Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Trajectory Tracking Control Simulation of a 4-DOF SCARA Robot Manipulator Using Fuzzy Sliding Mode Controller with PID Sliding Surface

A Thesis Submitted to Addis Ababa Institute of Technology, School of Graduate Studies, Addis Ababa University in Partial Fulfillment of the Requirement for the Degree of Master of Science in
Electrical and Computer Engineering (Control Engineering)

By

Temesgen Gelchu

Advisor: Dr. Dereje Shiferaw

ADDIS ABABA, ETHIOPIA

2017

Abstract

Research and development of controlling industrial robot is an active research area. The methodologies for controller design development are considered as the key for the robot in the industrial applications. One of the most widely used industrial robots is the SCARA robot due to its high precision, high speed, small dimensions, simple and reliable structure, and ease of installation. The study reported is on design and simulation of fuzzy sliding mode controller with PID sliding surface for a 4 DOF Selective Compliance Assembly Robot Arm (hereafter called SCARA).

The main objective of this thesis is to control a SCARA robot arm using two controllers to track the desired position. Sliding mode controller (SMC) is used as a reference benchmark to compare its result with fuzzy sliding mode controller (FSMC) with PID sliding surface result.

Euler – Lagrange approach has been used to drive the complete dynamic model of SCARA robot, the stability of the system has been analyzed by using Lyapunov method, the controller has been implemented using MATLAB and performance analysis has been done. The simulation results show that the newly proposed controller has removed chattering phenomena from input torque, minimized the magnitude of controller effort, and has reduced the tracking error (order of 10^{-5} rad *for first, second, fourth joint and of* 10^{-4} m *for third joint).*

Key words: SCARA robot, Fuzzy Control, Sliding Mode Control, and Fuzzy Sliding Mode Controller.

.

Acknowledgments

First of all I would like to thank my advisor Dr. Dereje Shiferaw for his help and support during this work. His knowledge and understanding of robotics, controls and fuzzy logic have aided my work greatly. Also his willingness to make time for me when I had a question and concern for me were greatly appreciated. He is truly a good man.

There are no words that describe how grateful I am to my parents for their support and encouragement through years.

Finally, my thank goes to my friends, especially Ashebr Netsanet, Zewde Tolossa, Tariku Anega and his family, and Addisie Belay.

Table of Contents

Lists of Figures

Lists of Tables

Lists of Abbreviations

PID ……. proportional–integral–derivative

DOF …….Degree of Freedom

SCARA …… Selective Compliance Assembly Robot Arm

DH …… Denavit Hartenberg

SMC …….. Sliding mode controller

FSMC …….. Fuzzy sliding mode controller

FLC …….. Fuzzy Logic Controller

PI ……..… Proportional Integral

PD ……..…… Proportional Derivative

Rot ……...……. Rotational

Trans ……..…….. Translational

N – E ……...……. Newton-Euler

E – L ……..…….. Euler Lagrange

L ……..……. Lagrangian

K ……..…….. Kinetic Energy

P ……..…….. Potential Energy

VSC …….……. Variable Structure Control

MIMO ……...……. Multiple in Multiple Out

B ……..……..… Coriolis Torque Matrix

C ……... Centrifugal Torque Matrix

G…… Gravity Force Vector

CHAPTER ONE

1. Introduction

1.1.Background

A robot is a reprogrammable, multifunctional manipulator designed to move materials, parts, tools or specialized devices through variable programmed motions for the performance of a variety of tasks. Robots are used for various jobs such as industrial manufacturing, handling of heavy objects, doing dangerous jobs like cleaning of waste materials in the nuclear power plant, and repetitive jobs that are boring, stressful, or labor intensive for humans.

Modern robot arms are becoming more and more compact, light and stable, and are often designed to operate in human environments. For these reasons and due to the vast applications of robotic manipulators, the design of controllers to optimize the tracking and speed performance of robots has become a necessity and very important research area.

The robots have five fundamental components; brain, body, actuator, sensors and power source supply. A brain controls the robot's actions to best response to desired and actual inputs. A robot body is physical structure which can use to holds all parts together. Actuators permit the robot to move based on electrical part (*e.g.*, motors) and mechanical part (*e.g.*, hydraulic piston). Sensors give robot information about its internal and external part of robot environment and power source supply is used to supply all parts of robot [1].

A SCARA robot is one type of a widely used industrial manipulator with three axes and four degrees of freedom widely utilized for various industrial applications where accuracy and precision are essential. Common applications of this robot are assembling, machine loading and uploading, material handling and packaging [2].

SCARA robot stands for Selective Compliance Assembly Robot Arm and is a horizontal - jointed articulating type robot which can rotates in x-y axis and move vertically in z- axis. SCARA robot has high compliance in the x-y plane in the sense that the arm moves freely to accomplish the assembly task accurately. It was first developed under the guidance of Professor Hiroshi Makino from University of Yamaneshi in 1979 [3]

Multi-DOF SCARA industrial robot is a highly non-linear, strong coupling, time-varying systems, so to design a high-precision SCARA robot control system, the robot dynamic model is very significant to generate the control input. It is very complicated operation to obtain its accurate mathematical model, because the coupling between links, the strict nonlinearity and the time varying.

In robot controlling problems, it is very difficult to traces the desired path, so the problem of controlling robot manipulators still offers many practical and theoretical challenges due to the complexities of the robot dynamics and requirements to achieve high – precision trajectory tracking in the cases of high – velocity movement and highly varying loads [4].

The control method depends mainly on mathematical modeling, analysis and synthesis. To obtain the dynamic model in the robotic system Euler-Lagrange method is used because it is direct method for analysis.

Each company produces SCARA robots with different features, but the basic structure is fairly much the same. It has similarities to a human arm with a shoulder, an elbow, and a wrist. Each joint is controlled separately by giving command signal to each of them separately. The two links and 4 axes structure allows four degrees of freedom. Two parallel rotary joints and a linear vertical joint allow freedom in the X-Y-Z space. The fourth degree of freedom is given by the rotational motion of the end effector along the vertical axis. A heavy base is used to make the structure stable [2].

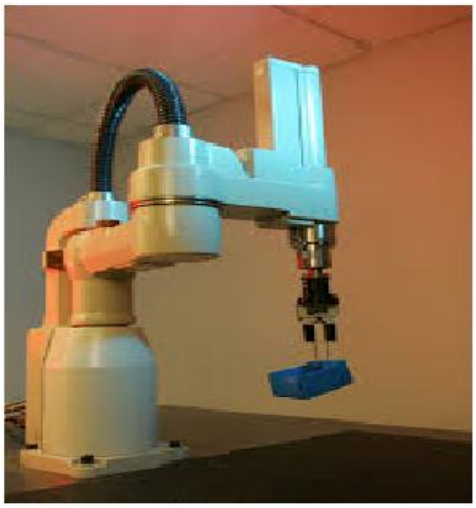

Figure 1.1 *Figure for SCARA robot manipulator [5]*

1.2 Statement of the problem

Robot manipulators dynamics is very complex because of extremely varying parameters (i. e inertia, mass matrix), unmolded joint friction and the large intensive contact with the environment. Such complicated dynamic process modeled by using an approach called the Euler-Lagrangian formulation. The main targets in design of control systems are stability, good disturbance rejection, and small tracking error. Several robots are controlled by linear methodologies, but when robot works with various payloads and has uncertainty in dynamic models this technique has limitations.

Sliding mode controller is a powerful nonlinear robust controller under condition of partly uncertain dynamic parameters of system. The chattering phenomenon problem in sliding mode controller is removed (eliminated) by integrating exponential function in reaching law. But, adding exponential function in reaching law to smooth signum function causes tracking error to be increased in magnitude. To handle this problem, proportional rate gain in the reaching law is tuned by fuzzy controller based on the distance of the states to the sliding surface. Therefore, in this thesis the design of trajectory tracking control of the four degree of freedom SCARA robot manipulator using fuzzy sliding mode with PID-sliding surface control scheme; which can assure chattering free and small tracking error will be studied.

1.3 Objective

1.3.1 General objective

The general objective of this thesis is to model, simulate and design chattering free fuzzy sliding mode controller with PID sliding surface for four degree of freedom SCARA robot manipulator.

1.3.2 Specific objective

The specific objectives of this thesis are:

- To adapt dynamic model of the robot
- To design PID sliding surface, sliding mode controller, fuzzy logic controller.
- To analyses the system stability using Lyapunov method and the performance of the controller.
- To simulate the controller for the system using MATLAB/Simulink

1.4 Contribution

The contribution of this thesis is to drive the mathematical model for the robot, apply the proposed controller to control the SCARA robot manipulator. Also, this thesis can be used as a document of reference for other researches that are interested in this area of robotics using fuzzy sliding mode control.

1.5. Literature Review

Robotics is the branch of rapidly developing technology that studies with the design, construction, manufacture and application of robots. Large number solutions for controlling the robot manipulator have been proposed by many researchers, some of literatures are listed next.

In 2007, E. Munoz, et.al presented terminal sliding control for a SCARA robot which investigate a tracking terminal sliding control law applied on a SCARA robot, a four degree of freedom manipulator. In order to evaluate the performance of the proposed controller, the result has compared in tracking performance and robustness to the computed torque control. From the simulation result terminal sliding control strategy shows better performance in tracking error and robustness. Since the computed torque control technique is based on the assumption of perfect cancellation of dynamic terms, but the composition may be imperfect from model uncertainty. However, in the terminal sliding control, when in the sliding, the closed – loop response become totally insensitive to both disturbance and internal parameter uncertainties [7].

The paper presented by Raul Rascon, et.al in 2013, addressed the trajectory tracking and force regulation of a SCARA robot subject to a position constraint, where impacts may occur. The control law is based on the sliding mode technique, designed for the whole system. That is, the mechanical system and the constraint are modeled as a single system and the controller was developed based on the model whose purpose is, first to force the system to attain the constraint and, once this constraint has been reached, maintain the contact of the system with the constraint. They proposed a sliding mode control algorithm using a sliding mode control which is designed for trajectory tracking and also, a dynamic sliding surface design for position regulation. From simulation result the controlled system exhibits robustness against perturbation and friction but, the system suffers from endless commutations of the control signal that may lead to damage on the actuators [8].

In 2011, Parzin Piltan, et.al proposed the design of sliding mode controller and sliding mode fuzzy controller and investigated the sensitivity of system performance to the sliding surface slope. They designed and implemented the proposed controller for first three joints of the PUMA robot. By adjusting surface of slope the sliding mode controller using fuzzy logic the optimum value of the sliding surface slope is obtained. In this way, the performance of the system has improved with respect to the conventional sliding mode controller. But, in the presence of uncertainties sliding mode fuzzy controller has slight chattering [9].

In 2013, Mohammad Ataei and S. Ehsan Shafie designed a chattering free sliding mode control (SMC) for a robot manipulator including PID part with a fuzzy tunable gain is designed. This controller achieves advantages of the robustness property of SMC and good response characteristics of PID are combined with fuzzy tuning gain approach to achieve more acceptable performance. For this purpose, in the first stage, a PID sliding surface is considered such that the robot dynamic equations can be rewritten in terms of sliding surface and its derivative and the related control law of the SMC design will contain a PID part. The stability guarantee of this sliding mode PID-controller is proved by a lemma using direct Lyapunov method. Then, in the second stage, in order to decrease the reaching time to the sliding surface and deleting the oscillations of the response, a fuzzy tuning system is used for adjusting both controller gains including sliding controller gain parameter and PID coefficient. The proposed methodology is applied to a two-link robot manipulator including model uncertainty and external disturbances as a case study, but the approach is complex and not easily implemented [10].

Another paper [11], deals with the design of sliding mode controller for a robot manipulator. Due to its order reduction property and its low sensitivity to disturbances and plant parameter variations, sliding mode control is an efficient tool to control complex high-order dynamic plants. The approach is based on a method called the reaching law method, which influenced the dynamic quality of the system during the reaching phase, and providing the means for controlling the chattering level. To demonstrate the performance of the proposed control scheme, a set of computer simulation run carried out for two degree of robot manipulator model. It is concluded that the proposed control topology produces better results for both dynamic and steady state operation. But the researcher did not study the influence of system perturbations and external disturbances, and also the problem of chattering is unsolved.

The main drawback of sliding mode control is that they exhibit chattering. Therefore in order to overcome the problem in this thesis FSMC which comprise the exponential function in the reaching law controller is designed for SCARA robot manipulator.

1.6. Methodology

In order to explore the capabilities of fuzzy sliding mode controller with PID sliding surface in effectively controlling the trajectory tracking of a SCARA robot manipulator and to show their validity and effectiveness, the following tasks are achieved throughout the thesis:

i. In the beginning of the research, a review of the literature and state-of-the-art related to this work is established.
ii. In order to design a controller for a specific system, a model for the system must be first developed. By using Euler-Lagrange formulation, which is based on the concepts of generalized coordinates and energy dynamic models for the system is obtained. Therefore, a detailed description of the manipulator structure and its parameters must be determined in order to facilitate the derivation of the kinematics and inverse dynamics equations of the manipulator.
iii. The design of controller
iv. Design the Simulink model: - MATLAB will be the platform to simulate the dynamic model and control system.
v. Simulate and compare the result: - The thesis will compare the results accomplished by the simulation with SMC and proposed controller.

1.7. Thesis Organization

This thesis is organized as follows;

Chapter 1 briefs the overall background of the study. The main part of study such as problem statement, thesis objective, thesis significances, methodology, literature review and thesis report layout is presented well through this chapter. Chapter two describes mathematical modeling, where the details of forward kinematics, inverse kinematics and dynamics of 4 DOF SCARA robot's manipulator are presented. Chapter three describes the methodology, where sliding mode control and fuzzy logic control details are covered. Chapter four shows the controller design and simulation setup of the model implemented on MATLAB/Simulink. Chapter five presents the simulation setup, simulation result and discussion. In chapter six conclusion of this work is given and some recommendations are made. References and some appendices are given at the end of this thesis report.

CHAPTER TWO

2. Modeling of Robot Manipulator

In this chapter the kinematic and dynamic equations of four degree of freedom SCARA robot manipulator will be fully analyzed together with link parameters including the mass, inertia tensor, as well as the geometrical dimensions of each link. The kinematics and dynamics equations of robot manipulator will be used in the controller design process.

2.1 Kinematic Modeling

The Kinematics is the science of motion that treats the motion of a body, without regard to the forces that cause it. Within the science of kinematics, one studies the position, the velocity, the acceleration, or the position and the derivatives of the position variables.

Hence, the study of the kinematics of manipulators refers to all the geometrical and time-based properties of the motion. Study of this part is very important to design controller and in practical applications. The study of motion without regard to the forces (manipulator kinematics) is divided into two main subjects: forward and inverse kinematics [12].

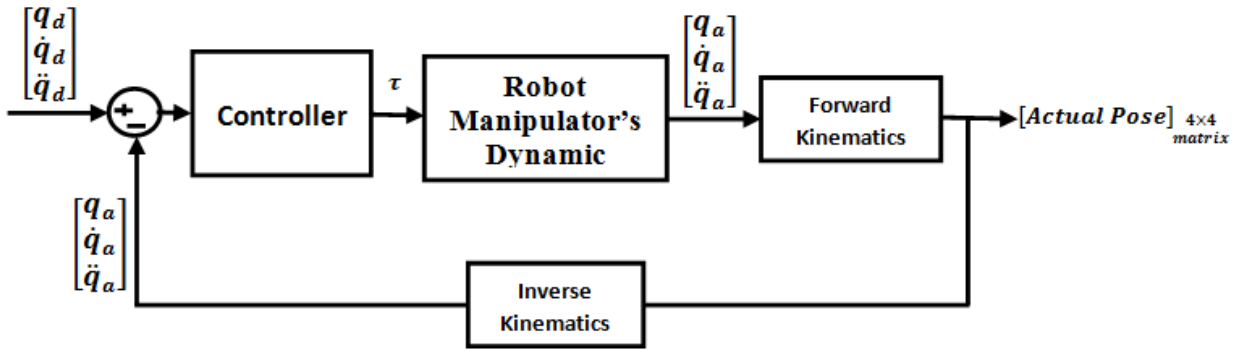

Figure 2.1 Application of forward and inverse kinematics

2.1.1 Forward kinematics

Forward Kinematic is used to determine the position and orientation of robot end effector relative to the robot base coordinate system. In order to obtain the forward kinematics of manipulator the following procedures should be done:

1. ***Obtain Denavit-Hartenberg (DH) convention equations:***

   The kinematic relationship between a pair of adjacent links $i-1$ and i connected through a one-degree-of-freedom. Joint i can be completely determined by a set of four parameters$(a_i, \alpha_i, \theta_i, d_i)$, called parameters of Denavit-Hartenberg. These parameters define the homogeneous transformation between the two coordinate frames attached to the two links.

   With the convention shown in Figure below, the Denavit-Hartenberg parameters are defined as:

   The twist angle α_{i-1} is the angle between the Z-axes of Z_{i-1} and Z_i; measured about X_{i-1};

   The link length a_{i-1} is the length of the common normal to the z-axes of Z_{i-1} andZ_i, measured along X_{i-1};

   The joint angle θ_i is the angle between the X-axes of X_{i-1} and X_i measured about Z_i;

   The joint offset d_i is the distance between the X-axes of X_{i-1} and X_i, measured along Z_i. [12]

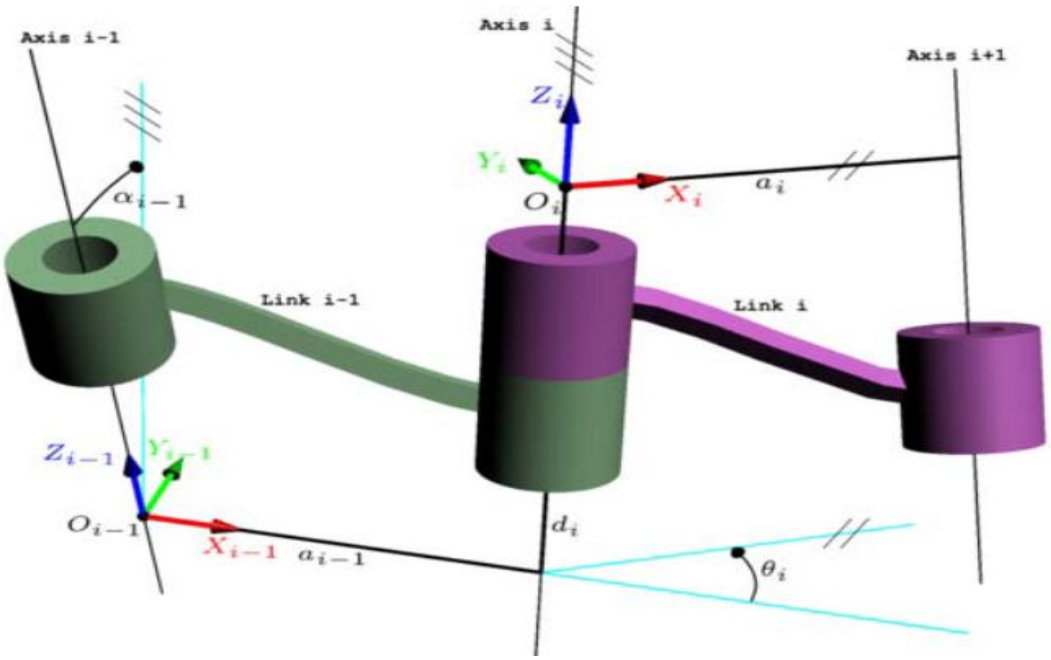

Figure 2.2 Coordinate Frames for the Manipulator[2]

Table 1: a DH convention for 4 DOF SCARA robot manipulator[13]

Link	Link parameters			
i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	a_1	0	θ_2
3	0	a_2	d_3	0
4	0	0	0	θ_4

2. **Derivation of the Homogeneous Transformation Matrices**

The general transformation matrix ${}^{i-1}_{i}T$ for a single link from joint i-1 to joint i is represented as [12]:

$${}^{i-1}_{i}T = Rot(x,\alpha_{i-1})Trans(x,a_{i-1})Rot(z,\theta_i)Trans(z,d_i) \quad \text{..... 2.1}$$

$${}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -d_i s\alpha_{i-1} \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & -d_i c\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{..... 2.2}$$

Where Rot and Trans denote rotation and translation respectively, and sθ_i and cθ_i are short hands for sinθ_i and cosθ_i respectively. The homogeneous rotation and translation matrices can be multiplied together to form a composite homogeneous transformation matrix T [14].

3. **Concatenating Link Transformation**

Using the values of the link parameters the individual link transformation matrix can be computed. Then, the link transformations can be multiplied together to find the single transformation that relates manipulator end effector to base [15].

$${}^{0}_{n}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T \ldots\ldots {}^{n-1}_{n}T \quad \text{..... 2.3}$$

Where {0} base frame and {n} end effector frames.

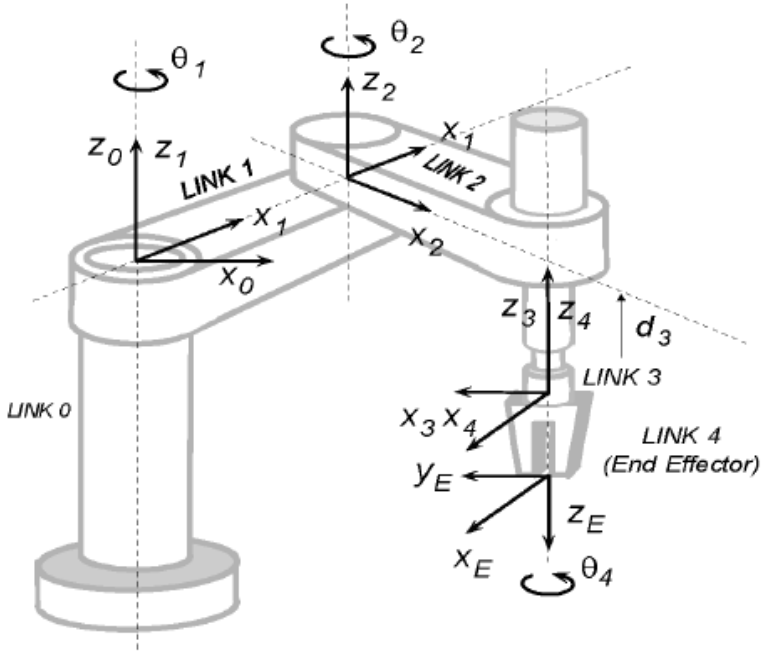

Figure 2.3: Some kinematic parameters and frame assignment for the SCARA Robot Manipulator [16]

The forward kinematics of the end effector with respect to the base frame is determined by substituting the parameters value given in the table (1) in equation (2.3). Hence, n is four in our case and equation (2.3) becomes

$$ {}^{0}_{4}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T \quad \text{..... 2.4} $$

$$ {}^{0}_{4}T = \begin{bmatrix} Ax & Bx & Cx & Px \\ Ay & By & Cy & Py \\ Az & Bz & Cz & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{..... 2.5} $$

Where the entries of the above transformation matrix is given in Appendix B

2.1.2 Inverse kinematics

While direct kinematics establishes relationship between the joint space and Cartesian space, inverse kinematics determine all possible sets of joint variables which will eventually bring the end effector to the set of desired position and orientations. There are two solutions approaches namely, geometric and algebraic used for deriving the inverse kinematics solution analytically. For the manipulators with more links and whose arm extends into 3 dimensions the geometry

gets much more tedious. Hence, algebraic approach is chosen for the inverse kinematics solution [14]. Recall the equation (2.5) to find the inverse kinematics solution for a four-axis manipulator**.**

$$ {}^{0}_{4}T = \begin{bmatrix} Ax & Bx & Cx & Px \\ Ay & By & Cy & Py \\ Az & Bz & Cz & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T $$

To find the inverse kinematics solution for the first joint (q_1) as a function of the known elements of ${}^{base}_{end-effector}T$, the link transformation inverses are premultiplied as follows.

$$ [{}^{0}_{1}T(q_1)]^{-1}\,{}^{0}_{4}T = {}^{1}_{2}T(q_2)\,{}^{2}_{3}T(q_3)\,{}^{3}_{4}T(q_4) \quad \ldots\ldots\ldots\ldots 2.6 $$

To find the other variables, the following equations are obtained as a similar manner.

$$ [{}^{0}_{1}T(q_1)\,{}^{1}_{2}T(q_2)]^{-1}\,{}^{0}_{4}T = {}^{2}_{3}T(q_3)\,{}^{3}_{4}T(q_4) \quad \ldots\ldots\ldots\ldots 2.7 $$

$$ [{}^{0}_{1}T(q_1)\,{}^{1}_{2}T(q_2)\,{}^{2}_{3}T(q_3)]^{-1}\,{}^{0}_{4}T = {}^{3}_{4}T(q_4) \quad \ldots\ldots\ldots\ldots 2.8 $$

There are 12 simultaneous set of nonlinear equations to be solved. The only unknown on the left hand side of equation (2.6) is q1. The 12 nonlinear matrix elements of right hand side are either zero, constant or functions of q2 through q4. If the elements on the left hand side which are the function of q1 are equated with the elements on the right hand side, then the joint variable q1 can be solved as functions of $A_x, A_y, A_z, \ldots C_z, P_x, P_y, P_z$ and the fixed link parameters. Once q1 is found, then the other joint variables are solved by the same way as before [14].

The possible sets of joint variables are:

$$ q_1 = \tan^{-1}\left(\frac{Py(a_1+a_1c_2)-Pxa_2s_2}{Pya_2s_2+Px(a_1+a_2c_2)}\right) \quad \ldots\ldots\ldots\ldots 2.9 $$

$$ q_2 = \cos^{-1}\left(\frac{Py^2+Px^2-{a_1}^2-{a_2}^2}{2a_1a_2}\right) \quad \ldots\ldots\ldots\ldots 2.10 $$

$$ q_3 = -p_z \quad \ldots\ldots\ldots\ldots 2.11 $$

$$ q_4 = q_1 - q_2 - arctan2(Ay, Ax) \quad \ldots\ldots\ldots\ldots 2.12 $$

2.2 Dynamic Modeling

Dynamics is a science of motion which treats motion with regard to the torques applied by the actuators applied to the manipulator. There are two problems related to dynamics of a manipulator. The first one is direct dynamics where the torque is calculated for a given values of angular displacement, angular velocity and acceleration. The second problem is of indirect

dynamics where the angular velocity, angular displacement and acceleration are calculated for a given torque [14], [15].

The dynamic study of a manipulator can be done by several methods such as: Newton-Euler (N-E) method, Lagrange Euler (L-E) method, etc. The Euler-Lagrange method depends on finding the total kinetic and potential energies of the manipulator and then using them to calculate the Lagrangian (L) of the whole robot system which can be used to calculate the torque of each robot joint. The Newton – Euler approach is based on the elementary dynamic formulation and on an analysis of forces and moments acting on each links. The N-E method involves a set of forward and backward recursive equations. The forward recursion propagates the kinematics information such as velocity and acceleration at the center mass of each link. The backward recursive formulation propagates the forces and moments exerted on each from end effector to the base of robot. Newton – Euler formulation might be said to be a "force balance" approach to dynamics. [15].In this paper, the mathematical formulation of the dynamic equation of motion is derived using the L-E method.

