التي تمت اليوم السبت 07 جمادي أول 1435هـ، الموافق 2014/03/08م الساعة الثانية عشرة ظهراً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. باسل محمود توفيق حمد	**مشرفاً ورئيساً**	[illegible]
د. اياد محمد أيوب أبو هدروس	**مشـــرفـــاً**	[illegible]
د. حاتم علي سالم العايدي	**مناقشاً داخليــاً**	[illegible]
د. أسعد نمر محمود أبو جاسر	**مناقشاً داخليــاً**	[illegible]

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في كلية *الهندسة*/ قسم <u>الهندسة الكهربائية - أنظمة التحكم</u>.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله ولزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق،،،

مساعد نائب الرئيس للبحث العلمي والدراسات العليا

[illegible]

أ.د. فؤاد علي العاجز

ABSTRACT

Sliding Mode Fuzzy Controller (SMFC) which has sliding surface gains is on-line tuned by minimum fuzzy inference algorithm. The main goal is to guarantee acceptable trajectories tracking between the robot manipulator actual and desired trajectory.

An educational simulation tool objective is to make practical teaching, learning kinematics and dynamic modeling and to apply different controllers on robot manipulator.

Pure Sliding Mode Controller (SMC) and Sliding Mode Fuzzy Controller have difficulty in handling unstructured model uncertainties. It is possible to solve this problem by combining sliding mode fuzzy controller and fuzzy-based tuning. Since the sliding surface gain is adjusted by fuzzy based tuning method, the sliding surface slope updating factor of fuzzy-based tuning part can be changed with the changes in error and change of error rate between half to one. Sliding surface gain is adapted on-line by sliding surface slope updating factor. In pure sliding mode controller and sliding mode fuzzy controller, the sliding surface gain is chosen by trial and error, which means that pure sliding mode controller and sliding mode fuzzy controller must have a prior knowledge of the system uncertainty.

Fuzzy-based tuning sliding mode fuzzy controller is a model-free stable control for robot manipulator. It is a one of the best solution to eliminate chattering phenomenon with switching function in structure and unstructured uncertainties.

ملخص

يحتوي المتحكم الضبابي ذو الوضع المنزلق على معامل تتم معايرة سطحه المنزلق لحظياً عن طريق خوارزمية ضبابية موجزة. إن الهدف الرئيس هو ضمان مسار دقيق بين ذراع المناورة الفعلي و المسار المطلوب.

يصعب التعامل مع النموذج في كل من المتحكم ذو الوضع الضبابي والوضع المنزلق في حالة عدم الدقة. ومن الممكن حل هذه المشكلة عن طريق ربط كل من المتحكم الضبابي ذو الوضع المنزلق مع المتحكم ذو الوضع المنزلق الذي تتم معايرته لحظياً.

حيث أن معامل السطح المنزلق يتم ضبطه باستخدام طريقة المعايرة الضبابية اللحظية.فإن معامل التحديث للميل الخاص بالسطح المنزلق للجزء المعتمد على المعايرة يمكن أن يتغير مع التغييرات في الخطأ والتغير في معدل الخطأ من نصف إلى واحد. يتم ضبط معامل السطح المنزلق آلياً عن طريق معامل ميل السطح المنزلق المحدث. في كلا من المتحكم ذو الوضع المنزلق والمتحكم الضبابي ذو الوضع المنزلق يتم اختيار معامل السطح المنزلق عن طريق التجربة والخطأ وهو ما يعني أن المتحكم ذو الوضع المنزلق والمتحكم الضبابي في الوضع المنزلق يجب أن يكون لديهم سابق معرفة عن عدم دقة النظام.

إن المتحكم الضبابي ذو الوضع المنزلق المعدل لحظياً يعتبر نموذجاً ثابتاً غير مقيد للتحكم في ذراع المناورة الآلي. وبالتالي فهو يعتبر واحداً من أفضل الحلول للقضاء على ظاهرة التذبذب (التشتت) مع تغيير الوظيفة في حالتي عدم الدقة.

This thesis is dedicated

To the soul of my Parents

To my faithful wife

My Kids HALA & YAZAN

ACKNOWLEDGEMENT

I thank Allah, the lord of the worlds, for His mercy and limitless help and guidance. My peace and blessings be upon Mohammed the last of the messengers.

I would like to express my deep appreciation to my thesis advisor Dr. Basil Hamed for providing his advice and encouragement. Also, I am heavily indebted to my supervisor Dr. Iyad Abu Hadrous for his excellent guidance, the warm discussion and regular meeting I had with him during research. I would like to thank the other committee members Dr. Assad Abu-Jasser and Dr. Hatem Elaydi for providing their valuable suggestions. In addition, my thanks go to my friends and colleagues for their advices and supports.

There are no words that describe how grateful I am to my parents for raising me and for their support and encouragement through years. I am grateful to my wife and my kids for their patience and understanding during my busy schedule.

Finally, My appreciation goes to all colleagues and friends for their suggestions and encouragement during the writing of this thesis. A very sincere appreciation to my dear friends, Eng. Ahmed Al Assar, Eng. Said Abu Al-Roos and Eng. Fady Farhouda for helping me to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

DC	Direct Current
PID	Proportional Integral Derivative
FLC	Fuzzy Logic Controller
FLS	Fuzzy Logic Supervisory
SMC	Sliding Mode Controller
ISMC	Integral Sliding Mode Controller
OL	Open Loop
CL	Closed Loop
IJC	Independent Joint Controller
SISO	System Input System Output
DOF	Degree of Freedom
ZN	Ziegler Nicholas
VSC	Variable Structure Control
MF	Member Function
SMFC	Sliding Mode Fuzzy Controller
PUMA	Programmable Universal Manipulation Arm
DOF	Degree of Freedom
GUI	Graphical User Interface
Z-N	Ziegler-Nichols
DH	Denavit Hartenberg
FK	Forward Kinematics
IK	Inverse Kinematics
OS	Overshoot
VSC	Variable structure control

NOMENCLATURE

J_{m_i}	***Moment of inertia***
R_i	***Armature resistance***
L_i	***Armature inductance***
B_{v_i}	***Viscous friction constant***
K_{v_i}	***Back EMF constant***
$\xi = m_L$	***Varying payload mass carried by the manipulator***
$\overline{A}$	***Nominal value of matrix A***
$\overline{B}$	***Nominal value of matrix B***
α	***Norm bound of continuous function H***
β	***Norm bound of continuous function E***
λ_i	***Desired poles***
γ_i	***Parameters of PISMC***
α	***Slop factor of SMC***

CHAPTER 1 INTRODUCTION

1.1 Motivation

Trajectory tracking with high accuracy is a very challenging topic in drive robot manipulator control. This is due to the nonlinearities and input couplings present in the dynamics of the manipulator. Robot manipulator field is one of the interested educational fields, therefore modeling and analysis of the robot manipulators and applying control techniques are very important before using them to work with high accuracy.

Determining the mathematical model of the robot manipulator system and specifying the corresponding control strategies based on these models so that the desired system response and performance is achieved.

Interesting of applying nonlinear controller such as Sliding Mode Controller (SMC) to Puma560 Manipulator Robot. Simulation environment using SIMULINK \GUI maybe usefully for studying of robot Manipulator and courses related to robotics.

1.2 Background

The robot has been applied to a wide variety of mechanical devices. An important class of robots are the manipulator arms, such as the PUMA robot. These manipulators are used primarily in materials handling, welding, assembly, spray painting, grinding, and other manufacturing applications. Robot manipulators are basically multi-degree of freedom positioning devices [1].

There are three main subsystems in robot manipulators: mechanical system, electrical system, and control system [2]. Mechanical system comprises of all movable parts. It consists of a group of links (rigid bodies) connected together by joints which allow the motion for the desired link. The mechanical system is used to move the end effector to desired position with respect to the base. This movement depends on the electrical system (e.g. motors, power amplifiers, and other electronic circuits) and it is done by some rotations and translations to the other links. Robot manipulators construction is divided into two main classes: serial manipulator and parallel manipulator [2].

Serial manipulators consist of some links connected in series, which form an open loop chain. At the end of the chain, the end effector is connected to the base by single kinematic chain. On the other side, the parallel manipulators form a closed loop chain finished by the end effector and is connected to the base by two, or more kinematic chains (e.g. arm, or legs). The only drawback of the parallel manipulator over the serial manipulator is that the parallel robots manipulators suffer from limited workspace as compared with serial robot manipulators [3] and [5].

Figure 1.1 shows a schematic diagram of a robot manipulator. It consists of three joints and each one of these joints will have a motor to actuate the desired link. There are two widespread types of joints on this manipulator. A cylinder represents the first type, and it allows only relative rotation between two links. This type of joint is called ***revolute*** or ***rotary*** joint (e.g. human joints), and it is the most common joint type in robots.

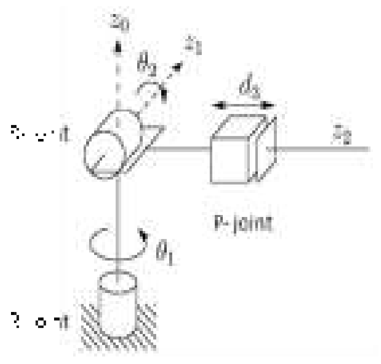


Figure 1.1 Manipulator with Revolute and Prismatic Joints [1]

The second type of joints is called ***prismatic*** or ***sliding*** joint. It is represented by square box. This type of joints allows only linear relative motion between two links along its axis. Both types are denoted as R and P joints. The robot manipulator whose all joint variables are prismatic is known as a Cartesian manipulator, while the robot whose all joints are revolute is known as an articulated manipulator. The robot manipulator in Figure1.1 is called revolute revolute prismatic (RRP) manipulator. For more information and classification of robot manipulators readers may be referred to [3], [4] and [5].

The third system in robot manipulators is a control system consists of some devices and tools (e.g. sensors, controllers, and knowledge base) that provide convenient duty to robot manipulators. When the controller is moving, the robot manipulator during the working environment, the sensor or feedback system is gathering the information about the robot manipulator state and the surrounding circumstances, and then exploiting the information to modify and enhance the system behavior.

To achieve higher speed and accuracy for robot manipulator over a wide range of application, the control technique needs to be improved. In general, the dynamic performance of a robot manipulator is directly dependent on the efficiency of the control algorithm and dynamic model of the robot manipulator. Thus the robot manipulator control problem consists of determining the mathematical model of the robot manipulator system and specifying the corresponding control strategies based on these models so that the desired system response and performance is achieved.

The main challenges in the motion control problem are the complexity of the dynamics, and uncertainties, both parametric and dynamic. Parametric uncertainties arise from imprecise knowledge of the dynamics, while dynamic uncertainties arise from joint and link, actuator dynamics, friction, sensor noise, and unknown environment dynamics.

The external input of a system is called the reference .When one or more output variables of a system need to follow a certain reference over time, a controller manipulates the inputs to a system to obtain the desired effect on the output of the system.

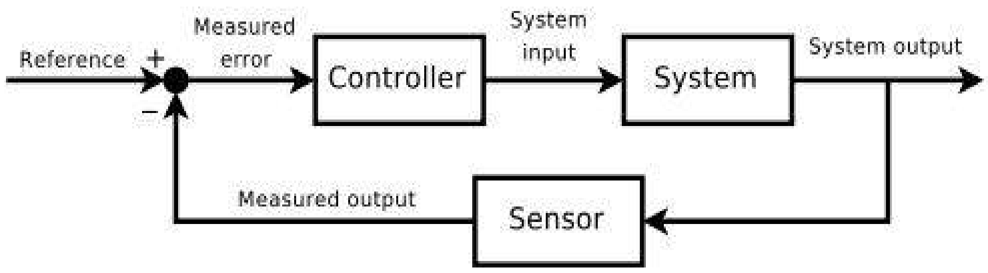


Figure 1.2 Block diagram of a closed-loop control system

1.2.1 Linear and Nonlinear Control

There are two methods used in control theory to control systems, linear method and nonlinear method. Using linear control is applicable only when the controlled system can be modeled mathematically [1]. The facts that the majority of physical systems have nonlinear characteristics; hence, linear controllers fail to meet the requirements due to system nonlinearities. The variations and the nonlinear parameters such as gear backlash, load variations and other parameters have unpredictable effects on the controlled systems (e.g. robot manipulator) diminish the performance. Therefore, the robot manipulator may be considered as a linear model when it works on small space, or it has a large gear ratio between the joints and their links. Nonlinear methods considered as general case when compared to linear methods because it can be applied successfully on the linear methods, but linear method is not sufficient to solve and control nonlinear problems. Common methodologies are used to solve the nonlinearities in control systems such as sliding mode control, and state feedback control are discussed in [4].

1.2.2 Control Techniques

Due to uncertainty and instability effects, unknown or unpredictable inputs that manipulate the plant output to the incorrect target. These inputs are called disturbance or noise, so analyzing and designing the mathematical model of the system includes the controller and plants to get the desired behavior is required. Many control techniques have been proposed to control robot manipulator ranging in complexity from linear to the advanced control system, which compute the robot dynamic and save it from damage in real environments. Three different control schemes namely SMC, SMFC, and Online tuning SMFC will be implemented through this thesis. The performance of these controllers will be based on the high precision in reducing the overshoot, minimizing steady state error, damping unwanted vibration of robot manipulator, and handling the unpredictable disturbances.

Sliding mode controller (SMC) is a significant nonlinear controller under condition of partly uncertain dynamic parameters of system. SMC is an important robust control approach. For the class of systems to which it applies, sliding mode controller design provides a systematic approach to the problem of maintaining stability and consistent performance in the face of modeling imprecision. On the other hand, by allowing the tradeoffs between modeling and performance to be quantified in a simple fashion, it can illuminate the whole design process. The sliding mode control which advantage is robustness with respect to model uncertainties and disturbances [11].

SMC controller is used to control of highly nonlinear systems especially for robot manipulators, because this controller is a robust and stable conversely, pure sliding mode controller is used in many applications; it has two important drawbacks namely; chattering phenomenon, and nonlinear equivalent dynamic formulation in uncertain dynamic parameter [11] and [12].

Fuzzy logic theory is used to estimate the system dynamic.However fuzzy logic controller is used to control complicated nonlinear dynamic systems, but it cannot guarantee stability and robustness. Fuzzy logic controller is used in adaptive methodology and this method is also can applied to nonlinear conventional control methodology to improve the stability, increase the robustness, reduce the fuzzy rule base and estimate the system's dynamic parameters [7].

1.3 Literature Review

Kinematics analysis of industrial and educational robots such as PUMA 560, SCARA, and SG5-UT robot manipulators [13]. Other papers discussed control technique problems, such as PID, FLC and other techniques [1], [2], [14], and [20].

Kinematics of robot arm was mathematically modeled using a Denavit Hartenberg (DH) method [3], [4] and [5]. Forward and Inverse equation analysis, were generated and implemented using a simulation program [16] and [17]. In [14] Annand derived the kinematics analysis of PUMA 560, and calculated the equation of motion of the robot by deriving the so-called Euler-Lagrange equations. In [18] achieving a high level of complexity for robotic system design was straightforward and highly intuitive when using the PTOLEMY II software environment. After deriving the inverse kinematics equations, Antonia used PTOLEMY II to design, and simulate robot arm. The benefit of this software is that designing complicated system requires simple building blocks.

Position control performed using independent joint control in [3]. This method was using PID controller, and it worked by controlling each joint independently. The coupling effect between the joints and links could be ignored if the gear ratio was large.

The study, fuzzy supervisory have attracted attention in many papers through the history of FLC. Good presentation on this subject presented by Ahmed Al Assar [1] the main idea presented how the parameters of the PID controller adapted on-line. The results showed that the variety of the process can be satisfactory controlled by the FSLC and these results the better in comparison to the PID results. Due to the characteristic variations in the physical system, PID controller may not be sufficient.

The principles of sliding mode control can be found in many publications [23], [24], [25] where the design of control system for robot manipulator using the sliding mode control algorithm was presented. According to the simulation results of PUMA 560 robot manipulator the system with proposed algorithm works properly, and the presented control algorithm shows a good robustness with respect to the robot model uncertainty.

The control algorithm does not require an accurate knowledge of the physical parameters of the manipulator, the bounds of the parameters are sufficient to construct the controller Young [20]. However, the discontinuous control law is very difficult for realization in practice and discontinuous controller results in chattering effects of the control signal. To reduce the chattering effects due to the discontinuous control inputs SMFC is used [21].

To overcome the problem of reduced order dynamics, a variety of the sliding mode control known as the Integral Sliding Mode Control has been successfully applied in a variety of control. Different from the conventional SMC design approaches, the order of the motion equation in ISMC is equal to the order of the original system, rather than reduced by the number of dimension of the control input. The method does not require the transformation of the original plant into the canonical form. Moreover, by using this approach, the robustness of the system can be guaranteed throughout the entire response of the system starting from the initial time instance [35].

Chattering phenomenon can cause some problems such as saturation and heat the mechanical parts of robot manipulators or drivers. To reduce or eliminate the chattering, various papers had been reported by many researchers which classified into two important methods: boundary layer saturation method can be estimated uncertainties method [6] and [22]. In boundary layer saturation method, the basic idea is the discontinuous method replacement by saturation (linear) method with small neighborhood of the switching surface. This replacement caused to increase the error performance against the considerable chattering reduction. Slotine and Sastry introduced boundary layer method instead of discontinuous method to reduce the chattering [23].

In [24], Temeltas proposed fuzzy adaption techniques and applied to SMC to have robust controller and solves the chattering problem. In this method however system's performance is better than sliding mode controller but it is depended on nonlinear dynamic equations.

Yoo and Ham proposed a MIMO fuzzy system to help the compensation and estimation the torque coupling. This method can only tune the consequence part of the fuzzy rules [17]. Medhafer et al proposed an indirect adaptive fuzzy sliding mode controller to control nonlinear system. This MIMO algorithm, applies to estimate the nonlinear dynamic parameters. Compared with the previous algorithm the numbers of fuzzy rules have reduced by introducing the sliding surface as inputs of fuzzy systems[19].

1.4 Problem Statement

Since the complete dynamic equations of the robot manipulator are highly nonlinear, coupled, and time varying. Furthermore, a varying payload carried by the manipulator during a task will create uncertainties in the manipulator dynamics.

Conventional linear controllers are inadequate and inappropriate, when used in dynamic systems such as serial link robot manipulator.

The pure sliding mode controller with switching function cause chattering phenomenon in certain and uncertain systems.

The nonlinear equivalent dynamic problem in uncertain system is the second challenge in pure sliding mode controller.

Pure sliding mode controller and sliding mode fuzzy controller have difficulty in handling unstructured model uncertainties.

1.5 Objective

- The overall objective is to design a sliding mode controller for PUMA robot that will achieve stability, robustness and reliability.
- Design a kinematic and dynamic model for the PUMA 560 robot manipulator.
- Design a simulation environment using GUI and SIMULINK.
- To eliminate the chattering phenomena.
- To estimate the system dynamics, fuzzy inference system is introduced

1.6 Methodology

This work consists of the flowing main stages:

Derive the forward and inverse kinematics equations of the robot.

Decomposition of the dynamic model using SIMULINK.

Two approaches for controlling PUMA Robot:

First approach is by using PISMC controller

Second approach is by using SMC and SFMC Controllers.

Finally, combining SMFC and fuzzy-based tuning with reduced rule base.

1.7 Contribution

- Design the dynamic model PUMA Robot using SIMULINK.
- Solving the chattering problem in pure SMC using fuzzy logic control with reduced rule base using trial and error.
- Solving the problem of handling unstructured model uncertainties by combining SMF controller and fuzzy-based tuning.

This study can be used as a document of reference for other researches that are interested in this area of research.

1.8 Thesis Outline

This section outlines the overall structure of the thesis, and provides a brief description for each chapter.

Chapter 2 provides some basic knowledge about robot manipulators and presents two common problems in a robot manipulator: the first one is the kinematics model of the robot manipulator; the kinematics problem separated into two parts: the forward kinematics and the inverse kinematics. The second problem that will be discussed through this chapter is the dynamic modeling and design the model using SIMULINK.

Chapter 3 presents most common controllers used in control theory; PID control and fuzzy logic control.

Chapter 4 presents the idea of the sliding mode controller. A preliminary of some basic concepts for sliding theory are discussed, and the PISMC design will be applied.

Chapter 5 presents the experimental tool GUI will presented and the simulation results of kinematics model and different controllers using SIMULINK and MATLAB are shown. It also shows the performance of pure SMC,SMFC and on-line SMFC with min. rule base.

Chapter 6 summarizes the work presented in this thesis and indicates some recommendation and suggestion for future works.

CHAPTER 2 MODELLING OF ROBOT MANIPULATOR

2.1 Introduction

The most important initial step in the controlling an industrial robot is to obtain a complete and accurate mathematical model of the robot manipulator. This model is useful for computer simulation of the robot arm motion and synthesis processes before the controller is applied into real robot action.

The purpose of manipulator control is to maintain the dynamic response of a manipulator in accordance with pre-specified objectives. The dynamic performance of a manipulator directly depends on the efficiency of the control algorithms and the dynamic model of the manipulator [4]. Since the actuators are part of robot manipulator system, it is necessary to consider the effect of actuator dynamics. Therefore, it is important to study the Kinematic and dynamic model.

2.2 Kinematic Modeling

Kinematics is the science of motion. In kinematics, the position, orientation, velocity, and acceleration of the robot manipulator are studied from the perspective of spatial geometry. To analyze the geometry, a link frame based on Denavit-Hartenberg description is attached to each link of the robot manipulator, kinematics are dividing into Forward (FK) and Inverse (IK) kinematics.

Figure (2.1) below shows a simplified block diagram of kinematic modeling.

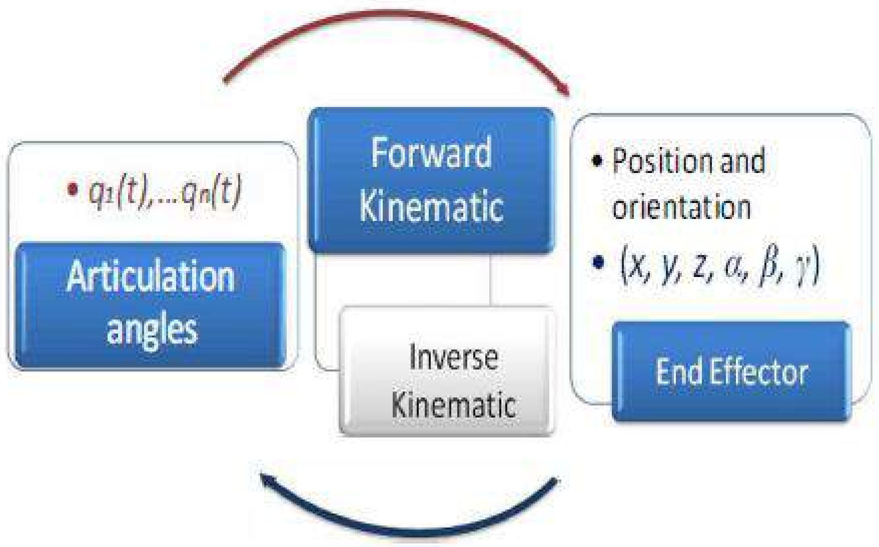


Figure 2.1 Kinematics Block Diagram.

A commonly used convention for selecting frames of reference in robotic applications is the Denavit-Hartenberg or D-H convention as shown in Figure (2.1).

2.2.1 Forward Kinematic

The forward kinematic equations, describe the functional relationship between the joint variables and the position and orientation of the end-effector. Suppose the robot has i-links, the joints and links numbered from 1 to i and 0 to i respectively. The joint variables are denoted by q_i. In the case of prismatic joint, q_i represents the displacement, similarly q_i represent the angle of rotation for the revolute joint.

Figure 2.1 illustrates the kinematic diagram and the frame assignment of a robot manipulator with n-DOF. We will derive the forward kinematics for i-links robot manipulator according to the DH convention.

Consider a fixed frame $o_0x_0y_0z_0$ and the rotation frame $o_1x_1y_1z_1$. The orientation is represented as a series of three revolute about a combination of the principle axes of the link frame. The rotation of the rotated frame about the fixed frame represented by the three angles α, β and γ. The first rotation about z axis by angle α, and the next rotation about current y by angle β and the third rotation about the current z axis by the angle γ.

According to [3] the rotational transformation matrix that represents the position of the frame i with respect to frame 0 is expressed in equation (2.1). This equation represents the rotation matrix of the ZYZ Euler angles:

$$\begin{aligned} R_{ZYZ} &= R_{Z,\alpha}R_{Y,\beta}R_{Z,\gamma} \\ &= \begin{bmatrix} c_\alpha c_\beta c_\varphi - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\alpha c_\beta c_\gamma + c_\alpha c_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix} \end{aligned} \tag{2.1}$$

To obtain the forward kinematic equations the following steps should be done:

a) Obtain the DH parameters.

To describe the kinematics of any robot, four parameters are given for each link $\theta_i, a_i, d_i, \alpha_i$ where two of them described the link, and the others describe the connection with other links. In the case of revolute and prismatic robots variable θ_i and d_i are denoted as joint variable. DH parameter is computed manually or using computer programs such as MATHEMATICA or MATLAB programs. Table (2.1) shows the DH parameters for i-link robot manipulator.

Table (2.1): DH parameter representation

Link	**Joint**	a_i	α_i	d_i	θ_i
1	0-1	a_1	α_1	d_1	θ_1
2	1-2	a_2	α_2	d_2	θ_2
3	--	--	--	--	**--**
4	--	--	--	--	**--**
i	$(i-1)\rightarrow i$	a_i	α_i	d_i	θ_i

b) Obtain link transformation matrices A_i (A matrices).

After obtaining the table of DH convention, a series of homogeneous matrices can be derived depending on the number of the DOF. The transformation matrix for each joint from joint 1 to the joint *i* can be calculated as:

$$A_i = Rot(z,\theta_i)Trans(z,d_i)Trans(x,a_i)Rot(x,\alpha_i) \tag{2.2}$$

or in terms of the full matrices

$$A_i = \begin{bmatrix} c_{\theta i} & -s_{\theta i} & 0 & 0 \\ s_{\theta i} & c_{\theta i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha i} & -s_{\alpha i} & 0 \\ 0 & s_{\alpha i} & c_{\alpha i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

By multiplication, we obtain:

$$A_i = \begin{bmatrix} c_{\theta i} & -s_{\theta i}c_{\alpha i} & s_{\theta i}s_{\alpha i} & a_i c_{\theta i} \\ s_{\theta i} & c_{\theta i}c_{\alpha i} & -c_{\theta i}s_{\alpha i} & a_i s_{\theta i} \\ 0 & s_{\alpha i} & c_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

where a_i is the distance along x_i from o_i to the intersection of x_i and z_{i-1} axes, d_i is the distance along z_{i-1} from o_{i-1} to the intersection of x_i and z_{i-1} axes, α_i is the angle between z_{i-1} and z_i measured about x_i, and θ_i is the angle between x_{i-1} and x_i measured about z_{i-1}. Equation (2.3) shows the symbolic of the i^{th} 4×4 homogenous transformation matrix. The homogeneous matrix houses the position and orientation information of a link frame with respect to adjacent link frame. If we employ equation (2.4) and Table (2.1), we can determine the *A* matrices for each link.

c) Obtain the manipulator transformation matrix H_i^0 (H matrix).

After the homogeneous matrix has been defined for each link of the robot manipulator, simple solution to find the total homogeneous matrix for robot manipulator with *i-links* is accomplished by multiplying all the transformation matrices from A_1 to A_i as follows:

$$H_i^0 = A_1 A_2 .... A_i \tag{2.5}$$

The matrices from to are the transformation matrices from joint 1 to joint i and H_i^0 is the location of the i^{th} coordinate frame with respect to the base coordinate.

d) Calculate the position and orientation of the end-effector.

The general homogeneous matrix for the desired position and orientation of the end-effector that obtained from Table (2.1) as follows:

$$H_i^0 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{32} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} = A_1 A_2 .... A_i \tag{2.6}$$

Equation (2.6) consists of two main components: the rotation matrix and the position vector of the end-effector as follows:

$$R_d = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.7}$$

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.8}$$

The orientation and the position of the end-effector solved directly once the homogeneous matrices for manipulator with *i*-links are multiplied.

The 3×3 rotation matrix provides the orientation of frame *i* with respect to the base frame. The position vector $d = (x, y, z)^T$ represents the desired position from the origin o_0 to the origin o_1 expressed in the frame $o_0x_0y_0z_0$. In the previous equations $c_1 = (r_{11}, r_{21}, r_{31})^T$ is a vector represents the direction of x_i in the $o_0x_0y_0z_0$ system, $c_2 = (r_{12}, r_{22}, r_{32})^T$ is a vector represents the direction of y_i, and $c_3 = (r_{13}, r_{23}, r_{33})^T$ represents the direction of z_i.

Solutions of the Euler angles are given as [5] :

$$\alpha = A\tan 2(r_{33}, \sqrt{1 - r_{33}^2}) \tag{2.9}$$

Or

$$\alpha = A\tan 2(r_{33}, -\sqrt{1 - r_{33}^2}) \tag{2.10}$$

If $s_\alpha > 0$ then:

$$\beta = A\tan 2(r_{13}, r_{23}) \tag{2.11}$$

$$\gamma = A\tan 2(-r_{13}, r_{32}) \tag{2.12}$$

If the value of $s_\alpha < 0$ is chosen, then

$$\beta = A\tan 2(-r_{13}, -r_{23}) \tag{2.13}$$

$$\gamma = A\tan 2(r_{31}, -r_{32}) \tag{2.14}$$

2.2.2 Inverse Kinematic

This section is concerned with the IK problem to find the joint variables of the robot manipulator for a given position and orientation of the end effector [3]. The problem of the inverse kinematics (IK) is more difficult than the forward kinematics problem. It can be mathematically expressed as:

$$\theta_k = f_k(x, y, z, \alpha, \gamma, \phi) \tag{2.15}$$

where $k = 1, \ldots, i$, θ_k joint angles and $(x, y, z, \alpha, \gamma, \phi)$ represents the position and orientation.

There are steps used to solve the inverse kinematics for robot manipulator as follows:

Equate the general transformation matrix to the final transformation matrix of the robot manipulator.

$$H_i^0 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} = A_1 A_2 \ldots A_i \tag{2.16}$$

For the both matrices define:

a) The elements that contain one joint variable.

b) Pairs of elements, which contain only one joint variable.

c) Elements, or combinations of elements, contain more than one joint variable.

After defining these elements, equate it to the corresponding elements in the other matrix to form equations, and then solve these equations to find the values of joint variables.

Repeat step (3) to identify all elements in the two matrices.

In the case of inaccuracy, solutions look for another one.

If there is more joint variable to be found, multiply equation (2.16) by the inverse of A matrix for the specified links.

Repeat steps (2) through (6) until solution to all joint variables have been found.

If there is no solution to the joint variable in terms of an element transformation matrix, it means that the arm cannot achieve the specified position and orientation; the position is outside the robot manipulator workspace.

Calculation of the inverse of A_i matrices and solution of the inverse kinematics for joint variables of the experimental robot are derived in [1].

2.3 Dynamic Modeling

The dynamics is the science of motion that represents the relationship between the joint torques and the robot motion.

Dynamic modeling means deriving equations that explicitly describes the relationship between force and motion. These equations are important to consider in simulation of robot motion, and in design of control algorithms.

2.4 Different Approaches

Computing the dynamics of robot manipulators can be challenging. Researchers have discovered different approaches, where in general there are two methods available; the Euler-Lagrange formulation and the Newton-Euler formulation.

In the standard Euler-Lagrange formulation, the manipulator is treated as a whole, and the system is analyzed based on its kinetic and potential energy. The

Newton-Euler formulation is quite different because each link of the manipulator is treated in turn. First, there is a forward recursion describing its linear and angular motion, then a backward recursion to calculate the forces and torques. Both of these formulations are derived from first principles in [26] and [27], including examples of how the methods can be applied. The resulting dynamic model is the same for both methods.

2.4.1 Dynamic Equation for the N DOF Mechanical Linkage

Using Lagrange-Euler method, the dynamic equations of an N DOF revolute robot manipulator can be written in the following form:

$$M(\theta,\xi)\ddot{\theta}(t)+\bar{D}(\theta,\xi)\hat{V}(\dot{\theta})+G(\theta,\xi)=T(t) \tag{2.17}$$

Where

$\theta(t)=[\theta_1(t)\theta_2(t)...\theta_N(t)]^T$ and $T(t)=[T_1(t)\,T_2(t)...T_N(t)]^T$

$\xi=m_L$ varying payload mass carried by the manipulator

$\bar{D}(\theta,\xi): N\ x\sum_{i=1}^{N} i\ \ matrix\ related\ to\ D(\theta(t),\dot{\theta}(t),\xi)\ vector$

$G(\theta,\xi)$: $N\ x$ 1 matrix related to coriolis and centrifugal velocity.

$G(\theta,\xi)$: $N\ x$ 1 vector of gravitational forces.

All of the matrices $M(\theta,\xi)$, $\bar{D}(\theta,\xi)\ and\ G(\theta,\xi)$ contain nonlinear elements.

The terms $\bar{D}(\theta,\xi)\hat{V}(\theta)\ \ and\ \ G(\theta,\xi)$ and in equation (2.18) can be written as

$\hat{D}(\theta,\dot{\theta},\xi)\dot{\theta}\ and\ \hat{G}(\theta,\xi)\theta\ \ respectively$. Thus, the manipulator dynamics can be rewritten as:

$$M(\theta,\xi)\ddot{\theta}(t)+\hat{D}(\theta,\dot{\theta},\xi)\dot{\theta}+\hat{G}(\theta,\xi)\theta=T(t) \tag{2.18}$$

Where

$\hat{D}(\theta,\dot{\theta},\xi)\theta:\ \ N\ \ x\ \ N\ \ marrix\ \ related\ \ to\ D(\theta(t),\dot{\theta}(t),\xi)\ vector$

$\hat{G}(\theta,\xi)\theta$: $N\ \ x\ \ N$ matrix related to the vector of gravitational forces

2.4.2 Dynamic Model of a three DOF Revolute Robot Manipulator

Using Lagrange-Euler method, the dynamic equations of a three DOF revolute robot manipulator can be written in the following form:

$$M(\theta,\xi)\ddot{\theta}(t)+\bar{D}(\theta,\xi)\hat{V}\left(\dot{\theta}\right)+G(\theta,\xi)=T(t) \tag{2.19}$$

where

$\theta(t)=[\theta_1(t)\ \theta_2(t)\ \theta_3(t)]^T$

$T(t)=[T_1(t)\,T_2(t)\,T_3(t)]^T$

$\xi=m_L=\mathrm{var}\,ying\ payload\ mass\ carried\ by\ manipulator$

$$M(\theta,\xi)=\begin{bmatrix} M_{11} & & \\ & M_{22} & M_{23} \\ & M_{32} & M_{33} \end{bmatrix} \tag{2.20}$$

$$\overline{D}(\theta,\xi)=\begin{bmatrix} 0 & \bar{D}_{12} & \bar{D}_{13} & 0 & 0 & 0 \\ \bar{D}_{12} & 0 & 0 & 0 & \bar{D}_{25} & \bar{D}_{26} \\ \bar{D}_{13} & 0 & 0 & \bar{D}_{34} & 0 & 0 \end{bmatrix} \tag{2.21}$$

$$G(\theta,\xi)=\begin{bmatrix} 0 \\ G_2 \\ G_3 \end{bmatrix} \tag{2.22}$$

Where the nonlinear elements of the matrices as show in [39].

2.4.3 Derivative of the Three DOF Mechanical Link Torque

Equation (2.20) can be differentiated with respect to time to obtain the derivative of the torque as follows:

$$\dot{T}(t)=M(\theta(t),\xi)\dddot{\theta}(t)+\tilde{C}(\theta(t),\dot{\theta}(t),\xi)\ddot{\theta}(t)+\tilde{D}(\theta(t),\dot{\theta}(t),\xi)\dot{\theta}(t) \tag{2.23}$$

Where

$$\tilde{C}(\theta(t),\dot{\theta}(t),\xi)\ddot{\theta}(t)=\dot{M}(\theta(t),\xi)\ddot{\theta}(t)+\bar{D}(\theta(t),\xi)\dot{V}(\dot{\theta}(t))$$

$$\tilde{D}(\theta(t),\dot{\theta}(t),\xi)\dot{\theta}(t)=\dot{\bar{D}}(\theta(t),\xi)V(\dot{\theta}(t))+\dot{G}(\theta(t),\xi)$$

$$\tilde{C}(\theta,\dot{\theta},\xi)=\begin{bmatrix} \tilde{C}_{11} & \tilde{C}_{12} & \tilde{C}_{13} \\ \tilde{C}_{21} & \tilde{C}_{22} & \tilde{C}_{23} \\ \tilde{C}_{31} & \tilde{C}_{32} & 0 \end{bmatrix}$$

$$\tilde{D}(\theta,\dot{\theta},\xi)=\begin{bmatrix} 0 & \tilde{D}_{12} & \tilde{D}_{13} \\ \tilde{D}_{21} & \tilde{D}_{22} & \tilde{D}_{23} \\ \tilde{D}_{31} & \tilde{D}_{32} & \tilde{D}_{33} \end{bmatrix}$$

2.5 Model of the DC motors

The augmented actuators dynamic equation for the three DOF robot manipulator as described in [39] can be written in the following form:

$$\dot{X}(t)=AX(t)+BU(t)+FT(t)+W\dot{T}(t) \tag{2.24}$$

Where

$$\begin{aligned} x(t)&=[x_1\,x_2\,x_3\,x_4\,x_5\,x_6\,x_7\,x_8\,x_9]^T \\ &=[\theta_1\;\dot{\theta}_1\;\ddot{\theta}_1\;\theta_2\;\dot{\theta}_2\;\ddot{\theta}_2\;\theta_3\;\dot{\theta}_3\;\ddot{\theta}_3]^T \end{aligned}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{32} & a_{33} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{98} & a_{99} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & b_3 \end{bmatrix}, F = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ f_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & f_2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & f_3 \end{bmatrix}, W = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ w_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & w_2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & w_3 \end{bmatrix},$$

$$U(t) = \begin{bmatrix} U_1(t) \\ U_2(t) \\ U_3(t) \end{bmatrix}, T(t) = \begin{bmatrix} T_1(t) \\ T_2(t) \\ T_3(t) \end{bmatrix}, \dot{T}(t) = \begin{bmatrix} \dot{T}_1(t) \\ \dot{T}_2(t) \\ \dot{T}_3(t) \end{bmatrix},$$

The non-zero element of A, B, F and W matrices are as follow:

$$a_{32} = -\frac{k_{v1}k_{t1} + B_{v1}R_1}{J_{m1}L_1}, \qquad a_{33} = -\frac{B_{v1}L_1 + J_{m1}R_1}{J_{m1}L_1},$$

$$a_{65} = -\frac{k_{v2}k_{t2} + B_{v2}R_2}{J_{m2}L_2}, \qquad a_{66} = -\frac{B_{v2}L_2 + J_{m2}R_2}{J_{m2}L_2},$$

$$a_{98} = -\frac{k_{v3}k_{t3} + B_{v3}R_3}{J_{m3}L_3}, \qquad a_{99} = -\frac{B_{v3}L_3 + J_{m3}R_3}{J_{m3}L_3},$$

$$b_1 = -\frac{k_{t1}}{J_{m1}L_1N_1}, \qquad b_2 = -\frac{k_{t2}}{J_{m2}L_2N_2}, \qquad b_3 = -\frac{k_{t3}}{J_{m3}L_3N_3}$$

$$f_1 = -\frac{R_1}{J_{m1}L_1N_1^2}, \qquad f_2 = -\frac{R_2}{J_{m2}L_2N_2^2}, \qquad f_3 = -\frac{R_3}{J_{m3}L_3N_3^2}$$

$$W_1 = -\frac{1}{J_{m1}N_1^2}, \qquad W_2 = -\frac{1}{J_{m2}N_2^2}, \qquad W_3 = -\frac{1}{J_{m3}N_3^2}.$$

Table (2.2): Puma 560 characteristics

Parameters	Link Name			
	Trunk	Base	Shoulder	Elbow
Mass m_i (Kg)			2	1.5
Length L_i (m)	0.3	0.5	0.6	0.5
Position of the center of gravity (m)			0.3	0.25
Moment of inertia I^i $(Kgm)^2$ with respect to :				
x-axis at cg^*			0.06	0.03
y-axis at cg^*			0.008	0.0025
z-axis at cg^*		0.04	0.06	0.03
cg^* :center of gravity				

Table (2.3): DC motor parameters

Parameters	Values
Moment of inertia J_{m_i}	1.52 kgm^2
Armature resistance R_i	2.45 Ω
Armature inductance L_i	025 H
Viscous friction constant B_{v_i}	1.5 Nm/rad/s
Back EMF constant K_{v_i}	7.0 V/rad/s
Inverse of gear ratio of Joint:	
Joint 1 N	16
Joint 2 N	18
Joint 3 N	18

2.6 The Dynamic Modeling

Figure (2.2) shows the modeling of robot using SIMULINK. Where the full model are shown in Appendix D.

