Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Passivity-Based Control of Delta Robot for trajectory tracking

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:

Meron Tesfaye

Advisor:

Dr. Dereje Shiferaw

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Passivity-Based Control of Delta Robot for trajectory tracking

By:- Meron Tesfaye Mengesha

APPROVED BY BOARD OF EXAMINERS

Chairman, Department of Graduate Committee	Signature
Dr. Dereje Shiferaw	
Advisor	Signature
Internal Examiner	Signature
External Examiner	Signature

DECLARATION

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

Name: Meron Tesfaye

Signature: _____

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

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

Dereje Shiferaw (Ph.D.) _____

Advisor's Name Signature

Abstract

Parallel manipulators have some special properties in comparison to serial robots, making them increasingly appealing in industrial applications as technology advances. such as: more rigid structure, high orientation accuracy, better control on the limits of velocities and accelerations, good positioning accuracy, high payload-to-weight-ratio and low inertia of moving parts. On the other hand, parallel robots are nonlinear multi-input multi-output (MIMO) systems whose dynamics are governed by a highly nonlinear coupled, time-varying system with many uncertainties such as load variation, friction and external disturbances, making robot control more difficult and necessitating a robust control system.

This thesis provides passivity-based control as one of the strategies for designing robust controllers for a 3-DOF Delta Robot trajectory tracking. Passivity is a fundamental attribute of many physical systems that may be nearly characterized in terms of energy dissipation and transformation and its intrinsic input output property quantifies and defines a system's energy balance when external inputs imitate to produce some outputs. Furthermore, it is a method of operating a system with the goal of making the closed loop system passive. The 3-D model of a 3-DOF Delta robot is designed in 3D-CAD (Solidworks). Hence, this modeling environment enables to introduce real but not approximated parameters of the robot and eased computation of large matrixes. Dynamical model with all the kinematic constraints for a robot will simply be found by exporting a 3D-CAD model of the robot to Simscape Multibody.

The kinematics, dynamics and singularity analysis of the 3-DOF Delta robot has been carried out, a passivity based controller has been designed and its performance is tested by tracking a circular trajectory centered at (15,15,Z_0) on X-Y plane with 160mm radius. The simulation result shows that the steady state tracking is reached in about 3 seconds for a circular trajectory and X & Y rms error of 2.88mm.

The robustness of the designed controller is illustrated mathematically and the simulation is carried out on MATLAB, and it shows that the controller is robust to external disturbances.

Keywords— Passivity Based Control (PBC), 3D CAD (solidworks) model of Delta Robot, 3-DOF Delta Robot, MATLAB/Simulink

Acknowledgment

I am incredibly appreciative to Female Scholarship awarded by Addis Ababa University for giving me the opportunity to study here. First and foremost, I want to express my gratitude to Dr. Dereje Shiferaw, my adviser, for his help, guidance and support throughout the whole thesis work. I also owe a huge debt of gratitude to my family, friends and colleagues for their unwavering support, encouragement and complete understanding throughout the years.

Contents

List of Figures

List of Tables

List of Acronmys

$H\infty$ H Infinity.

J_θ Jacobian matrix in joint space.

J_p Jacobian matrix in Cartesian space.

3-DOF Three Degree of Freedom.

3D Three - Dimensional.

CAD Computer aided design.

DC Direct current.

FPK Forward Position Kinematics.

IPK Inverse Position Kinematics.

PBC Passivity based controller.

PD Proportional derivative.

1 Introduction

1.1 Background

The design of intelligent, autonomous machines to perform tasks that are dull, repetitive, hazardous, or that require skill, strength, or dexterity beyond the capability of humans is the ultimate goal of robotics research. Undersea, space, and planetary exploration, manufacturing, excavation, toxic waste remediation, construction and robotic assisted surgery are all examples of such tasks [1].

From children's toys to guided missiles, the name "robot" has been applied to a wide range of mechanical devices. One of the most important criteria for classifying robots is the manipulator arm [1]. There are essentially two types of robot manipulators, serial and parallel. Serial manipulators are made up of a succession of interconnected links that form a kinematic chain and each joint of the kinematic chain is usually actuated. An open chained mechanism is the name given for this type of structure. Parallel manipulators, on the other hand, are made up of multiple kinematic chains that are linked in a parallel fashion. The kinematic chains operate together to move the end-effector, which is a common point and usually consists of manipulator that performs a certain task [1–6].

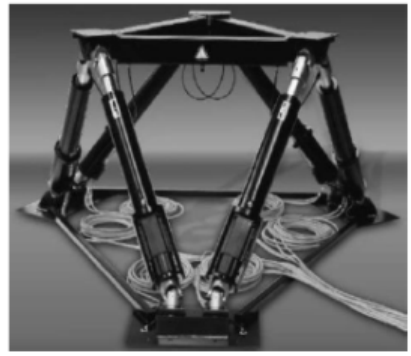
Day by day, the parallel manipulator's uses in numerous fields become more evident and it is increasingly being used in precise manufacturing, medical science, and commercial applications by orienting the manipulator in space at a high speed with the required accuracy [2–4, 7]. Even though conventional serial manipulators have a large workspace and dexterous movement, parallel manipulators have recently become very popular. The primary issue with serial ones is that their cantilever structure makes them prone to bending under high load and vibration at high speeds, resulting in a lack of precision and a slew of additional issues. [2, 4–7].

Parallel kinematic manipulators provide numerous advantages over their serial counterparts for certain applications. Among the advantages are greater load bearing capacities hence total load can be shared by number of parallel links connected to fixed base, low

inertia, higher structural stiffness, reduced sensitivity to certain errors, easy controlling and built-in redundancy but smaller and less dexterous workspace due to link interference, physical constraints of universal and spherical joints and range of motion of actuators and suffer from platform singularities. The fully parallel designs of robots have all actuators in or near the base, which results in a very low inertia of the part of the robot that has actually to be moved. Hence, a higher bandwidth can be achieved with the same actuation power. The high-speed motion under a high load, good position accuracy and light construction make the parallel robots very good solution in many applications such as laser cutting, flight simulation, pick and place operations, assembly and many others. [2, 5, 6].

Because of the closed kinematic loops, the kinematic equations for a parallel kinematic mechanism are far more complicated than for a serial (open) kinematic structure. Parallel manipulators are also known as closed loop manipulators. In order to develop high-performance robots, models will indeed be essential for simulation and performance prediction, as well as for model-based compensation in the control system to attain advanced performance. Any mechanical systems composed of conventional joints, traditional parallel manipulators suffer from errors due to backlash, hysteresis, and manufacturing errors in the joints. In contrast to serial manipulators, un-actuated or passive joints might be present. The inclusion of passive joints and multi-DOF joints distinguishes the kinematic analysis from that of serial manipulators. Direct kinematics is much harder and involves elimination of passive joint variables in parallel kinematics. Because of many parallel links and closed loop architecture, parallel robots are intrinsically more accurate than serial robots because their errors are averaged rather than added cumulatively [6].

(a)



(b)

(c)

Figure 1.1: (a) shows a SCARA-type serial-architecture Robot (www.mecademic.com) (b) Stewart platform /Parallel Robot (c) Flex Picker Robotics Parallel Delta Robot (www.abb.com)

Among parallel kinematic Machines, the simplest and sophisticated parallel robot which is capable of moving in 3D space with superior speed and invented in the early 1980s by Clavel, is the so-called Delta Parallel Robot [8, 9]. The Delta robot is one of the most common parallel robot manipulators, which is comprised of three parts: a fixed platform, a moving platform and some identical and symmetrical parallel links. Each parallel link includes upper link and lower link. When the upper links are actuated by motors, which attached to the fixed platform and apart at 120^0, the moving platform produces three translational motions because the parallelogram-based lower links maintain the orientations of the moving platform [10]. Optionally by introducing fourth inner leg at the end-effector platform, about the axis perpendicular to the platform, the degree of freedom of Delta robot can be extended to four, 3-dof for XYZ translation plus a fourth control angle rotational freedom at the end-effector platform. Based on the joint which driven the input, the delta robot can be categorized into Revolute-Input Delta Robot and Prismatic-Input Delta Robot [11]. In this thesis, Revolute-Input Delta Robot which has three DOF is used to demonstrate passivity based control for trajectory tracking.

The Delta parallel Robot was designed to operate light and small objects at a rapid speed, which has sparked interest in sorting and collating, pick-and-place applications, and even in the machine-tool sector due to enhanced stiffness. Industries that take advantage of the high speed of delta robots are the packaging industry, medical and pharmaceutical industry. For its stiffness it is also used for surgery and other applications include high precision assembly operations in a clean room for electronic components. The structure of

a delta robot can also be used to create haptic controllers. More recently, the technology has been adapted to 3D printers as well [8–11].

In recent decades, many results have been reported about dealing with control problems from the perspective of energy transformation, for which passivity-based control have been widely adopted [12]. The term passivity based control (PBC) was coined in 1989 in the context of adaptive control of robot manipulators to describe a controller methodology with the goal of making the closed loop system passive. The robot dynamics describe a passive map from input torque to output link velocities, thus this goal appeared very natural in this setting. As a matter of fact, passivity property is inherent to many other physical systems such as electrical and electromechanical [13, 14].

Passivity is one of the most fundamental properties of robotic or any mechanical systems. This passivity property, hold both for linear and nonlinear mechanical systems, has been a crucial tool for robot control, as it allows control designers to exploit the nonlinear dynamics of robotic systems, instead of attempting to completely cancel out and replace it by linear dynamics (e.g., computed torque control, feedback linearization. This then necessarily results in improved robustness of the controller. Furthermore, with its storage function often serving as a basis for Lyapunov function construction, this passivity property has also been instrumental to enforce stability of the closed-loop system in a robust manner, either when the robot is operating alone or interacting with exogenous objects and environments. Many important topics in robotics have been benefited substantially from this passivity-based control framework, including manipulator motion control, interaction control, underactuated robots, flexible manipulators, robot hand, biped walking, teleoperation, and even aerial robots [12–17].

1.2 Problem Statement

The applications of robot manipulators have increased considerably in several manufacturing tasks such as assembly, drilling, painting, and welding to mention a few. These tasks require modelling of the robot dynamics, trajectory planning and generation, and control strategies design and implementation. On the other hand, the dynamics of robot manipulators are usually affected by disturbances, including payload variation, the effects of friction in the joints, external forces, etc. If these disturbances are not compensated, the performance of the robot manipulator is severely affected and may lead to instability [18].

Though the dynamics of a robot manipulator is governed by highly nonlinear coupled, time-varying system with many uncertainties such as load variation, friction and external disturbances. With the advancement of technology, the parallel robot has become more appealing in industrial applications where high precision motion, high speed, high-accuracy and high-acceleration are need [10]. Thus, the design and development of robust control for such complex dynamics has received particular attention in the last decades. On the other hand, Passivity based controller is used for achieving stability of nonlinear systems by making the system passive, because passive systems are by nature asymptotically stable systems. It produces no unduly big control effort and has inherent robustness against plant uncertainty and disturbance since it heavily exploits system properties.

In this thesis one of the most successful industrial parallel robot, a three degree freedom of 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. For the inherent mechanical structural constraints of delta robots, they would make the motion control more complex than serial manipulators. A delta robot control system is a highly coupling system and easy to induce position errors or chatter due to the effects of other attached limbs.

1.3 Objective

1.3.1 General Objective

The general objective of this thesis is to design a Passivity Based Control(PBC) system for trajectory tracking of 3-DOF Delta Robot.

1.3.2 Specific Objectives

The specific objectives of the thesis are:

- Formulating the kinematics equations of the Delta Robot
- Building the 3-DOF Delta Robot 3D model using computer aided design (CAD)
- To derive the dynamics equations of the delta three robot
- Establish passivity-based controller for 3-DOF revolute delta robot to track the intended trajectory
- Demonstrate the performances of the system using MATLAB/Simulink

1.4 Contribution of the Thesis

One of the major considerations of control system design is achieving the control goal with minimized control effort and this thesis contribute the application of passivity based control to a three degree of freedom Delta Robot. By nature, passive systems are asymptotically stable systems and Passivity based controller strongly exploits the system properties. Therefore, it does not produce unduly large control effort and has inherent robustness against plant uncertainty and disturbance.

The other contribution of the thesis is the development of the 3D model of 3 DOF Delta Robot in CAD(Solidworks). This modeling environment enables to introduce real parameters of ABB Flex Picker IRB 360- 1/1600 commercial Delta Robot which can also be used for further researches. On top of this, the kinematics problem of the 3DOF Delta Robot is clearly formulated in detail using geometrical approach in this thesis. Thus, one can be clear on kinematics analysis of any robotic system.

1.5 Methodology

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

- Mathematical modeling: The first stage in mathematical modeling is formulating the kinematics of 3-DoF Delta robot. Specifically, the kinematics used here is the inverse kinematics which gives the determining joint variables values required to achieve the desired position and orientation for the end-effector. Secondly, the derivation of the Jacobian matrix performed. This matrix used for finding the robot dynamic parameters as well as for singularity analysis. Thirdly, the dynamics of the three-DoF Delta robot has been studied.
- CAD designing of 3-DOF Delta robot model: SolidWorks 2019 has been chosen as 3D-CAD modeling program to effectively meet the requirement of a 3-DOF Delta robot system
- Controller design: passivity-based controller has been designed for the three degree of freedom Delta robot.
- Simulation: MATLAB R2019b 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 passivity-based controller has been tested on the imported Cad model of three-dof Delta robot while tracking the circular path trajectory.

The below flow chart shows the overall methodology for passivity-based control of 3DOF Delta robot for trajectory tracking.

