

ADDIS ABABA UNIVERSITY
ADDIS ABABA INSTITUTE OF TECHNOLOGY (AAIT)
SCHOOL OF ELECTRICAL & COMPUTER ENGINNERING
DEPARTMENT OF ELCTRICAL ENGINEERING

"TRAJECTORY TRACKING CONTROL OF DELTA-ROBOT USING 3rd ORDER SLIDING MODE CONTROL"

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 Control Engineering.

By

AMDAIL SHEFAW

Advisor: Dr. Mengesha Mamo

Co-Adviser: Mr.Andnent Negash

April, 2016



ADDIS ABABA UNIVERSITY
ADDIS ABABA INSTITUTE OF TECHNOLOGY (AAIT)
SCHOOL OF ELECTRICAL & COMPUTER ENGINNERING
DEPARTMENT OF ELCTRICAL ENGINEERING

"TRAJECTORY TRACKING CONTROL OF DELTA-ROBOT USING 3rd ORDER SLIDING MODE CONTROL"

By :-AMDAIL SHEFAW

APPROVED BY BOARD OF EXAMINERS

Chairman, Department of Graduate Committee	**Signature**
Dr. Mengesha MAMO	
	Signature
Mr.Andinet Negash	
Co-advisor	**Signature**
Internal Examiner	**Signature**
External Examiner	**Signature**

DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been fully acknowledged.

Amdail Shefaw Muzyen _____
Name Signature

Place: Addis Ababa Institute of Technology, Addis Ababa University, Addis Ababa

Date of Submission: April,25, 2016

This thesis has been submitted for examination with our approval as a university advisor.

Dr. Mengesha Mamo _____
Advisor's Name Signature

Mr.Andnent Negash _____
Co-Adviser: Signature

Acknowledgment

First and forever, all praise and thanks for Allah, who gave me the strength, and patience to carry out this work in this good manner. I would like to deeply thank my advisor and co-advisor Dr. Mengesha Mamo and Mr.Andinet Negash for their assistance, guidance, support, patience, and encouragement. I would like to deeply thank the staff and my classmate for their assistance, help and encouragement. Great thanks also goes to my beloved family for their endless praying and continuous support. I feel it would be incomplete without regarding the heartfelt gratitude to my beloved wife for her love and endless support.

Contents

Acknowledgment i
List of Figures iv
List of Tables vi
List of Acronyms vii
Abstract viii
1. Chapter One: Introduction 1
1.1. Background 1
1.2. Problem Statement 5
1.3. Objective 6
1.3.1. General Objective 6
1.3.2. Specific Objectives 6
1.4. Thesis Contribution 6
1.5. Methodology 7
1.6. Literature Review 8
1.7. Thesis Overview 10
2. Chapter Two: Delta Robot Kinematics and Dynamics 11
2.1. Schematics of Delta Robot 11
2.2. Kinematics 12
2.2.1. Inverse (Indirect) Kinematics 13
2.2.2. Forward (Direct) Kinematics 18
2.2.3. Velocity Kinematics 21
2.2.4. Forward and Inverse Singularity Analysis 25
2.3. Dynamics 25
2.3.1. Virtual Work Dynamics 27
2.3.2. Non-Rigid Body Effects 30
2.3.3. Actuator Dynamics 31
3. Chapter Three: Sliding Mode Control 33
3.1. Sliding Order 34
3.2. Second Order Sliding Modes 35
3.2.1. Twisting Algorithm 36

3.2.2. Super-Twisting Algorithm 37
3.2.3. Sub-Optimal Algorithm 39
3.3. High Order Sliding Mode Controllers 41
4. Chapter Four :Simulations and Results 44
4.1. Multi-Body Modeling of 3-DOF Delta Robot 44
4.1.1. 3D-CAD Model for 3-DOF Delta Robot 45
4.1.2. Delta Robot SimMechanics Model 46
4.2. Trajectory Generation 49
4.3. Controller Design 50
4.3.1. Dynamic Model of Delta Robot 51
4.3.2. Sliding Mode Control 51
4.4. Simulation 54
4.4.1. Results 57
4.4.2. Discussions 72
4.4.3. Conclusion 73
4.4.4. Recommendations 74
References 75
Appendix 77

List of Figures

1. Chapter One
1.1. (a) Stewart platform /Parallel Robot, (b) FlexPicker Robotics Parallel Delta Robot
1.2. A SCARA-type serial-architecture Robot
1.3. Parallel Delta Robot Components
2. Chapter Two
2.1. Delta Robot's joint angle and end-effector orientation
2.2. Delta Robot's coordinate system with dimensions
2.3. Intersection of sphere and circle from the projected lower leg and upper leg
2.4. YZ-plane & base dimensions
2.5. Delta Robot symmetry and coordinate rotation
2.6. Delta Robot with coordinate point projected to form spheres
2.7. Coordinate point projection to the center of end-effector
2.8. Projection of link i on xizi plane, (b) end on view
2.9. DC motor model
3. Chapter Three
3.1. Twisting controller phase portrait
3.2. Super-twisting controller phase portrait
3.3. Sub-optimal controller phase trajectories
4. Chapter Four
4.1. A 3-D, 3-DOF Delta Robot assembly in SolidWorks
4.2. A 3D-CAD exported SimMechanic Second-generation model of 3-DOF Delta Robot
4.3. (a)desired trajectory generator (b) generated trajectory
4.4. Second order robust exact differentiator Simulink model
4.5. Circular Path Trajectory tracking for Delta Robot.
4.6. End effector trajectory tracking
4.7. (a)-(c) joint angles tracking (d) joint angle errors
4.8. The sliding variables $s, \dot{s}$, and $\ddot{s}$ joint each angles: (a) joint angle 1, (b) joint angle 2, (c) joint angle 3.
4.9. End effector X & Z axis Trajectory tracking

4.10. The 3-SMC control torques for each joint legs: (a) joint 1, (b) joint 2, (c) joint 3
4.11. End effector Trajectory tracking
4.12. Joint angle1 & angle2 tracking
4.13. Joint angle3 tracking & joint angle errors
4.14. The sliding variables $s, \dot{s}, and\ \ddot{s}$ for each joint angles:
4.15. End effector X & Z axis Trajectory tracking
4.16. The 3-SMC control torques for each joint legs: (a) joint 1, (b) joint 2, (c) joint 3.

List of Tables

4.1.Delta Robot design parameters

4.2.SimMechanics basic blocks

List of Acronyms

TCP	Tool Center Point
DOF	Degree of Freedom
{R}	Reference frame at the base plate.
Jx	Jacobian matrix in Cartesian space
Jq	Jacobian matrix in joint space
3-DOF	Three Degree of Freedom
3D	Three Dimensional
3-SMC	Third Order Sliding Mode Controller
H∞	H Infinity
HOSM	Higher Order Sliding Mode
IK	Inverse Kinematics
MIMO	Multi-Input Multi-Output

Abstract

Robots are multi-input multi-output (MIMO) nonlinear systems with uncertainties and high coupling effects ; this makes the control of Robot more difficult and needs a robust control system. This thesis describes, the concept of higher order sliding mode control as applied to trajectory tracking of a 3-DOF Delta Robot using tracking error as the sliding surface. Third order sliding mode control technique is specifically applied for this purpose.

With the Robot's complicated electromechanical parts, mutual interactions of Robot mechanics and drives, building the nonlinear dynamic model of the Robot using 3D-CAD (like SolidWorks) program was the best solution to model the Robot. Real parameters were introduced via 3D-CAD Modeling. The SimMechanics link utility bridges the gap between geometric modeling and block diagram modeling and simulation. Dynamical model with all the kinematic constraints for a Robot will simply be found by exporting a 3D-CAD model of the Robot to SimMechanics.

In this thesis a 3-DOF Delta Robot is designed in SolidWorks, the kinematic and dynamic models of the Robot have been developed, a 3-SMC sliding mode controller is designed and a circular trajectory tracking is achieved, with X & Z axis rms error of 3mm. The simulation result shows that the steady state tracking is reached before 3 seconds. Finite time convergence of sliding function and its first and second derivatives is also achieved. the plant has been subjected to the effects of joint internal mechanics, mass uncertainty and external disturbances of torques. The designed third order sliding mode control have fond to be smooth, model independent, robust and is insensitive to applied model parameter variations, external & internal effects beside eliminating the chattering effect of the standard SMC..

1. Chapter One: Introduction

1.1. Background

Robotic manipulators [1], [3] are a serial chain of links and each link is connected to next link with a joint . There are essentially two types of Robot manipulators, serial and parallel. Serial manipulators consist of a number of links connected in series to one another to form a kinematic chain. Each joint of the kinematic chain is usually actuated. This type of structure is known as an open chained mechanism. Parallel manipulators, on the other hand, consist of a number of kinematic chains connected in parallel to one another. The kinematic chains work in unison to move a common point. This common point usually consists of a manipulator that performs a certain task. For the purpose of the three degrees of freedom (3 DOF) Parallel Delta Robot system described in this thesis, the common point will also be referred to as the end-effector. Since the kinematic chains are eventually connected to a common point, a parallel manipulator is considered as a closed chained mechanism. The actuators in parallel manipulators are usually located at the base or close to the base of the system, which is in stark contrast to serial manipulators which have actuators at every joint. The advantages of this type of configuration include the fact that it could achieve a higher load capacity due to the decrease in the mass of the overall system, it can produce high accelerations at the End effector and it has a high mechanical stiffness to weight ratio [1].

The disadvantages of this type of configuration include the fact that the dynamic model is quite complex in nature and there are many instances of singularities that must be mapped out and avoided in order to maintain control of the system. Parallel Robots come in a wide variety of designs and applications ranging from the Stewart platform or Hexapod Parallel Robot shown in Fig. 1.1.a, which is used in aircraft motion simulators to the Delta Robot, which is used in packaging plants. This endows the fact that there cannot be a conclusive result as to which controller best suits the functionality of all parallel Robot. Therefore, it is logical to experiment with various control techniques to observe which controller would provide the most satisfactory results based on a specific mechanical system. It is impossible to adequately design any

controllers for the parallel Robot without a clear understanding of the dynamic model and the inverse kinematics of the mechanical model.

This thesis presents the reader with the simulation results obtained from the implementation of third order sliding mode control on 3 DOF Parallel Delta Robot. The derivation of the kinematics of the mechanical model is derived in detail followed by the dynamic model of the system. The non-singular region will be defined based on the results obtained in the inverse kinematics. It is important to map out the non-singular region since it is the only location in which the Parallel Robot is able to operate under stable conditions. If the parallel Robot were to enter a singular region, it would render the controller ineffective and cause the entire system to become unstable.

In recent years the number of studies and applications of Parallel Robot have increased. One of the most popular applications is in industry packaging. This is due to their ease of construction, the lightness of their structure and the high accelerations obtained by these devices.



(a) (b)

Fig. 1.1.(a) Stewart platform /Parallel Robot, (b) FlexPicker Robotics Parallel Delta Robot (www.abb.com)

Unlike the serial-type Robot manipulators, which only have an open-loop kinematic chain, parallel configuration allows for a distribution of payload among their two, or more closed-loop, kinematic chains. To illustrate this point consider Fig.1.1.a which shows a parallel-architecture Robot, used for object loading and unloading. Fig.1.2 shows a SCARA-type serial-architecture Robot. By comparing the two images it is easy to appreciate the difference between the two types of architecture. In the case of the serial manipulator greater robustness is required, as each link carries not only the weight of the successive links but also the motors and payload. This creates a cantilever effect in each link and, as a result, a greater overall deformation .In contrast, in the parallel architecture the actuators are fixed to the base of the manipulator so that the weight of the motors is not supported by the kinematic chains. In addition, the payload is distributed among the kinematic chains that conform the manipulator. This results in thinner and lighter kinematic chains, which in turn results in an increased payload capacity of the manipulator, relative to its total mass.



Fig.1.2 shows a SCARA-type serial-architecture Robot (www.mecademic.com)

A disadvantage of parallel Robot is their typically low cost effectiveness, complex kinematics and rather expensive control units, as well as the poor workspace to Robot-dimension ratio [1].

On the other hand, the advantages of parallel Robot stated before indicate that their capabilities can be optimally oriented if their specifications are task-adapted to the desired application. To facilitate flexibility and to enlarge the field of application, it is reasonable to use a reconfigurable Robot design. This will also help to overcome the typical challenges of parallel Robot, such as high costs and undersized workspaces.

Sliding mode control [21] is a robust and nonlinear control technique. In sliding mode control, motion of the system trajectory remains along a chosen surface of the state space. There are two phases of sliding mode control: reaching phase and sliding phase. Prior to reaching sliding surface, the system gets affected by disturbances, friction and uncertainties in the parameters. In sliding phase the system dynamics is governed by the sliding surface parameters and not the original system parameters and the system behavior becomes invariant to any disturbance or change in parameters. In addition, SMC is designed to achieve sliding mode in finite time. The main purpose of sliding mode control is to

1. Design a control law to effectively account for
   - ✓ Parameter uncertainty, like imprecision on the mass properties or loads, inaccuracies on the torque constants of the actuators, length of the links, friction, and so on.
   - ✓ Un-modeled dynamics, such as structural resonant modes, neglected time-delays (in the actuators, for instance), or finite sampling rate.
2. Quantify the resulting modeling Performance trade-offs, and in particular, the effect on tracking performance of discarding any particular term in the dynamic model.

In first order or conventional sliding mode control, the control signal is made to switch between two chosen structures about the sliding surface (or plane). This initiates high frequency oscillations in system states as the actuators cannot switch at infinite frequency. This is known as chattering. The problem of chattering can be overcome by using Higher Order Sliding Mode (HOSM) .In HOSM, the control is designed in a way to make higher order derivatives of the sliding function to reach zero in finite time.

1.2. Problem Statement

In this thesis a three degree of freedom Parallel Delta Robot is considered. The 3-DOF Delta Robot is a nonlinear, multivariable and coupled system. Parameters of the system such as gravitational load vary from task to task, and, may not be precisely known in advance. The system may also be subjected to uncertain nonlinearities such as external disturbances, link flexibility and joint friction. Controlling of Delta Parallel Robot requires true modeling for its dynamics [3], so, by using SolidWorks 3D-CAD program, it is possible to model the Robot and test its motion in Simulink Matlab tool, SimMechanics.

Good control strategy should take into account both parametric uncertainties and uncertain nonlinearities of the system itself. The optimal control problem can be stated as: find a closed loop optimal controller that minimizes the error between the measured phase and actual phase so as to track specified path.

Sliding mode control is nowadays a popular and well understood methodology to robustly control nonlinear dynamic systems under heavy uncertain conditions. However, sliding mode control has its own drawback, it can only be applied to systems having relative degree equal to one and is affected by chattering effect[21]. Higher order sliding mode control can be used to avoid the chattering phenomenon. It also has no limitation on the relative degree of the system i.e. with an increase in the relative degree of the system. A third order sliding mode controller can be used but another problem comes up, the knowledge of the higher derivative of a parameter called the sliding surface is needed. A robust exact differentiator is used to overcome this problem.

1.3. Objective

1.3.1. General Objective

The main objective of this thesis is to Design a control system for trajectory tracking of Delta Robot using third order sliding mode control (SMC). In order to analyze the trajectory of Delta Robot both the modeling and controlling of the Robot are important issues. Design of 3-DOF Delta Robot is made in SolidWorks 3D-CAD program. The controller has been designed in Simulink and the CAD model has been made available through SimMechanics Simulink link.

1.3.2. Specific Objectives

- ✓ To design a 3-DOF Delta Robot using computer aided design (CAD).
- ✓ To study the forward and inverse kinematic analysis: Both forward and inverse kinematic algorithms will be studied, which are essential for the motion planning and control of a Parallel Delta Robot.
- ✓ Singularity & Workspace analysis: It is necessary to ensure that Delta parallel Robot has a reasonable workspace volume.
- ✓ To design a third order sliding mode controller to track the trajectory of a 3-DOF Delta Robot's end-effector.

1.4. Thesis Contribution

The contribution of this thesis is the application of third order sliding mode control to a three degree of freedom Delta Robot. As far as we know, third order sliding mode has never been applied to 3-DOF Delta Robot before. Another contribution is the development of a dynamical model of the Delta Robot in SolidWorks 3D-CAD which satisfies the kinematic constraints of the Robot. The exported SimMechanics second-generation of the 3D model into a Simulink simulation environment represents the 3-DOF Delta Robot kinematic and dynamic model satisfactorily, which enable us to include the control law. The developed 3D-CAD model and SimMechanics Simulink model of the Delta Robot can also be used for further researches.

1.5. Methodology

The following methodologies have been used for the accomplishment of this thesis:

- ✓ Literature Survey: Different literatures which are related to this thesis work are studied and different concepts are adopted.

- ✓ CAD designing of 3-DOF Delta Robot: SolidWorks 2015 has been chosen as 3D-CAD modeling program to effectively meet the requirement of a 3-DOF Delta Robot system kinematics and dynamics and its integration with Simulink Matlab tool, SimMechanics.

- ✓ Controller design: non-linear controller of a third order SMC has been designed which is robust and insensitive to external disturbances and parameter uncertainties.

- ✓ Simulation: MATLAB R2015a / Simulink 8.5 version simulation tools has been utilized for proper CAD transformation of rigid bodies with their constraints to be maintained in MATLAB Simulink's simulation environment.

- ✓ Controller testing; the third order SMC has been tested subjected to different effects to ensure the robustness, parameter uncertainty and insensitivity to external disturbances while tracking the desired trajectory to follow.

1.6. Literature Review

Delta Robot is a three or four DOF parallel manipulator which was originally invented by Clavel [2] in 1988. It was designed to serve in the electronic, food and pharmaceutical industries which required a high level of hygiene and reliable standards for of the products.

The Delta Robot constitutes technological innovation in the Robot industry and has attracted much research interest since it was introduced to the industry for pick and place operations. The simple reason for this is that the Robot offers a very high accelerations due to the light weight of the moving parts, when compared with standard serial Robots. Furthermore, the Robot's closed link structure allows the actuators to be fixed on the base unlike the traditional serial Robots, which is usually actuated at each joint along the serial linkage. For a serial Robot, each axis has to drive not only the weight of the driven link but also the load of the servomotors for the following links. This results in moving heavy masses and a low dynamic motion response especially on a big machine. The links of the parallel Robot do not need to carry the load of actuators so the parallel links can be built as lightweight structures. Therefore, the delta Robot provides multiple advantages when compared with serial Robots; easy construction, higher stiffness of the structure, higher speed operation and better load to weight ratios.

On the other hand, Clavel pointed out that the delta Robot suffers the following disadvantage.

1. The working volume of the Robot is constrained by its mechanical construction.
2. Singularity configurations define the workspace

In recent years, Parallel manipulators have been studied extensively and several Parallel mechanisms have been developed by numerous researchers [2] in response to different industrial applications. These Robotic mechanisms are built with a combination of spherical joints and revolution joints on linkages between the base platform and moving platform thus forming closed chain loops to the manipulator.

The Parallel Delta Robot has gained its reputation in the industrial application due to its many advantages. As a result, the parallel Robot's application field has been extended beyond the production line and it is accepted widely in the agricultural and medical fields [2].

Modeling and control of a Closed Chain Parallel Delta Robot is very difficult especially, when using traditional methods in modeling. There are different papers that addressed the control of Delta Robot. Most of the papers use techniques that were used to control serial Robot.

The work presented in [3] dealt with a comparison of a model based predictive control MPC and PID controller. PID and MPC controllers were added at the output, the output torque command was limited, because that Servo DC motors attached at each actuated joint generate a limited torque. PID parameters were optimally tuned by desired response block in Matlab Simulink. Simulation results has been shown with PID Controller and MPC controller independently for different scenarios and their performances has been compared and contrasted. The two controllers attained accurate end-effector tracking results without compromising the amount of computation time and control effort usually found in more complex control techniques. In [3] MPC was the most suitable control technique to employ on the 3DOF Parallel Delta structure

While another paper [4] deals with the application of the H_∞ robust control law to the three degrees of freedom direct drive Delta parallel Robot. The authors developed the state space dynamic model and then carried out its linearization around an operating point. H_∞ controller is synthesized based on the linear model. A SimMechanics multi-body mechanical model was also developed to simulate the kinematics and the dynamics of the Robot and to validate the designed controller. To improve the tracking performances and increase the movement dynamics, the feed-forward pre-computed torque was considered. Two high speeds pick and place trajectories were been tested, robustness analyzed and simulation results were presented. In [4] the Delta Robot was first linearized about at an operating point while the system clearly has many operating points. Thus, the designed controller may perform well around the selected operating point but may not do the same away or at the other operating points.

[5] Claims to have achieved precise motion control using Parallel Robot with manufacturing tolerances and inaccuracies by migrating the measurements from their joint space to workspace in order to decrease the control system's sensitivity to any kinematical uncertainty rather than calibrating the parallel plant. The problem of dynamical model uncertainties and its effect on the derivation of the control law was also addressed in [5] through disturbance estimation and compensation. Eventually, both workspace measurement and disturbance estimations were

combined to formulate a control framework that is insensitive to both kinematical and dynamical system uncertainties. The developed control law was implemented on a 2-DOF parallel Robot. We noted that taking measurement in the workspace is more difficult and expensive than measurement in the joint space.

Two degree of freedom Parallel Delta Robot was considered in [21] and the kinematic and dynamic models of the robot have been developed. ADAMS sofware was used to model the dynamics of a 2DOF robot. Then a high order sliding mode controller is designed for the modeled in MATLAB/Simulink the model of the robot which effectively control the trajectory tracking of a 2-DOF delta robot simulated simultaneously in MATLAB/Simulink and ADAMS .

1.7. Thesis Overview

This thesis is organized in four chapters. This chapter introduces the background materials about a 3-DOF Delta Robot and sliding mode control in general. Chapter two covers the modeling of a 3-DOF Delta Robot; the modeling includes the kinematic and dynamic aspects. The kinematic modeling addresses the forward kinematic and inverse kinematic of the Robot. The dynamic modeling of the a 3-DOF Delta Robot is also presented using the virtual work principle. Chapter three addresses the design of HOSM controller for the a 3-DOF Delta Robot. In this section the theoretical background on standard sliding mode and high order sliding control is presented on its original formulation. Chapter four presents the 3D-CAD modeling, 3-SMC control simulation setup, simulation result, conclusion and discussion. In a sub-section of this chapter conclusion of this work is given and some recommendations are made.

2. Chapter Two: Delta Robot Kinematics and Dynamics

In this chapter the equations for the kinematic and dynamics of a 3-DOF Delta Robot will be derived. Analysis of inverse & forward kinematics, singularity and workspace of the Delta Robot will be discuss and derived in detail.

2.1. Schematics of Delta Robot

Fig. 2.1 shows the main components of a Delta Robot, which consists of three or four closed-loop kinematic chains. The Robot has three (optionally four) degrees of freedom (DOF). The parallelograms ensure the constant orientation between the fixed and the mobile platform, allowing only translation movements of the end-effector. The end-effector of the manipulator is located on the mobile platform [6].Parallel Robot can move products in a three dimensional Cartesian coordinate system.