2.2.1 Euler – Lagrange Formulation

The Lagrange formulation is an "Energy-based" approach to dynamics and describes the behavior of a dynamic system in terms of work and energy stored in the system rather than of forces and moment of the individual members involved. The closed – form dynamic equations can be derived systematically in any coordinate system.

Let $q_1, \ q_2, \dots, q_n$ be generalized coordinates that completely locate a dynamic system. Let K be the total kinetic energy and P be potential energy stored in the dynamic system.

We define the Lagrange L by

$$L(q_i, \ \dot{q}_i) = K - P \quad \dots\dots\dots\dots\dots\dots \ 2.13$$

Since kinetic energies and potential energies are functions of $q_i \ and \ \dot{q}_i \ \ (i = 1, \ 2, \dots, n)$; so is the Lagrangian L.

Using the Lagrangian, the equations of motion of dynamic system are given by [15]

$$\tau_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \quad \dots\dots\dots\dots\dots\dots \ 2.14$$

Where τ_i is the generalized torque corresponding to generalized coordinate q_i .

To obtain the Lagrangiam of the robot manipulator, the whole kinetic energies and potential energies of the robot must be derived first.

2.2.2. General Expression for Kinetic Energy

The total kinetic energy of robot manipulator is defined as the sum of the kinetic energies of the individual links of the manipulator. In order to obtain the total kinetic energy of the manipulator we begin by deriving the kinetic energy stored in an individual arm link. As shown in Figure (2.4) below, let v_{ci} be the 3 by 1 velocity vector of the centroid and ω_i be the 3 by 1 angular velocity vector with reference to the base coordinate frame, which is an inertial frame. The kinetic energy of link i is then given by [14], [15].

$$k_i = \frac{1}{2} m_i {v_{ci}}^T v_{ci} + \frac{1}{2} {\omega_i}^T I_{ci} \omega_i \quad \text{..... 2.15}$$

Where m_i is the mass of the link and I_{ci} is the 3 by 3 inertia tensor at the centroid expressed in the base coordinates. The first term in the above equation accounts for the kinetic energy resulting from the translational motion of the mass m_i, while the second terms represents the kinetic energy resulting from rotation about the centroid. The total kinetic energy stored in the whole arm is then given by

$$K = \sum_{i=1}^{n} K_i \quad \text{..... 2.16}$$

Since energy is additive.

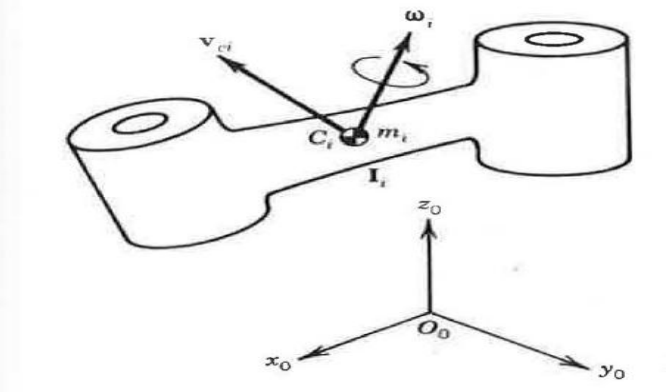

Figure 2.4: linear and angular velocity of link i whose mass is m_i

Inertia Tensor

Inertia tensor gives us an idea about how the mass is distributed in a rigid body. Let the mass density of the object be represented as a function of position $\rho(x, y, z)$. Then the inertia tensor in the body attached frame is computed as [14], [15]

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \quad \text{..... 2.16}$$

Where

$$I_{xx} = \iiint (y^2 + z^2)\rho(x, y, z)dxdydz$$

$$I_{yy} = \iiint (x^2 + z^2)\rho(x, y, z)dxdydz$$

$$I_{zz} = \iiint (y^2 + x^2)\rho(x, y, z)dxdydz$$

$$I_{xy} = I_{yx} = -\iiint xy\rho(x, y, z)dxdydz$$

$$I_{xz} = I_{zx} = -\iiint xz\rho(x, y, z)dxdydz$$

$$I_{yz} = I_{zy} = -\iiint zy\rho(x, y, z)dxdydz$$

The integrals in the above expression are computed over the region of space occupied by the rigid body. The diagonal elements of the inertia tensor, I_{xx}, I_{yy}, I_{zz} are called the principal moments of inertia about the x, y, and z axes respectively. The off diagonal terms $I_{xy}, I_{xz}, \ldots$, etc. are called the cross products of inertia. If the mass distribution of the body is symmetric with respect to the body attached frame then the cross product of inertia are identically zero [13], [14].

Kinetic Energy for an n – Link Robot

From forward kinematics the linear and angular velocities of any point on any link can be expressed in terms of the $Jacobian$ matrix and the derivatives of the joint variables [15]. Since, in our case the joint variables are the generalized coordinates. For appropriate $Jacobian$ matrices

$$v_i = J_{vi}(q)\dot{q}_i \quad \text{..... 2.18}$$

$$\omega_i = J_{\omega i}(q)\dot{q}_i \quad \text{..... 2.19}$$

Where $J_{vi}(q)$ is a $Jacobian$ matrix associated with linear velocity of link i and $J_{\omega i}(q)$ $Jacobian$ matrix associated with angular velocity of link i.

Suppose the mass of link i is m_i, and the inertia matrix of link i evaluated around a coordinate frame parallel to frame i whose origin is at the center of mass, equals I_i. Substituting equations (2.18) and (2.19) in equation (2.15), the overall kinetic energy of the manipulator becomes

$$K = \frac{1}{2}\dot{q}^T \sum_{i=1}^{n}\left[m_i {J_{vi}}^T J_{vi} + {J_{\omega i}}^T I_i J_{\omega i}\right]\dot{q} \quad \text{..... 2.20}$$

In other words, the kinetic energy of the manipulator is of the form:

$$K = \frac{1}{2}\dot{q}^T M(q)\dot{q} \quad \text{..... 2.21}$$

Where

$$M(q) = \sum_{i=1}^{n}\left[m_i {J_{vi}}^T J_{vi} + {J_{\omega i}}^T I_i J_{\omega i}\right] \quad \text{..... 2.22}$$

$M(q)$ is a symmetric positive definite matrix that is in general configuration dependent and it is called the inertia matrix [14],[15].

2.2.3. General Expression for Potential Energy

In the case of rigid dynamics, the only source of potential energy is gravity. The potential energy of the i^{th} link can be computed by assuming that the mass of the entire object is concentrated at its center of mass and given by [14],[15]

$$P_i = g^T P_{ci} m_i \quad \text{..... 2.23}$$

Where g is the vector giving the direction of gravity in the inertial frame, P_{ci} is the center of mass of link i.

The total potential energy of the an n – link robot is

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} g^T P_{ci} m_i \quad \text{..... 2.24}$$

Note that the potential energy is a function of only generalized coordinate and not their derivatives, i.e. the potential energy depends on the configuration of the robot but not on its velocity [14].

Lagrange Formulation for 4 DOF SCARA Robot Manipulator

In order to find the Lagrangian of 4 DOF SCARA robot manipulator, the total kinetic energies and potential energies of the manipulator must be driven. The derivation of the total kinetic energies and potential energies of 4 DOF SCARA robots is done as follows:

Formulation of Kinetic Energy

From equation (2.20) and (2.22) the kinetic energy mass matrix of the 4DOF SCARA manipulator is become

$$K = \frac{1}{2}\dot{q}^T \sum_{i=1}^{4}\left[m_i {J_{vi}}^T J_{vi} + {J_{\omega i}}^T I_i J_{\omega i}\right]\dot{q} \quad \text{Hence n = 4}$$

$$M(q) = \sum_{i=1}^{4}\left[m_i {J_{vi}}^T J_{vi} + {J_{\omega i}}^T I_i J_{\omega i}\right]$$

$$= m_1 {J_{v1}}^T J_{v1} + {J_{\omega 1}}^T I_1 J_{\omega 1} + m_2 {J_{v2}}^T J_{v2} + {J_{\omega 2}}^T I_2 J_{\omega 2} + m_3 {J_{v3}}^T J_{v3} + {J_{\omega 3}}^T I_3 J_{\omega 3} + m_4 {J_{v4}}^T J_{v4} + {J_{\omega 4}}^T I_4 J_{\omega 4} \quad \text{..... 2.25}$$

J_{vi} is the Jacobean associated with linear velocity of link i, it is direct differentiation of the vector $[P_{ci}]$ which is position vector of the center of mass of link i from base frame.

$$J_{vi} = \left[\frac{\partial P_{ci}}{\partial q_1} \;\; \frac{\partial P_{ci}}{\partial q_2} \;\; \ldots\ldots \frac{\partial P_{ci}}{\partial q_i} \;\; 0 \;\; 0 \;\; \ldots..0\right] \quad \text{..... 2.26}$$

And

$J_{\omega i}$ is the Jacobean associated with angular velocity of link i which is given by [14],[15]

$$J_{\omega i} = [\bar{\varepsilon}_1 z_1 \;\; \bar{\varepsilon}_2 z_2 \ldots\ldots \bar{\varepsilon}_i z_i \;\; 0 \;\; 0 \ldots. 0] \quad \text{..... 2.27}$$

$$with\ \varepsilon_i = \begin{cases} 0, & revolute \\ 1, & prismatic \end{cases} \ and\ \bar{\varepsilon}_i = 1 - \varepsilon_i$$

The inertia tensor matrix at the center of mass of the each link of the manipulator given by [14]

$$I_1 = \begin{bmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{bmatrix}, I_2 = \begin{bmatrix} I_{xx2} & 0 & 0 \\ 0 & I_{yy2} & 0 \\ 0 & 0 & I_{zz2} \end{bmatrix}, I_3 = \begin{bmatrix} I_{xx3} & 0 & 0 \\ 0 & I_{yy3} & 0 \\ 0 & 0 & I_{zz3} \end{bmatrix},$$

$$I_4 = \begin{bmatrix} I_{xx4} & 0 & 0 \\ 0 & I_{yy4} & 0 \\ 0 & 0 & I_{zz4} \end{bmatrix} \quad \text{..... 2.28}$$

Substituting all the values of J_v, J_ω and I in equation (2.25), the mass matrix of the 4DOF SCARA robot manipulator become:

$$M(q) = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \quad \text{..... 2.29}$$

Where the entries of the above inertia matrix is given in Appendix B

Finally, substituting the parameters' equation obtained above in equation (3.15), the kinetic energy of the 4DOF SCARA robot manipulator becomes

$K = m_{11}\dot{q}_1{}^2 + m_{22}\dot{q}_2{}^2 + m_{33}\dot{q}_3{}^2 + m_{44}\dot{q}_4{}^2 + 2m_{12}\dot{q}_1\dot{q}_2 + 2m_{41}\dot{q}_1\dot{q}_4 + m_{24}\dot{q}_2\dot{q}_4$ ….. 2.30

Potential Energy Calculation

For 4 DOF SCARA robot manipulator the potential energy given in equation (2.24) becomes [14], [15]:

$$P = \sum_{i=1}^{4} P_i = \sum_{i=1}^{4} g^T P_{ci} m_i \quad \text{…….. 2.31}$$

Where P_{ci} is the position vector of a link i at the center of mass, m_i is mass of link i and g is gravity vector in m/s^2 can be written as:

$$g = \begin{pmatrix} 0 \\ 0 \\ -10 \end{pmatrix}$$

Then the potential energy of the system becomes

$$P = 5m_3(d_3) + 10m_4(d_3) \quad \text{……..…… 2.32}$$

After computing the total kinetic energy and potential energy of the 4 DOF SCARA robot manipulator, substituting their values in equation (2.14), the standard form representation becomes:

$$\tau = M(q)\ddot{q} + B(q)[\dot{q}\dot{q}] + C(q)[\dot{q}^2] + G(q) \quad \text{…… 2.33}$$

Where

τ is actuator's torque and is n by 1 vector, $M(q)$ is positive definite mass matrix and is n by n symmetric matrix, $B(q)$ is a Coriolis torque matrix and is n by $\frac{n(n-1)}{2}$ matrix, $C(q)$ is Centrifugal torque matrix and is 4 by 4 matrix, $G(q)$ is gravity force vector and is 4 by 1 and their representation is given by

$$G(q) = \begin{bmatrix} 0 \\ 0 \\ 5(m_3 + 2m_4) \\ 0 \end{bmatrix}, \text{ Where } G(q) = \frac{\partial P}{\partial q_i}$$

$$B(q) = \begin{bmatrix} 2b_{1,12} & 2b_{1,13} & \dots & 2b_{1,(n-1)n} \\ 2b_{2,12} & 2b_{2,13} & \dots & 2b_{2,(n-1)n} \\ \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & \dots \\ 2b_{n,12} & 2b_{n,13} & \dots & 2b_{n,(n-1)n} \end{bmatrix}$$

$$B(q) = \begin{bmatrix} 2b_{112} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$2b_{112} = m_2 a_1 a_2 (s_1 c_{12} - c_1 s_{12}) + 2m_3 a_1 a_2 (s_1 c_{12} - c_1 s_{12}) + 2m_4 a_1 a_2 (s_1 c_{12} - c_1 s_{12})$$

$$C(q) = \begin{bmatrix} b_{1,11} & b_{1,22} & \dots & b_{1,nn} \\ b_{2,11} & b_{2,22} & \dots & b_{2,nn} \\ \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & \dots \\ b_{n,11} & b_{2,nn} & \dots & b_{n,nn} \end{bmatrix}$$

$$C(q) = \begin{bmatrix} 0 & b_{122} & 0 & 0 \\ b_{211} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$b_{122} = \frac{1}{2} m_2 a_1 a_2 (s_1 c_{12} - c_1 s_{12}) - m_3 a_1 a_2 (s_{12} c_1 - c_{12} s_1) - m_4 a_1 a_2 (s_{12} c_1 - c_{12} s_1)$$

$$b_{211} = -2m_3 a_1 a_2 s_1 c_{12} - 2m_4 a_1 a_2 s_1 c_{12} - m_2 a_1 a_2 (s_1 c_{12} - c_1 s_{12})$$

Where the term b_{ijk} is called Christoffel symbols and is given as

$$b_{ijk} = \frac{1}{2}\left(m_{ijk} + m_{ikj} - m_{jki}\right), \qquad such\ that\ \ m_{ijk} = \frac{\partial m_{ij}}{\partial q_k}$$

The equation (2.33) is a direct dynamic representation in which torque is computed for a given values of angular displacement, angular velocity, and angular acceleration. But, in controlling problem angular displacement, angular velocity, and angular acceleration are controlled by manipulating the torque, and the following expression is used [14] [15].

$$\ddot{q} = [M(q)]^{-1}\{\tau - B(q)[\dot{q}\dot{q}] - C(q)[\dot{q}^2] - G(q)\} \quad \text{..... 2.34}$$

Define a 2n- dimensional state vector x as:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \quad \text{..... 2.35}$$

The nonlinear representation of the system in equation (2.34) is can be described as follows

$$\dot{x} = A(x) + B(x)u \quad \text{..... 2.36}$$

Where $u = \tau$; $A(x) = \begin{bmatrix} x_2 \\ -M(x_1)^{-1}[B(x_1, x_2) + C(x_1, x_2) + G(x_1) + \tau_d] \end{bmatrix}$,

$$B(x) = \begin{bmatrix} 0 \\ M(x_1)^{-1} \end{bmatrix}$$

CHAPTER THREE

3. Methodology

3.1. Introduction

In chapter two we briefly discussed kinematics and dynamic of SCARA robot manipulator. This chapter provides a survey of sliding mode control theory and fuzzy logic control theory. Sliding mode control is known to possess merits such as the invariance to parametric uncertainties as well as its capability in disturbance rejections. Recently it has been applied successfully to various types of physical systems. However, this type of control suffers from the chattering phenomenon which is due to high frequency switching over discontinuity of the control signal (i.e. switching signal: $sgn(s(t))$). This effect may cause damage to the controlled physical systems. Nowadays, typical approaches have been developed to reduce the amount of chattering which the most significant of them is intelligent control approach (Kaynak et al., 2001) mainly includes fuzzy logic control and neural network control.

3.2. Sliding Mode Control

The term *sliding mode control* (SMC) first appeared in the context of variable structure systems. Sliding modes became the principal operational mode for this class of control systems. Practically all design methods for variable structure systems are based on deliberate introduction of sliding modes. Due to its order reduction property and its low sensitivity to disturbances and plant parameter variations, SMC is an efficient tool to control complex high-order dynamic plants operating under uncertainty conditions which are common for many processes of modern technology [17], [18].

Sliding mode control (SMC) is a variable-structure, robust control strategy which is capable in controlling different class of uncertain systems including nonlinear systems, MIMO systems. Such uncertainties may be structured, unstructured, or may result from nondeterministic features of the plant. Applications such as in aerospace, robotics, process control and many more are using this conventional nonlinear controller. Using this controller can resolve the challenge of stability and robustness in control theory [18].

VSC systems comprise a collection of different, usually quite simple, feedback control laws and a decision rule. Depending on the status of the system, a decision rule, often termed the

switching function, determines which of the control laws is "on-line" at any one time. The transient dynamics of a SMC system consists of two modes: a "reaching mode" (or "nosliding mode"), followed by a "sliding mode". Therefore the design of VSC involves, first, the design of an appropriate *n*-dimensional switching function $\boldsymbol{s}(\boldsymbol{x})$ for a desired sliding mode dynamics, and second, the design of a control for the reaching mode such that a reaching condition is met. The desired sliding mode dynamics is usually a fast and stable error-free response (an asymptotic convergence to the final state will be achieved in sliding mode). For the reaching mode, the desired response usually is to reach the switching manifold in finite time [19]

Let us, consider a given system from equation (2.36) represented by the state-variable form:

$$\dot{x} = A(x) + B(x)u \quad \ldots\ldots 3.1$$

The statement of the Sliding Mode Control problem can be stated as:

(1) Find switching functions, $s(x)$

(2) Find a variable structure control

$$u(x,t) = \begin{cases} u(x,t)^{+} \; when \; s(x) > 0 \\ u(x,t)^{-} \; when \; s(x) < 0 \end{cases} \quad \ldots\ldots 3.2$$

Such that the reaching modes satisfy the reaching condition, namely, reach the set $\boldsymbol{s}(\boldsymbol{x}) = \boldsymbol{0}$ (the switching surface) in finite time.

The physical meaning of above statement is as follows [20]:
(1) Design a switching surface $\boldsymbol{s}(\boldsymbol{x}) = \boldsymbol{0}$ to represent a desired system dynamics, which is of lower order than the given plant.

(2) Design a variable structure control $\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{t})$such that any state $\boldsymbol{x}$ outside the switching surface is driven to reach the surface in finite time. On the switching surface, the sliding mode takes place. In this way, the overall VSC system is globally asymptotically stable.

3.2.1. Sliding Surface

Once the system states are sliding on the sliding surface, the system characteristics are governed by the characteristics of the sliding surface only. Therefore, sliding surface design is performed to achieve the characteristics required from the system when sliding. For example, if the tracking error is to be kept at zero, then the sliding variable may be selected as a stable differential equation of tracking error.

According to the theory of sliding mode controller, the main important part to design this controller is the sliding surface, where the time-varying sliding surface $s(x, t)$ in the state space R^n is given by the following formulation [21].

Let as define $e = x_d - x$ the tracking error in the variable x, and let

$e = x_d - x = [e, \dot{e}, \ddot{e}, .. e^{n-1}]^T$ be the tracking error vector. Furthermore, let us define a time-varying surface

$$s(x,t) = \left(\frac{d}{dt} + \lambda\right)^{(n-1)} e \quad \text{..... 3.3}$$

Based on Eq. (3.3) λ is the sliding surface slope coefficient and it is a positive constant [22]. It is important to note that while the system is in sliding mode, (i.e. while the state trajectories join the sliding surface) the dynamics neither depends on the plant parameters nor the disturbance. This so-called ―invariance property looks promising for designing feedback control for the dynamic plants operating under uncertainty conditions [23].

The sliding surface can be defined as Proportional-Derivative (PD), Proportional-Integral (PI) and the Proportional-Integral-Derivative (PID). The following formulations represented the three groups are [21]:

$$s_{PD} = \dot{e} + \lambda e \quad \text{..... 3.4}$$

$$s_{PI} = \lambda e + \left(\frac{\lambda}{2}\right)^2 \int e \quad \text{..... 3.5}$$

$$s_{PID} = \dot{e} + \lambda e + \left(\frac{\lambda}{2}\right)^2 \int e \quad \text{..... 3.6}$$

Taking into consideration the fact that the static error cannot be completely canceled, especially in the case of uncertainties and parametric variations, hence the integral term added to the error expressions to push this static error to be totally cancelled.

In order to gain stability and minimum error in sliding mode controller, the sliding surface slope $s(x, t)$ is kept near to the zero. The system is said to be in the sliding mode when

$$s(x,t) = 0 \quad \text{..... 3.7}$$

If $s(x,t)$ gives rise to a stable linear differential equation by design, the states will automatically converge to the origin asymptotically. The condition on the sliding surface is that it should have

relative degree one with respect to the system input i.e., the first time derivative of the sliding variable is a function of the control input.

3.2.2. Sliding Mode Control Law

In the design of the sliding mode control, one must first find the equivalent control law u_{eq}, which will keep the state of the system on the sliding surface.

Equivalent Control Law

The equivalent control law is found by recognizing that: $\dot{s}/_{u=u_{eq}} = 0$. Differentiating $s(x,t)$ with respect to t along the trajectories of equation (3.1) gives [24]

$$\dot{s}(x,t) = e^{(n)} + \sum_{k=1}^{n-1}\binom{n-1}{k}\lambda^k e^{(n-k)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.. 3.7$$

$$= A(x,t) + B(x,t)u(t) - x^{(n)}{}_d + \sum_{k=1}^{n-1}\binom{n-1}{k}\lambda^k e^{(n-k)}$$

Thus, the equivalent control becomes:

$$u_{eq} = (B(x,t))^{-1}\left\{x^{(n)}{}_d - A(x,t) - \sum_{k=1}^{n-1}\binom{n-1}{k}\lambda^k e^{(n-k)}\right\} \quad \ldots\ldots\ldots\ldots\ldots.. 3.8$$

For stabilization problems, the derivative of the reference position is zero. Therefore, using the chain rule a more compact relation can be found [31]

$$\dot{s}(x,t) = \frac{\partial s}{\partial x}\dot{x} = \frac{\partial s}{\partial x}\{ A(x,t) + B(x,t)u(t) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.9$$

And setting $\dot{s}/_{u=u_{eq}} = 0$

$$u_{eq} = -\left(\frac{\partial s}{\partial x}B(x,t)\right)^{-1}\frac{\partial s}{\partial x}A(x,t) \quad \ldots\ldots\ldots\ldots\ldots..\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.10$$

Provided that the matrix $\frac{\partial s}{\partial x}B(x,t)$ is non singular

Switching Control Law

The condition under which the state will move toward and reach a sliding surface is called a reaching condition. The system trajectory under the reaching condition is called the reaching mode or reaching phase. Gao and Hung [25] Proposed a new approach, based on a new method called the *reaching law method*, for the design of VSC of nonlinear systems.

3.2.3. Reaching Law Method for VSC Design

The method simultaneously takes care of ensuring the reaching condition, arranging the logic for the free-order switching, influencing the dynamic quality of the system during the reaching phase, and providing the means for controlling the chattering level. The procedure of using this method is straightforward and easy to carry out, even for nonlinear systems.

The reaching law is a differential equation which specifies the dynamics of a switching function $\boldsymbol{s}(\boldsymbol{x})$. The differential equation of an asymptotically stable $\boldsymbol{s}(\boldsymbol{x})$ is itself a reaching condition. In addition, by the choice of the parameters in the differential equation, the dynamic quality of VSC system in the reaching mode can be controlled. A practical general form of the reaching law [25] is:

$$\dot{s} = -Qsgn(s) - Kh(s) \quad \text{..... 3.12}$$

Where: - $Q = diag[q_1, \ldots . q_n], q_i > 0, sgn(s) = [sgn(s_1), \ldots . sgn(s_n)]^T$

$K = diag[k_1, \ldots . k_n], k_i > 0$, $h(s) = [h_1(s_1), \ldots . h_n(s_n)]^T \;\; h_i(s_i) > 0$

Three practical special cases of (3.12) are given below.

1) **Constant rate reaching:**

$$\dot{s} = -Qsgn(s) \quad \text{..... 3.13}$$

This law forces the switching variable $\boldsymbol{s}(\boldsymbol{x})$ to reach the switching manifolds S at a constant rate $|\dot{s}_i| = -q_i$. The merit of this reaching law is its simplicity. But, as will be shown later, if qi is too small, the reaching time will be too long. On the other hand, a qi too large will cause severe chattering [25].

2) **Constant plus proportional rate reaching:**

$$\dot{s} = -Qsgn(s) - Ks \quad \text{..... 3.14}$$

Clearly, by adding the proportional rate term $-\boldsymbol{Ks}$, the state is forced to approach the switching manifolds faster when $\boldsymbol{s}$ is large. It can be shown that the reaching time for $\boldsymbol{x}$ to move from an initial state $\boldsymbol{x}_0$ to the switching manifold S_i is finite, and is given by [25]:

$$T_i = \frac{1}{K_i} \ln \frac{K_i|s_i| + q_i}{q_i} \quad \text{..... 3.15}$$

3) Power rate reaching :

$$\dot{s}_i = -k_i|s_i|^{\alpha} sgn(s_i) \quad \text{.....} \quad 3.16$$

$0 < \alpha < 1, i = 1, \dots, n$ [25]

This reaching law increases the reaching speed when the state is far away from the switching manifold, but reduces the rate when the state is near the manifold. The result is a fast reaching and low chattering reaching mode. Integrating (above equation) from $s_{i=}s_{i0}\ to\ s_i$ yields

$$T_i = \frac{|s_i|^{1-\alpha}}{(1-\alpha)k_i}, i = 1, \dots, n \quad \text{.....} \quad 3.17$$

Showing that the reaching time T_i is finite.

Thus power rate reaching law gives a finite reaching time. In addition, because of the absence of the $-\boldsymbol{Qsgn(s)}$ term on the right-hand side of (3.16), this reaching law eliminates the chattering.

Control Law

Having selected the reaching law equation, the control law can now be determined. Compute the time derivative of $\boldsymbol{s(x)}$ along the reaching mode trajectory, then from (3.12) and considering $\boldsymbol{s(x) = 0}$:

$$\dot{s} = \frac{\partial s}{\partial x}\dot{x}\ where\ \dot{x} = A(x) + B(x)u$$

$$\dot{s} = \frac{\partial s}{\partial x}A(x) + \frac{\partial s}{\partial x}B(x)u = -Qsgn(s) - Kh(s) \quad \text{.....} \quad 3.18$$

Noting that the matrix $\boldsymbol{(\partial s\ /\ \partial x)B(x)}$ is nonsingular, this equation is solved for the control law, giving:

$$u = -\left(\frac{\partial s}{\partial x}B(x)\right)^{-1}\left[\frac{\partial s}{\partial x}A(x) + Qsgn(s) + Kh(s)\right] \quad \text{.....} \quad 3.19$$

This control law appears independent of system perturbation and external disturbances, which is not realistic. In fact, the control $\boldsymbol{u}$ does depend on perturbation and disturbances, and it should include their parameters [25]. The principle of designing SMC law for arbitrary-order plants is to force the error and derivative of error of a variable to zero.