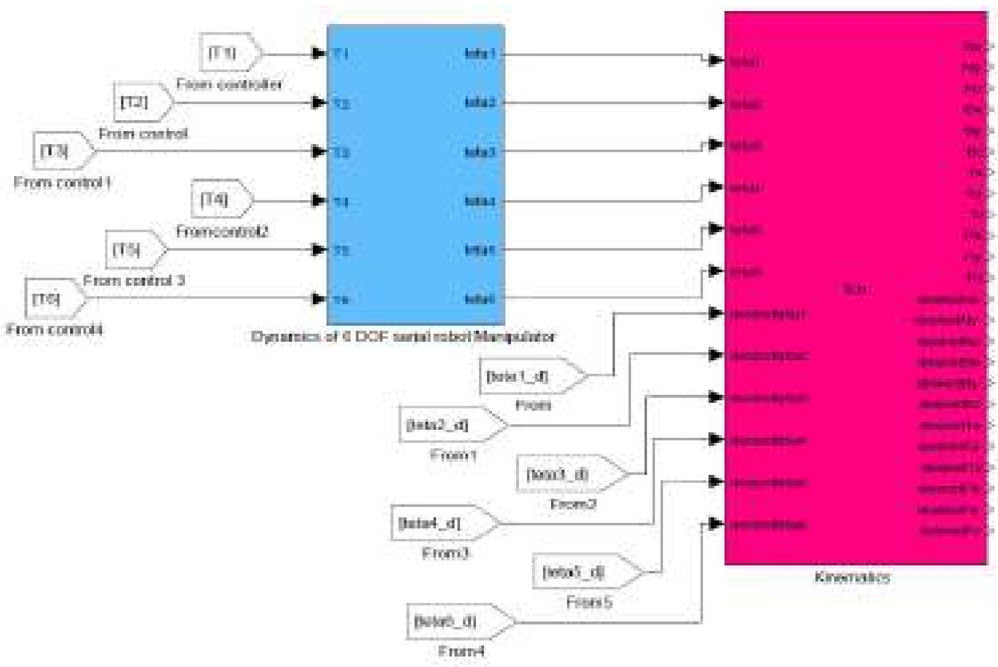


Figure 2.2 Modeling of Robot by SIMULINK

CHAPTER 3 PID AND FUZZY LOGIC CONTROLLERS

3.1 PID Introduction

Conventional PID controllers are characterized with simple structure and simple design procedures. They enable good control performance and are therefore widely applied in industry. However, in a number of cases, such as those when parameter variations take place and/or when disturbances are present, control system based on a fuzzy logic controller (FLC) or Nonlinear controller like SMC may be a better choice.

The PID controller is a universal controller which is used particularly in the field of material processing. Practical controllers are usually assembled with one or more operational amplifiers, whereby the PID behavior is realized by suitable feedbacks.

However, the PID controller has its own limitation; the PID performances can give only satisfactory performance if the requirement is reasonable and the process parameters variation are limited. Several approaches were developed for tuning PID controller such as the Ziegler-Nichols (Z-N) method, the Cohen-Coon (C-C) method, and Nyquist [1].

A PID controller calculates an "error" value as the difference between a measured process variable and a desired set point. The controller attempts to minimize the error by adjusting the process control inputs .

3.2 PID Structure

The PID controller calculation (algorithm) involves three separate constant parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P, I, and D, as show in Figure 3.1

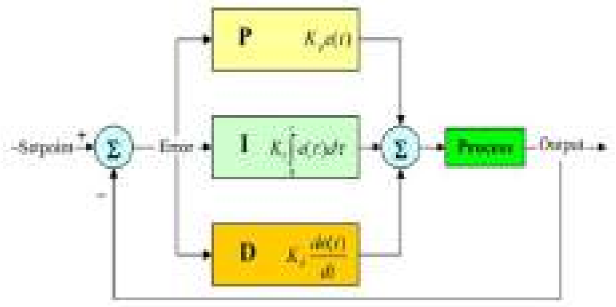


Figure 3.1 PID controller structure

The role of derivative mode is illustrated in Figure 3.2. It can be seen that two different situations are illustrated and one should expect different action from the controller. However, if PI controller is used the control signal will be the same in moment will be proportional to error.

$$u(t) = K_P e(t) \tag{3.1}$$

where $e(t)$, the error signal and $K_P = u(t)/e(t)$ indicates the change of the output signal to the change of the error signal. Integral part of the signal will be proportional to the area under error curve.

$$u(t) = K_P e(t) + K_D \frac{de(t)}{dt} \tag{3.2}$$

If $e(t_1)$ is the same in both cases, and if the area under error curve is the same. overall control signal in both cases will be the same.

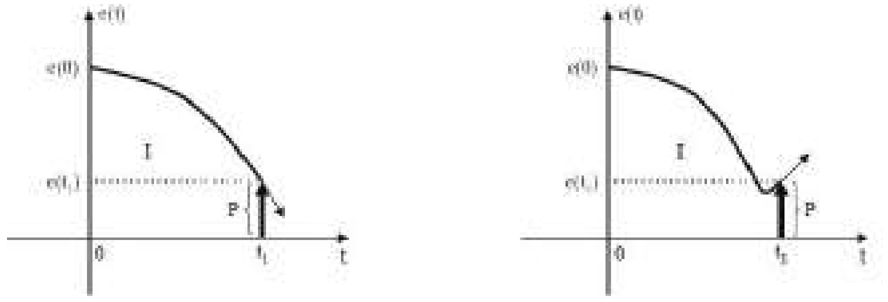


Figure 3.2 PI controller output

When error rapidly decreases. In that case, a role of the controller is to decrease control signal in order to avoid possible control signal overshoot. After a sharp, decrease the error start rising again. In this case, controller has to re-act by increasing control signal in order to decrease the error.

This example shows a need for a controller that will generate control signal that will be also proportional to the error change (error trend). Derivative mode in PID controller fulfills that role. Control signal of PID controller is:

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \tag{3.3}$$

Each term of the three components of PID controller, is amplified by an individual gain, the sum of the three terms is applied as an input to the plant to adjust the process. It will be noted that the purely derivative or integral plus derivative variations never used. In all cases except proportional control, the PID compensator gives at least one pole and one zero.

In real application, PID algorithm can be implemented in different forms depending on the process and control requirements. The easiest form introduced is the ***parallel form***, as shown in Figure 3.3, where the P, I and D elements has the same input signal $e(t)$.

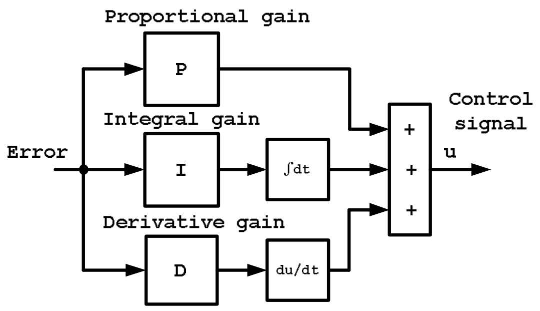


Figure 3.1 PID controller structure

The terms K_P, K_I and K_D stand for the proportional, integral, and derivative gains. The terms $e(t)$ and $u(t)$ represent the error and the control signal respectively.

The transfer function of the PID controller in parallel is:

$$G_{PID\_Paralle}(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s} \qquad (3.4)$$

Equation (3.4) may be rearranged to give the ideal form as follows:

$$G_{PID}(s) = K_P(1 + \frac{1}{T_I s} + T_D \mathrm{s}) \qquad (3.5)$$

where $T_I = K_P / K_I$, and $T_D = K_D / K_P$ are the integral and derivative time constant respectively.

Another type of the PID controller is known as the ***serial form***, and it has the mathematical form:

$$G_{PID\_series}(s) = (K_p + \frac{K_I}{s})(K_D s + 1) \qquad (3.6)$$

The construction of both forms indicates that the setting of the PID controller depends on the used algorithm.

3.3 PID Characteristics Parameters

Proportional action K_P improves the system rising time, and reduces the steady state error. This means the larger proportional gain, the larger control signal become to correct the error. However, the higher value of K_P produces large overshot and the system may be oscillating; therefore, integral action K_I is used to eliminate the steady-state error.

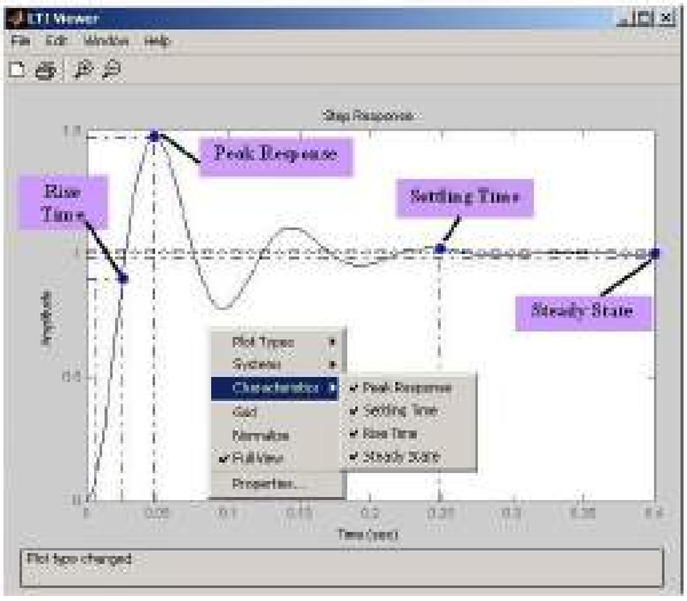
Despite the integral control reduces the steady state error, it may make the transient response worse [1]. Therefore, derivative gain K_D will have the effect of increasing the damping in system, reducing the overshoot, and improving the transient response.

As discussed previously, each one of the three gains of the classical PID control has an effect of the response of the closed loop system. Table (3.1) summarizes the effects of each of PID control parameters. It will be known that any changing of one of the three gains will affect the characteristic of the system response.

Table (3.1): PID characteristic parameters

Closed-Loop Response	**Rise Time**	**Overshoot**	**Settling Time**	**Steady State Error**
Increasing Kp	Fast	Increase	Small / No effect	Decrease
Increasing Ki	Fast	Increase	Increase	Decrease
Increasing Kd	Small / No effect	Decrease	Decrease	Small / No effect

These characteristics are shown graphically in Figure 3.5 for a unit step response. In addition, it will be defined as:



Plot type changed

Figure 3.5 Unit step response curve showing $t_r, t_s, O.S$ and SSE

1. *Maximum overshoot* (*OS*): it is the maximum peak value of response curve measured from unity.

$$\%OS = e^{-(\zeta\pi/\sqrt{1-\zeta^2})} \times 100 \tag{3.7}$$

2. *Rising time* (t_r): the rise time is the time required for the response to rise from 10% to 90% of its final value.

3. *Settling time* (t_s): the settling time is the time required for the response curve to reach and stay with a range about the final value ($\pm 2\%$).

$$T_s = 4/\zeta\omega_n \tag{3.8}$$

4. *Steady state error* (SSE): it expresses the final difference between the process variable and the set point.

3.4 FLC Introduction

Fuzzy logic idea was born in July 1964.Lotfi A. Zadeh [8] found that the traditional system analysis techniques were very precise for complex systems but these traditional techniques took long time to solve the system and very difficult to make it by hardware component. So he proposed different kind of mathematics to solve any system without making complex calculations.

In July 1964 the idea of fuzzy logic was introduced and in 1965 was the birth of fuzzy logic techniques. From 1965 to 1979, fuzzy logic was rules on the papers, but there were not any applications of it. The first use of fuzzy logic was in control systems, in 1979 was the first application of the fuzzy logic in industrial world by Blue Circle Cement and SIRA in Denmark (or Mamdani) [8].

The system was cement kiln controller. This system began to operate in 1982. After that system, Japanese attended to fuzzy logic and started to use it in control systems, and they designed the first automatic subway train controller in 1987. This controller was not the first Japanese applications on fuzzy logic [8]. But in 1985 Japanese starts to make the first general-purpose fuzzy logic controller which converted the logic from mathematics form into practical model. After the subway train, Japanese developed the water-treatment system by using fuzzy logic.

3.5 Fuzzy Logic have many reasons to used

- Fuzzy logic is used to control the complex, and nonlinear systems without making analysis for these systems.
- Fuzzy control enables engineers to implement the control technique by human operators to make ease of describing the systems, because of the man want to invent controller which like him (controller dose the same things which man can do it). Therefore, the best way is transfer the operators, which can the man, due to the controller, and that transfer is easy in fuzzy control.
- Fuzzy logic is flexible with any given systems [9]. If any changes are happening in the system we do not need to start from the first step, but we can add some functions on top of it.
- Fuzzy logic can be blended with conventional technique to simplify their implementation.

3.6 Applications of Fuzzy Logic

The Japanese used fuzzy logic in many of applications, such as (subway train and water-treatment control), but in these years there are many more applications of fuzzy logic. For example in the control field**:**

- Fuzzy logic is used to control the Camcorder to make stabilization in image if there are any rock.
- In washing machine, there is a soft and bad manner clothes, and there are different quantities of laundry. Control of washing cycle is based on these date.
- Robotics controls, Refrigerators for temperature control.
- Engine Control in the modern cars

3.7 FLC Structure

Figure 3.6 shows the basic configuration of MISO fuzzy system, which comprises four main building components: fuzzification method, rule base, inference mechanism, and defuzzification method. As seen in the figure, the input and output data of FLC are crisp (non-fuzzy) values. FLC components are:

1. The fuzzifier: measure the values of input variable and convert the input crisp values into suitable linguistic variables.
2. An expert and skilled operator define the knowledge base. The rule-base holds the knowledge, in the form of a set of rules, of how best to control the system.
3. The inference mechanism evaluates which control rules are relevant for the current time and then decides what the input to the plant should be.
4. The defuzzifier is the opposite operator of fuzzifier interface; it converts the conclusions reached by an inference mechanism into a real value as inputs to plant.

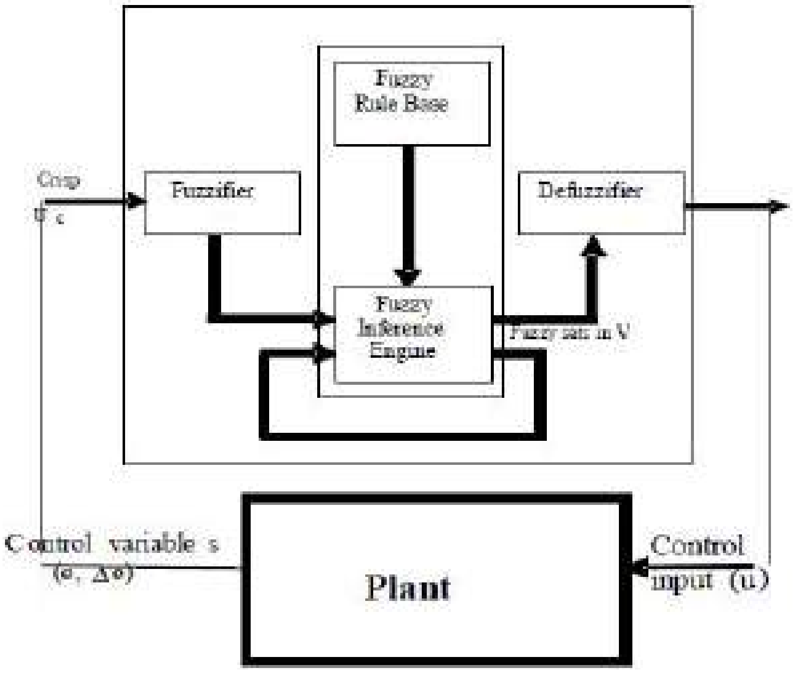


Figure 3.6 Fuzzy control system structure

3.8 Membership Function

Fuzzy set can be represented by a membership function, that gives the degree of membership [1], as shown in Figure 3.8. In that Figure µA(x) in Y –axis, is the symbol that refers to the degree of the membership function and µ (x) can be define as possibility function not probability function [12], and it takes value between (0,1).

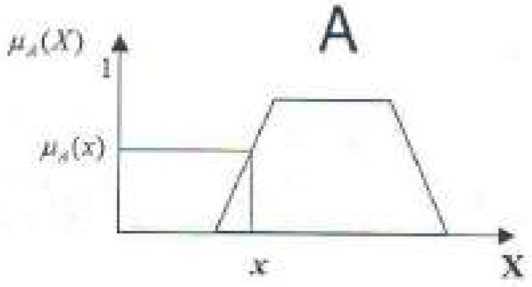


Figure 3.7 Membership Function

So if the probability of (x) = 0.7 that means (x) may be in the set by 70%. But in possibility if the possibility of (x) is 0.7 that means (x) is in the set and has degree 0.7. In the classical sets there is one type of membership function but in fuzzy sets, there are several types of membership functions such as a triangular, trapezoidal, Gaussian, singleton, and the sigmoid membership functions. The most commonly used shapes are triangular, trapezoidal, and singleton. The membership functions are show in Figure 3.8.

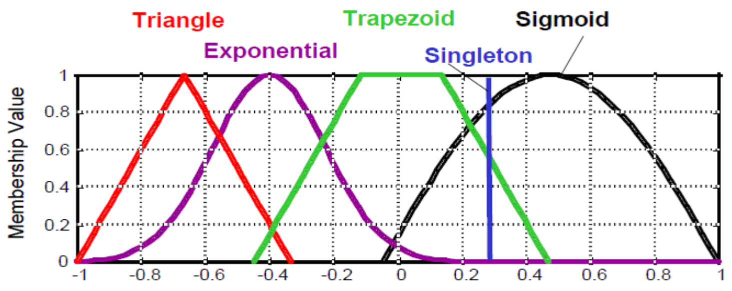


Figure 3.8 Membership functions shapes

3.8.1. Triangular Membership Function

$$\mu \text{ Triangular Membership } (x) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} \quad \cdots\cdots\cdots & b \le x \le c \\ 0 & x > c \end{cases}$$

3.8.2 Trapezoidal Membership Function

$$\mu_{\text{Trapezoidal Membership}}(x) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ 1 \quad \cdots\cdots\cdots & b \le x < c \\ \dfrac{d-x}{d-c} & c \le x < d \\ 0 & x \ge d \end{cases}$$

3.9 Linguistic Variables

Linguistic variable is an important concept in fuzzy logic [8].When fuzzy sets are used to solve the problem without analyzing the system; but the expression of the concepts and the knowledge of it in human communication are needed [8]. Human usually do not use mathematical expression but use the linguistic expression.

For example, if you see heavy box and you want to move it, you will say, "I want strong motor to move this box" we see that, we use strong expression to describe the force that we need to move the box. In fuzzy sets we do the same thing we use linguistic variables to describe the fuzzy sets. These variable are words or sentence in natural or synthetic language. For example if we take the universe U refer to the human age, we can take the interval between (50,60) years and give it name "old man" this name is called linguistic variable. We can use this name to refer to the set that contains the interval (50,60) years.

3.10 IF – THEN Rules

The previous section showed that fuzzy sets can be represented by linguistic variables, and it is an important concept in the fuzzy logic. But another important concept, this concept is fuzzy rules:

IF (input1 is MFa) AND (input2 is MFb) AND…AND (input n is MFn) THEN (output is MFc)

Where MFa, MFb, MFn, and MFc are the linguistic variables of the fuzzy membership functions that are in input 1,input 2…input n, and output. For example, in a system where the inputs of the system are Serves and Food and the output is the Tip. Food may be (good, ok, bad), and serves can be (good, ok, bad), the output tip can be (generous, average, cheap), where good, ok, bad, generous, average, and cheap are the linguistic variables of fuzzy membership function of the inputs (food, and serves) and the output (tip). We can write the rules such as:

$$\underbrace{\text{IF } (\textbf{Food} \text{ is } \textbf{bad}) \text{ and } (\textbf{Serves} \text{ is } \textbf{bad})}_{\text{antecedent}} \quad \textbf{THEN} \quad \underbrace{(\textbf{Tip} \text{ is } \textbf{cheap})}_{\text{consequent}}$$

The maximum number of rules of any system can be found by the next equation (if all inputs have same number of memberships).

max *num* of rules = M^N.

Where (M) is number of the membership function in the input and (N) is the number of the input. The previous example shows the way to write the rules of fuzzy systems. There are two main types of fuzzy inference rules in fuzzy logic reasoning namely, generalized modus pones (GMP) and generalized modus tollens (GMT).

The designer write the rules of fuzzy systems; there are four ways to derivation the fuzzy system rules.

1- Expert Experience and control engineering knowledge.

2- Based on fuzzy modeling of human operators central action.

3- Based on learning

4- Based on fuzzy model of a process.

3.11 Implication Functions

The implication operators can interpret the IF-THEN rules in classical logic. Suppose there is a statement such as "IF **a** THEN b", then the classical set represents this by **a** $\Rightarrow$ **b**. The truth table for this rule can be given as. For the first case a is false so by the rule b will be false too; second case a is false and b is true this is true case because a$\Rightarrow$b not b$\Rightarrow$a so it can happens b is activated when a is false; the third case a is true and by using the rule b must be true, but b is false so it is false statement; finally the fourth case is true because a is true so it implicate b is true.

Table (3.2) Classical Set Truth Table

a	**B**	a$\Rightarrow$ b
F	**F**	**T**
F	**T**	**T**
T	**F**	**F**
T	**T**	**T**

The implication operator can also be written as:

$$\bar{a} \vee b \quad \text{or} \quad (a \wedge b) \vee \bar{a}$$

Each rule in the fuzzy knowledge base corresponds to a fuzzy relation. Various approaches can be taken in determining the relation corresponding to a particular fuzzy rule. The common implication functions are Mini rule (Mamdani), Product rule (Larsen), Max-min rule (Zadeh), Arithmetic rule (Zadeh) and Boolean and others [1].

Common implication operators are:

.

1- Mamdani μR(x, y) = min[μA(x),μB(y)].

2- Zadeh μR(x,y) = max{min[μA(x),μB(y)], 1 - μA(x)}.

3- Larsen μR(x,y) = μA(x) • μB(y).

4- Lukasiewicz μR(x,y) = min{ 1, [1 - μA(x) + μB(y)]}

The fuzzy implication is expressed as a Cartesian product of the antecedent and the consequent as $R_1 = E_1 \times U_1$. In addition, this is similarly done for all rules. but the most widely used in control applications are Mamdani method (minimum-maximum) and Larsen method (product-maximum)

CHAPTER 4 SLIDING MODE CONTROLLER

4.1 Introduction

Variable structure control (VSC) with sliding mode control was proposed by several researchers from the sixties [28]. Sliding mode control use discontinuous feedback control laws to force the system state to reach, and then to remain on, a specified surface within the state space (sliding surface). There are two advantages of obtaining such a motion firstly the system behaves as a system of reduced order with respect to the original plant, and secondly the movement on the sliding surface of the system is insensitive to a particular kind of perturbation and model uncertainties.

Usually sliding mode control is used with nonlinear systems with uncertainty parameters or disturbances, MIMO systems, discrete time models, large-scale and in infinite-dimensional systems [29].

SMC design provides a systematic approach to the problem of maintaining stability and consistent performance in the face of modeling imprecision. Many different techniques have been developed for years to design sliding mode controllers. However, the baselines of all different design techniques are very similar and consist of two main steps:

Design the sliding surface in the state space so that the reduced-order sliding motion.

Determine the control law such that the trajectories of the closed-loop motion are directed towards the sliding surface and tried to be kept on the surface thereafter.

4.2 Nonlinear system

The nonlinear systems can be written in the next form [30]:

$$\dot{x}(t) = f(\mathrm{x},t) + g(t,x)\,\mathrm{u(t)} \tag{4.1}$$

where

x(t) $\in$ IR^{n}, u(t) $\in$ IR^{m}, f(t; x) $\in$ IR$^{n\times n}$, and g(t; x) $\in$ IR$^{n\times m}$

The component of the discontinuous feedback are given by:

$$u_i = \begin{cases} u_i^{+}(t,x) & \text{if } \mathrm{s}_i(\mathrm{x})>0 \\ u_i^{-}(t,x) & \text{if } \mathrm{s}_i(\mathrm{x})<0 \end{cases} \quad \text{for i=1,2,.....m} \tag{4.2}$$

$$\mathrm{s}(x) = [s_1(\mathrm{x}), s_2(\mathrm{x}), s_3(\mathrm{x}), \ldots\ldots\ldots, \mathrm{s}_m(\mathrm{x})]^T = 0 \tag{4.3}$$

where

$\mathrm{s}_i(\mathrm{x}) = 0$ is the i-th sliding surface and is the (n- m)dimensional sliding manifold.

That means the system effectively becomes another system of reduced order, for example, second order system has sliding line, third order system has sliding plane and higher order systems have sliding hyper planes[31]. The control problem consists in developing continuous function u_i^{+}, u_i^{-}, and the sliding surface s(x) = 0. Sliding mode controller design can be divide in two steps:

1- Selecting of stable hyper plane(s) in the state/error space on which motion should be restricted, called the switching function, and
2- Designing of a discontinuous control law which forces the system trajectory to the sliding surface and maintains it there [31].

A trajectory of sliding mode controller starting from a non-zero initial condition, happen in two modes:

- Reaching mode, in which it reaches the sliding surface.
- Sliding mode, in which the trajectory on reaching the sliding surface, remains there for all times and thus evolves according to the dynamics specified by the sliding surface [31].

4.3 Design of switching function

There are many switching functions that can be used in sliding mode control, but all of these functions must have the following properties:

Order of switching function is less than order of plant example: 2nd order system produces first order switching function.

$$s(x) = x_2 + cx_1 = 0 \tag{4.4}$$

Where $c \in R$

Figure 4.1 shows the example of linear sliding function of second order system.

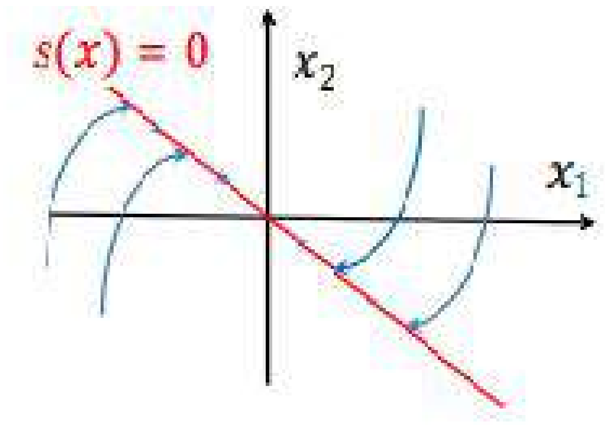


Figure 4.1 Sliding function of the second order example

Sliding mode does not depend on plant dynamics and is determined by parameters of the switching function.

Switching function does not depend on the control law.

Figure 4.2 a and b shows an example of nonlinear and linear switching function respectively.

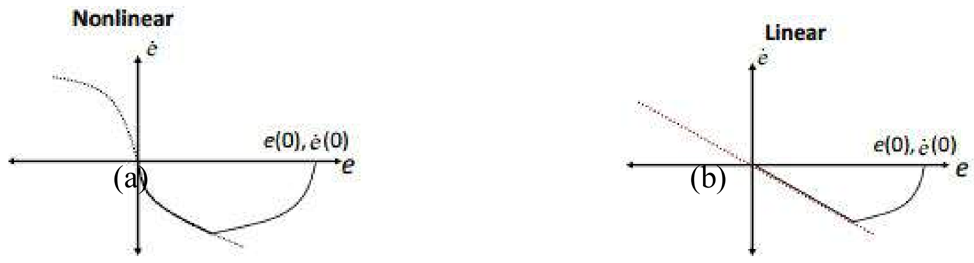


Figure 4.2 a) Nonlinear switching function, b) linear switching function

Nonlinear switching functions have some advantages over linear switching functions.

Appropriate for global dynamic properties of nonlinear systems.

Many design options.

Also, have disadvantages.

Difficult to find nonlinear functions.

Difficult to obtain the surface parameters.

The advantage of linear switching functions over nonlinear switching functions is easy to obtain the surface parameters, and disadvantages are:

May not be appropriate for system dynamics, in general.

Magnitude of the control signal increases directly proportional to the tracking error.

Fewer design options

Although general nonlinear switching surfaces are possible, linear ones are more common in design [3]. Equation 4.5 shows the famous linear switching function which are used in sliding mode control.

$$s(x) = \left(\frac{d}{dt} + c\right)^{n-1} x(t) \text{ ,} c>0, \qquad \text{n is the system order} \tag{4.5}$$

X(t) is states of the system, in this thesis the error and change of error will be the system states.Note that on the surface S (t), the error dynamics are governed by the equation.

$$\left(\frac{d}{dt} + c\right)^{n-1} x(t) = 0 \tag{4.6}$$

Note that on this surface, the error will converge to 0 exponentially. This implies that if there exists a control input u (t) such that x (t) is in S (t) it follows that x (T) is in S (T) for all T > t and the error will converge exponentially to 0 for this control input [30] Figure 4.3 shows an example of this surface. $s(x)=x_2+3x_1=0$.

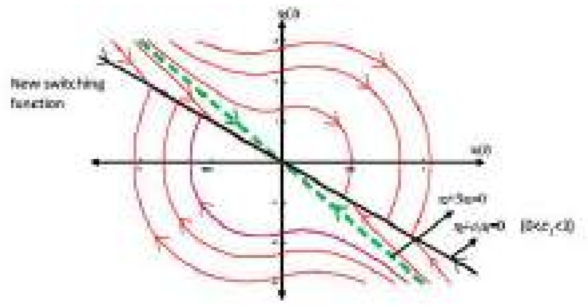


Figure 4.3 Switching function of example s=x_2+3$x_{1=0}$

4.4 Find a control low to reach S(x)

After designing the switching function, which produce the switching surface, the control law will be obtained to achieve a trajectory of sliding mode (reaching mode and sliding mode). To achieve this result the controller must solve the existence problem. Existence of a sliding mode requires stability of the state trajectory to the sliding surface s(x) = 0 at least in a neighborhood of {x: s(x) =0}, i.e., the system state must approach the surface at least asymptotically [32].

The existence problem considered as a generalized stability problem; there are three techniques to solve this problem.

Direct Switching Approach.

Reaching Law Approach.

Lyapunov Function Approach.

In this thesis, the Lyapunov Function Approach will be used. The second method of Lyapunov provides a natural setting for analysis. Specifically, stability to the switching surface requires to choose a generalized Lyapunov function V (t; x) which is positive definite and has a negative time derivative in the region of attraction.

Equation 4.7 shows V(t, x) for switching function:

$$V(x) = \frac{1}{2} s(x)^T s(x) \tag{4.7}$$

This function must satisfy the Lyapunov function conditions for stability:

First must be positive-definite function.

1- V(0) =0.

2- $V(x) > 0 \ \forall x \in U \setminus \{0\}$

Second $\frac{dV}{dt} < 0$

By Derivation of V(x):

$$\dot{V}(x) = \frac{\partial v(x)}{\partial s} . \frac{\partial s}{\partial t} = \frac{\partial v(x)}{\partial s} \dot{s} = s(x)^T \dot{s}(x) < 0 \tag{4.8}$$

$$\dot{s}(x) = \frac{\partial s(x)}{\partial x} \cdot \frac{\partial x}{\partial t} = \frac{\partial s(x)}{\partial x}\dot{x} = \frac{\partial s(x)}{\partial x}[f(x,t) + B(x)u(t)] \quad (4.9)$$

$$\dot{V}(x) = s(x)^T \frac{\partial s(x)}{\partial x}[f(x,t) + B(x)u(t)] < 0 \quad (4.10)$$

Figure 4.4 shows the block diagram of the sliding mode control [33]. As seen in this figure the control signal consist of two parts where u$_{eq}$ is the equivalent control; makes the derivative of the sliding surface equal zero to stay on the sliding surface, and u$_s$ is the corrective control compensate the deviations from the sliding surface to reach the sliding surface.

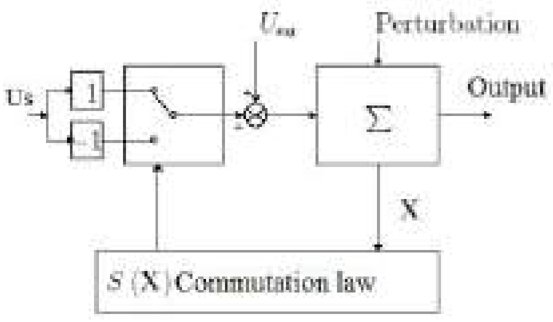


Figure 4.4 Control unit with addition of the equivalent control

$$u(t) = u_s + u_{eq} \quad (4.11)$$

Where $u(t)$ is the total input torque or voltage, u_s is the discontinues or corrective part of sliding mode controller and u_{eq} is the equivalent dynamic part of sliding mode controller.

To satisfy equation 4.11 the designer must choose the input amplitude

sufficiently large to make negative definite for any f(x). The typical choice is sign function. Figure 4.5 shows the signum function

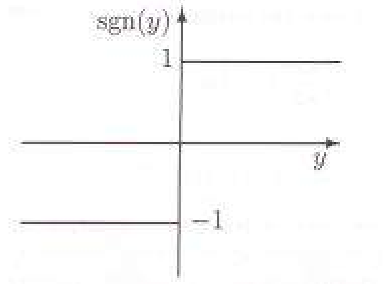


Figure 4.5 Sign function

$$u_s(t) = -M(x)\operatorname{sgn}(s), \operatorname{sgn}(s) = [\operatorname{sgn}(s_i)]$$

$$\operatorname{sgn}(s_i) = \left\{ \begin{matrix} 1, & s_i > 0 \\ -1, & s_i < 0 \end{matrix}, U(x) \in R^{mxm} \right\} \quad (4.12)$$

The sign function is added to the control law because the control law has to be discontinues across s(t) (s=0). u_s is called robust term; generally this term must be less than ueq, if it is higher than ueq the controller will become bang bang controller. u_{eq} must satisfy the derivative of the sliding surface equal zero [34].

$$\dot{s}(x) = \frac{\partial s(x)}{\partial x}.\frac{\partial x}{\partial t} = \frac{\partial s(x)}{\partial x}\dot{x} = \frac{\partial s(x)}{\partial x}[f(x,t) + B(x)u(t)] = 0 \qquad (4.13)$$

$$u_{eq} = -[\frac{\partial s(x)}{\partial x}B(x)]^{-1}\frac{\partial s(x)}{\partial x}f(x,t) \qquad (4.14)$$

For linear system and linear surface.

$$\dot{x} = Ax + Bu(t), \qquad \text{s(x)=sx,S} \in \text{R}^{mxn} \qquad (4.15)$$

$$\begin{aligned} \dot{s}(x) &= \frac{\partial s(x)}{\partial x}.\frac{\partial x}{\partial t} = s[Ax + Bu(t)] = 0 \\ u_{eq}(t) &= -[sB]^{-1}sAx \end{aligned} \qquad (4.16)$$

4.5 Linear system with disturbance

Let equation 4.17 shows the state equations for second order linear system with disturbance.

$$\dot{x} = Ax + bu + hd \qquad (4.17)$$

Where d is nonlinear disturbance. The switching surface is then defined by:

$$s(x) = c_1x_1 + c_2x_2 = p^T x \ \text{ where } p^T = [c_1 \quad c_2] \qquad (4.18)$$

To determine a control law that keeps the system on s(x) = 0, we introduce the Lyapunov function, The following control law design will ensure that $\dot{V}(x) < 0$ for all *t* except when s(x) = 0.

$$\begin{aligned} \dot{V} &= s^T \dot{s} \\ &= s^T p^T \dot{x} \\ &= s(p^T Ax + p^T Bu + p^T Hd) \\ u &= \text{u}_{\text{eq}} + \text{u}_{\text{s}} \end{aligned} \qquad (4.19)$$

To find u_{eq} the derivative of s(x) must equal 0

$$\begin{aligned} & p^T Ax + p^T Bu + p^T Hd = 0 \\ & u_{eq} = \frac{-p^T Ax}{p^T B} - \frac{p^T Hd}{p^T B} \\ & u = u_{eq} + u_s = \frac{-p^T Ax}{p^T B} - \frac{p^T Hd}{p^T B} - Msign(s) \end{aligned} \tag{4.20}$$

4.6 Chattering

Theoretical, the chattering is the trajectories sliding along the switching function. Chattering results in low control accuracy, high heat losses in electrical circuits, high wear of moving mechanical parts. It may also excite unmolded high frequency dynamics, which degrades the performance of the system and may even lead to instability. Figure 4.6 shows the chattering that causes by delay. In ideal sliding mode control the trajectory starts at (s) bigger than zero will heading toward the sliding manifold s = 0, and still sliding on the surface, but in practical the delay between the sign of s changes and the time the control switches makes the trajectory heading toward the region (s) less than zero.