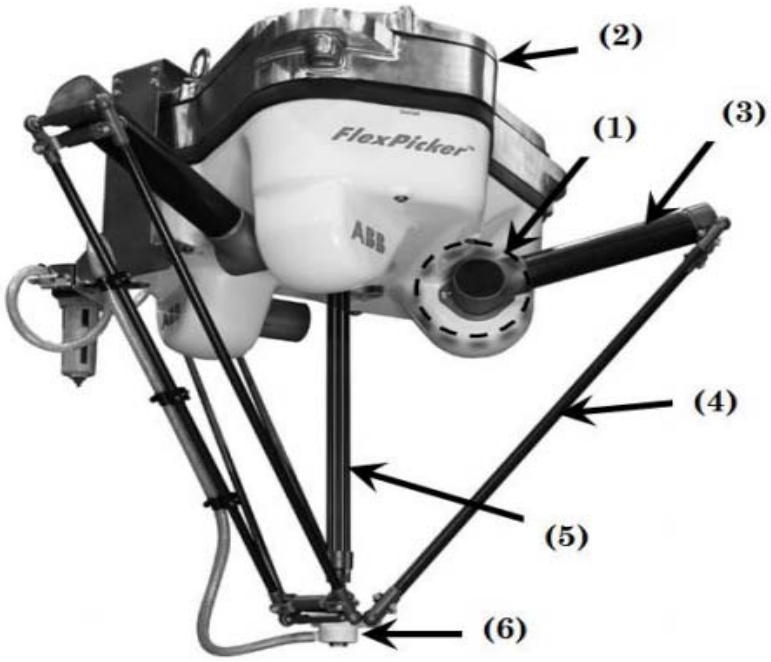


Fig. 2.1 .Parallel Delta Robot Components (www.abb.com)

The combination of the constrained motion of the three arms connecting the traveling plate to the base plate ensues in a resulting three translator degrees of freedom (DOF). As an option, with a

rotating axis at the Tool Center Point (TCP), four DOF are possible. The Robot consists of, consider Fig.2.1:

1) Three Actuators.
2) Base plate.
3) Upper Robot arm.
4) Lower Robot arm (Forearm).
5) Rotation arm (optional, 4-DOF).
6) Travelling plate, TCP or end-effector.

The upper Robot arms are mounted direct to the actuators to guarantee high stability and the three actuators are rigidly mounted on the base plate with 120° in between. Each of the three lower Robot arms consists of two parallel bars, which connects the upper arm with the travelling plate via ball joints. Lower frictional forces result from this. The wear reduces respectively as a result. The travelling plate (TCP) always stays parallel to the base platform and its orientation around the axis perpendicular to the base plate is constantly zero. The moving platform is connected to the fixed base through three parallel kinematic chains. Each chain contains a revolute joint activated by actuators in base platform. The motion is transmitted to the mobile platform through parallelograms. A fourth bar, rotational axes, is available for the Robot mechanics as an option. The actuator for this axis is then mounted on the upper side of the Robot base plate. The bar is connected directly to the tool and ensures for an additional rotation motion [7].

2.2. Kinematics

"Kinematics is the science of motion which treats motion without regard to the forces which cause it. Within the science of kinematics one studies the position, velocity, acceleration and all higher order derivatives of the position variables, with respect to time or any other variable(s)" [8].Generally, the kinematics of a closed loop manipulator is more difficult to calculate as compared to the kinematics of open chain. The kinematics for an industrial Robot can be distributed into three different problem formulations, Forward Kinematics, Inverse Kinematics and the Velocity Kinematics. As shown in the figure Delta Robot consists of two equilateral triangles platforms (the base plate and the travelling plate). The joint angles are θ_1,

θ_2 and θ_3, and point E_0 is the end-effector position with coordinates (x_0, y_0, z_0). To solve inverse kinematics problem we have to create function with E_0 coordinates (x_0, y_0, z_0) as parameters which returns $(\theta_1, \theta_2, \theta_3)$. Forward kinematics function gets $(\theta_1, \theta_2, \theta_3)$ and returns (x_0, y_0, z_0).

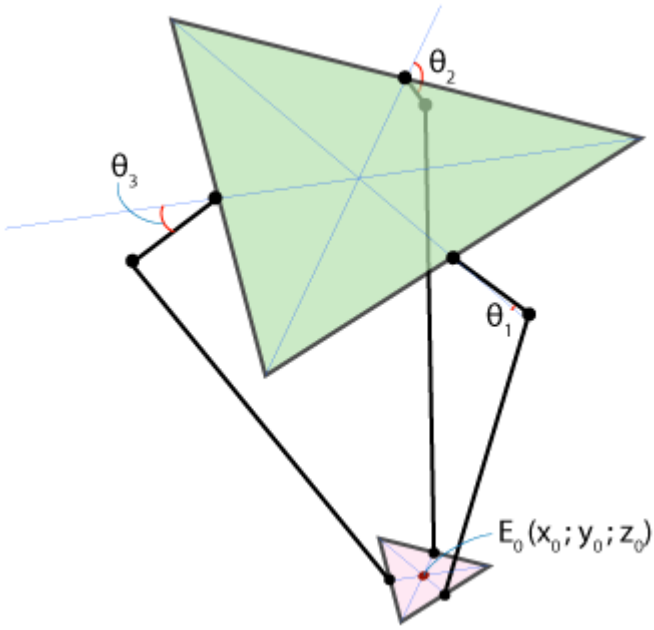


Fig. 2.2 .Delta Robot's joint angle and end-effector orientation

2.2.1. Inverse (Indirect) Kinematics

The inverse kinematics of a parallel manipulator determines the θ_i angle of each actuated revolute joint given the (x, y, z) position of the traveling plate in base-frame. First, let's determine some key parameters of the Robot's geometry. Let's designate the side of the fixed triangle as f, the side of the End effector triangle as e, the length of the upper arm as r_f, and the length of the parallelogram arm as r_e. These are physical parameters which are determined by design of the Robot. The reference frame will be chosen with the origin at the center of symmetry of the fixed triangle, as shown below, so z-coordinate of the End effector will always be negative.

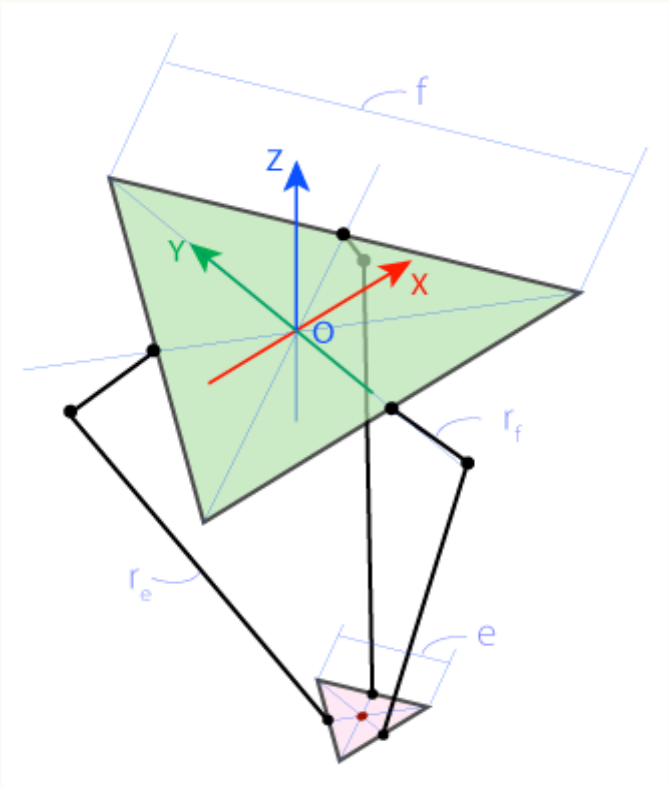


Fig. 2.3 .Delta Robot's coordinate system with dimensions

Because of Robot's design joint F_1J_1 (see fig. below) can only rotate in YZ plane, forming circle with center in point F_1 and radius r_f. As opposed to F_1J_1, and E_1 are so-called universal joints, which means that E_1J_1 can rotate freely relatively to E_1, forming sphere with center in point E_1 and radius r_e.

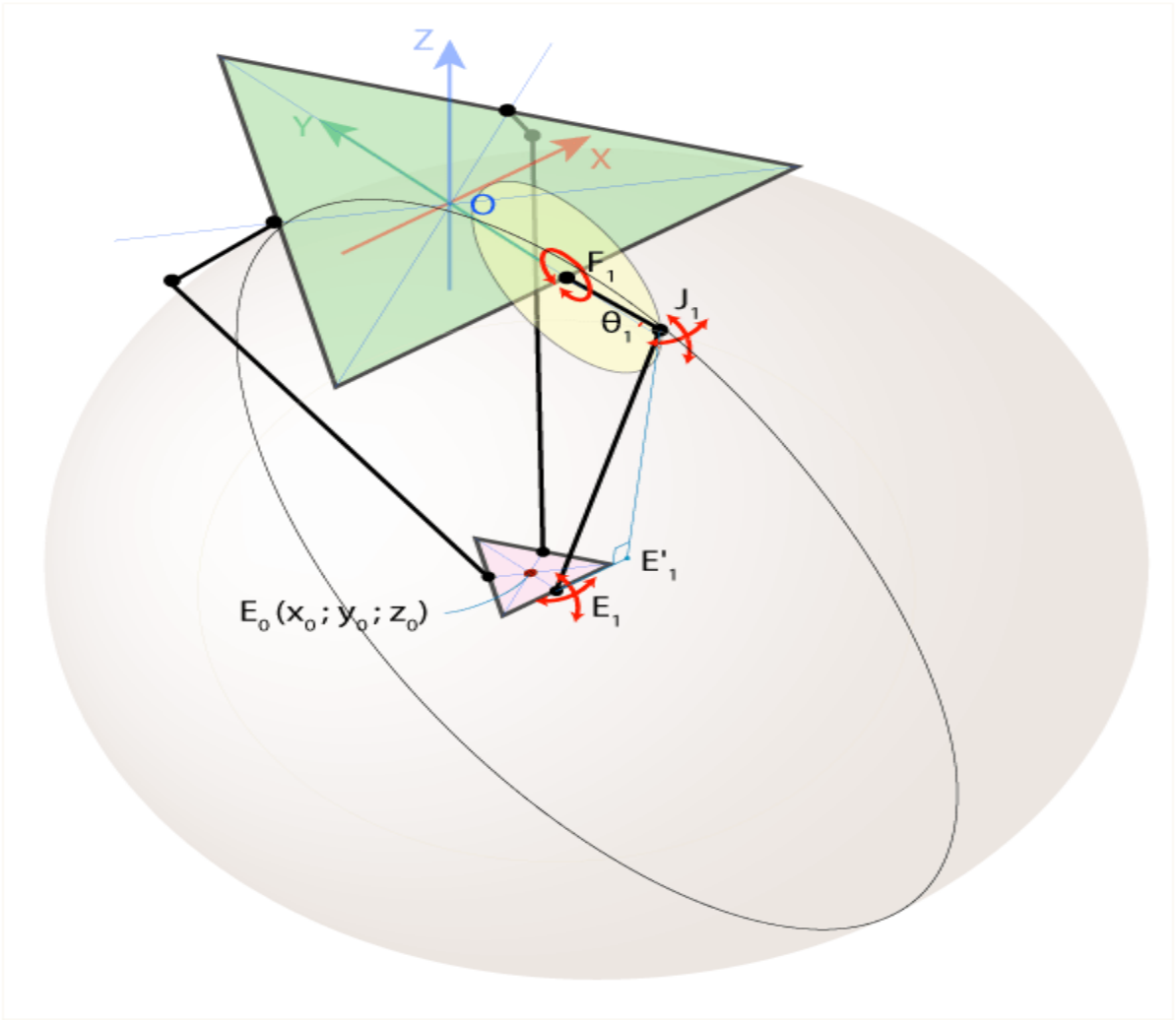


Fig. 2.4 .Intersection of sphere and circle from the projected lower leg and upper leg

Intersection of this sphere and YZ plane is a circle with center in point E'_1 and radius E'_1J_1, where E'_1 is the projection of the point E_1 on YZ plane. The point J_1 can be found now as intersection of two circles of known radius with centers in E'_1 and F_1 (we should choose only one intersection point with smaller Y-coordinate). And if J_1 is known, we can calculate angle θ_1.

The corresponding equations and YZ plane view are shown below .

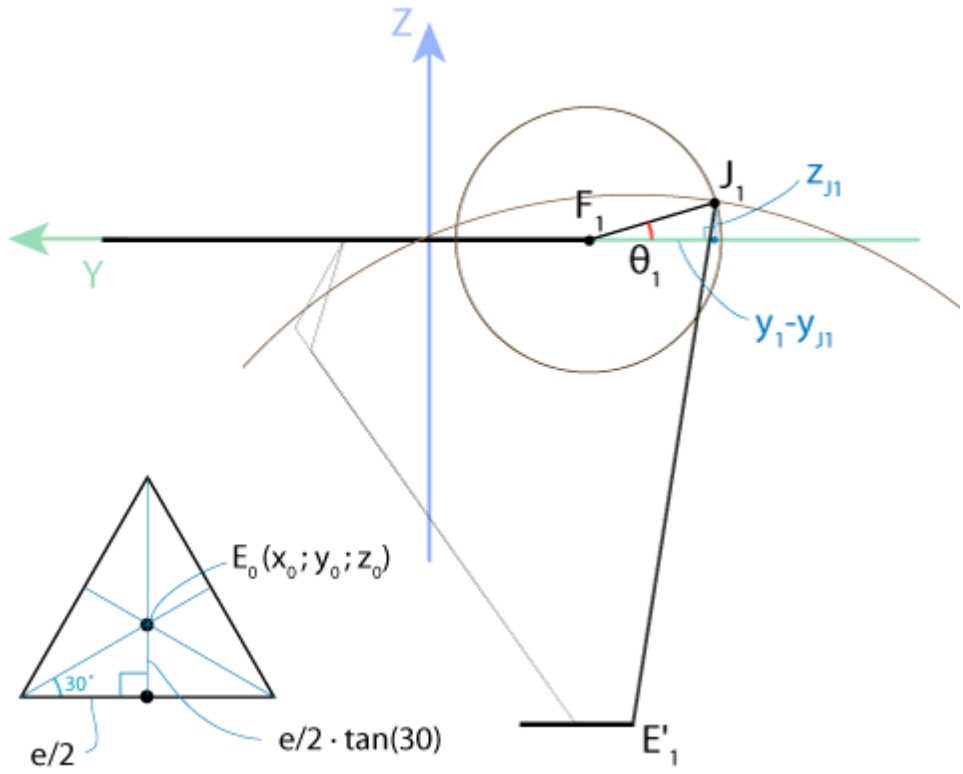


Fig. 2.5 .YZ-plane & base dimensions

$$E(X_O, Y_O, Z_O) \tag{2.1}$$

$$EE_1 = \frac{e}{2}\tan(30) = \frac{e}{2\sqrt{3}}$$

$$E_1(X_O, Y_O - \frac{e}{2\sqrt{3}}, Z_O)$$

$$E_1'(X_O, Y_O - \frac{e}{2\sqrt{3}}, Z_O)$$

$$EE_1' = X_O$$

$$E_1'J_1 = \sqrt{E_1J_1^2 - E_1E_1'^2} = \sqrt{r_e^2 - X_0^2} \quad (2.2)$$

$$F_1(0, \tfrac{f}{2\sqrt{3}}, 0) \quad (2.3)$$

$$(y_{J1} - y_{f1})^2 + (z_{J1} - z_{f1})^2 = r_f^2 \quad (2.4)$$

$$(y_{J1} - y_{E'1})^2 + (z_{J1} - z_{E'1})^2 = r_e^2 - X_0^2$$

$$(y_{J1} - \frac{f}{2\sqrt{3}})^2 + (z_{J1})^2 = r_f^2$$

$$(y_{J1} - y_0 + \frac{e}{2\sqrt{3}})^2 + (z_{J1} - z_0)^2 = r_e^2 - X_0^2$$

$$\theta_1 = arc\tan\frac{z_{J1}}{(y_{f1} - y_{J1})} \quad (2.5)$$

Such algebraic simplicity follows from good choice of reference frame: joint F_1J_1 moving in YZ plane only, so we can completely omit X coordinate. To take this advantage for the remaining angles θ_2 and θ_3, the symmetry of Delta Robot property is utilized. First, let's rotate coordinate system in XY plane around Z-axis through angle of 120 degrees counterclockwise, as it is shown below.

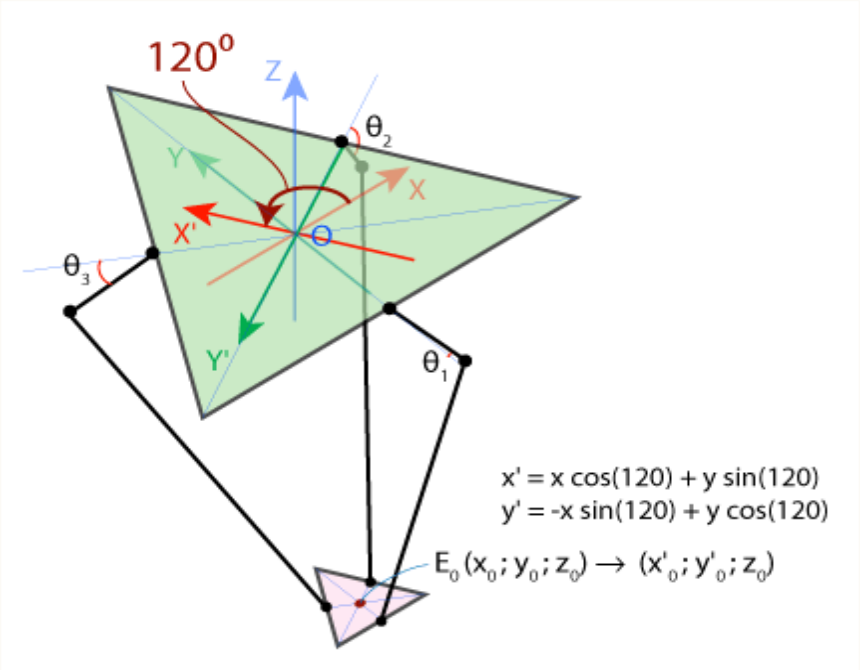


Fig. 2.5.Delta Robot symmetry and coordinate rotation

We've got a new reference frame $X'Y'Z'$, and with this frame it is simple to find angle θ_2 using the same algorithm that was used to find θ_1. The only change needed is to determine new coordinates X'_0 and Y'_0 for the point E_0, which can be easily done using corresponding rotation matrix. To find angle θ_3 we have to rotate reference frame clockwise. In general, there are a total of eight possible Robot postures corresponding to a given end-effector location [9].

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$X' = \cos(\alpha) . X - sin(\alpha) . y \tag{2.6}$$

$$Y' = sin(\alpha) . X + cos(\alpha) . y$$

$$Z' = Z$$

Where α is the angle of rotation about z axis, From Eq. (2.5) yields:

$$\theta_{2,3} = f(X', Y', Z') \tag{2.7}$$

Hence, there are generally two solutions of θ_1 and therefore two configuration of the kinematics chain corresponding to each end-effector location. When θ_1 has a double root, the two links of the kinematics chain are in a fully stretched-out or folded-back configuration named singular configuration. When θ_1 yields no real solution, the specified end-effector location is not reachable. Despite of the two possible solutions, only the negative root have to be taken because the positive one could cause interference between the elements of the Robot.

2.2.2. Forward (Direct) Kinematics

The forward kinematics also called the direct kinematics of a Parallel manipulator determines the position of the traveling plate in base-frame, given the configuration of each angle of the actuated revolute joints.

Now the three joint angles $(\theta_1, \theta_2, \theta_3)$ are given, and we need to find the coordinates (x_0, y_0, z_0) of end-effector point of E_0.As we know angles theta, we can easily find coordinates

of points J_1, J_2 and J_3 (see fig. below). Joints J_1E_1, J_2E_2 and J_3E_3 can freely rotate around points J_1, J_2 and J_3 respectively, forming three spheres with radius r_e.

Now let's do the following: move the centers of the spheres from points J_1, J_2 and J_3 to the points J'_1, J'_2 and J'_3 using transition vectors E_1E_0, E_2E_0 and E_3E_0 respectively. After this transition all three spheres will intersect in one point: E_0, as it is shown in fig. below:

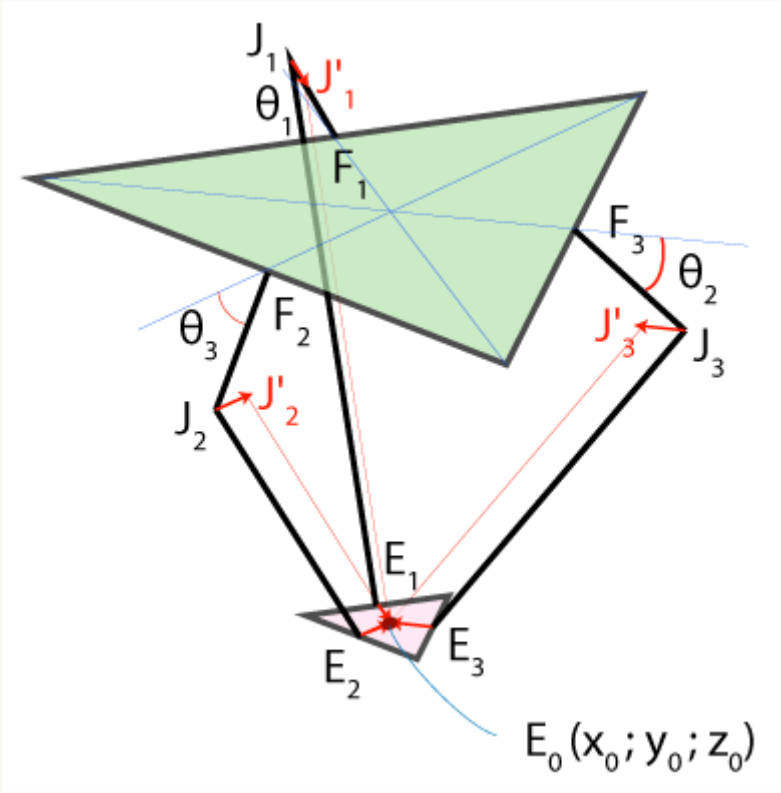
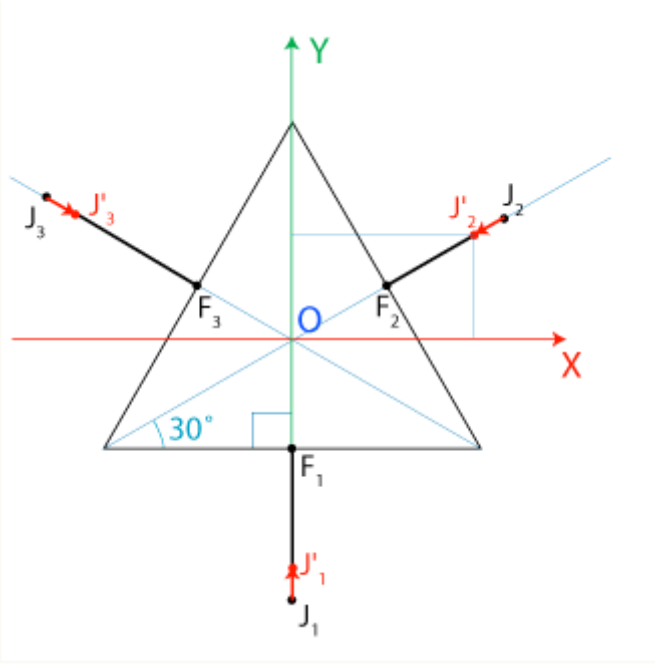


Fig. 2.6 .Delta Robot with coordinate point projected to form spheres

So, to find coordinates (x_0, y_0, z_0) of point E_0, we need to solve set of three equations like $(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2 = r_e^2$, where coordinates of sphere centers (x_j, y_j, z_j) and radius r_e are known. First, let's find coordinates of points J'_1, J'_2, J'_3:



$$OF_1 = OF_2 = OF_3 = \frac{f}{2}\tan(30) = \frac{f}{2\sqrt{3}}$$

$$JJ_1' = JJ_1' = JJ_1' = \frac{e}{2}\tan(30) = \frac{e}{2\sqrt{3}}$$

$$F_1J_1 = r_f COS(\theta_1)$$

$$F_2J_2 = r_f COS(\theta_2) \quad (2.8)$$

$$F_3J_3 = r_f COS(\theta_3)$$

Fig. 2.7 coordinate point projection to the center of end-effector

$$J_1'(0, -\frac{(f-e)}{2\sqrt{3}} - r_f\cos(\theta_1), -r_f\sin(\theta_1)$$

$$J_2'(\left[\frac{(f-e)}{2\sqrt{3}} + r_f\cos(\theta_2)\right]\cos(30), \left[\frac{(f-e)}{2\sqrt{3}} + r_f\cos(\theta_2)\right]\sin(30), -r_f sin(\theta_2) \quad (2.9)$$

$$J_3'(\left[\frac{(f-e)}{2\sqrt{3}} + r_f\cos(\theta_3)\right]\cos(30), \left[\frac{(f-e)}{2\sqrt{3}} + r_f\cos(\theta_3)\right]\sin(30), -r_f sin(\theta_3)$$

In the following equations , designate coordinates of points J_1, J_2, J_3 as (x_1, y_2, z_3), (x_2, y_2, z_2) and (x_3, y_3, z_3) Please note that $x_0 = 0$. Here are the equations of three spheres:

$$X^2 + (Y - Y_1)^2 + (Z - Z_1)^2 = r_e^2$$

$$(X - X_2)^2 + (Y - Y_2)^2 + (Z - Z_2)^2 = r_e^2 \quad (2.10)$$

$$(X - X_3)^2 + (Y - Y_3)^2 + (Z - Z_3)^2 = r_e^2$$

$$X^2 + Y^2 + Z^2 - 2Y_1Y - 2Z_1Z = r_e^2 - Y_1^2 - Z_1^2 \quad (2.11)$$

$$X^2 + Y^2 + Z^2 - 2X_2X - 2Y_2Y - 2Z_2Z = r_e^2 - X_2^2 - Y_2^2 - Z_2^2 \quad (2.12)$$

$$X^2 + Y^2 + Z^2 - 2X_3X - 2Y_3Y - 2Z_3Z = r_e^2 - X_3^2 - Y_3^2 - Z_3^2 \quad (2.13)$$

$$W_i = X_i^2 + Y_i^2 + Z_i^2$$

subtract equation 2.11 from 2.10

$$X_2X + (Y_1 - Y_2)Y + (Z_1 - Z_2)Z = \frac{W_1 - W_2}{2} \quad (2.14)$$

subtract equation 2.12 from 2.10

$$X_3X + (Y_1 - Y_3)Y + (Z_1 - Z_3)Z = \frac{W_1 - W_2}{2} \quad (2.15)$$

subtract equation 2.11 from 2.13

$$(X_2 - X_3)X + (Y_2 - Y_3)Y + (Z_2 - Z_3)Z = \frac{W_1 - W_2}{2} \quad (2.16)$$

$$X = a_1z + b_1 \quad (2.16) \qquad y = a_2z + b_2 \quad (2.17)$$

$$a_1 = \frac{1}{d}[(z_2 - z_1)(y_3 - y_1) - (z_3 - z_1)(y_2 - y_1)]$$

$$a_2 = -\frac{1}{d}[(z_2 - z_1)X_3 - (z_3 - z_1)X_2]$$

$$b_1 = -\frac{1}{2d}[(W_2 - W_1)(y_3 - y_1) - (W_3 - W_1)(y_2 - y_1)]$$

$$b_2 = \frac{1}{2d}[(W_2 - W_1)X_3 - (W_3 - W_1)X_2]$$

$$d = (y_2 - y_1)X_3 - (y_3 - y_1)X_2$$

now substitute

$$(a_1^2 + a_2^2 + 1)z^2 + 2(a_1 + a_2(b_2 - y_1) - z_1)z + (b_1^2 + (b_2 - y_1)^2 + z_1^2 - r_e^2) = 0 \quad (2.18)$$

Finally, we need to solve this quadric equation and find z0 (we should choose the smallest negative equation root), and then calculate x_0 and y_0 from eq. (2.16) and (2.17).