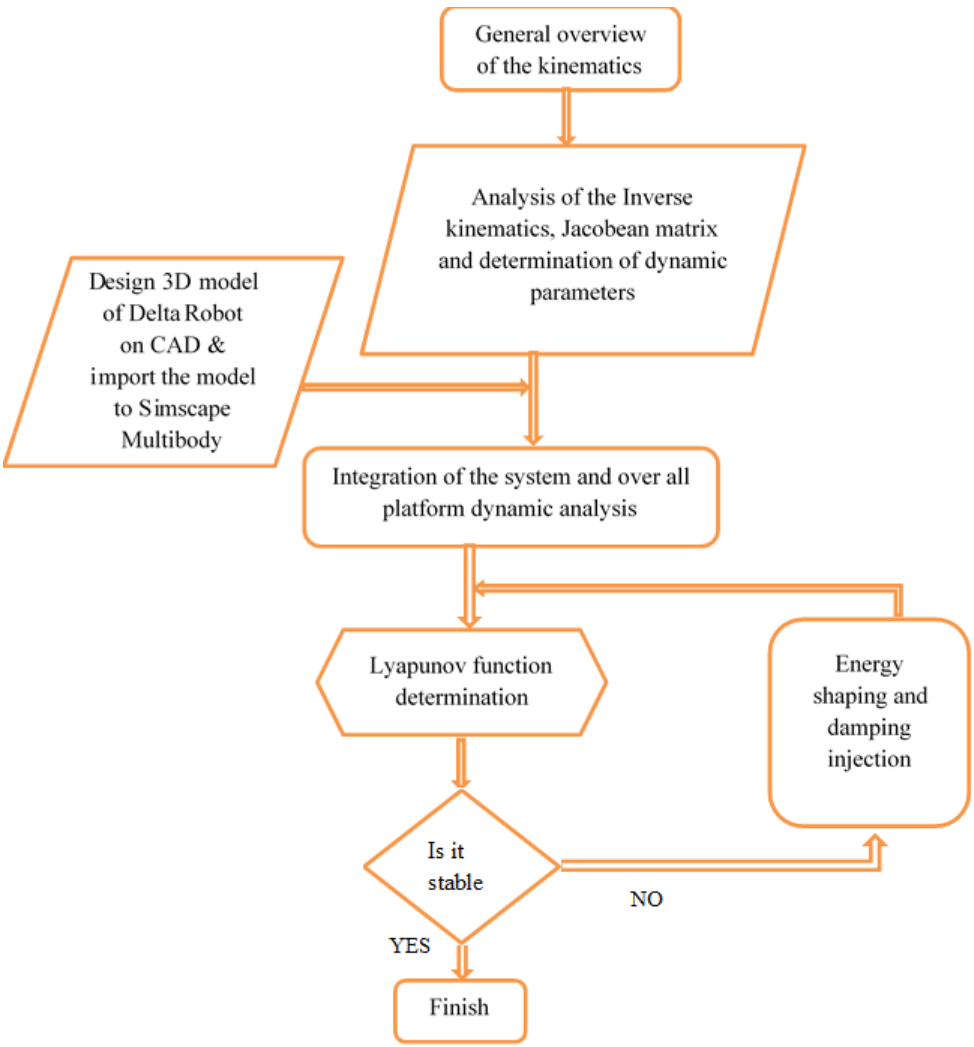


Figure 1.2: Flow chart of passivity-based control of 3DOF Delta robot for trajectory tracking

1.6 Outline of the Thesis

This paper is organized in five chapters. The background about 3-DOF Delta Robot and Passivity based controller, objective of the thesis and its contribution, methodology and reviewed literatures has been covered under chapter one. Chapter two introduce the modeling of a 3-DOF Delta Robot; the modeling includes both forward and inverse kinematic and dynamic aspects. Chapter three addresses the design control scheme, Passivity Based Controller (PBC). Stability analysis of the closed-loop system and the robustness of the designed controller has been covered as well. Chapter four presents the 3D-CAD modeling, PBC control simulation setup, simulation result and discussion. At last, the conclusion, recommendation and Future work of the thesis presented in chapter five.

2 Literature Review and Theoretical Background of Delta Robot

2.1 Literature Review

Parallel robots are more and more promising in the solution of recent application problems of robotics technology. Parallel robot applications own several outstanding features which make them more preferable than the use of serial robots. One of the most popular parallel machine is the Delta robot, which consists of a three (or four) degrees of freedom (DOF) parallel arms and a moving platform which is performing a moving sequence in terms of the actuators attached to a fixed base. The Delta robot originally invented by Clavel in 1988. It was designed to be used in industries such as electronics, food, and pharmaceutical that demand a high level of hygiene and consistent product standards. The Delta robot constitutes technological innovation in the robot industry and has attracted much research interest since it was brought to the industry for pic and place operations. The straightforward explanation for this is that the robot offers a very high accelerations compared to standard serial robots because of the light weight of the moving parts. 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 solely the weight of the driven link but also the load of the actuator for the following links. This results in moving heavy masses and a low dynamic motion response especially on a big machine. On top of this, the force (moment) characteristics of the parallel robots are better than those for serial ones. This is due to the parallel actions of arms. For example, the static force acting on a serial arm can be distributed to three forces in parallel robot cases giving less deformation and providing more accuracy. 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 [4, 11, 14, 19].

Several designs of Delta robot are available in the literatures, from translational/rotational Delta robots to topological variations of the robot design. On the other hand, day after

day with the development of technologies, the parallel robot became more attractive in industrial, tough the dynamics of a robot manipulator is governed by highly nonlinear coupled, time-varying system with many uncertainties. Thus, many scholars have been implementing different control strategies to get better performance and some are presented below. This thesis addresses the problem with passivity based control strategy.

Paper [10] proposes neural network based dynamic trajectory tracking of Delta parallel robot. The author develops both forward and inverse kinematics using geometric method and built a simplified dynamic model by using virtual work principle. As a result, computed torque method-based controller is obtained. The author mentioned that the major impediment to development the work was the high coupling and nonlinear properties of the dynamics model. In order to compensate errors caused by the uncertainties of the model's parameters, Neural Network based controller is proposed and executed by the joint simulation of Adams and MATLAB. So as the computed torque method-based controller. According to the work the results show that the performance of Neural-Network based controller is quite effective in comparison to the Computed-Torque method.

[15] claims due to some desired properties such as low cost, reliability, ruggedness, and high output torque, the induction motor has been used as actuator for the position control of each joint of the SCARA robot. However, this generates complex nonlinearities that make it difficult to control. In order to adders this problem [15] proposes a PI^{ϑ} and a fractional-order passivity-based adaptive controller for trajectory tracking of the SCARA robot. The suggested control consists of a PI^{ϑ} and a fractional-order passivity-based adaptive controller, based on the Caputo-Fabrizio and Atangana-Baleanu derivatives, respectively. [15] uses the Euler-Lagrange formalism technique to develop the fractional-order dynamic model of the robot manipulator, as well as to obtain the model of the induction motors which are the actuators that drive their joints. The effectiveness and robustness of the suggested control approach were shown through simulation results. The suggested fractional-order control approach is contrasted to its integer-order counterpart, composed of the PI controller and the conventional passivity based adaptive controller, in order to emphasize the performance superiority and efficacy of the proposed control strategy.

The work presented in [18] developed an adaptive robust iterative learning control in order to solve the trajectory tracking problem of a parallel Delta robot performing repetitive tasks and subjected to external disturbances. The proposed control scheme in-

cludes an adaptive proportional–derivative controller to rump up the convergence rate, a proportional–derivative-type iterative learning control to improve the tracking performances through the repetitive trajectory as well as a robust term to compensate for both repetitive and nonrepetitive disturbances. The author made of an assumption of alignment condition instead of the classical assumption of resetting conditions. The disadvantage of using controller which works under the resetting condition is that, the external disturbances were assumed to be repetitive over a finite time interval, which is not always the case in practical. The other advantage of the proposed controller compared to some control strategy like PD control, PD plus feedforward control and PD plus sliding mode control, it can learn from repetitive operation and improve tracking performance. The asymptotic convergence of the proposed control [18] is proved using Lyapunov analysis and according to the simulation result the tracking error decreases through the iterations. To demonstrate the efficiency of the suggested control approach, simulations and experiments are carried out on a Delta robot.

A self-learning interval type-2 fuzzy neural network (SLIT2FNN) control technique for the Delta robot's trajectory tracking problem is presented in [20]. The controller has a parallel structure, that combines an interval type-2 fuzzy neural network (IT2FNN) controller and a traditional proportional-derivative (PD) controller. [20] use the PD controller to compensate the transient performance, and use the IT2FNN to learn the dynamic characteristics of the system. The work presented in [20] used a trapezoidal interval type-2 fuzzy membership functions (IT2MF) arrangement, thus it enables the adaptation laws to have an analytical form. A learning algorithm based on sliding mode control (SMC) theory is proposed for the parameter training of the IT2FNN system. The control algorithm learns from the feedback error online and tunes the parameters of the IT2FNN, and will become the main source of the control signal after several learning iterations. The benefit of the proposed control is unlike model-based control, it has no requirement of prior information and constraint conditions from the robot plant. Lyapunov stability method is employed to prove asymptotic stability of the proposed approach. [20] validate the proposed controller on trajectory tracking problem of delta robot. Simulation results illustrate that the proposed SLIT2FNN control approach produces higher trajectory tracking accuracy and more robust to uncertainties as compared to its counterparts.

The work presented in [21] concerns the application of the $H\infty$ robust control law to the three degrees of freedom direct drive Delta parallel robot. In this thesis the state space dynamic model was developed first and then its linearization around an operating point was

carried out. H∞ controller is synthesized based on the linear model. In order to simulate the kinematics and the dynamics of the robot and to validate the control results of the mechanical model, a Simmechanics1 multibody has been developed. The presenter considers the feed forward pre-computed torque to improve the tracking performances and increase the movement dynamics. Two high speeds pick and place trajectories have been tested and robustness analysis through methodical simulation results are presented and commented. According to the paper the results for the semi-elliptic trajectory are very satisfactory and meet the requirements of the specifications even for a high cadence of 3 Hz. According to the presenter the point to point rectangular trajectory is not recommended, since the high accelerations of the trajectory and the important torques lead to the saturation of actuators.

The above mentioned works are only a few , many scholars have been implemented different control strategies to improve the performance hence, the 3-DOF Delta robot is a nonlinear, multivariable, coupled system and parameters of the system such as gravitational load vary from task to task and may also be subjected to uncertain nonlinearities. On the other hand, researchers evaluate the performance of their controllers on different schematics of Delta robot. Thus, several designs of Delta robot are available in the literatures ranging from translational/rotational Delta robots to topological variations of the robot design. This thesis addresses the control difficulty of the 3-DOF revolute type Delta robot with passivity based control strategy. Most of the controller provide a control effort that compensates the effect of the error. But in PBC case, by nature passive systems are asymptotically stable systems and Passivity based controller strongly exploits the system properties and quantify and qualify or define system energy balance when external inputs imitate to produce some outputs, it is possible to reduce the error from occurring in the first place rather than compensation. Therefore, this control schema does not produce unnecessarily large control effort and has inherent robustness against plant uncertainty and disturbance.

2.2 Schematics of Delta Robot

The Delta robot is one of the most common parallel robot manipulators [10], it consists of three parts: a fixed platform, a moving platform and some identical and symmetrical parallel links. Each parallel link includes upper link and lower link. When the upper links are actuated by motors, which attached to the fixed platform and apart at 120^0, the moving platform produces three translational motions because the parallelogram-based lower links maintain the orientations of the moving platform. Thus, the Delta Robot has 3-degrees-of-freedom (dof) (4^{th} optional), 3-DOF for XYZ translation, plus a fourth inner leg to control angle rotational freedom at the end-effector platform about the axis perpendicular to the platform [11, 22]

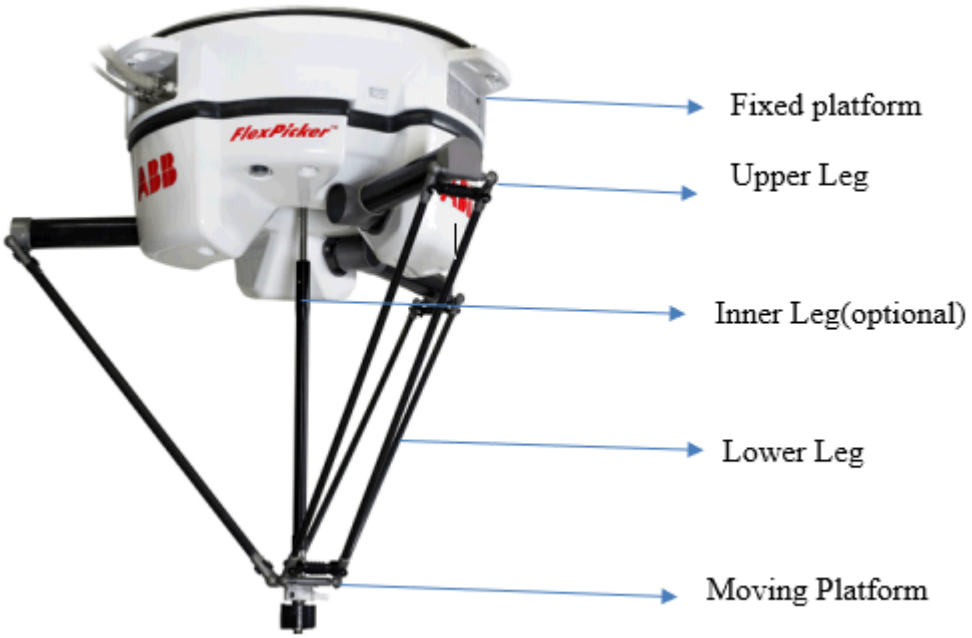


Figure 2.1: A 4 DOF ABB Flex Picker Delta Robot Structure (www.abb.com)

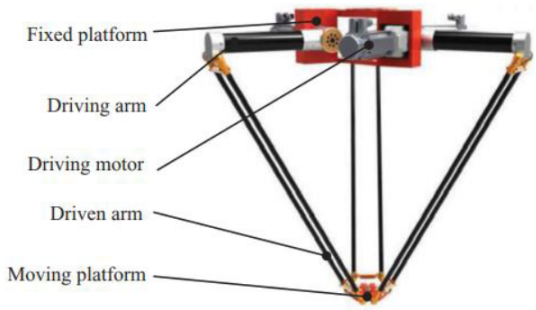


Figure 2.2: The structure of the 3DOF Delta Robot [22]

Based on the joint which driven the input, [11] categorize the Delta Robot into Revolute-Input Delta Robot and Prismatic-Input Delta Robot.

2.2.1 Revolute Input Delta Robot

As shown Fig [2.3], the 3-dof Delta Robot is composed of three identical RUU legs in parallel between the top fixed base and the bottom moving end-effector platform. The top revolute joints are actuated via base-fixed rotational actuators. Their control variables are $\theta_i, i = 1, 2, 3$ about the axes shown. In this model θ_i are measured with the right hand, with zero angles defined as when the actuated link is in the horizontal plane. The parallelogram 4-bar mechanisms of the three lower links ensure the translation-only motion. The universal (U) joints are implemented using three non-collocated revolute (R) joints (two parallel and one perpendicular, six places).

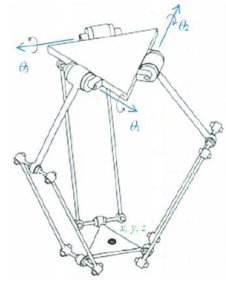


Figure 2.3: Revolute Input Delta Parallel Robot Diagram [elmomc.com/capabilities]

2.2.2 Prismatic Input Delta Robot

The Prismatic-Input Delta Robot is fundamentally similar to the original Revolute-Input Delta Robot. The major difference is that the three inputs now are driven by three linear-sliding prismatic joints instead of three revolute joints. This design change simplifies the kinematics equations and the inverse position kinematics and forward position kinematics equations and solutions significantly, because the three prismatic inputs are aligned with the B frame Z axis, and there are neither sines nor cosines required as in the revolute-input case.

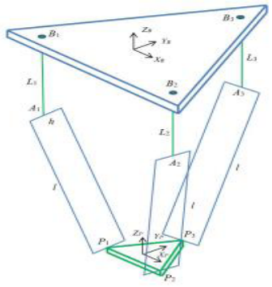


Figure 2.4: Prismatic-Input Delta Robots[deltamaker.com]

As shown below, the 3-dof prismatic-input Delta Robot is composed of three identical PUU legs in parallel between the top fixed base and the bottom moving end-effector platform. The control variables are $L_i, i = 1, 2, 3$ The three-dof Delta Robot is again capable of XYZ translational control of its moving platform within its workspace.