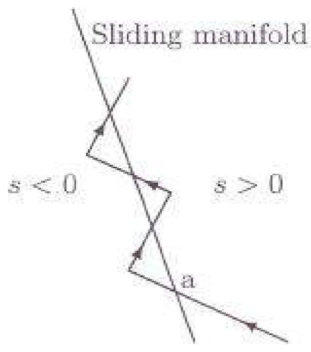


Figure 4.6 Chattering

To solve the chattering problem the sign function can be replaced by saturation function as shown in figure 4.7.

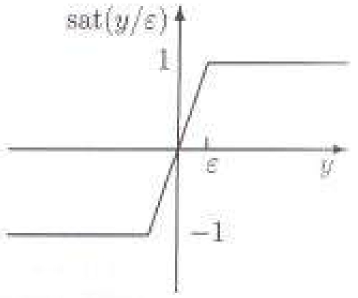


Figure 4.7 Saturation function

4.7 System dynamic during Sliding Mode:

Integral Sliding Mode Control (ISMC) has been successfully applied in a many of control applications. It is different from the conventional SMC design approached, the order of the motion equation in ISMC is less than the order of the original system by the number of dimension of the control input. The method does not require the transformation of the original plant into the canonical form. Moreover, by using this approach, the robustness of the system can be guaranteed throughout the entire response of the system starting from initial time instance [35].

4.7.1 Decomposition the dynamic system

Consider the dynamics of the robot manipulator as an uncertain system described by [35]:

$$\dot{x}(t) = [\overline{A} + \Delta A(t)]x(t) + [\overline{B} + \Delta B(t)]u(t) \tag{4.21}$$

$x(t) \in R^n, U(t) \in R^m$, *represent the state and input vectors, respectively.*

M-file used to compute the value of the elements in matrices **A**, **B**, **ΔA** and **ΔB** is given in Appendix A. These values will be used in simulation later.

Define the state vector of the system as x(t):

$$\mathrm{x(t)} = \begin{bmatrix} x_1(t) & x_2(t) & \dots & x_n(t) \end{bmatrix}^T \tag{4.22}$$

Let a continuous function be the desired state trajectory, where $\mathrm{x}_d(t)$ defined as:

$$x_d(t) = \begin{bmatrix} x_{d1}(t) & x_{d2}(t) & \dots & x_{dn}(t) \end{bmatrix}^T \tag{4.22}$$

Define the tracking error, Z(t) as:

$$z(t) = x(t) - x_d(t) \tag{4.23}$$

In this study, the following assumptions are made:

The state vector x(t) can be fully observed, since all the possible initial states of the system can be observed and the rank is n in this case.

There exist continuous functions H(t) and E(t) such that for all $x(t) \in R^n$ and all t:

$$\Delta A(t) = BH(t); \|H(t)\| \le \alpha \tag{4.24}$$

$$\Delta B(t) = BE(t); \|E(t)\| \le \beta \tag{4.25}$$

- There exist a Lebesgue function $\Omega(t) \in R$ which is integral on bounded interval such that [35]:

$$\dot{x}_d(t) = Ax_d(t) + B\Omega(t) \tag{4.26}$$

where **A** and **B** are the nominal system and the input matrices, respectively[35].

The matrices A, B, are controllable, because system states can be changed by changing the system input.

From the above assumption the control, u(t) which enters the system through B can compensates the uncertainty in the system.

Derivative of equation (4.23) we can obtain an error dynamic system:

$$\dot{z}(t) = \dot{x}(t) - \dot{x}_d(t) \tag{4.27}$$

Hence, the tracking problem is stabilizing the system error, so rewrite equation (4.21)

$$\dot{x}(t) = [\overline{A} + BH(t)]x(t) + [\overline{B} + BE(t)]u(t), \tag{4.28}$$

So the error of the dynamic system from equation 4.28 into 4.27

$$\dot{z}(t) = [A + BH(t)]z(t) + BH(t)x_d(t) + [B + BE(t)]u(t) - B\Omega(t) \tag{4.29}$$

Now to achieved Sliding mode controller design which divide in two steps[4]:

4.7.2 Sliding surface

Proportional-Integral (PI) sliding surface as:

$$s(t) = Cz(t) - \int_0^t [CA + CBK]z(t)dt \tag{4.30}$$

where C and K are constant matrices. The structure of the matrix C is as follows:

$C = diag[c_1 \quad c_2 \quad c_{ni}]$ *where* n_i *is the state* var*iable*, i *is the system input* but the matrix K is designed such that: $\lambda_{\max}$ (A+BK) $\prec$ 0

4.7.3 Control law

Derivative of equation (4.30) give:

$$\dot{s}(t) = C\dot{z}(t) - [CA + CBK]z(t) \tag{4.31}$$

substituting equation 4.29 into 4.31

$$\dot{s}(t) = CAZ(t) + CBH(t)z(t) + CBH(t)x_d(t) + [CB + CBE(t)]u(t) - CB\Omega(t) - CAZ(t) \text{ -CBKZ(t)} \tag{4.32}$$

To guaranty the stability $\dot{S}(t)$=0

Smplification $\{[CB + CBE(t)]^{-1} = [I_n + E(t)]^{-1}\}$

$$\dot{z}(t) = [A + BK]z(t) \tag{4.32}$$

It easy to observe from equation (4.32) that:

The system error dynamics during sliding mode is independent of the system uncertainties.

Couplings between the inputs.

Insensitive to the parameter variations, and can be determined through a proper selection of the desired closed loop poles locations since we assume that both matrices A, B, are controllable.

4.8 Sliding Mode Tracking Controller Design

The system state trajectory x(t) tracks the desired state trajectory $x_d(t)$ as closely as possible for all t even if there uncertainties and non-linearities present in the system. In view of the error space as provide in equation 4.32 the tracking problem has become the problem of stabilizing the error system of equation.

The manifold or sliding surface as show in equation (4.30) is asymptotically stable as show in [35]

So $\dot{V}(t) \prec 0$

Following the Lyapunov stability theory $V(t) = \|s(t)\|$

So $\dot{V}(t) = (s^T(t)\dot{s}(t)) / \|s(t)\|$

The proof is show in details in [35] and Matlab file to get the parameters required below shown in Appendix A and B.

The rate of change of the Lyapunov function:

$$\dot{V}(t) = \dot{V}_1(t) + \dot{V}_2(t) + \dot{V}_3(t)$$

$$\dot{V}_1(t) \le -\{(1+B)\gamma_1 - [\alpha\|CB\| + \|CBK\|\}\|z(t)\| \tag{4.33}$$

$$\dot{V}_2(t) \le -\{(1+B)\gamma_2 - [\alpha\|CB\|\}\|x_d(t)\| \tag{4.34}$$

$$\dot{V}_3(t) \le -\{(1+B)\gamma_3 - [\beta\|CB\|\}\|\Omega(t)\| \tag{4.35}$$

Where α and β is the norm bound of continuous function H and E respectively.

4.8.1 Integral Sliding Mode Controller Simulation on A PUMA like Robot Manipulator

Run the M-file in Appendix B, Based on Table (2.2) and (2.3).

The Matlab Results show:

Transfer function of the DC of the Robot.

Nominal Value of Matrixes A and B.

The bounds of $\Delta A(t) = BH(t); \|H(t)\| \le \alpha$

$\Delta B(t) = BE(t); \|E(t)\| \le \beta$

The gain K and eigenvalues $\lambda_1, \lambda_2 and\, \lambda_3$.

Finally the controller parameters γ_i .

The result value of $\gamma_1 = 33.9958$, $\gamma_2 = 3.7909$ and $\gamma_3 = 0.4670$.

4.8.2 The Effect of the Value of Controller Parameter, γ_i

The effect of the controller parameter is studied in this section. Two different cases of satisfying and unsatisfying where values used are tabulated in Table 4.1.

Table(4.1): Cases of controller Parameters

γ_i	**Satisfied Case**	**Unsatisfied Case**
γ_1	450	15
γ_2	65	2.8
γ_3	10	.01

The simulation was then carried out on both cases with the Puma like robot manipulator at no load (0 kg) and full load (20 kg).

4.8.3 The cycloidal trajectory

Suppose that the trajectory flow the cycloidal function:

$$\theta_{di}(t) = \{\theta_i(0) + \frac{\Delta_i}{2\pi}[\frac{2\pi t}{\tau} - \sin(\frac{2\pi t}{\tau})], 0 \le t \le \tau \qquad (4.36)$$

$$\theta_i(\tau), \tau \le t$$

Where $\tau = 2$ sec,

with initial position of $[\theta_1, \theta_2, \theta_2]^T = [-0.8, -1.5, -0.5]^T$ radians and the desired position of $[\theta_1(\tau), \theta_2(\tau), \theta_3(\tau)]^T = [1.5, -0.2, 1.7]^T$ as show in figures (4.8)

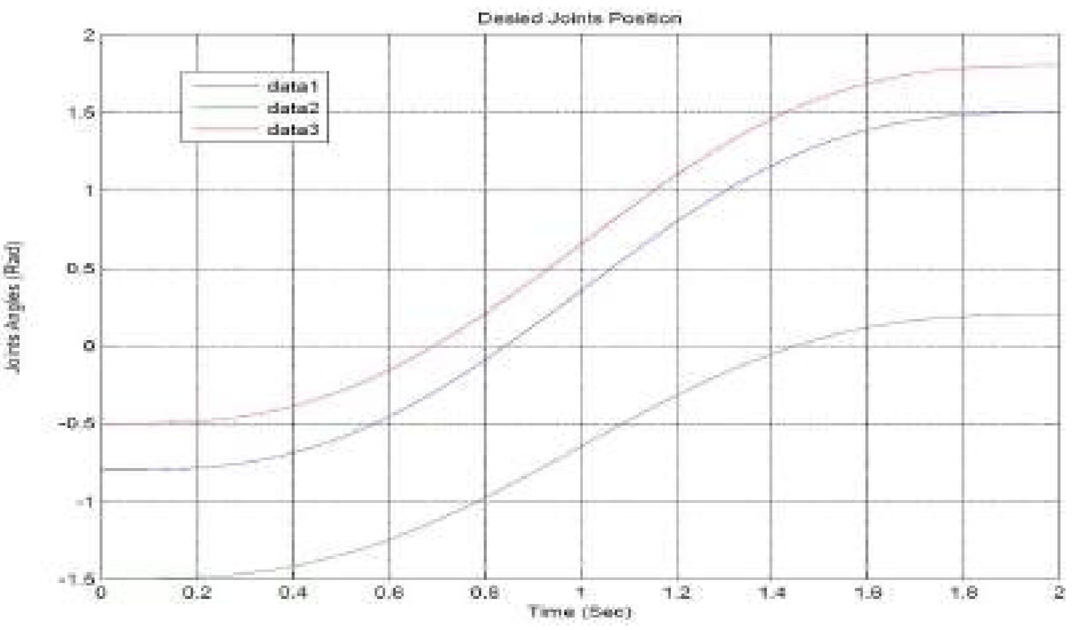


Figure 4.8 Desired Joints Position

4.8.3 Experiment with unsatisfied parameters

First experiment with unsatisfied parameters in table (4.1) the simulation for the first three joint following these steps:

The trajectory flow the cycloidal function as show in figure (4.8)

Initial position of $[\theta_1, \theta_2, \theta_2]^T = [-0.8, -1.5, -0.5]^T$ radians and the desired position of $[\theta_1(\tau), \theta_2(\tau), \theta_3(\tau)]^T = [1.5, -0.2, 1.7]^T$

Finally the simulation will show :

A Sliding surfaces for the three joints (unsatisfied and satisfied).

Control input in both case, and

The trajectory in case of unsatisfied .

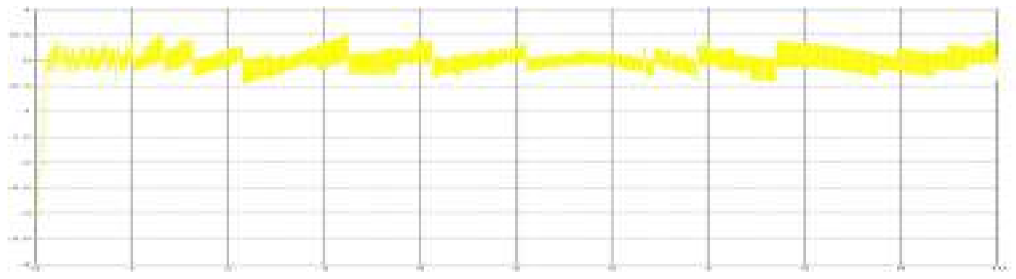


Figure 4.9 Joint 1 Sliding Surface (Unsatisfied)

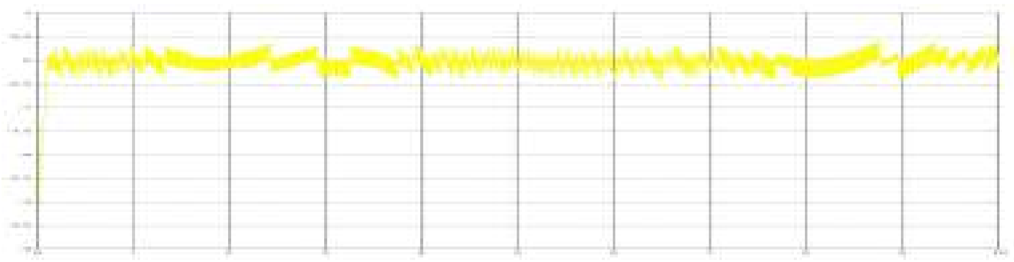


Figure 4.10 Joint 2 Sliding Surface (Unsatisfied)

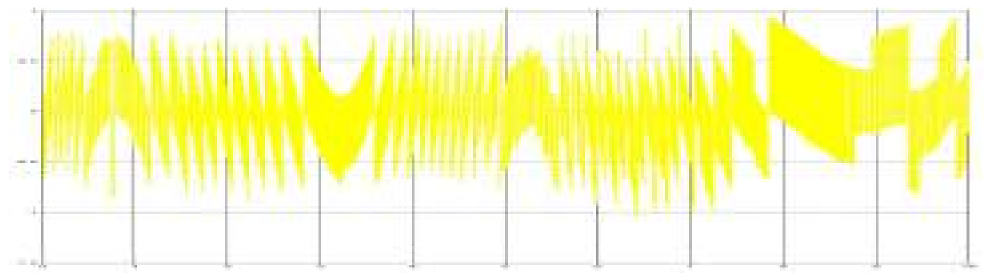


Figure 4.11 Joint 3 Sliding Surface (Unsatisfied)

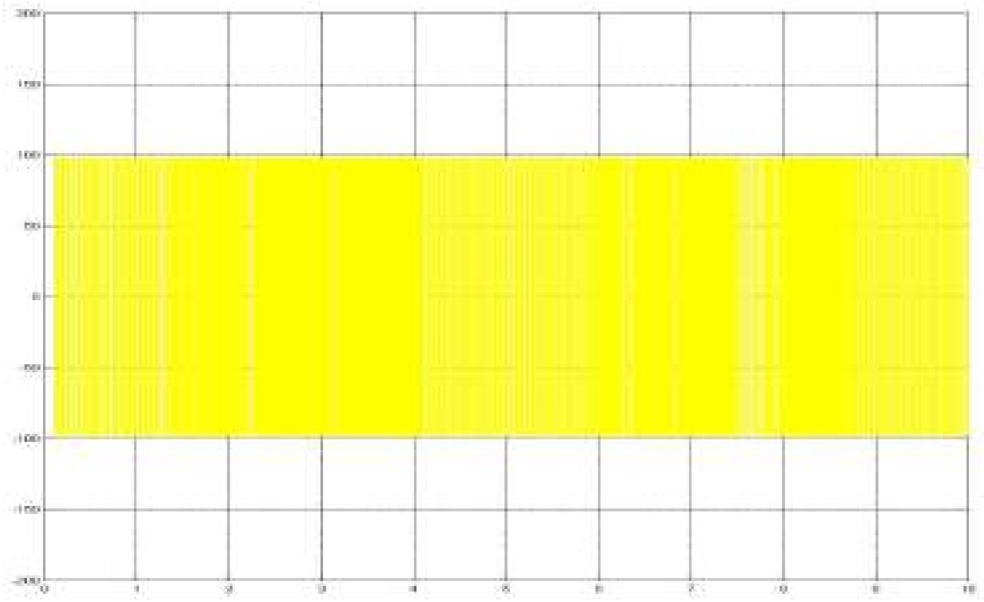


Figure 4.12 Control input (Unsatisfied)

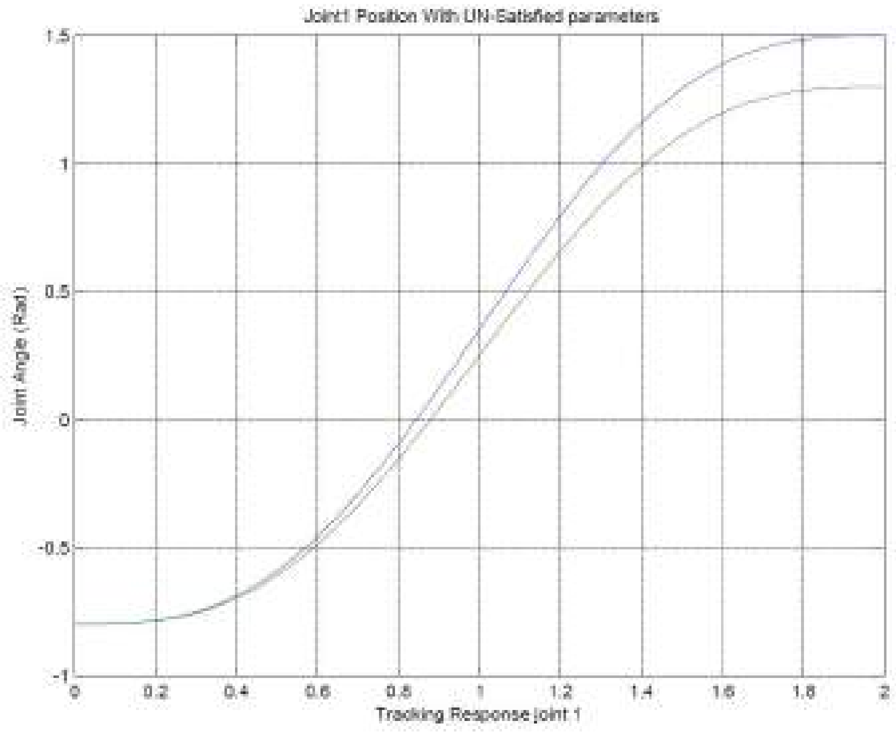


Figure 4.13 Tracking Response joint1

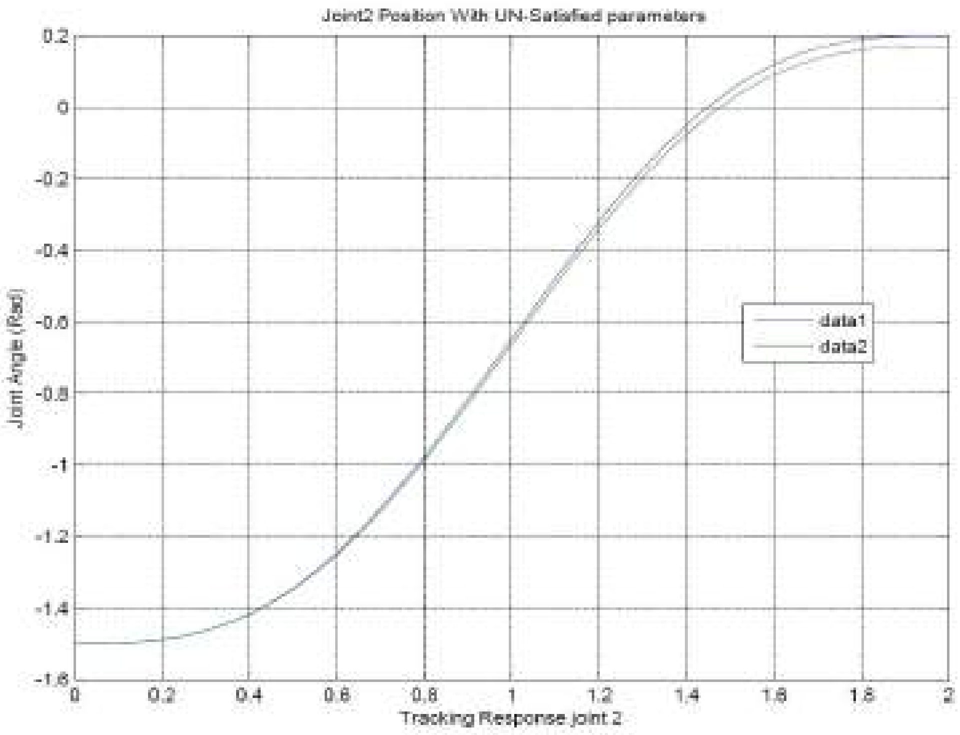


Figure 4.14 Tracking Response joint2

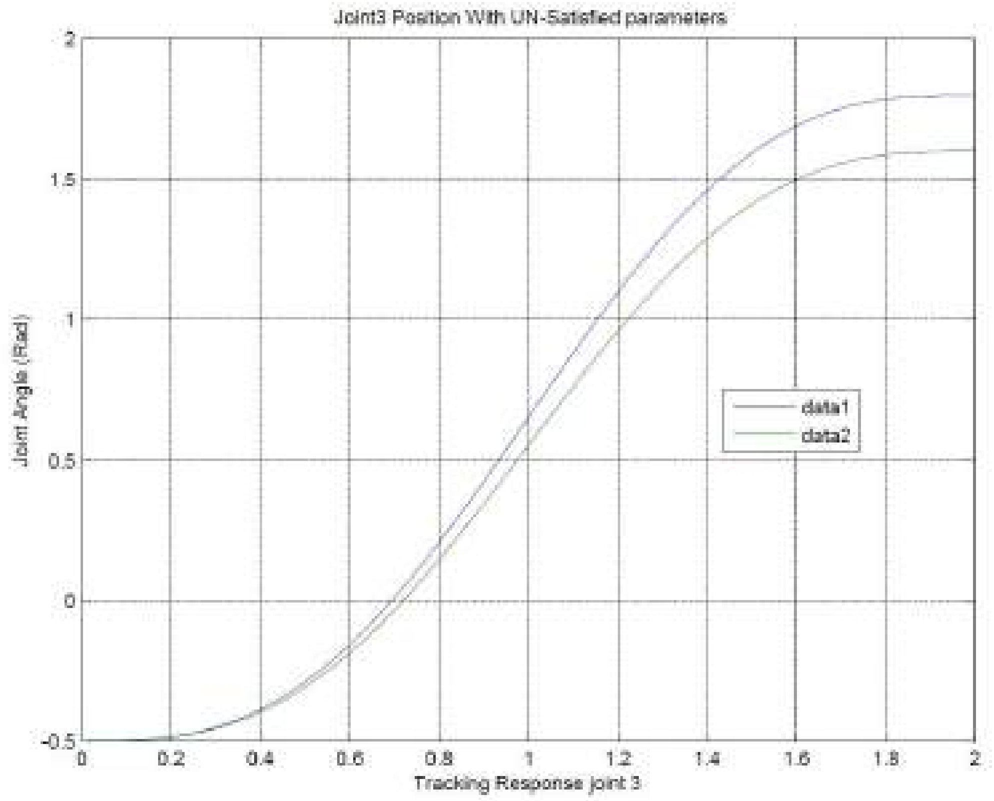


Figure 4.15 Tracking Response joint3

4.8.4 Simulation Result of an experiment with unsatisfied parameters

The simulation results for unsatisfied cases are shown in Figures 4.12 to 4.18

The sliding surfaces of joint 1, joint 2 and joint 3 are oscillated with large deviations as shown in Figure 4.9, 4.10 and 4.11.

The discontinuous Control law cannot force the system trajectory to the sliding surface and kept on.

Control inputs in each of sub-system switch fast Figure 4.12

Controller fails to track the desired positions if the controller parameters conditions are unsatisfied for joint 1, joint 2 and joint 3 as shown in Figure 4.13, 4.14 and 4.15 respectively.

4.8.5 Experiment with satisfied parameters and no load

Second experiment with satisfied parameters in table (4.1) the simulation will carried with no load for the first three joint following these steps:

The trajectory flow the cycloidal function as show in figure (4.8)

Initial position of $[\theta_1, \theta_2, \theta_2]^T = [-0.8, -1.5, -0.5]^T$ radians and the desired position of $[\theta_1(\tau), \theta_2(\tau), \theta_3(\tau)]^T = [1.5, -0.2, 1.7]^T$

Finally the simulation will show :

A Sliding surfaces for the three joints (satisfied no load).

Control input in both case, and

The trajectory in case of unsatisfied .

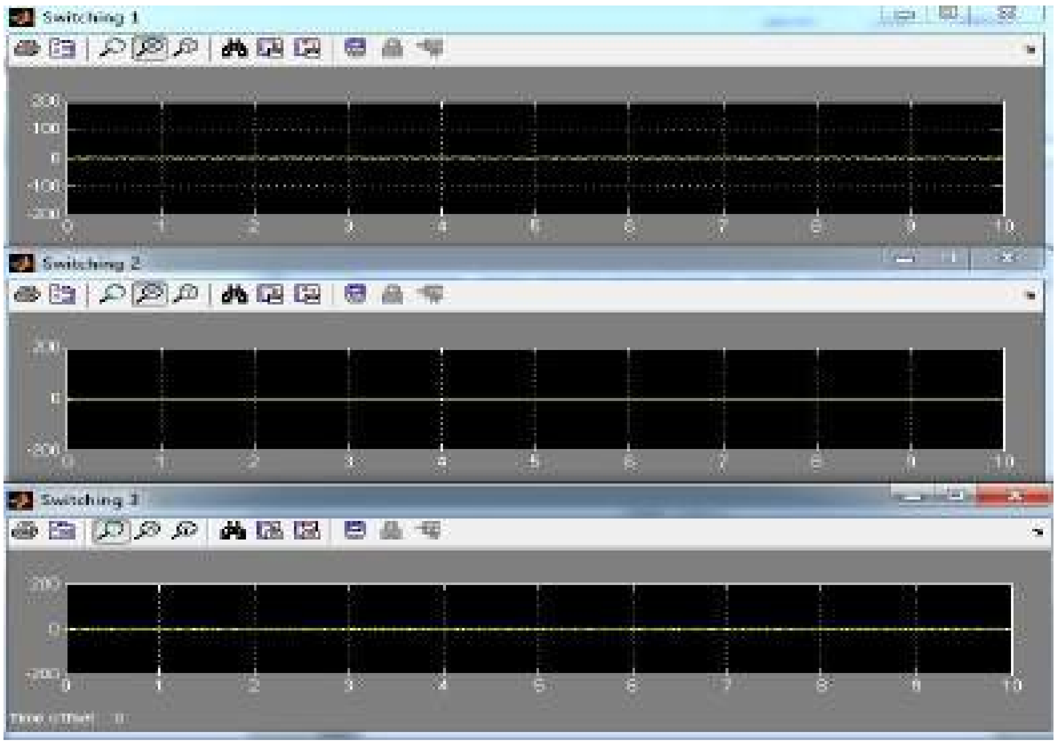


Figure 4.16 Sliding Surfaces of Joints (1, 2 and 3) (Satisfied-No Load)

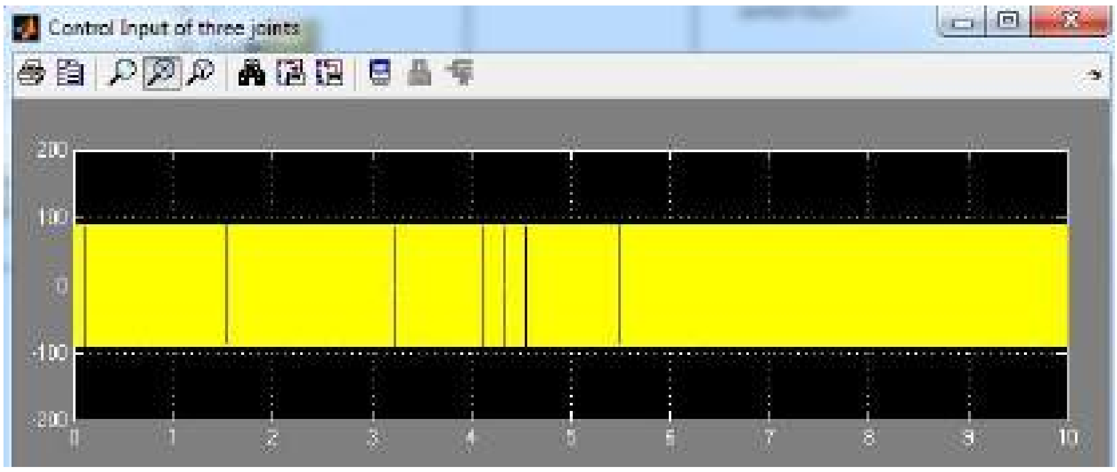


Figure 4.17 Joints Control Inputs No load (Satisfied Parameters)

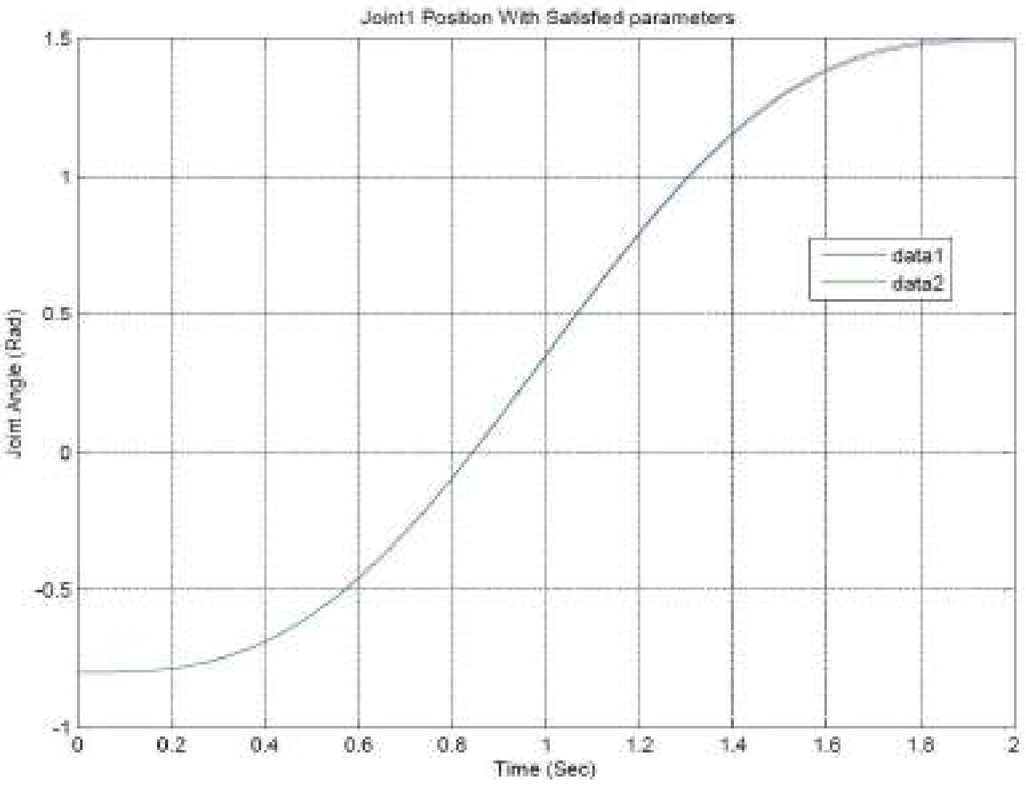


Figure 4.18 Joint 1 Tracking Response (Satisfied Parameters)

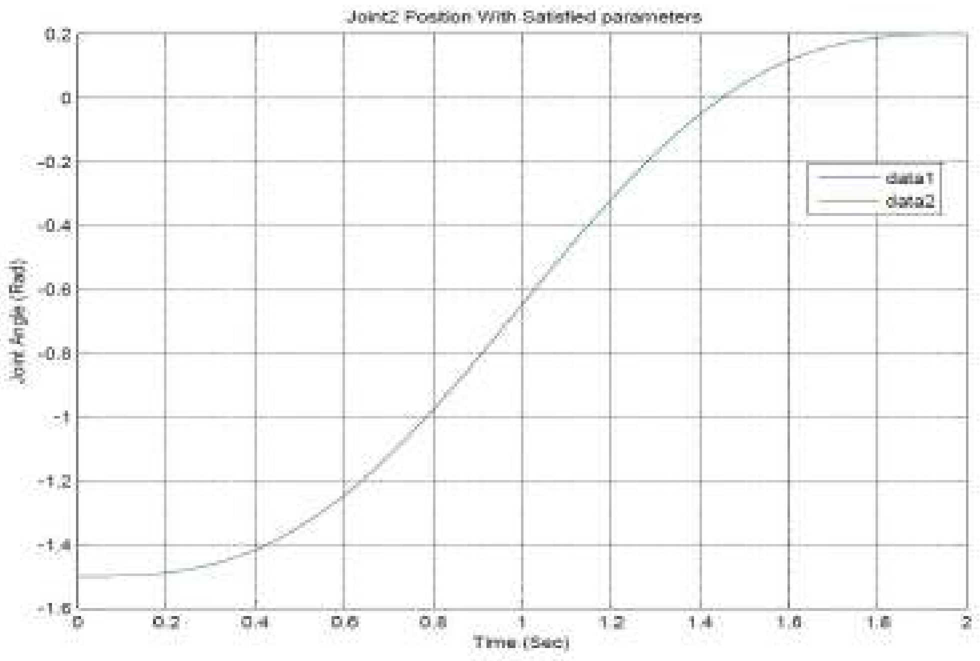


Figure 4.19 Joint 2 Tracking Response (Satisfied Parameters)

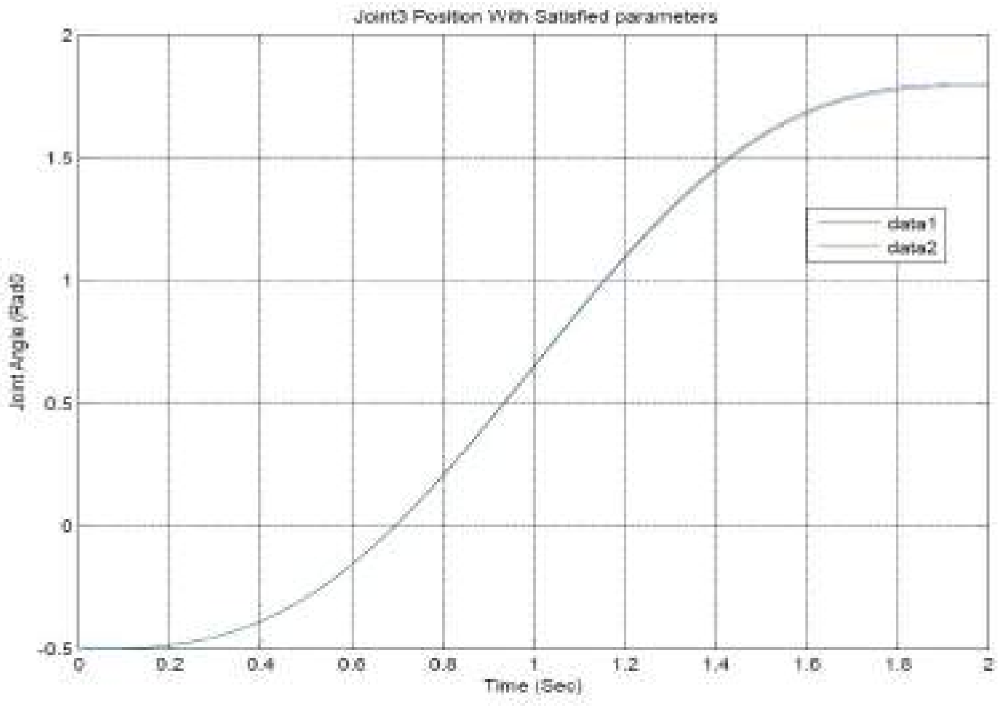


Figure 4.20 Joint 3 Tracking Response (Satisfied Parameters)

4.8.6 Simulation Result of an experiment with satisfied parameters with No load

The simulation results for satisfied cases are shown in Figures 4.16 to 4.20.

The sliding surfaces of joint 1, joint 2 and joint 3 are oscillated with little deviations (almost zero) as shown in Figure 4.16.

The discontinuous Control law forces the system trajectory to the sliding surface and kept on.

Control inputs in each of sub-system switch fast Figure 4.17

The tracking performance is good for all joints as show in Figures 4.18,4.19 and 4.20

4.8.7 Simulation Result of an experiment with satisfied parameters and max. load

The simulation was done again with the manipulator local set to the maximum load :

The tracking performance is good for all joints as show in Figures 4.18, 4.19 and 4.20, the results are satisfactory.

The sliding surfaces of joint 1, joint 2 and joint 3 are oscillated with small deviations shown in Figure 4.25. While the tracking performance is still good for all joints.

In Case of maximum load, the Control inputs in each of sub-system switch fast and continue especially in joint 2 and 3 because of gravity force.