It can be discerned that there are some problems for the VSC using the reaching law expressed in (3.14) and (3.18) [26]:

The increase of Q may result in the increase of the reaching speed, but this will cause severe chattering, on the contrary, the decrease of Q will lead to the decrease of the reaching speed, the reaching time will be too long and the system response of the sliding mode will go bad.

The increase of K may shorten the time from the reaching mode to the sliding mode, but this will lead to very high gain control when $|s|$ is large, it is undesired from the engineering point of view; on the other hand, the decrease of K can reduce the strength of control, which will affect the performance of the controller.

Therefore, the reaching law method implies an unavoidable contradiction and tradeoff between the dynamic quality and the chattering must be made.

Chattering: - Defined as unwanted fast oscillations of the system trajectories near the sliding surface. It is dangerous high-frequency vibration of the controlled system. It needs to be considered in any first order sliding mode control (SMC) implementation. Because any SMC control signal has fast switching nature, those fast switching will lead to chattering which may excite unmolded high frequency dynamics, which is not accounted in the design and the system performance degrades and may even lead to instability.

To overcome these problems artificial intelligence idea is introduced to tune gains.

3.3. Introduction to Fuzzy Control

Fuzzy logic was conceived by Lotfi Zadeh and brought to the attention of the world in his paper "Fuzzy Set" in 1965. At that time, fuzzy logic was a radical deviation from classical logic, and initially received little interest and attention. Slowly fuzzy logic is gaining more and more credit in the United States as a legitimate method for controls [27]. The driving forces behind the adoption of fuzzy logic to controls applications are: model independence, use of expert knowledge to control systems, robustness to noise/disturbances, capable of controlling nonlinear systems, etc.

3.3.1. Basic Concepts of Fuzzy Logic [28], [29]

Fuzzy Set

Fuzzy set is an extension of the classical set. In classical crisp set theory, the membership of elements complies with a binary logic either the element belongs to the crisp set or the element does not belong to the set. While in fuzzy set theory, it can contain elements with degree of membership between completely belonging to the set and completely not belonging to the set. This is because a fuzzy set does not have a crisp, clearly defined boundary, and its fuzzy

boundary is described by membership functions which make the degree of membership of elements range from 0 to 1.

Membership Functions

The membership function maps each element of the universe to a membership grade between 0 and 1.

We define *X* as the universe of discourse (or the universe). A fuzzy set A in X is given by

$$A = \{(x, \mu_A(x)) | x \in X\}$$

Where $\mu_A(x)$ is the membership function (MF) of the fuzzy set *A* and *x* are elements of *X*. The membership function $\mu_A(x)$ of the fuzzy set *A* can have values between 0 and 1 [28].

If $\mu_A(x)$ can only have values of 1 or 0, the fuzzy set *A* is changed to a classical set. Classical sets are also called crisp sets. Figure 3.1 illustrates the typical membership functions of fuzzy sets "small," "medium," and "big. The universe *X* is defined as "torque". There are three different fuzzy sets with three membership functions $\mu_{small}(x)$, $\mu_{midum}(x)$, $\mu_{big}(x)$.

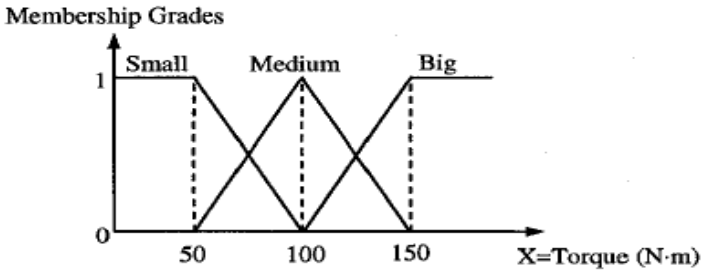

Figure 3.1: Membership functions of fuzzy sets

Figure below shows a general membership function curve. The horizontal axis represents an input variable *x*, and the vertical axis defines the corresponding membership value *μ(x)* of the input variable *x*. The Support of membership function curve explains the range where the input variable will have nonzero membership value. In this figure, $\mu_A(x) \neq 0$ when *x* is any point located between point *a* and point *d*. While the Core of membership function curve interprets the range where the input variable *x* will have full degree of membership ($\mu_A(x)$= 1), in other words the arbitrary point within the interval [*b, c*] completely belongs to a fuzzy set which is defined by this membership function.

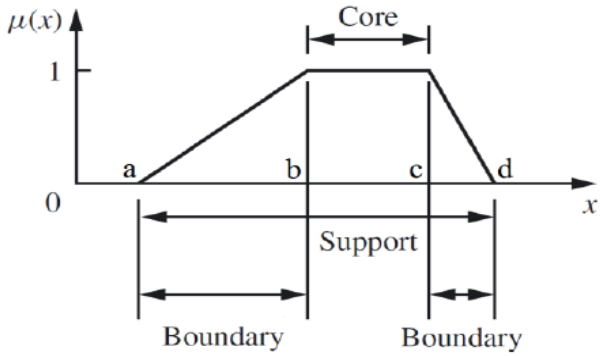

Figure 3.2: A sample of membership function

Generally, there are five common shapes of membership function: Triangle MF, Trapezoidal MF, Gaussian MF, Generalized Bell MF, and Sigmoidal MF. Regardless of the shape, a single MF may only define one fuzzy set.

A triangular membership function of a fuzzy set is defined as [27]

$$triangle(x; a, b, c) = \begin{cases} 0 & x \le a \\ \frac{x-a}{b-a} & a \le x \le b \\ \frac{c-x}{c-b} & b \le x \le c \\ 0 & d \le x \end{cases} \quad \text{..... 3.20}$$

Where the parameters $\{a, b, c\}$ are the corner points of the membership function.

A Gaussian membership function is defined as [14]

$$gaussian(x;\ \alpha, \delta) = e^{-\frac{(x-\alpha)^2}{2\delta^2}} \quad \text{..... 3.21}$$

Where α, is the mean and δ is the variance. Figure 3.3 shows a Gaussian membership function with $\alpha = 100$ and $\delta = 30$.

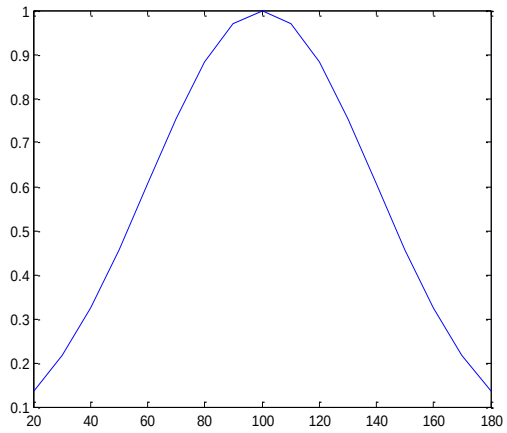

Figure 3.3: Gaussian membership function of fuzzy set "Medium"

Logical Operation

Because the standard binary logic is a special case of fuzzy logic where the membership values are always 1 (completely true) or 0 (completely false), fuzzy logic must hold the consistent logical operations as the standard logical operations. The most foundational logical operations are AND, OR and NOT. Unlike standard logical operation, the operands *A* and *B* are membership values within the interval [0, 1]. In fuzzy logical operations, logical AND is expressed by function *min*, so the statement *A* AND *B* is equal to *min* (*A*, *B*). Logical OR is defined by function *max*, thus *A* OR *B* becomes equivalent to *max* (*A*, *B*). And logical NOT makes operation NOT *A* become the operation 1 – *A*.

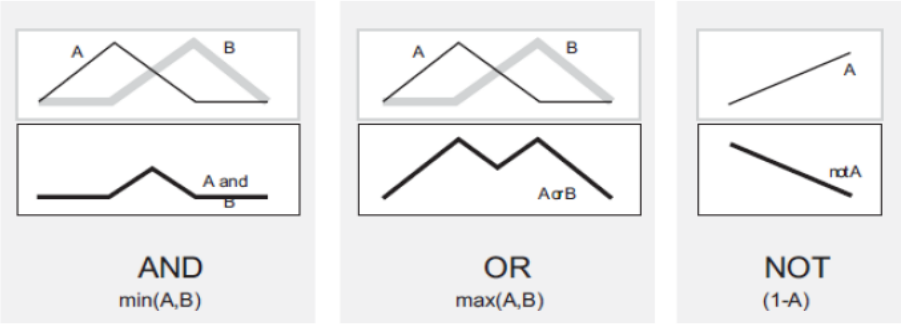

Figure 3.4: Fuzzy logical operations

If-Then Rules

In fuzzy inference process, parallel If-Then rules form the deducing mechanism which indicates how to project input variables onto output space. A single fuzzy If-Then rule follows the form

If *x* is *A*, Then *y* is *B*

The first If-part is called the antecedent, where *x* is input variable. The rest Then-part is called the consequent, and *y* is output variable. The reason why If-Then conditional statements are universally applicable is because both *A* and *B* are linguistic values, or adjectives in most cases, and this form of conditional statement works the concordant way with human judgment. For example, an appropriate If-Then rule might be "IF *material hardness* is *hard*, THEN *cutting speed* is s*low*". *A* can be regarded as fuzzy set and defined by specific membership function, and *B* can be a fuzzy set with respect to input *x* depending on specific fuzzy inference method. In the antecedent, the If part is aimed at working out the membership value of input variable *x* corresponding to fuzzy set *A*. While in the consequent, the Then-part assigns a crisp value back to the output variable *y*.

3.3.2. Fuzzy Inference [28], [29]

Fuzzy inference is the process of mapping the given input variables to an output space via fuzzy logic based deducing mechanism which is comprised by If-Then rules, membership functions and fuzzy logical operations. Because the form of If-Then rule fits in human reasoning, and fuzzy logic approximates to people's linguistic habits, this inference process projecting crisp quantities onto human language and promptly yielding a precise value as result is widely adopted.

Fuzzification and fuzzy reasoning

The fuzzy logic process starts with obtaining some crisp values from user input or sensor on the system that is being controlled. The first thing that has to be done is to fuzzify these crisp numbers into fuzzy numbers. This process is of course accomplished by use of the appropriate MFs to which the crisp numbers belong.

Once all of the inputs are fuzzified, the rules need to be applied to the new fuzzy numbers. In fuzzy logic, all rules are executed in parallel; thus no rules in the rule base are ever not evaluated.

Defuzzification

Once the fuzzy reasoning is complete, a fuzzy output can be produced. However, this output is only meaningful to humans and not to computers and other classical logic equipment. Thus, a method is necessary to transform the resulting fuzzy output into a crisp numerical output. There are several methods to accomplish this. Some of the methods used in the defuzzification process are: Center of Largest Area, Max membership Function, Centroid Method, Weighted Average Method, Mean Max Principle, Center of Sums and First (or last) of Maxima. However, only Centroid method shall be used and discussed next.

Centroid Method: is also called center of area, center of gravity. This is the most popular defuzzification method. This method finds the center of mass of the output region; its algebraic expression [28]:

$$y_{COG} = \frac{\int \mu_A(x) x dx}{\int \mu_A(x) dx} \quad \text{..... 3.32}$$

Where, y_{COG} is defuzzified value and $\mu_A(x)$ is the membership of x in fuzzy set A.

The entire process is shown in Figure 3.5. The n inputs on the left are fuzzified by MFs. Then each of the fuzzified values is put into the rules that produce m outputs. Each of these m outputs are combined (there are various methods) and the defuzzification of the resulting area is found. Now the whole process of fuzzification, rule application, and defuzzification can be put together.

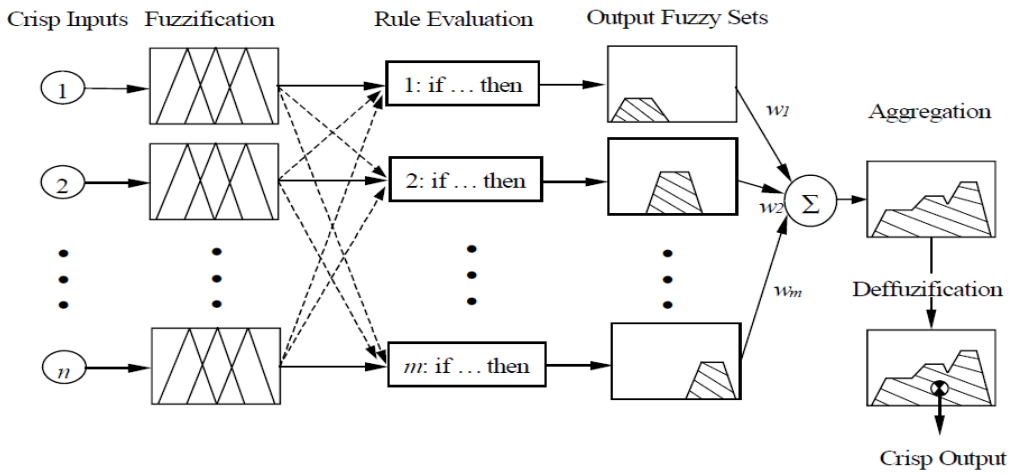

Figure 3.5 Fuzzy Logic from input to output [29]

CHAPTER FOUR

4. Controller Design and Simulation Setup

4.1. Controller Design

This chapter deals with the design of a proposed controller for four degree of freedom SCARA robot manipulator. SIMULINK MATLAB is used to simulate and evaluate the performance of the proposed controller that applied on the robot. Dynamic model of the four-DOF robot manipulator has been utilized in the synthesis process of control. The general block diagram of the whole system, which is implemented in Simulink, is given in Figure 4.1

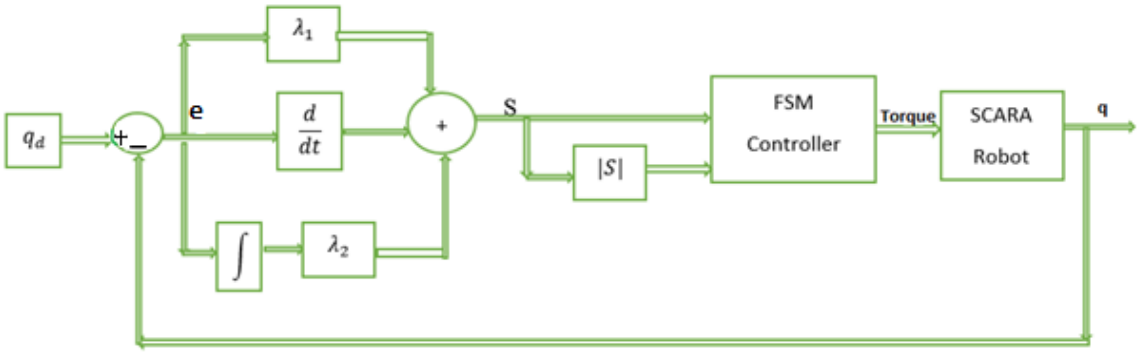

Figure 4.1: Over all structure of the system

Our aim is to track the desired trajectories of the robotic manipulators by using a sliding mode controller. The objective of tracking control is to design a control law for obtaining the suitable input torque τ such the position vector q can track the desired trajectory q_d. Define a position error, velocity error, and acceleration error respectively:

$$\left.\begin{aligned} e &= q_d - q \\ \dot{e} &= \dot{q}_d - \dot{q} \\ \ddot{e} &= \ddot{q}_d - \ddot{q} \end{aligned}\right\} \quad \text{..... 4.1}$$

In order to apply the SMC, the sliding surface is considered as the relation (3.6) that contains the integral part in addition to the derivative term:

$$s = \dot{e} + \lambda e + (\tfrac{\lambda}{2})^2 \int_0^t e dt \text{..... 4.2}$$

Where λ is diagonal positive definite matrix. Therefore, $s = 0$ is a stable sliding surface and $e \to 0$ as$t \to \infty$.

Taking the time derivative of switching function $s(q\ ,t)$

$$\dot{s}(q\ ,t) = \ \ddot{e} + \ \lambda\dot{e} + \left(\frac{\lambda}{2}\right)^2 e$$

$$= \ \ddot{q}_d - \ddot{q} + \lambda\dot{e} + \left(\frac{\lambda}{2}\right)^2 e$$

Substituting equation (2.33)

$$\dot{s}(q\ ,t) = \ddot{q}_d - M^{-1}\left[u - B - C - G\right] + \ \lambda\dot{e} + \left(\frac{\lambda}{2}\right)^2 e \ldots\ldots\ldots\ldots\ldots 4.3$$

Adopting the reaching law given in (3.14)

$$\dot{s} = -Qsgn(s) - Ks \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.4$$

Equating equation (4.3) and equation (4.4); and solving for the control input, the control input becomes

$$u = M\left[Qsign(s) + Ks + \ \ddot{q}_d + \ \lambda\dot{e} + \left(\frac{\lambda}{2}\right)^2 e\right] + B + C + G \ldots\ldots\ldots\ldots\ldots 4.5$$

$K = diag[k_1\ \ k_2\ \ k_3\ \ k_4]\ Q = diag[Q_1\ \ Q_2\ \ Q_3\ \ Q_4]$ are diagonal positive definite matrices.

This is the conventional sliding mode control input to the system by reaching law approach. We choose K and Q based on reaching condition. From position tracking error curve and velocity tracking curve Fig. 5.3 to Fig. 5.10, illustrates that the tracking accuracy is guaranteed. But, the control output chattering problem exists as shown in Fig. 5.1 and Fig. 5.2.

To attenuate chattering problem, we introduce an exponential function in the reaching law which smooth sign function in (4.4). The reaching law becomes.

$$\dot{s} = -Qexp\left(-\frac{\alpha}{|s|}\right)sgn(s) - Ks \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.6$$

The control input becomes

$$u = M\left[Q\ exp\left(-\frac{\alpha}{|s|}\right)sign(s) + Ks + \ \ddot{q}_d + \ \lambda\dot{e} + \left(\frac{\lambda}{2}\right)^2 e\right] + B + C + G \ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.7$$

From the simulation result in Fig 5.19 and Fig 5.20 we observed that chattering phenomena is completely removed from the input torque. But, the magnitude of error tracking is increased. To solve this problem we introduce the fuzzy logic technique to tune the gain k in reaching law.

Fuzzy Logic Approach to Tune Gain K

As mentioned earlier, using a high gain Q in SMC can reduce reaching time to the sliding surface, but increasing the gain Q causes the increment of the oscillation of input torque around the sliding surface. This problem is solved by introducing exponential function in reach law. Applying exponential function in the reaching law, caused to increase the tracking error. To overcome such drawback, in this section we apply fuzzy logic to tune gain k to achieve a desirable performance.

The first term in equation (4.6) is constant rate and the second term is proportional rate. Clearly, due to the second term the state is forced to reach the switching manifold faster when s is large. Secondly, from equation (4.7), a large value k will cause extremely high gain control u when $|s|$ is large. On the other hand, if k is too small, hence, the reaching time will be too long. From this qualitative discussion, we can draw the conclusion as follow.

A small value k can be selected when $|s|$ is large, to keep the control input not high and the reaching rate fast. A large value of k can be selected when $|s|$ is small in the neighbor of the switching surface $s = 0$, to maintain reaching rate fast. Therefore, if this gain can be tuned based on the distance of the states to the switching surface, a more acceptable performance (fast reaching time and small error) can be achieved. This can be done by using fuzzy logic. We shall fuzzify the relationship between the gain k and the $|s|$ using fuzzy logic. The larger $|s|$ is the smaller must be the value of k to keep the control input not high. The smaller $|s|$ is the larger must be the value of k to commence sliding as fast as possible [30]. Thus to have a value of k which can maintain a balance between the two facts, a fuzzy controller rule base is proposed for tuning "k". One -input one-output fuzzy system is designed for this application. Input taken is $|s|$ while output is "k". Rule base used is given in Table (1). Input $|s|$ has five membership functions and output k has also five membership functions.

Five fuzzy sets named $VS,\ S,\ M,\ L,\ VL$ respectively for very small, small, medium, large and very large are chosen to fuzzify the sliding surface.

Similarly the proportional rate gain k is fuzzified into five fuzzy sets named $VS,\ S,\ M,\ L,\ VL$ respectively for very small, small, medium, large and very large.

The rule tables for the controller are shown below

Table 4.1 Fuzzy Rules for K

$\|s\|$	VL	L	M	S	VS
K	VS	S	M	L	VL

The membership function used here is Gaussian function defined in (3.21) because of its smoothness that approximate input precisely.

$$\mu(s_i) = \exp\left[-\frac{1}{2}\left(\frac{s_i - c_i}{\delta_i}\right)^2\right] \qquad i = 1, \ldots m \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.12$$

Membership functions of input for each joint and output are shown in Figs. 4.2 to Fig. 4.6.

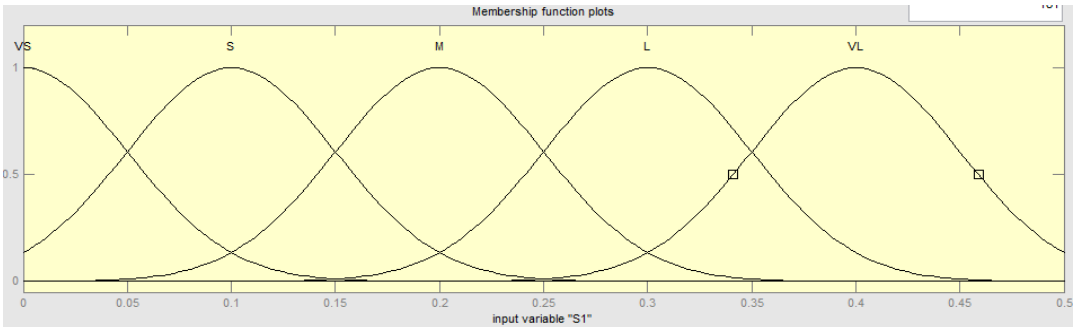

Figure 4.2 membership functions for s_1

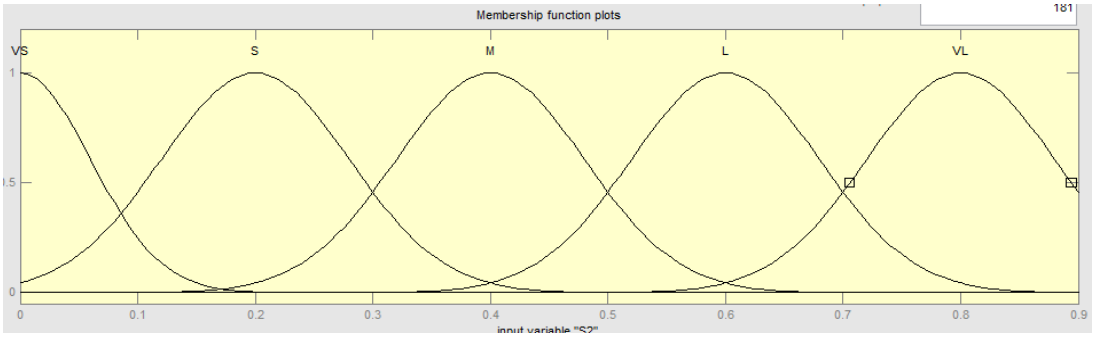

Figure 4.3 membership functions for s_2

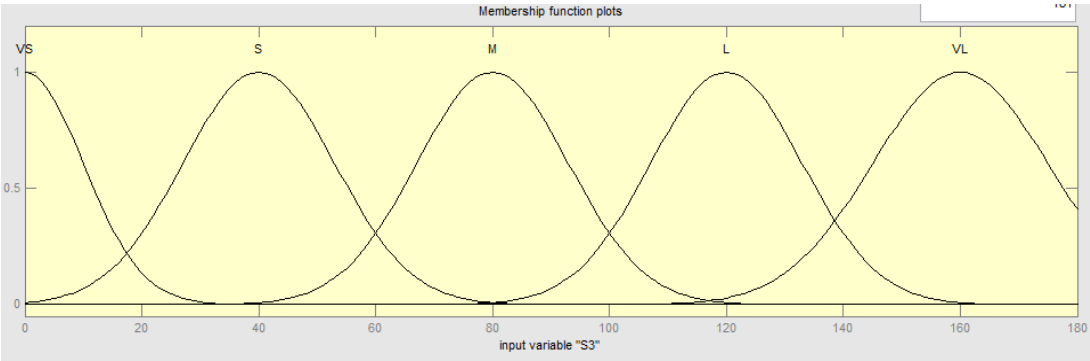

Figure 4.4 membership functions for s_3

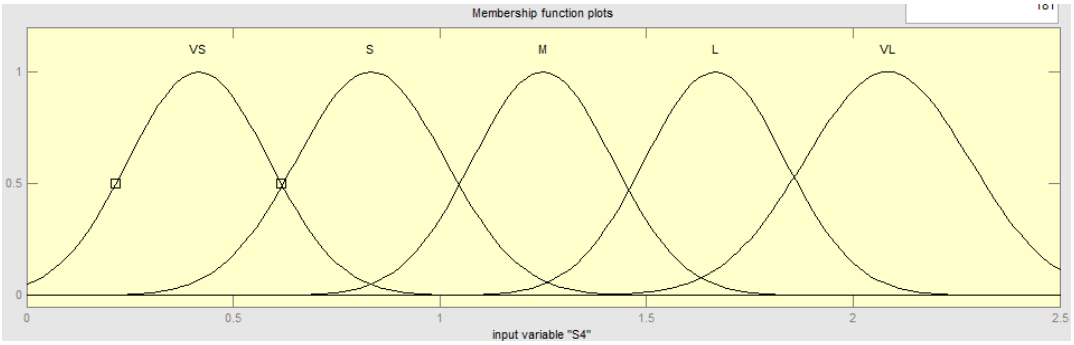

Figure 4.5 membership functions for s_4

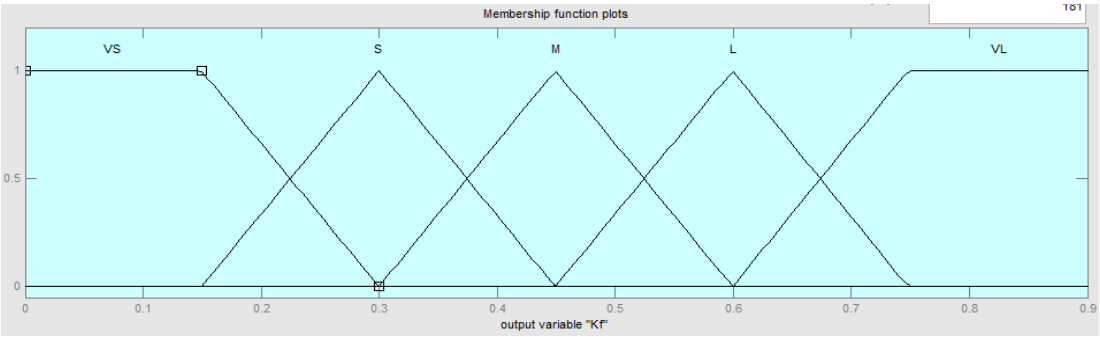

Figure 4.6 membership functions for output K_f

The rules used to map the input and output fuzzy sets are as follows:

Rule1: if $|s|$ is VL, then K is VS

Rule2: if $|s|$ is L, then K is S

Rule3: if $|s|$ is M, then K is M

Rule4: if $|s|$ is S, then K is L

Rule5: if $|s|$ is VS, then K is VL

Where the membership functions of the fuzzy sets VS, S, M, L, VL are defined as $\mu_i^l(x), \mu^l(k)$ in our case $x = |s|$. The fuzzy system in (3.32) is rewritten as [27]

$$y_{CA} = \frac{\sum_{l=1}^{M} k^l \{\prod_{i=1}^{n} \mu^l(x)\}}{\sum_{l=1}^{M} \{\prod_{i=1}^{n} \mu^l(x)\}} = k_f \quad \text{..... 4.8}$$

Where $l = 1, \dots, M$ denote M fuzzy rules and $i = 1, \dots, n$ denote n antecedents; k^l is the point at which $\mu^l(k)$ achieves its maximum value (we assume $\mu^l(k) = 1$); x denotes the crisp inputs (in our case $|s_1|,\ |s_2|,\ |s_3|,\ |s_4|$); k_f is one crisp output for specific x. In the case where we have a Gaussian membership function as (3.21); the fuzzy system k_f changes to

$$k_f = \frac{\sum_{l=1}^{M} k^l [\prod_{i=1}^{n} \exp(-\frac{1}{2}\left(\frac{x_i - c_i^l}{\delta_i^l}\right)^2)]}{\sum_{l=1}^{M} [\prod_{i=1}^{n} \exp(-\frac{1}{2}\left(\frac{x_i - c_i^l}{\delta_i^l}\right)^2)]} \quad \text{..... 4.9}$$

In our case we only have one antecedent $(n = 1)$ for every rule given above, the fuzzy output k_f in (4.9) becomes

$$k_f = \frac{\sum_{l=1}^{M} k^l [\exp\left(-\frac{1}{2}\left(\frac{x - c^l}{\delta^l}\right)^2\right)]}{\sum_{l=1}^{M} [\exp\left(-\frac{1}{2}\left(\frac{x - c^l}{\delta^l}\right)^2\right)]} \quad \text{..... 4.10}$$

Where $l = 1,2, \dots, 5$

The output of the fuzzy system is denoted by k_f; for applying this gain to the control input, it normalized as

$$k_{fuz} = N k_f \quad \text{..... 4.11}$$

N can be selected by trial and error such that the reaching condition is satisfied.