2.2.3. Velocity Kinematics

The most relevant loop should be picked up for the intended Jacobian analysis. Let $\vec{\theta}$ be the vector made up of actuated joint variables and $\vec{P}$ is the position vector of the moving platform.

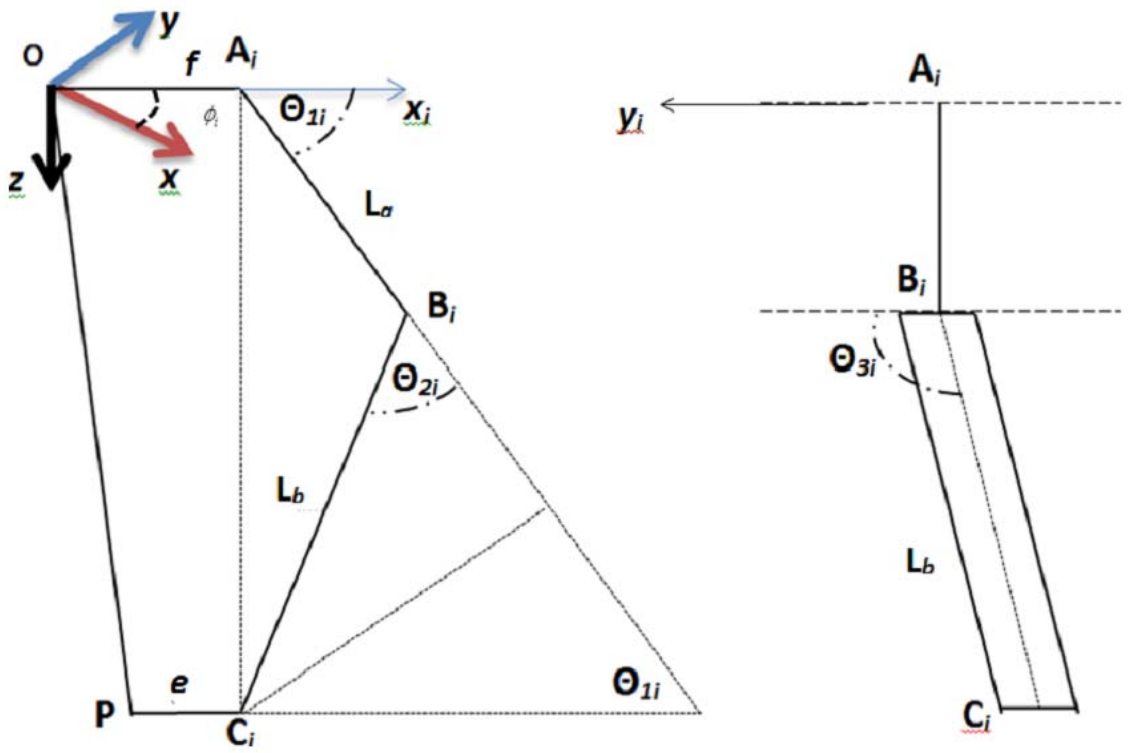


Fig. 2.8.Projection of link i on $x_i z_i$ plane, (b) end on view

$$\vec{\theta} = \theta_{1i} = \begin{bmatrix} \theta_{11} \\ \theta_{12} \\ \theta_{13} \end{bmatrix}, \ \vec{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.19}$$

The Jacobian matrix will be derived by differentiating the appropriate loop closure equation and rearranging the result in the following form.

$$J_\theta = \begin{bmatrix} \dot{\theta}_{11} \\ \dot{\theta}_{12} \\ \dot{\theta}_{13} \end{bmatrix} = J_p \begin{bmatrix} \dot{P}_x = V_x \\ \dot{P}_y = V_y \\ \dot{P}_z = V_z \end{bmatrix} \tag{2.20}$$

where $V_x, V_y \ and \ V_z$ are the $x, y, and \ z$ components of the velocity of the point P on the moving platform in the xyz frame. In order to arrive at the above form of the equation, we look at the loop $OA_iB_iC_iP$. The corresponding closure equation in the $x_i y_i z_i$ frame is

$$\overrightarrow{OP} + \overrightarrow{PC_\iota} = \overrightarrow{OA_\iota} + \overrightarrow{A_\iota B_\iota} + \overrightarrow{B_\iota C_\iota} \tag{2.21}$$

In the matrix form we can write it as

$$\begin{bmatrix} P_x\cos(\phi_i) - P_y\sin(\phi_i) \\ P_x\sin(\phi_i) + P_y\cos(\phi_i) \\ P_z \end{bmatrix} = \begin{bmatrix} f \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} e \\ 0 \\ 0 \end{bmatrix} + r_f \begin{bmatrix} \cos(\theta_{1i}) \\ 0 \\ \sin(\theta_{1i}) \end{bmatrix} + r_e \begin{bmatrix} \sin(\theta_{3i})\ \cos(\theta_{2i} + \theta_{1i}) \\ \sin(\theta_{3i}) \\ \sin(\theta_{3i})\ \cos(\theta_{2i} + \theta_{1i}) \end{bmatrix} \quad (2.22)$$

Time differentiation of this equation leads to the desired Jacobian equation. The loop closure equation Eq.2.18 can be re-written as

$$\vec{P} + \vec{e} = \vec{f} + \overrightarrow{a_i} + \overrightarrow{b_i} \quad (2.23)$$

Where a_i and b_i represents vectors $\overrightarrow{A_iB_i}$ and $\overrightarrow{B_iC_i}$ respectively.

Differentiating Eq.2.20 with respect to time and using the fact that f is a vector characterizing the fixed platform, and e is a vector characterizing the moving platform

$$\dot{\vec{P}} = \vec{v} = \vec{f} + \dot{\overrightarrow{a_i}} + \dot{\overrightarrow{b_i}} \quad (2.24)$$

The linear velocities on the right hand side of Eq.2.20 can be readily converted into the angular velocities by using the well-known identities.

Thus

$$\vec{v} = \overrightarrow{W_{ai}} * \overrightarrow{a_i} + \overrightarrow{W_{bi}} * \overrightarrow{b_i} \quad (2.25)$$

W_{ai} and W_{bi} is the angular velocity of the link i. To eliminate W_{bi} , it is necessary to dot-multiply both sides of Eq. 2.20 and bi. Therefore

$$b_i.\vec{v} = \overrightarrow{W_{ai}}.(\overrightarrow{a_i} + \overrightarrow{b_i}) \quad (2.26)$$

Rewriting the vectors of Eq.2.21 in the $x_iy_iz_i$ coordinate frame leads to

$$a_i = r_f \begin{bmatrix} \cos(\theta_{1i}) \\ 0 \\ \sin(\theta_{1i}) \end{bmatrix}, \qquad b_i = r_e \begin{bmatrix} \sin(\theta_{3i})\ \cos(\theta_{2i} + \theta_{1i}) \\ \sin(\theta_{3i}) \\ \sin(\theta_{3i})\ \cos(\theta_{2i} + \theta_{1i}) \end{bmatrix}$$

$$W_{ai}\begin{bmatrix}0\\-\dot{\theta}_{1i}\\0\end{bmatrix}\ ,\quad v_i=\begin{bmatrix}v_x\cos(\phi_i)-v_y\sin(\phi_i)\\v_x\sin(\phi_i)+v_y\cos(\phi_i)\\v_z\end{bmatrix}$$

Substituting the values of a_i, b_i, v_i and v in Eq.2.21 leads to

$$J_{ix}V_x+J_{iy}V_y+J_{iz}V_z=r_f\sin(\theta_{2i})\sin(\theta_{3i})\,\dot{\theta}_{1i} \tag{2.27}$$

Where

$$J_{ix}=\cos(\theta_{2i}+\theta_{1i})\sin(\theta_{3i})\ \cos(\phi_i)-\cos(\theta_{3i})\ \sin(\phi_i)$$

$$J_{iy}=\cos(\theta_{2i}+\theta_{1i})\sin(\theta_{3i})\ \sin(\phi_i)+\cos(\theta_{3i})\ \cos(\phi_i)$$

$$J_{iz}=\sin(\theta_{2i}+\theta_{1i})\sin(\theta_{3i})$$

Expanding Eq.2.21 for i = 1, 2 and 3 yields three scalar equations which can be assembled into a matrix form as

$$J_xV=J_q\dot{q} \tag{2.28}$$

where

$$J_x=\begin{bmatrix}J_{1x}&J_{1y}&J_{1z}\\J_{2x}&J_{2y}&J_{2z}\\J_{3x}&J_{3y}&J_{3z}\end{bmatrix}$$

$$J_q=\begin{bmatrix}\sin(\theta_{21})\sin(\theta_{31})&0&0\\0&\sin(\theta_{22})\sin(\theta_{32})&0\\0&0&\sin(\theta_{23})\sin(\theta_{33})\end{bmatrix}$$

$$\dot{q}=\begin{bmatrix}\dot{\theta}_{11}&\dot{\theta}_{12}&\dot{\theta}_{13}\end{bmatrix}$$

After algebraic manipulations, it is possible to write

$$V=J\dot{q} \tag{2.29}$$

where

$$J = J_x{}^{-1} J_q = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \\ \frac{\partial x}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \\ \frac{\partial x}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \end{bmatrix} \quad (2.24)$$

2.2.4. Forward and Inverse Singularity Analysis

From Eq.2.28 it can be observed that singularity occurs:

1. When $det(J_q) = 0$. This means that either $\theta_{2i} = 0$ or θ_{3i} or π for i = 1,2,3
2. When $det(J_x) = 0$. This means that $\theta_{2i} + \theta_{3i} = 0$ or π or $\theta_{3i} = 0$ or π for i = 1,2,3
3. When $det(J_q) = 0$ *and* $det(J_x) = 0$. This situation occurs when $\theta_{3i} = 0$ or π for i = 1,2,3

In summary, singularity of the parallel manipulator occurs:

1. When all three pairs of the follower rods are parallel. Therefore, the moving platform has three degrees of freedom and moves along a spherical surface and rotates about the axis perpendicular to the moving platform
2. When two pairs of the follower rods are parallel. The moving platform has one degree of freedom; i.e. the moving platform moves in one direction only.
3. When two pairs of the follower rods are in the same plane or two parallel planes. The moving platform has one degree of freedom; i.e. the moving platform rotates about the horizontal axis only.

2.3. Dynamics

Dynamics is the science of motion, that describes why and how a motion occurs when forces and moments are applied on massive bodies. The motion can be considered as evolution of the position, orientation, and their time derivatives. In Robotics, the dynamic equation of motion for manipulators is utilized to set up the fundamental equations for control. The links and arms in a Robotic system are modeled as rigid bodies[13].

Therefore, the dynamic properties of the rigid body take a central place in Robot dynamics. Since the arms of a Robot may rotate or translate with respect to each other, translational and rotational equations of motion must be developed and described in body-attached coordinate frames B1, B2, B3 … or in the global reference frame G.

There are basically two problems in Robot dynamics.

Problem1 (forward dynamics). Given the forces, work out the accelerations. We want the links of a Robot to move in a specified manner. What forces and moments are required to achieve the motion?

The first Problem is called direct dynamics and is easier to solve when the equations of motion are in hand because it needs differentiating of kinematics equations. The first problem includes Robot statics because the specified motion can be the rest of a Robot. In this condition, the problem reduces finding forces such that no motion takes place when they act. However, there are many meaningful problems of the first type that involve Robot motion rather than rest. An important example is that of finding the required forces that must act on a Robot such that its end-effector moves on a given path and with a prescribed time history from the start configuration to the final configuration.

Problem2 (inverse dynamics). Given the accelerations, work out the forces. The applied forces and moments on a Robot are completely specified. How will the Robot move?

Inverse dynamics is more difficult to solve since it needs integration of equations of motion. However, the variety of the applied problems of the second type is interesting. Problem 2 is essentially a prediction since we wish to find the Robot motion for all future times when the initial state of each link is given. The inverse dynamics problem is to find the actuator torques and/or forces required to generate a desired trajectory of the manipulator.[10]

It is often convenient to express the dynamic equations of a manipulator in a single equation that hides some of the details, but shows some of the structure of the equations. The state-space equation when the Newton—Euler equations are evaluated symbolically for any manipulator, they yield a dynamic equation that can be written in the form.

$$T = M(\theta)\ddot{\theta} + \mathrm{V}(\theta, \dot{\theta}) + \mathrm{G}(\theta) \qquad (2.30)$$

where $M(\theta)$ is n x n mass matrix of the manipulator, $\mathrm{V}(\theta, \dot{\theta})$ is a n x 1 vector of centrifugal and Coriolis terms, and $\mathrm{G}(\theta)$ is an n x 1 vector of gravity terms. We use the term state-space equation because the term $\mathrm{V}(\theta, \dot{\theta})$ has both position and velocity dependence. Each element

of $M(\theta)$ and $G(\theta)$ is a complex function that depends on θ, the position of all the joints of the manipulator. Each element of $V(\theta, \dot{\theta})$ is a complex function of both θ and $\dot{\theta}$.

We may separate the various types of terms appearing in the dynamic equations and form the mass matrix of the manipulator, the centrifugal and Coriolis vector, and the gravity vector [10].

Different modeling techniques can be used to find the dynamic model of parallel Robot [11]:

1. Lagrange method
2. Newton-Euler method and
3. Virtual work principle

In this thesis a dynamic model based on the virtual work principle as follows.

2.3.1. Virtual Work Dynamics

In this section, we will perform the inverse dynamic modeling of the parallel manipulator based upon the principle of virtual work. Virtual work is the work done by a real force acting through a virtual displacement or a virtual force acting through a real displacement. A virtual displacement is any displacement consistent with the constraints of the structure, i.e., that satisfy the boundary conditions at the supports. A virtual force is any system of forces in equilibrium.[12]

The inverse dynamics problem is to find the actuator torques and/or forces required to generate a desired trajectory of the manipulator . Without losing generality of model, we can simplify the dynamic problem by the following hypotheses[13]:

The connecting rods of lower links can be built with light materials such as the aluminum alloy, so

- ✓ The lower links rotational inertias are neglected.
- ✓ The mass of each lower links, is divided evenly and concentrated at the two endpoints of the parallelogram.

Also it is supposed that:

- ✓ The friction forces in joints are neglected.
- ✓ No external forces suffered.

We consider that

$T = [T_1, T_2, T_3]$ and $\delta\theta = [\delta\theta_1, \delta\theta_2, \delta\theta_3]$ are the vector of actuator torques and vector of corresponding virtual angular displacements. Furthermore, $\delta p = [\delta_x, \delta_y, \delta_z]$ represents the virtual linear displacements vector of the mobile platform. We can derive the following equations by applying the virtual work principle.

$$T^T \delta\theta + \mathrm{M}_{\mathrm{Ga}}^{\mathrm{T}} \delta\theta + \mathrm{F}_{\mathrm{Gp}}^{\mathrm{T}} \delta\mathrm{p} - \mathrm{M}_{\mathrm{a}}^{\mathrm{T}}\, \delta\theta - \mathrm{F}_{\mathrm{p}}^{\mathrm{T}}\, \delta\mathrm{p} = 0 \tag{2.31}$$

where

$$M_{Ga} = \left(\tfrac{1}{2} M_a + M_b \right) . g . la . I [\cos(\theta_1) + \cos(\theta_2) + \cos(\theta_3)]^T \tag{2.32}$$

is the upper links gravity torques vector M_a and M_b are mass of upper link and each connecting rod of lower link, respectively. Here g denotes the gravity acceleration, and I represent the 3x3 identity matrix.

$$F_{Gp} = [0\ 0 - (M_{tcp} + 3M_b)]^T \tag{2.33}$$

Denotes the mobile platform gravity force vector, and m_{tcp} is mass of the mobile platform.

$$M_a = I_a \ddot{\theta} = I_a [\ddot{\theta}_1\ \ddot{\theta}_2\ \ddot{\theta}_3]^T \tag{2.34}$$

where

$$I_a = \left(\frac{1}{3} M_a L_a^2 + M_b L_b^2 \right) . I$$

Represents the upper links inertia torques vector and denotes the upper links inertial matrix with respect to the fixed frame $O\{x, y, z\}$, and,

$$F_p = M_p \ddot{P} = (M_{tcp} + 3M_b) . I . [\ddot{X}\ \ddot{Y}\ \ddot{Z}\ ]^T \tag{2.35}$$

Denote the mobile platform inertial forces vector. Eq.2.24 in section 2.2.4 can be rewritten to,

$$\dot{P} = J\dot{\theta} \tag{2.36}$$

Consequently,

$$\delta p = J\delta\theta \tag{2.37}$$

Substituting Eq. 2.29 into Eq. 2.25 results,

$$(T^T + M_{Ga}^T + F_{Gp}^T J - M_a^T - F_P^T J)\delta\theta = 0 \tag{2.38}$$

Eq. 2.30 holds for any virtual displacements $\delta\theta$, so we have

$$T = M_a + J^T F_p - M_{Ga} - J^T F_{Gp} \tag{2.39}$$

Substitute Eqs.2.26 and 2.27 into Eq. 2.31, allows the generation of

$$T = I_a\ddot{\theta} + J^T M_P \ddot{P} - M_{Ga} - J^T F_{Gp} \tag{2.40}$$

Differentiating Eq. 2.28 with respect to time, yields

$$\ddot{P} = J\ddot{\theta} + \dot{J}\dot{\theta} \tag{2.41}$$

Substituting Eq. 2.33 into Eq. 2.32, we can derive that

$$T = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) \tag{2.42}$$

The previous equation described in Eq. 2.25 represents the dynamic model of parallel manipulator in joint space. Here, $\theta \in R^3$ is the controlled variables, and

$$M(\theta) = I_a + J^T M_P J \tag{2.43}$$

Denotes a symmetric positive definite inertial matrix, that $M(\theta) \in R^{3x3}$

$$V(\theta, \dot{\theta}) = J^T M_P \dot{J} \tag{2.44}$$

Where $V(\theta, \dot{\theta}) \in R^{3x3}$ is the centrifugal and Coriolis forces matrix, and

$$G(\theta) = -M_{Ga} - J^T F_{Gp} \tag{2.45}$$

Represents the vector of gravity forces, and $G(\theta) \in R^3$

2.3.2. Non-Rigid Body Effects

It is important to realize that the dynamic equations we have derived do not encompass all the effects acting on a manipulator. They include only those forces which arise from rigid body mechanics. The most important source of forces that are not included is friction. All mechanisms are, of course, affected by frictional forces. In present-day manipulators, in which significant gearing is typical, the forces due to friction can actually be quite large - perhaps equaling 25% of the torque required to move the manipulator in typical situations[13]. In order to make dynamic equations reflect the reality of the physical device, it is important to model (at least approximately) these forces of friction. A very simple model for friction is viscous friction, in which the torque due to friction is proportional to the velocity of joint motion. Thus, we have

$$T_{friction} = v\dot{\theta} \tag{2.46}$$

where v is a Viscous-friction constant. Another possible simple model for friction, Coulomb friction, is sometimes used. Coulomb friction is constant except for a sign dependence on the joint velocity and is given by

$$T_{friction} = C * sgn(\dot{\theta}) \tag{2.47}$$

where c is a Coulomb-friction constant. The value of c is often taken at one value when $\dot{\theta} = 0$ the static coefficient, but at a lower value, the dynamic coefficient, when $\dot{\theta} \neq 0$, whether a joint of a particular manipulator exhibits Viscous or Coulomb friction is a complicated issue of lubrication and other effects. A reasonable model is to include both, because both effects are likely:

$$T_{friction} = v\dot{\theta} + C * sgn(\dot{\theta}) \tag{2.48}$$

It turns out that, in many manipulator joints, friction also displays a dependence on the joint position. A major cause of this effect might be gears that are not perfectly round-their eccentricity would cause friction to change according to joint position. So a fairly complex friction model would have the form

$$T_{friction} = f(\theta, \dot{\theta}) \tag{2.49}$$

These friction models are then added to the other dynamic terms derived from the rigid-body model, yielding the more complete model

$$T = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) + F(\theta, \dot{\theta}) \tag{2.50}$$

2.3.3. Actuator Dynamics

The system is basically composed of dc motor, precision revolute bearing and coupling elements. Dc motor model is given below. The symbols represent the following variables here θ_m is the motor position (radian), τ_m is the produced torque by the motor (Nm), τ_1 is the load torque, v_a is the armature voltage (V), La is the armature inductance (H), Ra is the armature resistance (Ω), Em is the reverse EMF (V), Ia is the armature current (A), Kb is the reverse EMF constant, Km is the torque constant [14].