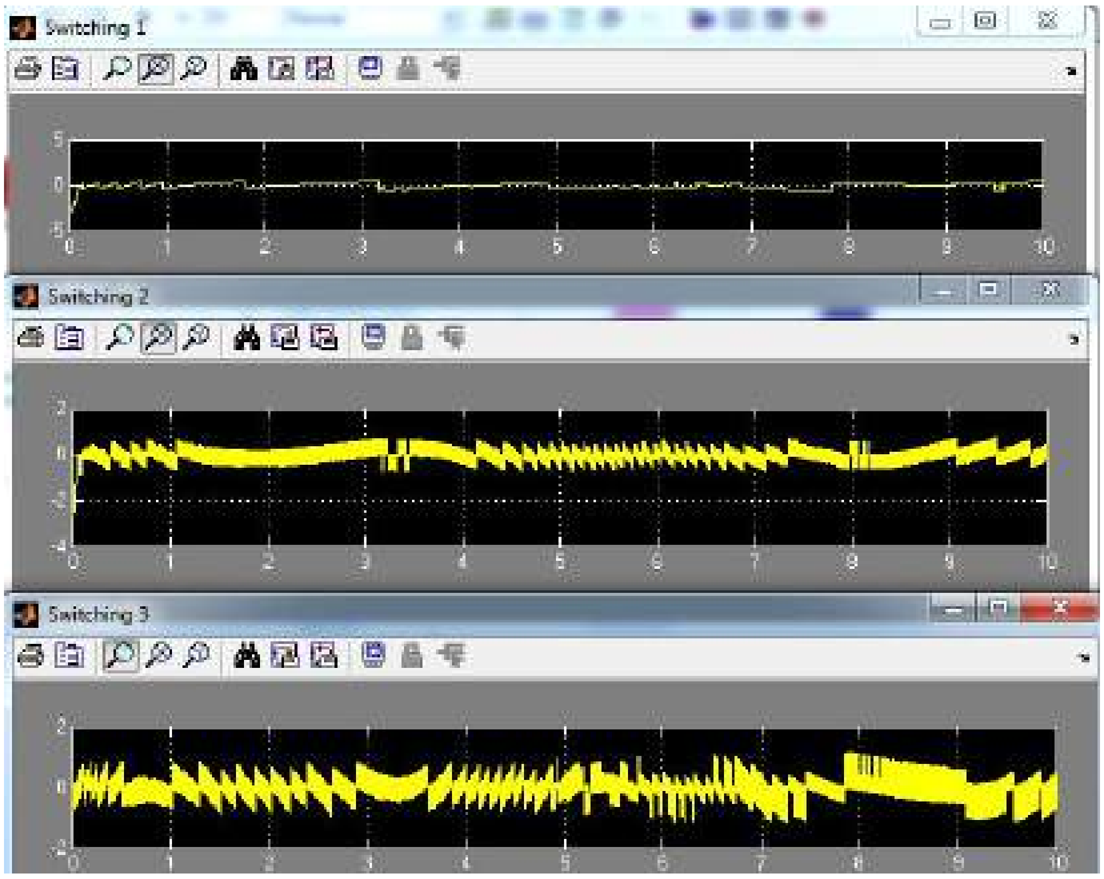


Figure 4.25 Sliding Surfaces of Joints (1, 2 and 3) with load (Satisfied Parameters)

4.8.8 Effect of Load Variation

In the simulations, the manipulator carry a varied load, maximum is 20 kg. The graphs show that the controller is capable to forcing the robot manipulator to track the desired trajectories under different loading conditions with small tracking errors especially for joint 1. The tracking error for joints 2 and 3 are large compare to joint 1 for the 20 Kg. load in particular due to the gravitational forces acting on joints 2 and 3.

4.8.9 Experiments Conclusion

The tracking performance is good for all joints in satisfied parameters case, but not good in case of not satisfied.

The proposed controller make the system:

Insensitive to parameter variations, uncertainties and couplings.

Guarantee stability based on Lyapunov theory.

CHAPTER 5 SIMULATION AND RESULT

5.1 Introduction

In this chapter, the design description of GUI of Puma robot will be shown then; Implementation of the proposed controllers for the robot arm. The results will be discussed. MATLAB and SIMULINK are used to simulate and evaluate the performance of the proposed controllers that applied on the robot. The control algorithms of SMC, SMFC, and ISMC have implemented to control the Puma robot. The purpose of the controllers are good path trajectory with minimum error and free chattering.

5.1 PUMA 560

The PUMA (Programmable Universal Machine for Assembly, or Programmable Universal Manipulation Arm) is an industrial robot arm.

The Puma 560 is a six degree of freedom robot manipulator. The end-effector of the robot arm can reach a point within its workspace from any direction. Six brushed DC servomotors control the six degrees of freedom. The robot can determine its global position from the given feedback information.

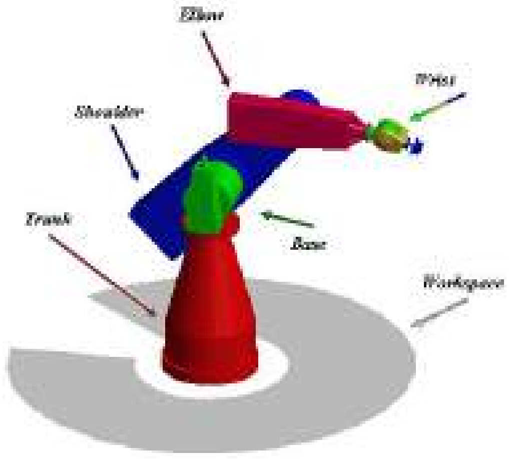


Figure 5.1 PUMA 560

5.3 Design Description

The first step in this process is to design the arm using AutoCAD software. The program chosen for this was Autodesk Inventor. Inventor allows the arm to be designed and visualized at the same time. It also allows the arm to be checked for possible collisions and link interference. Because each link depends upon the previous link, the design of the arm needs to begin from the Base and finish at the End-Effector. Link 1 is therefore the first to be designed, followed by link 2, and so on. This means that the design process is involved, as each link has to redesign several times.

Trunk

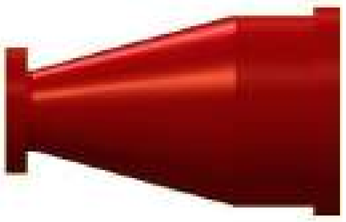


Figure 5.2 Trunk

Base

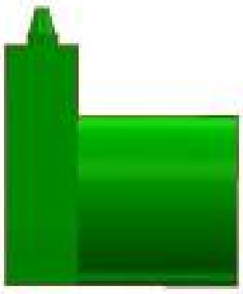


Figure 5.3 Base

Shoulder

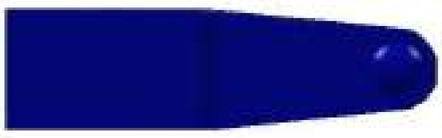


Figure 5.4 Shoulder

Elbow

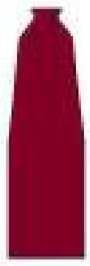


Figure 5.5 Elbow

Spherical wrist

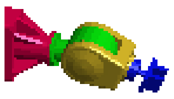


Figure 5.6 Spherical wrist

The axes of rotation of spherical wrist are denoted Yaw, Pitch, and Roll and intersect at a point called the wrist center point.

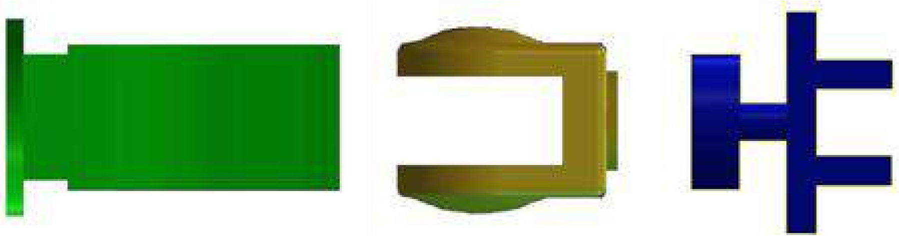


Figure 5.7 spherical wrists

Now and after we finished the graphics design, it must be connected to Matlab since we use the CAD2Matlab function, where this function takes a CAD file in (.stl or .slp format) and convert it to Matlab.

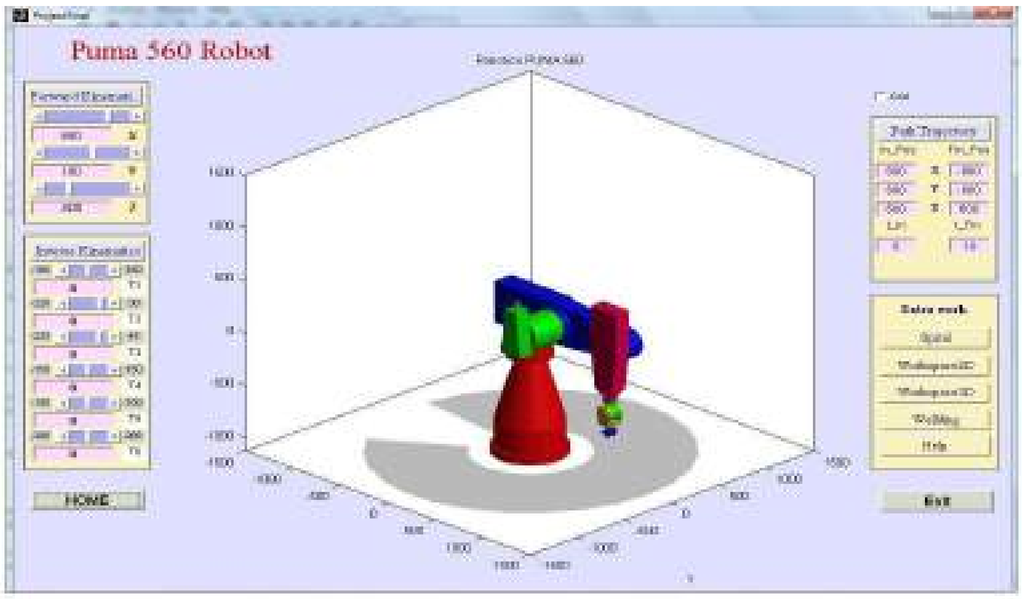


Figure 5.9 PUMA GUI

5.4 GUI User Manual

Open the file, which contain programs and run the Projectfinal.m. Or you can set the MATLAB path to the file enclosing all the programs and type "Projectfinal" at the command window, then follow the steps below:

If it is the first time to run it or you have cleared the work space of MATLAB, you should see the buttons Forward Kinematics, Inverse Kinematics, path trajectory, extra work, Home, and exit button.

Ok. Supposing everything goes well, and you see the figure. You can easily set each variable and the robot will be demonstrated every time you do the change.

The problem will become straightforward when you try to run the program.

Forward Kinematics button, by entering the angles values the robot will animate and get the end effector position and orientation.

Inverse Kinematics button, there are three coordinates X, Y, and Z. Which refer to the end effector position, then angles will be calculate.

For your convenience, the sliders are added to show what the result after modifying the bases. Of course, if you know exactly the coordinate of each base, what you should do is just fill them in the blank, which blew each slider.

Path trajectory the user will specify the initial position and the final position in the three coordinates x, y, and z, and select the initial time and final time.

There are some extra work in the fourth button as spiral, workspace2D, workspace3D, and welding.

Ok. Now do some final regulations for the robot, in home button actually, you can come back to the initialized status at any time. "In this case robot goes to home position".

The controllers button are used to selecting the controller, there are three controllers are used, as will show in the advance part of GUI.

5.5 Forward Kinematics

Enter the values of angles from θ_1 to θ_6 or move them by sliders and click the Forward Kinematics button the robot will be animate and the end effector will go to the desired position.

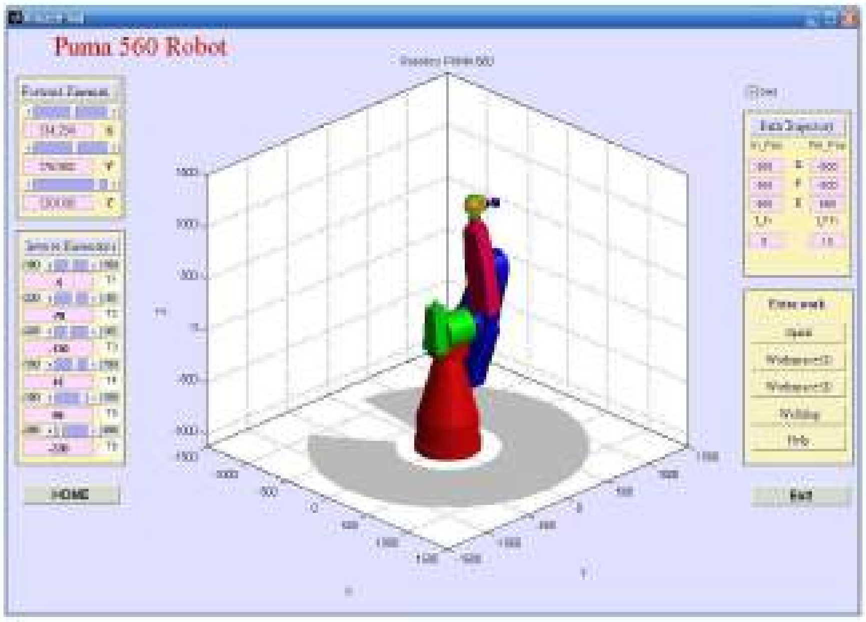


Figure 5.10 Forward Kinematics

5.6 Singularity

If the user enter the input data that out of the range the robot cant complete its mission, the desired position is out of work space rang.

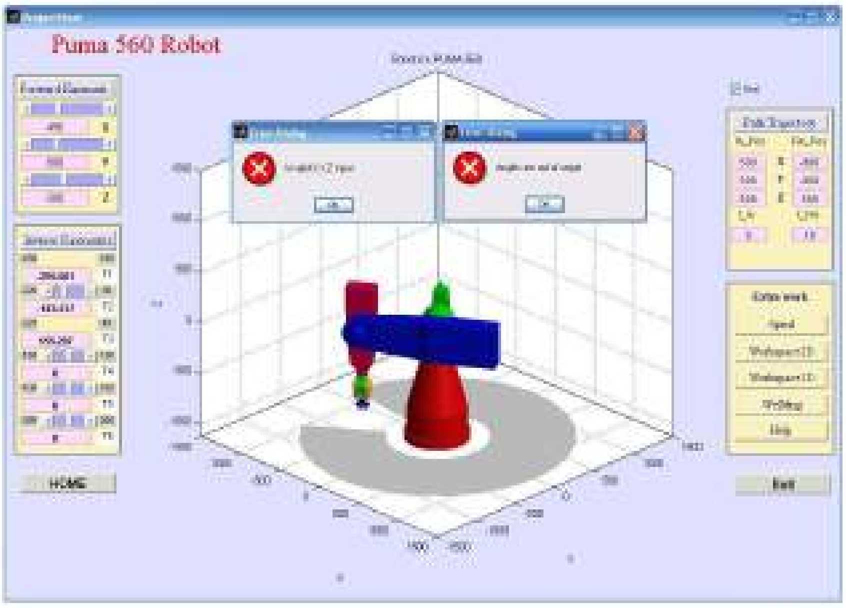


Figure 5.11 Singularity

5.7 Path Trajectory

In path trajectory the user enter the initial position and final position with initial and final time, and click the path trajectory button, the robot will go to desired position automatically, and appear tow figures the first for path tracking angle, velocity, and acceleration, and the second fig for the 6 angles

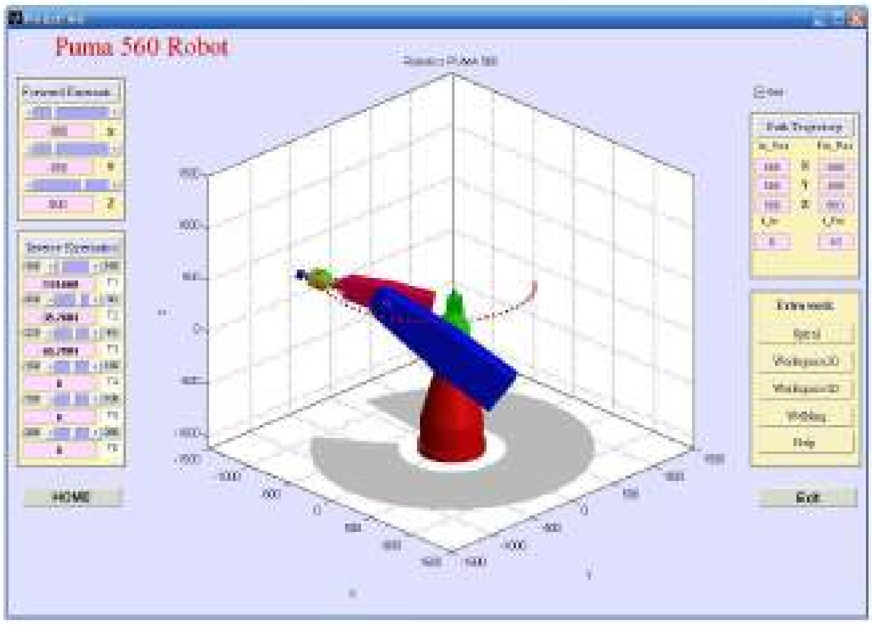


Figure 5.12 Path Trajectory

5.8 Inverse Kinematics

Enter the values of the desired position then the angles from θ_1 to θ_6 will be calculated and robot will move.

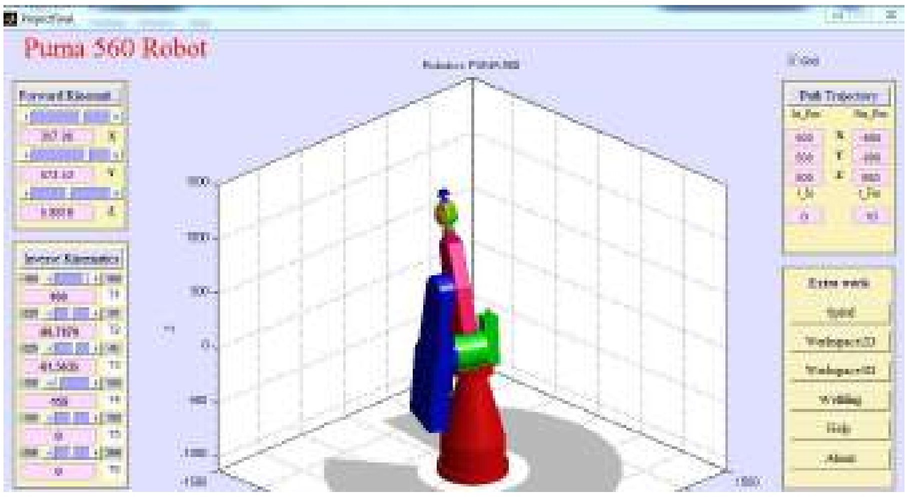


Figure 5.13 Inverse Kinematics

Some extra work the user can test them as spiral, workspace2D, workspace3D, and welding.

1) Spiral

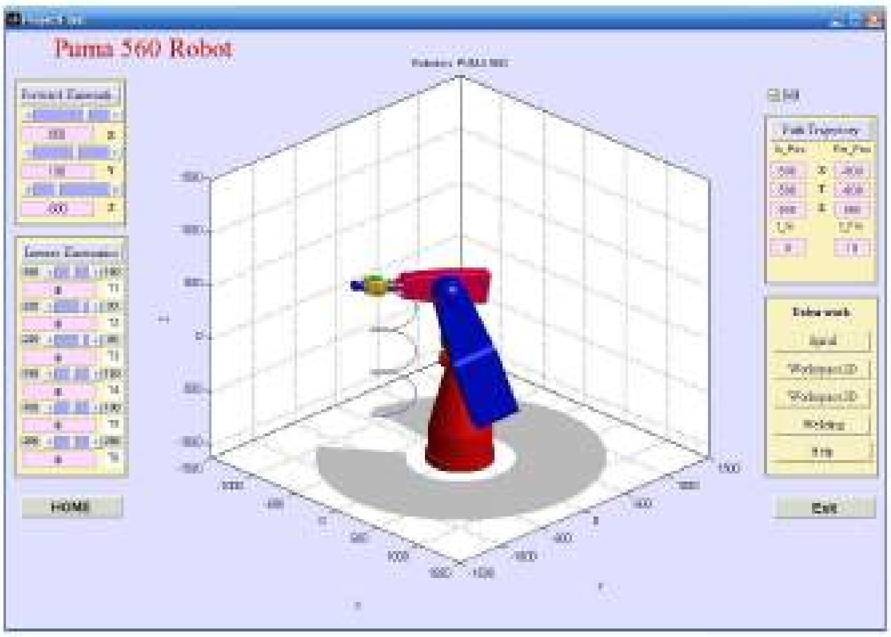


Figure 5.14 Extra Work

2) Workspace2D

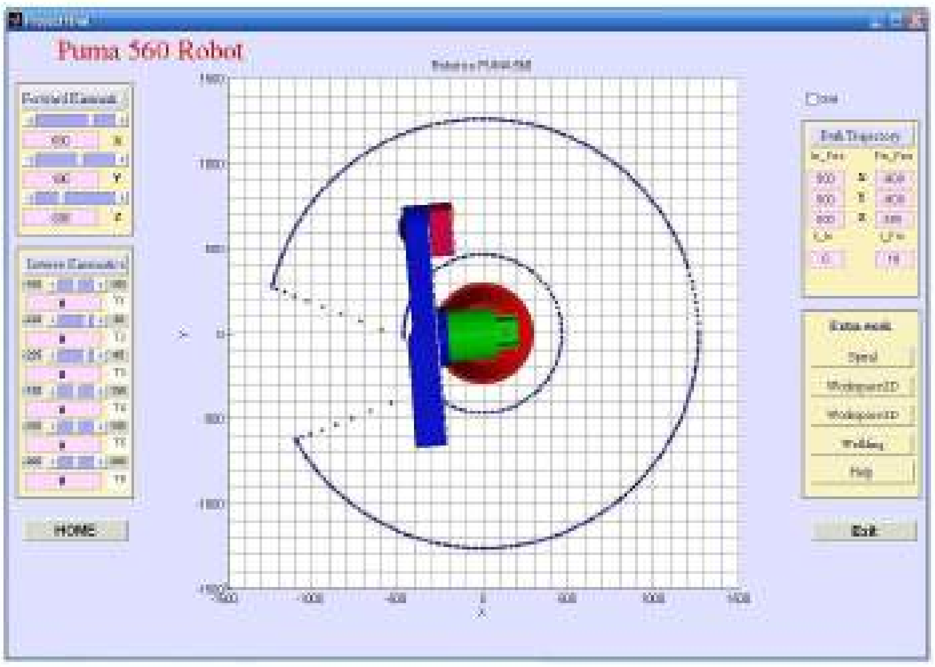


Figure 5.15 Work Space 2D

3) Workspace3D

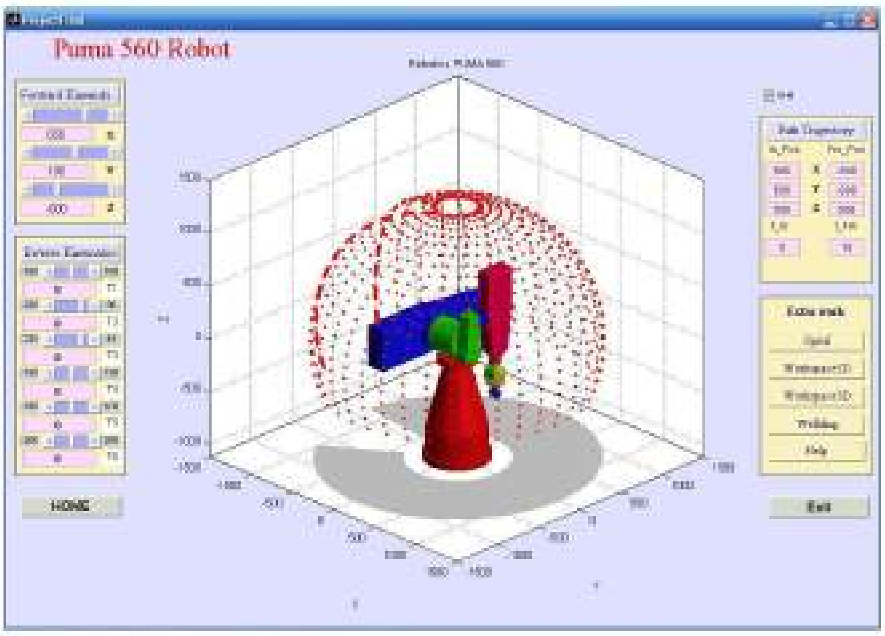


Figure 5.16 Work Space 3D

4) Welding
Before welding

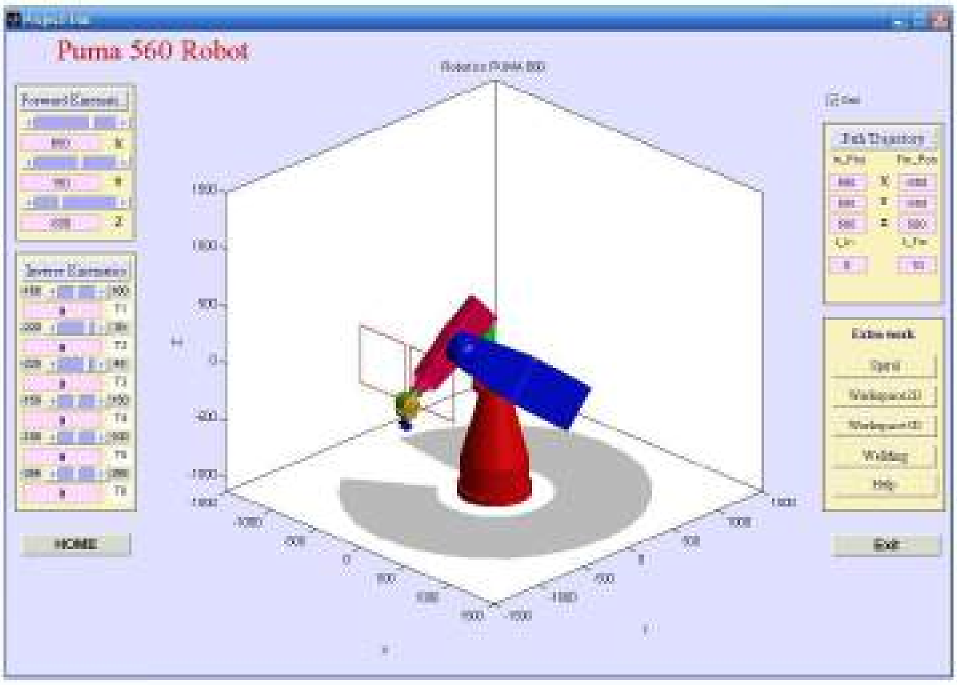


Figure 5.17 before Welding

After welding

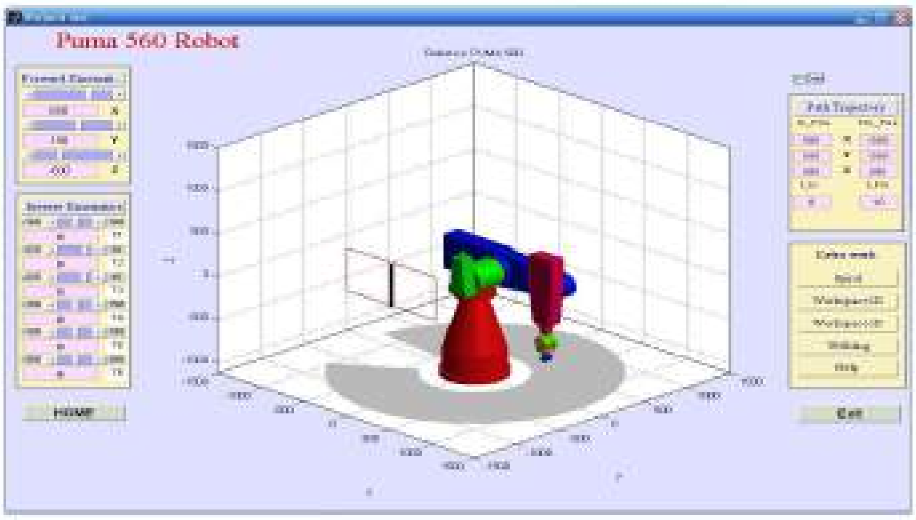


Figure 5.18 after Welding

5.9 Advance GUI controller comparative

The second GUI will show each controller like PID, FLC and SMC as shown in figure 5.19 below:

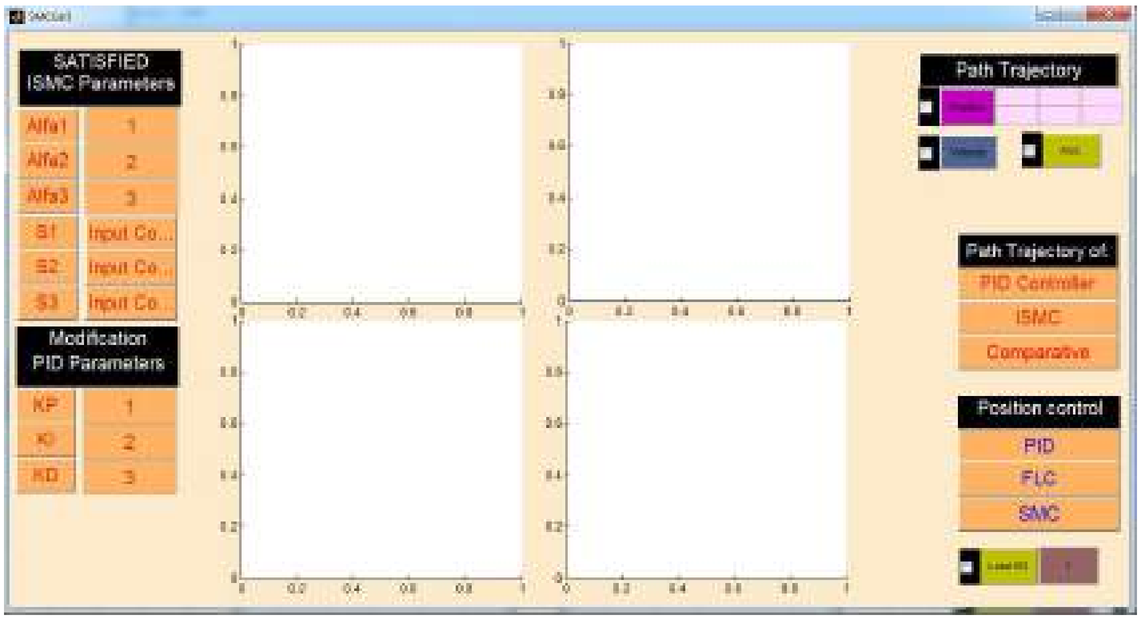


Figure 5.19 Advance GUI

Figure 5.19 shows an advance part of the educational tool with each controller individually. After the user enter the values of coordinates or the values of angles, he has to choose one of the controllers to be applied and then the program will show the results according to the choices. So in this part each controllers results will show individually.

Note that the values of entered data of the coordinates and values of angles must be constant if the user decided to know which controllers are more robust.

5.9 PID and ISMC Path Trajectory

First comparatives will be between PID and ISMC (Path Trajectory), open the advance GUI shown in Figure (5.19) and follow the steps below:

- Chose the Path as shown in chapter 4 as shown in Figure (5.20)
- Then enter the values of satisfied ISMC parameters in advance GUI.
- Tune the PID controller parameters from advance GUI.
- Finally get the comparative results of using PID or ISMC controllers for Path planning. Figure (5.21) present the response of tracking for PID, ISMC and comparative between them.

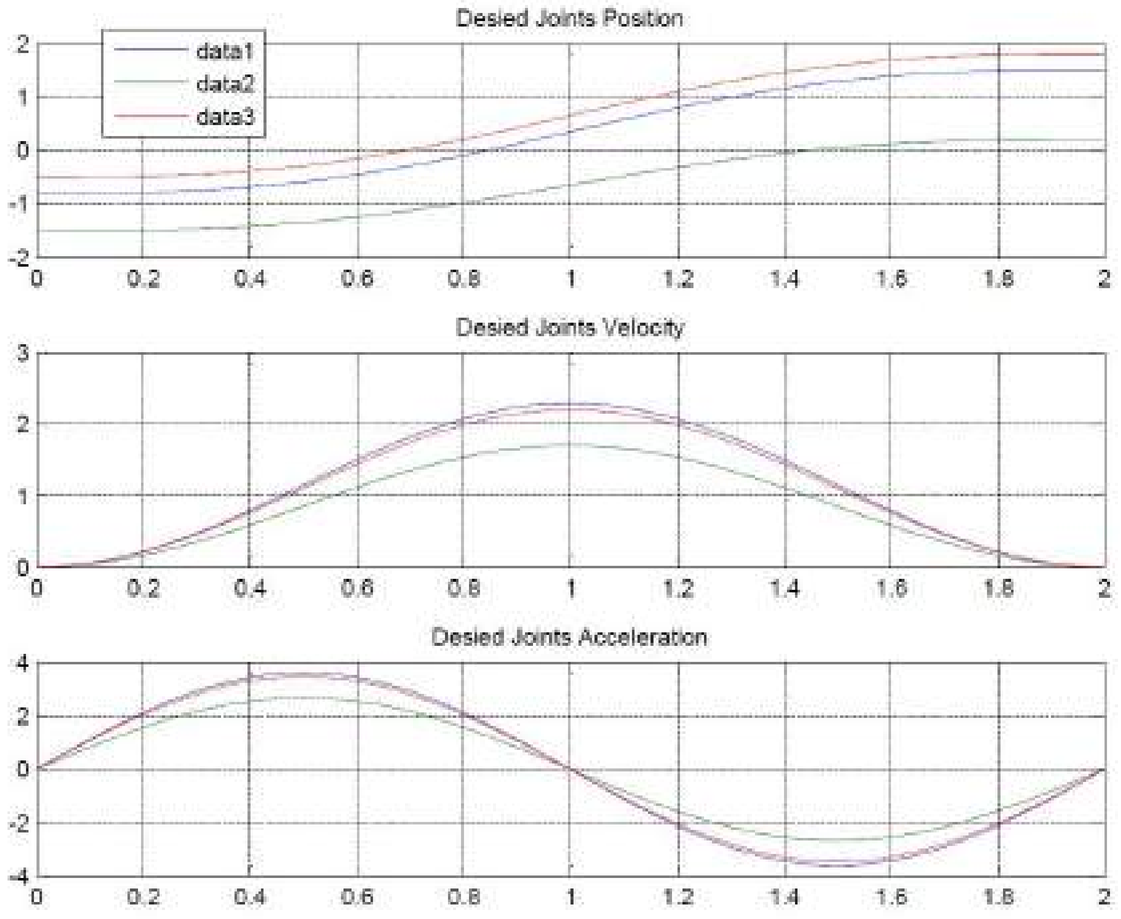
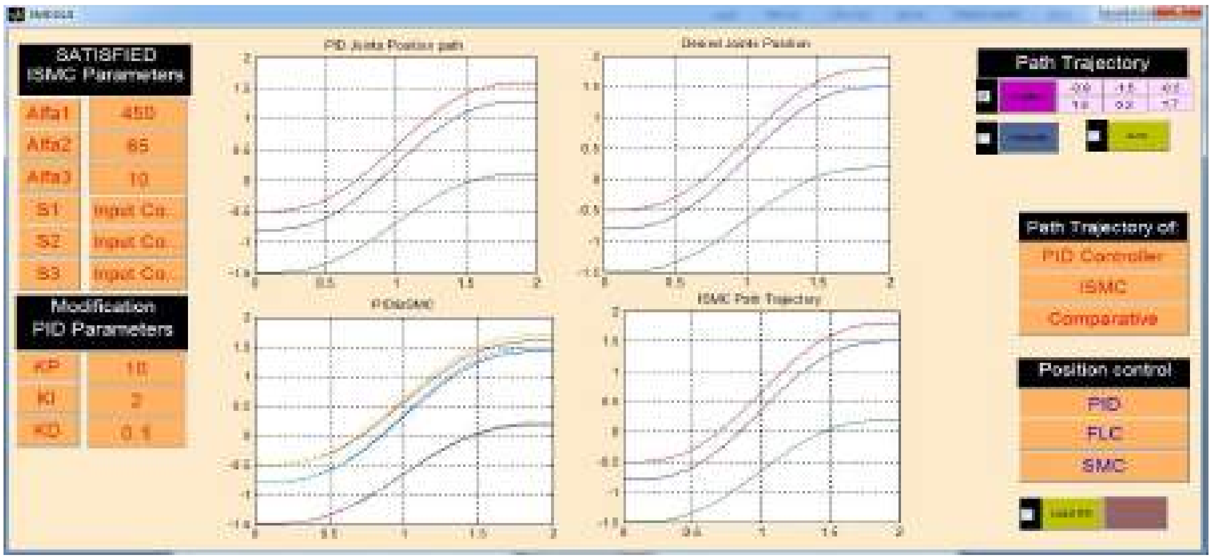


Figure 5.20 Desired Path

The mean idea of this part is to apply the selected controllers and showing the results in same figure, which make the decision of controllers choices easily.



SATISFIED ISMC Parameters

Alfa1	450
Alfa2	65
Alfa3	10
S1	Input Co.
S2	Input Co.
S3	Input Co.

Modification PID Parameters

KP	10
KI	2
KD	0.1

Figure 5.21 Path Tracking Response of PID and ISMC

Path Trajectory Experiment Result:

ISMC achieve the asymptotic tracking and disturbance rejection of bounded uncertainties, its follow the desired path with no effect of load and uncertainty.

PID follow the desired path with some limitation and its affected of load and uncertainty.

ISMC not need tuning if load is change since its depend on the parameters of switching function.

PID need on line tuning of its parameters.

ISMC not need to linearization its nonlinear controller.

PID needs linearization.

5.11 SMC Challenge and Solving of Disadvantages

The main target in this section is analyses and design of the position controller for links of PUMA robot manipulator to reach an acceptable performance. In the first part studies about classical sliding mode controller (SMC) which shows that: although this controller has acceptable performance with known dynamic parameters but by comparing the performance regarding to uncertainty, the SMC's output has fairly fluctuations and slight oscillation. Although SMC has many advantages such as stability and robustness but there are two important disadvantages as below:

Chattering phenomenon.

Mathematical nonlinear dynamic equivalent controller part.

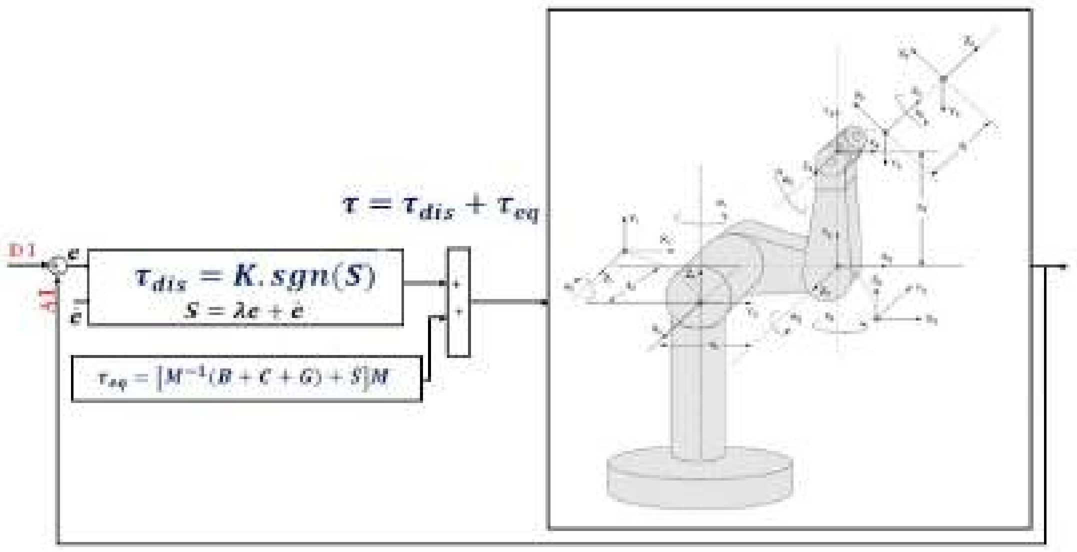

$\tau = \tau_{dis} + \tau_{eq}$
D1
A1
e
$\dot{e}$
$\tau_{dis} = K.sgn(S)$
$S = \lambda e + \dot{e}$
$\tau_{eq} = [M^{-1}(B + C + G) + \dot{S}]M$
+
+

Figure 5.22 Block diagram of a sliding mode controller

As shown in Figure 5.22, sliding mode controller is divided into two main parts: discontinuous part and equivalent part.