Now, incorporating fuzzy tuned gain k_{fuz} in the reaching law defined in equation (4.7), the new reaching law becomes:

$$\dot{s} = -Qexp(-\frac{\alpha}{|s|})sgn(s) - k_{fuz}s \quad \text{..... 4.12}$$

Where $k_{fuz} \in [K_1, K_2], 0 < K_1 < K_2$ [25]

Following the same procedure as did in previous section, the control input becomes:

$$u = M\left[Qexp(-\frac{\alpha}{|s|})sign(s) + k_{fuz}s + \ddot{q}_d + \lambda_1\dot{e} + \lambda_2 e\right] + B + C + G \quad \text{.....4.13}$$

Stability Analysis

The task of the proposed control law is to track the system trajectory and guarantee the global stability of the closed-loop system. That is, using this control, the system trajectory approaches the switching surface in the finite time, and the system carries out the sliding motion.

***Theorem* 1:** For the nonlinear system (2.36) and the switching function (4.4), the system trajectory will approach the switching surface in the finite time, and the system will fulfill the sliding motion if the control law (4.12) is used.

Proof: We construct a Lyapunov's function

$$V = \frac{1}{2}s^2 \quad \text{..... 4.14}$$

Differentiate V with respect to time,

$$\dot{V} = s\dot{s} \quad \text{..... 4.15}$$

Substituting equation (4.11) in equation (4.14), we have

$$\dot{V} = s\left[-Qexp(-\frac{\alpha}{|s|})sgn(s) - k_{fuz}s\right]$$

$$= -(Q|s|exp\left(-\frac{\alpha}{|s|}\right) + k_{fuz}s^2) \quad \text{..... 4.16}$$

Since, $exp\left(-\frac{\alpha}{|s|}\right)$ is positive for all s; Q positive definite matrix and k_{fuz} relation is $k_{fuz} \in [K_1, K_2], 0 < K_1 < K_2$ as given above. It is obvious that,

$$\dot{V} = -\left(Q|s|exp\left(-\frac{\alpha}{|s|}\right) + k_{fuz}s^2\right) < 0, s \neq 0 \quad \text{..... 4.17}$$

Equation (4.16) implies that the reaching control law incorporating exponential function with fuzzy reaching law in (4.11) always forces the nonlinear system (2.36) to approach the switching surface $s = 0$, for $s \neq 0$, $\dot{V} < 0$. Therefore, the system is stable and carries out the sliding motion.

4.2. Simulation Setup

The following parameters are used for simulation.

Table 4.1 SCARA parameters for numerical study [29]

Parameter	Value	Unit
m_1	15	kg
m_2	12	kg
m_3	3	kg
m_4	3	kg
I_1	$0.02087m_1$	kgm^2
I_2	$0.08m_2$	kgm^2
I_3	0.05	kgm^2
I_4	$0.02m_4$	kgm^2
a_1	0.5	m
a_2	0.4	m

In order to check the validity of the above control method, we carry a simulation in MATLAB/Simulink to test our controller's effectiveness and simulation results are given in this section.

MATLAB/Simulink Modeling

The model and controller are built in Simulink using basic Simulink building blocks and embedded MATLAB functions. In this model the MATLAB/Simulink block implements the mentioned SCARA robot manipulator's model, the trajectory generator block generated the desired trajectory as defined by user, the controllers are a SMC and FSMC with PID sliding surface that generates the control output that the torques for joint 1, joint 2, joint 4 and force for joint 3. The overall Simulink model implemented on MATLAB is shown in figure (4.7) and (4.8). The model is used to study the trajectory tracking, chattering phenomena and control efforts.

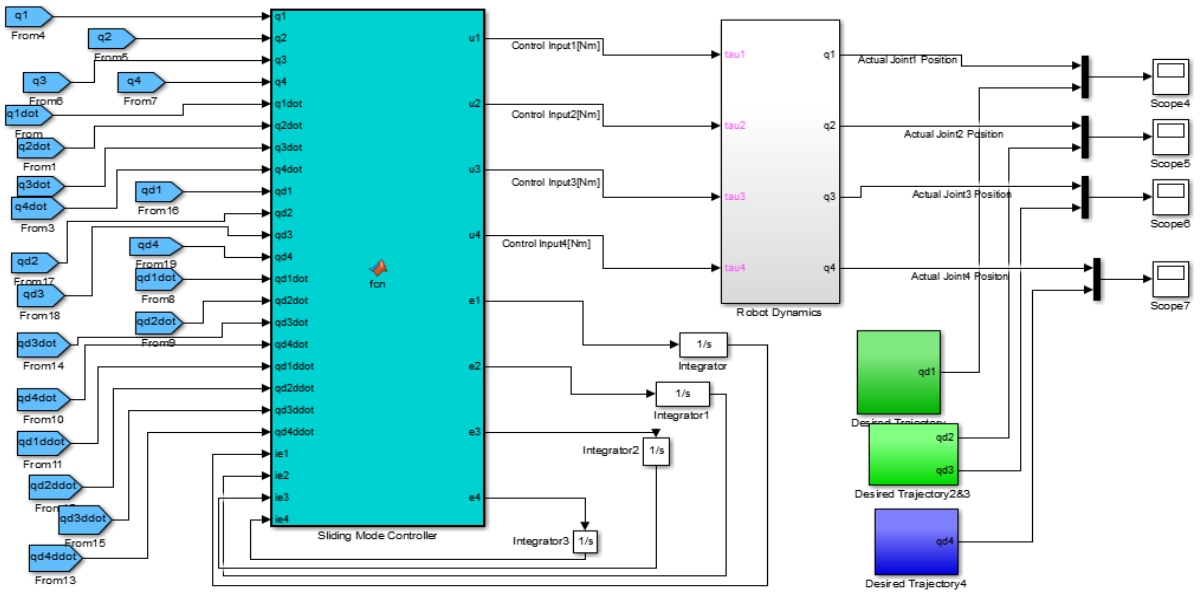

Figure 4.7 Simulink model and sliding mode controller

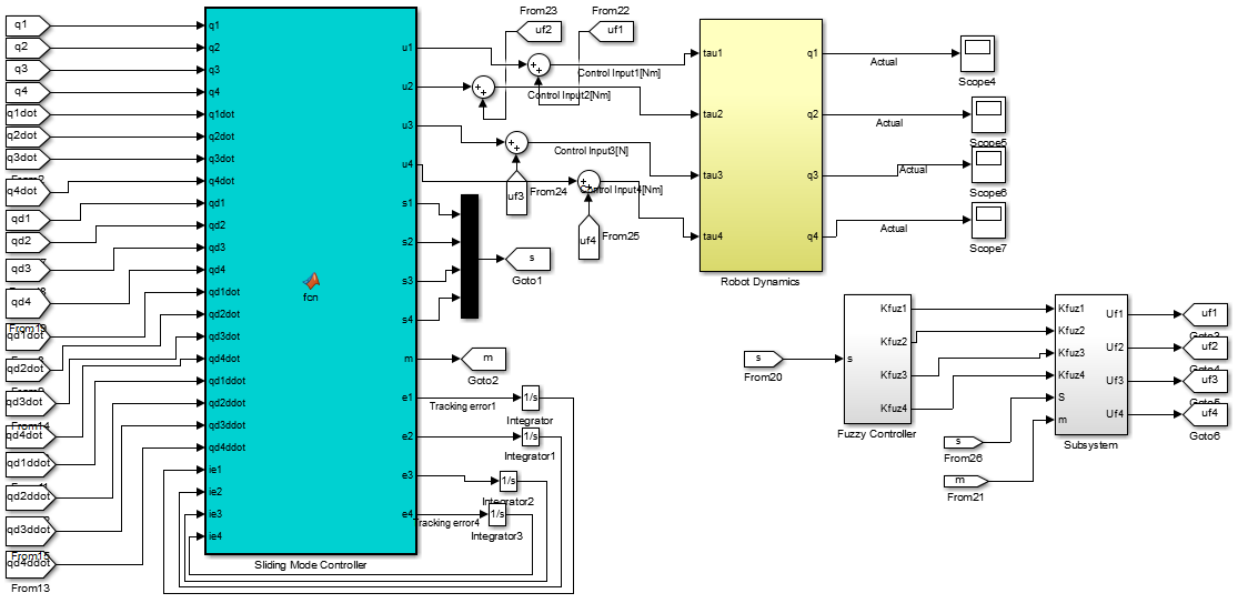

Figure 4.8 Simulink model and fuzzy sliding mode controller

Figure 4.9 shows the Simulink block of the robot manipulator dynamics implemented for equation 2.33

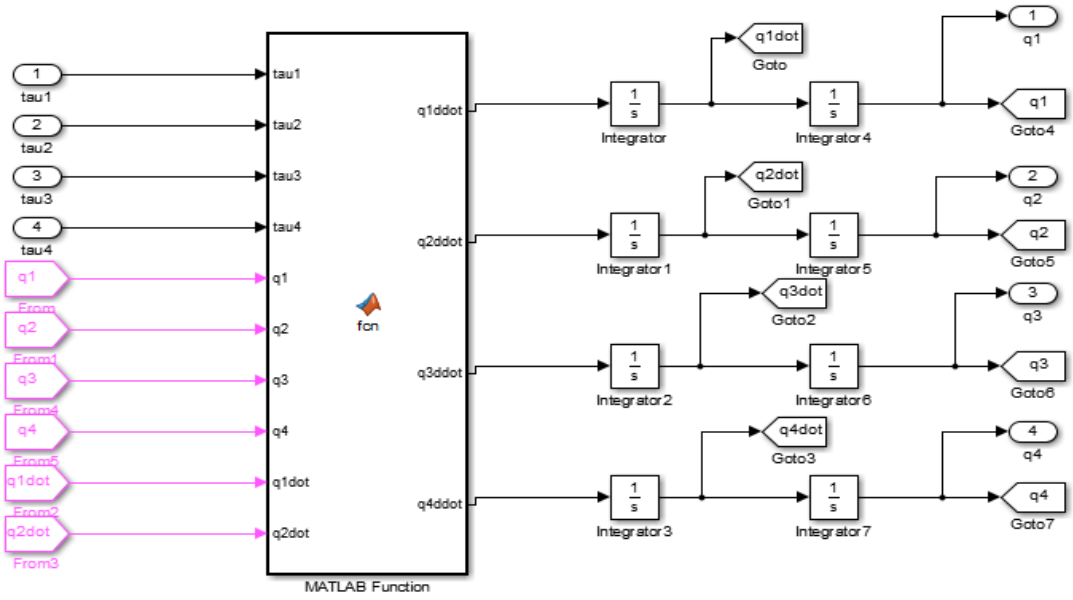

Figure 4.9 *Simulink block for Robot Manipulator Dynamics*

Figure (4.10) shows the block diagram of the subsystem of the fuzzy controller.

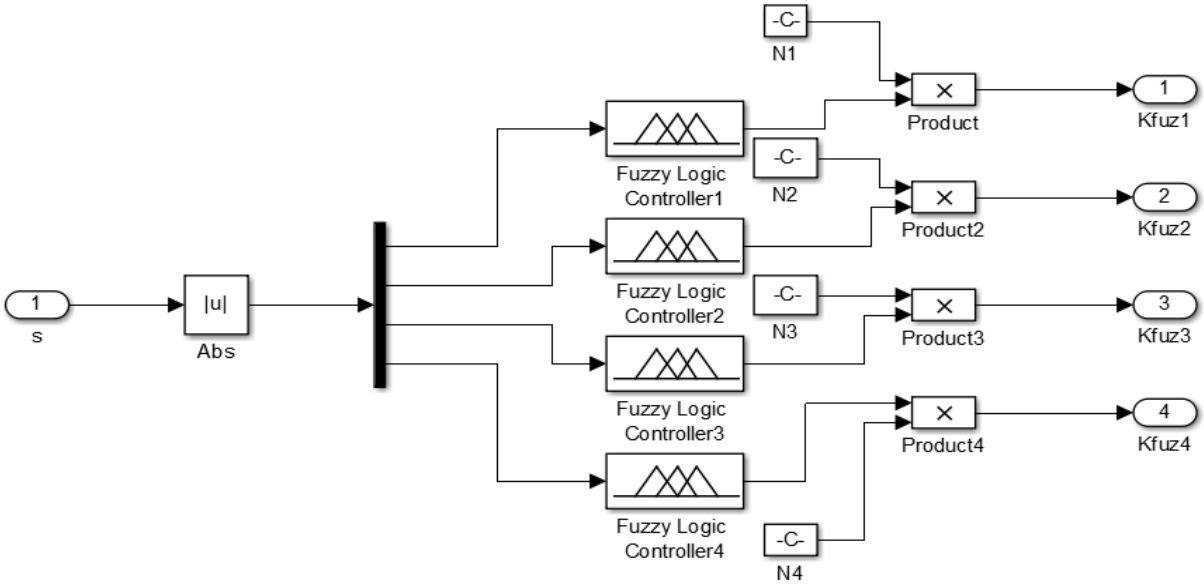

Figure 4.10 Simulink block fuzzy logic controller

CHAPTER FIVE

5. Simulation Results and Discussion

In this section, we examine the performance of the proposed fuzzy sliding mode controller with PID sliding surface through simulations on a 4 DOF SCARA robot manipulator.

Moreover, we also make a comparison between the sliding mode controller, sliding mode controller with exponential function and proposed controller

The design parameters for SMC are given as:

$$\lambda = diag[40, 40, 60, 60]$$

$$Q = diag[20, 15, 28, 8]$$

$$k = diag[60, 20, 13, 11]$$

Such that $\lambda,\ Q\ and\ k$ are diagonal positive definite matrix and selected by trial and error to achieve the best tracking error.

Using these gains and values given in table 4.1 simulation is carried out for SMC. Figures from Fig. 5.1 to Fig. 5.18 shows input torque, position tracking curve, position error tracking curve, velocity tracking curve and velocity error tracking curve when the system is subjected to SMC.

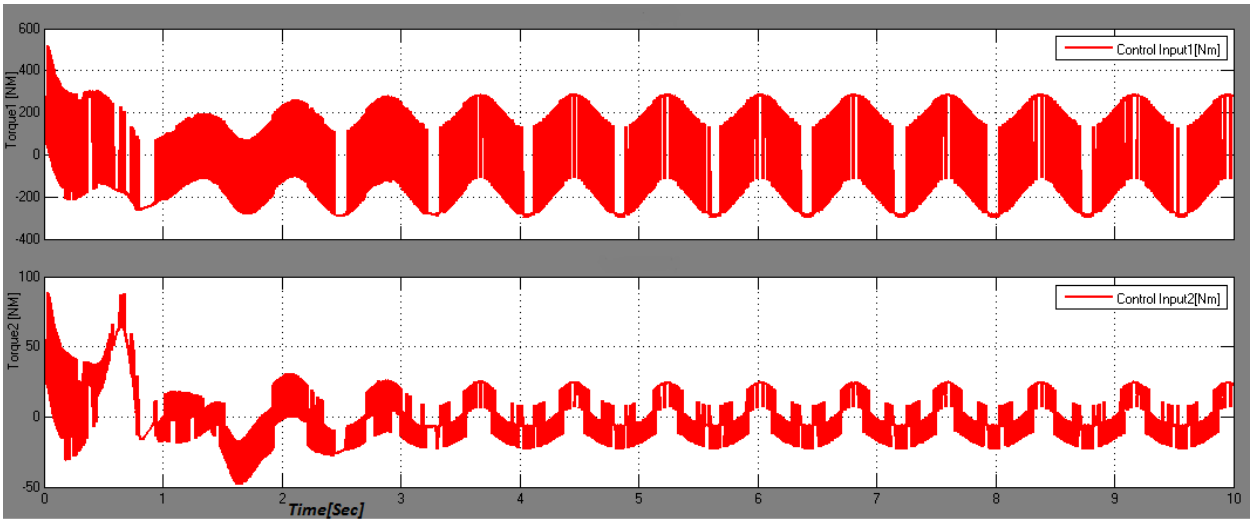

Figure 5.1: Input torque for joint 1 and joint 2 using SMC.

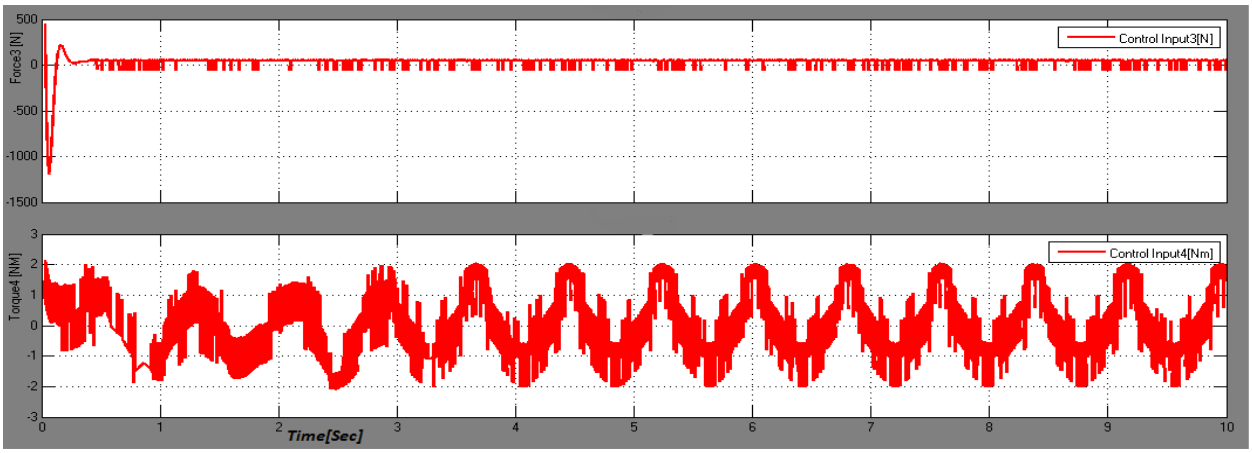

Figure 5.2: Input torque for joint 3 and joint 4 using SMC.

It is clear from Figure (5.1) and (5.2) that SMC is showing higher magnitude of chattering. It is a well-known fact that high frequency or high magnitude of chattering of a sliding control is dangerous which can damage the system when an implementation is done and an actuator has to obey a sliding/ switching control command.

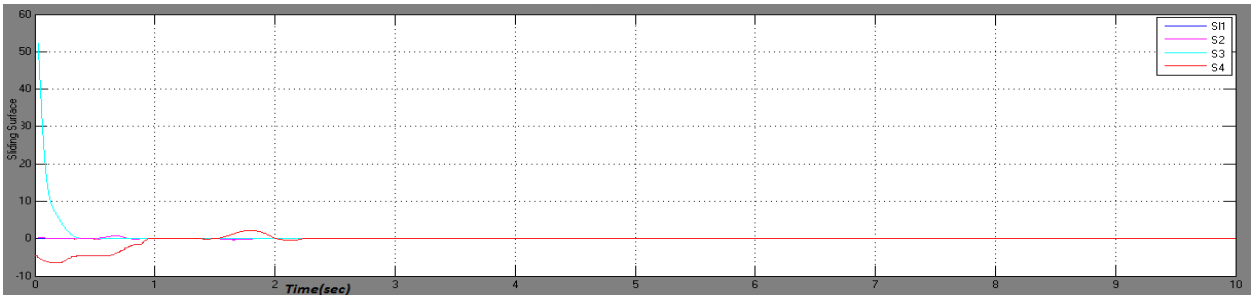

Figure 5.3 Sliding Surfaces using SMC with PID Sliding Surface

Figure 5.4 shows the position tracking response of the first joint when the designed controller allows the joint to follow the desired trajectory. Figure 5.5 represents its respective error plot. It can be seen that the tracking performance of SMC is acceptable as the tracking error is low.

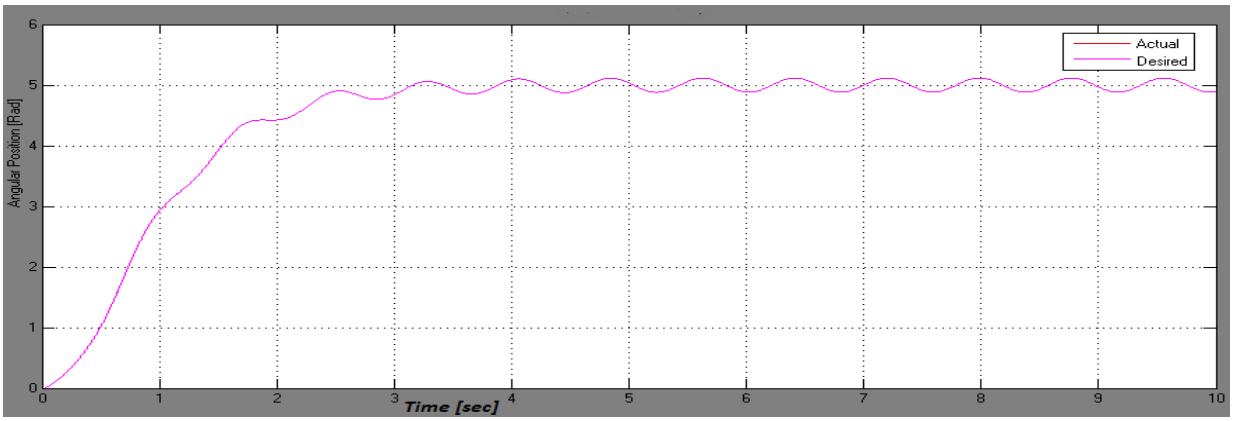

Figure 5.4: *the desired position and actual position of SMC for first Joint rad*

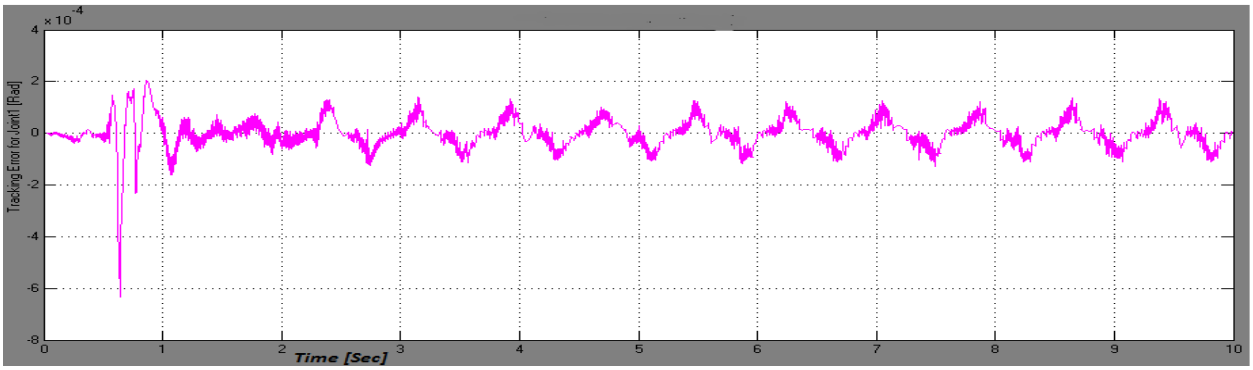

Figure 5.5: *Position Tracking Error Curve of the First Joint rad*

Figure 5.6 shows the position tracking curve of the second joint, the same desired trajectory for the first joint chosen for second and Figure 5.7 represents its tracking error.