Figure 2.5: Prismatic Input Delta Robots [deltamaker.com]

2.3 Kinematics of Delta Robot

Kinematics aims to provide a description of the spatial position of bodies or systems of material particles, the rate at which the particles are moving (velocity), and the rate at which their velocity is changing (acceleration) disregarding the causative forces. Basically, the kinematics for an industrial Robot can be distributed into three different problem formulations, Forward Kinematics, Inverse Kinematics and the Velocity Kinematics.

The forward kinematics problem is to determine the position and orientation of the end-effector, given the values for the joint variables of the robot. The joint variables are the angles between the links in the case of revolute or rotational joints, and the link extension in the case of prismatic or sliding joints. contrasted with this, inverse kinematics problem concerned with determining values for the joint variables that achieve a desired position and orientation for the end-effector of the robot [23]. In case of the 3-dof Delta Robot the joint variables are θ_1 , θ_2 and θ_3 and the desired position of the moving platform is given by the Cartesian X,Y,Z as shown in the following figure.

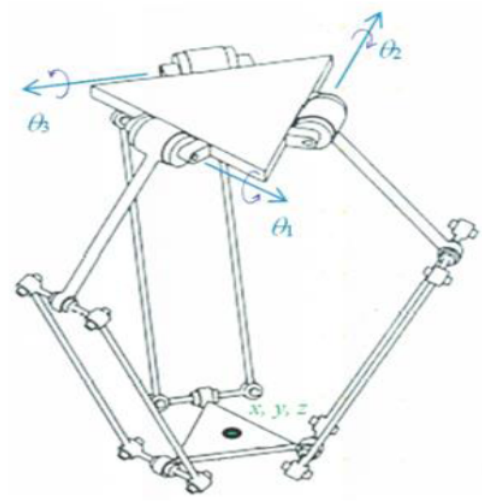


Figure 2.6: Delta Parallel Robot Diagram [elmomc.com/capabilities]

2.3.1 Inverse Kinematics of Delta Robot

The 3-dof Delta Robot inverse position kinematics (IPK) problem is stated: Given the Cartesian position of the moving platform E_0 with coordinates (X_0, Y_0, Z_0) , calculate the three required actuated revolute joint angles $\theta_1, \theta_2, \theta_3$.

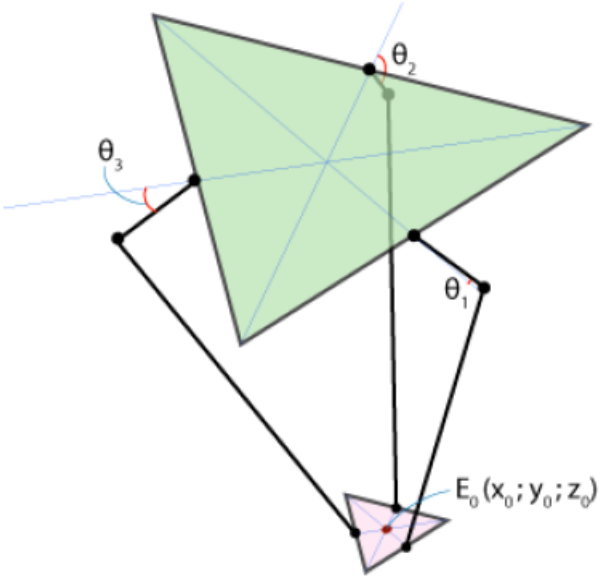


Figure 2.7: The joint angle and end-effector orientation of the Delta Robot [24]

The kinematics of Delta robot can be done either geometrically/trigonometrically or analytically [11]. However, the kinematics solution of the model will be now accomplished geometrically.

First, let's look at the kinematic scheme of delta robot. Basically, the 3-dof Delta Robot (See Fig 2.8) is comprised of a stationary upper platform (1) with three drives fixed in it, and a smaller lower platform (5). Platforms are equilateral-triangle shaped and are joint with three pairs of upper arms (3) and parallel arms (4). Arms (4) form three parallelograms which maintain fixed orientation of the lower and upper platform. Presupposing that the lower platform restricts the robot's workspace, the upper platform remains parallel in relation to the workspace. Arms (4) are fixed in pairs with upper arms (3), which are

moved by drives (2) and transfer movement of the tool in a Cartesian coordinate system of the workspace (6). The reference frame will be chosen with the origin at the center of symmetry of the fixed triangle, so z-coordinate of the end effector will always be negative [25].

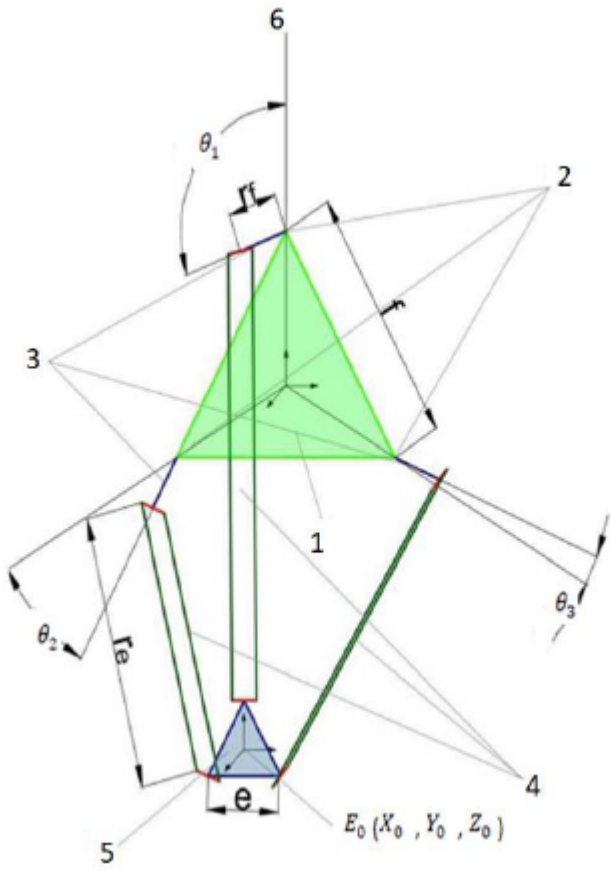


Figure 2.8: Kinematics model of the Delta robot: (1) Fixed platform, (2) drives, (3) the upper arm, (4) lower arms 5) moving platform, and (6) the coordinate system

Robots' arms' (3) circular motion, expressed by ($\theta_1, \theta_2, \theta_3$) angles, leads to the movement of the tool fixed to the lower platform by (x, y, z) in the coordinate system of the workspace. Range of movement motivated by the motion of the arms is strictly associated with the geometrical dimensions of the construction expressed by re, rf, f and e parameters, which denote the following:

rf- length of the upper Leg,
re- distance between the upper arm and the lower platform,
f- length of the upper platform's side,
e- length of the lower platform's side

Stand on the delta robot configuration structure (refer Fig. 2.9) joint F_1J_1 can only rotate in YZ plane, forming circle with center in point F_1 and radius r_f. On the contrary E_1 is universal joint. As a result, E_1J_1 can rotate freely relatively to E_1 , forming sphere with center in point E_1 and radius r_e. The inverse kinematics problem is found at the intersection of this circle and sphere and overall, there are four possible cases [26]:

- Generic solution: the circle penetrates the sphere, resulting in two solutions.
- Singular solution: the circle is tangent to the sphere, resulting in one solution.
- Singular solution: the circle lies on the sphere, producing an infinite number of solutions. This is unlikely since it requires the moving and stationary platforms to occupy the same plane simultaneously.
- No real solution: The circle and the sphere may not intersect at all.

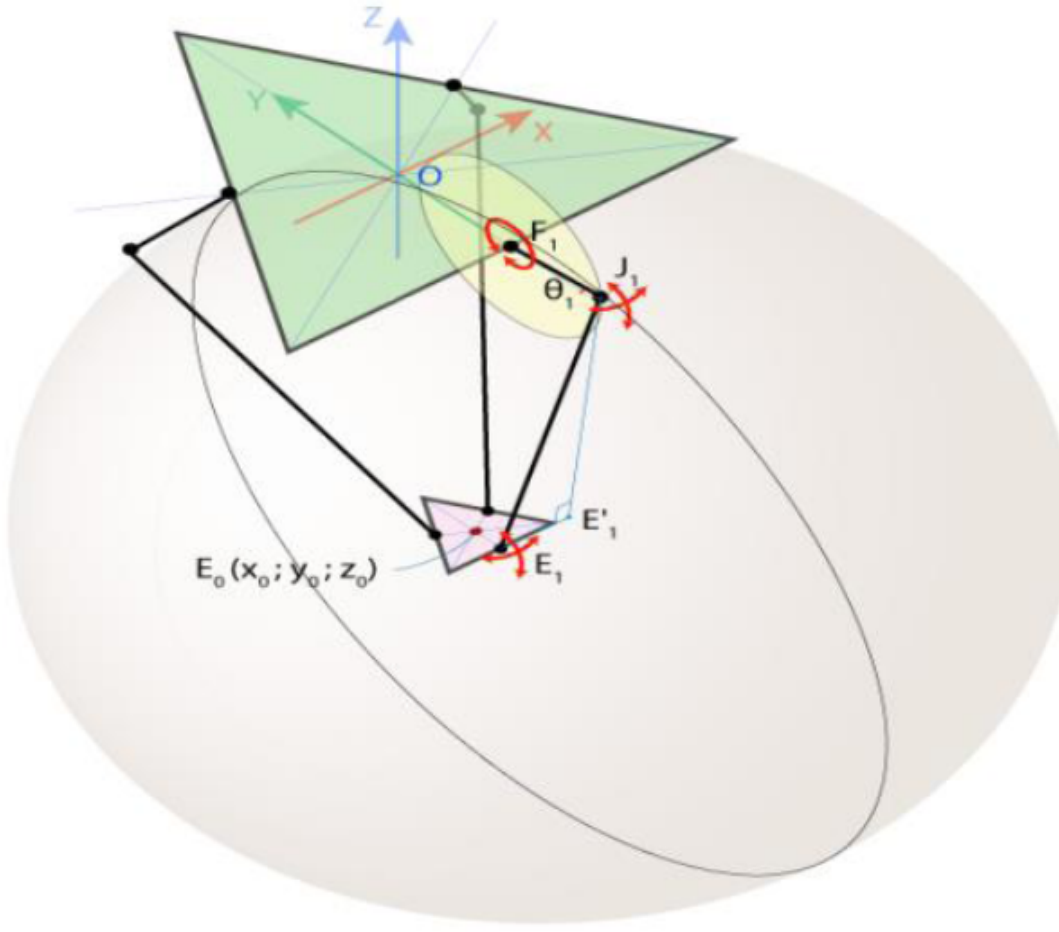


Figure 2.9: Intersection of sphere and circle from the projected lower leg and upper leg [24]

Intersection of this sphere and YZ plane is a circle with center in point $E_1^{'}$ and radius $E_1^{'}J_1$, where $E_1^{'}$ is the projection of the point E_1 on YZ plane. The point J_1 can be found as intersection of two circles of known radius with centers in $E_1^{'}$ and F_1.The intersection point should be chosen with smaller Y-coordinate and if J_1 is known then angle θ_1 [24] can be calculated. To get a clear view for analysis, let's take a part view of the spare and circular intersection.

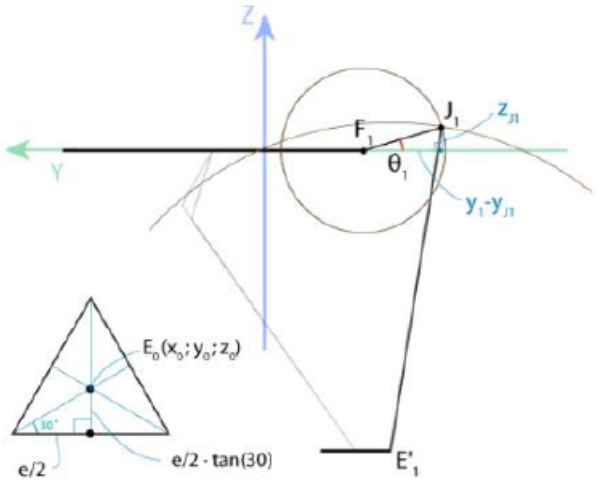


Figure 2.10: YZ-plane & base dimensions

$$E(X_0, Y_0, Z_0) \tag{2.1}$$

$$EE_1 = \frac{e}{2}\tan(30) = \frac{e}{2\sqrt{3}} \tag{2.2}$$

$$E_1\left(X_0, Y_0 - \frac{e}{2\sqrt{3}}, Z_0\right)$$
$$E_1'\left(X_0, Y_0 - \frac{e}{2\sqrt{3}}, Z_0\right)$$

$$EE_1' = X_0 \tag{2.3}$$

$$E_1'J_1 = \sqrt{E_1J_1^2 - E_1E_1'^2} = \sqrt{r_e^2 - X_0^2} \tag{2.4}$$

$$F_1\left(0, \frac{f}{2\sqrt{3}}, 0\right) \tag{2.5}$$

$$\left(y_{J1} - y_{f1}\right)^2 + \left(z_{J1} - z_{f1}\right)^2 = r_f^2 \tag{2.6}$$

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

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

$$\left(y_{J1} - y_0 + \frac{e}{2\sqrt{3}}\right)^2 + \left(z_{J1} - z_0\right)^2 = r_e^2 - X_0^2 \tag{2.7}$$

$$\theta_1 = \arctan \frac{z_{J1}}{\left(y_{f1} - y_{J1}\right)} \tag{2.8}$$

Hence joint F_1J_1 moves only in YZ plane, X coordinate can be completely omitted. In order to find the remaining angels (θ_2 and θ_3), the symmetry of Delta property should be utilized carefully. To simplify mathematical complexity, First, let's rotate coordinate system in XY plane around Z-axis through angle of 120 degrees counterclockwise (see Fig 2.11).

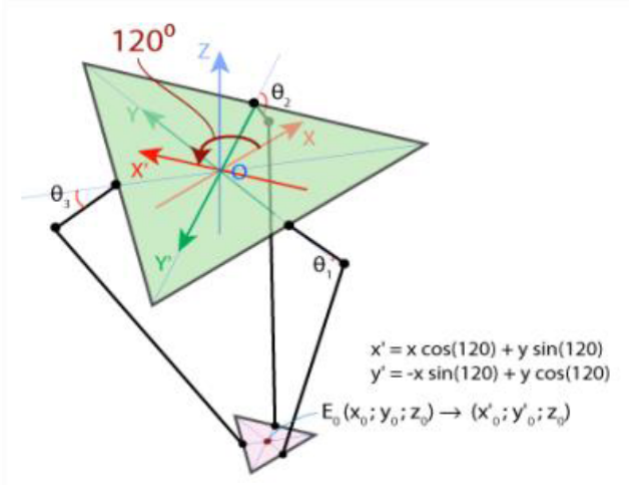


Figure 2.11: Delta Robot coordinate system rotation by angel of 120^0

$X^{'}Y^{'}Z^{'}$ become the new reference frame and thus the same algorithm can be used to find θ_2 as of θ_1. The only parameters to be determined is the new coordinates of E_0, which are $X_0^{'}$ and $Y_0^{'}$. which can be easily done using corresponding rotation matrix. To find angle θ_3, the reference frame should be rotated clockwise [24].