Discontinuous part is based on switching function, which this method is used to good following trajectory.

Equivalent part is based on robot manipulator's dynamic formulation, which these formulations are nonlinear MIMO.

Now we are going to design pure SMC as a first step, then discuss the system response (Overshoot, Rising time, Error and RMS error).

The second step will show the chattering phenomena from result and how to solve these problems.

5.12 Classical (Pure) SMC:

As shown in Figure 5.22 the SMC consist of two man part and these part are provide and discussed in mane paper [37] and [38].

The control output can be written as the sum of the two main part so:

$$\tau = \tau_{eq} + \tau_{dis} \tag{5.1 }$$

$$\text{Where } \tau_{dis} = K.\text{sgn}(S) \tag{5.2}$$

$$\text{And } \tau_{eq} = [M^{-1}(B + C + G) + S^{\cdot}]M \tag{5.3}$$

Therefore, the total output is the total output torque from the discontinuous and equivalent part

$$\tau = [M^{-1}(B + C + G) + S^{\cdot}]M + K.\text{sgn}(S) \tag{5.4}$$

The last equation will illustrated in SIMULINK as shown in Figure 5.23. Where the full SIMULINK is shown is Appendix D.

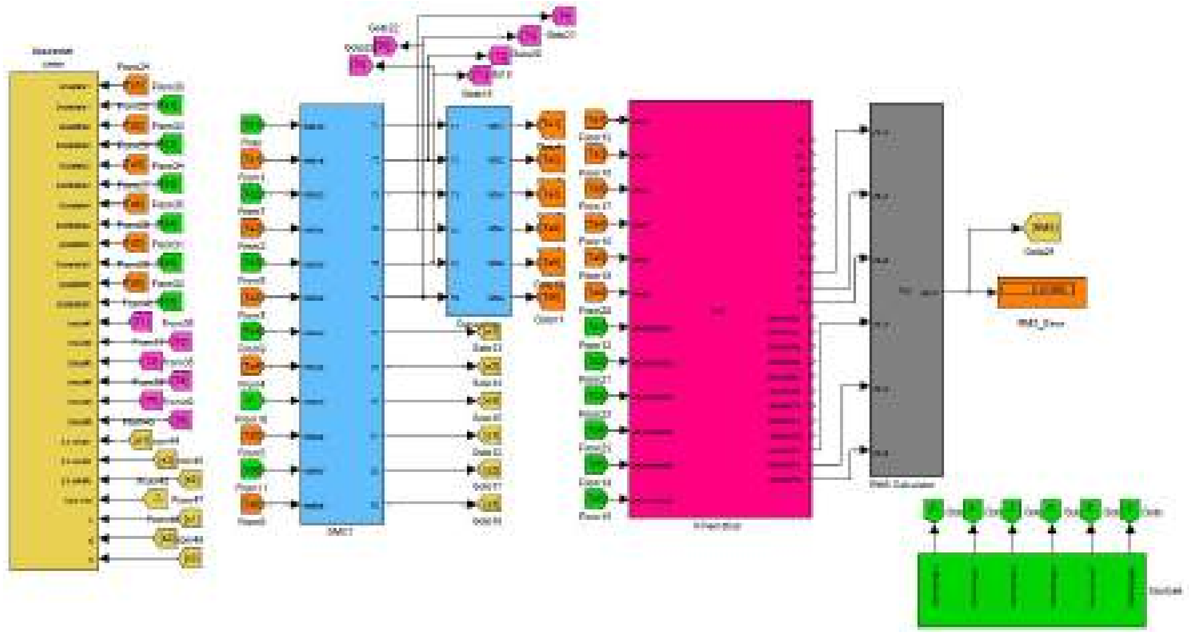


Figure 5.23 SIMULINK Pure SMC

5.12.1 Simulation Results of Classical SMC:

Experiment 1:

Enter the desired joint trajectory for each joint for example $[\theta_1(\tau)......\theta_6(\tau)]^T = [5]^T$ we suppose that the initial values of each angle are the same.

The simulation results will show the following :

The Trajectory for each joint of robot.

Sliding surface for the first three joints.

System response will summarized in table (5.1).

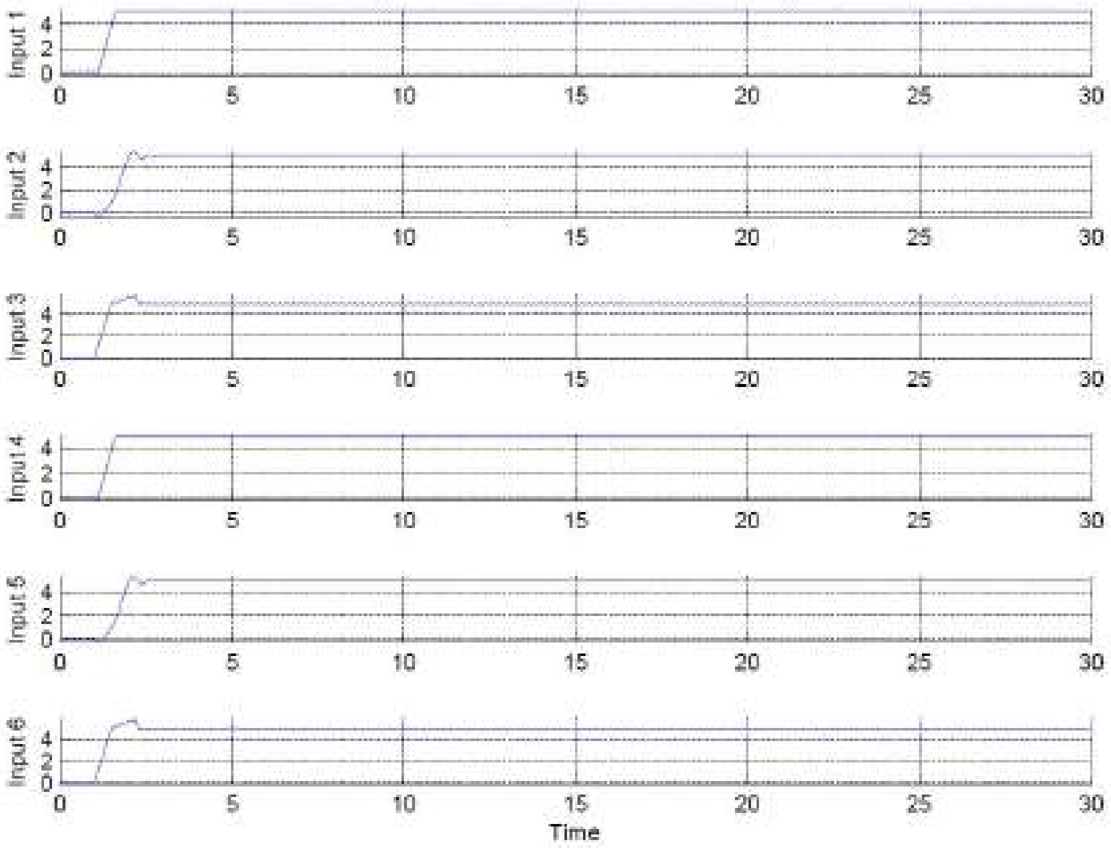


Figure 5.24 Robot Trajectory with Pure SMC

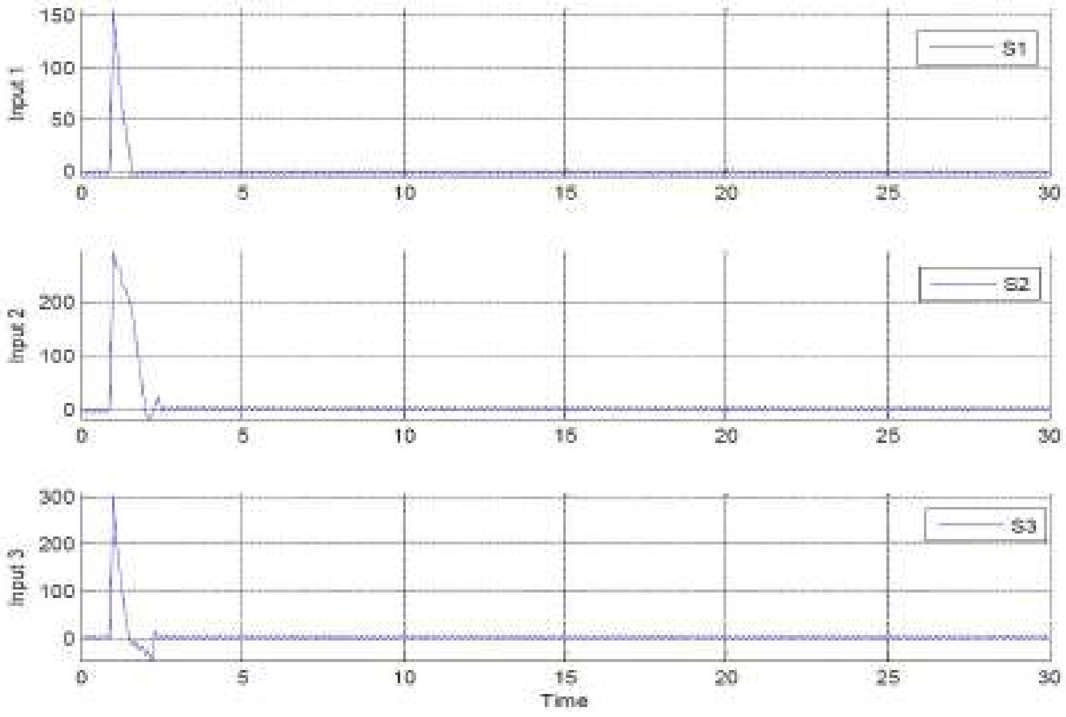


Figure 5.25 Sliding Surface of Pure SMC Links 1,2 and 3

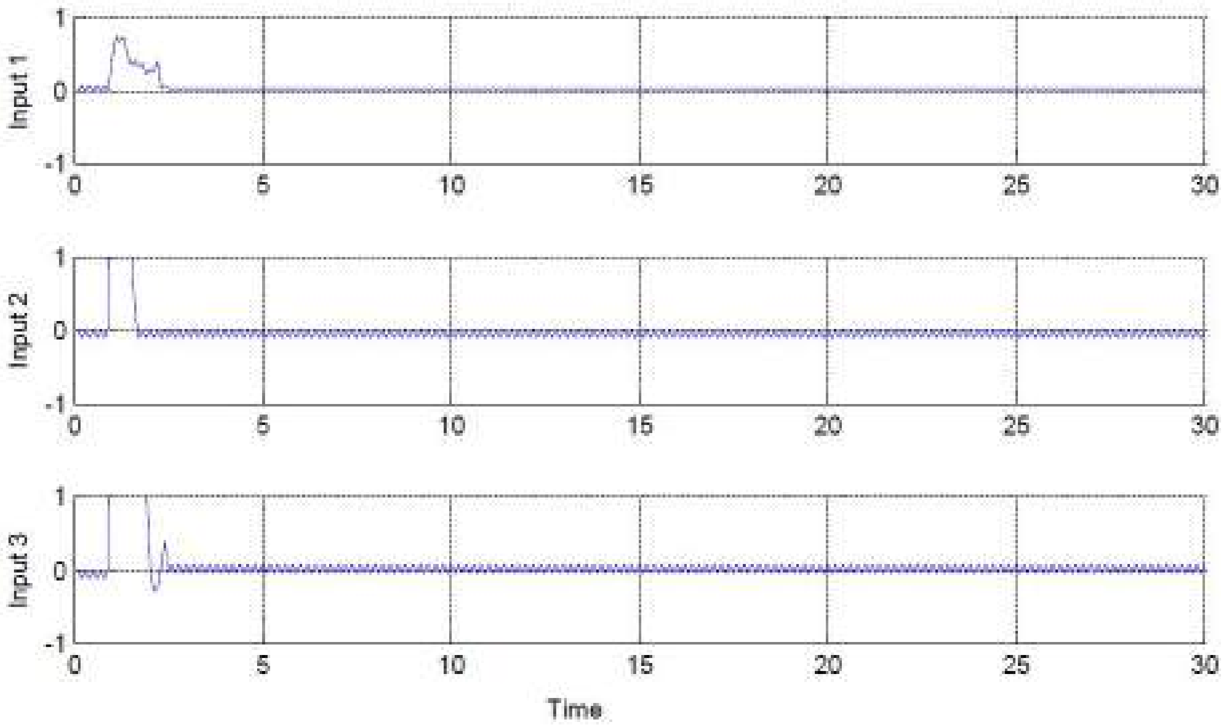


Figure 5.26 SMC Links Steady State Error

It is so clear from the Figures 5.24,5.25 and 5.29 that the pure SMC suffering from Chattering phenomena and the results of simulation are shown in table 5.1.

Table (5.1) Result of SMC

Properties Controller	RMS Error	Links Error	Overshoot	Rising Time Sec.	Chattering
SMC	0.01086	0	From 4% to 6%	0.4	Not Free
		0.07542			
		0			

5.13 Boundary Layer Saturation Method:

To reduce or eliminate the chattering, various papers have been reported by many researchers which classified into two most important methods: boundary layer saturation method and estimated uncertainties method. In boundary layer saturation method, the basic idea is the discontinuous method replacement by saturation method with small neighborhood of the switching surface. This replacement caused to increase the error performance against with the considerable chattering reduction.

5.13.1 Simulation Results of Boundary Layer Saturation Method:

Experiment 2:

Enter the desired joint trajectory for each joint for example $[\theta_1(\tau)......\theta_6(\tau)]^T = [5]^T$ we suppose that the initial values of each angle are the same.

The simulation results will show the following :

The Trajectory for each joint of robot.

Sliding surface for the first three joints.

System response will summarized in table (5.2).

Replace the SGN function with Saturation function as show in figure 5.27

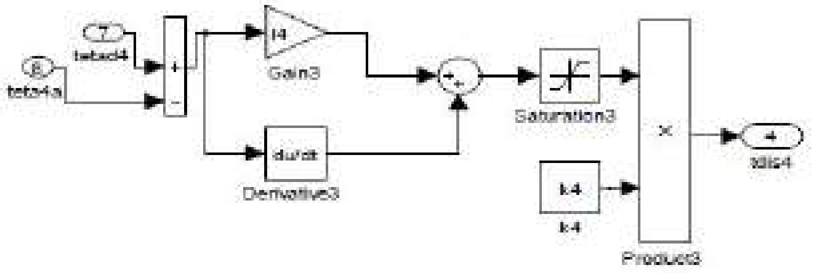


Figure 5.27 Boundary Layer Saturation

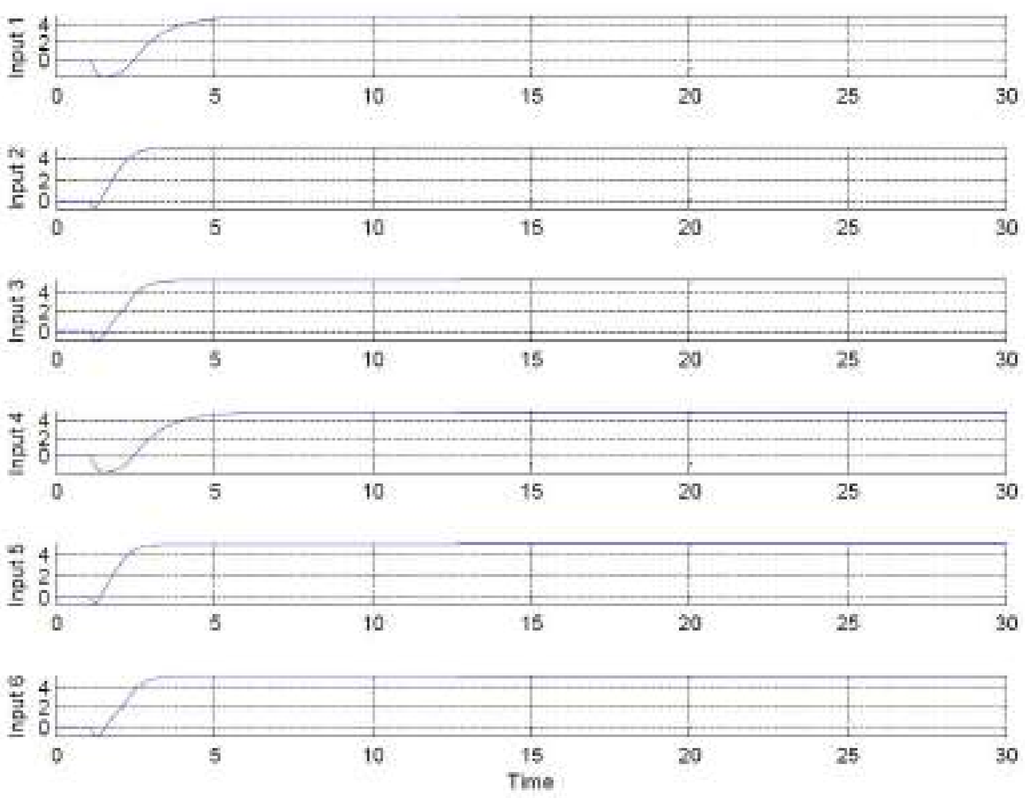


Figure 5.28 Robot Links Trajectory with Boundary

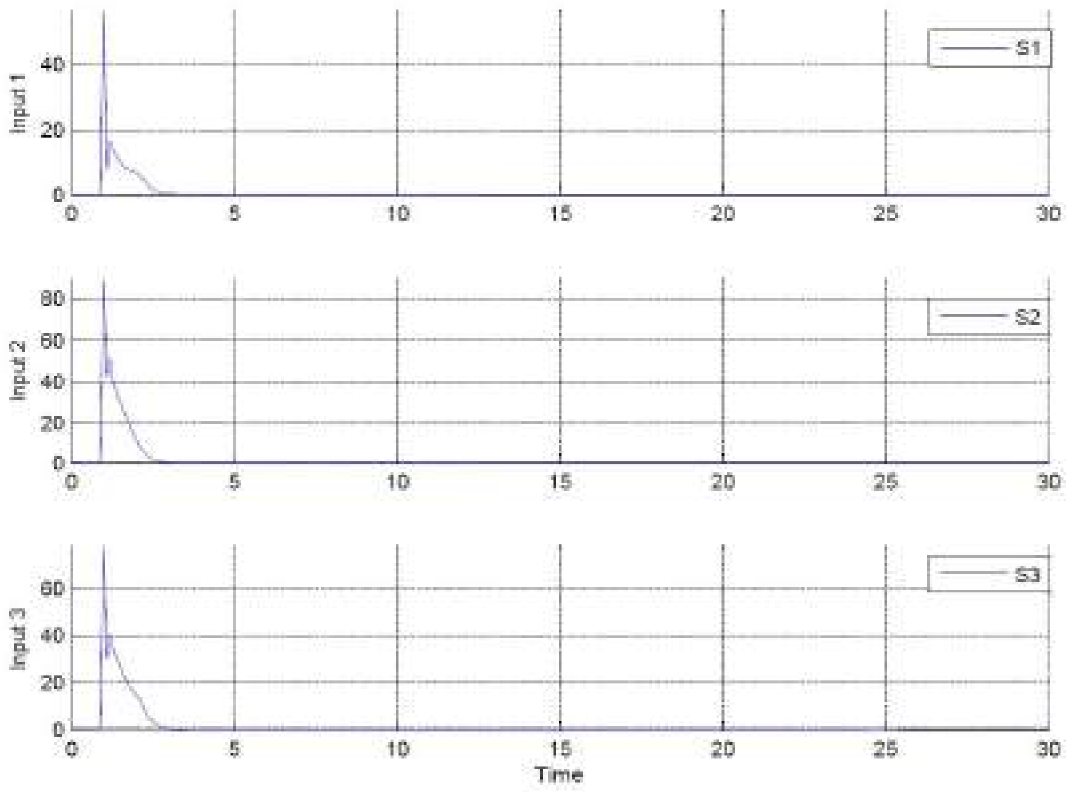


Figure 5.29 Sliding Surface BLS Links 1, 2 and 3

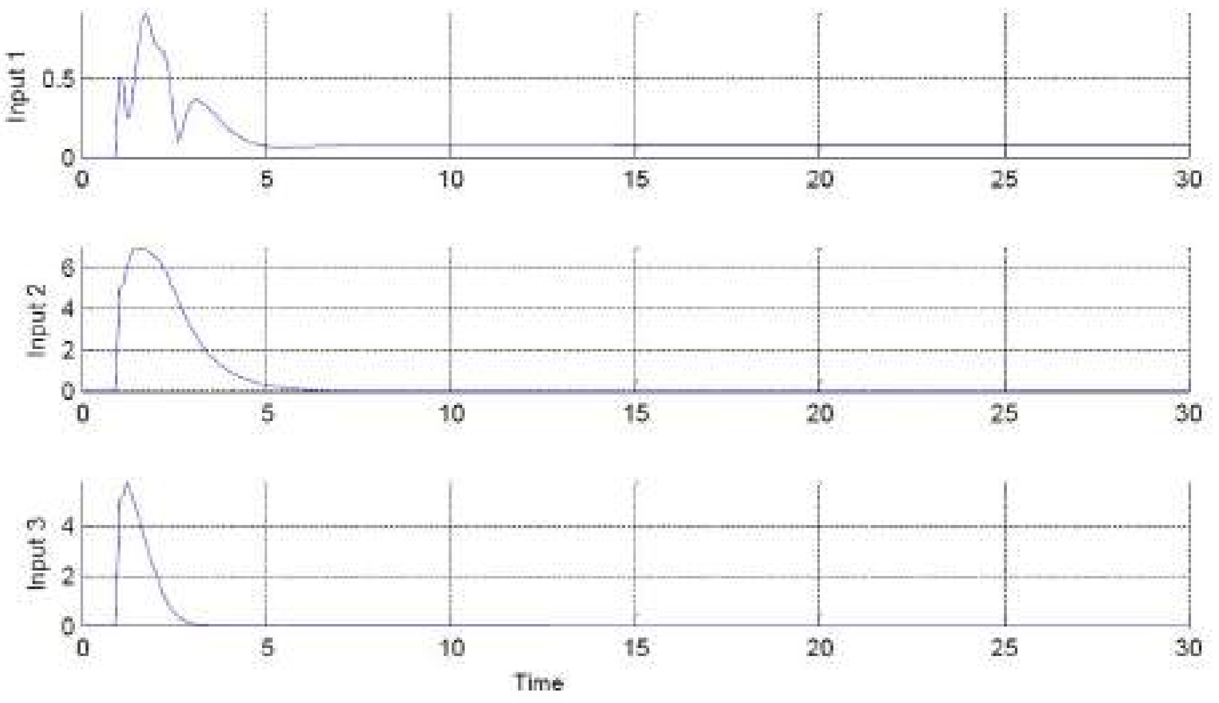


Figure 5.30 Boundary Layer Links Steady State Error

Table (5.2) Result of Boundary Layer Saturation

Properties Controller	RMS Error	Links Error	Overshoot	Rising Time Sec.	Chattering
Boundary Layer Saturation	0.07572	3.606e-013	From 4 % To 6 %	1.6	Free
		0			
		-0.1748			

From the above Experiment and table 5.2 results show that boundary layer saturation increase the RMS error and Error for each Links as rising time also increase.

5.14 SMFC Chattering Free and Eliminate Dynamic Part:

Equivalent part of sliding mode controller is based on nonlinear dynamic formulations of robot manipulator. Robot manipulator's dynamic formulations are highly nonlinear, therefore design a controller based on dynamic formulation is complicated. To solve this challenge fuzzy logic methodology is applied to sliding mode controller as show in Figure 5.31

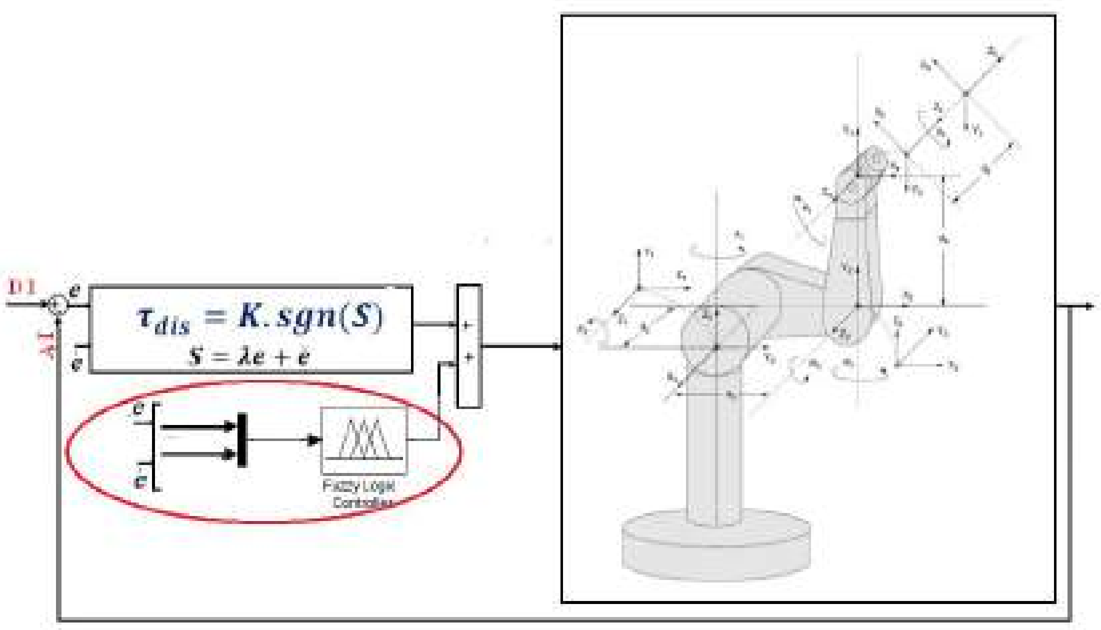


Figure 5.31 Block Diagram of Sliding Mode Fuzzy Controller

To design fuzzy controller that will estimate the equivalent dynamic part the following steps must done :

- **Determine Inputs and Outputs**

Two inputs error and change of error and the output name's is the Equivalent torque , where each input have 7 membership function that will give 49 rule base.

- **Find linguistic Variable**

The linguistic variables for error(e) are; Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (Z), Positive Small (PS), Positive Medium (PM), Positive Big (PB), the linguistic variables for change of error (e-) are ;Fast Left (FL), Medium Left (ML), Slow Left (SL),Zero (Z), Slow Right (SR), Medium Right (MR), Fast Right (FR). the linguistic variables for equivalent torque are; Zero (ZE), Very Small (VS), Small (S), Small Big (SB), Medium Big (MB), Big (B), and Very Big (VB).

- **Type of membership function**

Triangular membership function is selected because it has a high-quality response and they are shown in Figure 5.32, 5.33 and 5.34.

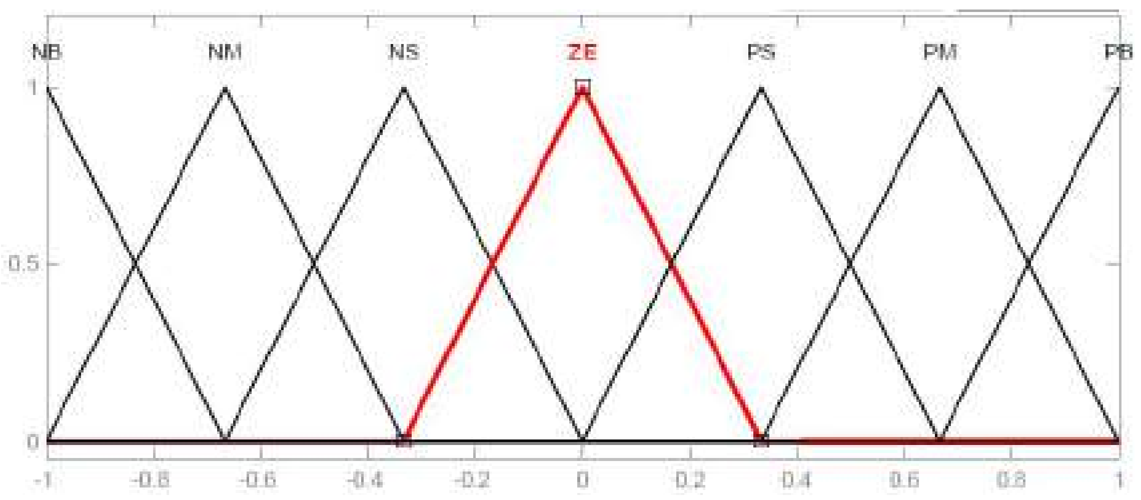


Figure 5.32 Input (1) Error MSF

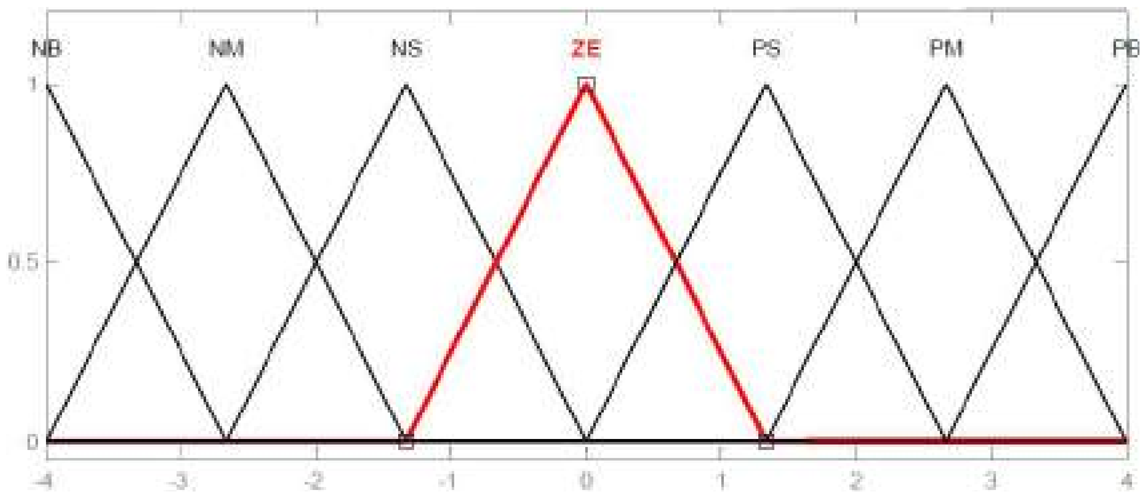


Figure 5.33 Input (2) Error change MSF

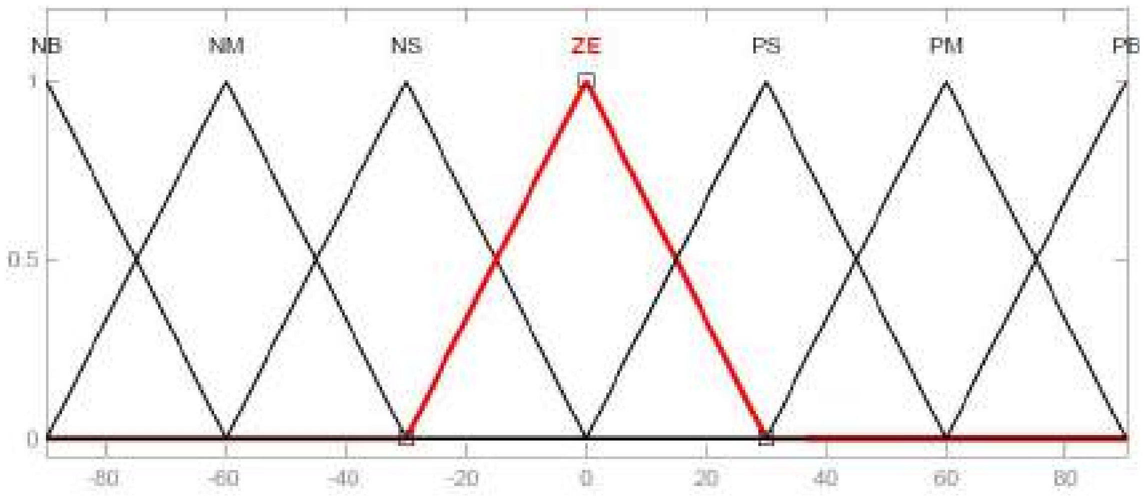


Figure 5.34 Output Equivalent Torque MSF

- **Design fuzzy rule table**

The rule base for equivalent torque consists of 49 Rule with IF…..THEN

FR^1 $\quad$ $IF\ e\ is\ NB\ and\ e^{\cdot}\ is\ \text{NB}\ THEN\ \ \tau\ is\ \text{PB}$

Table (5.3) Rule base of Torque

$e \backslash e^{\cdot}$	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
NB	**PB**	**NB**	**NB**	**NM**	**NS**	**NS**	**ZE**
NM	**NB**	**NM**	**NM**	**NM**	**NS**	**ZE**	**PS**
NS	**NB**	**NM**	**NS**	**NS**	**ZE**	**PS**	**PM**
ZE	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
PS	**NM**	**NS**	**ZE**	**PS**	**PS**	**PM**	**PB**
PM	**NS**	**ZE**	**PS**	**PM**	**PM**	**PM**	**PB**
PB	**PS**	**PS**	**PM**	**PB**	**PB**	**NB**	**ZE**

- **Defuzzification**

The defuzzification will done by using Center Of Graph (COG) method .

5.14.1 Simulation Results of SMFC

Experiment 3: Enter the desired joint trajectory for each joint for example $[\theta_1(\tau)......\theta_6(\tau)]^T = [5]^T$ we suppose that the initial values of each angle are the same.

The simulation results will show the following :

The Trajectory for each joint of robot.

Sliding surface for the first three joints.

System response will summarized in table (5.4).

Replace the dynamic part with fuzzy controller as show in figure 5.31.

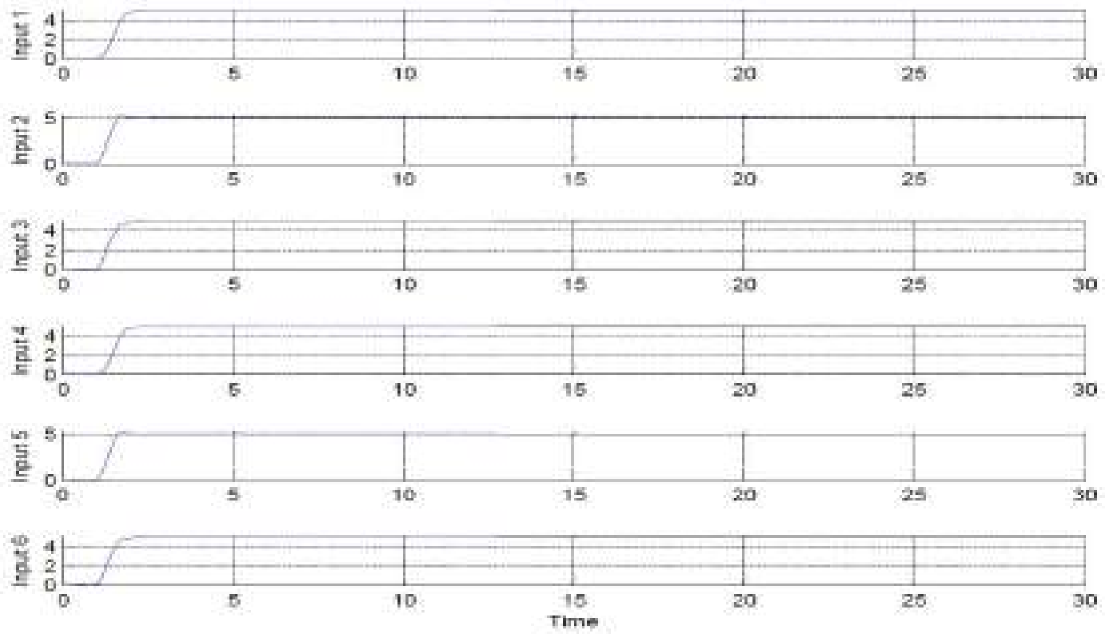


Figure 5.35 Robot Links Trajectory with SMFC

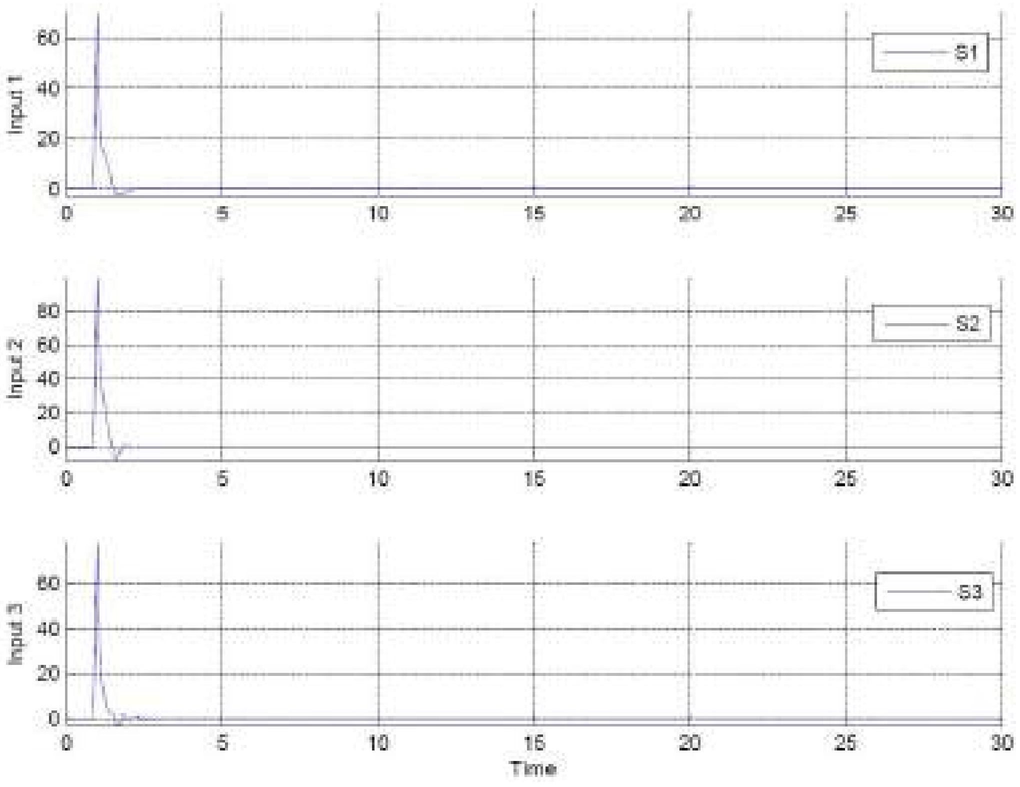


Figure 5.36 Sliding Surface of SMFC

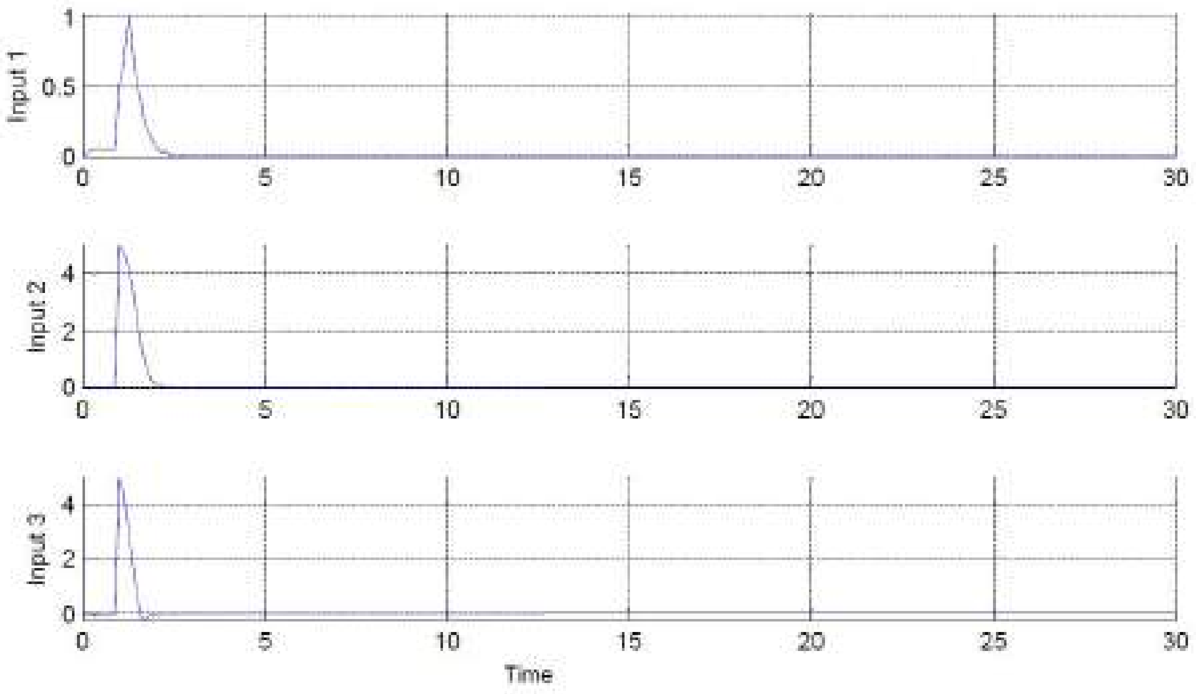


Figure 5.37 SMFC Links Steady State Error

Table (5.4) Result of SMFC

Properties Controller	RMS Error	Links Error	Overshoot	Rising Time Sec.	Chattering
SMFC	0.004908	0.0003947	From 4 % to 6 %	0.6	Free
		- 0.01038			
		- 0.01024			

Base on the experiment 3 and the table 5.4 it is clear that:

- Performance/error-based fuzzy logic controller estimates dynamic nonlinear equivalent part.
- In sliding mode fuzzy controller; error based Mandeni's fuzzy inference system has considered with two inputs, one output and totally 49 rules to estimate the dynamic equivalent part.
- A model free Mandeni’s fuzzy inference system has considered based on error-based fuzzy logic controller to estimate the nonlinear equivalent part.
- RMS error is less than last two previous in SMFC it is 0.004908, in SMC it is 0.01086 and in boundary layer it is 0.07572.
- There is No chattering in SMFC and SMC with boundary layer.