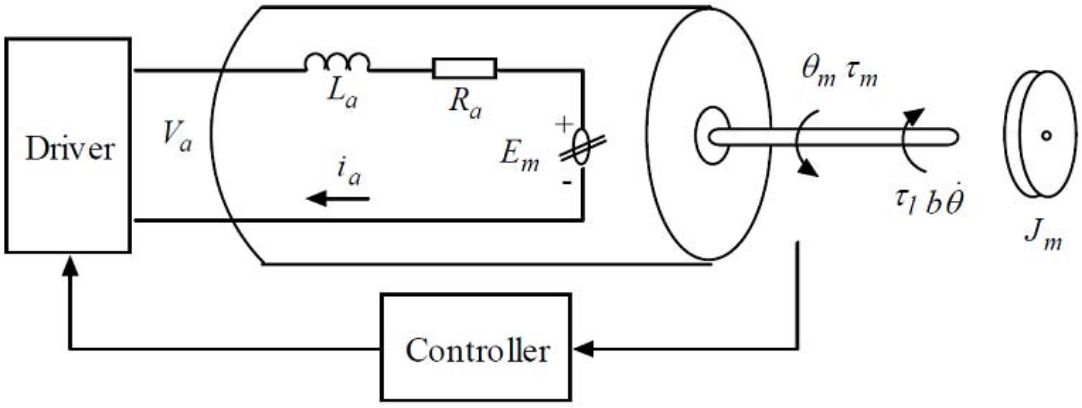


Fig. 2.9.DC motor model

$$L_a \frac{dI_a}{dt} + R_a i_a = V_a - E_a$$

$$E_m = K_b \frac{d\ddot{\theta}_m}{dt} \tag{2.51}$$

$$\tau_m = K_m\, i_a$$

$$\tau_m - \tau_1 = J_m \frac{d^2\theta_m}{dt^2}$$

On the assumption of a rigid transmission and with no backlash the relationship between the input forces (velocities) and the output forces (velocities) are purely proportional. This gives,

$$\theta_l = K_r \theta_l \tag{2.52}$$

Where, constant K_r is a parameter which describes the gear reduction ratio . τ_1 is the load torque at the Robot axis and τ_m is the torque produced by the actuator at the shaft axis. In view of Eq. 2.52 one can write

$$\tau_m = \frac{\tau_1}{K_r} \tag{2.53}$$

To simulate the motion of a manipulator, we must make use of a model of the dynamics such as the one we have just developed. Given the dynamics written in closed form as in (2.40), simulation requires solving the dynamic equation for acceleration:

$$\ddot{\theta} = M^{-1}(\theta)[\tau - V(\theta, \dot{\theta}) - G(\theta) - F(\theta, \dot{\theta}) \tag{2.54}$$

We can then apply any of several known numerical integration techniques to integrate the acceleration to compute future positions and velocities. Given initial conditions on the motion of the manipulator, usually in the form.

$$\theta(0) = \theta_0 \tag{2.55}$$

$$\dot{\theta}(0) = 0$$

3. Chapter Three: Sliding Mode Control

The Robotic manipulator is a complex system with uncertainties and parameter variations. Moreover, presence of friction in the joints pose a problem for faithful tracking of the Robotic manipulator. Due to this, any model dependent control technique becomes very tedious. Therefore, a robust control technique such as sliding mode control will be useful.

The appearance of the sliding mode approach occurred in the Soviet Union in the Sixties with the discovery of the discontinuous control and its effect on the system dynamics. This approach is classified in the monitoring with Variable System Structure (VSS). The sliding mode is strongly requested seen its facility of establishment, its robustness against the disturbances and models uncertainties. The principle of the sliding mode control is to force the system to converge towards a selected surface and then to remain there and to slide on in spite of uncertainties and disturbances [15]. The surface is defined by a set of relations between the system variables state. The synthesis of a control law by sliding mode includes two phases:

- ✓ The sliding surface is defined according to the control objectives and to the wished performances in closed loop,
- ✓ The synthesis of the discontinuous control is carried out in order to force the system state trajectories to reach the sliding surface, and then, to evolve in spite of uncertainties and of parametric variations

The sliding mode control has largely proved its effectiveness through the reported theoretical studies. Its principal scopes of application are Robotics and the electrical motor.

Sliding mode control shows high accuracy and robustness with respect to various internal and external disturbances, it also has its own drawback: the chattering effect, which is a dangerous high-frequency vibration of the controlled system. Such an effect was considered as an inherent feature of sliding mode which is the result of immediate powerful reaction to a smallest deviation from the chosen constraint [16]. Some methods were proposed to tackle chattering. One of the methods is to change the dynamics in small vicinity of the discontinuity surface in order to avoid real discontinuity and at the same time to preserve the main properties of the whole system.

In particular, high-gain control input with saturation approximates the sign-function and diminishes the chattering, while on-line estimation of the equivalent control is used to reduce the discontinuous-control component. However, by using saturation function and equivalent control the accuracy and robustness of the sliding mode were partially lost. On the contrary, higher order sliding modes (HOSM) generalize the basic sliding mode idea acting on the higher order time derivatives of the sliding surfaces instead of influencing the first derivative like it happens in standard sliding modes. Keeping the main advantages of the original approach, at the same time they totally remove the chattering effect and provide for even higher accuracy in realization [17].

In this chapter the basic concepts of standard, second order and HOSM controllers will be reviewed and the derivation of third order sliding mode control for 3-DOF Delta Robot will be designed.

3.1. Sliding Order

The sliding order of Sliding mode characterizes the dynamic smoothness degree in the vicinity of the sliding mode. If the task is to provide for keeping a constraint given by equality of a smooth function to zero, the sliding order is a number of continuous total derivatives of (including the zero one) in the vicinity of the sliding mode. Hence, the rth order sliding mode is determined by the equalities forming an r-dimensional condition on the state of the dynamic system [16].

$$s = \dot{s} = \cdots = s^{r-1} = 0 \tag{3.1}$$

Standard sliding mode is called 1-sliding mode, in 1-sliding $\dot{s}$ model is discontinuous and in r^{th} order sliding mode s^r is discontinuous. In the subsequent sections second order and high order sliding modes will be discussed.

3.2. Second Order Sliding Modes

Following equation 3.1 second order sliding mode is obtained when:

$$s = \dot{s} = 0 \tag{3.2}$$

The control goal for a 2-sliding mode controller is that of steering s and $\dot{s}$ to zero in finite time by means of a time-dependent control $u(t)$. In order to state a rigorous control problem, the following conditions are assumed [18]:

1. Control values belong to the set $U = \{u: |u| \leq U_m$,where U_m >1 is a real constant; furthermore the solution of the system is well defined for all t , provided $U(t)$ is continuous U and $\forall t \;\; U(t) \in U$.
2. There exists $U_1 \in (0,1)$ such that for any continuous function $u(t)$ with $|u(t)| > U_1$, there is t_1, such that $s(t)u(t) > 0$ for each $t > t_1$.Hence, the control $u(t) = -sign(s(t_0))$, where t_0 is the initial time, ensures hitting the manifold $s = 0$ in finite time.
3. Let $\dot{s}(t,x,u)$ be the total time derivative of the sliding variable $s(t,x)$.There are positive constants $s_0, u_0 < 1, \Gamma_M, \Gamma_m$ such that if $|s(t,x)| < s_0$ then $0 < \Gamma_m \leq \frac{\partial}{\partial t}\dot{s}(t,x,u) \leq \Gamma_M$ for all $u \in U, x \in X$ and the inequality $|u(t)| >, u_0$ entails $\dot{s}u < 0$.
4. There is a positive constant ϕ such that within the region $|S| < s_0$ the following inequality holds

$$\forall t, u \in U, x \in X, \left|\frac{\partial}{\partial t}\dot{s}(t,x,u)\frac{\partial}{\partial x}\dot{s}(t,x,u)f(t,x,u)\right| < \phi$$

With the above assumptions, depending on the relative degree of the system, different cases can be considered

a. relative degree $r = 1 \; i.e \; \frac{\partial}{\partial t}\dot{s} \neq 0$

b. relative degree $r > 2 \;\; i.e \; \frac{\partial}{\partial t} s^r \neq 0$

The following are the most common type of second order sliding mode controllers:

3.2.1. Twisting Algorithm

Let the relative degree be one. Consider local coordinates $y_1 = s \; and \; y_2 = \dot{s}$, then after a proper initialization phase, the second order sliding mode control problem is equivalent to the finite time stabilization problem for the uncertain second-order system with $|\varphi| \leq \varphi, 0 \leq \Gamma_m \leq \gamma \leq \Gamma_M, \varphi > 0$.

$$\dot{y}_1 = y_2 \tag{3.3}$$

$$\dot{y}_2 = \varphi(t,x) + \gamma(t,x)\dot{u} \tag{3.4}$$

With $y_2$immeasurable but with a possibly known sign, $\varphi(t,x)$ and $\gamma(t,x)$ uncertain functions, this algorithm features twisting around the origin of the 2-sliding plane $y_2 O y_2$. The trajectories perform an infinite number of rotations while converging in finite time to the origin. The vibration magnitudes along the axes as well as the rotation times decrease n geometric progression. The control derivative value commutes at each axis crossing, which requires availability of the sign of the sliding-variable time-derivative y_2 [18].

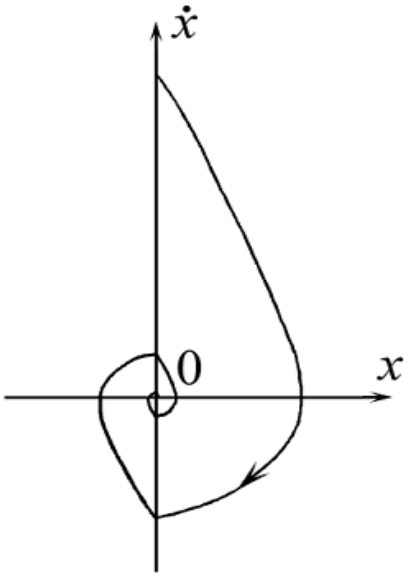


Fig. 3.1. Twisting controller phase portrait

The control algorithm is defined by the following control law, in which the condition on $|U|$ provides for $|U| \leq 1$:

$$\dot{u}(t) = \begin{cases} -u, & if\ |u| > 1 \\ -V_m SIGN(y_1),\ if\ y_1 y_2 \leq 0, |u| < 1 \\ -V_m SIGN(y_1),\ if\ y_1 y_2 \geq 0, |u| < 1 \end{cases} \quad (3.5)$$

The corresponding sufficient conditions for the finite time convergence to the sliding manifold are

$$V_M > V_m \quad (3.6)$$

$$V_M > \frac{4\Gamma_M}{S_0} \quad (3.7)$$

$$V_M > \frac{\phi}{\Gamma_m} \quad (3.8)$$

$$\Gamma_m V_m - \phi > \Gamma_M V_M + \phi \quad (3.9)$$

A similar controller can be used in order to control system (3.3-3.4) when the relative degree is 2.

$$u(t) = \begin{cases} -V_m SIGN(y_1),\ if\ y_1 y_2 \leq 0 \\ -V_M SIGN(y_1),\ if\ y_1 y_2 > 0 \end{cases} \quad (3.10)$$

In practice when $y_2$is immeasurable, its sign can be estimated by the sign of the first difference of the available sliding variable y_1 in a time interval τ , i.e., $sign(y_2(t))$ is estimated by $sign(y_1(t) - y_2(t - \tau))$.

3.2.2. Super-Twisting Algorithm

This algorithm has been developed to control systems with relative degree one in order to avoid chattering in VSC. Also in this case the trajectories on the 2-sliding plane are characterized by twisting around the origin (Figure 3.2), but the continuous control law is constituted by two terms. The first is defined by means of its discontinuous time derivative, while the other is a continuous function of the available sliding variable.

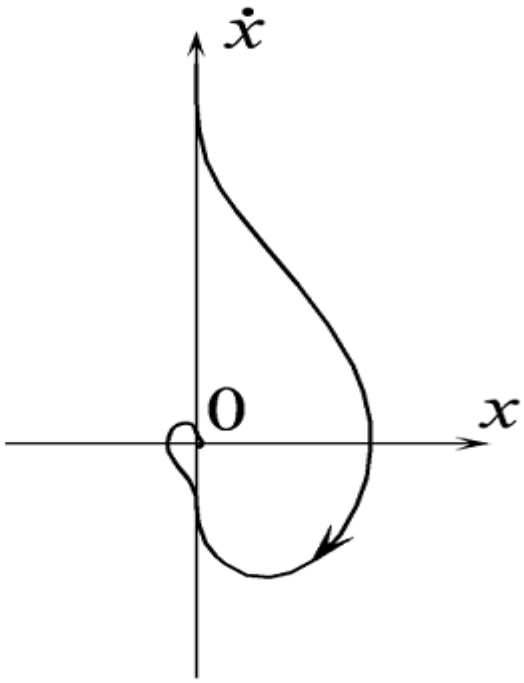


Figure 3.2 Super-twisting controller phase portrait

The control algorithm is defined by the following control law [18]:

$$u(t) = u_1(t) + u_2(t) \tag{3.11}$$

$$\dot{u_1}(t) = \begin{cases} -u & if\ |u| > 1 \\ -Wsign(y_1) if\ |u| \leq 1 \end{cases} \tag{3.12}$$

$$u_2(t) = \begin{cases} -\lambda |s_0|^p sign(y_1) & if |y_1| > s_0 \\ \lambda |y_1|^p sign(y_1) & if |y_1| \leq s_0 \end{cases} \tag{3.13}$$

And the corresponding sufficient conditions for the finite time convergence to the sliding manifold are

$$W > \frac{\phi}{\Gamma_m} \tag{3.14}$$

$$\lambda^2 \geq \frac{4\phi\Gamma_M(W+\phi)}{{\Gamma_m}^2\Gamma_M(W-\phi)} \tag{3.15}$$

$$0 < p < 0.5 \tag{3.16}$$

The super-twisting algorithm does not need any information on the time derivative of the sliding variable. If $p = 1$ is used in the control law (3.11) it will give us an exponentially stable 2-sliding mode. The choice $p = 0.5$ ensures that the maximal possible for 2-sliding realization real-sliding order 2 is achieved. Being robust, that controller is successfully used for real-time robust exact differentiation [18].

3.2.3. Sub-Optimal Algorithm

2-sliding controller was developed as a sub-optimal feedback implementation of a classical time-optimal control for a double integrator. Let the relative degree be 2. The auxiliary system is

$$\dot{y}_1 = y_2 \tag{3.17}$$

$$\dot{y}_2 = \varphi(t,x) + \gamma(t,x)\dot{u} \tag{3.18}$$

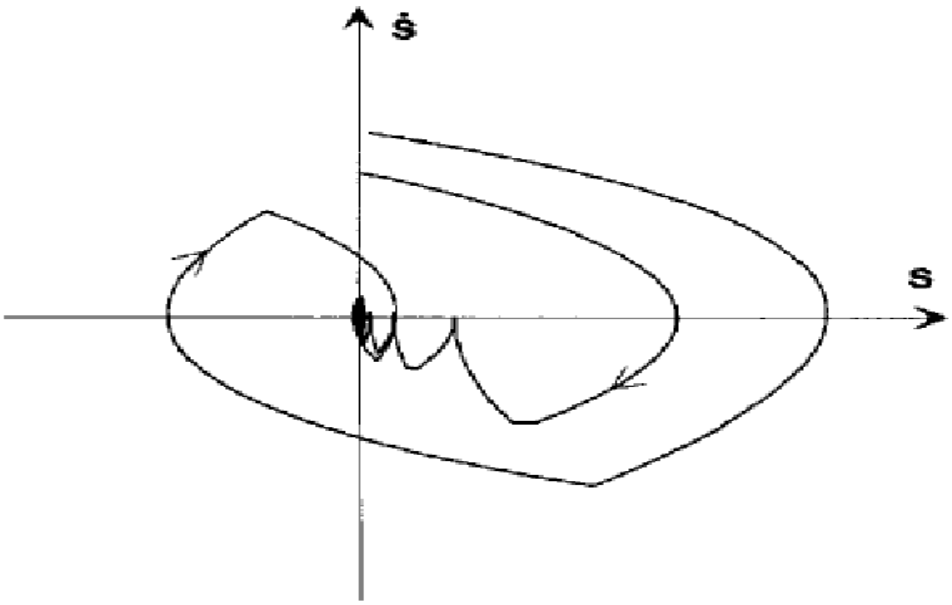


Figure 3.3.Sub-optimal controller phase trajectories

The trajectories on the $y_1 O y_2$ plane are confined within limit parabolic arcs which include the origin, so that both twisting and leaping (when $y_1 and\ y_2$ do not change sign) behaviors are possible.

Also here the coordinates of the trajectory intersections with axis y_1 decrease in geometric progression. After an initialization phase the algorithm is defined by the following control law:

$$u(t) = -\alpha(t)V_m sign(y_1(t) - \frac{1}{2}y_{1M}) \tag{3.19}$$

$$\alpha(t) = \begin{cases} \alpha^* & if \ \left[y_1(t) - \frac{1}{2}y_{1M}\right]\left[y_{1M} - \frac{1}{2}y_1(t)\right] > 0 \\ 1, \ if \ \left[y_1(t) - \frac{1}{2}y_{1M}\right]\left[y_{1M} - \frac{1}{2}y_1(t)\right] \leq 0 \end{cases} \tag{3.20}$$

Where y_{1M} is the latter singular value of the function $y_1(t)$, i.e. the latter value corresponding to the zero value of $y_2 = \dot{y}_1$. The corresponding sufficient conditions for the finite-time convergence to the sliding manifold are as follows:

$$\alpha^* \ \mathcal{E}\,(0,1] \cap (0, \frac{3\Gamma_m}{\Gamma_M}) \tag{3.21}$$

$$V_M > \max\left(\frac{\phi}{\alpha^* \ \Gamma_m}, \frac{4\phi}{3\Gamma_m - \alpha^* \ \Gamma_M}\right) \tag{3.22}$$

The sub-optimal algorithm requires some technique in order to detect the singular values of the available sliding variable $y_1 = S$. In the most practical case y_{1M} can be estimated by checking the sign of the quantity $D(t) = [y_1(t-\tau) - D(t) = \ y_1(t)]\ y_1(t)$ in which $\frac{\tau}{2}$ is the estimation delay. In that case the control amplitude V_M must belong to an interval instead of a half-line:

$$V_M \mathcal{E}\,(\max\left(\frac{\phi}{\alpha^* \ \Gamma_m}, V_{M1}(\tau, y_{1M})\right), V_{M2}(\tau, y_{1M}) \tag{3.23}$$

where

$$V_{M1} < V_{M2} \tag{3.24}$$

are the solutions of the second-order algebraic equation.

$$\left[\frac{(3\Gamma_m - \alpha^* \ \Gamma_M)V_{M1}}{\phi} - 4\right]\frac{y_{1M}}{\phi\delta^2} - \frac{V_{Mi}}{8\phi}\left[\Gamma_M + \Gamma_M(2 - \alpha^*)\left(\frac{\Gamma_M V_{Mi}}{\phi} + 1\right)\right] = 0 \tag{3.25}$$

3.3. High Order Sliding Mode Controllers

There are two well-known r-sliding controller families .These are Nested Sliding Controllers and Quasi-continous Sliding Controllers .For a system with relative degree r given by

$$S^{(r)} = h(t,y) + g(t,y)u\ ,\ \ g(t,y) \neq 0 \tag{3.26}$$

and for some $K_m, K_M, C > 0$

$$0 < K_m \leq g(t,y) \leq K_M,\ \ |h(t,y| \leq C \tag{3.27}$$

The controllers are of the form

$$u = -\alpha\, \psi_{r-1,r}(s, \dot{s}, \dots, s^{(r-1)}) \tag{3.28}$$

and are defined by recursive procedures, have magnitude $\alpha > 0$, and solve the general output regulation problem. The parameters of the controllers can be chosen in advance for each relative degree r. Only the magnitude of α is to be adjusted for any fixed K_m, K_M, C most conveniently by computer simulation, thus avoiding complicated and redundantly large estimastions Obviously,α has to be negative with estimastions.

$$\frac{\partial(S^r)}{\partial u} < 0$$

The following procedure defines the nested r-sliding controllers:

$$N_{i,r} = (\ |s|^{\frac{q}{r}} + \ |\dot{s}|^{\frac{q}{(r-1)}} + \cdots + |s^{(i-1)}|^{\frac{q}{(r-i+1)}}\ )^{\frac{1}{q}} \tag{3.29}$$

$$\psi_{0,r} = \ sign(s) \quad \psi_{i,r} = \ sign\left(s^i + \beta_i N_{i,r} \psi_{i-1,r}\right) \tag{3.30}$$

$\beta_1 \dots \dots \beta_r > 0$ are the controller parameters, which define the convergence rate. The number of choices of β_i is infinite. [19] proposed the following nested sliding mode

controllers with β_i tested for $r \leq 4$

1. $U = -\alpha sign(s)$
2. $U = -\alpha sign(\dot{s} + |s|^{\frac{1}{2}}\, sign(s))$

3. $U = -\alpha sign(\ddot{s} + 2(|\dot{s}|^3 + |s|^2)^{\frac{1}{6}}\ sign(\dot{s} + |s|^{\frac{2}{3}} sign(s)))$
4. $U = -\alpha sign(\dddot{s} + 3(\ddot{s}^6 + \dot{s}^4 + |s|^3)^{\frac{1}{12}}\ sign(\ddot{s} + (\dot{s}^4 + |s|^3)^{\frac{1}{6}} sign$
$(\dot{s} + 0.5|s|^{\frac{3}{4}} sign(s))))$

The idea of the controller is that a 1-sliding mode is established on the smooth parts of the discontinuity set Γ of equation 3.26. that sliding mode is described by the differential equation $\psi_{i-1,r} = 0$. But the primary sliding mode disappears at the moment when the secondary one is to appear. The resulting movement takes place in some vicinity of the subset of Γ satisfying $\psi_{i-2,r} = 0$, transfers in finite time into some vicinity of $\psi_{r-3,r} = 0$ the and so on. While the trajectory approaches the r-sliding set, set Γ retracts to the origin in the coordinates $s, \dot{s}, \dots, s^{(r-1)}$.

Any r-sliding homogeneous controller can be complemented by an $r-1$th order differentiator producing an output-feedback controller. In order to preserve the demonstrated exactness, finite-time stability, and the corresponding asymptotic properties, the natural way to calculate $s, \dot{s}, \dots, s^{(r-1)}$ in real time is by means of a robust finite-time convergent exact homogeneous differentiator.

Let the input signal $f(t)$ be a function defined on $[0, \infty)$ consisting of a bounded Lebesgue-measurable noise with unknown features and of an unknown base signal $f_0(t)$ whose k[th] derivative has a known Lipschitz constant $L > 0$. The problem of finding real-time robust estimations of $\dot{f}_0(t), \ddot{f}_0(t), \dots, {f_0}^k(t)$ which are exact in the absence of measurement noises is solved by the differentiator [19].

$$\dot{Z}_0 = V_0, V_0 = -\lambda_k L^{\frac{1}{k+1}} |Z_0 - f(t)|^{\frac{k}{k+1}} sign\big(Z_0 - f(t)\big) + Z_1$$

$$\dot{Z}_1 = V_1, V_1 = -\lambda_{k-1} L^{\frac{1}{k}} |Z_1 - V_0|^{\frac{k-1}{k}} sign(Z_1 - V_0) + Z_2$$

$$\vdots$$

$$\dot{Z}_{k-1} = V_{k-1}, V_{k-1} = -\lambda_1 L^{\frac{1}{2}} |Z_{k-1} - V_{k-2}|^{\frac{1}{2}} sign(Z_{k-1} - V_{K-2}) + Z_k$$

$$\dot{Z}_k = -\lambda_0 Lsign(Z_k - V_{k-1})$$

If the parameters $\lambda_0, \lambda_1 \ldots\ldots \lambda_k > 0$ are properly chosen, the following equalities are true in the absence of input noise, after a finite-time transient process:

$$Z_0 = f_0(t), \ldots, Z_i = V_{i-1} = {f_0}^{i-1}(t), \qquad i = 1,2,3 \ldots$$

For any given $\lambda_0 > 1$ there exists an infinite positive sequence $\{ \lambda_n \}$ such that for each natural K the parameter $\lambda_0, \lambda_1 \ldots\ldots \lambda_k$ provide finite-time convergence of the kth-order differentiator.