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.9}$$

$$X' = \cos(\alpha) \cdot X - \sin(\alpha) \cdot Y$$
$$Y' = \sin(\alpha) \cdot X + \cos(\alpha) \cdot Y$$

$$Z' = Z \tag{2.10}$$

Where α is the angle of rotation about Z axis. From Eq. (2.8) yields:

$$\theta_{2,3} = f(X', Y', Z') \tag{2.11}$$

Because there are two legitimate options, knee left and knee right, there are two θ_1 solutions, both of which are correct. For each leg of the Delta robot, this results in two Inverse Position Kinematics (IPK) branch solutions, for a total of eight valid solutions. The option with all knees kinked out rather than in is chosen [11].

2.3.2 Forward (Direct) Kinematics of Delta Robot

The 3-dof Delta robot forward position kinematics (FPK) problem is stated: Given the three actuated joint angles $\theta_1, \theta_2, \theta_3$, calculate the resulting Cartesian position of the moving platform control point E_0 with coordinates (X_0, Y_0, Z_0). If angle theta is known then finding the coordinate points of J_1, J_2, and J_3 would be easy.

Joints J_1E_1 , J_2E_2 and J_3E_3 can be freely rotate around points J_1, J_2, and J_3 respectively, forming there spheres with radius r_e .

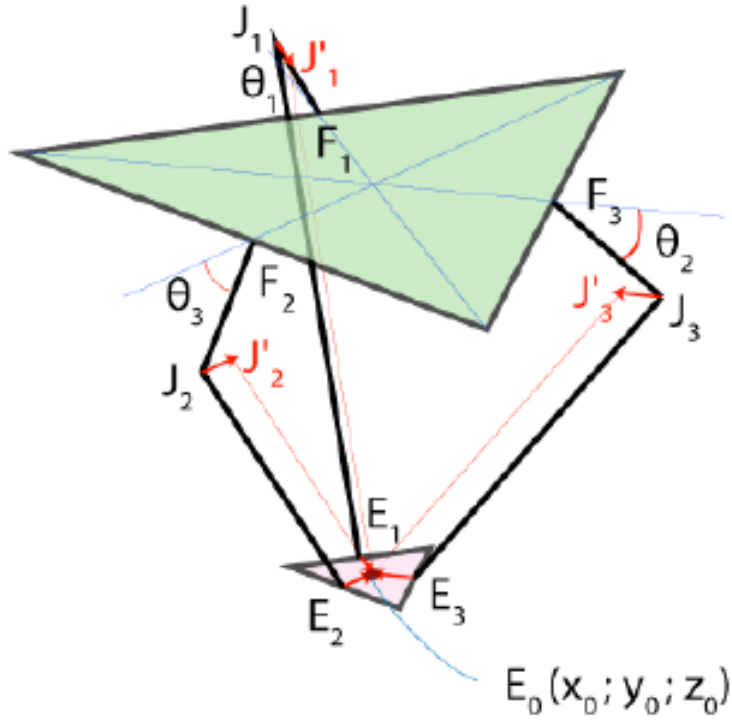


Figure 2.12: Projection of coordinate points in order to form spheres

Now let's do the following: move the center of the spheres from points J_1, J_2, and J_3 to the points $J_1^{'}, J_2^{'}$, and $J_3^{'}$ using transition vector E_1E_0, E_2E_0 and E_3E_0 respectively. After this transition all three spheres intersect in point: E_0 , as it is shown in fig below.

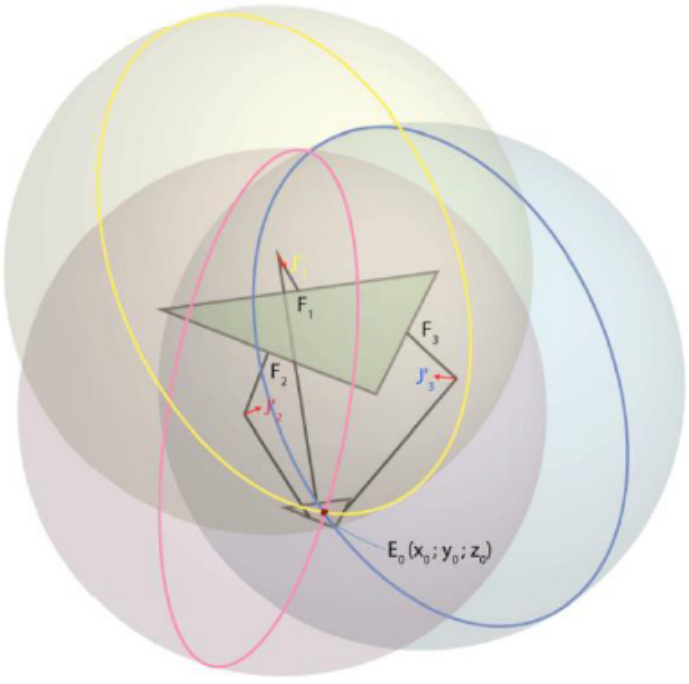


Figure 2.13: Intersection of the three spheres [24]

The intersection of these three spheres represents the solutions to the direct kinematics problem. In the generic case, there are two solutions, since the intersection of two spheres forms a circle, which generally intersected by the third sphere in two locations. Four cases are possible [26]:

1. Generic solution. The two solutions are realized at the intersection of three spheres.

2. Singular solution. One sphere is tangent to the circle of intersection of the other two spheres. Hence there is only one solution possible.

3. Singular solution. The centers of any two spheres coincide, resulting in an infinite number of solutions. This is an unlikely configuration for most embodiments of the manipulator, except for the situation when $\theta_1 = \theta_2 = \theta_3 = \frac{\pi}{2}$ and $e = f$

4. No solution. The three spheres do not intersect.

In order to find coordinates (X_0, Y_0, Z_0) of point E_0 (see Fig 2.13), the three sphere equations stated as follows mustbe solved [24]:

$$(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2 = r_e^2 \tag{2.12}$$

Where coordinates of sphere centers (X_j, Y_j, Z_j) and radius r_e are known. First let's find coordinate points of $J_j^{'}$, $Y_j^{'}$, $Z_j^{'}$. For clear vision let's take a top view of Fig [2.13].

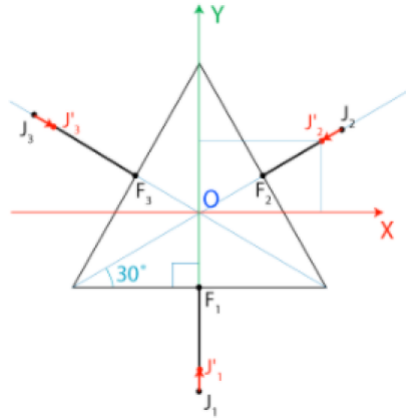


Figure 2.14: Coordinate point projection to the center of end-effector

$$\begin{gathered}
OF_1 = OF_2 = OF_3 = \frac{f}{2}\tan(30) = \frac{f}{2}\tan(30) = \frac{f}{2\sqrt{3}} \\
J_1J_1' = J_2J_2' = J_3J_3' = \frac{e}{2}\tan(30) = \frac{e}{2\sqrt{3}} \\
F_1J_1 = r_f\cos(\theta_1) \\
F_2J_2 = r_f\cos(\theta_2) \\
F_3J_3 = r_f\cos(\theta_3) \\
J_1'\left(0, -\frac{(f-e)}{2\sqrt{3}} + r_f\cos(\theta_1), -r_f\sin(\theta_1)\right)
\end{gathered} \tag{2.13}$$

$$J_2'\left(\left[\frac{f-e)}{2\sqrt{3}}+r_f\cos(\theta_2)\right]\cos(30),\left[\frac{(f-e)}{2\sqrt{3}}+r_f\cos(\theta_2)\right]\sin(30),-r_f\sin(\theta_2)\right) \tag{2.14}$$

$$J_3'\left(\left[\frac{(f-e)}{2\sqrt{3}}+r_f\cos(\theta_3)\right]\cos(30),\left[\frac{(f-e)}{2\sqrt{3}}+r_f\cos(\theta_2)\right]\sin(30),-r_f\sin(\theta_3)\right)$$

Lates design coordinates of points J_1, J_2, J_3 as (X_1, Y_1, Z_1) , (X_2, Y_2, Z_2) and (X_3, Y_3, Z_3). As a result, the equation of the three spheres become as shown below by taking $X_0 = 0$

$$(X)^2+(Y-Y_1)^2+(Z-Z_1)^2=r_e^2 \tag{2.15}$$

$$(X-X_2)^2+(Y-Y_2)^2+(Z-Z_1)^2=r_e^2 \tag{2.16}$$

$$(X-X_3)^2+(Y-Y_3)^2+(Z-Z_3)^2=r_e^2 \tag{2.17}$$

$$X^2+Y^2+Z^2-2Y_1Y-2Z_1Z=r_e^2-Y_1^2-Z_1^2 \tag{2.18}$$

$$X^2+Y^2+Z^2-2Y_2Y-2Z_2Z=r_e^2-X_2^2-Y_2^2-Z_2^2 \tag{2.19}$$

$$X^2+Y^2+Z^2-2Y_3Y-2Z_3Z=r_e^2-X_3^2-Y_3^2-Z_3^2 \tag{2.20}$$

$$W_i=X_i^2+Y_i^2+Z_i^2 \tag{2.21}$$

subtract equation (2.18) from (2.16)

$$X_2X+(Y_1-Y_2)Y+(Z_1-Z_2)Z=\frac{W_1-W_2}{2} \tag{2.22}$$

subtract equation (2.19) from (2.16)

$$X_3X+(Y_1-Y_3)Y+(Z_1-Z_3)Z=\frac{w_1-W_2}{2} \tag{2.23}$$

subtract equation (2.16) from (2.20)

$$(X_2 - X_3)X + (Y_2 - Y_3)Y + (Z_2 - Z_3)Z = \frac{W_1 - W_2}{2} \tag{2.24}$$

$$X = a_1 z + b_1 \tag{2.25}$$

$$Y = a_2 z + b_2$$

$$a_1 = \frac{1}{d}[(Z_2 - Z_1)(Y_3 - Y_1) - (Z_3 - Z_1)(Y_2 - Y_1)]$$

$$a_2 = \frac{1}{d}[(Z_2 - Z_1)X_3 - (Z_3 - Z_1)X_2]$$

$$b_1 = \frac{1}{2d}[(W_2 - W_1)(Y_3 - Y_1) - (W_3 - W_1)(Y_2 - Y_1)]$$

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

$$d = (Y_2 - Y_1)X_3 - (Y_3 - Y_1)X_2)$$

Now substitute equation (2.24) and equation (2.25)

$$\left(a_1^2 + a_2^2 + 1\right)Z^2 + 2(a_1 + a_2(b_2 - y_1) - Z_1)Z + b_1^2 + (b_2 - Y_1)^2 + Z_1^2 + r_e^2 = 0 \tag{2.26}$$

At last, Z_0 can be found from this quadratic equation and thus the smallest negative equation root must be selected. Afterwards, X_0 and Y_0 can be calculated from equation (2.24) and (2.25) [24]

2.3.3 Velocity Kinematics

The forward and inverse position equations are used to relate joint positions and end-effector positions and orientations. Whereas the velocity kinematics relate both the linear velocity and angular velocity of the robot end-effector or any other point on the manipulator to the joint velocity.

Mathematically, the forward kinematic equations define a function between the space of cartesian positions and orientations and the space of joint positions. The velocity relationships are then determined by the Jacobian of this function. The Jacobian is a matrix valued function that can be conceived as the vector version of a scalar function's ordinary derivatives. In the analysis and control of robot motion, one of the most indispensable quantity is the Jacobian or Jacobian matrix. It appears in almost every area of robotic manipulation. For instance, singular configurations determination, path planning and execution of smooth trajectories, in the execution of coordinated anthropomorphic motion, dynamic equation derivation of the robot motion, and in the transformation of forces and torques from the end-effector to the manipulator joints [27].

The Jacobian of delta robot can be analyzed by selecting the significant loop-closer equation of the below figure.

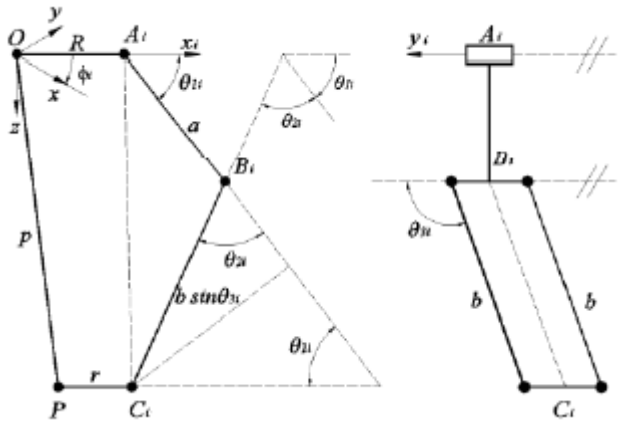


Figure 2.15: Description of the joint angles [28]

Let $\vec{\theta}$ be the vector encompass the actuated joint variables while $\vec{P}$ be the position vector of the moving platform. Then

$$\vec{\theta} = \theta_{1i} = \begin{vmatrix} \theta_{11} \\ \theta_{12} \\ \theta_{13} \end{vmatrix}, \vec{P} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{2.27}$$

By differentiating the appropriate loop closure equation and rearranging the result as illustrated below, the Jacobian matrix can be obtained [29]

$$J_\theta \begin{bmatrix} \dot{\theta_{11}} \\ \dot{\theta_{12}} \\ \dot{\theta_{13}} \end{bmatrix} = J_p \begin{bmatrix} \dot{p_x} = V_x \\ \dot{p_y} = V_y \\ \dot{p_z} = V_z \end{bmatrix} \tag{2.28}$$

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, look at the loop $OA_iB_iC_iP$. The corresponding closure equation in the $x_iy_iz_i$ frame is

$$\overrightarrow{OP} + \overrightarrow{PC_l} = \overrightarrow{OA_l} + \overrightarrow{A_lB_l} + \overrightarrow{B_lC_l} \tag{2.29}$$

The loop closure equation can be written in matrix form as follows

$$\begin{bmatrix} p_x \cos \emptyset_i - p_y \sin \emptyset_i \\ p_x \sin \emptyset_i + p_y \cos \emptyset_i \\ p_z \end{bmatrix} = \begin{bmatrix} R \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} + r_f \begin{bmatrix} \cos \theta_{1i} \\ 0 \\ \sin \theta_{1i} \end{bmatrix} + r_e \begin{bmatrix} \sin \theta_{3i} \cos(\theta_{2i} + \theta_{1i}) \\ \cos \theta_{3i} \\ \sin \theta_{3i} \sin(\theta_{2i} + \theta_{1i}) \end{bmatrix} \tag{2.30}$$