For both sliding mode controller and sliding mode fuzzy controller applications the system performance is sensitive to the sliding surface slope coefficient α . For instance, if large value of α is chosen the response is very fast the system is unstable and conversely, if small value of α is considered the response of system is very slow but system is stable. Therefore, to have a good response, compute the best value of sliding surface slope coefficient is very important.

To solve this challenge Fuzzy based tuning Sliding Mode Fuzzy Controller is designed .

5.14 Fuzzy Based Tuning Sliding Mode Fuzzy Controller:

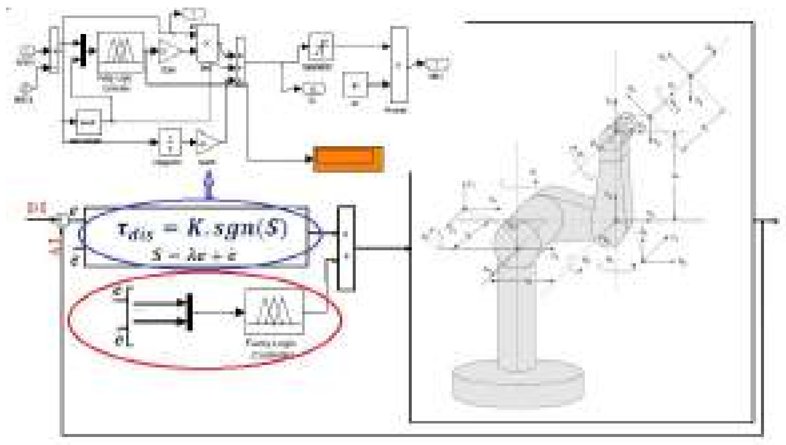


Figure 5.38 Fuzzy Based Tuning Sliding Mode Fuzzy Controller

As shown in Figure 5.38 combining sliding mode fuzzy controller and fuzzy-based tuning. It is based on resolve the on line sliding surface gain λ as well as improve the output performance. The sliding surface gain λ of this controller is adjusted online depending on the last values of error e and change of error $\dot{e}$ by sliding surface slope updating factor α .

Fuzzy-based tuning sliding mode fuzzy controller is stable model-free controller which eliminate the chattering phenomena.

Step 1:
The equivalent part as show in SMFC method will be replaced by fuzzy part, with the same steps in experiment 3 and the same fuzzy controller will be used.

Step 2:
Determine the second fuzzy controller that will online tuning the slop factor as follows:

- **Determine Inputs and Outputs**

Two inputs error and change of error and the output name's is sliding surface slope factor α .

- **Find linguistic Variable**

The linguistic variables for error(e) are; Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (Z), Positive Small (PS), Positive Medium (PM), Positive Big (PB), the linguistic variables for change of error (e-) are ;Fast Left (FL), Medium Left (ML), Slow Left (SL),Zero (Z), Slow Right (SR), Medium Right (MR), Fast Right (FR). the linguistic variables for sliding surface slope factor α .are; Zero (ZE), Very Small (VS), Small (S), Small Big (SB), Medium Big (MB), Big (B), and Very Big (VB).

- **Type of membership function**

Triangular membership function is selected because it has linear equation with regard to has a high-quality response and they are shown in Figure 5.42, 5.42 and 5.43

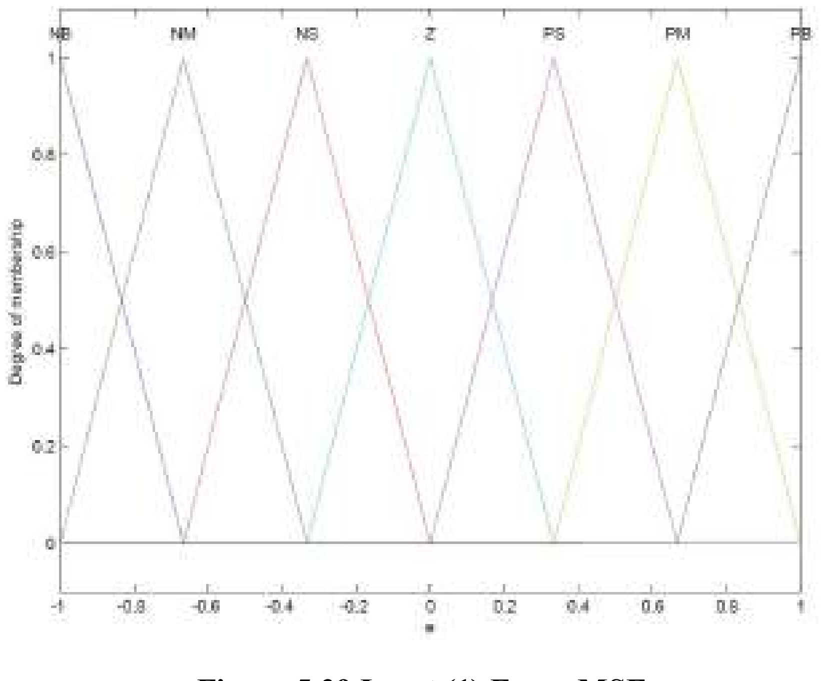


Figure 5.39 Input (1) Error MSF

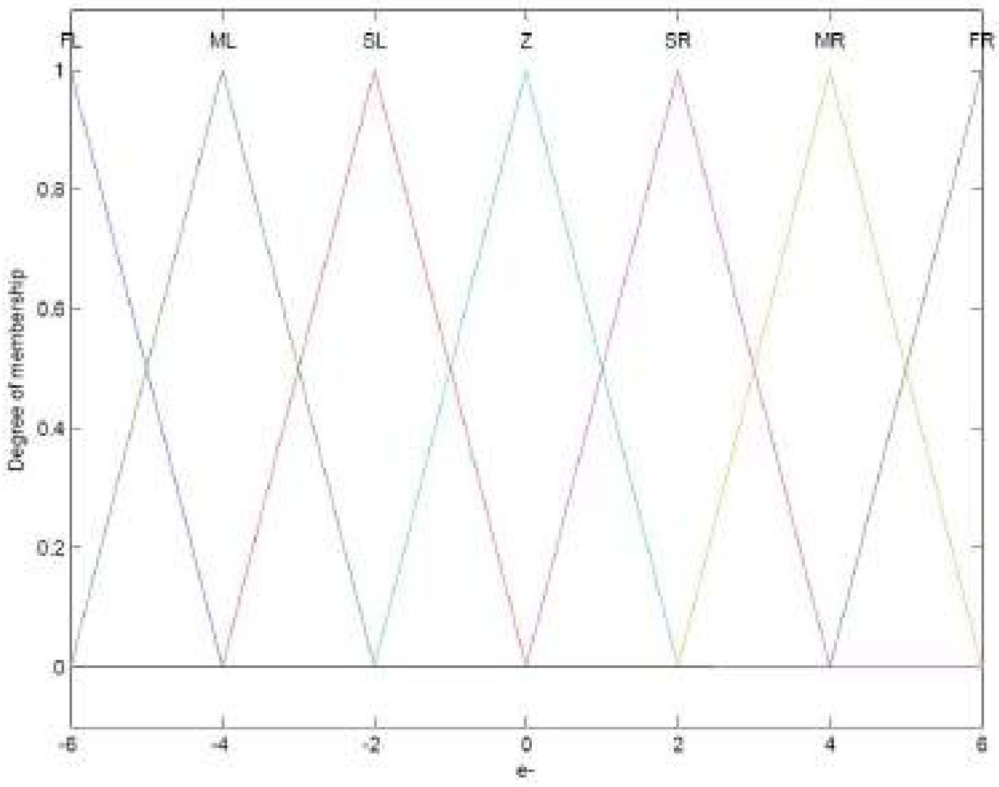


Figure 5.40 Input (2) Error Change MSF

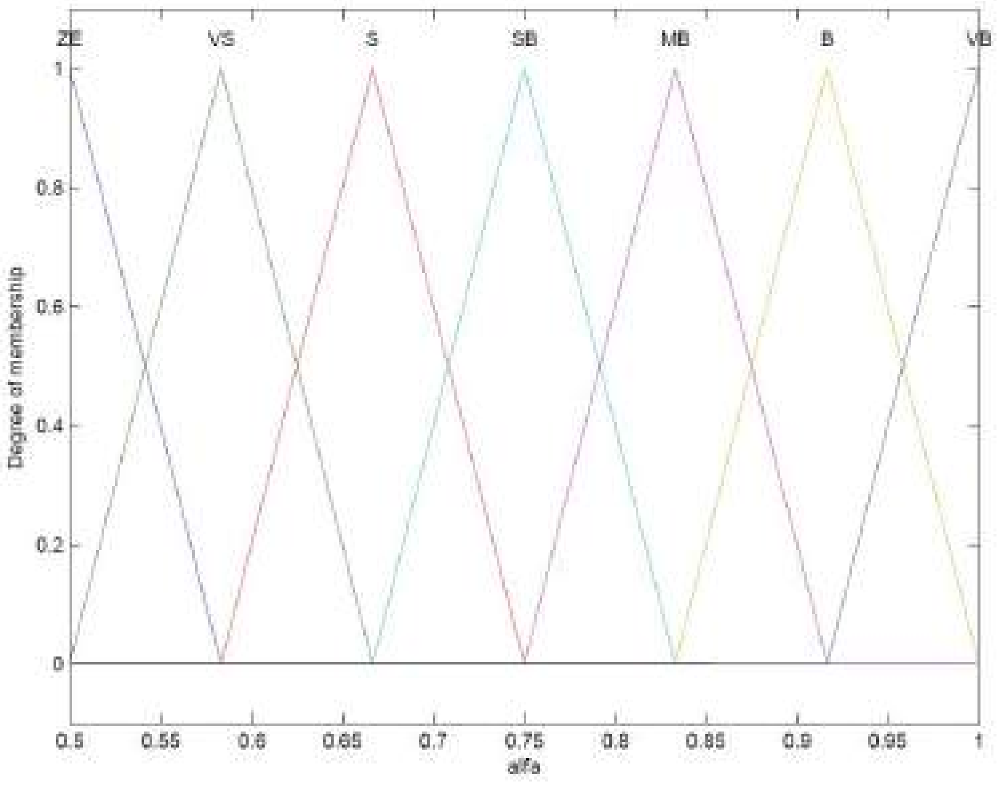


Figure 5.41 Output Sliding Surface Slope Factor α . MSF

- **Design fuzzy rule table**

The rule base for Sliding Surface Slope Factor α . consist of 49 Rule with IF…..THEN

FR^1 $\quad IF\ e\ is\ NB\ and\ e^{\cdot}\ is\ \mathrm{SL}\ THEN\ \ \alpha\ is\ \mathrm{VB}$

Table (5.5) Rule base of Slop Factor

$e \backslash e^{\cdot}$	**FL**	**ML**	**SL**	**ZE**	**SR**	**MR**	**FR**
NB	**VB**	**VB**	**VB**	**B**	**SB**	**S**	**ZE**
NM	**VB**	**VB**	**B**	**B**	**MB**	**S**	**VS**
NS	**VB**	**MB**	**B**	**VB**	**VS**	**S**	**VS**
ZE	**S**	**SB**	**MB**	**ZE**	**MB**	**SB**	**S**
PS	**VS**	**S**	**VS**	**VB**	**B**	**MB**	**VB**
PM	**VS**	**S**	**MB**	**B**	**B**	**VB**	**VB**
PB	**ZE**	**S**	**SB**	**B**	**VB**	**VB**	**VB**

- **Defuzzification**

The defuzzification will done by using Center Of Graph (COG) method .

5.14.1 Simulation Results of fuzzy-based tuning-SMFC

Experiment 4: Enter the desired joint trajectory for each joint for example $[\theta_1(\tau)......\theta_6(\tau)]^T = [5]^T$ we suppose that the initial values of each angle are the same.

The simulation results will show the following :

The Trajectory for each joint of robot.

Sliding surface for the first three joints.

System response will summarized in table (5.6).

Replace the dynamic part with fuzzy controller as show in figure 5.38.

Apply the second fuzzy controller which give the slop factor values.

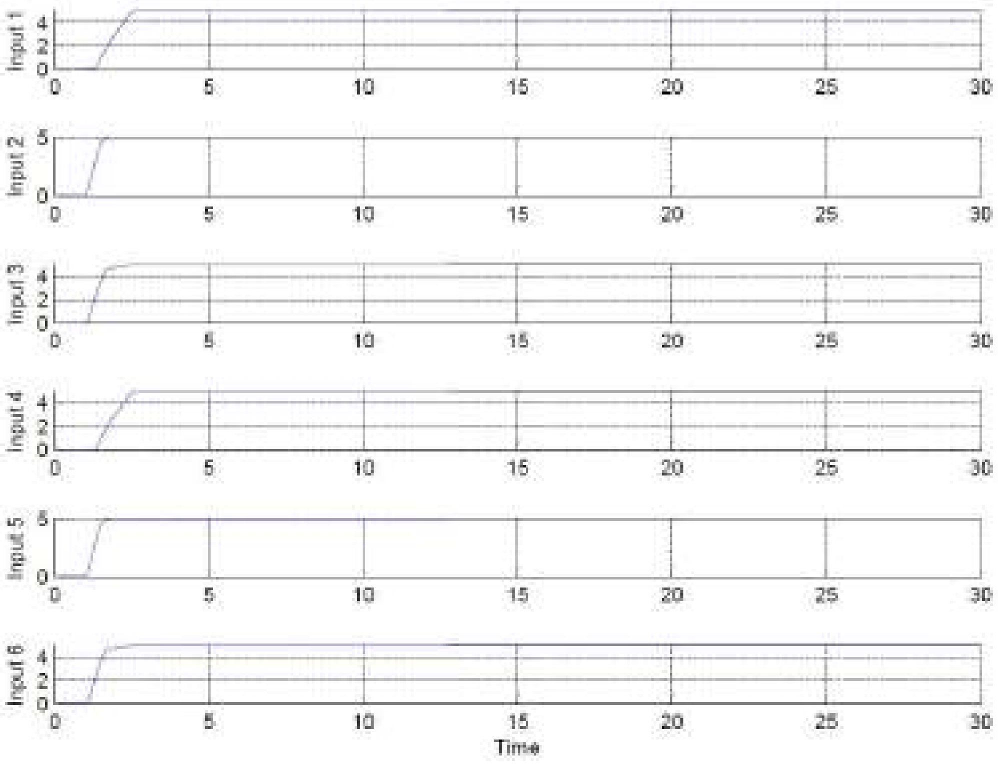


Figure 5.42 Robot Links Trajectory Online Tuning SMFC

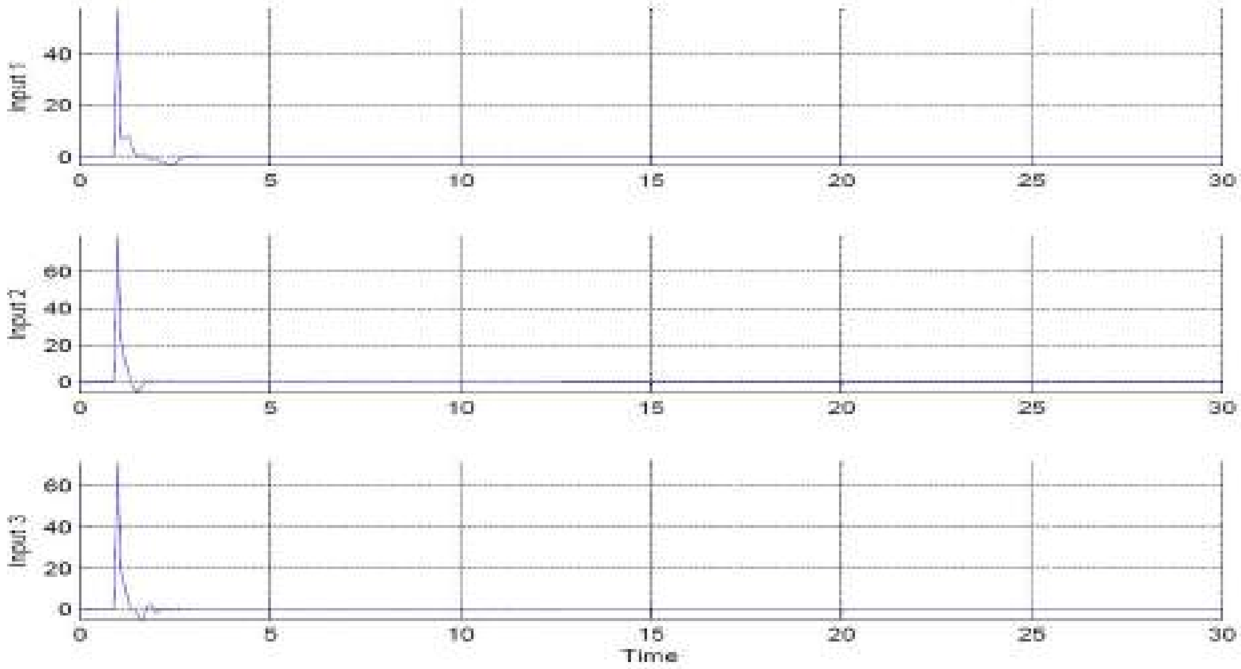


Figure 5.43 Sliding Surface of Online Tuning SMFC

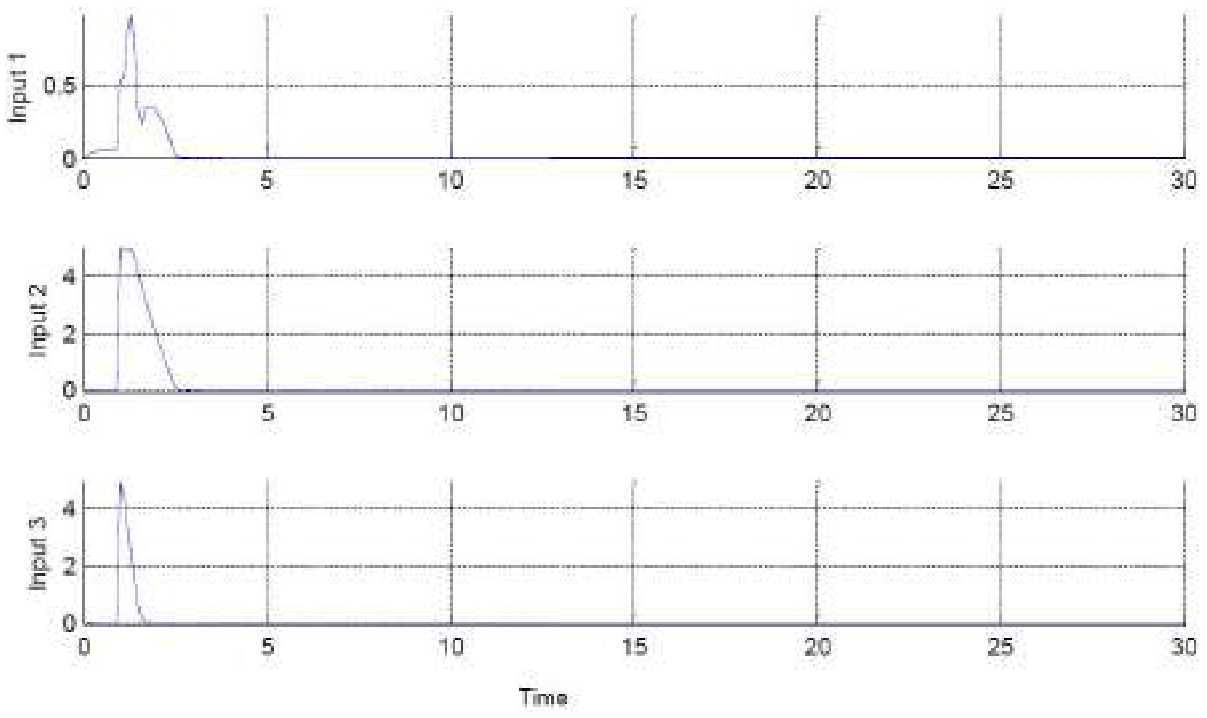


Figure 5.44 On-line Tuning SMFC Links Steady State Error

Table (5.6) Result of Online Tuning SMFC

Properties Controller	RMS Error	Links Error	Overshoot	Rising Time Sec.	Chattering
On-Line SMFC	0.003528	-1.371e005	Zero	0.4	Free
		-0.01677			
		0.007039			

Based on table 5.6 and results of experiment 4 : Fuzzy-based tuning sliding mode fuzzy controller is stable model-free controller which eliminate the chattering phenomena, estimate dynamic part and adjusting the slop factor. The results show that the RMS error is 0.003528. But as we note that rule base number is very big since each fuzzy controller used 49 rule so the total Number is 98 rule. Reducing rule base by trial and error may be hard mission but as we will introduced in the next section it will achieve the best performance. As we will see, So the number of rules in both fuzzy controller become 28 rules instead off 49.

5.15 Fuzzy Based Tuning Sliding Mode Fuzzy Controller With Reduced Rule Base

The same as section 5.14 will be repeated again but we will reducing the Rule base:

we have two rule base each one is 49 it will be reduced to 28 rule as shown in table :

Design fuzzy rule table

The rule base for equivalent torque consist of 28 Rule with IF…..THEN

FR^1 $\quad$ $IF\ e\ is\ NB\ and\ e^{\cdot}\ is\ \mathrm{NB}\ THEN\ \ \tau\ is\ \mathrm{PB}$

Table (5.7) Modified Torque rule base

$e \backslash e^{\cdot}$	NB	NM	NS	ZE	PS	PM	PB
NB	PB	NB	NB	🟦	NS	NS	ZE
NM	🟦	NM	NM	🟦	NS	🟦	PS
NS	🟦	🟦	🟦	NS	🟦	PS	🟦
ZE	NB	🟦	🟦	ZE	PS		PB
PS	NM	NS	ZE	PS	🟦	PM	
PM	🟦	ZE	PS	🟦	🟦	🟦	PB
PB	PS	🟦	🟦	PB	PB	NB	🟦

- **Design fuzzy rule table**

The rule base for Sliding Surface Slope Factor α . consist of 28 Rule with IF…..THEN

FR^1 $\quad$ $IF\ e\ is\ NB\ and\ e^{\cdot}\ is\ \mathrm{SL}\ THEN\ \ \alpha\ is\ \mathrm{VB}$

Table (5.8) Modified Slope Factor rule base

$e \backslash e^{\cdot}$	FL	ML	SL	ZE	SR	MR	FR
NB	VB	VB	VB	B	SB	S	ZE
NM	⬛	⬛	⬛	⬛	⬛	⬛	VS
NS	⬛	MB	⬛	VB	VS	⬛	⬛
ZE	S	SB	MB	ZE	MB	SB	S
PS	⬛	S	VS	⬛	⬛	MB	⬛
PM	VS	⬛	⬛	⬛	B	⬛	⬛
PB	ZE	⬛	SB	B	VB	⬛	VB

After reducing the rule, base the response of the system as shown:

5.15.1 Simulation Results of fuzzy-based tuning-SMFC With Reduced Rule Base

Experiment 5: Enter the desired joint trajectory for each joint for example $[\theta_1(\tau)......\theta_6(\tau)]^T = [5]^T$ we suppose that the initial values of each angle are the same.

The simulation results will show the following :

The Trajectory for each joint of robot.

Sliding surface for the first three joints.

System response will summarized in table (5.9).

Replace the dynamic part with fuzzy controller as show in figure 5.38.

Apply the second fuzzy controller which give the slop factor values.

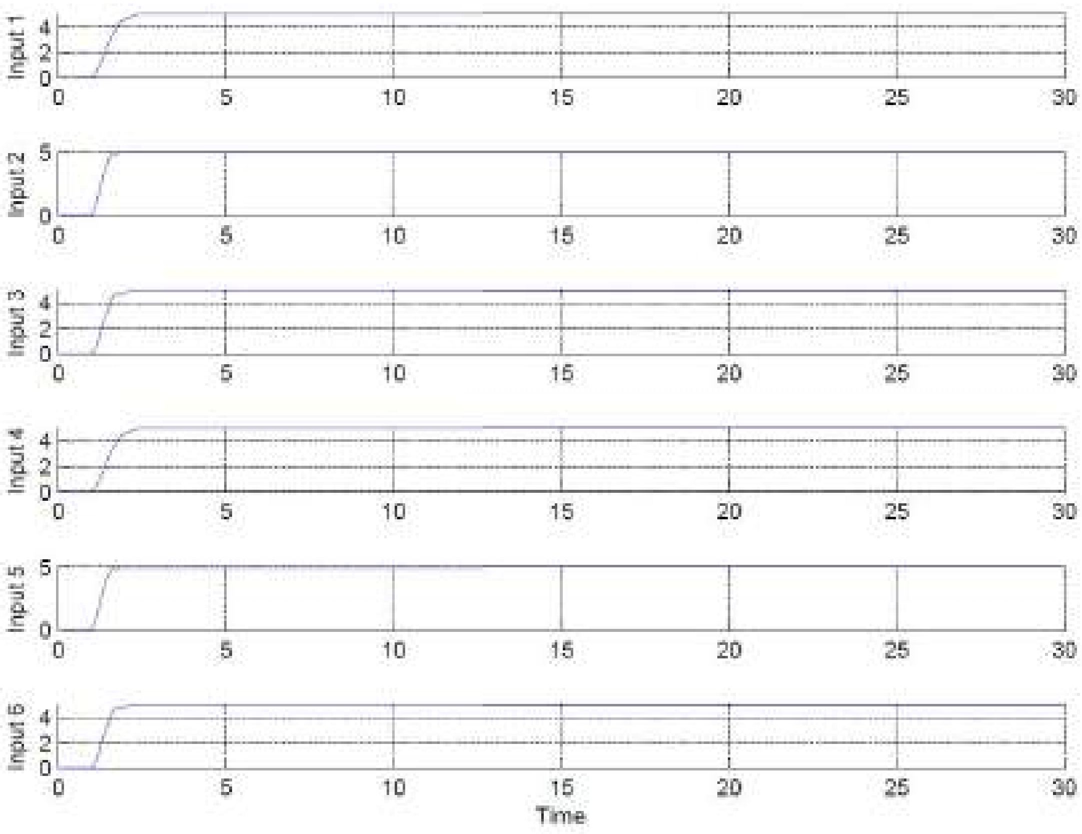


Figure 5.45 Trajectory On-line Tuning SMFC with Min. Rule Base

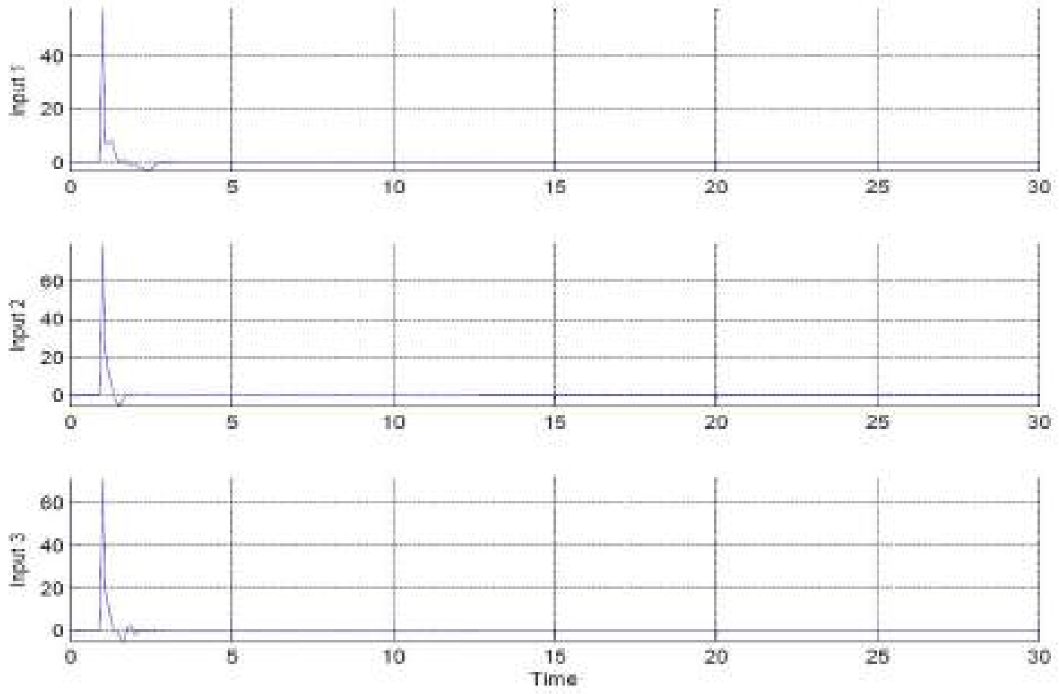


Figure 5.46 Sliding Surface of Online Tuning SMFC with Min. Rule Base.

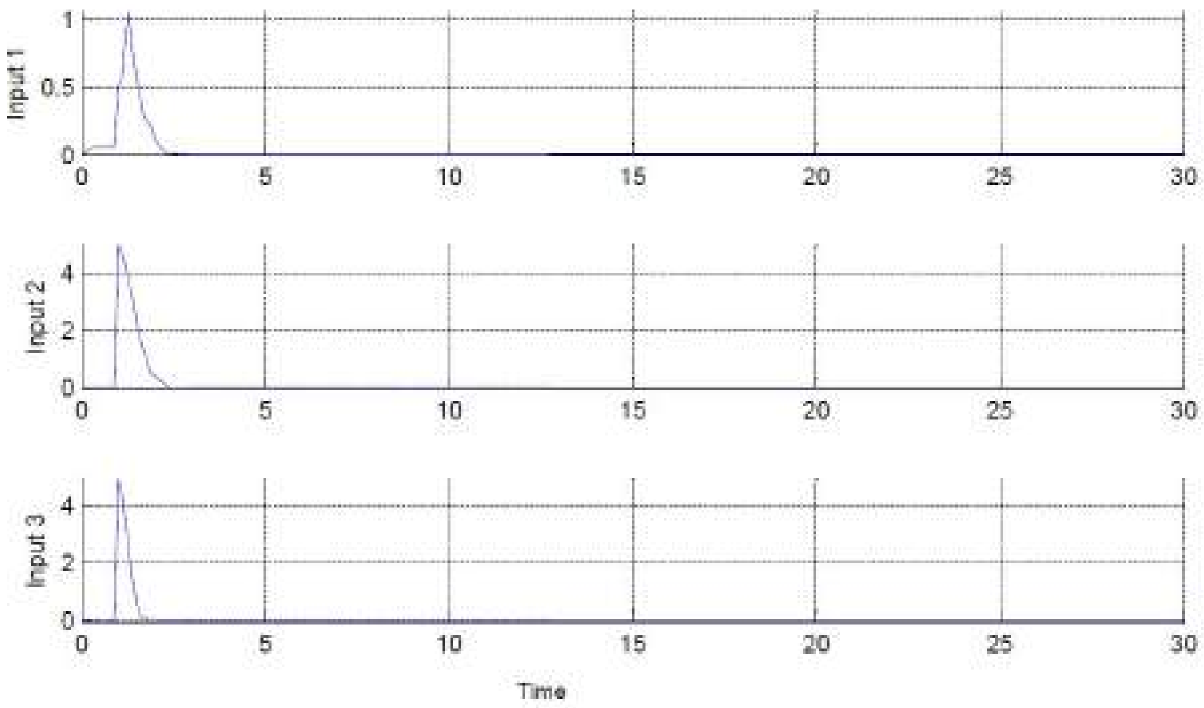


Figure 5.47 On-line Tuning SMFC Links Steady State Error with Min. Rule Base

Table (5.9) Result of Online Tuning SMFC using Modified rule base

Properties Controller	RMS Error	Links Error	Overshoot	Rising Time Sec.	Chattering
On-Line SMFC With Min. Rule	0.002712	0	Zero	0.4	Free
		-0.01442			
		0.004994			

The results demonstrate that the sliding mode fuzzy controller is a model-based controllers which works well in certain and partly uncertain system. Pure sliding mode controller and sliding mode fuzzy controller have difficulty in handling unstructured model uncertainties. It is possible to solve this problem by combining sliding mode fuzzy controller and fuzzy-based tuning. Since the sliding surface gain λ is adjusted by fuzzy based tuning method. The sliding surface slope updating factor α of fuzzy-based tuning part can be changed with the changes in error and change of error rate between half to one.

Sliding surface gain is adapted on-line by sliding surface slope updating factor. In pure sliding mode controller and sliding mode fuzzy controller the sliding surface gain is chosen by trial and error, which means pure sliding mode controller and sliding mode fuzzy controller must have a prior knowledge of the system uncertainty. If the knowledge is not available, steady state error and chattering phenomenon are go up. it is observed that fuzzy-based tuning sliding mode fuzzy controller is a model-free stable control for robot manipulator. It is a best solution to eliminate chattering phenomenon with switching function in structure and unstructured uncertainties.

CHAPTER 6 CONCLUSION AND FUTURE WORK

6.1 Conclusion

Modeling design of manipulator robot like Puma560, using SIMULINK\GUI was established and the proposed model was used as simulation environment.

The GUI presented kinematic analysis in 3D and included showing of 3 D work space. The model in SIMULINK presented the overall specification of the dynamic part of the robot and DC motors.

The problem of control techniques was discussed where SMC was implemented and applied, to control the robot manipulator, then the boundary layer saturation wan implemented and considered as the first choice to solving the chattering problem .the second controller was SMFC with Mamdani rule base was implemented to estimate the equivalent dynamic part . The third controller was combining between SMFC and fuzzy base tuning. We compared the results of using the three controllers for controlling the robot manipulator. First, we compared the results of pure SMC and Boundary layer saturation techniques in terms of RMS error, steady state error, overshoot and transient response.

Second, we compared the results of pure SMC, SMFC and Fuzzy base tuning SMFC simulations were presented using MATLAB and SIMULINK, which are used widely in control applications.

The simulations and numerical results of the previous controllers were presented in this thesis and summarized in tables. We proved that the Fuzzy base tuning-SMFC is more efficient in the time response behavior than other controllers. The RMS error of Fuzzy base tuning SMFC was 0.003528, while in SMFC it was 0.004908 and it was 0.01086 in pure SMC. We also showed that the rising time for the SMC and Fuzzy base tuning SMFC were 0.4 Sec. while it was 0.6 Sec in SMFC and marked worst rising time 1.6 Sec. with boundary layer saturation method with highest RMS error 0.07572.

The rule base was reduced in fuzzy base tuning SMFC to 28 rule instead of 49 in each controller of the combination after implemented this controller the results was more satisfactory and RMS error was 0.002712 with no chattering.

The simulation results if the combining SMFC and Fuzzy base tuning controller solved the problem of handling unstructured mode uncertainties and achieved stability, robustness and reliability.

6.2 Future work

Further study may be carried to:

- Realize the performance and practicality of the SM Controller.
- Apply different nonlinear controller to robot manipulator and make comparative.