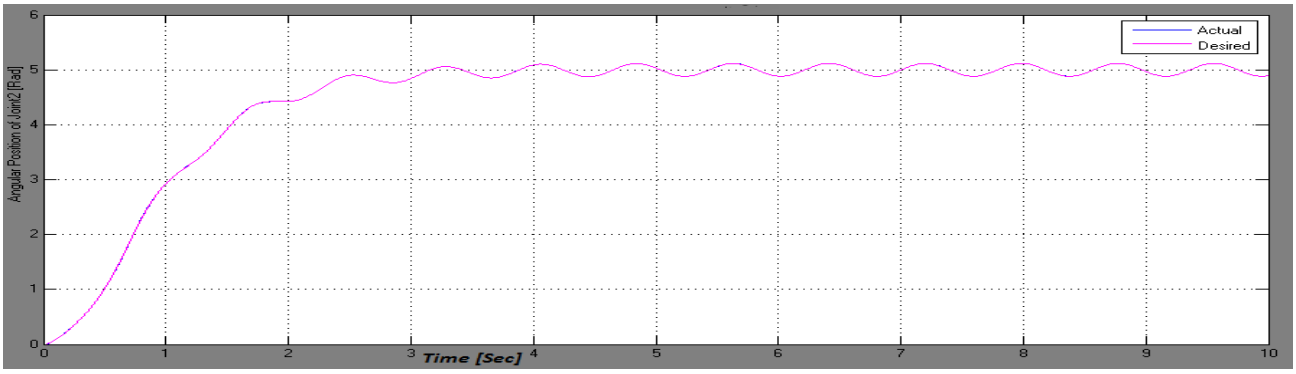

Figure 5.6: *the desired position and actual position of SMC for second Joint rad*

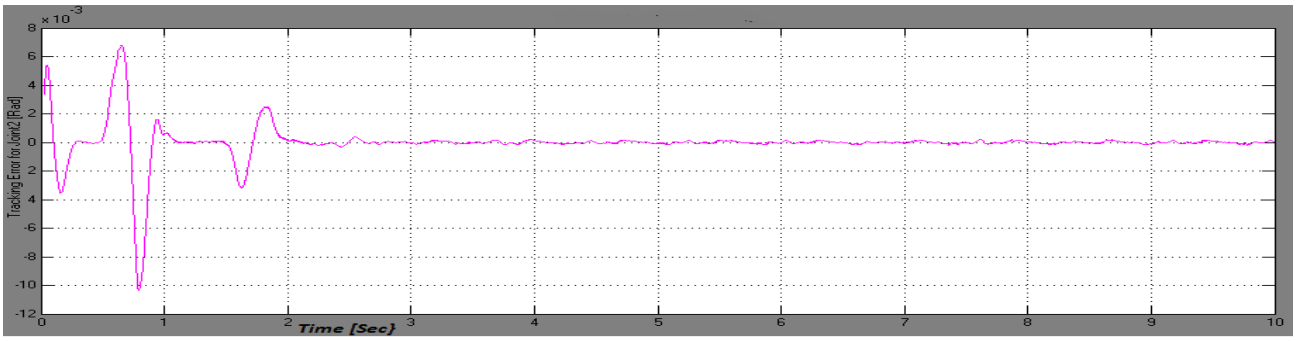

Figure 5.7: *Position Tracking Error Curve of the Second Joint in Rad*

Figure 5.8 shows the position tracking curve of the third joint, a linear function was chosen as a desired trajectory because the motion of third joint is linear. As the figure show, the third joint tracks the desired path accurately, with small tracking error as shown in figure 5.9

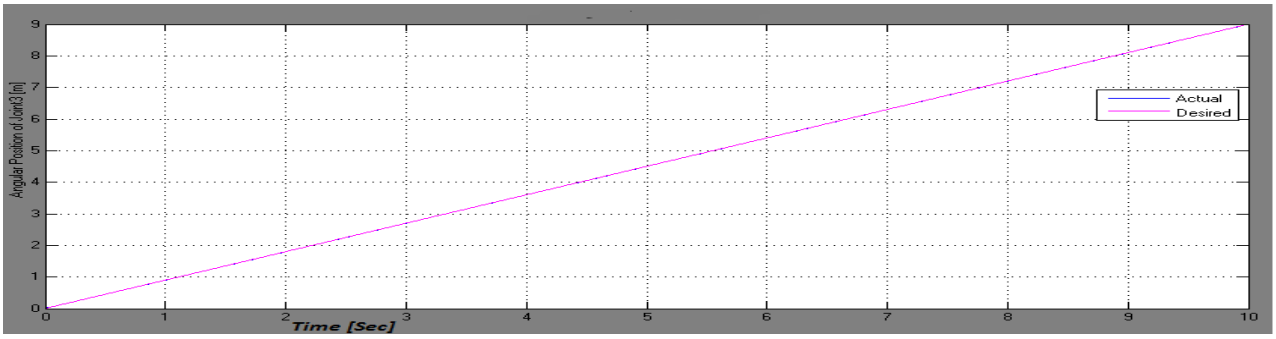

Figure 5.8: *the desired position and actual position of SMC for third Joint in meter*

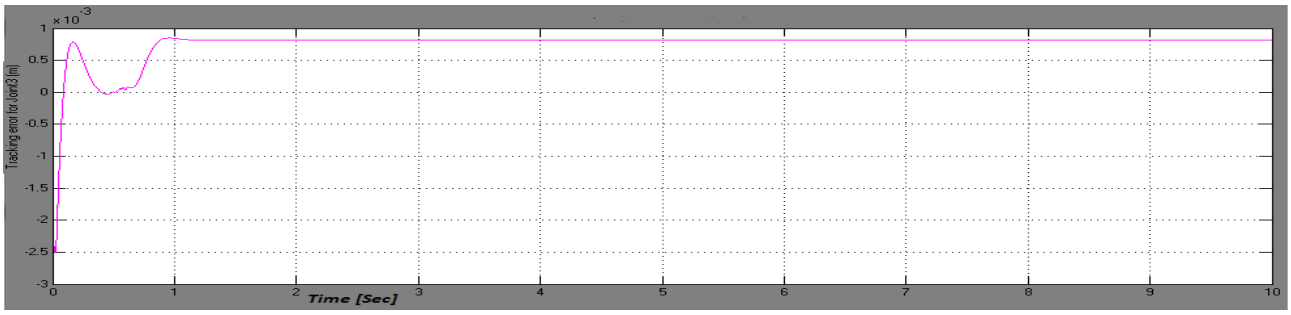

Figure 5.9: *Position Tracking Error Curve of the Third Joint in meter*

Figure 5.10 and 5.11 are the plots of the position tracking and tracking error of fourth joint, respectively. It can be shown that the tracking performance of SMC is good as the tracking error is low.

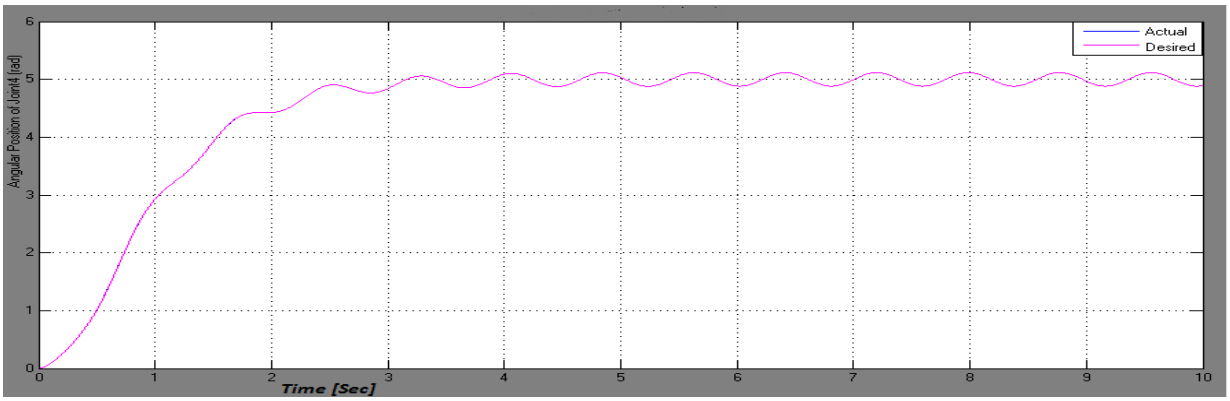

Figure 5.10: *the desired position and actual position of SMC for fourth joint rad*

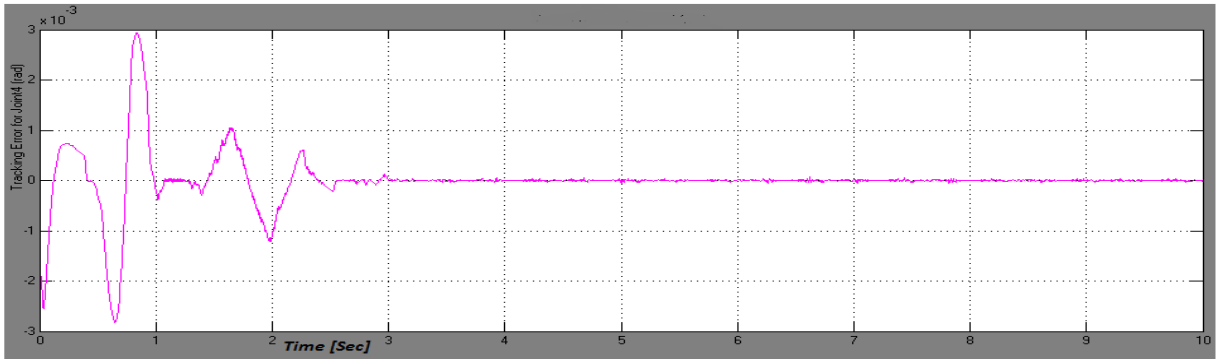

Figure 5.11: *Position Tracking Error Curve of the Fourth Joint in rad*

Figure 5.12 shows the velocity tracking curve of the first joint, the derivative of the angular position trajectory was taken as an angular velocity trajectory. Figure 5.13 is the plot of velocity tracking error curve of the first join in rad/sec.

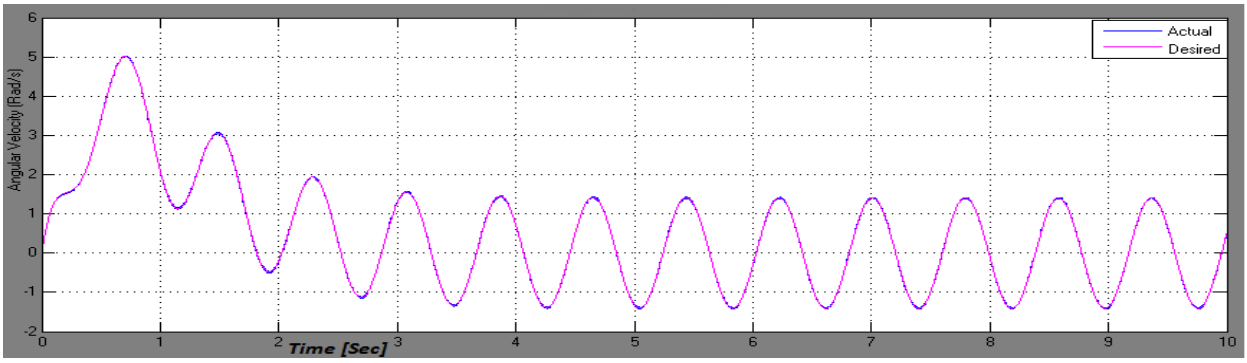

Figure 5.12: *desired velocity and actual velocity of SMC for joint1 in rad/s*

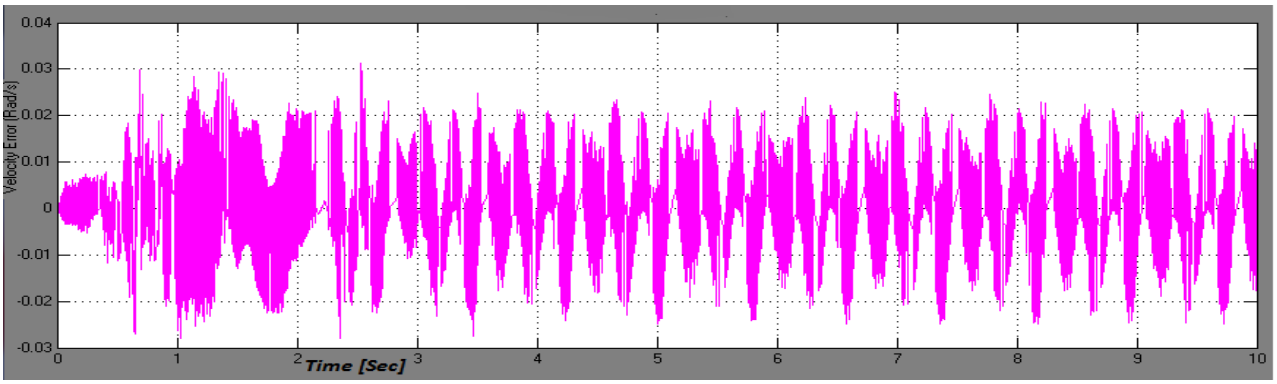

Figure 5.13: *Velocity Tracking Error Curve of SMC for Joint1 in rad/s*

The plot of angular velocity tracking and velocity tracking error curve of second joint in figure 5.14 and 5.15, respectively shows that the designed controller is able to perform satisfactorily.

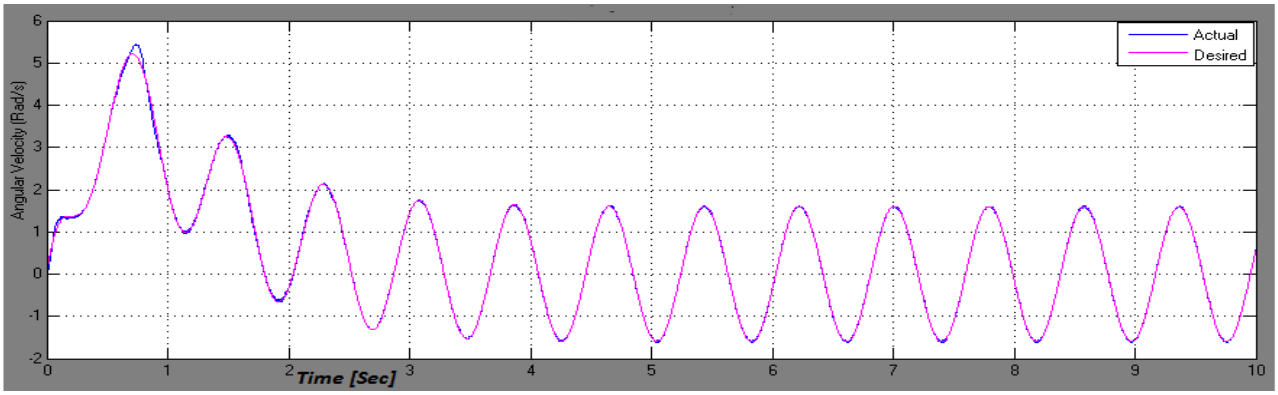

Figure5.14: *desired velocity and actual velocity of SMC for joint2 in rad/s*

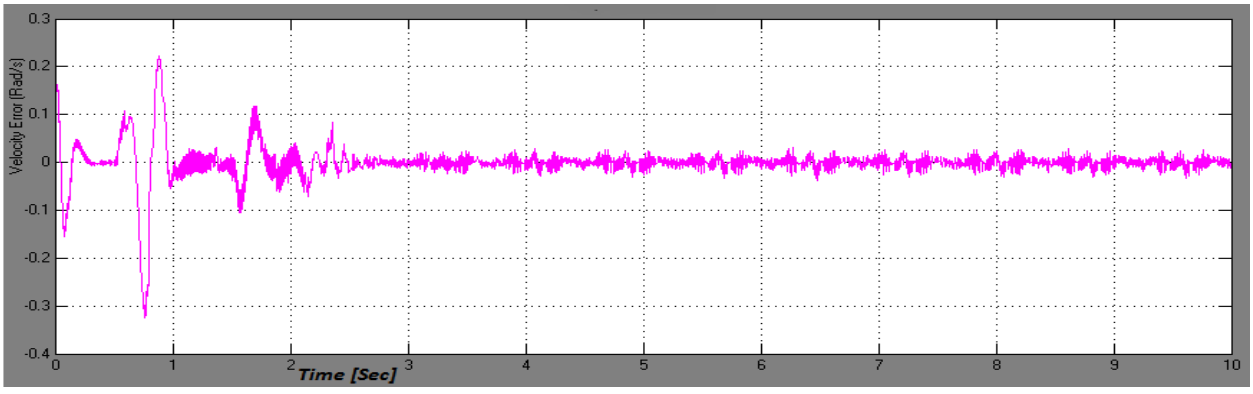

Figure 5.15: *Velocity Tracking Error Curve of SMC for Joint2 in rad/s*

The velocity tracking and velocity tracking error of third joint in m/sec is shown in figure 5.16 and 5.17.

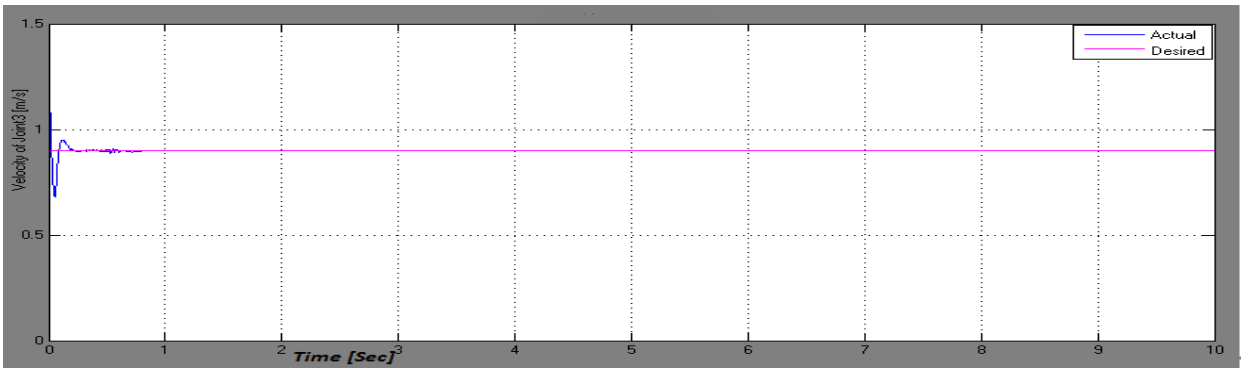

Figure 5.16: *desired velocity and actual velocity of SMC for joint3 in m/s*

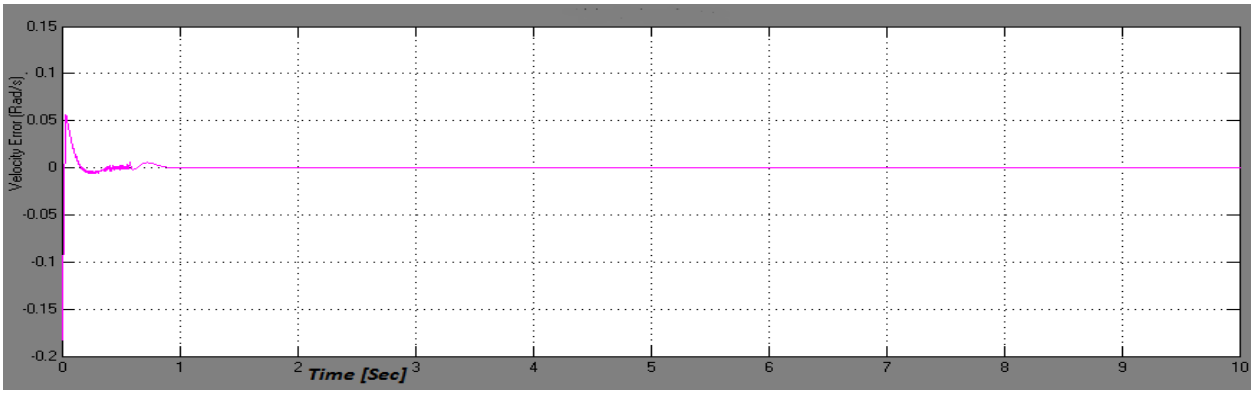

Figure 5.17: *Velocity Tracking Error Curve of SMC for Joint3 in m/s*

Figure (5.18) shows the velocity tracking curve of the fourth joint, the derivative of the desired position was taken as a desired velocity trajectory. The velocity tracking error of the fourth joint is small as shown in figure (5.19)

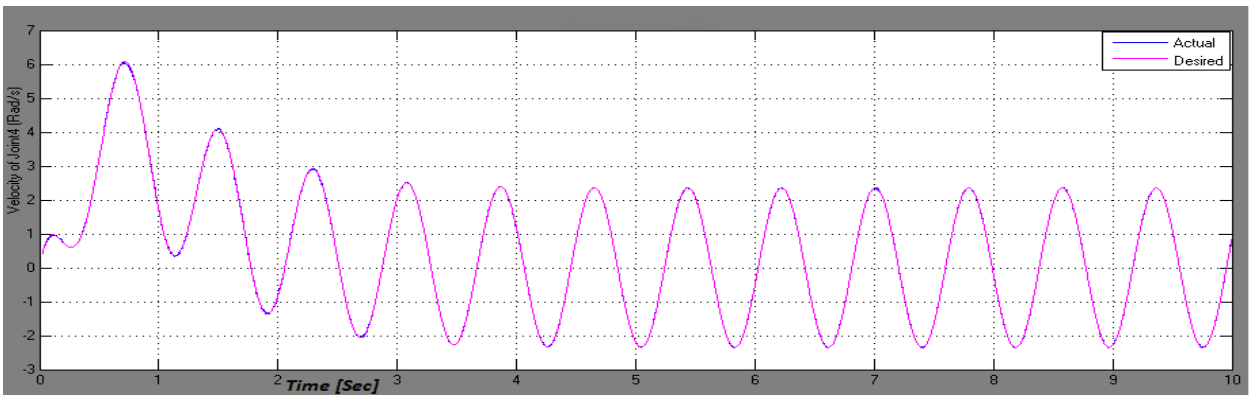

Figure 5.18: *desired velocity and actual velocity of SMC for joint4 in rad/s*

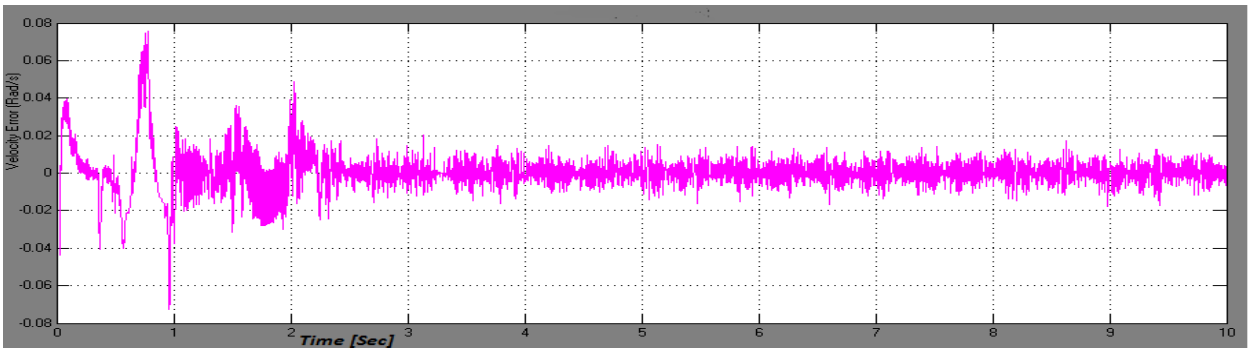

Figure 5.19: *Velocity Tracking Error Curve of SMC for Joint4 in rad/s*

The parameters of SMC with exponential function are

$\lambda = diag[40, 40, 90, 60],\ Q = diag[35, 25, 30, 20], k = diag[20, 30, 11, 3],\ \alpha = 0.8$

The input torques for the manipulator are show in Fig 5.20 and 5.21

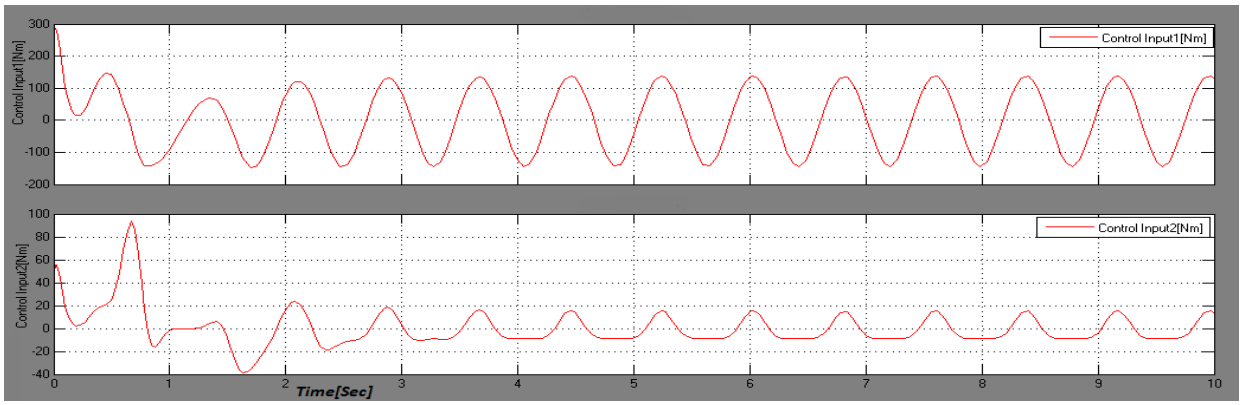

Figure 5.20: Input torque for joint 1 and joint 2 using SMC with exponential function

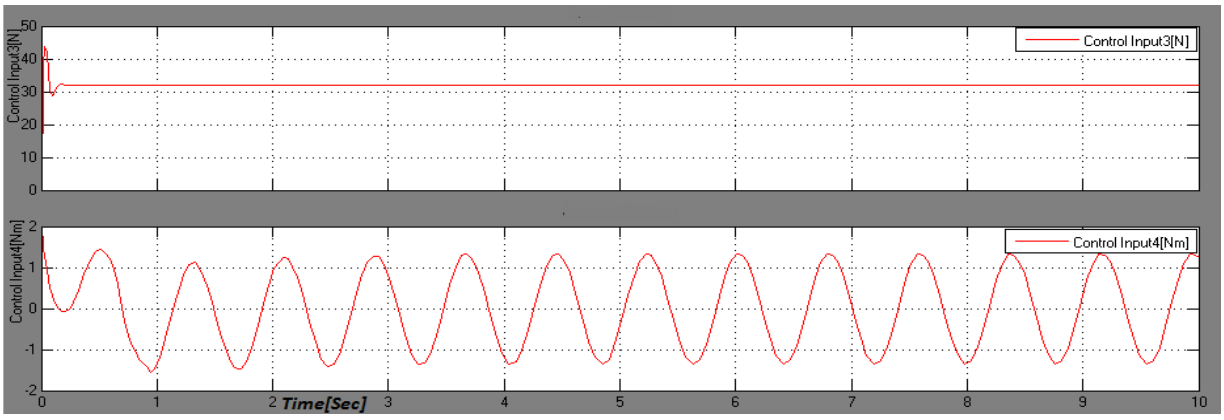

Figure 5.21: Input torque for joint 3 and joint 4 using SMC with exponential function

The trajectory tracking curve for each joint and its respective tacking error plot of the simulation result of SMC with exponential function is shown in from Fig. 5.22 to Fig. 5.29