The differentiation of this equation leads to the desired Jacobian equation [29]. Late $a_i = \overrightarrow{A_lB_l}$ and $b_i = \overrightarrow{B_lC_l}$, then equation (2.29) can be written in the form of

$$\overrightarrow{p} + \overrightarrow{e} = \overrightarrow{f} + \overrightarrow{a_l} + \overrightarrow{b_l} \tag{2.31}$$

Taking the time derivative of this equation and furthermore, considering the fact that $\overrightarrow{f}$ is a vector characterizing the fixed platform

$$\underbrace{\dot{\vec{p}} + \dot{\vec{e}}}_{\dot{\vec{p}}} = \dot{\vec{a}_l} + \dot{\vec{b}_l}$$

In this formulation, the velocity of each point on the moving platform is quite the same. Therefore

$$\dot{\vec{p}} = \vec{V} = \dot{\overrightarrow{a_l}} + \dot{\overrightarrow{b_l}} \tag{2.32}$$

Using the well-known identities , the linear velocity on the right part of equation (2.32) can readily be converted into the angular velocity.

$$\vec{v} = \overrightarrow{w_{a_l}} \times \overrightarrow{a_l} + \overrightarrow{w_{b_l}} \times \overrightarrow{b_l} \tag{2.33}$$

Thus, the presence of $\overrightarrow{w_{b_l}}$ introduce dependance upon the variables $\dot{\theta_{2l}}$ and $\dot{\theta_{3l}}$. However, it can be eradicated by taking a scalar product of expression (2.33) with the unit vector $\overrightarrow{b_l}$

$$\widehat{b_l}\left[\vec{v} = \overrightarrow{w_{a_l}} \times \overrightarrow{a_l} + \overrightarrow{w_{b_l}} \times \overrightarrow{b_l}\right]$$

Hence the triple product with two identical vectors is zero, what is left is merely

$$\widehat{b_l} \cdot \vec{v} = \widehat{b_l} \cdot \overrightarrow{w_{a_l}} \times \overrightarrow{a_l} \tag{2.34}$$

In the component form, the left hand side of this equation can be written as

$$\begin{aligned} \widehat{b_l} \cdot \vec{v} = [\sin\theta_{3i}\cos(\theta_{2i}+\theta_{1i})]\left[V_x\cos\emptyset_i - V_y\sin\emptyset_i\right] + \\ \cos\theta_{3i}\left[V_x\sin\emptyset_i - V_y\cos\emptyset_i\right] + \\ [\sin\theta_{3i}\sin(\theta_{2i}+\theta_{1i})V_z] = J_{ix}V_x + J_{iy}V_y + J_{iz}V_z \end{aligned} \tag{2.35}$$

Where,

$$\begin{aligned} J_{ix} &= \sin\theta_{3i}\cos(\theta_{2i}+\theta_{1i})\sin\emptyset_i + \cos\theta_{3i}\sin\emptyset_i \\ J_{iy} &= \sin\theta_{3i}\cos(\theta_{2i}+\theta_{1i})\sin\emptyset_i + \cos\theta_{3i}\cos\emptyset_i \end{aligned}$$

$$J_{iz} = \sin\theta_{3i}\sin(\theta_{2i}+\theta_{1i}) \tag{2.36}$$

The movement of the joint a is in the $x_i z_i$-plane. Thus, it only has a component of velocity in y axis.

$$\overrightarrow{w_{a_l}} = \begin{bmatrix} 0 \\ -\dot{\theta}_{1l} \\ 0 \end{bmatrix} \tag{2.37}$$

$$\overrightarrow{w_{a_l}} \times \overrightarrow{a_l} = \begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ 0 & -\dot{\theta}_{1l} & 0 \\ a_{1i} & a_{2i} & a_{3i} \end{vmatrix} = -a_{3i}\dot{\theta_{1l}}\hat{\imath} + a_{1i}\dot{\theta_{1l}}\hat{k}$$

Equation 2.34 can be simplified now as follows

$$\widehat{b_l} \cdot \left(\overrightarrow{w_{a_l}} \times \overrightarrow{a_l}\right) = -a\sin\theta_{2i}\sin\theta_{3i}\dot{\theta_{1l}} \tag{2.38}$$

Every value of i can be evaluated by equating eq (2.37) and (2.38)

$$J_{1x}V_x + J_{1y}V_y + J_{1z}V_z = -a \sin\theta_{21} \sin\theta_{31}\dot{\theta_{11}}$$
$$J_{2x}V_x + J_{2y}V_y + J_{2z}V_z = -a \sin\theta_{22} \sin\theta_{32}\dot{\theta_{12}}$$
$$J_{3x}V_x + J_{3y}V_y + J_{3z}V_z = -a \sin\theta_{23} \sin\theta_{33}\dot{\theta_{13}}$$

Which readily implies

$$J_p\vec{V} = J_\theta\dot{\vec{\theta}} \tag{2.39}$$

Where,

$$J_p = \begin{bmatrix} J_{1x} & J_{1y} & J_{1z} \\ J_{2x} & J_{2y} & J_{2z} \\ J_{3x} & J_{3y} & J_{3z} \end{bmatrix} \tag{2.40}$$

$$J_\theta = a \times \begin{bmatrix} \sin\theta_{21} \sin\theta_{31} & 0 & 0 \\ 0 & \sin\theta_{22} \sin\theta_{32} & 0 \\ 0 & 0 & \sin\theta_{23} \sin\theta_{33} \end{bmatrix} \tag{2.41}$$

$$J = J_p^{-1}J_\theta \tag{2.42}$$

$$V = J\dot{\theta} \tag{2.43}$$

2.4 Singularities

The manipulator's singularities are revealed by finding across conditions that yield singular Jacobian matrices. The tow part Jacobian aids in the classification of these singularities [29].

2.4.1 Inverse kinematics singularities

Inverse kinematics singularities are linked to the inverse Jacobian and occur when

$$det\ (J_\theta) = 0.$$

This happens at the boundary of the workspace

1. When the limb α_i is in the plane of the parallelogram formed by limb b_i for any of the for any of the i $\theta_{2i} = 0$ or π for any of the i

2. When any of the limbs b_i is parallel (or anti-parallel) to y-axis. As α_i is in the xz-plane, it means that in this configuration, $\alpha_i \perp b_i$. $\theta_{3i} = 0$ or π any of the i.

2.4.2 Direct kinematics singularities

The direct kinematics singularities are related to the singularities of the direct Jacobian given in equation (2.32). Recalling that the determinant of a Jacobian vanishes when any row or any column is identically zero.

1. When the third J_p column of is zero. Physically, the posture of the robot corresponds to when all the limbs bi are in the plane of the moving platform and in fact lie entirely along y_i-axes.

$$\theta_{3i} = 0 \text{ or } \pi \ \forall i$$

2. When the third column of J_p is zero. Physically, the posture of the robot again corresponds to when all the limbs bi are in the plane of the moving platform. However, they can have both x_i and y_i components non-zero, where $\theta_{2i} + \theta_{1i} = \pi$. This situation is depicted by a virtual horizontal plane as the actual relative lengths of $a_i i$ and $b_i i$ prevent reaching that singular configuration.

$$\theta_{2i} + \theta_{1i} = 0 \text{ or } \pi \forall i$$

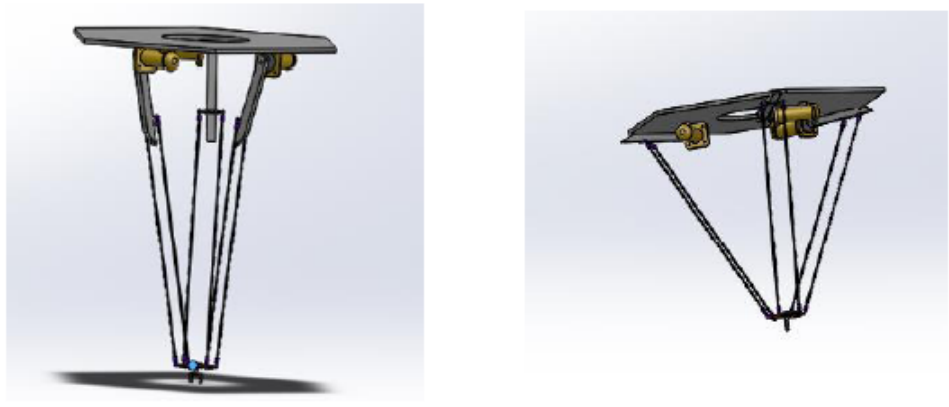


Figure 2.16: Depicting condition (a) Fully stretched robot posture (b) Fully contracted configuration of the robot

2.5 Dynamic Analysis

Dynamics has been applied in many fields such as spacecraft dynamics, machine design and robotics. Dynamic modelling is very important for the design, analysis, simulation and control of robots. Robot dynamic deals with the mathematical formulation of the equations of robot arm motion. These equations describe the dynamic behavior of the manipulator. Profoundly, it concerned with the relationship between the forces acting on a robot mechanism and the accelerations they produce. The robot mechanism is modelled as a rigid-body system, in which case robot dynamics is the application of rigid-body dynamics to robots. The two main problems in robot dynamics are:

- Forward dynamics problem: given the forces, work out the accelerations.
- Inverse dynamics problem: given the accelerations, work out the forces.

Forward dynamics is also known as "direct dynamics," or sometimes simply as "dynamics." It is mainly used for simulation. Inverse dynamics has various uses, including on-line control of robot motions and forces, trajectory design and optimization, design of robot mechanisms, and as a component in some forward-dynamics algorithms.

Hence the inverse dynamics problem is to find the actuator torques and/or forces required to generate a desired trajectory of the manipulator, it is essentially required to meet the objective of these thesis. The inverse dynamics modeling of the 3DOF of delta robot is performed based upon the principle of virtual work.

According to [28], Without losing generality of model, the dynamic problem can be simplified by the following hypotheses. The connecting rods of lower links can be built with light materials such as the aluminum alloy, so

- The lower links rotational inertial are neglected
- The mass of each lower links, is divided evenly and concentrated at the two end points of the parallelogram.
- The friction forces in joints are neglected.
- No external forces suffered.

$\tau = [\tau_1\ \tau_2\ \tau_3]^T$ and $\delta q = [\delta\theta_1\ \delta\theta_2\ \delta\theta_3]^T$ are the vectors of torque and the corresponding virtual angular displacement vector. Moreover, $\delta p = [\delta\theta_x\ \delta\theta_y\ \delta\theta_z]^T$ represent the virtual angular displacement vector of the moving platform. The dynamic equations are going to be derived by applying virtual wok principle.

$$\tau^T \delta\theta + M_{Ga}^T \delta\theta + F_{Gp}^T \delta p - M_a^T \delta\theta - F_p^T \delta p = 0 \tag{2.44}$$

Where,

$$M_{Ga} = \left(\frac{1}{2}M_a + M_b\right) \cdot g \cdot l_a \cdot I[\cos(\theta_1) + \cos(\theta_2) + \cos(\theta_3)]^T$$

Is the upper link gravity torques vector M_a and M_b are mass of the upper link and each connecting rode of the lower link respectively. Here g denotes the gravity acceleration and I represent the 3×3 identity matrix.

$$F_{Gp} = \left[00 - \left(M_{tcp} + 3M_b\right)\right]^T \tag{2.45}$$

Denote the mobile platform gravity force vector, and M_{tcp} is mass of the mobile platform.

$$M_a = I_a\ddot{\theta} = I_a\left[\ddot{\theta}_1\dot{\theta}_2\ddot{\theta}_3\right]^T \tag{2.46}$$

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 link inertia matrix with respect to the fixed frame $O\{x, y, z\}$ and,

$$F_p = M_p\ddot{p} = \left(M_{tcp} + 3M_b\right).I.[\ddot{X}\ddot{Y}\ddot{Z}]^T \tag{2.47}$$

Denote the mobile platform inertia forces vector. equation (2.34) can be rewritten to

$$\dot{P} = J^{-1}\dot{\theta}$$

$$\delta p = J^{-1}\delta\theta \tag{2.48}$$

Substituting equation 2.43 into equation 2.33 results,

$$\left(\tau^T + M_{Ga}^T + F_{Gp}^T J^{-1} - M_a^T - F_p^T J^{-1}\right)\delta\theta = 0 \tag{2.49}$$

For any virtual displacement $\delta\theta$,

$$\tau = M_a + J^{-T}F_p - M_{Ga} - J^{-T}F_{Gp} \tag{2.50}$$

Hence the general dynamics equation for any manipulator can be stated as

$$\tau = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta})\dot{\theta} + G(\theta) \tag{2.51}$$

Substituting equation2.26 and 2.28 into equation 2.36 yields

$$\tau = I_a\ddot{\theta} + J^{-T}M_p\ddot{P} - M_{Ga} - J^{-T}F_{Gp} \tag{2.52}$$

Time derivative of equation 2.39 yields

$$\ddot{P} = J^{-1}\ddot{\theta} + \dot{J^{-1}}\dot{\theta} \tag{2.53}$$

Then the general dynamic equation can be derived by substituting eq.2.45 into eq.2.44

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) \tag{2.54}$$

The below equation represents the dynamic model of parallel manipulator in joint space. Here, $\theta \in R^3$ is the controlled variable and

$$M(\theta) = I_a + J^{-T} M_p J^{-1} \tag{2.55}$$

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

$$C(\theta, \dot{\theta}) = J^{-T} M_p J^{-1} \tag{2.56}$$

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

$$G(\theta) = -M_{Ga} - J^{-T} F_{Gp} \tag{2.57}$$

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

2.5.1 Non-Rigid Body Effects

It is important to realize that the derived dynamics equations 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 affected by frictional forces. Now days manipulators, in which significant gearing is typical, the forces due to frictional can be actually be quite large, perhaps equaling 25% of the torque required to move the manipulator in typical situations. In order to make dynamic equation reflect the reality of the physical device, it is important to include at least the approximation of 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 [28].

$$\tau_{\text{friction}} = V\dot{\theta} \tag{2.58}$$

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

$$\tau_{\text{friction}} = \text{csgn}(\dot{\theta}) \tag{2.59}$$

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:

$$\tau_{\text{friction}} = V\dot{\theta} + \text{csgn}(\dot{\theta}) \tag{2.60}$$

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

$$\tau_{friction} = f(\theta, \dot{\theta}) \tag{2.61}$$

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

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) + F(\theta, \dot{\theta}) \tag{2.62}$$

2.5.2 Actuator Dynamics

The active leg of the parallel manipulator is basically composed of DC motor, precision revolute bearing and coupling elements. The model of dc motor is given below