Apossible choice of the differentiator parameters for $K < 5$ is $\lambda_0 = 1.1, \lambda_1 = 1.5, \lambda_{32} = 3, \lambda_3 = 5, \lambda_4 = 8, \lambda_5 = 12$ another possible choice is $\lambda_0 = 1.1, \lambda_1 = 1.5, \lambda_2 = 2, \lambda_3 = 3, \lambda_4 = 5, \lambda_5 = 8$

4. Chapter Four :Simulations and Results

4.1. Multi-Body Modeling of 3-DOF Delta Robot

Robot present considerably complicated electromechanical systems with mutual interactions of Robot mechanics and drives, at design of which the mechatronic approach should be taken into consideration. The computer modeling presents such basic tool for mentioned mechatronic approach. When designing control of a Robot, we need to know necessary torque and angle of rotation of each motor, to visualize behavior of the Robot, and to obtain mathematical model of each part. Generally, this inverse kinematic task is not solvable analytically and the numerical calculation often entails difficulties. The design of a control law for the drive system is also connected with the need of transfer function derivation and with simulation of dynamical properties of the Robot mechanical system as a whole.

The physical modeling in the SimMechanics environment considerably facilitates simulation efforts of complex mechanical systems regardless of their complication by elastic and damping elements and by number of degrees of freedoms. The SimMechanics program scheme having the form of interconnected blocks shows how the physical components with geometric and kinematic relationships of the Robot are mutually interconnected. The SimMechanics program enables one to model mechanical systems by bodies and joints, to simulate their motion, to change easily the structure, to optimize system parameters, and to analyze results all within the Simulink environment. This approach does not require cumbersome deriving differential equations of the system and presents an easy and fast way to obtain the dynamic model of the system and saves time and effort.

There are several approaches for the Robot model development in the SimMechanics environment. The Robot models were developed on basis of the Robot configuration. To simplify the development task ,in this paper a special feature announced by the MathWorks in 2003 which integrates SimMechanics with The SolidWorks CAD Assemblies has been utilized. Mathworks collaboration with SolidWorks Corporation extended the engineering analysis capabilities of SimMechanics by allowing seamless integration of SolidWorks CAD Assemblies into the SimMechanics simulation and design environment. This means that the SolidWorks

models can be simulated in the Simulink environment in order to analyze forces and torques in mechanical joints, plot accelerations and displacements of each part of the system, to visualize motion of the CAD assembly, while taking into consideration masses of individual objects. This facility is enabled by installing an appropriate plug-in in SimMechanics which imports the 3D CAD model of the full system with bodies, joints, couplings, and masses from the SolidWorks program into the SimMechanics for further work with the model , basically the controller design.

Calculations were performed in the MATLAB/SimMechanics environment that enables a simple physical modeling of mechanical systems without any necessity of motion equations derivation.

A 3D-CAD model of the Robot mechanics was developed using SolidWorks program. This procedure also enables verification of the model whether it corresponds to the reality and whether it behaves according to the presumptions and requirements. By importing the 3D-CAD model into SimMechanics a basic simulation scheme is obtained. Once the SimMechanics model of the Robotic system is obtained, it enable one to perform various experiments and to obtain required outputs: position (angles), forces and torques.

4.1.1. A 3D-CAD Model for 3-DOF Delta Robot

With the help of 3D-CAD SolidWorks 2015 program, 3-DOF Delta Robot has been designed successfully. Mass and inertia of the Robot parts depends on the material type, so, for more flexibility and Robot speed, in this paper the materials are chosen to have little weight .The model values below are for a specific commercial Delta Robot, the ABB FlexPicker IRB 360-1/1600, scaled from a figure (new.abb.com/products).

	Delta Robot parts	Dimension in mm
1	Base equilateral triangle side	840
2	Platform equilateral triangle side	190
3	Upper legs length	300
4	Lower legs parallelogram length	800

Table 4.1.Delta Robot design parameters

Each part of the Delta Robot has been designed with the appropriate dimension and material type separately , then the Robot parts are assembled into a sub-assembly and assembly by applying mate or constraint properties between two parts ,which later becomes a joint in SimMechanics model which determines the of motion between two rigid parts and define the number of degree of freedom (DOF)s between parts.

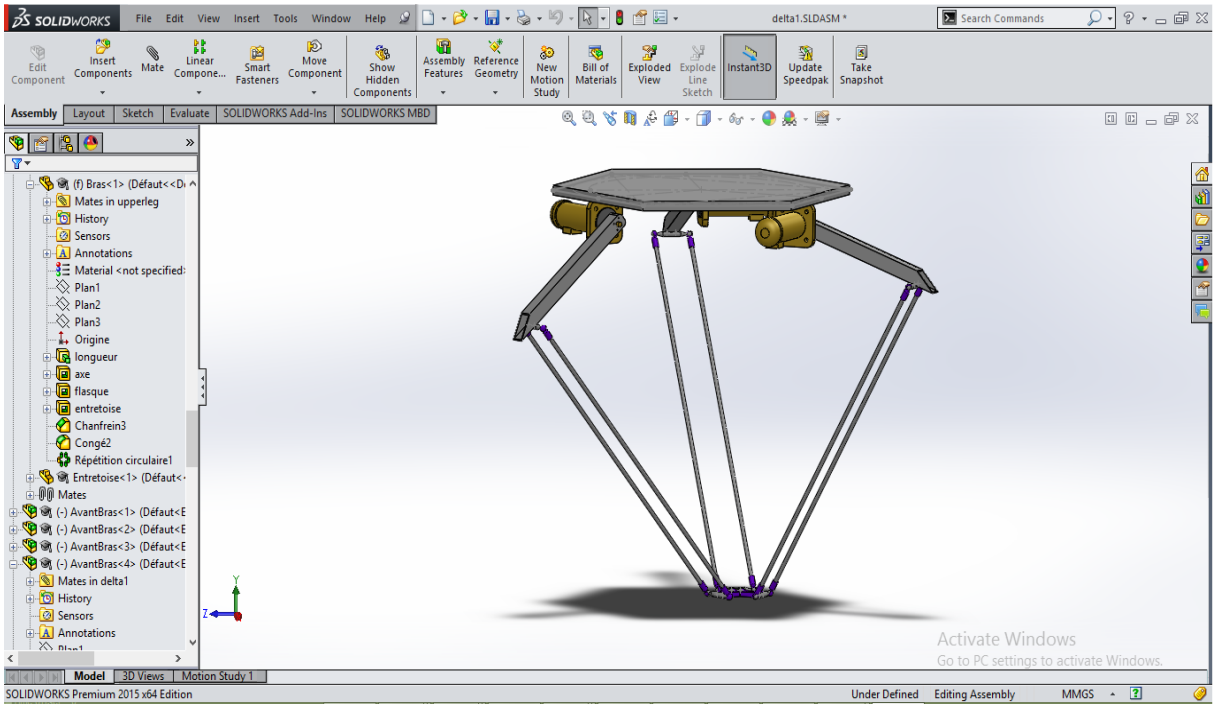


Fig .4.1. A3-D, 3-DOF Delta Robot assembly in SolidWorks

4.1.2. Delta Robot SimMechanics Model

3D-CAD tools allow us to model mechanical systems in 3-D space. Although this approach is excellent for geometric modeling, incorporating controllers into this environment is difficult. Simulink with SimMechanics software uses a block-diagram schematic approach for modeling control systems around mechanical devices. The SimMechanics Link utility bridges the gap between geometric modeling and block diagram modeling and simulation, by combining the power of Simulink and SimMechanics software with3D- CAD. With the SimMechanics Link utility, we can create a SimMechanics model from a CAD assembly, in two steps.

1. Exporting CAD Assemblies Into Physical Modeling XML

The first translation step is to use the SimMechanics Link exporter to create an intermediate Physical Modeling XML file from a CAD assembly. Using SimMechanics software, we can then import that XML file to automatically generate a SimMechanics model.

With SimMechanics Link export, we can initiate the translation of CAD assemblies into dynamical block diagram models. It exports:

- ✓ CAD assemblies into Physical Modeling XML format. The XML file captures the mass and inertia of each part in the assembly and the constraint definitions between parts.
- ✓ Graphics files to define the body geometries of the assembly parts. The graphics files capture the body geometries of the assembly parts.

SimMechanics software contains two technologies: First Generation and Second Generation. First-generation technology includes the block library and visualization utility found in SimMechanics releases prior to R2012a. Second-generation technology introduces a simpler modeling paradigm with a new block library, a powerful computational engine, an advanced visualization utility based on OpenGL computer graphics, and tighter integration with Simscape products. SimMechanics first and second-generation technologies have different sets of capabilities. Which technology to use depends on the effects we need to model. In this paper SimMechanic second-generation is used to model the exported 3D-CAD model of the Robot.

2. Importing Physical Modeling XML to Generate SimMechanics Models

The second translation step is to import the Physical Modeling XML to generate the SimMechanics model, then use that model together with the body geometry graphics files to simulate and visualize the original mechanical system.

- ✓ The XML representations of parts and constraints become bodies and joints in a SimMechanics model.
- ✓ The generated SimMechanics model uses the exported body geometry graphics files to visualize the bodies.

While important, rigid body subsystem and joint blocks are not sufficient to represent the Delta Robot model in the generated SimMechanics model, the exported model automatically

incorporates other blocks which serve an important purpose in simulation environment. These includes World Frame, Rigid Transform, Mechanism Configuration, and Solver Configuration blocks which are described below.

Block	Purpose
World Frame	Provides the ultimate reference frame in a model. All remaining frames are defined with respect to this frame. It is inertial and it defines absolute rest.
Rigid Transform	Applies a fixed spatial relationship between frames. This block defines the offset distance and angle between two frames.
Mechanism Configuration	Identifies the gravity vector in a model.
Solver Configuration	Provides essential simulation parameters required to simulate the model.

Table 4.2. SimMechanics basic blocks

SimScape's SimMechanic second-generation library allows as to apply Forces and Torques to a rigid body or to a joint and to sense motion between two arbitrary rigid body frames.

To represent forces and torques acting on a rigid body, SimMechanics provides a Forces and Torques library. One block represents an external force or torque—External Force and Torque. The other blocks represent internal forces between two rigid body frames. To represent forces and torques acting at a joint, SimMechanics provides a selection of actuation inputs in each Joint block. Each joint primitive provides a selection of actuation inputs that we can select. We can represent a torque acting on a revolute or spherical joint primitive, or a force acting on a prismatic joint primitive.

We can sense motion between two arbitrary rigid body frames. For this task, SimMechanics provides the Transform Sensor block. This block provides the broadest motion sensing capability in SimMechanics. With it, we can sense position, velocity, and acceleration, both rotational and translational, between any two frames in a model.

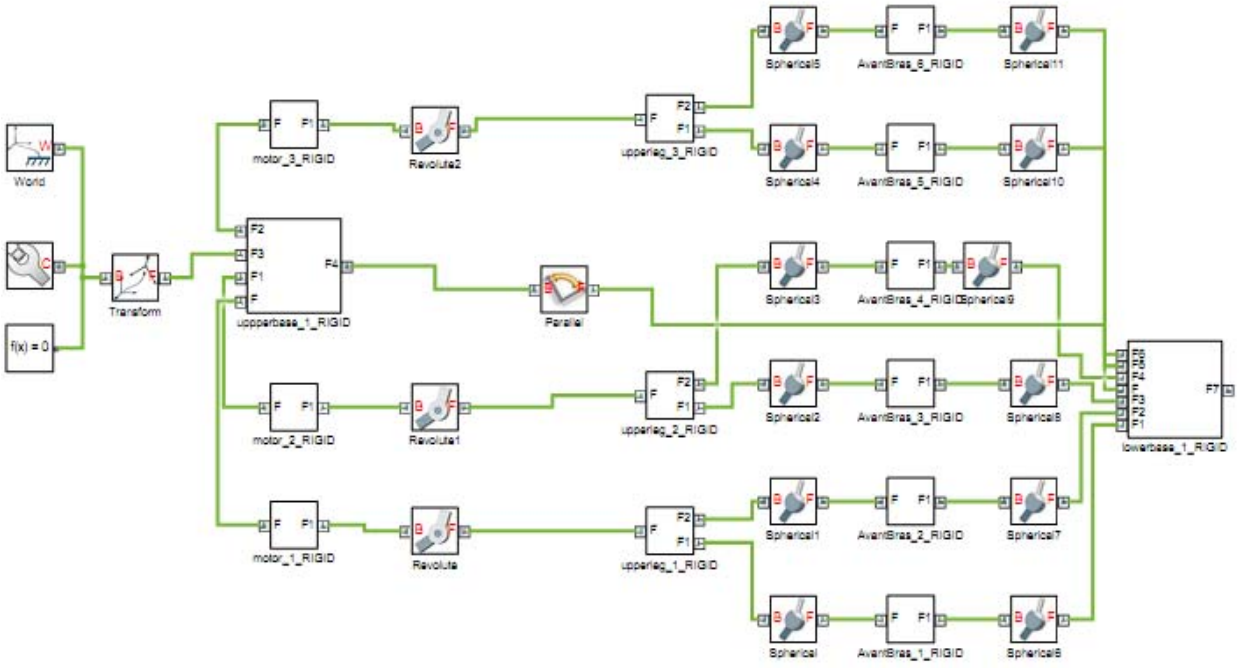
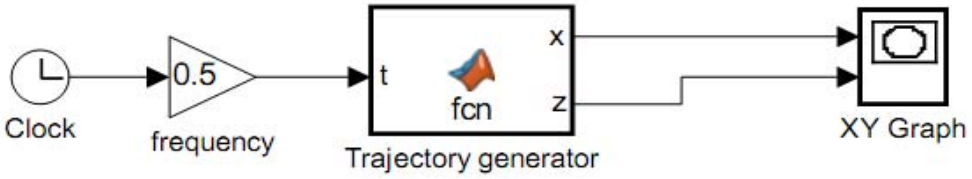


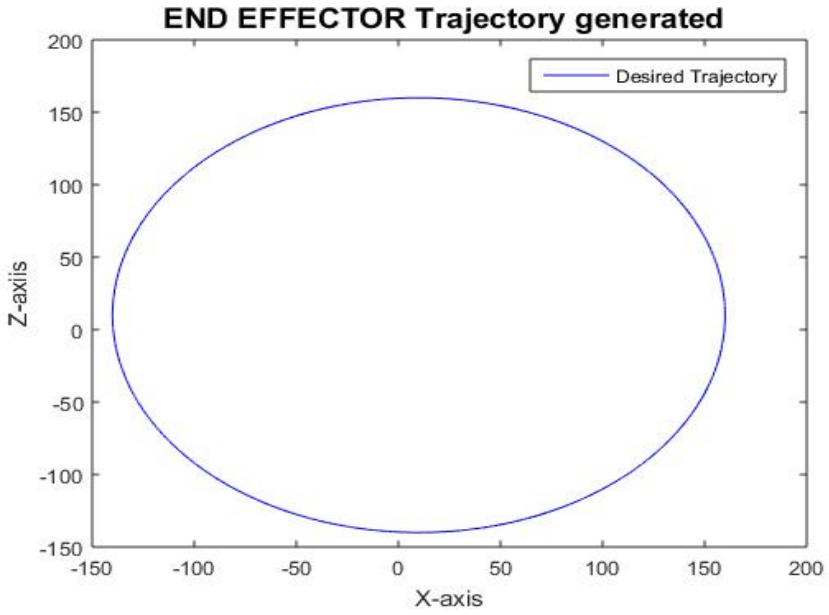
Fig .4.2.A 3D-CAD exported SimMechanic Second-generation model of 3-DOF Delta Robot

4.2. Trajectory Generation

The end-effector of the Delta Robot was simulated to follow a circular trajectory based on the implementation of the desired controller. The tracking speed utilized is defined by the angular velocity formula: $\omega = 2\pi f$, where f is the tracking frequency of the end-effector. The origin of the circle based on the Cartesian Coordinate system in millimeters is defined as (10, Y_0,10), where Y_0 is the distance of travelling in y- direction .The radius of the circular trajectory is 250 millimeters and the frequency implemented is 0.5 Hertz. It should be noted that the trajectory defined in this thesis never impedes or approaches any singular point.



(a)



(b)

Fig .4.3. (a)desired trajectory generator (b) generated trajectory

4.3. Controller Design

The Robotic manipulator is a complex system with uncertainties and parameter variations. Moreover, presence of friction in the joints pose a problem for faithful tracking of the Robotic manipulator. Due to this, any model dependent control technique becomes very tedious. Therefore, a robust control technique such as sliding mode control and third order sliding mode control in particular will be useful as described in section three of this paper. In third order sliding mode control the controller parameters are not dependent on exact values of system parameters. in this paper 3-DOF Delta Robot with uncertainties and friction in joints is

considered for controller design using third order sliding mode. The joints in the upper arm are considered active /actuated revolute joints and the remaining joints are passive or non actuated joints.

4.3.1. Dynamic Model of Delta Robot

As described in section 2.3, the dynamic model of any n link Robotic manipulator can be described in joint space:

$$T = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) + F(\dot{\theta})$$

Here, $\theta = [\theta_1, \theta_2, \theta_3]^T$ is the vector of joint angle variables, $\omega_1 = [\omega_1, \omega_2, \omega_3]^T$ is the vector of joint velocities with $\theta_i \varepsilon [0,2pi)$ and $T = [T_1, T_2, \dots, T_n]$ is the controlling torque vector. $M(\theta)$ is the inertia matrix, $V(\theta, \dot{\theta})$ is vector of Centripetal and Coriolis terms, $G(\theta)$ is vector of gravity terms and $F(\dot{\theta})$ is vector of joint friction. It is assumed that the mass of each link is concentrated at the mid-point of the respective link. Thus, defining

$$\omega = \dot{\theta},$$

we get,

$$\dot{\omega} = -M^{-1}(\theta)V(\theta, \omega) + M^{-1}(\theta)[T - G(\theta) - F(\dot{\theta})]$$

The system is uncertain and has friction, therefore

$$-M^{-1}(\theta)V(\theta, \omega)\ \varepsilon\ [-F, F]$$

$$M^{-1}(\theta)[T - G(\theta) - F(\dot{\theta})\ \varepsilon\ [G_1, G_2]$$

4.3.2. Sliding Mode Control

Sliding mode control changes the dynamics of the system, motion of the system trajectory remains along a chosen surface of the state space. There are two phases of sliding mode control: reaching phase and sliding phase. Prior to reaching sliding surface s, the system gets affected by disturbances, friction and uncertainties in the parameters. In sliding phase the system dynamics is governed by sliding surface parameters and not original system parameters and the system

behavior becomes invariant to any disturbance or change in parameters. Sliding mode control is therefore designed to achieve sliding mode in finite time. In third order sliding mode, the control is designed in a way to make $s, \dot{s}, and\ \ddot{s}$ of the sliding function also reach zero in finite time.

1. Choice of sliding surface for 3-DOF Delta Robot

For efficient design of sliding mode control, the proper choice of sliding surface is very important. The control task is that each upper joint of Delta Robot should track the desired given trajectory obtained from the inverse kinematics computation to it, in the presence of friction in joints and uncertainties in the system. The error between actual and desired joint angles has to be brought to zero in finite time and hence error is considered as sliding function. As sliding function approaches zero in finite time, the desired joint position is also achieved. Therefore, the sliding surface is the point of intersection of zero angular error in the three joint angles, i.e.,

$$S = \begin{pmatrix} \theta_{actual1} - \theta_{desired1} \\ \theta_{actual2} - \theta_{desired2} \\ \theta_{actual3} - \theta_{desired3} \end{pmatrix}$$

Here, $\theta_{actual1}, \theta_{actual2}$ and $\theta_{actual3}$, are the actual joint angles and $\theta_{desired1}, \theta_{desired2}$ and $\theta_{desired3}$ are the desired upper joint angles for the Delta Robot.

2. Third order sliding mode

After the sliding surface is chosen, a controller is designed to reach the sliding surface in finite time and then remain on sliding surface. For the present system, the sliding function is of relative degree 2 and hence any order of sliding mode greater than or equal to 2 can be achieved. The second order sliding function removes chattering in states but control remains discontinuous. Using third order sliding mode control, the control torque also becomes smooth. In third order SMC, $s, \dot{s}\ and\ \ddot{s}$ reach zero in finite time. Here, a control is needed so as to shape the third order derivative of s. Here, the rate of change of torque is manipulated in order to achieve third order SMC. The control expression is thus given as described in section three as;

$$\dot{T}_{3rdsmc} = -\alpha sign(\ddot{s} + 2(|\dot{s}|^3 + |s|^2)^{\frac{1}{6}}\ sign(\dot{s} + |s|^{\frac{2}{3}} sign(s)))$$

In this paper the control torque output from the designed 3-SMC controller with the value of α tunned to $\alpha = [0.5 \;\; 0.5 \;\; 0.5]^t$ for each actuated arm

The complicated discontinuity structure of this controller can be smoothed by replacing the discontinuities under the sign-function with a finite-slope approximation called a regularized controller. As a result, the transient process becomes smoother [12].

The function $\dot{s} + |s|^{\frac{2}{3}} sign(s)$ can be replaced with

$$\max\left[-1, \min\left(1, |S|^{-\frac{2}{3}}\left(\frac{\dot{s} + |s|^{\frac{2}{3}} sign(s)}{\mathcal{E}}\right)\right)\right]$$

for some sufficiently small $\mathcal{E}$, $\mathcal{E}$ is selected to be 0.001.Since control signal , rate of change of torque, equation involves finding $\dot{s}$ and $\ddot{s}$ which are not readily available. Calculating them using an ordinary differentiator will introduce noise; these terms can be found by using an exact robust differentiator of second order or higher. In this paper $f_0(t) = \theta_{\text{actual1}} - \theta_{\text{desired1}}$ is the measured joint angle error ,and estimates of $\dot{s}$ and $\ddot{s}$ can be found by for $k = 2$

$$\lambda_0 = 1.1, \; \lambda_1 = 1.5 \,, \lambda_{32} = 3 \,, L = 30$$

$$\dot{Z}_0 = V_0 \,, V_0 = -9.322 \, |Z_0 - s|^{\frac{2}{3}} \, sign(Z_0 - s) + Z_1$$

$$\dot{Z}_1 = V_1 \,, V_1 = -8.216 \, |Z_1 - V_0|^{\frac{1}{2}} \, sign(Z_1 - V_0) + Z_2$$

$$\dot{Z}_2 = -33 sign(Z_2 - V_1)$$

where $\dot{s} = Z_1$ and $\dot{s} = Z_2$

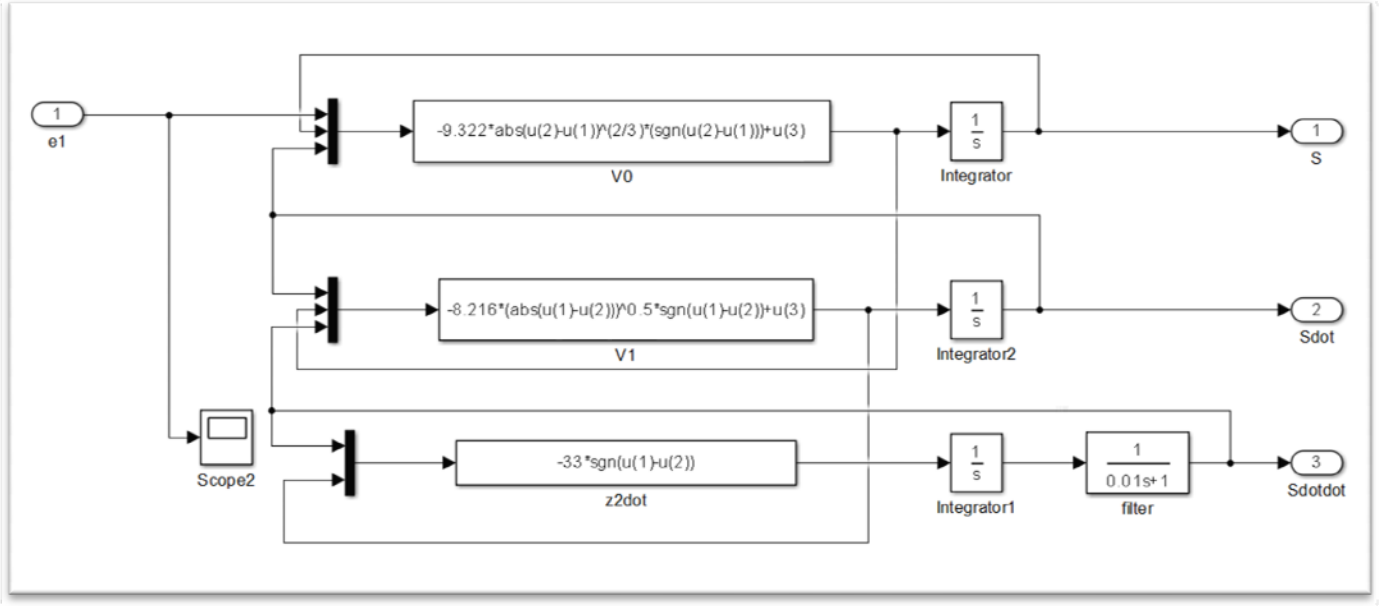


Fig .4.4. Second order robust exact differentiator Simulink model

4.4. Simulation

The Robot has been designed in Solidwork's 3D-CAD modeling and this model is exported to Simulink as a SimMechanics second-generation model and its kinematic constraints, mass , inertia and graphic properties of the Robot are generated automatically. The exported model is simulated in an environment of gravitation field to check the constraints has been maintained.