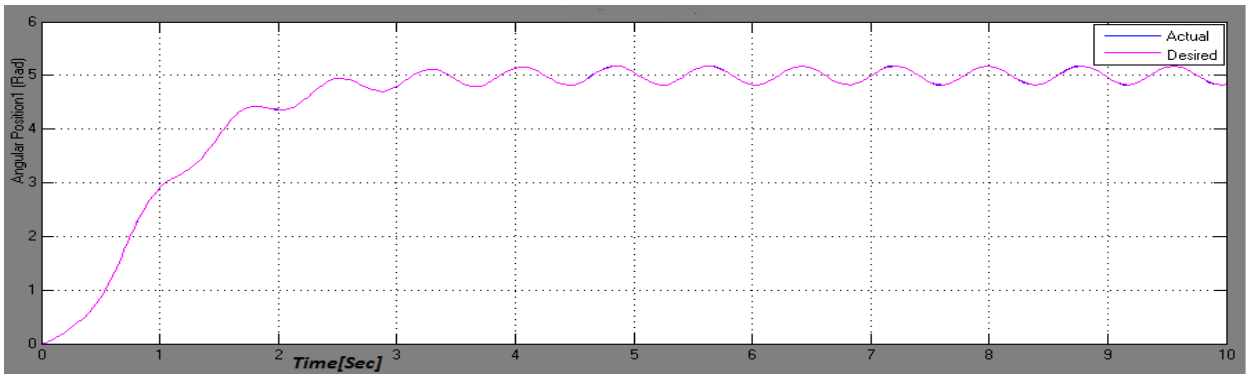

Figure 5.22: *desired and actual position of SMC with exponential function for joint1 in rad*

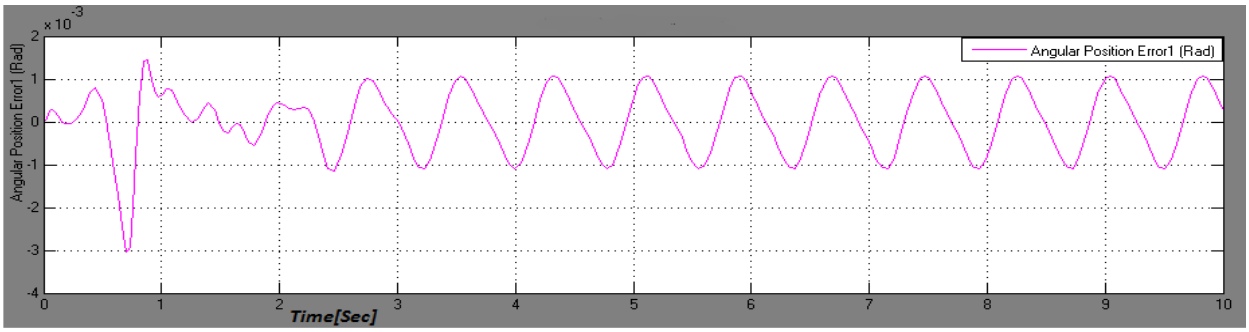

Figure 5.23: *Position Tracking Error Curve of SMC with exponential function for Joint1 in rad/s*

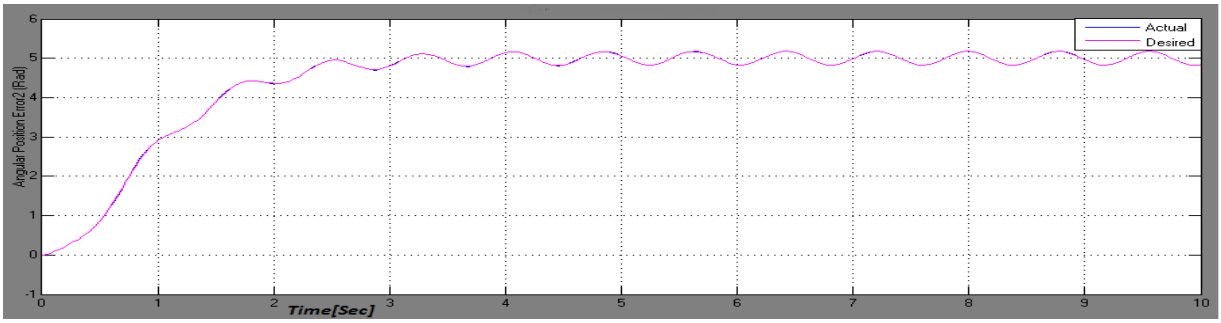

Figure 5.24: *desired and actual position of SMC with exponential function for joint2 in rad*

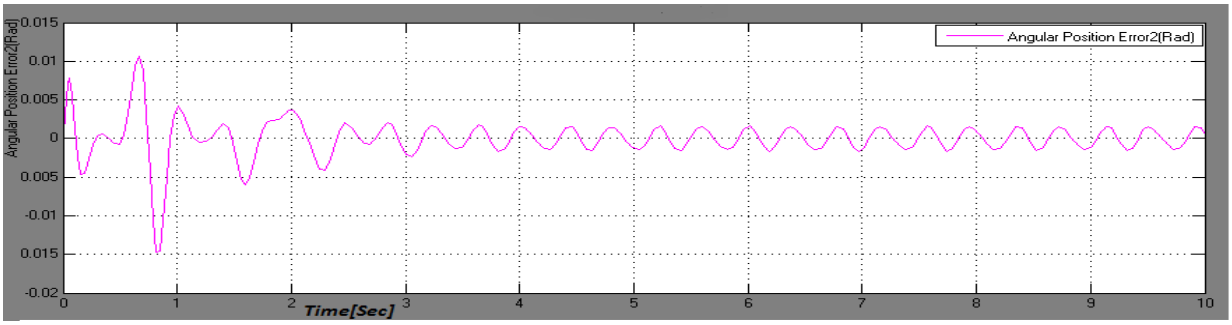

Figure 5.25: *Position Tracking Error Curve of SMC with exponential function for Joint2 in rad/s*

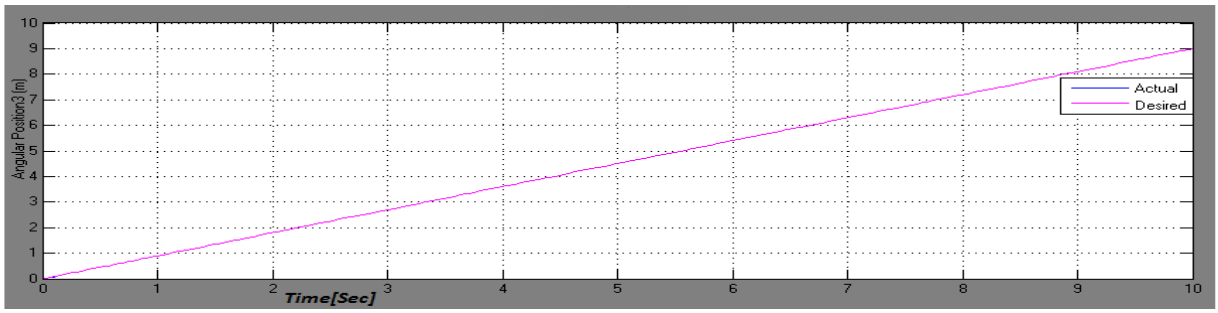

Figure 5.26: *desired and actual position of SMC with exponential function for joint3 in meter*

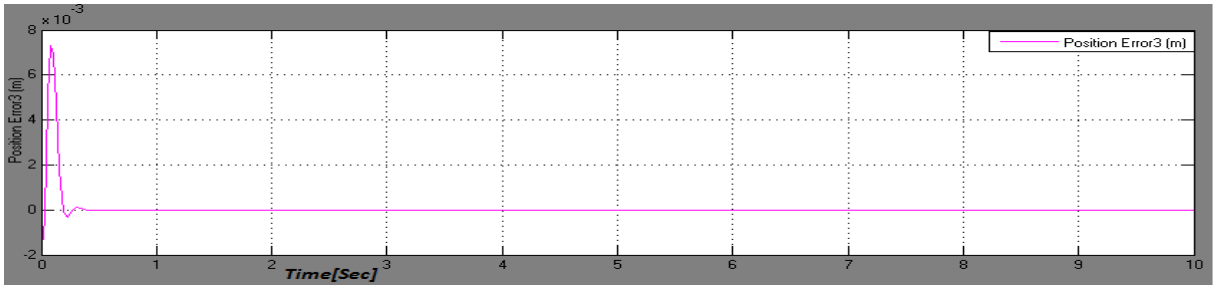

Figure 5.27: *Position Tracking Error Curve of SMC with exponential function for Joint3 in m*

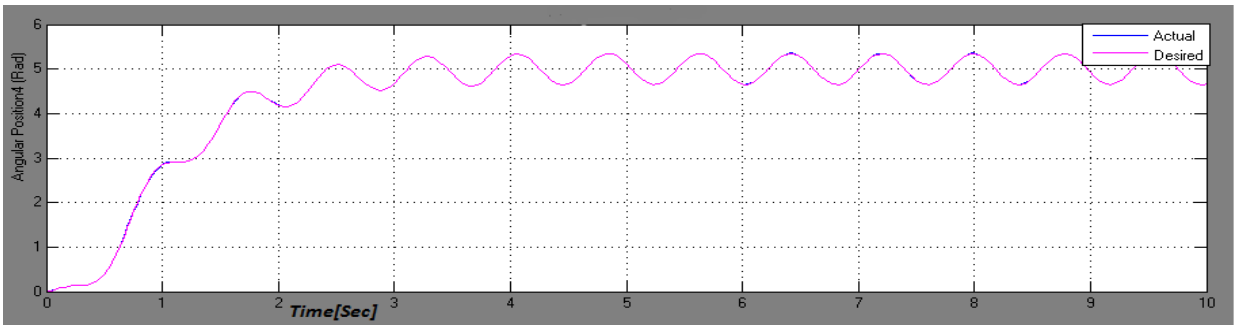

Figure 5.28: *desired and actual position of SMC with exponential function for joint4 in rad*

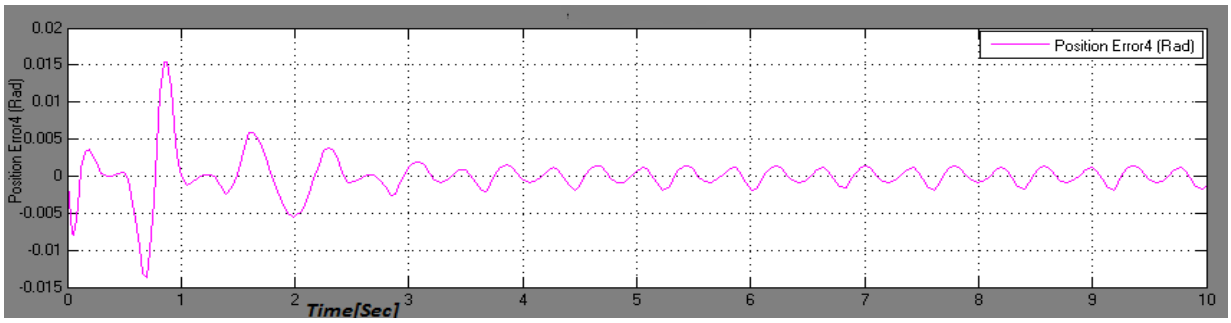

Figure 5.29: *Position Tracking Error Curve of SMC with exponential function for Joint4 in rad/s*

The velocity tracking from joint 1 up to joint 4 and its respective velocity error is given in Fig 5.30 to Fig 5.37

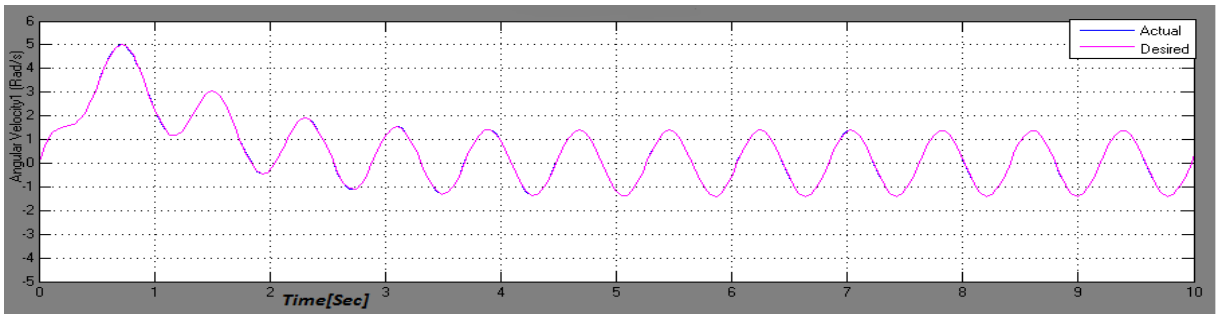

Figure 5.30: *Velocity Tracking Curve of SMC with exponential function for Joint1 in rad/s*

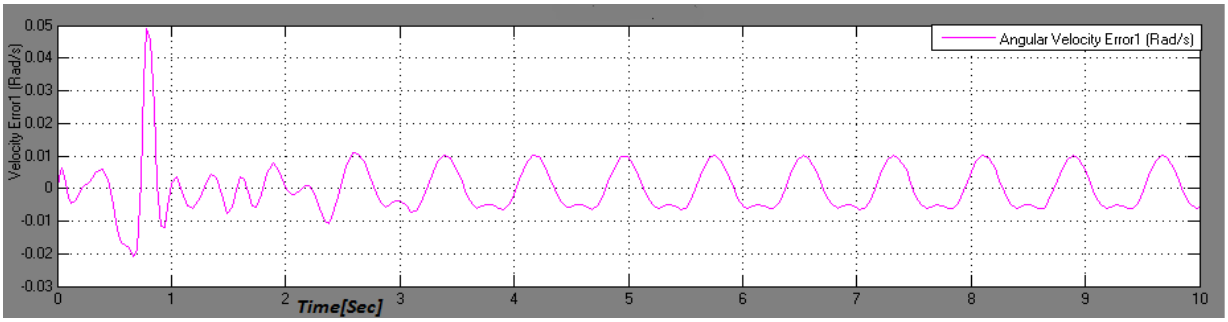

Figure 5.31: *Velocity Error Curve of SMC with exponential function for Joint1 in rad/s*

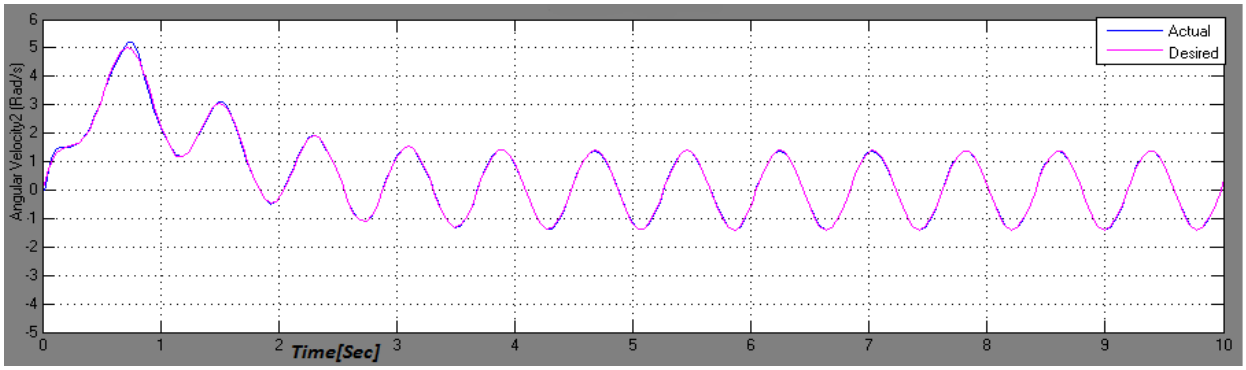

Figure 5.32: *Velocity Tracking Curve of SMC with exponential function for Joint2 in rad/s*

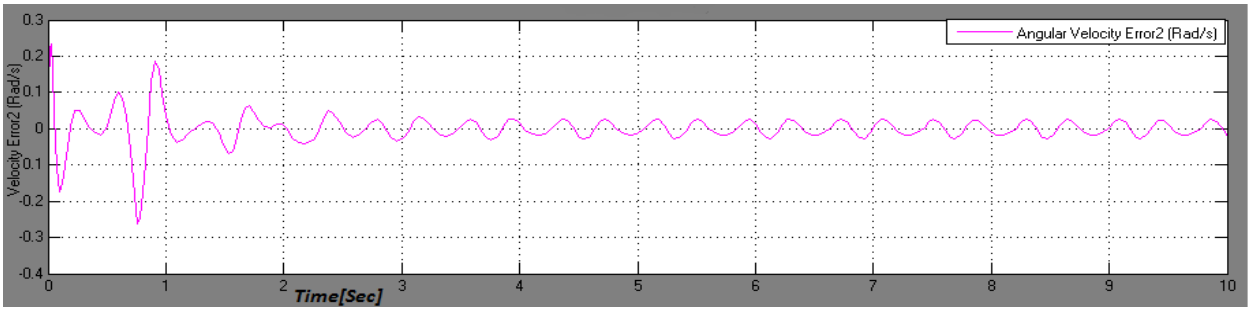

Figure 5.33: *Velocity Error Curve of SMC with exponential function for Joint2 in rad/s*

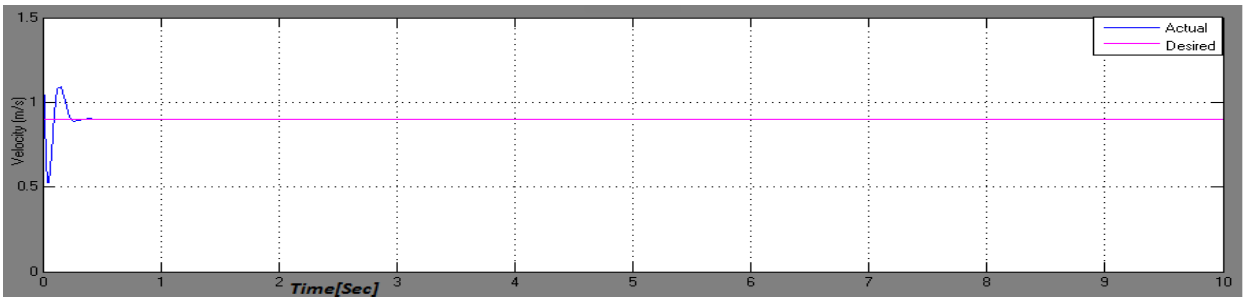

Figure 5.34: *Velocity Tracking Curve of SMC with exponential function for Joint3 in m/s*

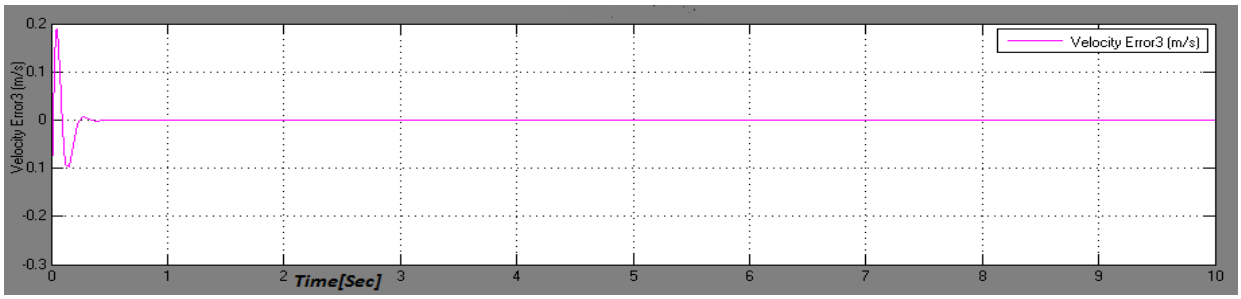

Figure 5.35: *Velocity Error Curve of SMC with exponential function for Joint3 in m/s*

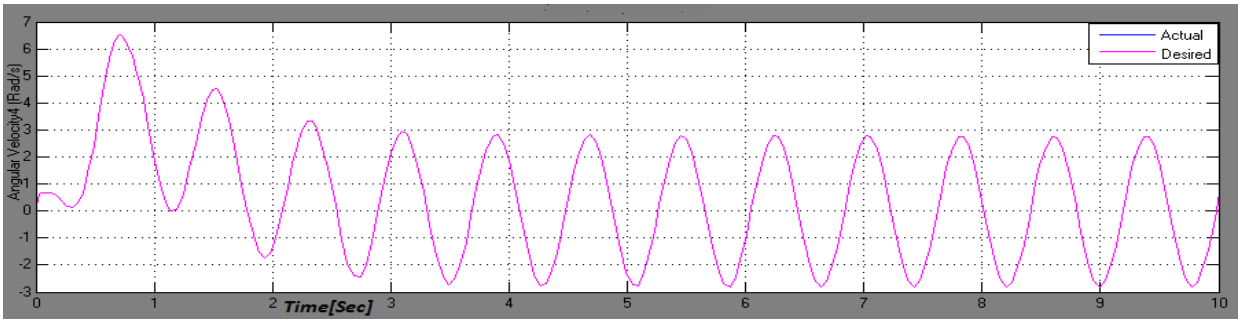

Figure 5.36: *Velocity Tracking Curve of SMC with exponential function for Joint4 in rad/s*

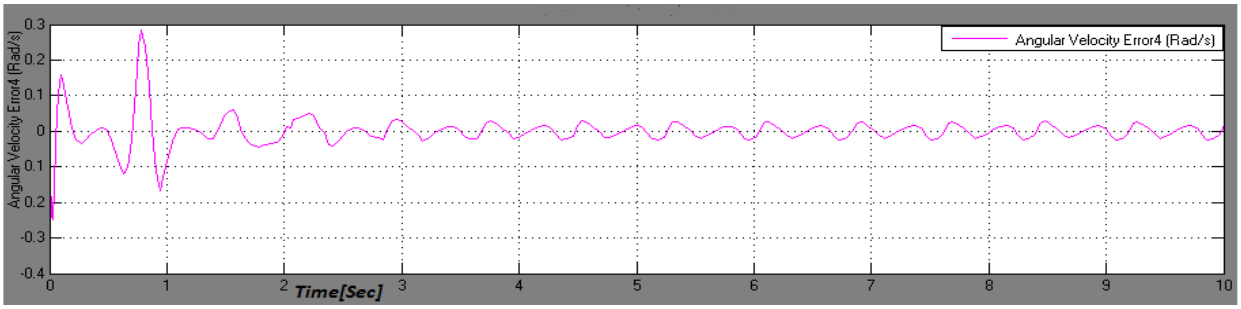

Figure 5.37: *Velocity Error Curve of SMC with exponential function for Joint4 in rad/s*

From the simulation results from Fig 5.20 and fig 5.21 it has been observed that SMC that contain exponential function in reaching mode chattering effects have been completely removed from input torque but the tracking errors became large compared with SMC.

For the comparison of FSMC (with PID sliding surface and exponential function in the reaching law) with the SMC and SMC with exponential function, the design parameters of FSMC with exponential function are given as:

$$\lambda = diag[30, 30, 35, 35\ ]$$

$$Q = diag[35, 30, 33, 10]$$

$$N = diag[120, 120, 80, 65]$$

$$\alpha = 0.8$$

Such that $\lambda,\ Q\ and\ N$ are diagonal positive definite matrices and selected by trial and error to achieve the best tracking error.

Using these gains and values given in table 4.1 as we did for SMC with PID sliding surface and SMC with PID sliding surface and exponential function in reaching law above, simulation is carried out for FSMC with PID sliding surface and exponential function in reaching law. Figs. 5.38 and 5.39 shows input torque of FSMC. We can see that the chattering is eliminated by using exponential function in reaching law and there is significant reduction in input torque.

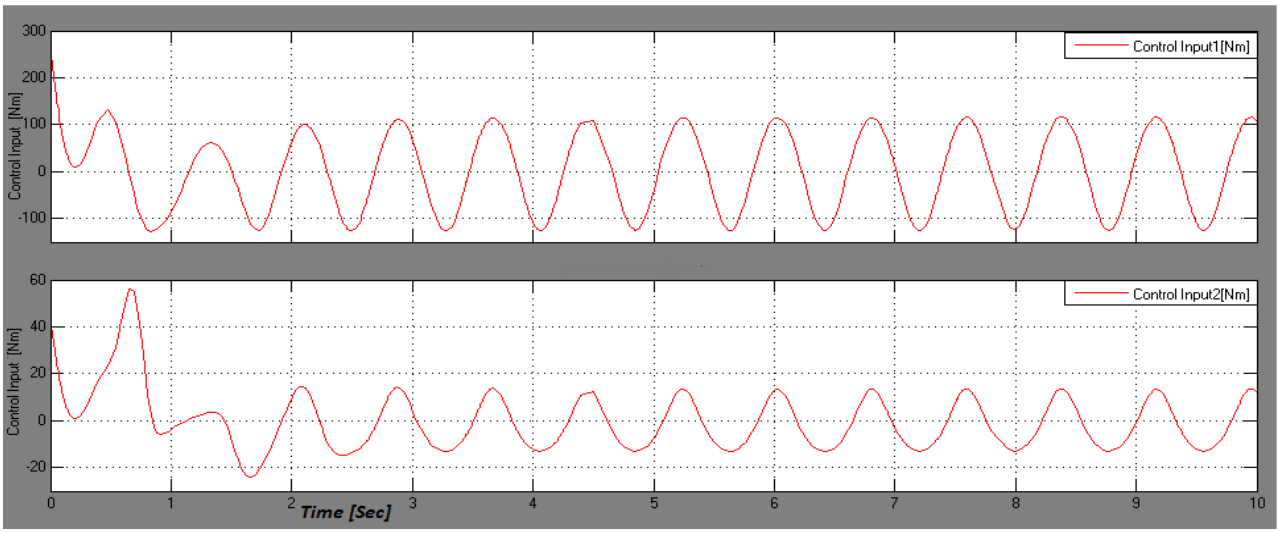

Figure 5.38: Input torque for joint 1 and joint 2 using FSMC

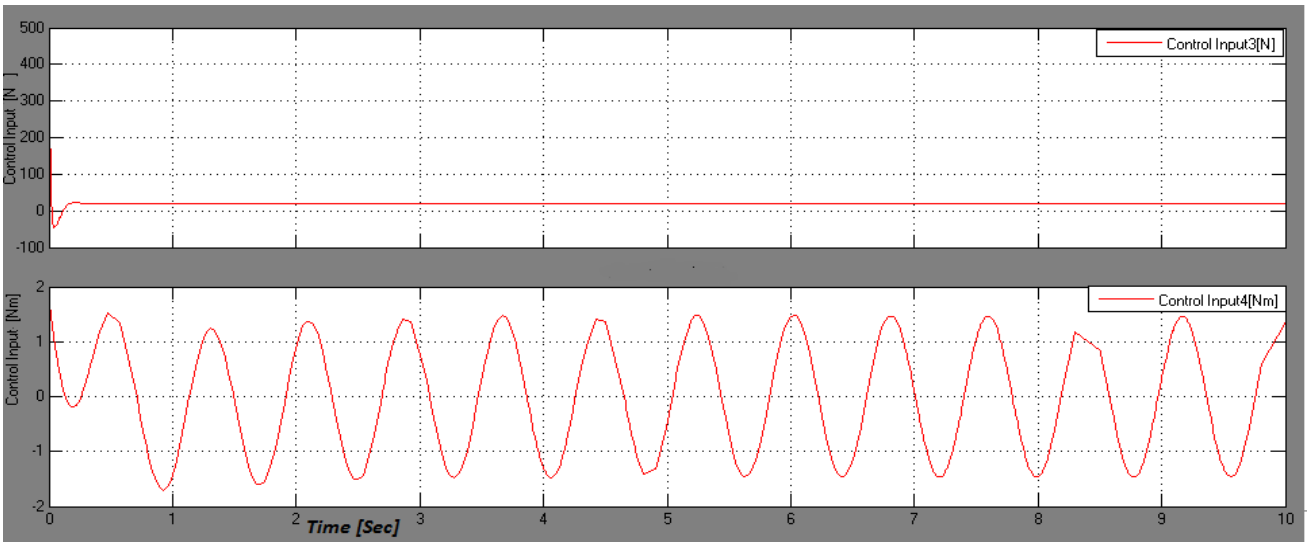

Figure 5.39: Input torque for joint 3 and joint 4 using FSMC

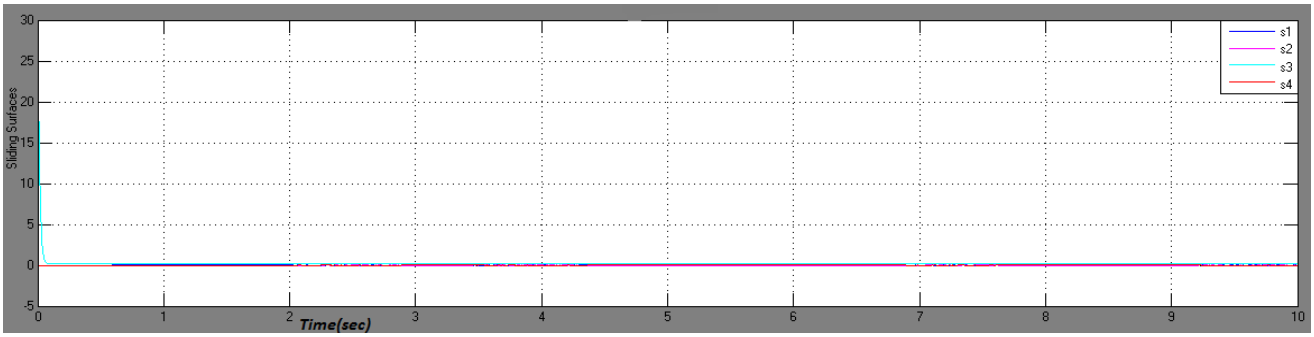

Figure 5.40: Sliding Surfaces using FSMC

Figure (5.41) shows the position tracking curve of the first joint by using FSMC. The figure shows that the first joint tracks the desired path very accurately, with very small tracking error as shown in figure (5.42) which is less than tracking error in SMC.