REFERENCES

[1] A.Z. Alassar, I.M. Abuhadrous and H.A. Elaydi "Control of 5DOF Robot Arm Using PID Controller with Feedforward Compensation", The 2nd Conference on Computer and Automation Engineering, Singapore, 5. Dec. 2009.

[2] A.Z. Alassar, I.M. Abuhadrous and H.A. Elaydi "Comparison between FLC and PID Controller for 5DOF Robot Arm", The 2nd IEEE International Conference on Advanced Computer Control, Shenyang, Dec. 2009.

[3] J. Angeles, Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms, 2nd Edition, Springer, 2003.

[4] J.J. Crage, Introduction to Robotics Mechanics and Control, 3rd Edition, Prentice Hall, 2005.

[5] M.W. Spong, S. Hutchinson and M. Vidyasagar, Robot Modeling and Control, Jon Wiley & Sons, Inc, 2005.

[6] N. Sadati, and A. Talasaz, "Sliding mode control for a flexible transmission system," 2004 IEEE Conference on System, Man & Cybernetics, Nethertands, in press.

[7] Farzin Piltan, A. R. Salehi and Nasri B Sulaiman.," Design artificial robust control of second order system based on adaptive fuzzy gain scheduling," world applied science journal (WASJ), 13 (5): 1085-1092, 2011.

[8] E.H. Mamdani, "Applications of fuzzy logic to approximate reasoning using linguistic synthesis," IEEE Transactions on Computers, Vol. 26, No. 12, pp. 1182–1191, Dec. 1977.

[9] G.K.I. Mann, B.-G. Hu, and R.G. Gosine, "Analysis of Direct Action Fuzzy PID Controller Structures," IEEE Transactions on Systems, Man, and Cybernetics, Part B, Vol. 29, No. 3, pp. 371–388, Jun. 1999.

[10] A. Visioli, "Tuning of PID Controllers with Fuzzy Logic," IEEE Proceedings Control Theory and Applications., Vol. 148, No. 1, pp. 1-8, 2001.

[11] DE Carlo, R. A., Zak, S. H., and Matthews, G. P. (1988). Variable Structure Control of Nonlinear Multivariable Systems: A Tutorial. Proceedings of the IEEE. 76(3): 212-232.

[12] F. H. Arashima, H. Hashimoto, and K. Maruyama, "Practical robust control of robot arm using variable structure system", Proc. IEEE, Int. Conf. on Robotics and Automation, San Francisco, pp. 532-538, 1986.

[13] S. Elgazzar, "Efficient Kinematic Transformations for the Puma 560 Robot," IEEE Journal of Robotics and Automation., Vol. 1, No. 3, pp. 142–151, 1985.

[14] G.M. Khoury, M. Saad, H.Y. Kanaan and C. Asmar, "Fuzzy PID Control of a Five DOF Robot Arm," Journal of Intelligent and Robotic systems., Vol. 40, No.3, pp.299–20, 2004.

[15] B.W. Bekit, L.D. Seneviratne, J.F. Whidborne and K. Althoefer, "Fuzzy PID Tuning for Robot Manipulators," IEEE Industrial Electronics Society, in 24th Annual Conference., Vol. 4, pp. 2452–2457, Sep. 1998.

[16] B. Koyuncu and M. Güzel, “Software Development for the Kinematic Analysis of a Lynx 6 Robot Arm,” Proceedings of World Academy of Science, Engineering and Technology., Oct. 2007.

[17] B. K. Yoo and W. C. Ham, "Adaptive control of robot manipulator using fuzzy compensator", Fuzzy Systems, IEEE Transactions on, vol. 8, no. 2, (2002), pp. 186-199.

[18] A.Y. -Nones, “Heterogeneous Modeling & Design of Robot Arm Control System,” University of Puerto Rico, 2004.

[19] H. Medhaffar, N. Derbel and T. Damak, "A decoupled fuzzy indirect adaptive sliding mode controller with application to robot manipulator", International Journal of Modelling, Identification and Control, vol. 1, no. 1, (2006), pp. 23-29.

[20] Young, K-K.D. (1978),” A Variable Structure Model Following Control Design for Robotics Application.”IEEE Journal of Robotics and Automation, Vol 4, No.5, pp. 556-561.

[21] Slotine, J., and Sastry, [1983]: Tracking Control of Nonlinear Systems Using Sliding Surfaces, with application To robot Manipulators. Int. J. Control, 38, pp.465-492..

[22] T. R. Kurfess, Robotics and automation handbook: CRC, 2005.

[23] J. J. Slotine and S. Sastry, "Tracking control of non-linear systems using sliding surfaces, with application to robot manipulators†," International Journal of Control, No. 2, vol. 38, pp. 465-492, 1983.

[24] Farzin Piltan, SH. Tayebi HAGHIGHI, N. Sulaiman, Iman Nazari, Sobhan Siamak, “Artificial Control of PUMA Robot Manipulator: A-Review of Fuzzy Inference Engine And Application to Classical Controller ,” International Journal of Robotics and Automation, 2 (5):401-425, 2011.

[25] Gao W., Hung J. C.: Variable Structure Control of Nonlinear Systems: A New Approach, IEEE Transactions on Industrial Electronics , vol. 40, pp.45-56, 1993.

[26] Jakub Mo˙zaryn, Jerzy E. Kurek.: DESIGN OF THE SLIDING MODE CONTROL FOR THE PUMA 560 ROBOT, 1Institute of Automatic Control and Robotics, Warsaw University of Technology, ul. ´Sw. Andrzeja Boboli 8, 02-525, Warszawa, Poland.

[27] JAKUB MO˙ZARYN, JERZY E. KUREK Warsaw University of Technology, Institute of Automatic Control and Robotics, 02-525 Warsaw, POLAND.

[28] Y. Guo and P. Y. Woo, "An adaptive fuzzy sliding mode controller for robotic manipulators", Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, vol. 33, no. 2, (2003), pp. 149-159.

[29] Mulyana, "Two Wheeled Inverted Pendulum Balancing Control Using Sliding Mode Control and PD controle," Taiwan, University of Science and Technology, 2008.

[30] Claudio Vecchio, "Sliding Mode Control: Theoretical Developments and Applications to Uncertain Mechanical Systems," PhD Thesis, University of Degli Studi di Pavia, Italy.

[31] Duy-Ky Nguyen, " Basic Sliding Mode Controller Design: Advanced Design Techniques," PhD Thesis, University of Technology, Sydney, Australia,1998.

[32] Itkis, U., "Control Systems of Variable Structure," Wiley. New York, NY, USA, 1976.

[33] Ahmed Rhif, "Stabilizing Sliding Mode Control Design and Application for a DC Motor: Speed Control," International Journal of Instrumentation and Control Systems (IJICS) Vol.2, No.1, January 2012.

[34] M. Sami Fadali, " Sliding Mode Control," presentation, University of Nevada, Reno.

[35] Ahmad M.N., (2003). “Modelling and Control of Direct Drive Robot Manipulators”, Universe Technology Malaysia, PhD Thesis.

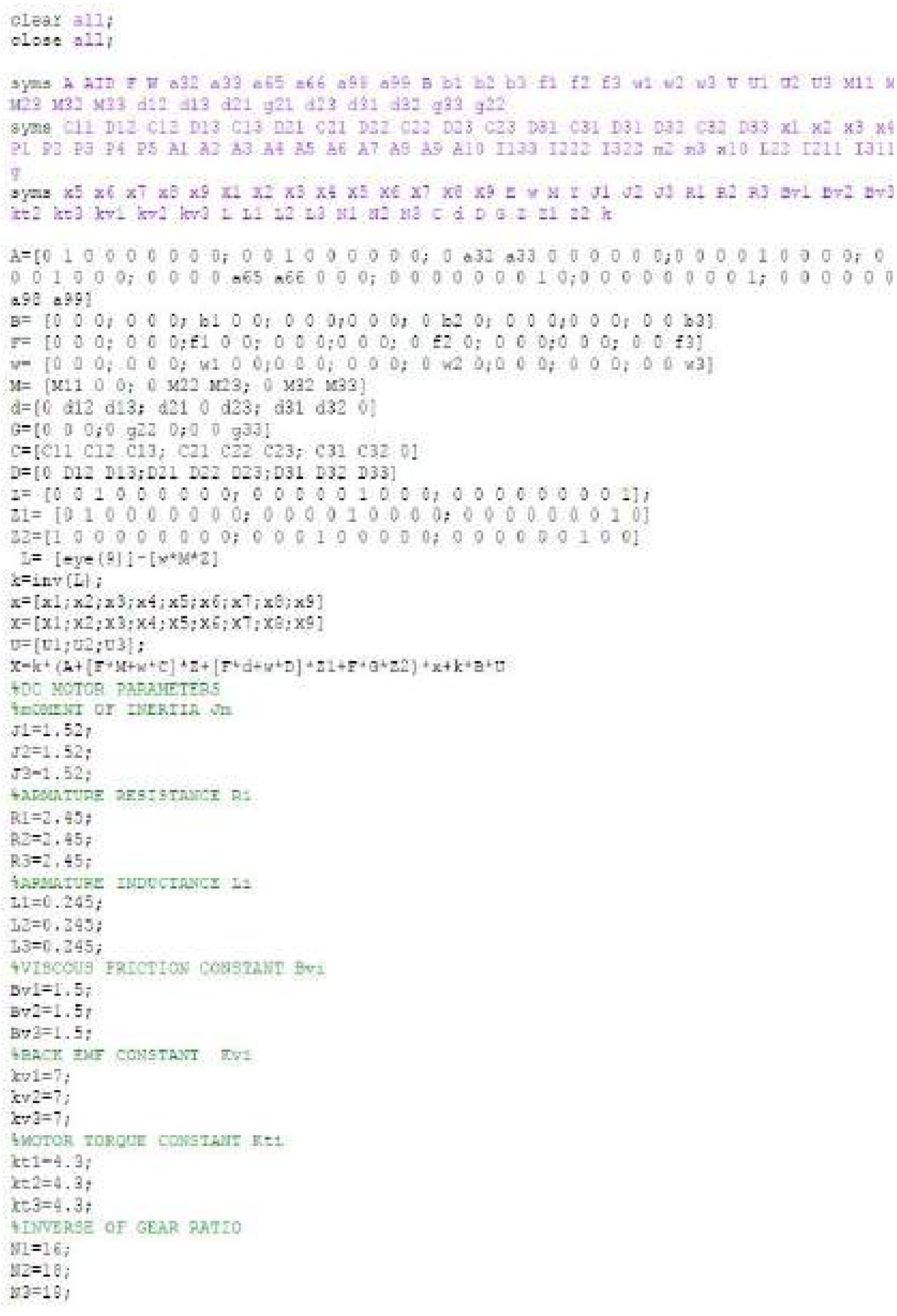
[36] Han, M.C., and Chen, Y.H., “Decentralized Control of Nonlinear Uncertain Systems with Bounded Uncertainties”, Proc. 30th. Conf Decision and Control, Brighton, England, pp. 321-326, 1991.

[37] Weng C. C., W. S. Yu. Adaptive fuzzy sliding mode control for linear time-varying uncertain systems, IEEE conference on fuzzy systems, 2008, P.P: 1483-1490.

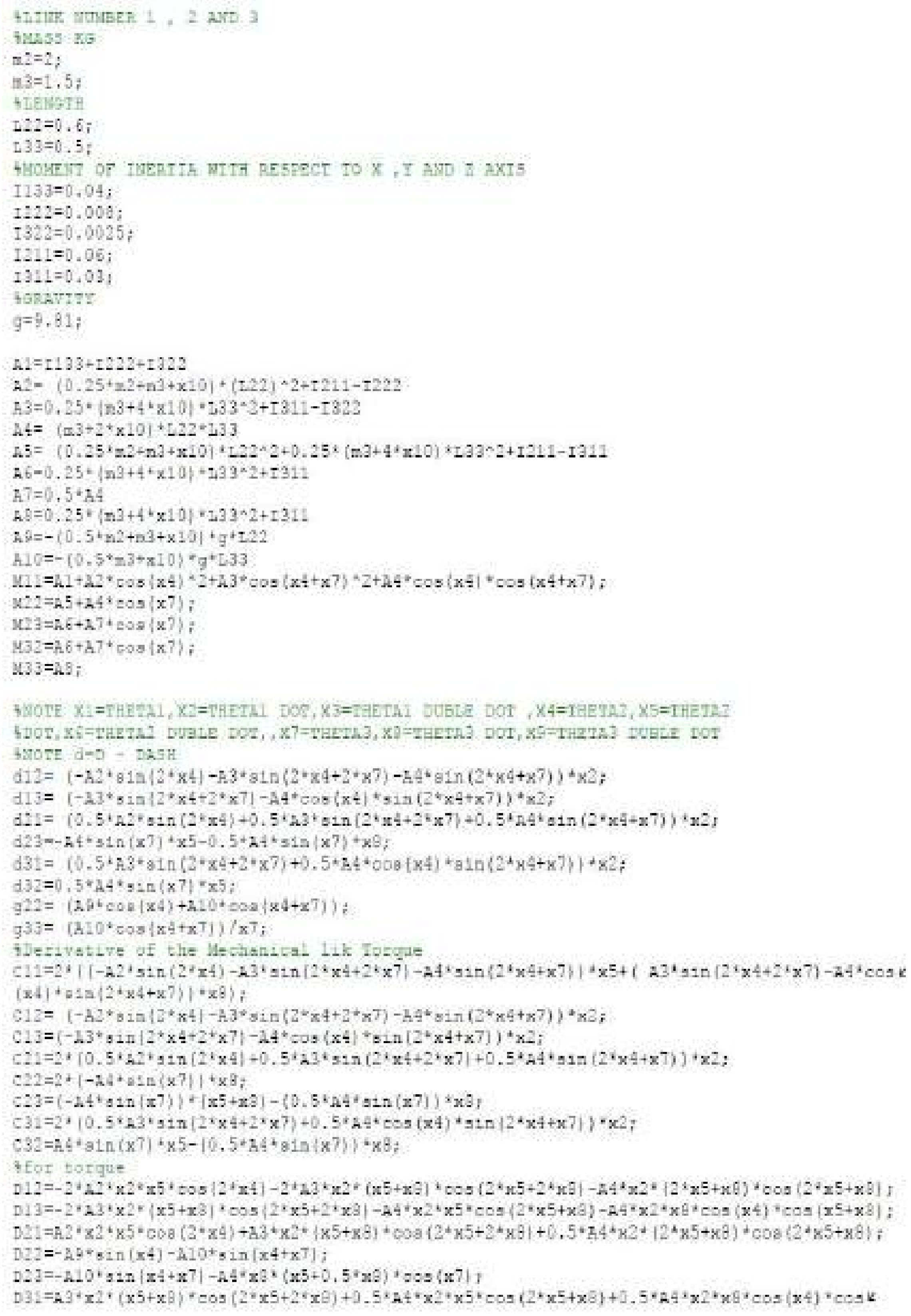
[38] An Adaptive sliding surface slope adjustment in PD Sliding Mode Fuzzy Control for Robot Manipulator, International Journal of Control and Automation Vol. 4 No. 3, September, 2011

[39] Osman, J.H.S.,“Decentralized and Hierarchical Control of Robot Manipulators.” City University, London, UK: PhD . 1991.

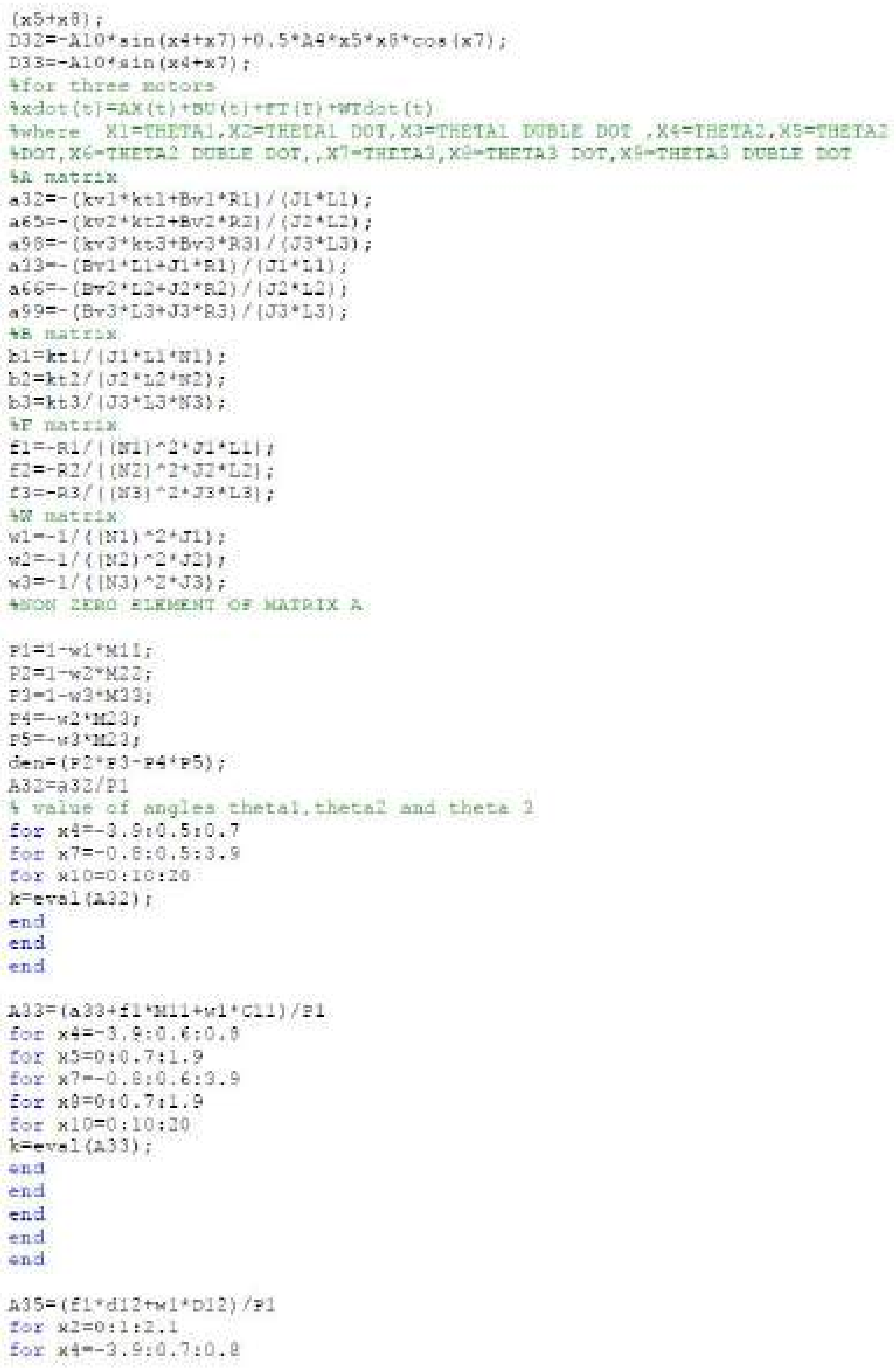
APPENDIX A:



```
clear all;
close all;

syms A AID F W a32 a33 a65 a66 a98 a99 B b1 b2 b3 f1 f2 f3 w1 w2 w3 U U1 U2 U3 M11 M
M23 M32 M33 d12 d13 d21 g21 d23 d31 d32 g33 g22
syms C11 D12 C12 D13 C13 D21 C21 D22 C22 D23 C23 D81 C31 D31 D32 C32 D33 x1 x2 x3 x4
P1 P2 P3 P4 P5 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 I133 I222 I322 m2 m3 x10 L22 I211 I311
q
syms x5 x6 x7 x8 x9 X1 X2 X3 X4 X5 X6 X7 X8 X9 E w N r J1 J2 J3 R1 R2 R3 Bv1 Bv2 Bv3
kt2 kt3 kv1 kv2 kv3 L L1 L2 L3 N1 N2 N3 C d D G z Z1 Z2 k

A=[0 1 0 0 0 0 0 0 0; 0 0 1 0 0 0 0 0 0; 0 a32 a33 0 0 0 0 0 0;0 0 0 0 1 0 0 0 0; 0
0 0 1 0 0 0; 0 0 0 0 a65 a66 0 0 0; 0 0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 0 1; 0 0 0 0 0 0
a98 a99]
B= [0 0 0; 0 0 0; b1 0 0; 0 0 0;0 0 0; 0 b2 0; 0 0 0;0 0 0; 0 0 b3]
F= [0 0 0; 0 0 0;f1 0 0; 0 0 0;0 0 0; 0 f2 0; 0 0 0;0 0 0; 0 0 f3]
w= [0 0 0; 0 0 0; w1 0 0;0 0 0; 0 0 0; 0 w2 0;0 0 0; 0 0 0; 0 0 w3]
M= [M11 0 0; 0 M22 M23; 0 M32 M33]
d=[0 d12 d13; d21 0 d23; d31 d32 0]
G=[0 0 0;0 g22 0;0 0 g33]
C=[C11 C12 C13; C21 C22 C23; C31 C32 0]
D=[0 D12 D13;D21 D22 D23;D31 D32 D33]
z= [0 0 1 0 0 0 0 0 0; 0 0 0 0 0 1 0 0 0; 0 0 0 0 0 0 0 0 1];
Z1= [0 1 0 0 0 0 0 0 0; 0 0 0 0 1 0 0 0 0; 0 0 0 0 0 0 0 1 0]
Z2=[1 0 0 0 0 0 0 0 0; 0 0 0 1 0 0 0 0 0; 0 0 0 0 0 0 1 0 0]
 L= [eye(9)]-[w*M*Z]
k=inv(L);
x=[x1;x2;x3;x4;x5;x6;x7;x8;x9]
X=[X1;X2;X3;X4;X5;X6;X7;X8;X9]
U=[U1;U2;U3];
X=k*(A+[F*M+w*C]*Z+[F*d+w*D]*Z1+F*G*Z2)*x+k*B*U
%DC MOTOR PARAMETERS
%mOMENT OF INERTIA Jm
J1=1.52;
J2=1.52;
J3=1.52;
%ARMATURE RESISTANCE Ri
R1=2.45;
R2=2.45;
R3=2.45;
%ARMATURE INDUCTANCE Li
L1=0.245;
L2=0.245;
L3=0.245;
%VISCOUS FRICTION CONSTANT Bvi
Bv1=1.5;
Bv2=1.5;
Bv3=1.5;
%BACK EMF CONSTANT  Kvi
kv1=7;
kv2=7;
kv3=7;
%MOTOR TORQUE CONSTANT Kti
kt1=4.3;
kt2=4.3;
kt3=4.3;
%INVERSE OF GEAR RATIO
N1=16;
N2=18;
N3=18;
```

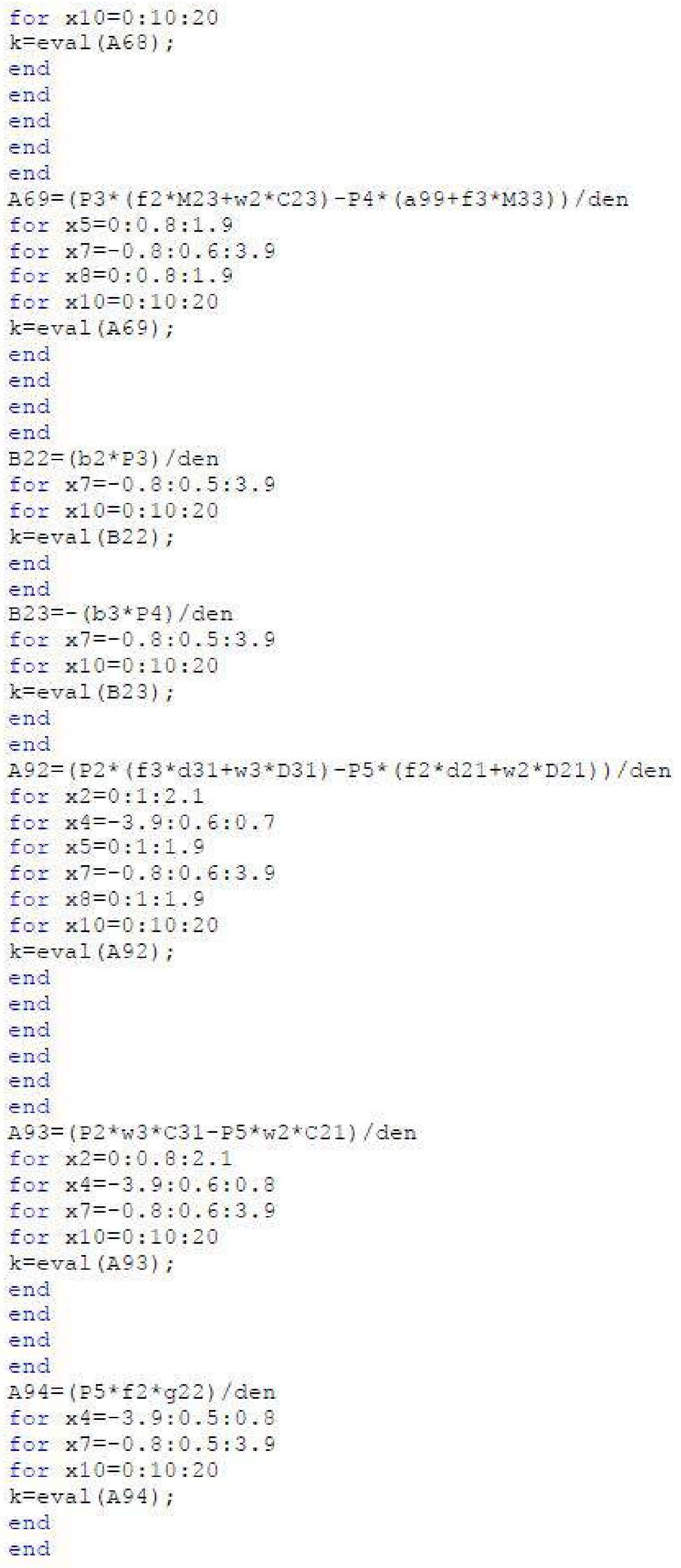


```
%LINK NUMBER 1 , 2 AND 3
%MASS KG
m2=2;
m3=1.5;
%LENGTH
L22=0.6;
L33=0.5;
%MOMENT OF INERTIA WITH RESPECT TO X ,Y AND Z AXIS
I133=0.04;
I222=0.008;
I322=0.0025;
I211=0.06;
I311=0.03;
%GRAVITY
g=9.81;

A1=I133+I222+I322
A2= (0.25*m2+m3+x10)*(L22)^2+I211-I222
A3=0.25*(m3+4*x10)*L33^2+I311-I322
A4= (m3+2*x10)*L22*L33
A5= (0.25*m2+m3+x10)*L22^2+0.25*(m3+4*x10)*L33^2+I211-I311
A6=0.25*(m3+4*x10)*L33^2+I311
A7=0.5*A4
A8=0.25*(m3+4*x10)*L33^2+I311
A9=-(0.5*m2+m3+x10)*g*L22
A10=-(0.5*m3+x10)*g*L33
M11=A1+A2*cos(x4)^2+A3*cos(x4+x7)^2+A4*cos(x4)*cos(x4+x7);
M22=A5+A4*cos(x7);
M23=A6+A7*cos(x7);
M32=A6+A7*cos(x7);
M33=A8;

%NOTE X1=THETA1,X2=THETA1 DOT,X3=THETA1 DUBLE DOT ,X4=THETA2,X5=THETA2
%DOT,X6=THETA2 DUBLE DOT,,X7=THETA3,X8=THETA3 DOT,X9=THETA3 DUBLE DOT
%NOTE d=D - DASH
d12= (-A2*sin(2*x4)-A3*sin(2*x4+2*x7)-A4*sin(2*x4+x7))*x2;
d13= (-A3*sin(2*x4+2*x7)-A4*cos(x4)*sin(2*x4+x7))*x2;
d21= (0.5*A2*sin(2*x4)+0.5*A3*sin(2*x4+2*x7)+0.5*A4*sin(2*x4+x7))*x2;
d23=-A4*sin(x7)*x5-0.5*A4*sin(x7)*x8;
d31= (0.5*A3*sin(2*x4+2*x7)+0.5*A4*cos(x4)*sin(2*x4+x7))*x2;
d32=0.5*A4*sin(x7)*x5;
g22= (A9*cos(x4)+A10*cos(x4+x7));
g33= (A10*cos(x4+x7))/x7;
%Derivative of the Mechanical lik Torque
C11=2*((-A2*sin(2*x4)-A3*sin(2*x4+2*x7)-A4*sin(2*x4+x7))*x5+( A3*sin(2*x4+2*x7)-A4*cos↙
(x4)*sin(2*x4+x7))*x8);
C12= (-A2*sin(2*x4)-A3*sin(2*x4+2*x7)-A4*sin(2*x4+x7))*x2;
C13=(-A3*sin(2*x4+2*x7)-A4*cos(x4)*sin(2*x4+x7))*x2;
C21=2*(0.5*A2*sin(2*x4)+0.5*A3*sin(2*x4+2*x7)+0.5*A4*sin(2*x4+x7))*x2;
C22=2*(-A4*sin(x7))*x8;
C23=(-A4*sin(x7))*(x5+x8)-(0.5*A4*sin(x7))*x8;
C31=2*(0.5*A3*sin(2*x4+2*x7)+0.5*A4*cos(x4)*sin(2*x4+x7))*x2;
C32=A4*sin(x7)*x5-(0.5*A4*sin(x7))*x8;
%for torque
D12=-2*A2*x2*x5*cos(2*x4)-2*A3*x2*(x5+x8)*cos(2*x5+2*x8)-A4*x2*(2*x5+x8)*cos(2*x5+x8);
D13=-2*A3*x2*(x5+x8)*cos(2*x5+2*x8)-A4*x2*x5*cos(2*x5+x8)-A4*x2*x8*cos(x4)*cos(x5+x8);
D21=A2*x2*x5*cos(2*x4)+A3*x2*(x5+x8)*cos(2*x5+2*x8)+0.5*A4*x2*(2*x5+x8)*cos(2*x5+x8);
D22=-A9*sin(x4)-A10*sin(x4+x7);
D23=-A10*sin(x4+x7)-A4*x8*(x5+0.5*x8)*cos(x7);
D31=A3*x2*(x5+x8)*cos(2*x5+2*x8)+0.5*A4*x2*x5*cos(2*x5+x8)+0.5*A4*x2*x8*cos(x4)*cos↙
```

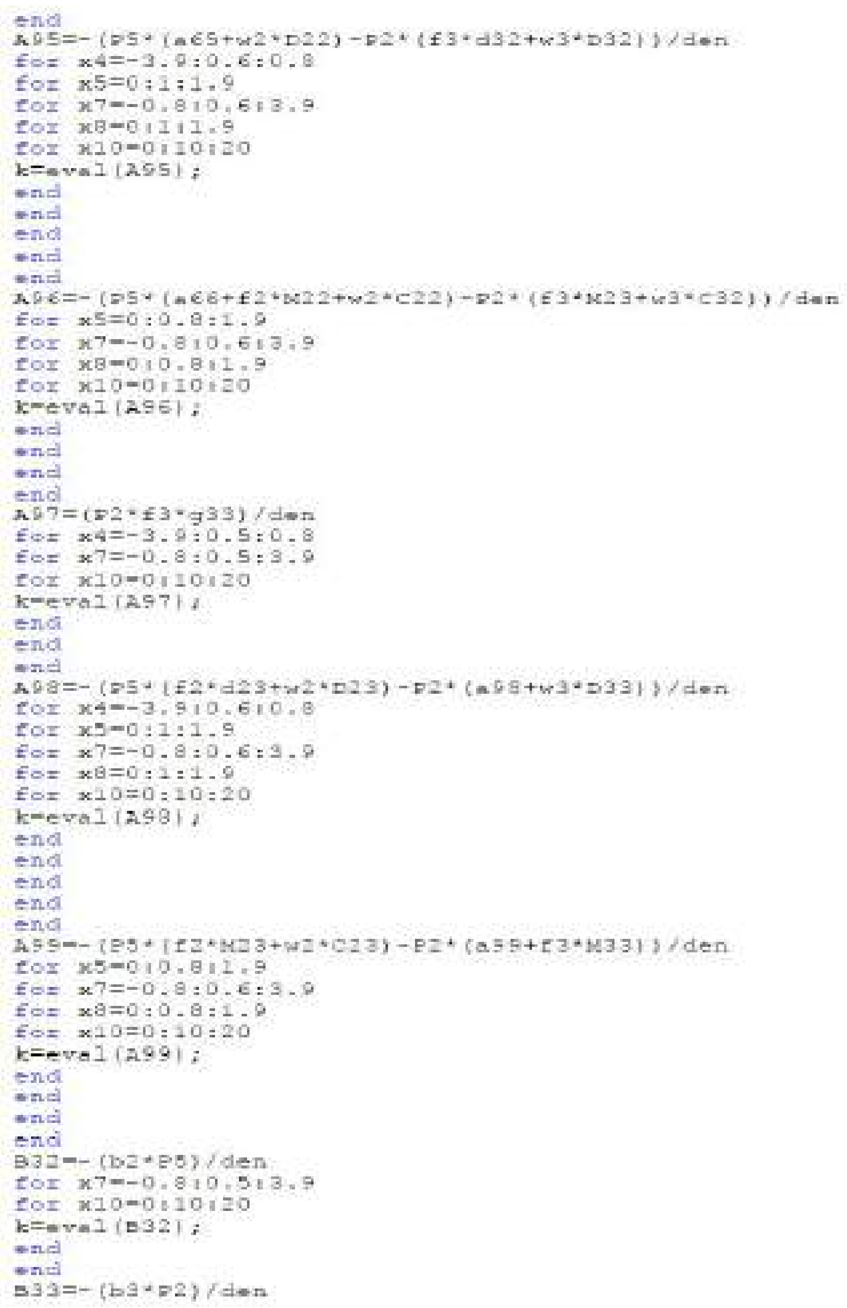


```
(x5+x8);
D32=-A10*sin(x4+x7)+0.5*A4*x5*x8*cos(x7);
D33=-A10*sin(x4+x7);
%for three motors
%xdot(t)=AX(t)+BU(t)+FT(T)+WTdot(t)
%where  X1=THETA1,X2=THETA1 DOT,X3=THETA1 DUBLE DOT ,X4=THETA2,X5=THETA2
%DOT,X6=THETA2 DUBLE DOT,,X7=THETA3,X8=THETA3 DOT,X9=THETA3 DUBLE DOT
%A matrix
a32=-(kv1*kt1+Bv1*R1)/(J1*L1);
a65=-(kv2*kt2+Bv2*R2)/(J2*L2);
a98=-(kv3*kt3+Bv3*R3)/(J3*L3);
a33=-(Bv1*L1+J1*R1)/(J1*L1);
a66=-(Bv2*L2+J2*R2)/(J2*L2);
a99=-(Bv3*L3+J3*R3)/(J3*L3);
%B matrix
b1=kt1/(J1*L1*N1);
b2=kt2/(J2*L2*N2);
b3=kt3/(J3*L3*N3);
%F matrix
f1=-R1/((N1)^2*J1*L1);
f2=-R2/((N2)^2*J2*L2);
f3=-R3/((N3)^2*J3*L3);
%W matrix
w1=-1/((N1)^2*J1);
w2=-1/((N2)^2*J2);
w3=-1/((N3)^2*J3);
%NON ZERO ELEMENT OF MATRIX A

P1=1-w1*M11;
P2=1-w2*M22;
P3=1-w3*M33;
P4=-w2*M23;
P5=-w3*M23;
den=(P2*P3-P4*P5);
A32=a32/P1
% value of angles theta1,theta2 and theta 2
for x4=-3.9:0.5:0.7
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(A32);
end
end
end

A33=(a33+f1*M11+w1*C11)/P1
for x4=-3.9:0.6:0.8
for x5=0:0.7:1.9
for x7=-0.8:0.6:3.9
for x8=0:0.7:1.9
for x10=0:10:20
k=eval(A33);
end
end
end
end
end

A35=(f1*d12+w1*D12)/P1
for x2=0:1:2.1
for x4=-3.9:0.7:0.8
for x5=0:1:1.9
for x7=-0.8:0.7:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A35);
end
end
end
end
end
end
A36=(w1*C12)/P1
for x2=0:0.7:2.1
for x4=-3.9:0.6:0.8
for x7=-0.8:0.6:3.9
for x10=0:10:20
k=eval(A36);
end
end
end
end
A38=(f1*d13+w1*D13)/P1
for x2=0:1:2.1
for x4=-3.9:0.7:0.8
for x5=0:1:1.9
for x7=-0.8:0.7:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A38);
end
end
end
end
end
end
A39=(w1*C13)/P1
for x2=0:0.7:2.1
for x4=-3.9:0.6:0.8
for x7=-0.8:0.6:3.9
for x10=0:10:20
k=eval(A39);
end
end
end
end
B11=b1/P1
for x4=-3.9:0.5:0.8
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(B11);
end
end
end
A62=(P3*(f2*d21+w2*D21)-P4*(f3*d31+w3*D31))/den
for x2=0:1:2.1
for x4=-3.9:0.7:0.8
for x5=0:1:1.9
for x7=-0.8:0.7:3.9
for x8=0:1:1.9
for x10=0:10:20
```