The dialog of a Revolute joint in SimMechanics second-generation primitive now has two fields in the Actuation section: Torque and Motion. The Torque field gives us the options; none, provided by input and automatically computed. The Motion field also gives us two options: Provided by Input and Automatically Computed. This makes a total of six possible configurations for each joint primitives. From the possible six configurations . Inverse Dynamics and Forward Dynamics are mainly used for Robot's kinematics and dynamics analysis.

In Inverse Dynamics actuation of a revolute joint , motion is specified by input and torque is automatically computed. If we know how a joint should move and want to know the amount of torque or force necessary to make that happen, this mode is useful. SimMechanics second-generation has eased a complex computation of forces or torques required to accomplish a

desired path or motion to be achieved which mainly involves singularity in computation. Model based controllers usually need this torque to be available in order to perform well.

In this paper the desired trajectory is converted into joint angles using inverse kinematic calculation as described in section two and is applied as an input to the Robot, Inverse Dynamics mode, then the required torques are sensed from each of the revolute joints. The other most important use of this mode is, it enables as to check whether inverse kinematic calculations produce the required trajectory on the end-effector or not.

In forward Dynamics, actuation of a revolute joint with a force or torque is specified for the joint primitive and the engine computes the resulting motion. This mode is used in the rest of the analysis in this paper. MATLAB/ Simulink has been utilized for inverse and forward kinematics , dynamics, third order sliding mode controller modeling, design and simulation.

For each actuated leg of the Delta Robot it has been designed a 3-SMC controller of nested type and tuned to track the joint angles. After tuning the 3-SMC controller the controller gain $\alpha = [0.5 \;\; 0.5 \;\; 0.5]^T$ is found, the 3-DOF Parallel delta Robot is made to track a circular trajectory constructed in section 4.2 with Simulink's ODE-45 solver, which is the recommended solver for mechanical systems.

While simulating, MATLAB shows the result of control like joint angle error, reference tracking and sliding surfaces & Mechanics explorer shows the motion of the Robot in a 3D space.

The whole simulation scheme is shown in the figure below and the simulation is carried in two separate parts. The first part is a simulation with no external torque, disturbance and mass uncertainty included . The second part is a simulation which includes the effects of external torque, disturbance and mass uncertainty included throughout the simulation course.

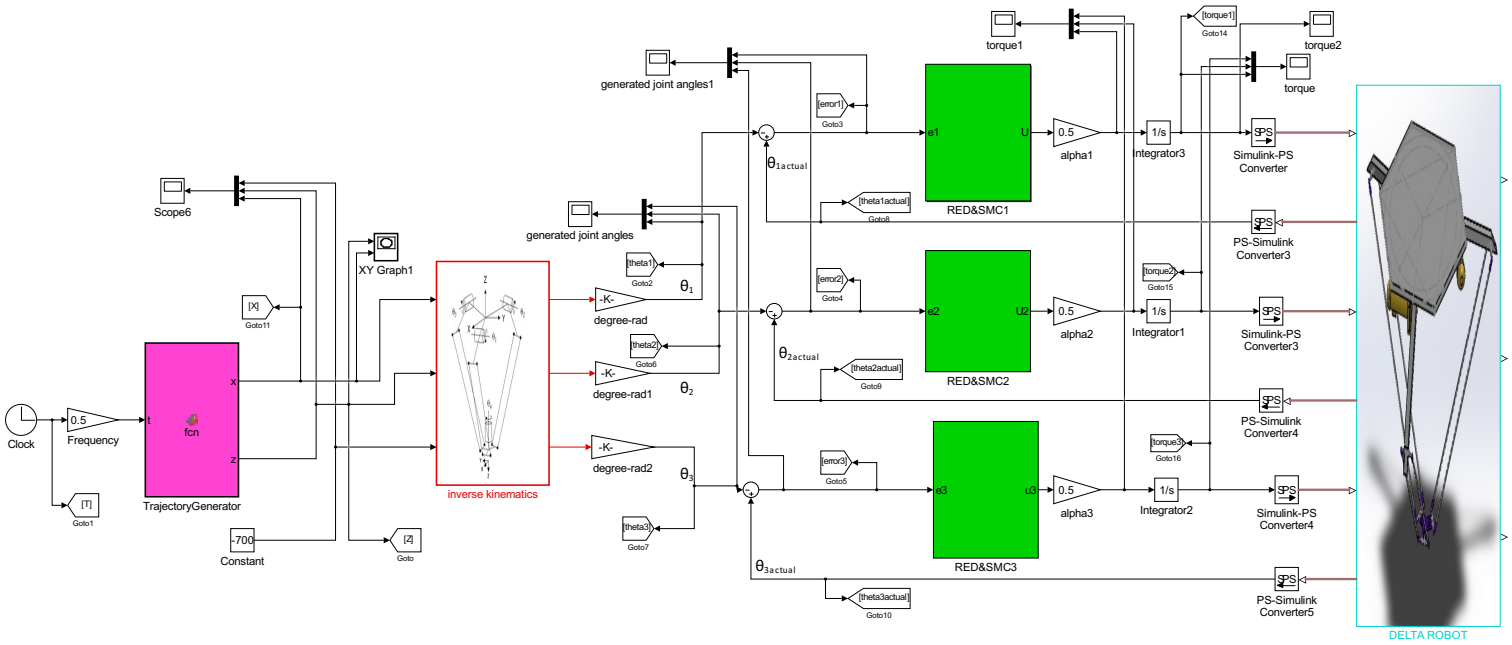


Fig 4.5. Circular Path Trajectory tracking for Delta Robot.

4.4.1. Results

1. Results for no external torque ,disturbance and mass uncertainty included

End effector Trajectory tracking

The next set of figures portrays the trajectory tracking of the end-effector along with its respective error between its desired and actual position. In order to determine the actual location of the end effector, inverse kinematic equations were employed to solve for θ_1, θ_2 and θ_3 respectively. The results from the joint angles of θ_1, θ_2 and θ_3 play a crucial role in the determination of the actual location of the end effector. The plots of the end-effector error in the X-axis and Z-axis are shown in order to easily identify the severity of the absolute accuracy.

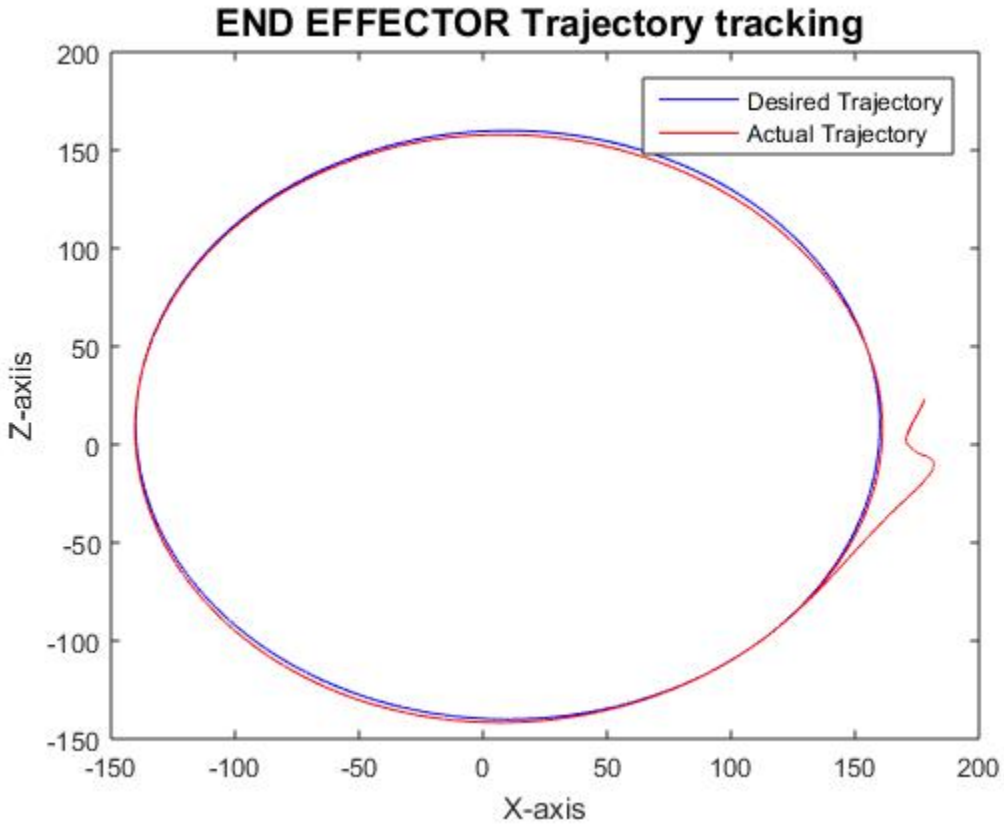
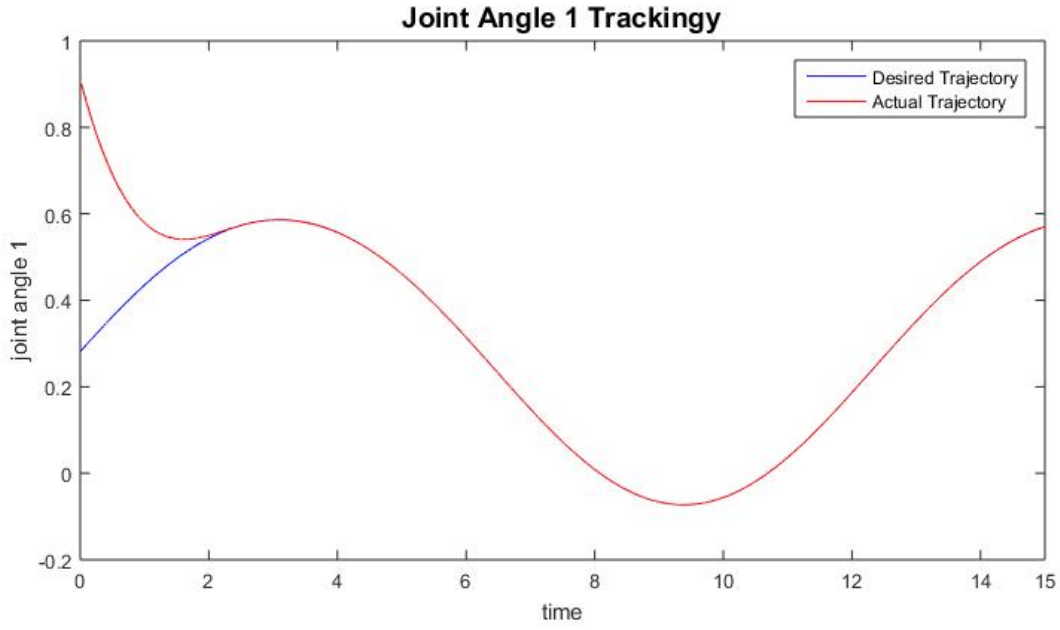


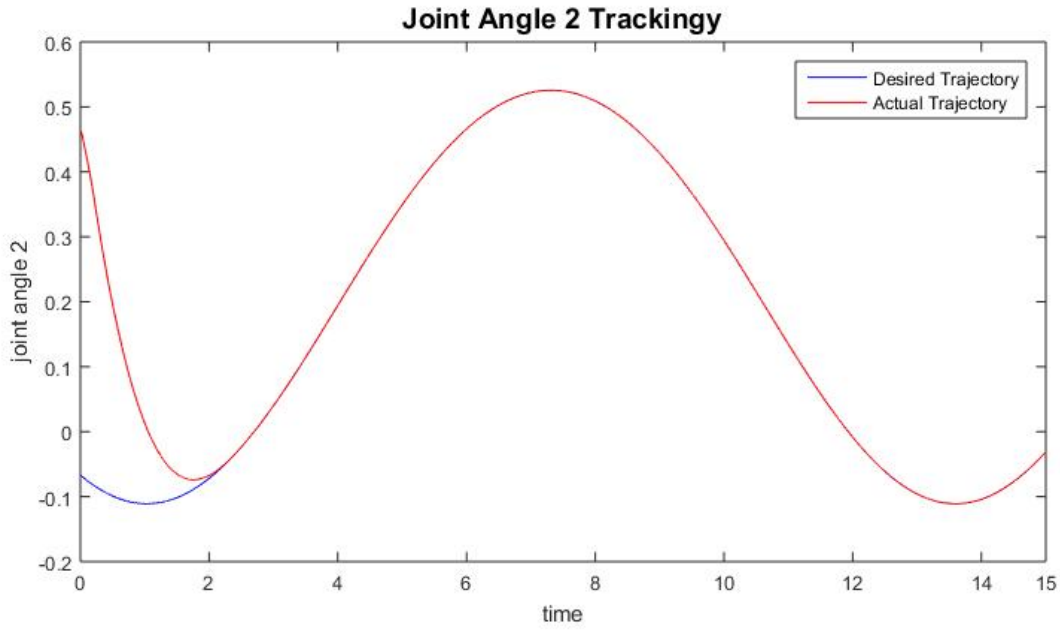
Fig 4.6. End effector trajectory tracking

The above figure shows the trajectory tracking of the Robot in the XZ-plane. The rms error after reaching phase of the controller is 1mm in X and 2mm in Z -axis.

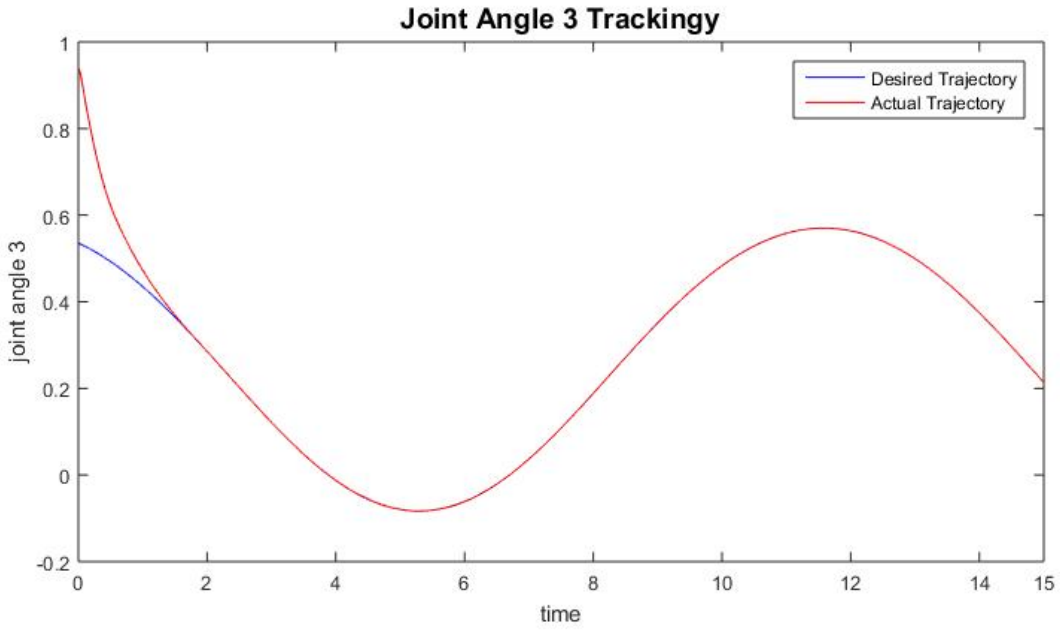
The tracking of the joint angles and the joint angle errors are shown in Figures 4.6 (a) ,(b), (c) and (d). As the figures shows the controller made the uper leg to follow the desired trajectory in less than 3 seconds and the joint angle errors between the desired and actual trajectory dimminshes to zero in finite time.



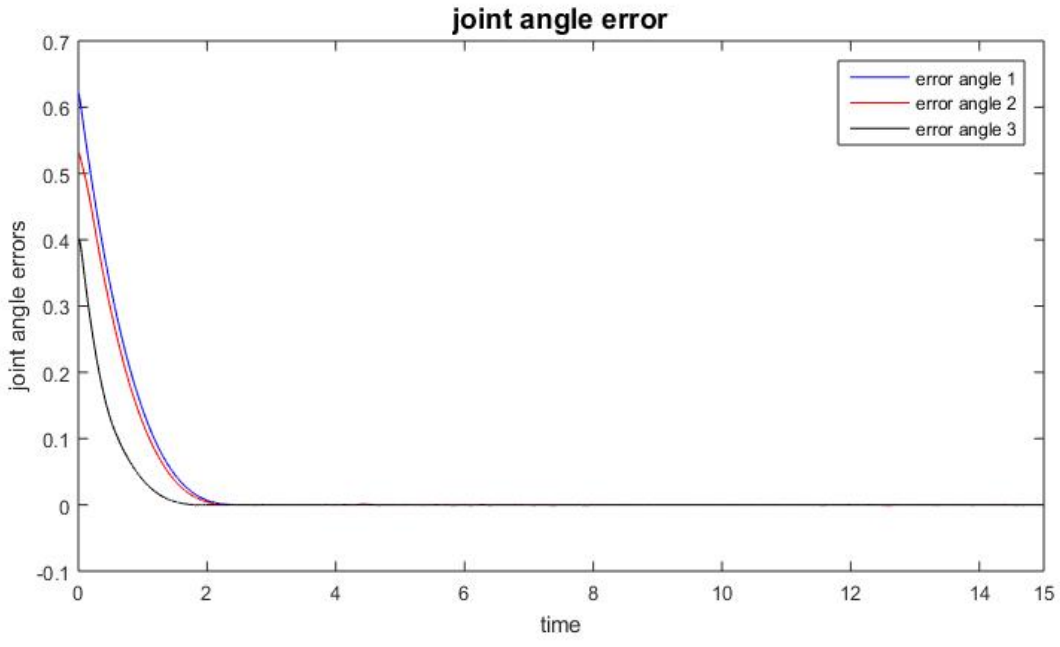
(a)



(b)



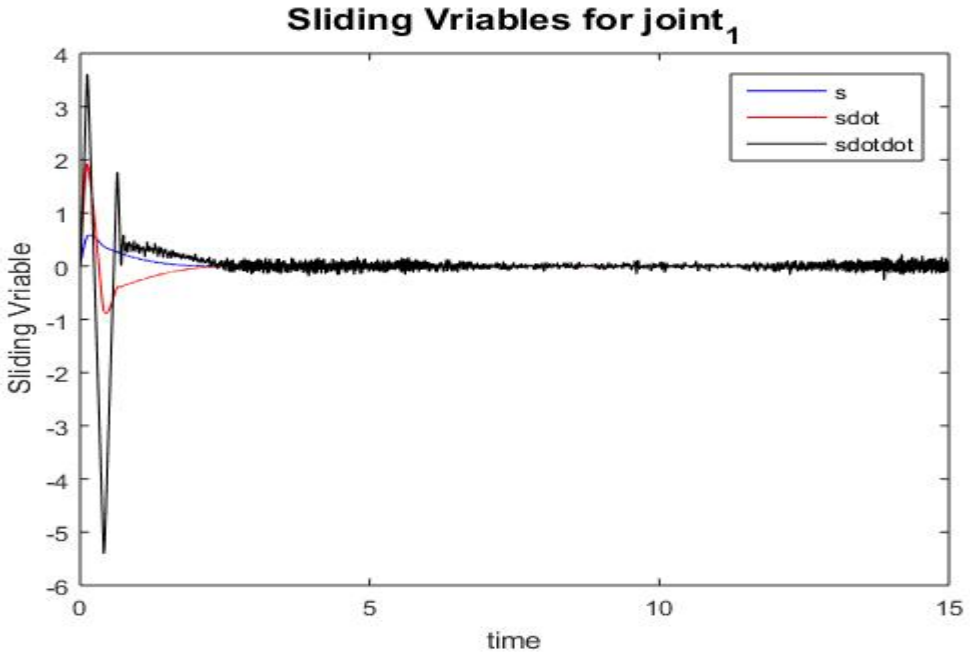
(c)



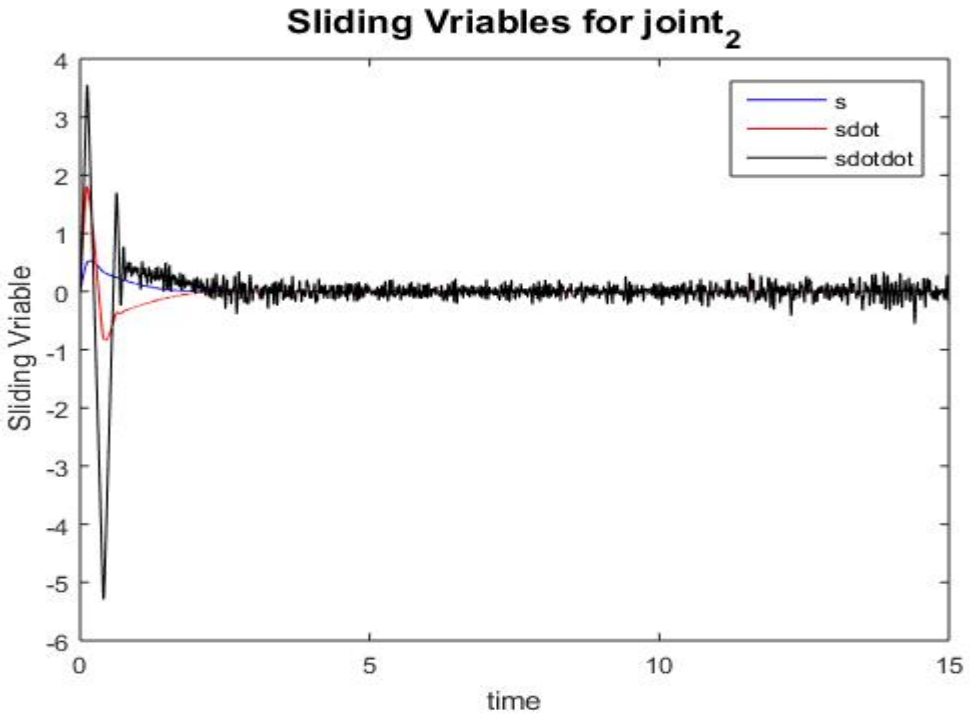
(d)

Fig 4.7. (a)-(c) joint angles tracking (d) joint angle errors

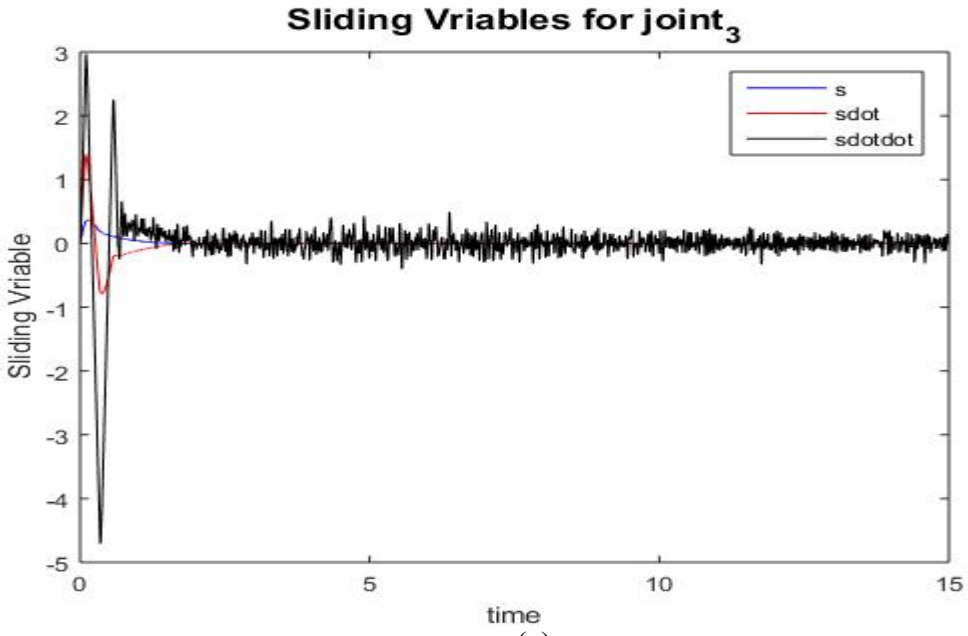
Fig 4.5. (a)-(c) shows the sliding variables $s, \dot{s}, and\ \ddot{s}$ for each joint angle. The sliding variables settles to zero before 3 seconds, as needed from the design of 3-SMC controller.