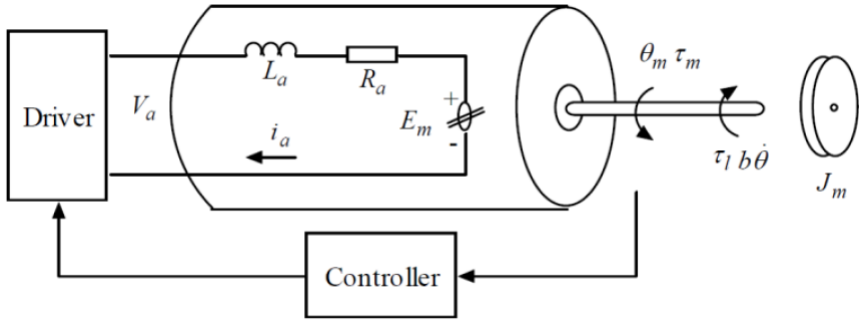


Figure 2.17: DC motor model [23]

The symbols represent the following variables. θ_m is the motor position in rad, τ_m is the produced torque by the motor in Nm, τ_1 is the load torque, V_a is the armature voltage (V), L_a is the armature inductance in H, R_a is the armature resistance in Ω, E_m is the reverse EMF(V), I_a is the armature current (A), K_b is the reverse EMF constant and K_m is the torque constant [28].

$$L\frac{di_a}{dt} + R_a i_a = V_a - E_m \tag{2.63}$$

$$E_m = K_b \frac{d\theta_m}{dt}$$

$$\tau_m = K_m i_a$$

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

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,

$$\theta_m = K_r \theta_l \tag{2.64}$$

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. considering the above equation, we can write the motor torque as follows

$$\tau_m = \frac{\tau_1}{K_r} \tag{2.65}$$

In order to simulate the motion of a manipulator, the dynamic mode of the given manipulator should be utilized. simulation requires solving the dynamic equation for acceleration as follows

$$\ddot{\theta} = M^{-1}(\theta)[\tau - C(\theta, \dot{\theta})\dot{\theta} - G(\theta) - F(\theta, \dot{\theta}) \tag{2.66}$$

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

$$\theta(0) = \theta_0 \tag{2.67}$$

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

3 Control System Design

3.1 Passivity Based Control

In recent decades, many results have been reported about dealing with control problems from the perspective of energy transformation, for which passivity-based control have been widely adopted [12]. Passivity based control is a methodology which consists in controlling a system with the aim at making the closed loop system passive and it effectively exploit the structure of the considered systems and physical knowledge and thus provide physical interpretations to the control actions. To better understand the passivity concept and passivity-based control (PBC), we need to leave behind the notion of state of a system and think of the latter as a device which interacts with its environment by transforming inputs into outputs. From an energetic viewpoint we can define a passive system as a system which cannot store more energy than is supplied by some source with the difference between stored energy and supplied energy being the dissipated energy [13, 30].

A fundamental property of passive systems is that, regarding a feedback interconnection of other physical passive systems, passivity is invariant under negative feedback interconnection. In other words, the feedback interconnection of two passive systems yields a passive system. Conversely, passive systems can be decomposed into passive subsystems. Thus, in this philosophy the controller can be designed as a passive system [12–17, 30].

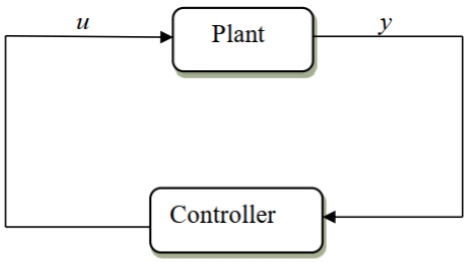


Figure 3.1: Feedback interconnection of plant and controller

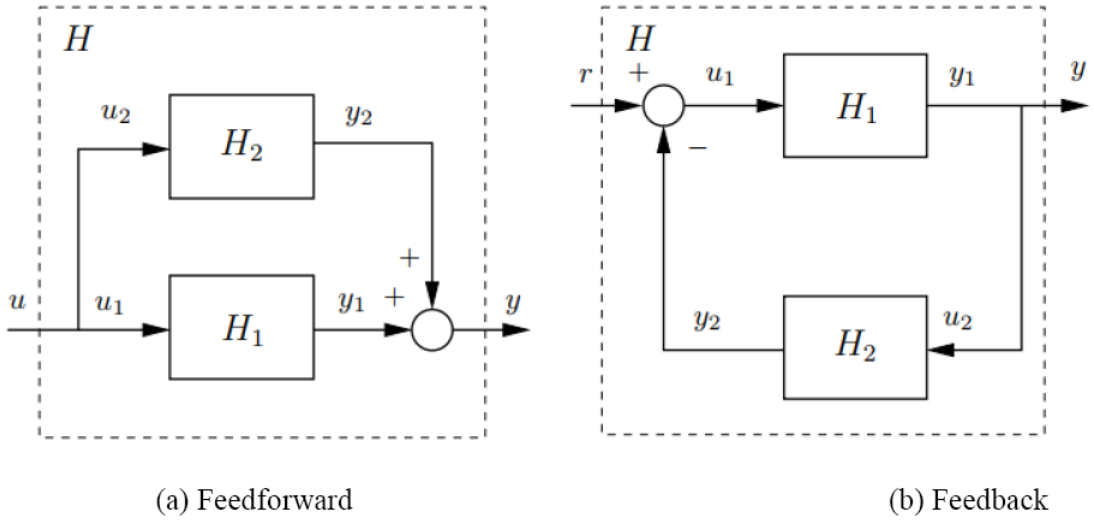


Figure 3.2: Interconnections of passive systems [14]

The controller which is going to be designed here with will be based on the passivity theorem, input output stability approach.

The map $u \longrightarrow y$ is passive if there exists a state function H(x), bounded from below, and a non-negative function $d(t) \geq 0$ such that

$$\int_0^t u^T(s)y(s)ds = H(x(t)) - H(x(0)) + d(t) \tag{3.1}$$

The term on the left side is the energy supplied to the system, the first and the second term on the right side represent the stored energy and dissipated energy respectively [14].

The passivity property described in equation (3.1) is sufficient to solve regulation tasks in mechanical systems, where the PBC only needs to modify the potential energy and the dissipation function. However, to study tracking problems or treat electrical or electro-mechanical systems, we need a stronger property. In this case a desired behavior should be imposed, not only on generalized coordinate (q), but on derivative of generalized coordinate ($\dot{q}$) as well, which in its turn translates into the need for modifying the kinetic energy. The PBC approach may be viewed as an extension of the well-known energy-shaping plus

damping injection technique [14, 30]. Mainly the designing process has two main steps. The first step is energy shaping, where a controller is designed such that the closed-loop system matches a desired energy function that similar to the natural energy function of the open-loop system. In this stage, passivity property of the robot system can be preserved in the closed loop system. The second step is damping injection, where velocity feedback is employed to introduce damping in the closed-loop system, to ensure stability. With the assumption that the robot dynamic is known, the well-known passivity-based controller proposed by Slotine and Li (1987) can be applied [16].

Passivity based controller is used for achieving stability of nonlinear systems by making the system passive, because passive systems are by nature asymptotically stable systems. Since it strongly exploits the system properties, it does not produce unnecessarily large control effort and has inherent robustness against plant uncertainty and disturbance. Thus, a PBC is one of most widely-used controllers for robot manipulators [16]. Refer Appendix A of [14] for detail proof of the asymptotic stability of passive system.

3.2 Passivity Property of the Robot Dynamic System

A robot nominal dynamic model in the world coordinate system is given as follows [16]:

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) \tag{3.2}$$

Where q, $\dot{q}$, $\ddot{q}$ are n-dimensional vectors representing joint displacements, velocity and acceleration respectively $M(q) \in R^{nxn}$ is the inertia matrix, $C(q,\dot{q})\dot{q} \in R^n$ is the Coriolis and centrifugal forces, $G(q) \in R^n$ is the vector of gravity forces and $\tau \in R^n$ is the vector of torque input. The dynamic described by Eq (3.2) has the following property (see [10, 16, 17, 28, 29]).

Property 1: The inertia matrix is symmetric and positive definite for all q

$$M(q) = M(q)^T > 0 \tag{3.3}$$

Also, for some positive constants a and b, with $b > a$, it is satisfied

$$a \leq \|M(q)\| \leq b \quad \forall q \tag{3.4}$$

Property 2: The inertia matrix and the Coriolis matrix satisfy

$$\dot{q}^T \left[\frac{1}{2} \dot{M}(q) - C(q, \dot{q}) \right] \dot{q} = 0 \quad \forall q, \dot{q} \tag{3.5}$$

Property 3: the Coriolis and centripetal force vector satisfies

$$\| C(q, \dot{q}) \dot{q} \| \leq c \| \dot{q} \|^2 \quad \forall q, \dot{q} \tag{3.6}$$

For some positive constant c.

Property 4: The gravity is obtained as the gradient of the robot potential energy $U(q)$.

$$G(q) = \partial U(q) / \partial q)^T \tag{3.7}$$

Also, for some positive constant d, it is satisfied

$$\| G(q) \| \leq d \quad \forall q \tag{3.8}$$

Note that a direct differential of the total energy of the robot

$$H(q, \dot{q}) = \frac{1}{2} \dot{q}^T M(q) \dot{q} + U(q) \tag{3.9}$$

Along (3.2) and after using property 2 and 4, yields

$$\dot{H}(q, \dot{q}) = \dot{q}^T \tau \tag{3.10}$$

This expression indicate that the increase of energy in the system is equal to the delivered work. The input-output map from generalized torques to generalized velocities, $\tau \rightarrow \dot{q}$. Thus, the system is passive. This passivity property will be useful in the proposed controller design.

3.3 Controller Goal and Derivation of The Control Law

The controller goal is to track the desired smooth trajectory q_d, where $q_d \in R^n$ and define the tracking errors as

$$\tilde{q} = q - q_d, \quad \dot{\tilde{q}} = \dot{q} - \dot{q}_d$$

The problem is to find a model-based controller τ such that the following objectives be achieved

$$\lim_{t\to\infty} \tilde{q}_i = 0$$
$$\lim_{t\to\infty} \dot{\tilde{q}}_i = 0$$

The first step is to reshape the robot system's natural energy such that the tracking control objective is achieved and followed by damping in order to achieve stable trajectory tracking.

Tracking control objective $q(t) \rightarrow q_d(t)$

Shifting energy minimum at $(q, \dot{q}) = (0, 0)$ to $(\tilde{q}, \dot{\tilde{q}}) = (0, 0)$, $\tilde{q} = q - q_d$

Energy storage function with coordinate $(e, \dot{e})$ is given as follows

$$V = \frac{1}{2}\dot{\tilde{q}}^T M(q)\dot{\tilde{q}} + \frac{1}{2}\tilde{q}^T K_p \tilde{q} \tag{3.11}$$

$$\dot{V} = \dot{\tilde{q}}^T M(q)\ddot{\tilde{q}} + \frac{1}{2}\dot{\tilde{q}}^T \dot{M}(q)\dot{\tilde{q}} + \tilde{q}^T K_p \dot{\tilde{q}} \tag{3.12}$$

$$= \dot{\tilde{q}}^T (M\ddot{q} - M\ddot{q}_d) + \frac{1}{2}\dot{\tilde{q}}^T \dot{M}\dot{\tilde{q}} + \tilde{q}^T K_p \dot{\tilde{q}} \tag{3.13}$$

$$= \dot{q}^T \left(\tau - C\dot{q} - g - M\ddot{q}_d + \frac{1}{2}\dot{M}\dot{\tilde{q}} + K_p \tilde{q} \right) \tag{3.14}$$

$$\dot{V} = \tilde{q}^T \left(\tau + \frac{1}{2}(\dot{M} - 2C)\tilde{q} - M\ddot{q}_d - C\dot{q}_d - g + K_p \tilde{q} \right) \tag{3.15}$$

$\dot{M} - 2C = 0$, hence it is square symmetric and the above equation can be reduced to:

$$= \dot{\tilde{q}}^T \left(\tau - M\ddot{q}_d - C\dot{q}_d - g + K_p\tilde{q}\right) \tag{3.16}$$

Let's define the control law as:

$$\tau = M\ddot{q}_d + C\dot{q}_d + g - K_p\tilde{q} + v \tag{3.17}$$

Insert equation (3.17) into equation (3.16)

$$\dot{V} = \dot{\tilde{q}}^T v \tag{3.18}$$

let's set control input as $v = -K_d\dot{\tilde{q}}$, then $\dot{V} = -K_d\|\dot{\tilde{q}}\|^2 \leq 0$, Globally stable.
Thus, the control law becomes

$$\tau = M(q)\ddot{q}_d + C(q,\dot{q})\dot{q}_d + g(q) - K_p\tilde{q} - K_d\dot{\tilde{q}} \tag{3.19}$$

Looking into the control equation in (3.19), $\dot{V} = -K_d\|\dot{\tilde{q}}\|^2$ tell us $\dot{\tilde{q}} \to 0$ as $\dot{V} \to 0$ But it does not guarantee $\tilde{q} \to 0$. In order to solve this problem lets restrict them to lie on a sliding surface $s = \dot{\tilde{q}} + \Lambda\tilde{q} = 0$ and energy storage function with coordinate s given as:

$$V = \frac{1}{2}S^T MS \tag{3.20}$$

$$\dot{V} = S^T M\dot{S} + \frac{1}{2}S^T \dot{M}S$$

$$= S^T\left(M\ddot{\tilde{q}} + M\Lambda\dot{\tilde{q}} + \frac{1}{2}\dot{M}S\right)$$

$$= S^T\left(\tau - C\dot{q} - g - M\ddot{q}_d + M\Lambda\dot{\tilde{q}} + \frac{1}{2}\dot{M}S\right)$$

$$= S^T\left(\tau - M\ddot{q}_d + M\Lambda\dot{e} - C\dot{q}_d + C\Lambda\tilde{q} - g + \frac{1}{2}(\dot{M} - 2C)S\right) \tag{3.21}$$

Introduce virtual "reference trajectory", $\dot{q}_r = \dot{q}_d - \Lambda\tilde{q}$

$$\dot{V} = S^T(\tau - M\ddot{q}_r - C\dot{q}_r - g) \tag{3.22}$$

Same as before, the control law can be defined as,

$$\tau = M\ddot{q}_r + C\dot{q}_r + g + v \tag{3.23}$$

Eq (3.23) ⟶ eq (3.22)

$$\dot{V} = \dot{S}^T v \tag{3.24}$$

As like the previous, new control input can be set as $v = -K_d S$, then $\dot{V} = -K_d \|S\|^2 \leq 0$, Thus globally stable and guarantee $(\tilde{q}, \dot{\tilde{q}}) \rightarrow (0, 0)$

As a result, the final control law is

$$\tau = M(q)\ddot{q}_r + C(q, \dot{q})\dot{q}_r + g(q) - K_d S \tag{3.25}$$

Were, $\dot{q}_r = \dot{\tilde{q}}_d - \Lambda\tilde{q}, S = \dot{\tilde{q}} - \Lambda\tilde{q}$ and $\Lambda > 0, K_d > 0$ are diagonal matrices

3.4 Robustness of the Passivity Based Control (PBC)

Rigid-body dynamics with disturbance

$$\tau + d = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) \tag{3.26}$$

Passivity-based control input from equation (3.25)

$$\tau = M(q)\ddot{q}_r + C(q, \dot{q})\dot{q}_r + g(q) - K_d S$$

$$\dot{q}_r = \dot{\tilde{q}}_d - \Lambda\tilde{q}, S = \dot{\tilde{q}} - \Lambda\tilde{q}$$

Then, closed-loop system is,

$$d = M(q)\dot{S} + C(q, \dot{q})S + K_d S \tag{3.27}$$

For input-output pair (s, d)

$$S^T d = S^T M(q)\dot{S} + S^T C(q, \dot{q})S + S^T K_d S \tag{3.28}$$

$$= \frac{d}{dt}\left\{\frac{1}{2}S^T M(q)S\right\} - \frac{1}{2}S^T \dot{M}(q)S + S^T C(q,\dot{q})S + S^T K_d S$$
$$= \frac{d}{dt}\left\{\frac{1}{2}S^T M(q)S\right\} + S^T K_d S$$

$\frac{d}{dt}\left\{\frac{1}{2}S^T M(q)S\right\}$ is the stored energy and $S^T K_d S$ refers the dissipated energy which is greater than zero. Therefor, input-output pair (S, d) system is passive and always stable for any disturbance d. Hence, $S^T d \leq \frac{d}{dt}\left\{\frac{1}{2}S^T M(q)S\right\}$

4 Simulation Result and Discussion

4.1 Modeling Multi-body Systems

A 3D model of mechanical Robot can be developed using different modeling environment but after analyzing the cons and pros of different tools, CAD(solidworks) has been selected to model the 3-DOF Delta Robot. SolidWorks models can be exported and 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 and to perform various experiments and to obtain required outputs: position (angles), forces and torques without any necessity of motion equations derivation and Simscape physical modeling of the mechanical system.