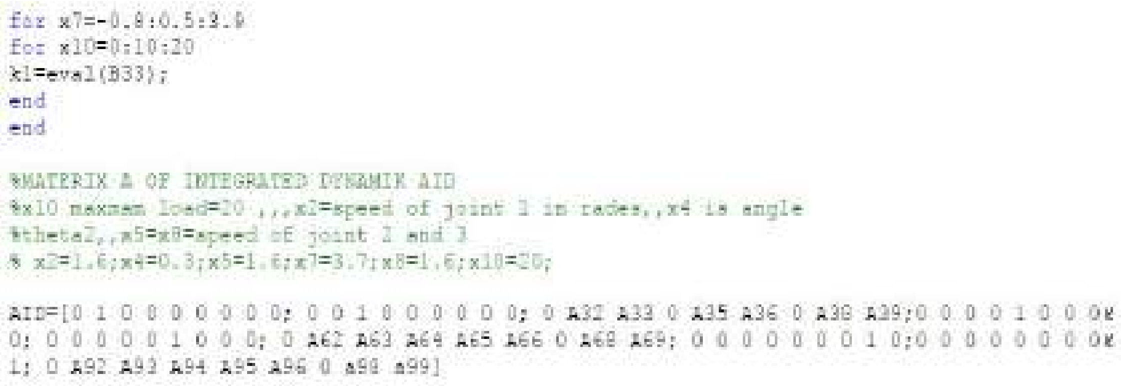
```
k=eval(A62);
end
end
end
end
end
end
A63=(P3*w2*C21-P4*w3*C31)/den
for x2=0:0.7:2.1
for x4=-3.9:0.6:0.8
for x7=-0.8:0.6:3.9
for x10=0:10:20
k=eval(A63);
end
end
end
end
A64=(P3*f2*g22)/(x4*den)
for x4=-3.9:0.5:0.8
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(A64);
end
end
end
A65=(P3*(a65+w2*D22)-P4*(f3*d32+w3*D32))/den
for x4=-3.9:0.6:0.8
for x5=0:1:1.9
for x7=-0.8:0.6:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A65);
end
end
end
end
end
A66=(P3*(a66+f2*M22+w2*C22)-P4*(f3*M23+w3*C32))/den
for x5=0:0.8:1.9
for x7=0.8:0.6:3.9
for x8=0:0.8:1.9
for x10=0:10:20
k=eval(A66);
end
end
end
end
A67=-(P4*f3*g33)/den
for x4=-3.9:0.5:0.78
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(A67);
end
end
end
A68=(P3*(f2*d23+w2*D23)-P4*(a98+w3*D33))/den
for x4=-3.9:0.6:0.7
for x5=0:1:1.9
for x7=-0.8:0.6:3.9
for x8=0:1:1.9
```



```
for x10=0:10:20
k=eval(A68);
end
end
end
end
end
A69=(P3*(f2*M23+w2*C23)-P4*(a99+f3*M33))/den
for x5=0:0.8:1.9
for x7=-0.8:0.6:3.9
for x8=0:0.8:1.9
for x10=0:10:20
k=eval(A69);
end
end
end
end
B22=(b2*P3)/den
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(B22);
end
end
B23=-(b3*P4)/den
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(B23);
end
end
A92=(P2*(f3*d31+w3*D31)-P5*(f2*d21+w2*D21))/den
for x2=0:1:2.1
for x4=-3.9:0.6:0.7
for x5=0:1:1.9
for x7=-0.8:0.6:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A92);
end
end
end
end
end
end
A93=(P2*w3*C31-P5*w2*C21)/den
for x2=0:0.8:2.1
for x4=-3.9:0.6:0.8
for x7=-0.8:0.6:3.9
for x10=0:10:20
k=eval(A93);
end
end
end
end
A94=(P5*f2*g22)/den
for x4=-3.9:0.5:0.8
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(A94);
end
end
```

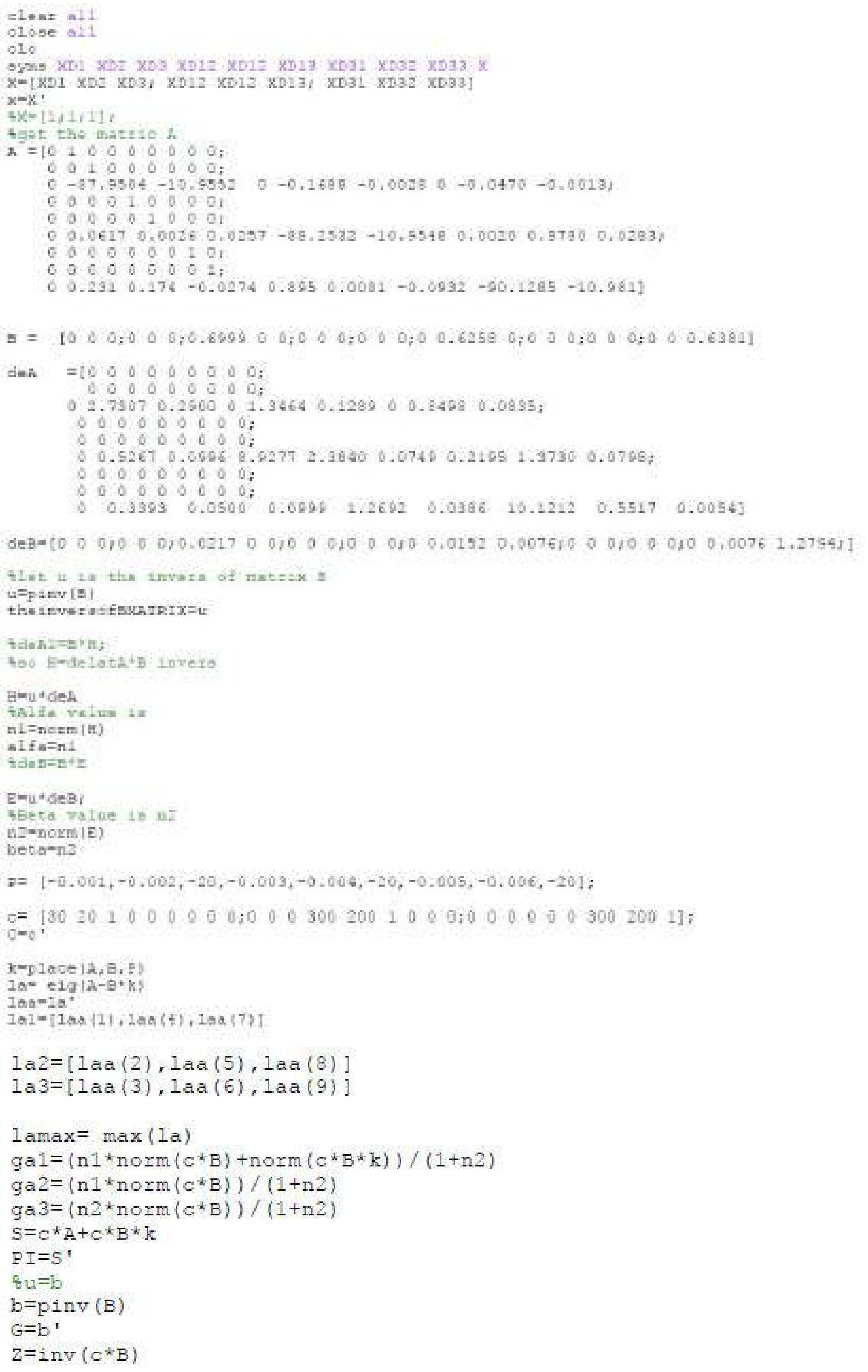


```
end
A95=-(P5*(a65+w2*D22)-P2*(f3*d32+w3*D32))/den
for x4=-3.9:0.6:0.8
for x5=0:1:1.9
for x7=-0.8:0.6:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A95);
end
end
end
end
end
A96=-(P5*(a66+f2*M22+w2*C22)-P2*(f3*M23+w3*C32))/den
for x5=0:0.8:1.9
for x7=-0.8:0.6:3.9
for x8=0:0.8:1.9
for x10=0:10:20
k=eval(A96);
end
end
end
end
A97=(P2*f3*g33)/den
for x4=-3.9:0.5:0.8
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(A97);
end
end
end
A98=-(P5*(f2*d23+w2*D23)-P2*(a98+w3*D33))/den
for x4=-3.9:0.6:0.8
for x5=0:1:1.9
for x7=-0.8:0.6:3.9
for x8=0:1:1.9
for x10=0:10:20
k=eval(A98);
end
end
end
end
end
A99=-(P5*(f2*M23+w2*C23)-P2*(a99+f3*M33))/den
for x5=0:0.8:1.9
for x7=-0.8:0.6:3.9
for x8=0:0.8:1.9
for x10=0:10:20
k=eval(A99);
end
end
end
end
B32=-(b2*P5)/den
for x7=-0.8:0.5:3.9
for x10=0:10:20
k=eval(B32);
end
end
B33=-(b3*P2)/den
```



```
for x7=-0.8:0.5:3.9
for x10=0:10:20
k1=eval(B33);
end
end

%MATERIX A OF INTEGRATED DYNAMIK AID
%x10 maxmam load=20 ,,,x2=speed of joint 1 in rades,,x4 is angle
%theta2,,x5=x8=speed of joint 2 and 3
% x2=1.6;x4=0.3;x5=1.6;x7=3.7;x8=1.6;x10=20;

AID=[0 1 0 0 0 0 0 0 0; 0 0 1 0 0 0 0 0 0; 0 A32 A33 0 A35 A36 0 A38 A39;0 0 0 0 1 0 0 0 0↙
0; 0 0 0 0 0 1 0 0 0; 0 A62 A63 A64 A65 A66 0 A68 A69; 0 0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 0 0↙
1; 0 A92 A93 A94 A95 A96 0 a98 a99]
```

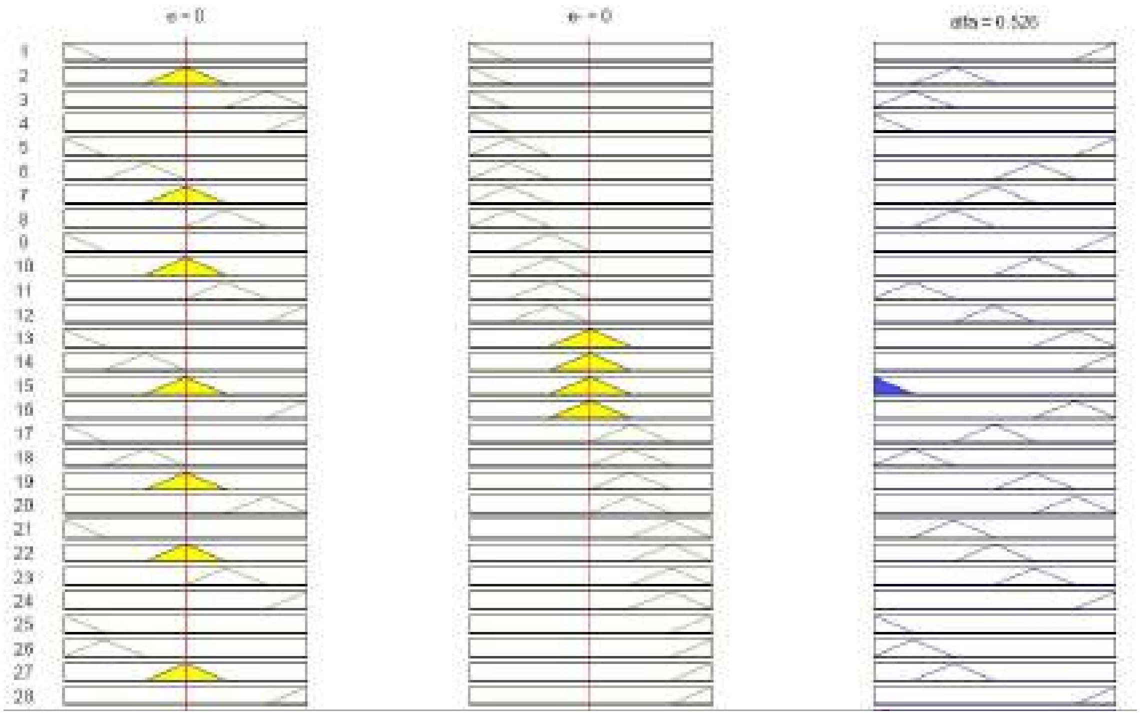
APPENDIX B:



```
clear all
close all
clc
syms XD1 XD2 XD3 XD12 XD12 XD13 XD31 XD32 XD33 X
X=[XD1 XD2 XD3; XD12 XD12 XD13; XD31 XD32 XD33]
x=X'
%X=[1;1;1];
%get the matric A
A =[0 1 0 0 0 0 0 0 0;
    0 0 1 0 0 0 0 0 0;
    0 -87.9504 -10.9552  0 -0.1688 -0.0028 0 -0.0470 -0.0013;
    0 0 0 0 1 0 0 0 0;
    0 0 0 0 0 1 0 0 0;
    0 0.0617 0.0026 0.0257 -88.2532 -10.9548 0.0020 0.9780 0.0283;
    0 0 0 0 0 0 0 1 0;
    0 0 0 0 0 0 0 0 1;
    0 0.231 0.174 -0.0274 0.895 0.0081 -0.0932 -90.1285 -10.981]


B =  [0 0 0;0 0 0;0.6999 0 0;0 0 0;0 0 0;0 0.6258 0;0 0 0;0 0 0;0 0 0.6381]

deA   =[0 0 0 0 0 0 0 0 0;
        0 0 0 0 0 0 0 0 0;
      0 2.7307 0.2900 0 1.3464 0.1289 0 0.8498 0.0835;
       0 0 0 0 0 0 0 0 0;
       0 0 0 0 0 0 0 0 0;
       0 0.5267 0.0996 8.9277 2.3840 0.0749 0.2195 1.3730 0.0798;
       0 0 0 0 0 0 0 0 0;
       0 0 0 0 0 0 0 0 0;
       0  0.3393  0.0500  0.0999  1.2692  0.0386  10.1212  0.5517  0.0054]

deB=[0 0 0;0 0 0;0.0217 0 0;0 0 0;0 0 0;0 0.0152 0.0076;0 0 0;0 0 0;0 0.0076 1.2796;]

%let u is the invers of matrix B
u=pinv(B)
theinverseofBMATRIX=u

%deA1=B*H;
%so H=delatA*B invers

H=u*deA
%Alfa value is
n1=norm(H)
alfa=n1
%deB=B*E

E=u*deB;
%Beta value is n2
n2=norm(E)
beta=n2

P= [-0.001,-0.002,-20,-0.003,-0.004,-20,-0.005,-0.006,-20];

c= [30 20 1 0 0 0 0 0 0;0 0 0 300 200 1 0 0 0;0 0 0 0 0 0 300 200 1];
C=c'

k=place(A,B,P)
la= eig(A-B*k)
laa=la'
la1=[laa(1),laa(4),laa(7)]

la2=[laa(2),laa(5),laa(8)]
la3=[laa(3),laa(6),laa(9)]

lamax= max(la)
ga1=(n1*norm(c*B)+norm(c*B*k))/(1+n2)
ga2=(n1*norm(c*B))/(1+n2)
ga3=(n2*norm(c*B))/(1+n2)
S=c*A+c*B*k
PI=S'
%u=b
b=pinv(B)
G=b'
Z=inv(c*B)
```

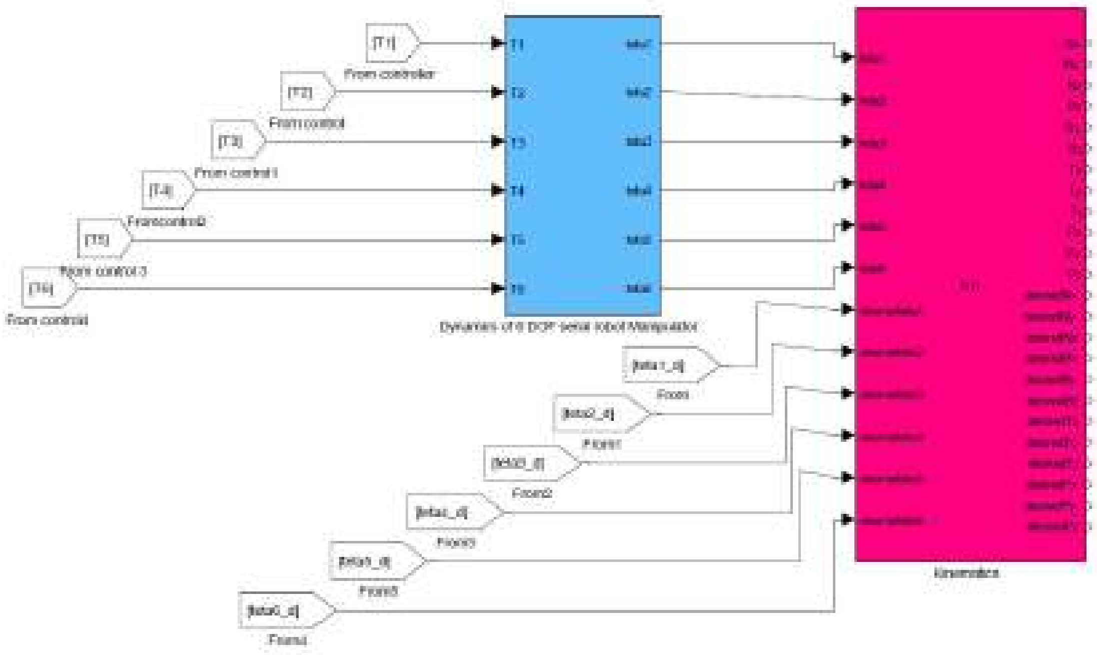
APPENDIX C

1-Rule Base Viewer after reducing:

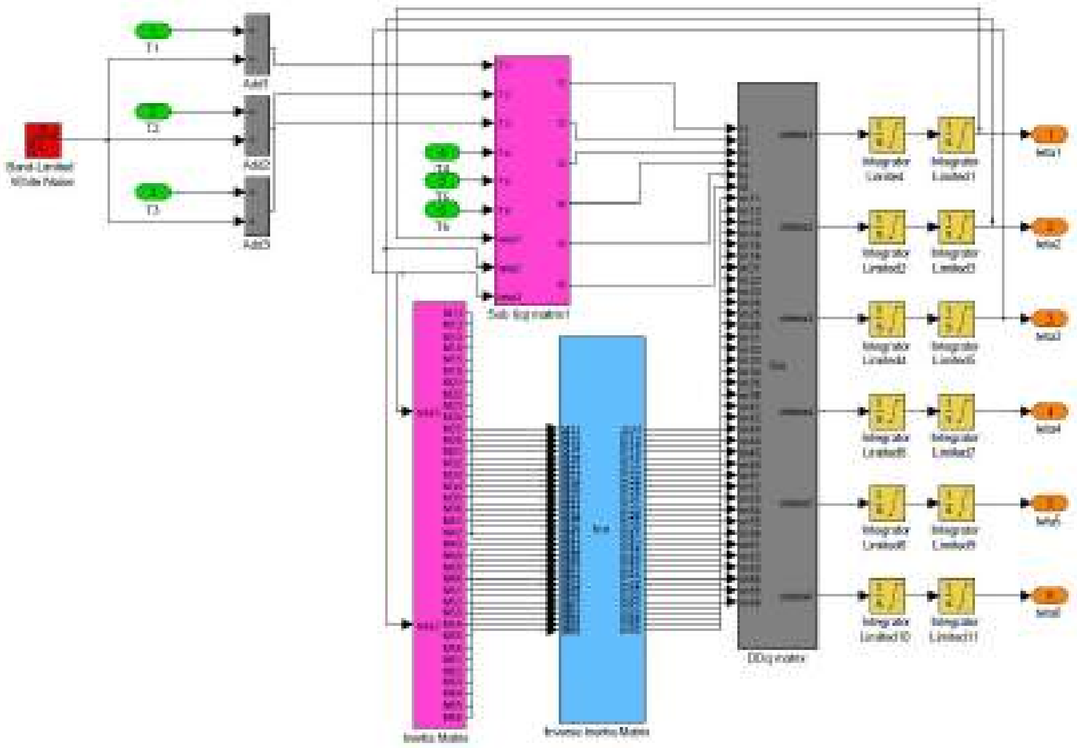


APPENDIX D

1. Kinematic and Dynamic Part



2-Dynamic Part:



Appendix

The dynamic formulations of six Degrees of Freedom serial links PUMA robot manipulator are computed by:

$$A(\ddot{\theta})\begin{bmatrix}\ddot{\theta}_1\\ \ddot{\theta}_2\\ \ddot{\theta}_3\\ \ddot{\theta}_4\\ \ddot{\theta}_5\\ \ddot{\theta}_6\end{bmatrix}+\mathrm{B}(\theta)\begin{bmatrix}\dot{\theta}_1\dot{\theta}_2\\ \dot{\theta}_1\dot{\theta}_3\\ \dot{\theta}_1\dot{\theta}_4\\ \dot{\theta}_1\dot{\theta}_5\\ \dot{\theta}_1\dot{\theta}_6\\ \dot{\theta}_2\dot{\theta}_3\\ \dot{\theta}_2\dot{\theta}_4\\ \dot{\theta}_2\dot{\theta}_5\\ \dot{\theta}_2\dot{\theta}_6\\ \dot{\theta}_3\dot{\theta}_4\\ \dot{\theta}_3\dot{\theta}_5\\ \dot{\theta}_3\dot{\theta}_6\\ \dot{\theta}_4\dot{\theta}_5\\ \dot{\theta}_4\dot{\theta}_6\\ \dot{\theta}_5\dot{\theta}_6\end{bmatrix}+C(\theta)\begin{bmatrix}\dot{\theta}_1^2\\ \dot{\theta}_2^2\\ \dot{\theta}_3^2\\ \dot{\theta}_4^2\\ \dot{\theta}_5^2\\ \dot{\theta}_6^2\end{bmatrix}+G(\theta)=\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\\ \tau_4\\ \tau_5\\ \tau_6\end{bmatrix}$$

$$A(q)=\begin{bmatrix}A_{11} & A_{12} & A_{13} & 0 & 0 & 0\\ A_{21} & A_{22} & A_{23} & 0 & 0 & 0\\ A_{31} & A_{32} & A_{33} & 0 & A_{35} & 0\\ 0 & 0 & 0 & A_{44} & 0 & 0\\ 0 & 0 & 0 & 0 & A_{55} & 0\\ 0 & 0 & 0 & 0 & 0 & A_{66}\end{bmatrix}$$

$$\mathrm{B}(\theta)\begin{bmatrix}\theta_1\theta_2\\ \dot{\theta}_1\dot{\theta}_3\\ \dot{\theta}_1\dot{\theta}_4\\ \dot{\theta}_1\dot{\theta}_5\\ \dot{\theta}_1\dot{\theta}_6\\ \dot{\theta}_2\dot{\theta}_3\\ \dot{\theta}_2\dot{\theta}_4\\ \dot{\theta}_2\dot{\theta}_5\\ \dot{\theta}_2\dot{\theta}_6\\ \dot{\theta}_3\dot{\theta}_4\\ \dot{\theta}_3\dot{\theta}_5\\ \dot{\theta}_3\dot{\theta}_6\\ \dot{\theta}_4\dot{\theta}_5\\ \dot{\theta}_4\dot{\theta}_6\\ \dot{\theta}_5\dot{\theta}_6\end{bmatrix}$$

$$b(q)=\begin{bmatrix} b_{112} & b_{113} & 0 & b_{115} & 0 & b_{123} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b_{214} & 0 & 0 & b_{223} & 0 & b_{225} & 0 & 0 & b_{235} & 0 & 0 & 0 & 0 \\ 0 & 0 & b_{314} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ b_{412} & b_{413} & 0 & b_{415} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b_{514} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{C(q)}=\begin{bmatrix} 0 & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{21} & 0 & C_{23} & 0 & 0 & 0 \\ C_{31} & C_{32} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ C_{51} & C_{52} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C(\theta)\begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \\ \dot{\theta}_3^2 \\ \dot{\theta}_4^2 \\ \dot{\theta}_5^2 \\ \dot{\theta}_6^2 \end{bmatrix}$$

$$[G(q)]_{6\times 1}=\begin{bmatrix} 0 \\ G_2 \\ G_3 \\ 0 \\ G_5 \\ 0 \end{bmatrix}$$

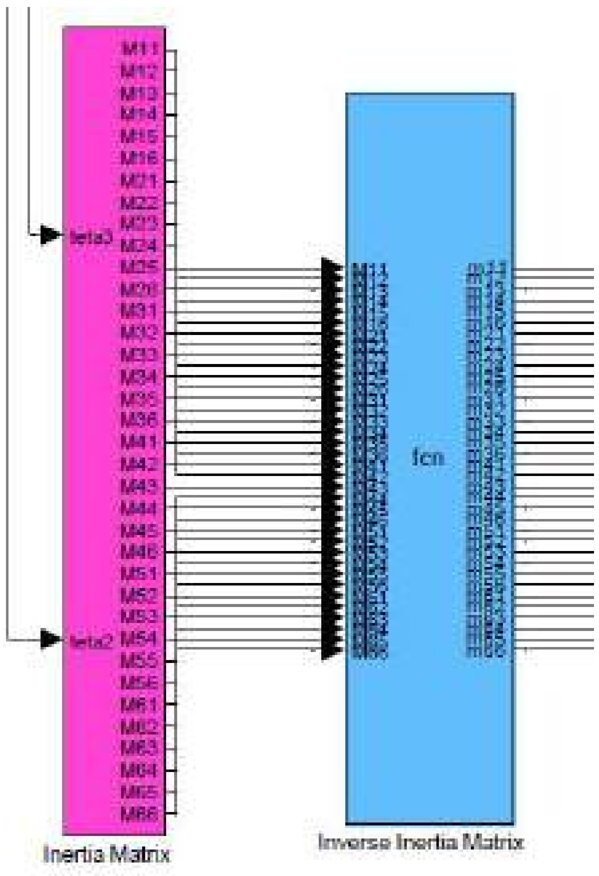
$$[I]_{6\times 1}=[B]_{6\times 1}+[C]_{6\times 1}+[G]_{6\times 1}$$

$$[K]_{6\times 1}=\{[\tau]_{6\times 1}-[I]_{6\times 1}\}$$

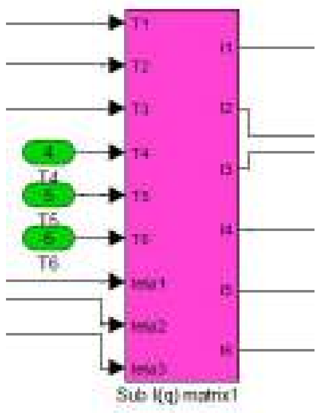
$$[\ddot{q}]_{6\times1} = [A^{-1}(q)]_{6\times6}(q) \times [K]_{6\times1}$$

$$[q]_{6\times1} = \iint [A^{-1}(q)]_{6\times6}(q) \times [K]_{6\times1}$$

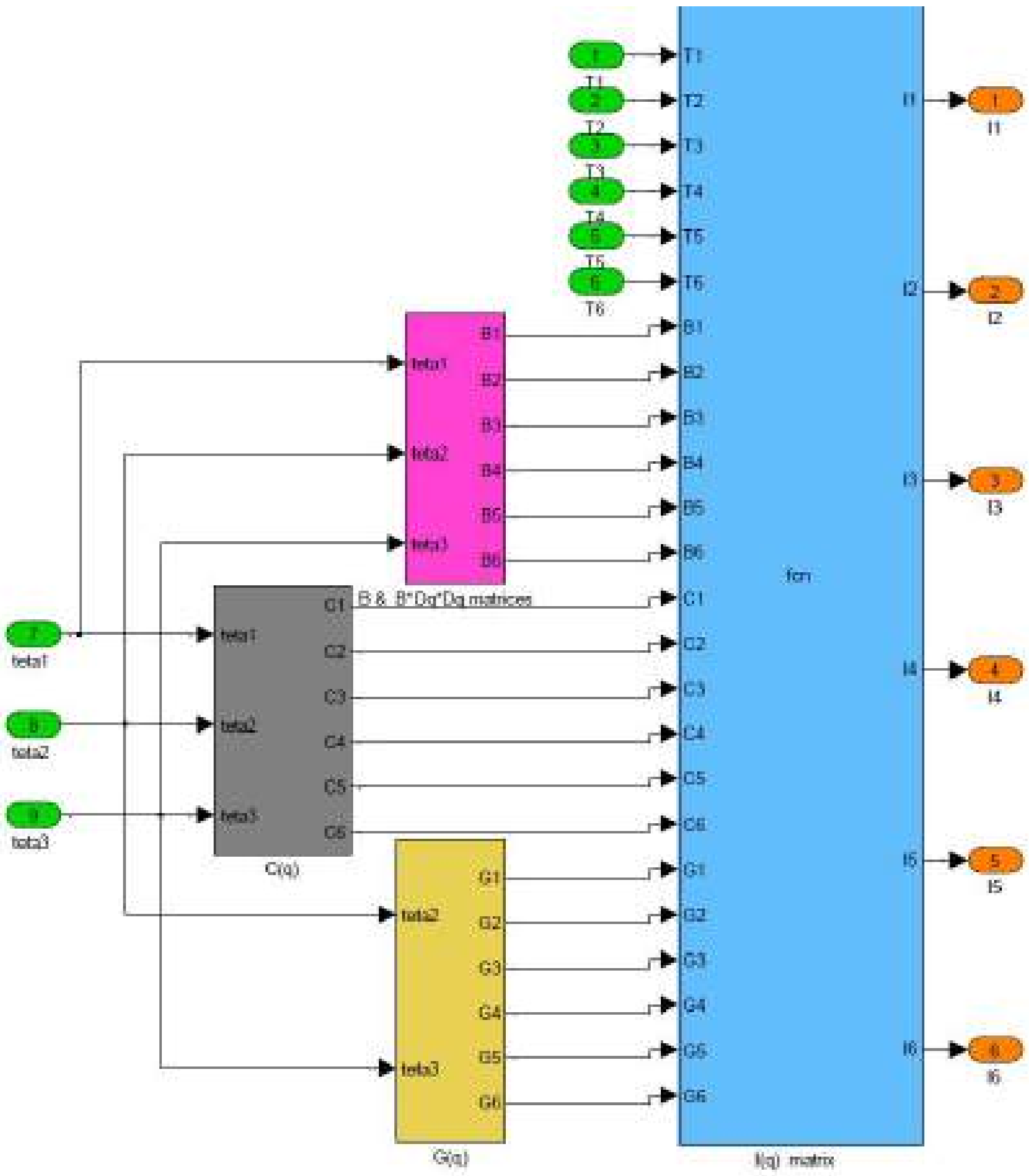
A-Inertia Matrix



B-Sub I(q) Matrix

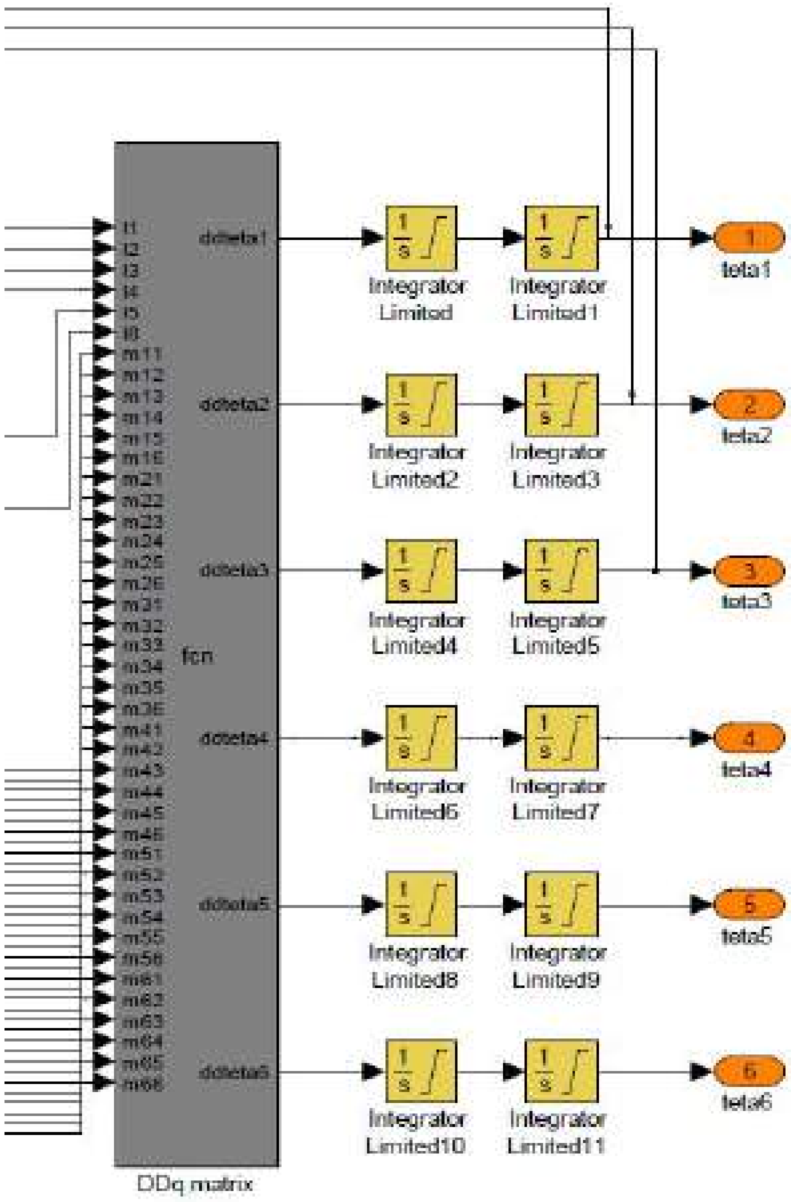


$$[I]_{6\times 1} = [B]_{6\times 1} + [C]_{6\times 1} + [G]_{6\times 1}$$

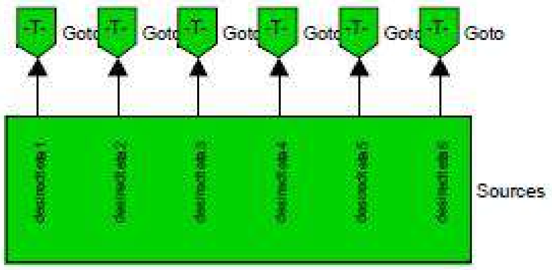


$$[\ddot{q}]_{6\times 1} = [A^{-1}(q)]_{6\times 6}(q) \times [K]_{6\times 1}$$

$$[q]_{6\times 1} = \iint [A^{-1}(q)]_{6\times 6}(q) \times [K]_{6\times 1}$$



Sources (Input)



2-SMC

This part is conventional SMC:

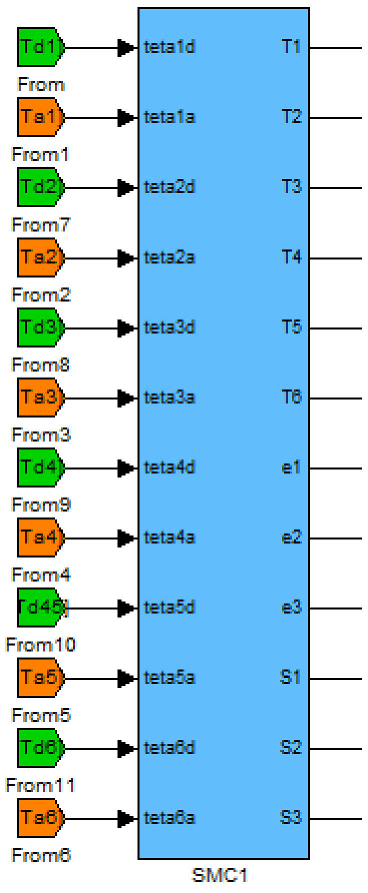
This controller has 12 inputs and 6 outputs.

Inputs are: theta 1 desired to theta 6 desired, and teta1 actual to theta 6 actual. The outputs are: Torque 1 to Torque 6 (T1-T6).

This controller has 2 parts:

1-equivalent is related to system dynamic.

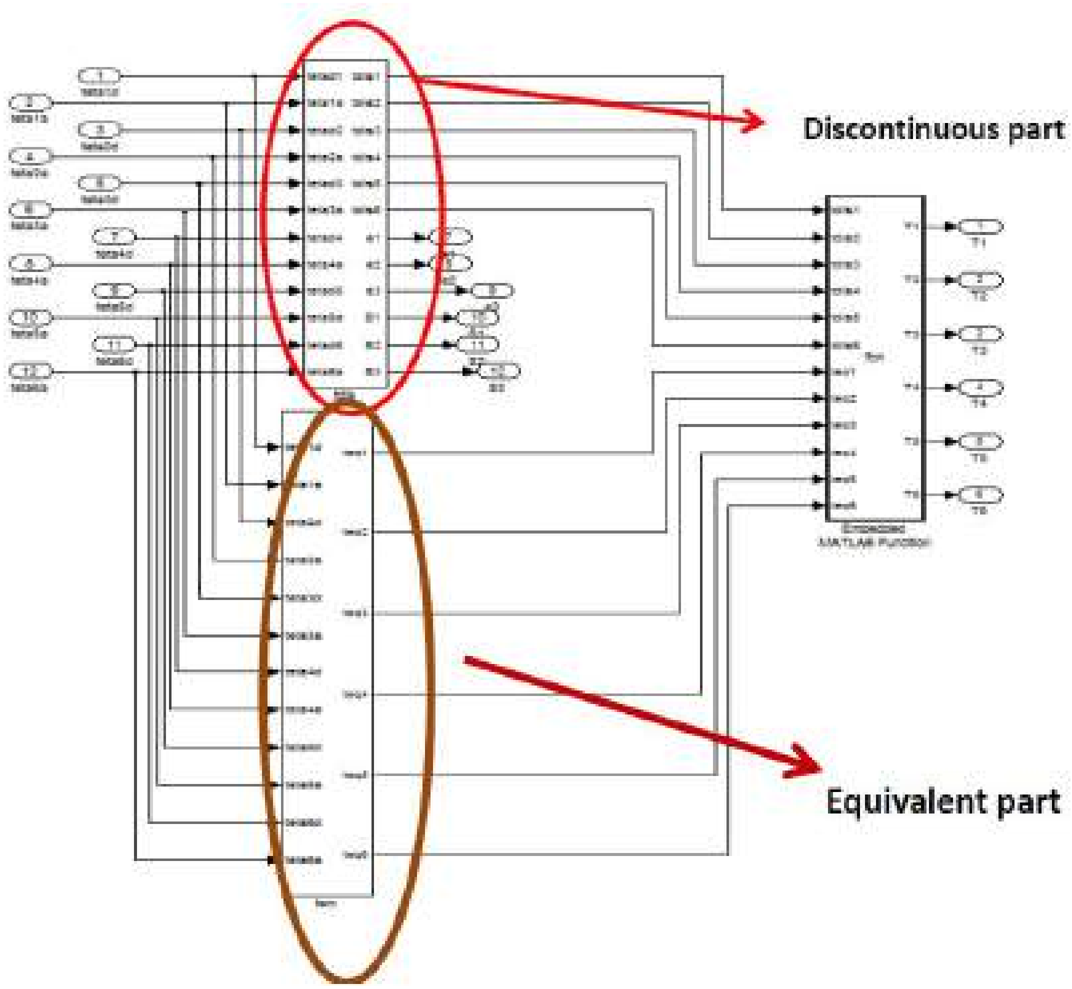
2-discontinuous part is related to (sign) function



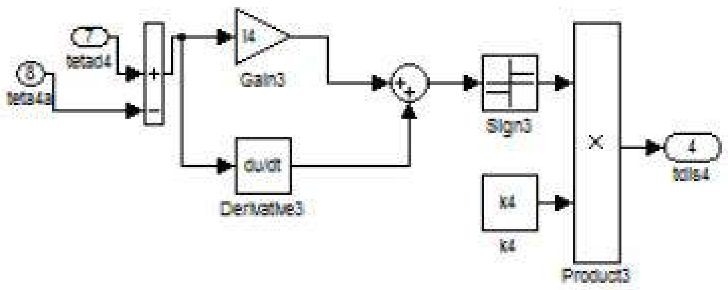
This controller has 2 parts:

1-Discontinuous part is related to (sign) function.

2-Equivalent is related to system dynamic



Discontinuous part is related to (sign) function.



Equivalent is related to system dynamic are: $\tau_{eq} = [M^{-1}(B + C + G) + S^{\cdot}]M$