(a)



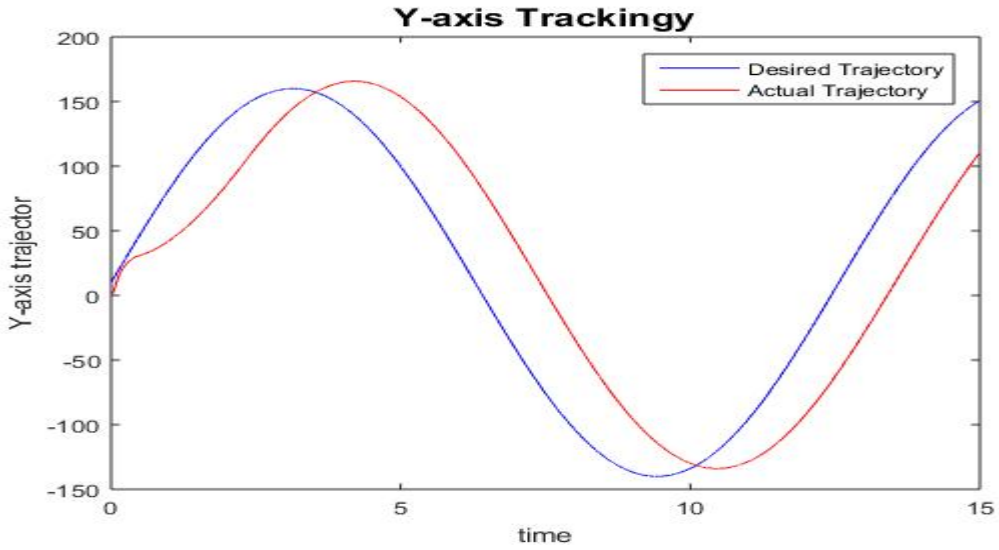
(b)



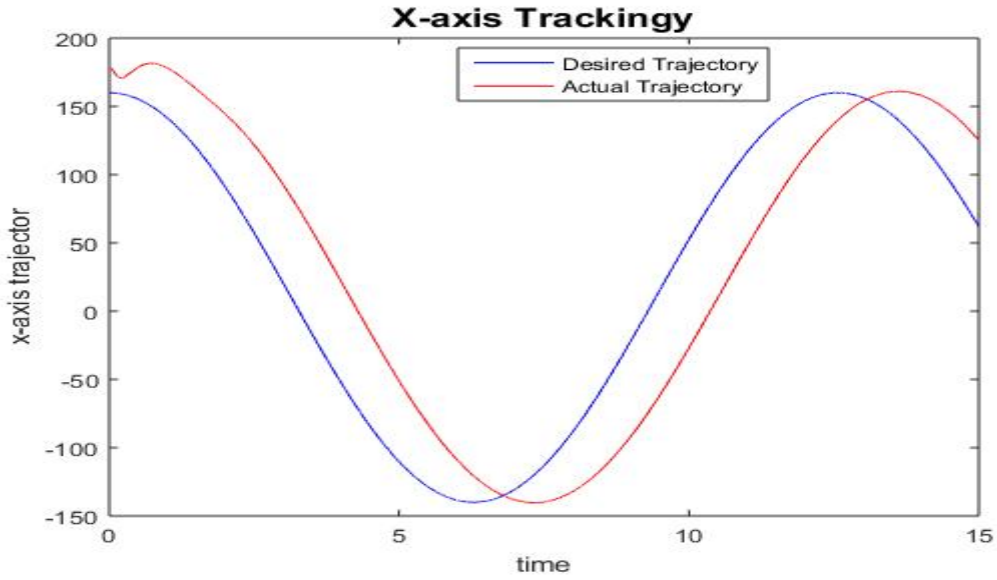
(c)

Fig 4.8. The sliding variables $s, \dot{s}, and\ \ddot{s}$ joint each angles: (a) joint angle 1, (b) joint angle 2, (c) joint angle 3.

The figures below show the tracking of the X and Z axis trajectories respectively. As these plots show the X and Z axis tracking settles before 2 seconds.



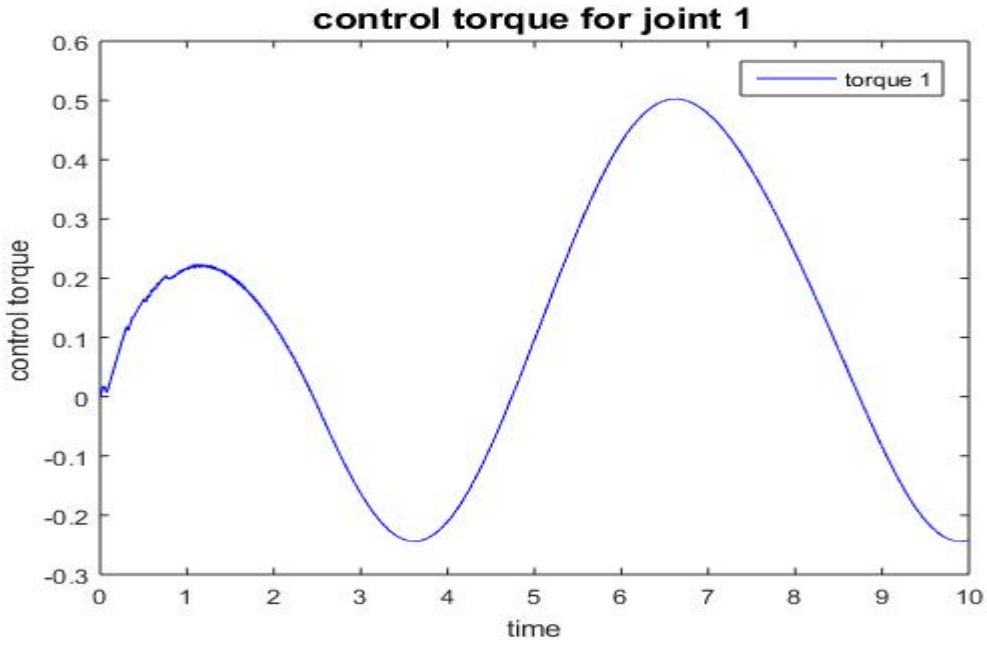
(a)



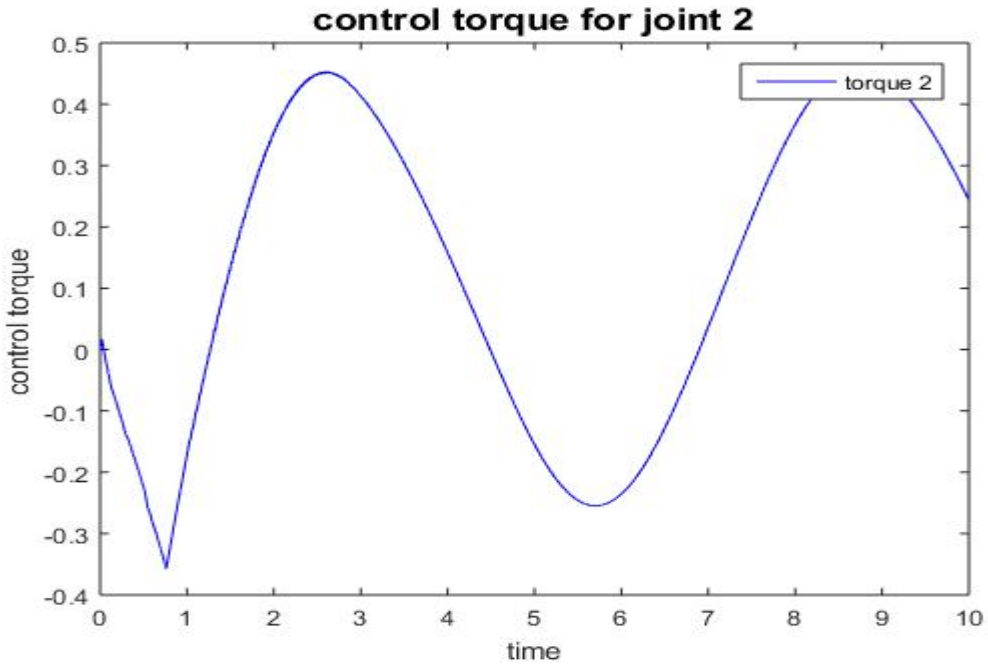
(b)

Fig 4.9. End effector X & Z axis Trajectory tracking

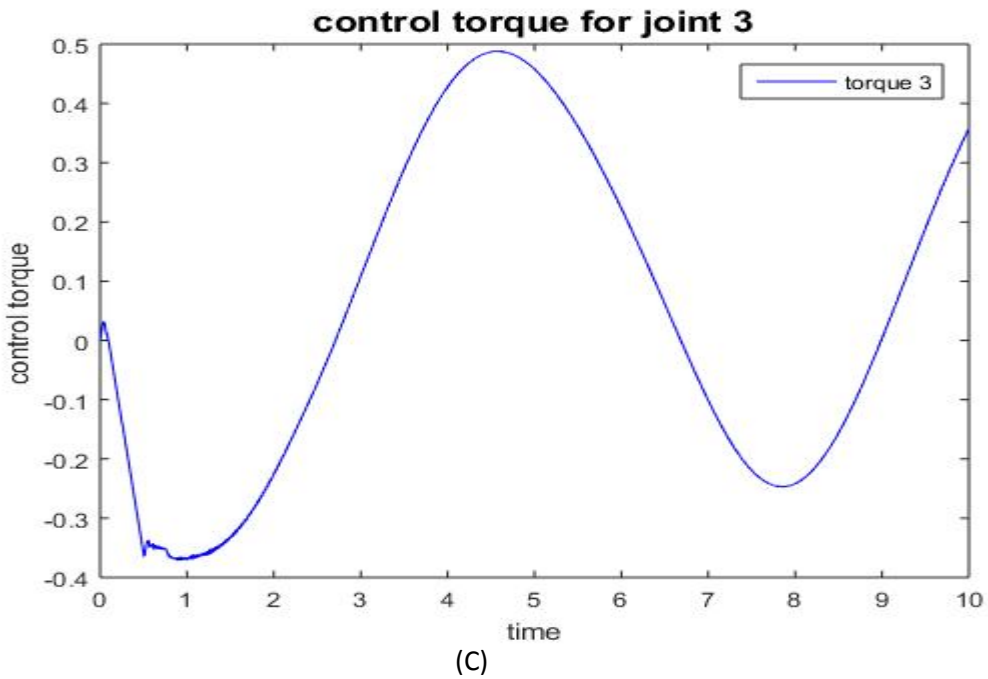
The control torques from the 3-SMC controller are shown for each leg that are required to accomplish the desired trajectory as designed in section 4.2. As shown the 3-SMC control law avoids the chattering usually in the standard SMC control .



(a)



(b)



(C)

Fig 4.10. The 3-SMC control torques for each joint legs: (a) joint 1, (b) joint 2, (c) joint 3.

2. Results for external torque ,disturbance and mass uncertainty introduced into the system

To represent forces and torques acting on a rigid body, SimMechanics provides a Forces and Torques library. One block represents an external force or torque. We can connect a force or a torque to the rigid body frame(s) that we want to apply the force or torque to. The other blocks represent internal forces between two rigid body frames. We can also represent a general internal force, using the Internal Force block, or a specific internal force type, using Spring and Damper Force and Inverse Square Law Force blocks..

In this paper a time varying of external torque disturbance $T = [0.3 \quad 0.4 \quad 0.6]^T$ is applied at the three revolute joint which are also made to have an internal mechanics of damping coefficient of 1,(which is the torque required to maintain a constant joint primitive angular velocity between the base and follower frames).To test the robustness of the 3-SMC controller further ,additional mass of 50g is added to a single lower leg of the parallelogram, which distorts the symmetry of Delta Robot design and changes the dynamics.

The simulation is carried out with all internal mechanics, external disturbance of torque and mass uncertainties included at the same time for the whole course of simulation. The simulation results are shown for trajectory tracking of circular path as done in the previous part .

The next set of figures portrays the trajectory tracking of the end-effector along with its respective error between its desired and actual position with the effect of internal mechanics, external disturbance of torque and mass uncertainties included.

In order to determine the actual location of the end effector, the Inverse Kinematics was employed to solve for θ_1, θ_2 and θ_3 respectively based on the defined desired path. The results from the joint angles of θ_1, θ_2 and θ_3 play a crucial role in the determination of the actual location of the end-effector. The plots of the end effector error in the X-axis and Z-axis are shown in order to easily identify the severity of the absolute accuracy.

The next Simulink Model illustrates the End effector tracking for a circle that lies on X- Z plane with a radius r= 150 mm center at (10,Y_o,10).

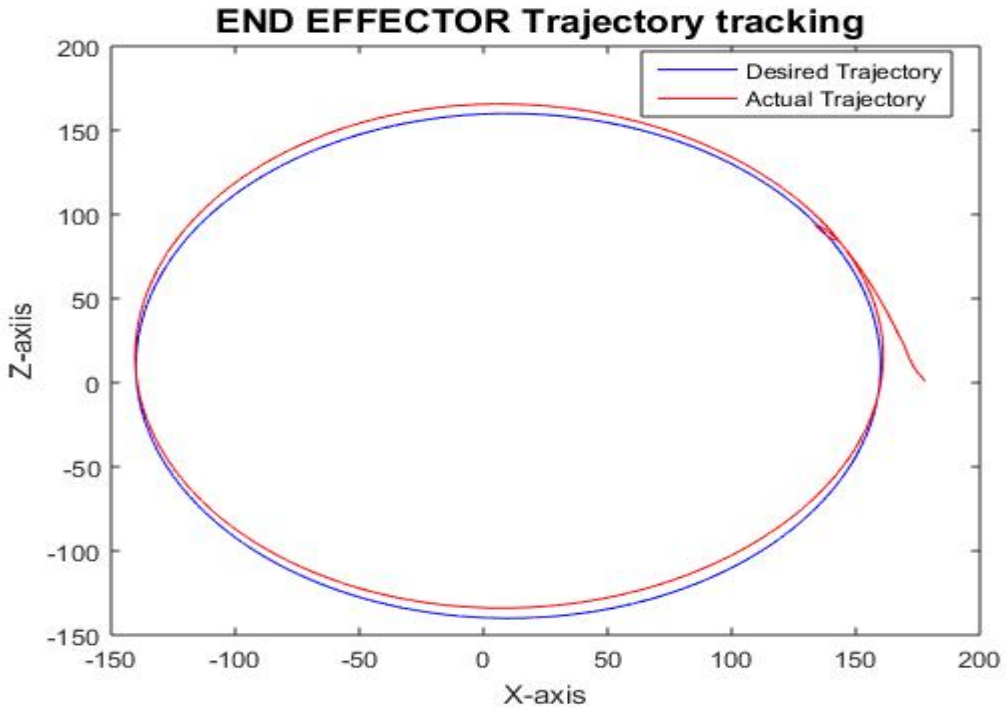
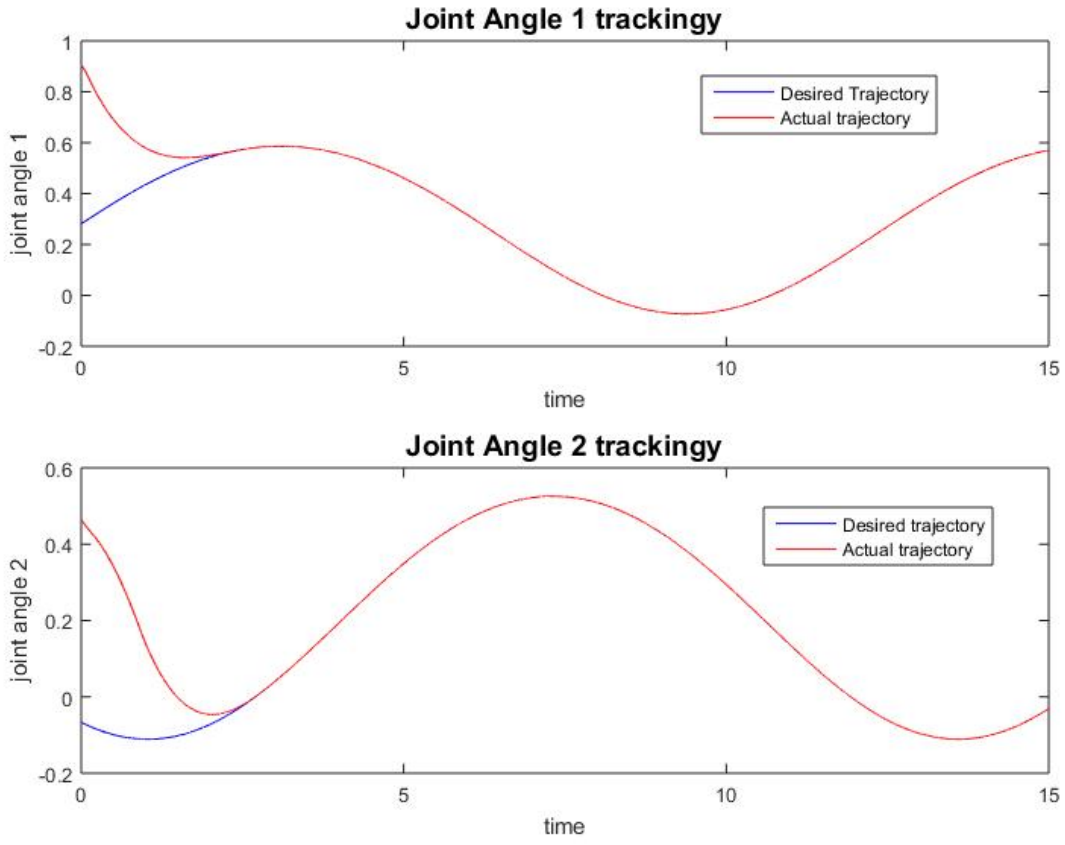


Fig 4.11. End effector trajectory tracking

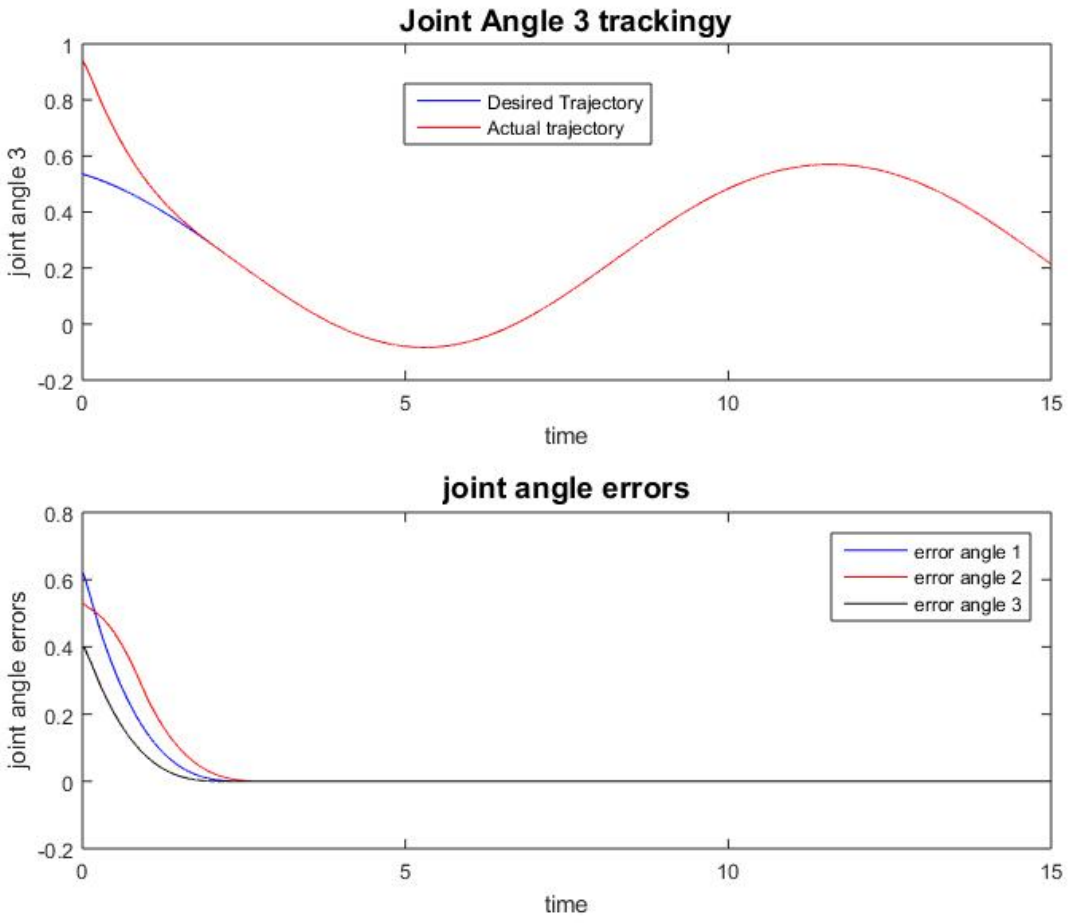
As Fig 4.11. shows the robot is able to track the desired path with an rms of 3mm error irrespective of the applied external disturbance torque, internal mechanics and mass uncertainty.

As the Fig 4.12. (a) & (b) below shows the designed 3-SMC controller has made the system insensitive to external disturbance, internal mechanics and parameter variation by absolutely tracking the desired joint angles in less than 3seconds. the joint angles for the arms follow the desired trajectory in finite time as required from the design of 3-SMC control law.



(a)

Fig 4.12. Joint angle1 & joint angle2 tracking



(b)

Fig 4.13. Joint angle3 tracking & joint angle errors

The figure also shows that the 3-SMC controller has made the joint angle errors to converge to zero in finite time(in less than 3 sec)

As the figures below illustrate the sliding variables $s, \dot{s}, and\ \ddot{s}$ for each joint angles converges to zero in finite time without being affected by the newly introduced internal mechanics, external disturbance of torque and mass uncertainties into a system.