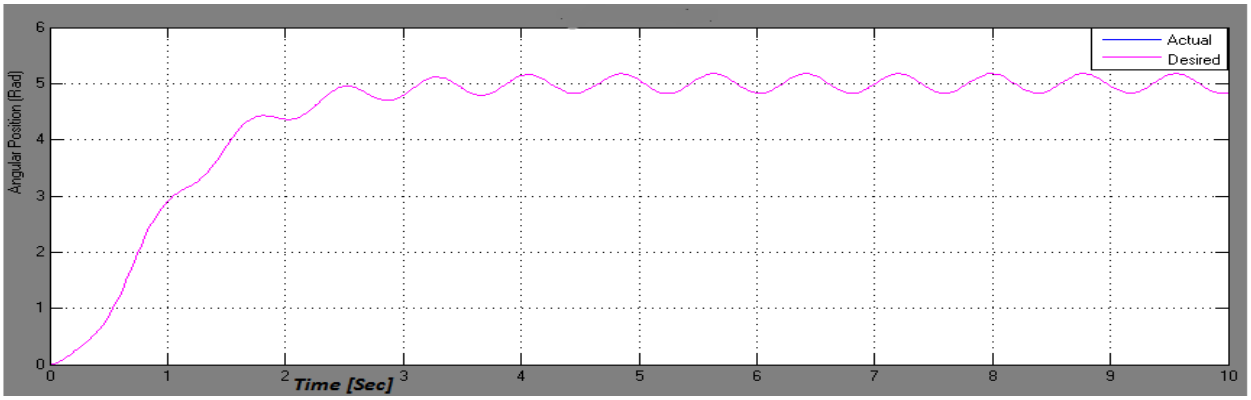

Figure 5.41: *Position Tracking Curve of the First Joint in rad*

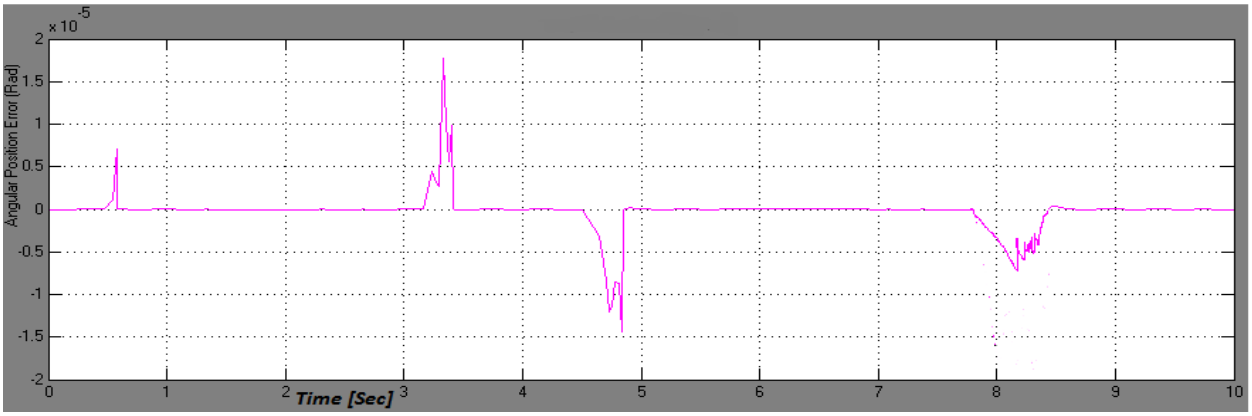

Figure 5.42: *Position Tracking Error Curve of the First Joint in rad*

Figure 5.43 shows the position tracking curve of the second joint, the same desired trajectory for the first joint chosen for second and Figure 5.44 represents its tracking error which is very low, reduced from order of 10^{-3} to of 10^{-5} by using FSMC.

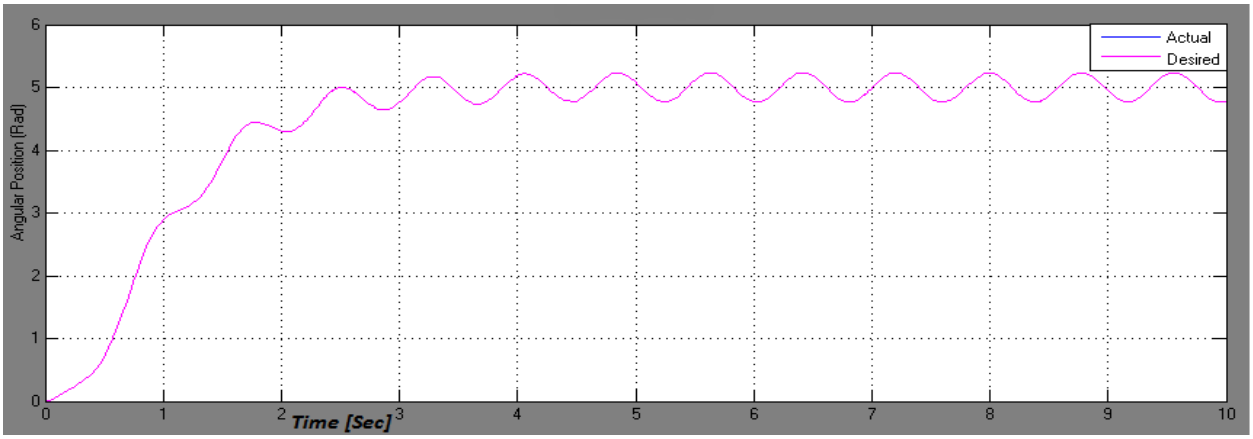

Figure 5.43: *Position Tracking Curve of the Second Joint in rad*

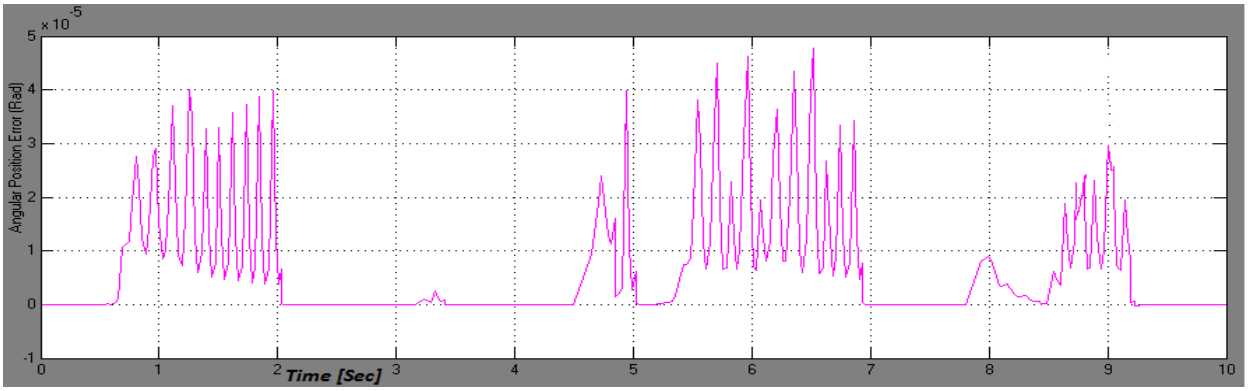

Figure 5.44: *Position Tracking Error Curve of the Second Joint in rad*

Figure (5.45) shows the position tracking curve of the third joint. As the figure show, the third joint tracks the desired path accurately, with very small tracking error that reduced from order of 10^{-3} to of 10^{-4} as shown in figure (5.46).

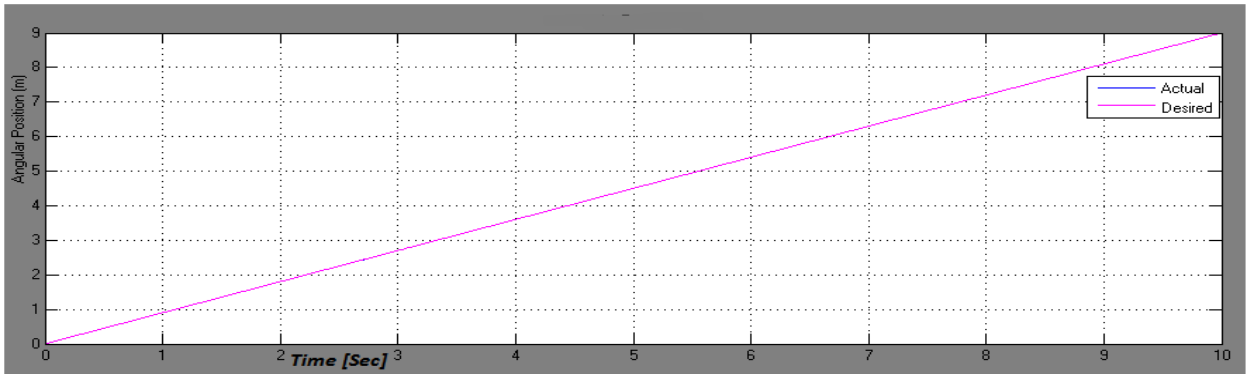

Figure 5.45: *Position Tracking Curve of the Third Joint in meter*

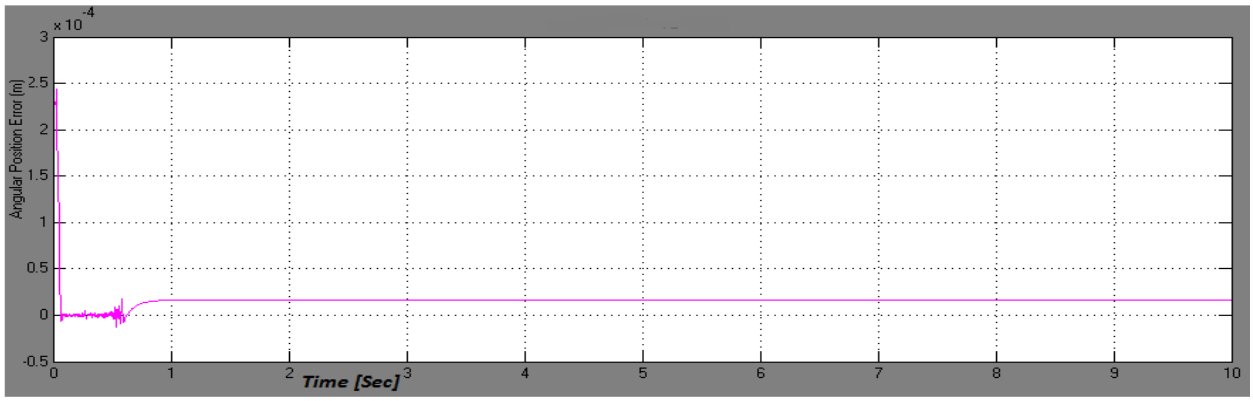

Figure 5.46: *Position Tracking Error Curve of the Third Joint in meter*

Figure 5.47 shows the position tracking response of the fourth joint when the designed controller allows the joint to follow the desired trajectory. Figure 5.48 represents its respective error plot. It can be seen that the tracking performance of FSMC is better than SMC that the tracking error reduced form of order of 10^{-3} to 10^{-5}.

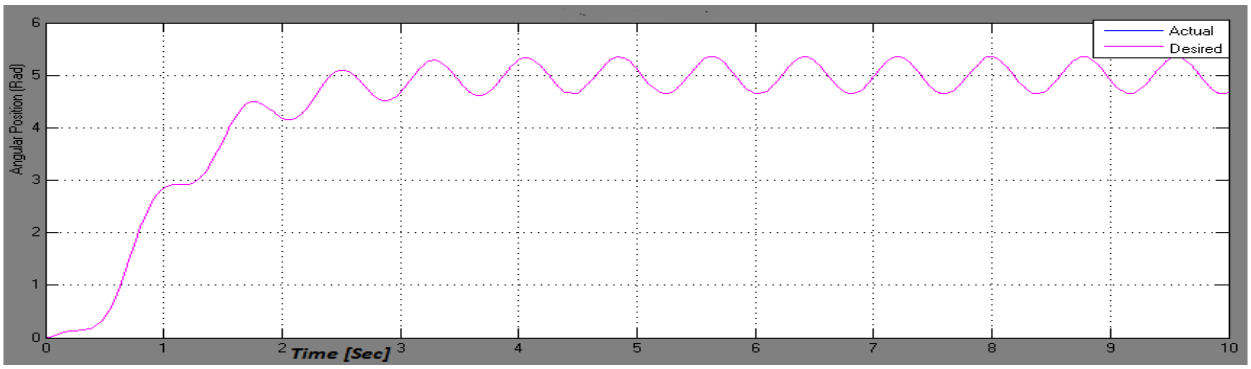

Figure 5.47: *Position Tracking Curve of the Fourth Joint in rad*

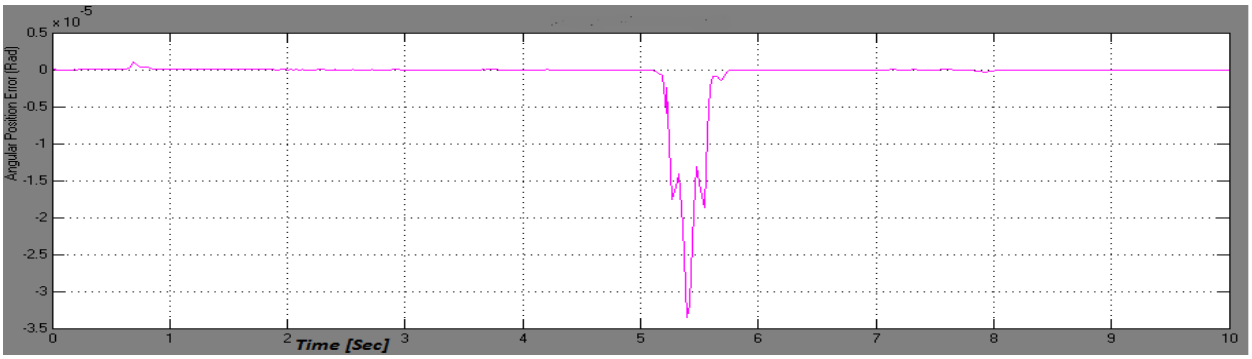

Figure 5.48: *Position Tracking Error Curve of the Fourth Joint in rad*

Figure from Fig. 5.49 to Fig. 5.56 shows angular velocities tacking and their respective velocity error tracking plot of FSMC. It is observed that with the proposed controller, the velocity tracking errors of first, second and fourth joint are within the range of [-0.04, 0.04] rad/s and of the third joint is [-0.015, 0.015] m/s.

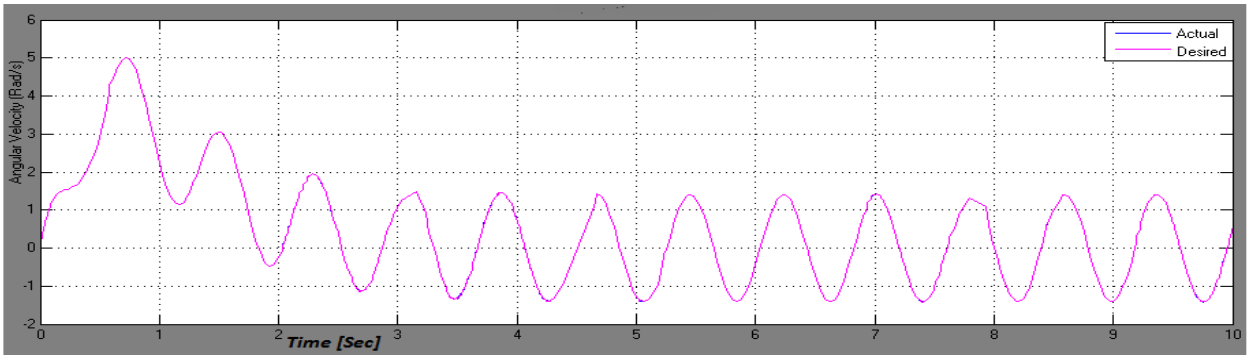

Figure 5.49: *Velocity Tracking Curve of the First Joint in rad/s*

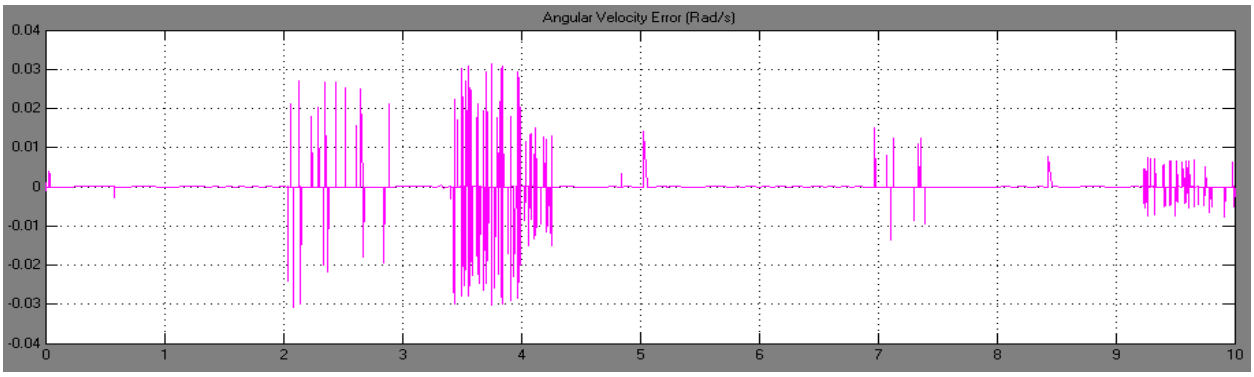

Figure 5.50: *Velocity Tracking Error Curve of the First Joint in rad/s*

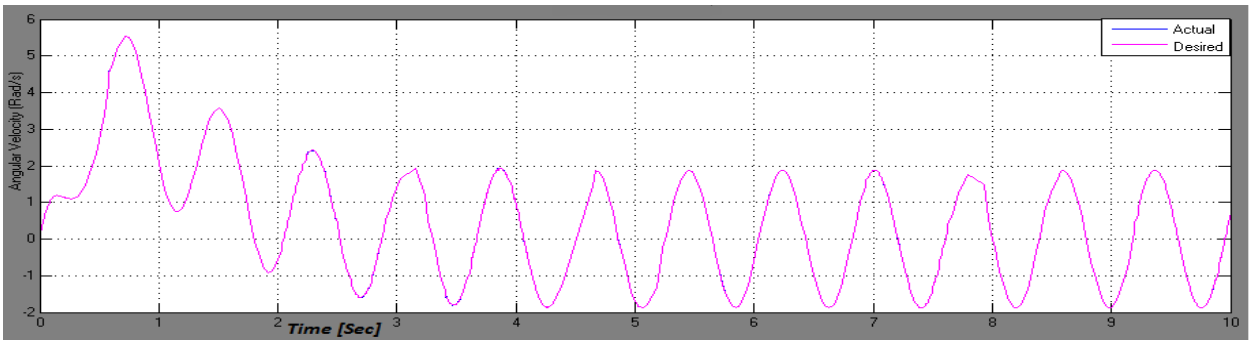

Figure5.51: *Velocity Tracking Curve of the Second Joint in rad/s*

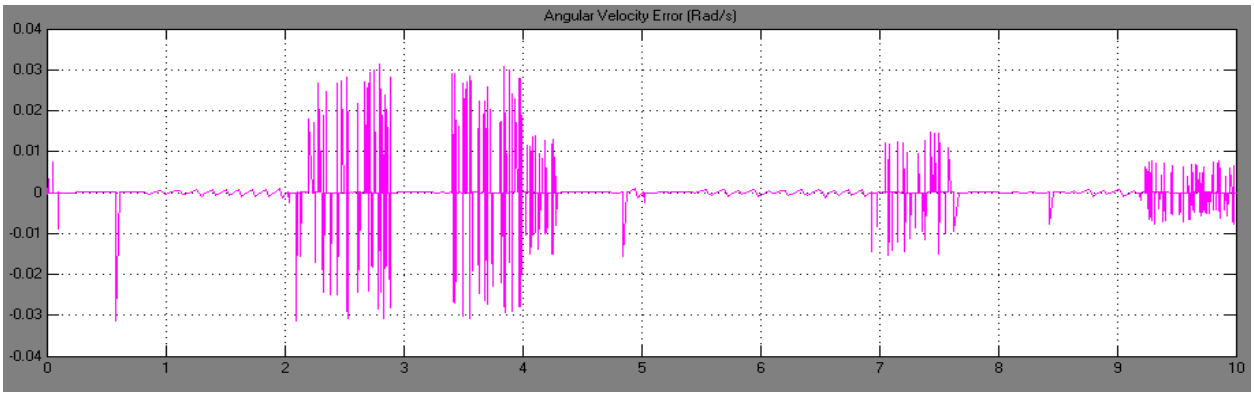

Figure 5.52: *Velocity Tracking Error Curve of the Second Joint in rad/s*

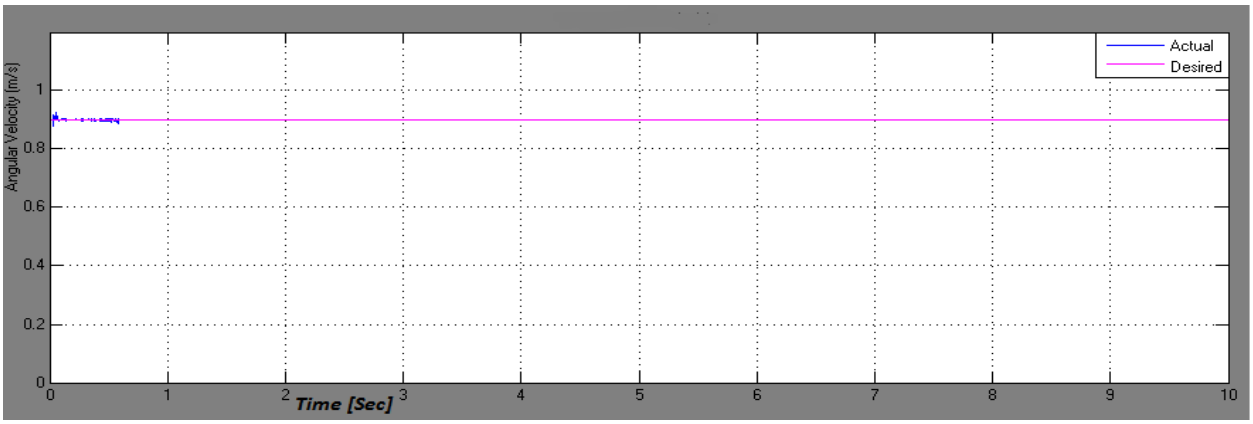

Figure5.53: *Velocity Tracking Curve of the Joint3 in m/s*

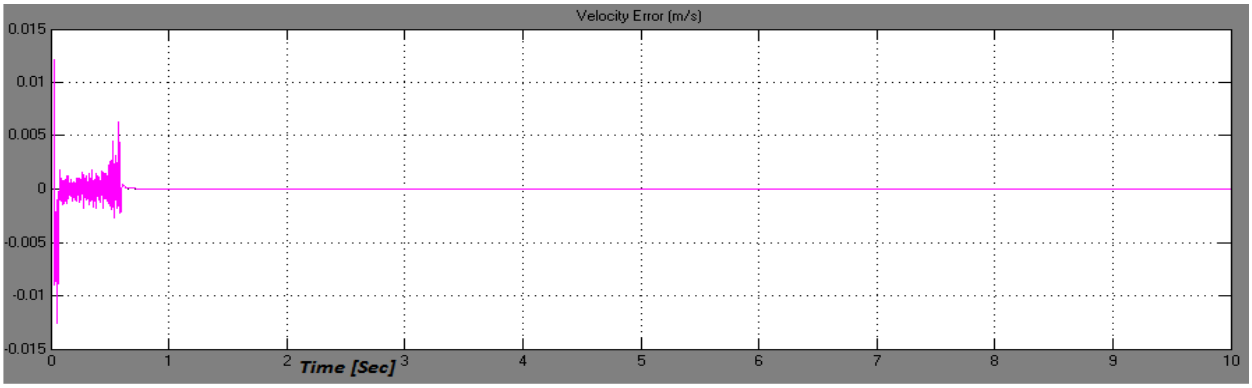

Figure 5.54: *Velocity Tracking Error Curve of Joint3 in m/s*

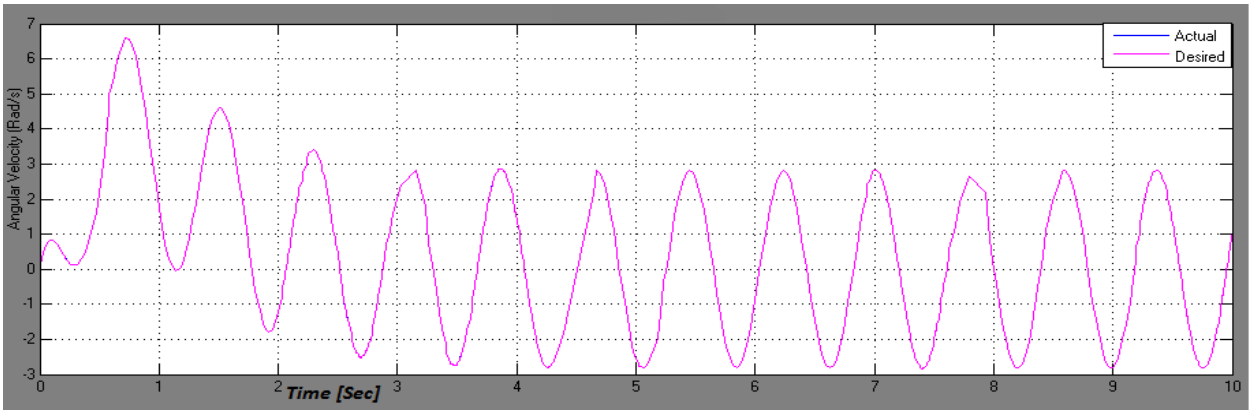

Figure 5.55: *Velocity Tracking Curve of the Fourth Joint in rad/s*

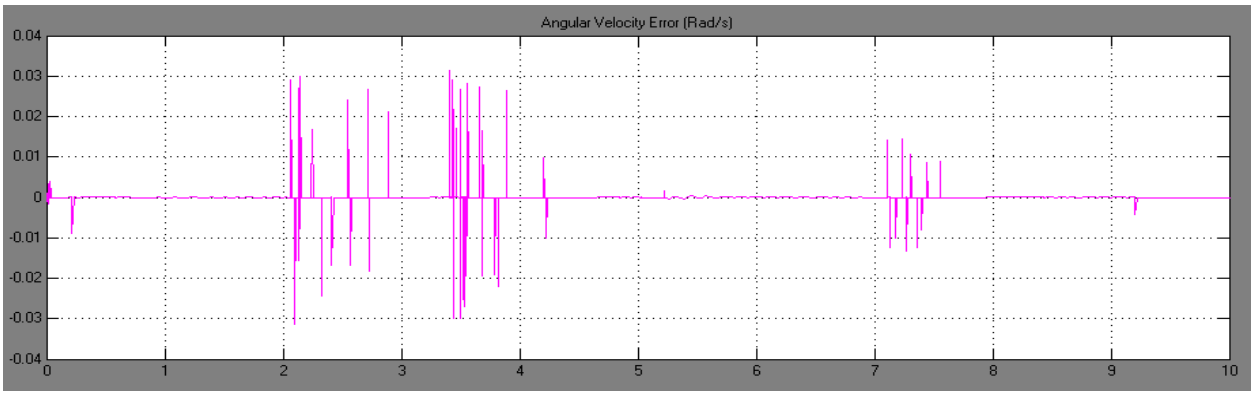

Figure 5.56: *Velocity Tracking Error Curve of the Fourth Joint in rad/s*

The random variations of position tracking error and velocity tracking error occurred for revolute joints (joint 1, 2, and 4) may be due to parameter (mass matrix and inertia tensor) variation and nonlinearities term (coriolis forces and centrifugal forces) but, it requires further research.