The CAD export procedure generates one XML multibody description file, XML is a suitable file format to export a model from any CAD system or modeling environment to Simscape Multibody, and a set of geometry files. The XML file contains the structure of the assembly and the parameters that define each part. The geometry files define the 3-D geometry of each part. Once the export procedure is complete, the XML multibody description file can be imported into Simscape Multibody software. Simscape Multibody uses the file to automatically generate a new Simscape Multibody model. In order to do so, Simscape Multibody plugin must be installed first based on the Matlab/Simscape Multibody version used. This project has been done on Matlab R2019b.

The Simscape Multibody Link plug-in provides the primary interface for exporting CAD assemblies into Simscape Multibody software. The plug-in is compatible with three CAD applications: Autodesk Inventor, Creo Parametric, and SolidWorks. There are few steps that should be done to enable the Simscape Multibody plug-in. The very first step is to get the installation files which match with Matlab release number and system architecture from Simscape Multibody Link download page. Then the saved installation file must be run as administrator. Each time when the CAD assembly model is exported, the Simscape Multibody link plug-in attempt to connect to Matlab. Therefore, the third step will be to

register Matlab as an Automation Server. Finally the model assembly can be exported once the Simscape Multibody Link Plugin enabled on the CAD(SolidWorks application).

4.2 DELTA Robot CAD Modelling

The Simscape Multibody import XML Schema enables to import models into Matlab/Simscape from any CAD system or modeling environment that exports an XML file that follows this schema. Solidworks modeling process includes designing each and every part of the robot separately at first with appropriate selection of mass property, material type and dimension. And then assembled into subassembly and finally into one general assembly. Throughout the assembly processes special emphasis should be given for every mates or constraints adopted between each following parts. Hence later on they become joins of the Simscape Multibody model and determine degree of freedom of each part. The forgoing picture shows the A3-D, 3-DOF Delta Robot assembly designed with the help of 3D-CAD SolidWorks 2019 programming.

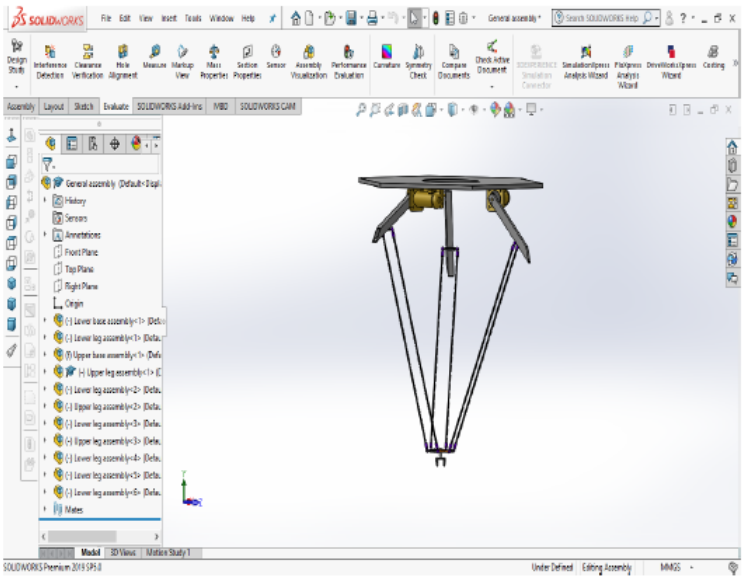


Figure 4.1: Solidworks 3-D model assembly of 3-DOF Delta Robot

The values of 3-DOF Delta Robot CAD model has been taken from a specific commercial Delta Robot, ABB FlexPicker IRB 360- 1/1600. For the purpose of speed and flexibility, light material has been chosen for this paper. The following table shows the design parameters and dedicated symbols used for each part of the robot.

No	Parts of the 3-DOF Delta Robot	Symbols used for the parts	Dimension (mm)
I	Upper Leg	r_f	300
II	Lower leg	r_e	800
III	Base equilateral triangle side	f	840
IV	Platform equilateral triangle side	e	190

Table 4.1: Delta Robot design parameters
[new.abb.com/products/robotics/industrial-robots/irb-360]

4.3 Delta Robot Simscape Multibody Model

Simscape Multibody provides a multibody simulation environment for 3D mechanical systems, such as robots, vehicle suspensions, construction equipment, and aircraft landing gear. Simscape Multibody and Simulink software helps to create control systems and assess system-level performance by allowing users to construct Multibody systems using blocks that represent constraints, bodies, force elements, joints and sensors. Beside Simscape Multibody Link utility incorporates the library to import a complete CAD assembly, including all masses, inertias, joints, constraints, and 3D geometry.

The smimport function should be used to import a CAD model into Simscape Multibody. The method parses an XML multibody description file and builds a model for it automatically. The body geometries are provided in geometry files that accompany the multibody description file format for visualization reasons. The function creates a new Simscape Multibody block diagram as well as a data file to go with it. The block diagram uses Simscape Multibody blocks to reproduce the original CAD assembly model. The data file contains the numerical values for the model's block parameters.

Basically, there are two steps to be caried out to create Simscape Multibody model from a CAD assembly. The first translation step is to use the Simscape Multibody Link exporter to create an intermediate Physical Modeling XML file from a CAD assembly. Using Simscape Multibody software, we can then import that XML file to automatically generate a Simscape Multibody model. The second stage is to import the Physical Modeling XML in order to build the Simscape Multibody model, which will then be used in conjunction with geometry graphics files of the body to simulate and visualize the original mechanical system.

In a Simscape Multibody model, the XML representations of parts and constraints become bodies and joints, and the resulting Simscape model visualizes the bodies using the exported body geometry graphics files. Fig.4.2 shows Simscape Multibody model of the Delta robot and its 3D visual using Matlab Mechanics Explore.

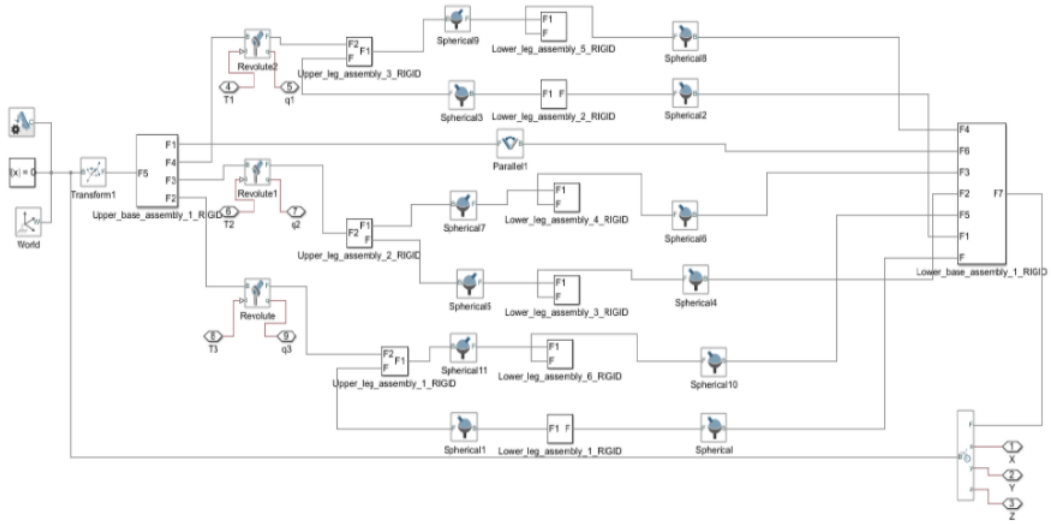
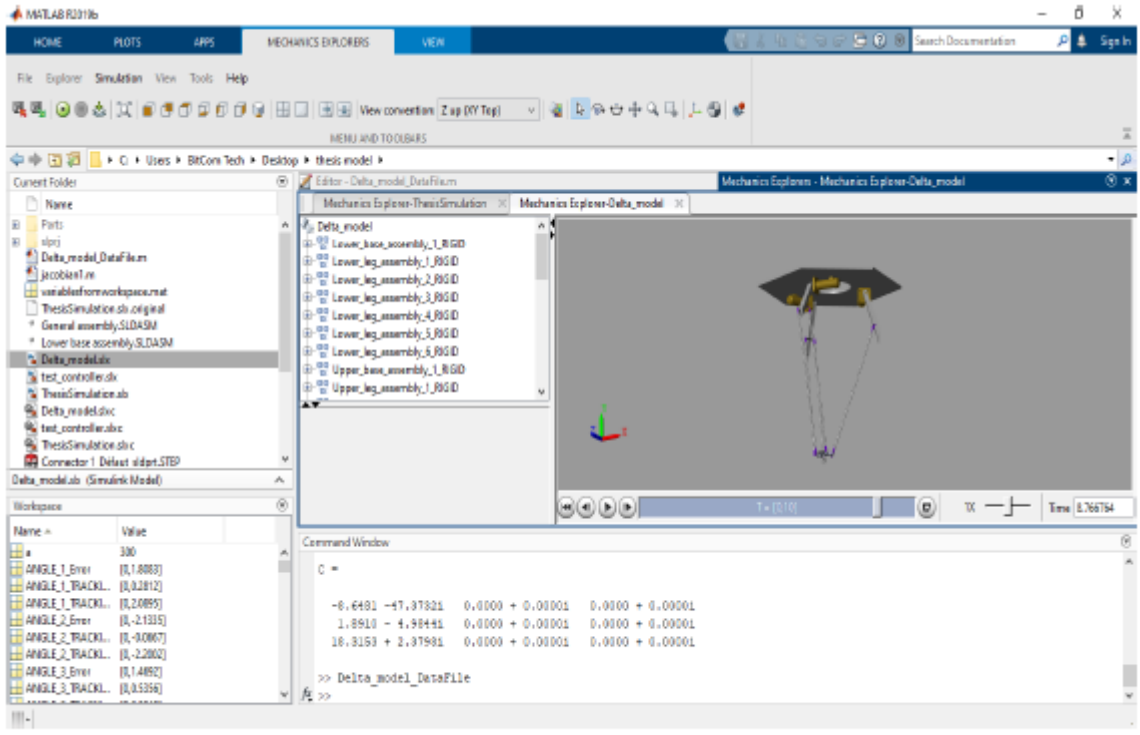


Figure 4.2: Illustrate the 3-DOF Delta Robot Simscape Multibody model exported from 3D-CAD/ solidworks environment



(b)

Figure 4.3: Matlab Mechanics Explorer of the Delta Robot model

After importing the model, unexpected rigid connections between bodies must be checked. unsupported CAD constraints are replaced by rigid connection automatically in Simscape Multibody software. It may be replaced with weld joint, direct frame connection line or rigid transform blocks. Stand on a warning message in the MATLAB command window artificial rigid connections between the bodies that arise due to unsupported constraints must be replaced with suitable joint or constraint from the Simscape Multibody library. Furthermore, physical signals are used instead of simulation signals in Simscape block diagrams. As a result, when connecting Simscape diagrams to Simulink sources and scopes, physical signal (PS) convertor blocks are required.

Rigid body subsystem and joint blocks alone are insufficient to represent the Delta Robot model in the generated Simscape model, the exported model automatically includes additional blocks that are relevant in the simulation development. The World Frame, Rigid Transform, Mechanism Configuration, and Solver Configuration blocks are among them. The following table illustrate their relevance in multibody system modeling.

Blocks	Intent
Solver Configuration	Offers essential simulation parameters required to simulate the model
World Frame	In a model, this is the ultimate reference frame. All subsequent frames are specified in relation to this one
Rigid Transform	Applies a fixed spatial relationship between frames. This block defines the offset distance and angle between two frames
Mechanism Configuration	In a model, this block identifies the gravity vector.

Table 4.2: Simscape Multibody basic blocks

Simscape Multibody 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. Motion between two arbitrary rigid body frames can be sensed using Transform Sensor block. This block provides the broadest motion sensing capability in Simscape. It lets to sense position, velocity and acceleration, both rotational and translational, between any two frames in a model. On top of this, Simscape Multibody provides a selection of actuation inputs in each Joint block.

4.4 Trajectory Generation

The end effector of the three degrees of freedom Delta robot simulated to follow a circular path trajectory based on the implementation of passivity based control strategy. 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 implemented frequency is 0.5 Hz and the radius of the desired trajectory to be tracked by the robot is 160mm centered at (15, 15, Z_0) based on the Cartesian Coordinate system where Z_0 is the traveled distance in Z-direction. Stress that the trajectory defined in this report never impedes any singular point

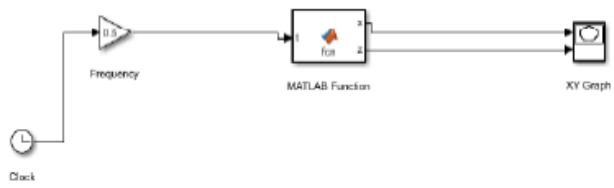


Figure 4.4: Circular path trajectory generator

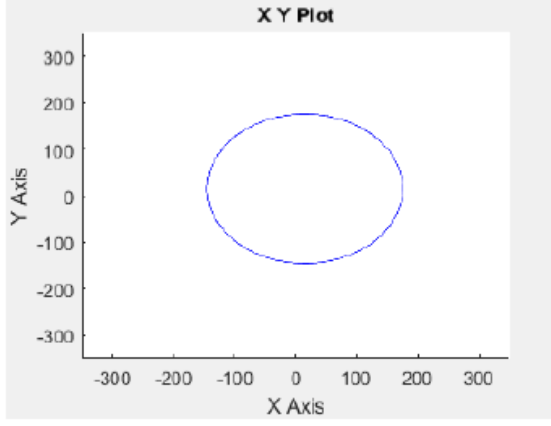


Figure 4.5: Output of the desired trajectory

4.5 Kinematics Simulink Model of Delta Robot

The detail derivation of inverse and direct kinematics of Delta robot has been discussed on chapter two section 2.2.1 and 2.2.2. Even though only the inverse kinematics is required to work the simulation of this paper, forward kinematics also studied and used to proof the kinematics algorism as follows.