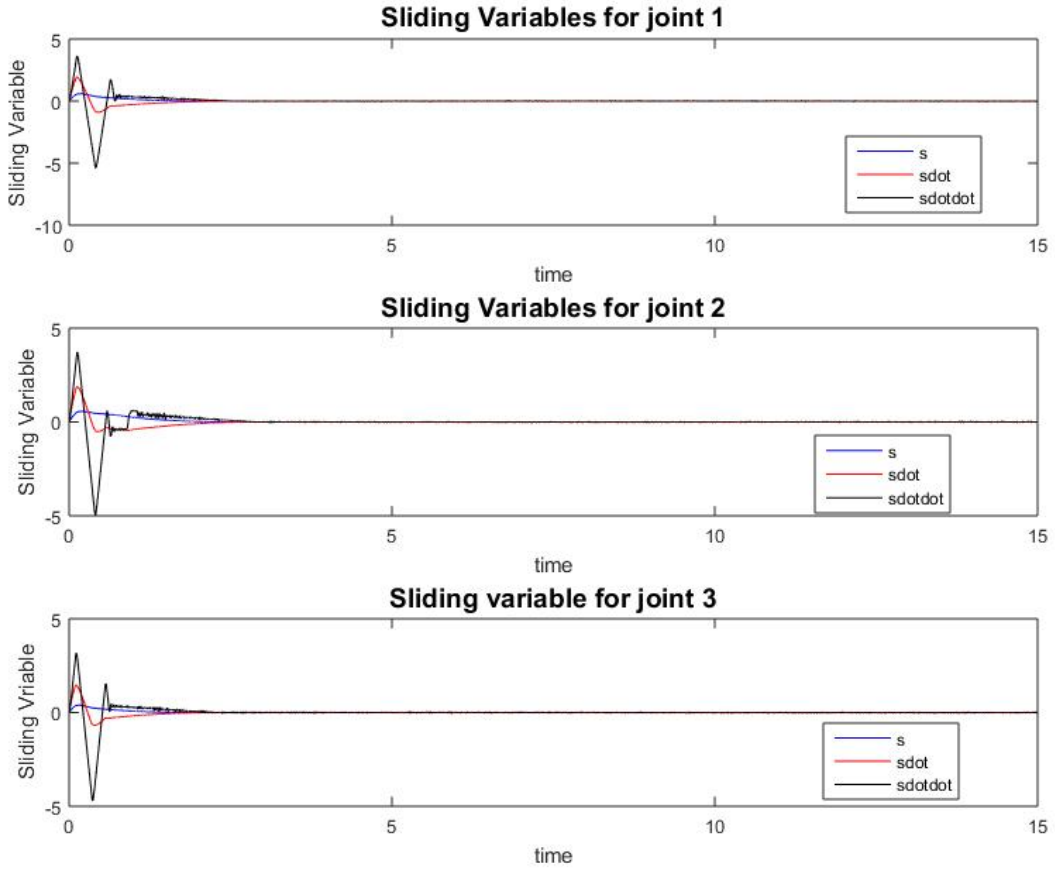


Fig 4.14. The sliding variables $s, \dot{s}, and\ \ddot{s}$ for each joint angles

As the plots shows, the sliding variables $s, \dot{s}, and\ \ddot{s}$ for each joint angles. The sliding variables settles to zero before 3 seconds

Fig 4.15. shows the sliding variables the X and Y-axis trajectory tracking between the desired path and the robots end effector .

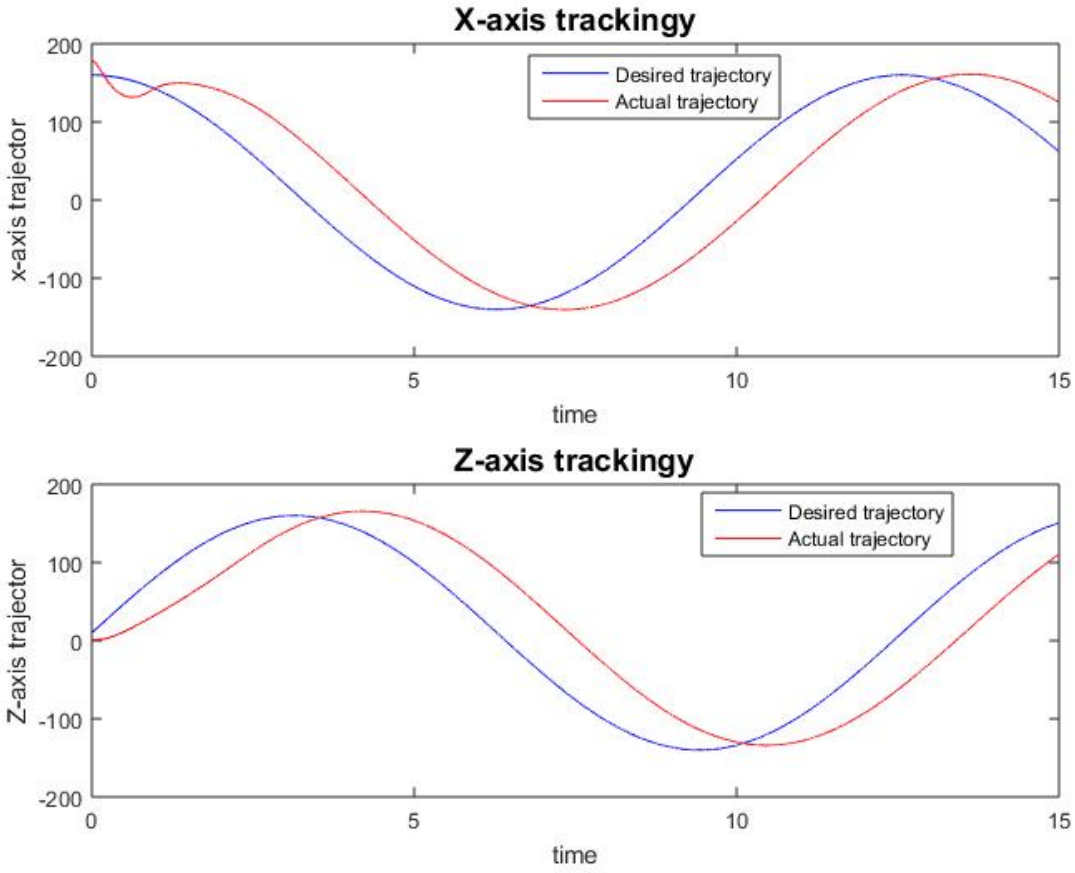
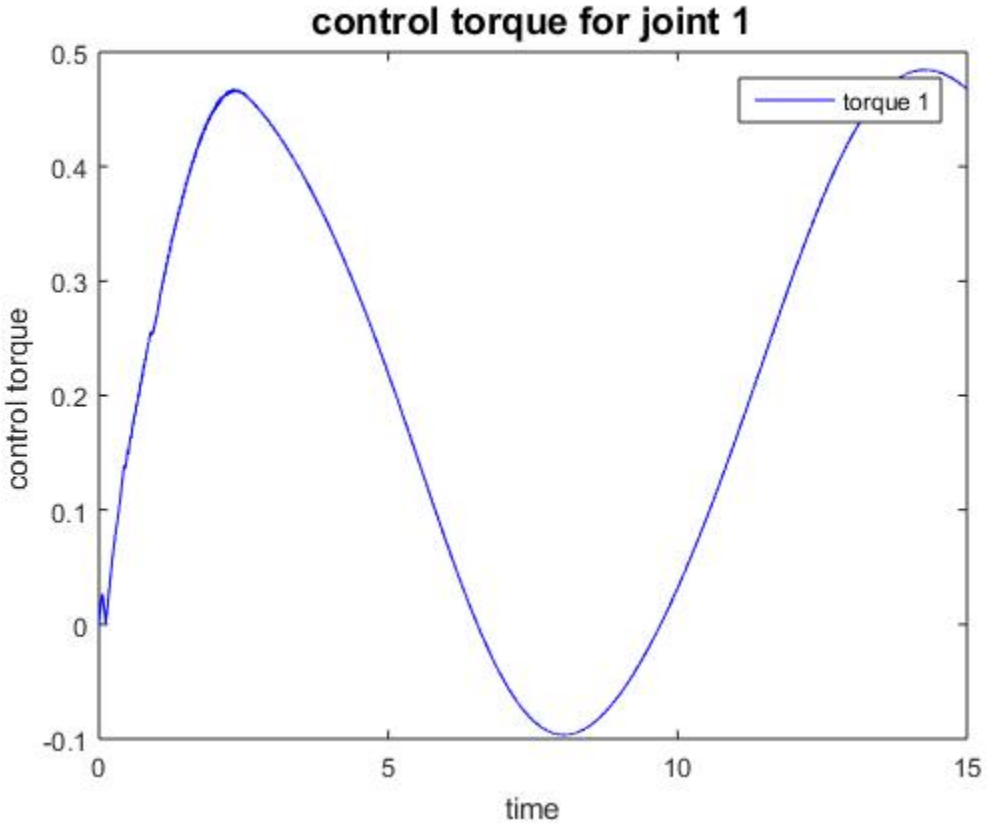
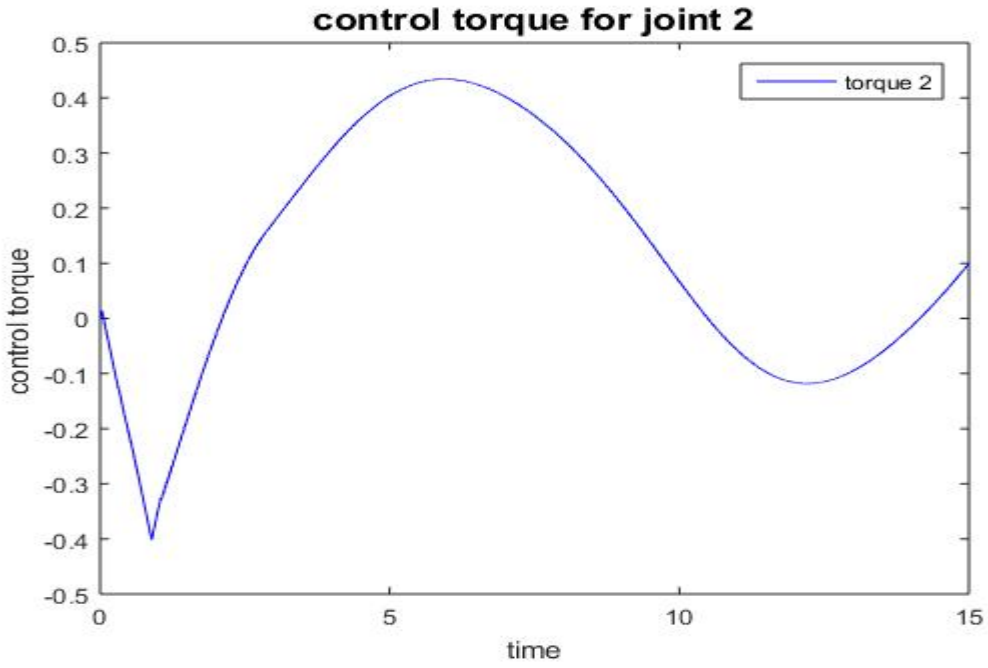


Fig 4.15. End effector X & Z axis trajectory tracking

As the system is now affected by internal mechanics, external disturbance of torque and mass uncertainties which affects the general dynamics of the system, the designed 3-SMC controller ,in principle, has to produces a proper control torque to make the system insensitive to the applied internal and external changes and to maintain the desired trajectory in minimum time. The following figures show the produced torques for the three legs under the mentioned effects acting on the system.


control torque for joint 1
torque 1
control torque
time
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
5
10
15

(a)


control torque for joint 2
torque 2
control torque
time
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
5
10
15

(b)

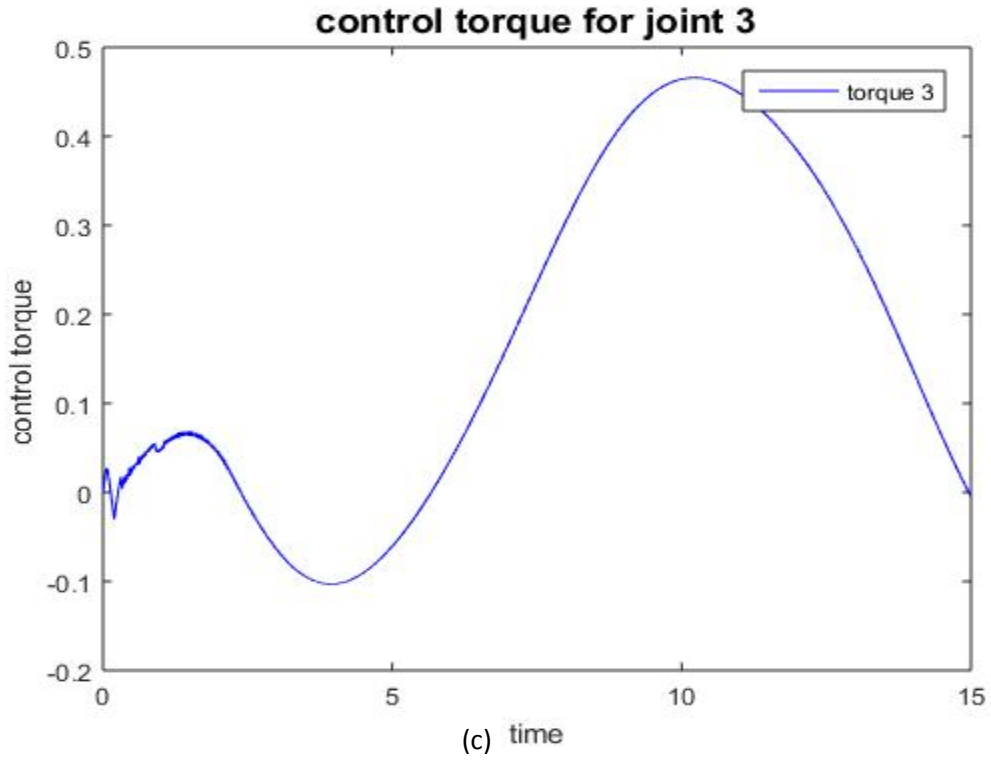


Fig 4.16. The 3-SMC control torques for each joint legs: (a) joint 1, (b) joint 2, (c) joint 3.

As the figure shown above illustrates are the control torques from the 3-SMC controller are shown for each leg that are required to accomplish the desired trajectory as designed 3-SMC control law avoids the chattering usually in the standard SMC control .

4.4.2. Discussions

The final purpose of this thesis's Simulink simulation is to test trajectory tracking response of a 3-DOF Delta Robot. Building the nonlinear dynamic model using SolidWorks program was the best solution to model the Robot, real and not approximated parameters were introduced via CAD dynamic Modeling which is one of the advantages of CAD modeling, but in the other side, the simulation speed is low. The simulation speed of Mathematical dynamic Model is more rapid than CAD dynamic model.

SimMechanics tool introduced an easy way to actuate Robot joints by joint actuator block. Sensing computed torques, angles, angular speeds, and accelerations were possible by joint sensor. Sensing of Cartesian Coordinates of the moving platform was by Body sensor, which facilitated the process of simulation.

Delta Robot needed a high precision controller to control it, a 3-SMC controllers were designed. One of the big advantage of 3-SMC controllers is handling coupling effects in the system and its model independence, which was a typical problem in Robot control design.

4.4.3. Conclusion

SimMechanics presents a powerful tool for modeling mechanics of rigid bodies. It is suitable for modeling of dynamics and kinematics of considerably complicated systems with many joints without using any mathematical description. For these advantageous properties it is often used in the first phase of designing Robotic systems, especially due to simplicity of changing parameters and dimensions of particular bodies without necessity to repeat design of new model.

To get a dynamical model for the systems with more complex bodies of various shapes and connections connected through joints it is advantageous to model them in a compatible 3D-CAD software and then to import the model from the program into the SimMechanics program. The CAD software automatically calculates this tensor and moreover it adds visualization to the developed model beside reducing a possibility of error occurrence at modeling of the system. Mathlab's SimMechanics is continuously including the kinematic constraints of a 3-DOF model to be exactly represented in SimMecahnics ,However ,it has not yet able to to include the limitation of rational freedom for spherical joints.

The 3-DOF CAD model of Delta Robot from SolidWorks is exported to SimMechanics. Inverse kinematics and non-singularity analysis were done in order to determine the workspace region adequately for the system. The concept of sliding mode control from standard SMC through HOSM has been clearly identified. A 3-SMC controller has been designed to ensure stability and robustness of the designed controller subjected to; external disturbance, parameter uncertainty ,mass uncertainty and coupling effects between the Robot parts. A circular trajectory is generated that was used by the robot's end-effector to trace to .

In this thesis a 3-DOF Delta Robot is designed in SolidWorks, a 3-SMC sliding mode controller is developed and a trajectory tracking is achieved. The simulation result shows that the steady state tracking is reached in about 3 seconds for a circular trajectory and X & Z rms error of 3mm. Chattering effect which was seen in 1-SMC is also eliminated by using a 3-SMC controller. The robustness is also demonstrated with a joint's internal mechanics, mass uncertainty. It is also shown that the controller is insensitive to external disturbances.

4.4.4. Recommendations

Model based controllers usually need the amount of torque to be computed in order to make the Robot perform the desired motion. SimMechanics second-generation's Inverse Dynamics actuation mode has eased a complex computation of these forces or torques required to accomplish a desired motion to be achieved which mainly involves singularity in computation. Now it is possible if we know the desired motion, SimMechanics automatically computes torques or forces for us, which ease the complex mathematical computation for this type of control law.

Finally, based on this model a SimDrive blocks may be incorporated to the Simmulink model in order to incorporate the real effects of actuators on Robot kinematics and dynamics.

References

[1] De Luca, A. and Siciliano, B., 1991, "Closed-Form Dynamic Model ofPlanar Multilink Lightweight Robot", IEEE transactions on Systems,Man and Cybernetics, Vol.21, No.4, pp.826-839

[2] Edward Wong Ting Ping," Use of a Delta Robot as a Aalking Machine" 1998 University of Canterbury

[3] R. Clavel; "DELTA, a fast Robot with Parallel geometry", pg 91-100, April, Proc. Int. Symposium on Industrial Robot, 1988.

[3] Eng. Hamdallah A. H., 354/2007,Modeling and High Precision Motion Control of 3 DOF Parallel Delta Robot Manipulator

[4] M. Rachedi, M. Bouri, B. Hemici, Application of an control strategy to the

Parallel Delta, IEEE, 2012

[5] P.J. Zsombor Murray," An Improved Approach to the Kinematics of Clavel's Delta Robot", Center for Intelligent Machines, McGill University, 2009

[6] YangminLi, Qingsong Xu," Dynamic modeling and robust control of a 3 PRC translational parallel kinematic machine", Robotics and Computer-Integrated Manufacturing journal, Science Direct, 2009.

[7] André Olsson, Modeling and control of a Delta-3 Robot, master thesis, Department of Automatic Control, Lund University, 2009.

[8] Craig, J. John. Introduction to Robotics Mechanics & Control: Addison-Wesley Publishing Company Inc, 1986

[9] Manuel Napoleon Cardona Gutierrez, "Kinematics Analysis of a DeltaParallel Robot", IEEE, University of Sonsonate, Salvador,2010.

[10] John J. Craig, "Introduction to Robotics, Mechanics and Control, Third Edition, 2005.

[11] N. Andreff, Parallel Robot modeling, identification and control, 2008

[12] Henri P. Gavin, Matrix Structural Analysis, Department of Civil and Environmental Engineering

[13] Mohsen, Mahdi, Mersad," Dynamics and Control of a Novel 3-DoF Spatial Parallel Robot", International Conference on Robotics and Mechatronics,2013

[14] Serdar Kucuk and Zafer Bingul,"Robot Kinematics: Forward and Inverse Kinematics, Industrial Robotics: Theory, Modelling and Control", Sam Cubero (Ed.), ISBN: 3-86611-285-8, InTech,2006.

[15] D.S. Lee, M.J.Youn, "Controller design of variable structure systems with nonlinear sliding surface," Electronics Letters, vol. 25, no. 25, pp.1715-1716, 1989

[16] Reza N. Jazar," Theory of Applied Robotics, Kinematics, Dynamics and control", second edition, Springer, RMIT university,2010.

[17] W. Perruquetti, J. P. Barbot, Sliding mode control in engineering, Marcel Dekker, 2002

[18] A. Levant, Higher order sliding: collection of design tools, Proc. of the 4th European Control Conference, 1997

[19] A. Levant, Higher order sliding: collection of design tools, Proc. of the 4th European Control Conference, 1997

[20] Y. Shtessel et al., Sliding mode control and observation, control engineering, Springer, 2014

[21] Ahmed Mehamed "control of parallel robot using higher order sliding mode control" Addis Ababa University ,2014

Appendix

A1.Inverse Kinematics

```
function [theta1,theta2,theta3]=Delta Robotinversekinematics(x0,z0,y0)
%calculates the inverse kinematics of 3-DOF Delta Robot
% Robot geometric dimentions

            theta1 = Delta_calcAnglezy(x0,z0,y0);
            x01=x0*cosd(120) + z0*sind(120);
            x02=x0*cosd(120) - z0*sind(120);
            z01=z0*cosd(120)-x0*sind(120);
            z02=z0*cosd(120)+x0*sind(120);

       theta2= Delta_calcAnglezy(x01, z01, y0);  % rotate coords to +120 deg
       theta3= Delta_calcAnglezy(x02, z02, y0);  % rotate coords to -120 deg

function [theta]=Delta_calcAnglezy(x0,z0,y0)

                    e = 199.01;     % end effector
                    f = 841.77;     %base
                    re = 800;       %lower leg length
                    rf = 300;       %lower leg length

        z1 = -0.5 * 0.57735 * f;          % f/2 * tg 30
        z0 =z0 - 0.5 * 0.57735 * e;        % shift center to edge
         % y = a + b*z
            a = (x0*x0 + z0*z0 + y0*y0 +rf*rf - re*re - z1*z1)/(2*y0);
            b = (z1-z0)/y0;

        d = -(a+b*z1)*(a+b*z1)+rf*(b*b*rf+rf);
        if (d > 0)                                      % existing point
         zj = (z1 - a*b - sqrt(d))/(b*b + 1);           % choosindg outer point
         yj = a + b*zj;
             if (zj>z1)
            theta = 180*atan(-yj/(z1 - zj))/pi + 180;
            else
                theta = 180*atan(-yj/(z1 - zj))/pi ;
             end
        else
         theta=0;                                       %non-existing point
        end
```

A2.Direct Kinematics

```
function [x0,z0,y0,d]=  Delta_calcForward( theta1,theta2,theta3)
%calculates the forward kinematics of 3-DOF Delta Robot
% Robot geometric dimentions

                    e = 199.01;     % end effector
                    f = 841.77;     %base
                    re = 800;       %lower leg length
                    rf = 300;       %lower leg length

      t = (f-e)*tand(30)/2;
      dtr = pi/180;
    theta1=theta1 * dtr;
    theta2=theta2 * dtr;
    theta3=theta3 * dtr;

      z1 = -(t + rf*cos(theta1));
      y1 = -rf*sin(theta1);

    z2 = (t + rf*cos(theta2))*sind(30);
    x2 = z2*tand(60);
    y2 = -rf*sin(theta2);

    z3 = (t + rf*cos(theta3))*sind(30);
    x3 = -z3*tand(60);
    y3 = -rf*sin(theta3);

    dnm = (z2-z1)*x3-(z3-z1)*x2;

    w1 = z1*z1 + y1*y1;
    w2 = x2*x2 + z2*z2 + y2*y2;
    w3 = x3*x3 + z3*z3 + y3*y3;

    % x = (a1*y + b1)/dnm
    a1 = (y2-y1)*(z3-z1)-(y3-y1)*(z2-z1);
    b1 = -((w2-w1)*(z3-z1)-(w3-w1)*(z2-z1))/2.0;

    % z = (a2*y + b2)/dnm;
    a2 = -(y2-y1)*x3+(y3-y1)*x2;
    b2 = ((w2-w1)*x3 - (w3-w1)*x2)/2.0;

    % a*y^2 + b*y + c = 0
    a = a1*a1 + a2*a2 + dnm*dnm;
    b = 2*(a1*b1 + a2*(b2-z1*dnm) - y1*dnm*dnm);
    c = (b2-z1*dnm)*(b2-z1*dnm) + b1*b1 + dnm*dnm*(y1*y1 - re*re);
       % discriminant
    d = b*b - 4*a*c;
    if (d < 0)  % non-existing point
        x0=0;
        z0=0;
        y0=0;
```

```
        else
     y0 = -0.5*(b+sqrt(d))/a;
     x0 = (a1*y0 + b1)/dnm;
     z0 = (a2*y0 + b2)/dnm;
    end
```

A3.Circular Trajectory

```
function [x,z] = fcn(t)
x = 10+150*cos(t);
z =10+150*sin(t);
```