From the simulation results of angular velocity tracking curve we observed the overshot of angular velocity around 0.75 second for revolute joints. We know that the derivative of angular position gives angular velocity, the derivative of angular position around 0.75 second for joint 1, joint 2, and joint 4 results angular velocity to have overshot.

Chattering phenomena

As noticed in earlier section, chattering is one of the most important challenge in sliding mode controller which is one of the main objective in this thesis is to remove chattering in control output (torque). As shown in figures 5.19, 5.20, 5.37 and 5.38 the chattering effect is completely removed. The reaching law with exponential function $\exp(-\frac{\alpha}{|s|})$ in SMC and FSMC is very strong enough to remove chattering from control output.

Comparison of the Control Effort

The control input, forces the robot manipulator to track the desired trajectories. Referring to Fig. 5.1, 5.2, Fig. 5.20, 5.21 and Fig 5.38, 39 shows that the amount of control effort required in SMC with PID sliding surface is very much higher than the control effort required in SMC with PID sliding surface, exponential function in reaching law and FSMC with PID sliding surface and exponential function in reaching. The amount of control effort decides the size of actuator required for the system. In the control efforts, smaller control effort means less torque. Therefore, the proposed controller requires less torque than SMC.

So, the obtained simulations show major advantages of the FSMC that comprise $\exp\left(-\frac{\alpha}{|s|}\right)$ in the reaching law , which proves its superior ability to eliminate the chattering phenomenon and to clearly minimize input torques values.

Tracking Error Comparison

This part utilized to test the controller joint variable accuracy. From figures listed above the position tracking error curves and velocity tracking error curves it has been observed that FSMC with PID sliding surface and exponential function in reaching law errors when compared with SMC with PID sliding surface and SMC with PID sliding surface and exponential function in reaching law errors have been remarkably reduced.

CHAPTER SIX

6. Conclusion and Future Work

6.1. Conclusion

This paper, addressed the kinematic and dynamic model of a SCARA-type manipulator robot of four degrees of freedom. The dynamic model derived by using Euler-Lagrange formulation. Two controllers were developed: sliding mode controller with PID sliding surface and fuzzy sliding mode controller with PID sliding surface which contain constant and exponential rate reaching law. A simulation was done, using the MATLAB/Simulink *software*. Simulation tests on this model were presented, under each controller through a path tracking test. The results obtained through the simulation environment were displayed.

We noticed the increment of the torque value with the increment of the velocity, which can easily be predicted from the structure of the dynamics equations.

From the simulation result, the joint position tracking and joint velocity tracking responses can be controlled to follow the reference trajectories very accurately. That means, the manipulator model presented a trajectory tracking response whose errors were greater when using the sliding mode controller than the fuzzy sliding mode case. The control efforts were high when using the sliding mode controller than in SMC with exponential function and the proposed controller. Also, the proposed controller completely removes chattering, has good tracking error, and requires less torque which is superior to the one obtained with a SMC. And finally, this controller is simple and easy to implement.

6.2 Future Work

Although this thesis has successfully accomplished the mission which was targeted; not all things have been done. Further research can extend to include the study of influence of system perturbation and external disturbances.

References

[1]. Farzin Piltan.et al, "*Modeling and Control of Four Degrees of Freedom Surgical Robot Manipulator Using Matlab/Simulink"*, International Journal of Hybrid Information Technology, Vol. 8, No. 11, 2015

[2]. Luca Enrico Ferrari, "*Mat lab-based Control of a SCARA Robot"*, Master of Science in Electrical Engineering. University of Padua, 2014/15.

[3]. Westerlund, L., "*The Extended Arm of Man; a Book on History of the Industrial Robot"*, Stockholm, first edition 2000.

[4]. M. Y. Ibrahim, *Dynamic Performance Characteristics of Robot manipulator*, University of Wollogong, 1992.

[5]. Gregor B. "*Industrial Manufacturers SCARA Robot*". Internet: www.directindustry.com/74147.html, Jun 25, 2014 [Oct. 16, 2016]

[6]. Oussama Khatib, *Advanced Robotic Manipulation*, Stanford University, lecture note, 2005.

[7]. E. munoz, C. Gavinia, A. Vivas, *Terminal Sliding Control for a SCARA Robot*, International Conference on Control, Instrumentation and Mechatronics Engineering, Malaysia, May 28-29, 2007.

[8]. Raul Rascon, David Rosas and Luis Moreno, *Tracking and Force Control of a SCARA Robot Under a Constraint Using Sliding Mode Control* national conference on control and automation Ensenada, Baja California, Mexico, Oct 16-18, 2013

[9]. Farzin Piltan, et.al, *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

[10]. Ataei Mohammad, S. Ehsan Shafiei , *Sliding Mode PID-Controller Design for Robot Manipulators by Using Fuzzy Tuning Approach*, Proceedings of the 27th Chinese Control Conference , Kunming,Yunnan, China , July 16-18, 2008.

[11]. Florin Moldoveanu, *Sliding Mode Controller Design for Robot Manipulators*, Bulletin of the *Transilvania* University of Braşov, Vol. 7 (56) No. 2 - 2014

[12]. Mark W. Spong, Seth Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, First Edition, John Wiley & Sons, INC.

[13] Jian Fang and Wei Li, *Four degrees of freedom SCARA robot kinematics modeling and simulation analysis*, International Journal of Computer, Consumer and Control (IJ3C), Vol. 2, No.4 (2013)

[14]. John J. Craig, *Introduction to robotics mechanics and control*, second edition, 1955.

[15]. Mark W. Spong and M. Vidyasagar, *Robot Modelling and Control*, first edition, year unknown.

[16]. Mahmoud A. Abualsebah, *Trajectory Tracking Control of 3-DOF Robot Manipulator Using TSK Fuzzy Controller,* Master of Science in Electrical Engineering, Deanery of Graduate Studies, Electrical Engineering Department, University of Gaza.

[17]. V.I.Utkin, *Sliding Mode Control: Mathematical Tools, Design and Applications*, Lecture note, 2004.

[18]. Sanjay. Rao, *Some Issues in the Sliding Mode Control of Rigid Robotic Manipulators, Master of Science (Computer Science)*, School of Information Technology and Mathematics, Edith Cowan University,Perth, Western Australia ***October, 1995***.

[19]. Florin Moldoveanu, *Sliding Mode Controller Design for Robot Manipulators*, Bulletin of the *Transilvania* University of Braşov, Series I: Engineering Sciences Vol. 7 (56) No. 2, 2014

[20]. Andreja Rojko and Karel Jezernik, *Adaptive Fuzzy Sliding Mode Control of Robot Manipulator*, University of Maribor, Slovenia, Faculty of Electrical Engineering and Computer Science, 2000**.**

[21]. J. Wang, et al., *Indirect adaptive fuzzy sliding mode control: Part I: fuzzy switching, Fuzzy Sets and Systems*, Vol.122, Issue 1, 16 August 2001.

[22]. F. Ashrafzadeh, E.P. Nowicki, R. Boozarjomehry and J.C. Salmon, *Optimal Synthesis of Fuzzy Sliding Mode Controllers*, IEEE, 1996.

[23]. V.I.Utkin, *Sliding Mode Control, Mathematical Tools, Design and Applications,* Lecture note, 2004.

[24]. Chung-Chun Kung and Chia-Chang Liao, *Fuzzy-Sliding Mode Controller Design For Tracking Control Of Non-Linear System,* Department of Electrical Engineering, Tatung Inst. of Technology.

[25]. Gao, W., Hung, J.C, *Variable Structure Control of Nonlinear Systems: A New Approach,* IEEE Transaction on Industrial Electronics, 1993.

[26]. Jinming Zbang, et. al, *Analysis and Design of Fuzzy Controller Based on Fuzzy Reaching Law*, 14th Triennial World Congress Beijing, P.R. China, 1999.

[27]. Kevin M. Passino, *Fuzzy Control,* Department of Electrical Engineering the Ohio State University, first edition, 1998.

[28]. Xiaosong Lu, *An Investigation of Adaptive Fuzzy Sliding Mode Control for Robotic Manipulators*, Master of Applied Science in Electrical Ottawa-Carleton Institute of Electrical and Computer Engineering, 2007.

[29]. Kevin J. Walchko, *Development of A Generic Fuzzy Logic MIMO Controller For Satellite Attitude Control*, Master of Science in Mechanical Engineering University of Florida, 1999.

[30]. Philip Voglewede, Anton H. C. Smith, and Antonello Monti, *Dynamic Performance of a SCARA Robot Manipulator with Uncertainty Using Polynomial Chaos Theory*, IEEE Transactions on Robotics, 2009

[31]. Andinet Negash, *Design of Fuzzy Sliding Mode Controller for the Ball and Plate System,* Masters of science, Addis Ababa Institute of Technology, School of Graduate Studies, Addis Ababa University, 2011.

APPENDIX

APPENDIX A: MATLAB Code

A1 Embedded MATLAB function Code for Robot Dynamics

```
function [q1ddot,q2ddot,q3ddot,q4ddot] = fcn(tau1,tau2,tau3,tau4,q1,q2,q3,
q3,q1dot,q2dot,q3dot,q4dot)
%%%%Define the Parameters of Robot Manipulator
a1=0.5;a2=0.4;m1=15;m2=12;m3=3;m4=3;I1=0.02087*m1;I2=0.08*m2;I3=0.05;I4=0.02*
m4;
m11=0.25*m1*(a1)^2+m2*(a1)^2+m2*a1*a2*(cos(q1)*cos(q1+q2))+m2*(a1)^2+m2*a1*a2
*sin(q1)*sin(q1+q2)+0.25*m2*(a2)^2+I2+m3*(a1)^2+2*m3*a1*a2*(cos(q1)*cos(q1+q2
)+sin(q1)*sin(q1+q2))+I3+m4*((a1)^2+(a2)^2+2*a1*a2*(cos(q1)*cos(q1+q2)+
sin(q1)*sin(q1+q2)))+I4;
m12=0.25*m2*(a2)^2+0.5*m2*a1*a2*(cos(q1)*cos(q1+q2)+sin(q1)*sin(q1+q2))+I2+m3
*((a2)^2+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2))) +I3+ m4*((a2)^2
+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2)))+I4;
m13=0; m14=I4; m21=m12;
m22=0.25*m2*(a2)^2+I2+m3*(a2)^2+m4*(a2)^2+I4; m23=0; m24=I4; m31=0; m32=0;
m33=0.25*m3+m4; m34=0; m41=I4; m42=I4; m43=0; m44=I4;
m=[m11 m12 m13 m14;m21 m22 m23 m24;m31 m32 m33 m34;m41 m42 m43 m44];%Inertial
Mass Matrix
b112=m2*a1*a2*(sin(q1)*cos(q1+q2)-
cos(q1)*sin(q1+q2))+2*m3*a1*a2*(sin(q1)*cos(q1+q2)-
sin(q1+q2)*cos(q1))+2*m4*a1*a2*(cos(q1+q2)*sin(q1)-cos(q1)*sin(q1+q2));
b122=0.5*m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m3*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m4*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
b211=-2*m3*a1*a2*sin(q1)*cos(q1+q2)-2*m4*a1*a2*sin(q1)*cos(q1+q2)-
m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
N=[b112*q1dot*q2dot+b122*q2dot^2;b211*q1dot^2;5*(m3+2*m4);0];
qddot=inv(m)*([tau1;tau2;tau3;tau4]-N);
q1ddot=qddot(1,1);
q2ddot=qddot(2,1);
q3ddot=qddot(3,1);
q4ddot=qddot(4,1);
```

A2 Embedded MATLAB function Code for SMC

```
function [u1,u2,u3,u4,s1,s2,s3,s4,e1,e2,e3,e4] =
fcn(q1,q2,q3,q4,q1dot,q2dot,q3dot,q4dot,qd1,qd2,qd3,qd4,qd1dot,qd2dot,qd3dot,
qd4dot,qd1ddot,qd2ddot,qd3ddot,qd4ddot,ie1,ie2,ie3,ie4)
%% q-actual position; qd- desired position;
a1=0.5;a2=0.4;m1=15;m2=12;m3=3;m4=3;I1=0.02087*m1;I2=0.08*m2;I3=0.05;I4=0.02*
m4;
m11=0.25*m1*(a1)^2+m2*(a1)^2+m2*a1*a2*(cos(q1)*cos(q1+q2))+m2*(a1)^2+m2*a1*a2
*sin(q1)*sin(q1+q2)+0.25*m2*(a2)^2+I2+m3*(a1)^2+2*m3*a1*a2*(cos(q1)*cos(q1+q2
)+sin(q1)*sin(q1+q2))+I3+m4*((a1)^2+(a2)^2+2*a1*a2*(cos(q1)*cos(q1+q2)+
sin(q1)*sin(q1+q2)))+I4;
m12=0.25*m2*(a2)^2+0.5*m2*a1*a2*(cos(q1)*cos(q1+q2)+sin(q1)*sin(q1+q2))+I2+m3
*((a2)^2+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2))) +I3+ m4*((a2)^2
+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2)))+I4; m13=0; m14=I4; m21=m12;
m22=0.25*m2*(a2)^2+I2+m3*(a2)^2+m4*(a2)^2+I4; m23=0;
m24=I4; m31=0; m32=0; m33=0.25*m3+m4; m34=0; m41=I4; m42=I4; m43=0; m44=I4;
```

```
m=[m11 m12 m13 m14;m21 m22 m23 m24;m31 m32 m33 m34;m41 m42 m43 m44];
b112=m2*a1*a2*(sin(q1)*cos(q1+q2)-
cos(q1)*sin(q1+q2))+2*m3*a1*a2*(sin(q1)*cos(q1+q2)-
sin(q1+q2)*cos(q1))+2*m4*a1*a2*(cos(q1+q2)*sin(q1)-cos(q1)*sin(q1+q2));
b122=0.5*m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m3*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m4*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
b211=-2*m3*a1*a2*sin(q1)*cos(q1+q2)-2*m4*a1*a2*sin(q1)*cos(q1+q2)-
m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
%let N=B+C+N;where B is coriolis force; C is centerfugal force and G is
%gravitational force
N=[b112*q1dot*q2dot+b122*q2dot^2;b211*q1dot^2;5*(m3+2*m4)*0;0];
%define error and error dot
e1=qd1-q1;
e2=qd2-q2;
e3=qd3-q3;
e4=qd4-q4;
e1dot=qd1dot-q1dot;
e2dot=qd2dot-q2dot;
e3dot=qd3dot-q3dot;
e4dot=qd4dot-q4dot; % where e eror, edot is error dot
%define lambda and gain Q
lambda1=40;
lambda2=40;
lambda3=60;
lambda4=60;
s1=e1dot+lambda1*e1+ie1*((lambda1/2)^2);
s2=e2dot+lambda2*e2+ie2*((lambda2/2)^2);
s3=e3dot+lambda3*e3+ie3*((lambda3/2)^2);
s4=e4dot+lambda4*e4+ie4*((lambda4/2)^2);
Q1=20;
Q2=15;
Q3=28;
Q4=8;
%let v=m*(qdddot+lambda*edot+(lambda/2)^2*e where qdddot is qd double dot
V=m*([qd1ddot;qd2ddot;qd3ddot;qd4ddot]+[lambda1 0 0 0;0 lambda2 0 0;0 0
lambda3 0;0 0 0 lambda4]*[e1dot;e2dot;e3dot;e4dot]+[(lambda1/2)^2 0 0 0;0
(lambda2/2)^2 0 0; 0 0 (lambda3/2)^2 0;0 0 0 (lambda4/2)^2]*[e1;e2;e3;e4])+N;
ks=[60 0 0 0;0 20 0 0;0 0 13 0;0 0 0 11]*[s1;s2;s3;s4];
u=V+m*ks+m*[Q1 0 0 0;0 Q2 0 0;0 0 Q3 0;0 0 0
Q4]*[sign(s1);sign(s2);sign(s3);sign(s4)];
u1=u(1,1);
u2=u(2,1);
u3=u(3,1);
u4=u(4,1);
```

A4 Embedded MATLAB function Code for FSMC

```
function [u1,u2,u3,u4,s1,s2,s3,s4,m,e1,e2,e3,e4] =
fcn(q1,q2,q3,q4,q1dot,q2dot,q3dot,q4dot,qd1,qd2,qd3,qd4,qd1dot,qd2dot,qd3dot,
qd4dot,qd1ddot,qd2ddot,qd3ddot,qd4ddot,ie1,ie2,ie3,ie4)
%% q-actual position; qd- desired position;
a1=0.5;a2=0.4;m1=15;m2=12;m3=3;m4=3;I1=0.02087*m1;I2=0.08*m2;I3=0.05;I4=0.02*
m4;
m11=0.25*m1*(a1)^2+m2*(a1)^2+m2*a1*a2*(cos(q1)*cos(q1+q2))+m2*(a1)^2+m2*a1*a2
*sin(q1)*sin(q1+q2)+0.25*m2*(a2)^2+I2+m3*(a1)^2+2*m3*a1*a2*(cos(q1)*cos(q1+q2
```

```
)+sin(q1)*sin(q1+q2))+I3+m4*((a1)^2+(a2)^2+2*a1*a2*(cos(q1)*cos(q1+q2)+
sin(q1)*sin(q1+q2)))+I4;
m12=0.25*m2*(a2)^2+0.5*m2*a1*a2*(cos(q1)*cos(q1+q2)+sin(q1)*sin(q1+q2))+I2+m3
*((a2)^2+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2))) +I3+ m4*((a2)^2
+a1*a2*(cos(q1)*cos(q1+q2)-sin(q1)*sin(q1+q2)))+I4; m13=0; m14=I4;m21=m12;
m22=0.25*m2*(a2)^2+I2+m3*(a2)^2+m4*(a2)^2+I4; m23=0; m24=I4; m31=0; m32=0;
m33=0.25*m3+m4; m34=0; m41=I4; m42=I4; m43=0; m44=I4;
m=[m11 m12 m13 m14;m21 m22 m23 m24;m31 m32 m33 m34;m41 m42 m43 m44];
b112=m2*a1*a2*(sin(q1)*cos(q1+q2)-
cos(q1)*sin(q1+q2))+2*m3*a1*a2*(sin(q1)*cos(q1+q2)-
sin(q1+q2)*cos(q1))+2*m4*a1*a2*(cos(q1+q2)*sin(q1)-cos(q1)*sin(q1+q2));
b122=0.5*m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m3*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2))-
m4*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
b211=-2*m3*a1*a2*sin(q1)*cos(q1+q2)-2*m4*a1*a2*sin(q1)*cos(q1+q2)-
m2*a1*a2*(sin(q1)*cos(q1+q2)-cos(q1)*sin(q1+q2));
N=[b112*q1dot*q2dot+b122*q2dot^2;b211*q1dot^2;5*(m3+2*m4)*0;0];
%define error and error dot
e1=qd1-q1;
e2=qd2-q2;
e3=qd3-q3;
e4=qd4-q4;
e1dot=qd1dot-q1dot;
e2dot=qd2dot-q2dot;
e3dot=qd3dot-q3dot;
e4dot=qd4dot-q4dot;
%define lambda and gain Q
lambda1=30;
lambda2=30;
lambda3=41;
lambda4=35;
Q1=35;
Q2=15;
Q3=50;
Q4=10;
%%%%Define Sliding Surface
s1=e1dot+lambda1*e1+ie1*((lambda1/2)^2);
s2=e2dot+lambda2*e2+ie2*((lambda2/2)^2);
s3=e3dot+lambda3*e3+ie3*((lambda3/2)^2);
s4=e4dot+lambda4*e4+ie4*((lambda4/2)^2);
 %let v=m*(qdddot+lambda*edot+(lambda/2)^2*e where qdddot is qd double dot
v=m*([qd1ddot;qd2ddot;qd3ddot;qd4ddot]+[lambda1 0 0 0;0 lambda2 0 0;0 0
lambda3 0;0 0 0 lambda4]*[e1dot;e2dot;e3dot;e4dot]+[(lambda1/2)^2 0 0 0;0
(lambda2/2)^2 0 0;0 0 (lambda3/2)^2 0;0 0 0 (lambda4/2)^2]*[e1;e2;e3;e4])+N;
%% N=B+C+N;where B is coriolis force; C is centerfugal force and G is
gravitational force

u=v+m*[Q1*exp(-0.8/abs(s1)) 0 0 0;0 Q2*exp(-0.8/abs(s2)) 0 0;0 0 Q3*exp(-
0.8/abs(s3)) 0; 0 0 0 Q4*exp(-
0.8/abs(s4))]*[sign(s1);sign(s2);sign(s3);sign(s4)];
 %where 0.8 is the value of alpha
u1=u(1,1);
u2=u(2,1);
u3=u(3,1);
u4=u(4,1);
```

Appendix B

Transformation Matrix

$$ {}^{0}_{4}T = \begin{bmatrix} c_4(c_1c_2 - s_1s_2) - s_4(c_1s_2 + s_1c_2) & s_4(s_1s_2 - c_1c_2) - c_4(c_1s_2 + s_1c_2) & 0 & a_2(c_1c_2 - s_1s_2) + a_1c_1 \\ c_4(c_2s_1 + c_1s_2) - s_4(s_1s_2 - c_1c_2) & -s_4(c_2s_1 + c_1s_2) + c_4(-s_1s_2 + c_1c_2) & 0 & a_2(c_2s_1 + c_1s_2) + a_1s_1 \\ 0 & 0 & 1 & -d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ Ax = c_4(c_1c_2 - s_1s_2) - s_4(c_1s_2 + s_1c_2);\ Ay = c_4(c_2s_1 + c_1s_2) - s_4(s_1s_2 - c_1c_2);\ Az = 0 $$

$$ Bx = s_4(s_1s_2 - c_1c_2) - c_4(c_1s_2 + s_1c_2);\ By = -s_4(c_2s_1 + c_1s_2) + c_4(-s_1s_2 + c_1c_2);\ Bz = 0 $$

$$ Cx = 0; Cy = 0; Cz = 1 $$

$$ Px = a_2(c_1c_2 - s_1s_2) + a_1c_1\,; Py = a_2(c_2s_1 + c_1s_2) + a_1s_1; Pz = -d_3 $$

Inertia matrix

$$ M(q) = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} $$

Where

$$ m_{11} = 0.25m_1{a_1}^2 + m_2({a_1}^2 + a_1a_2(c_1c_{12} + s_1s_{12}) + 0.25{a_2}^2) + I_{zz2} + m_3\left({a_1}^2 + {a_2}^2 + 2a_1a_2(c_1c_{12} + s_1s_{12})\right) + I_{zz3} + m_4\left({a_1}^2 + {a_2}^2 + 2a_1a_2(c_1c_{12} + s_1s_{12})\right) + I_{zz4} $$

$$ m_{12} = m_2\left(0.25{a_2}^2 + 0.5a_1a_2(c_1c_{12} + s_1s_{12})\right) + I_{zz2} + m_3\left({a_2}^2 + a_1a_2(c_1c_{12} - s_1s_{12})\right) + I_{zz3} + m_4\left({a_2}^2 + a_1a_2(c_1c_{12} - s_1s_{12})\right) + I_{zz4} $$

$$ m_{13} = 0, \qquad m_{14} = I_{zz4}, m_{21} = m_{12} $$

$$ m_{22} = 0.25m_2{a_2}^2 + I_{zz2} + {a_2}^2m_3 + {a_2}^2m_4 + I_{zz4} $$

$$ m_{23} = 0 $$

$$ m_{24} = I_{zz4} $$

$$ m_{31} = m_{13} = 0 $$

$$m_{32} = m_{23} = 0$$

$$m_{33} = \frac{1}{2}m_3 + m_4$$

$$m_{34} = 0$$

$$m_{41} = m_{14}$$

$$m_{42} = m_{24}$$

$$m_{43} = m_{34}$$

$$m_{44} = I_{zz4}$$