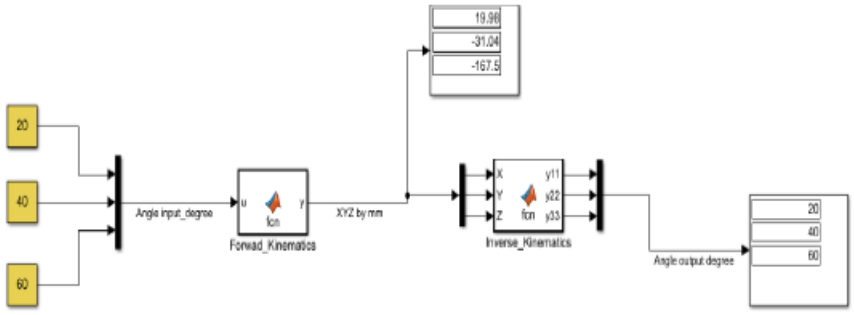


Figure 4.6: Kinematics of 3-DOF Delta Robot

4.6 Passivity Based Control Design

Mainly the designing process has two main steps. The first step is energy shaping, which involves designing a controller so that the closed-loop system matches a desired energy function that is comparable to the open-loop system's natural energy function. Passivity property of the robot system can be retained in the closed loop system at this step. The second step is damping injection, where velocity feedback is employed to introduce damping in the closed-loop system, to ensure stability.

The controller goal is to track the desired smooth trajectory q_d, where $q_d \in R^n$ and define the tracking errors as

$$\tilde{q} = q - q_d, \quad \dot{\tilde{q}} = \dot{q} - \dot{q}_d$$

The problem is to find a model-based controller τ such that the following objectives can be achieved

$$\lim_{t\to\infty} \tilde{q}_i = 0$$
$$\lim_{t\to\infty} \dot{\tilde{q}}_i = 0$$

To meet the desired control goal, the designed control law is governed by the following equation

$$\tau = M(q)\ddot{q}_r + C(q,\dot{q})\dot{q}_r + g(q) - K_d S$$

Were,

$$\dot{q}_r = \dot{\tilde{q}}_d - \Lambda\tilde{q}$$

$$S = \dot{\tilde{q}} - \Lambda\tilde{q}$$

and $\Lambda > 0, K_d > 0$ are diagonal matrices

Refer chapter three section 3.3 and 3.4 for the derivation of the final control law and proof of the robustness of the designed controller. The forgoing picture shows Simulink model of the designed controller.

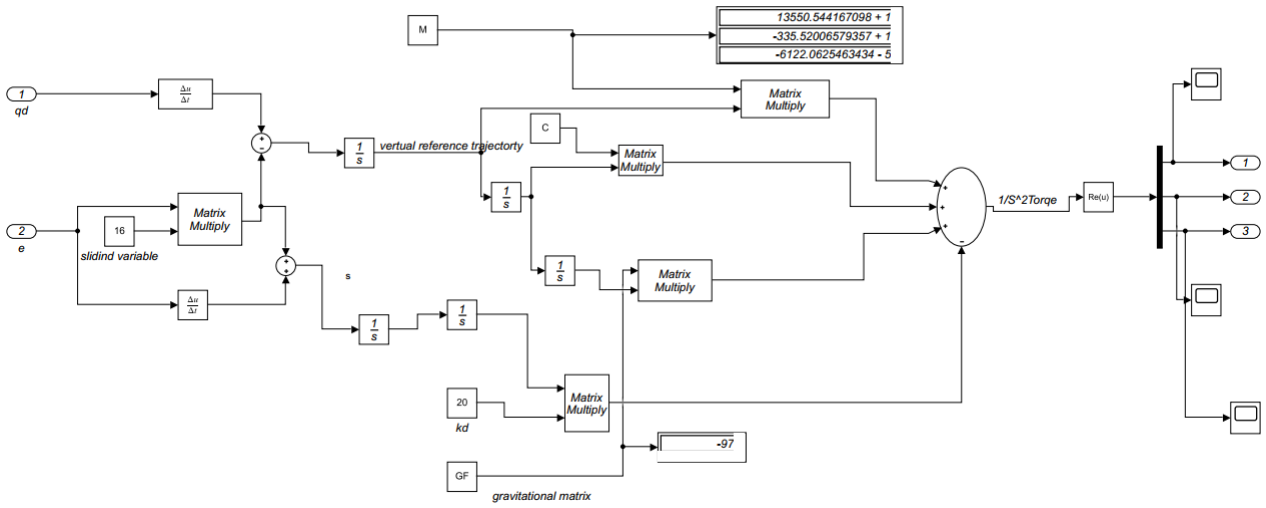


Figure 4.7: Passivity based controller Simulink model

4.7 Simulation and Result

There are major sequences of tasks to be followed in order to complete simulation of passivity based control of Delta Robot for circular trajectory tracking. The first step is to build the Solidworks model of the delta robot and generate XML file for the model and export it into Simscape Multibody. And finally, the generated Simscape Multibody model of Delta Robot has been combined with the Simulink model of the control strategy and the rest system.

The block of a Revolute joint in Simscape Multibody primitive has State Targets, Internal Mechanics, limits, Actuation and Sensing fields. Among them Actuation section incorporate torque and motion. Both fields give us the options; none, provided by input and automatically computed. This makes a total of six possible configurations for each joint primitive. From the possible six configuration, Inverse Dynamics and Forward Dynamics are mainly used for Robot's kinematics and dynamics analysis.

Motion is determined by input and torque is computed automatically in Inverse Dynamics actuation of revolute joints. This mode is useful if we know how a joint should move

and want to know how much torque or force is required. In case of 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, Passivity Based controller modeling, design and simulation. Beside the 3-DOF Parallel delta Robot is made to track a circular trajectory constructed in section 4.4 with Simulink's ODE-45 solver, which is the recommended solver for mechanical systems. Figure 4.7 shows the final setup of the Simulink simulation after exporting the CAD (solidworks) model to Simscape Multibody and integrate it with the rest Simulink block and designed control system.

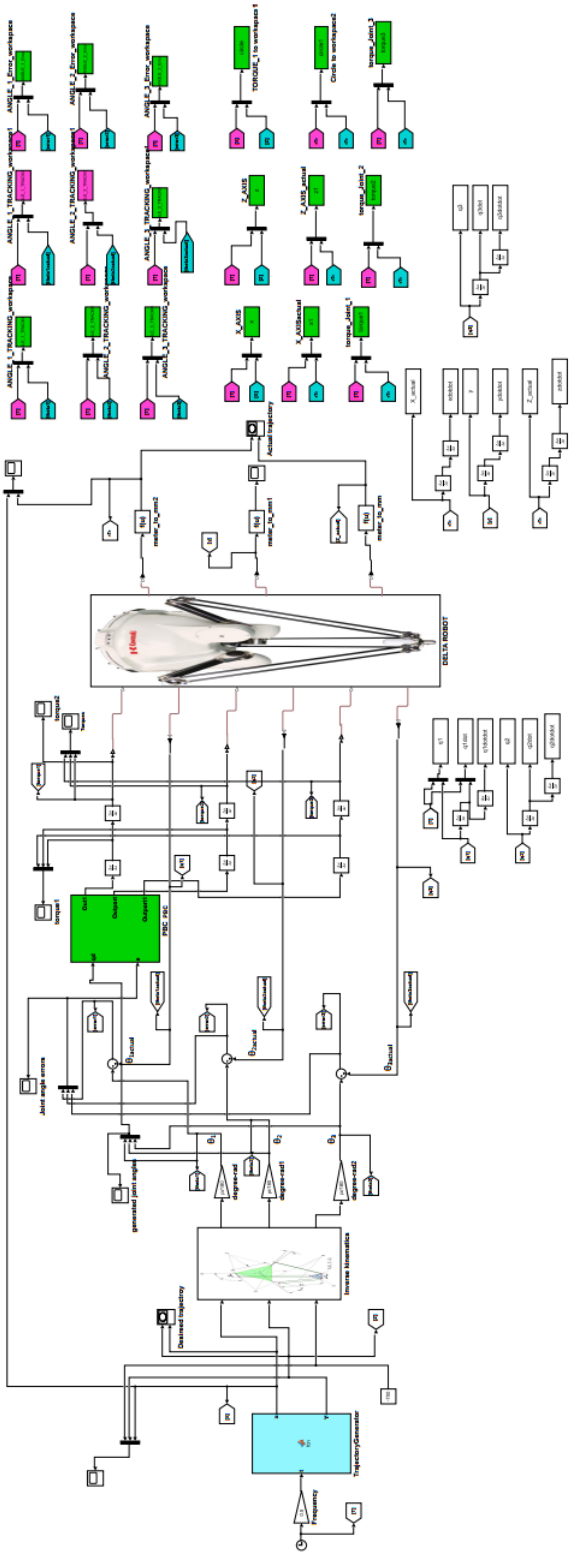


Figure 4.8: Simulink model of Passivity-Based Control of Delta Robot for circular path trajectory tracking

In this paper the desired trajectory is converted into joint angle using inverse kinematic calculation as described in section two and is applied as input to the controller (PBC). Then the computed control torque is applied to the Robot, Forward Dynamics mode, so that the motion is automatically sensed from each of the revolute joint primitive. The actual joint angle is then compared with the desired and fed to the designed controller so as to adjust the required control torque accordingly. Simulink multibody has eased a complex computation of the required torque to track or accomplish a given task or motion to be achieved which mainly involves large mathematical analysis and singularity in computation. Whereas in this scenario not only the specific value of torque or motion is required to meet control goal but also the dynamic constants such as: Inertial matrix $M(\theta)$, the Centrifugal and Coriolis forces matrix $C(\theta, \dot{\theta})$ and Gravity $G(\theta)$ as well. Which in return require computation of Jacobean matrix. Detail derivation of the 3 Dof Delta Robot dynamic constants has been discussed in chapter two and also refer Appendix C for Matlab script.

MATLAB displays control results such as joint angle error, required control torque and reference tracking. Whereas the Mechanics Explorer displays the Robot's motion in 3D space during simulation. The entire simulation system is illustrated in Figure 4.8 and the simulation is split into two sections. The first portion is a simulation in which no external torque disturbances are present and in the second section, the effect of external disturbances are included throughout the simulation course.

4.7.1 Result Without External Disturbance

The 3-dof Delta Robot inverse kinematics problem is computed given the Cartesian position of the desired path trajectory in order to find the required actuated revolute joint angels θ_1 , θ_2 and θ_3. The values of the joint angles of θ_1 , θ_2 and θ_3 are vital in determining the exact position of the end effector. The following portrait shows the required angle generated for the three active joints in order to track the desired circular path trajectory, which is a circle on X-Y plane centered at (15,15, Z_0) with 160mm radius.

Figure 4.9: Generated joint angles

The calculated torque (see chapter two) imposed on each active joint of the three DoF Delta Robot and then actual position of the end effector is sensed. Which in return fed back to the Passivity Based Controller (PBC) against the desired position to get the controlled torque. If the strategy is effective, this controlled torque should enforce the system to be in stable state and the end effector must track the desired circular trajectory within acceptable time. The below figure shows the required control torques needed to achieve the desired circular trajectory for each of the three active revolute joints.

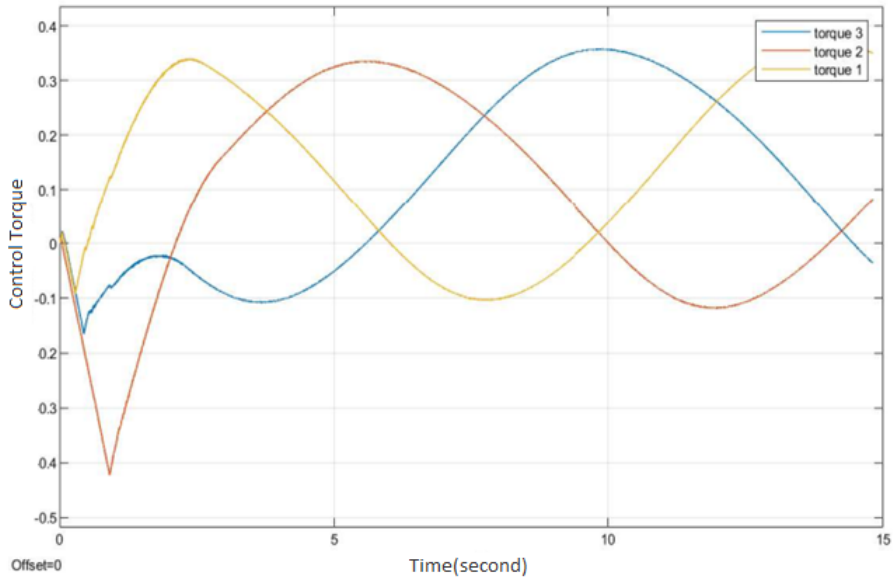
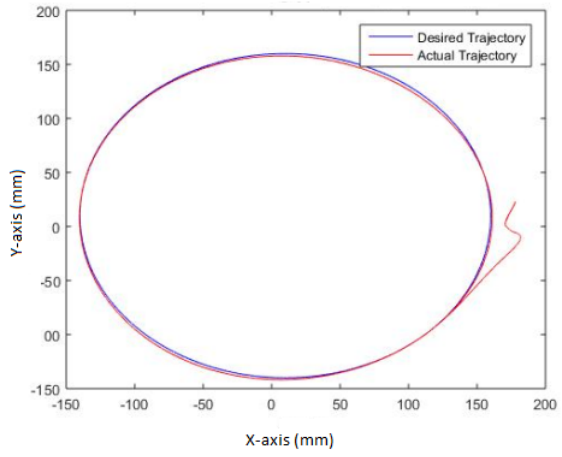
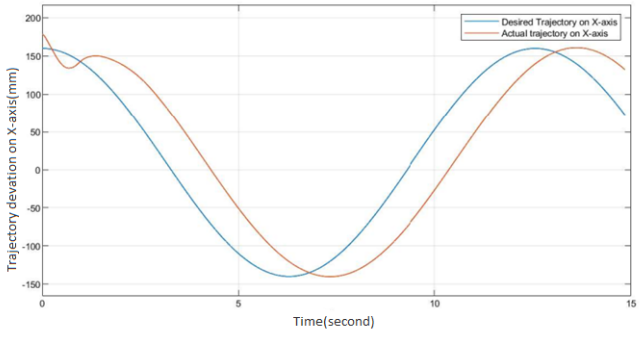


Figure 4.10: Required PBC Control Torques for the three active revolute joints

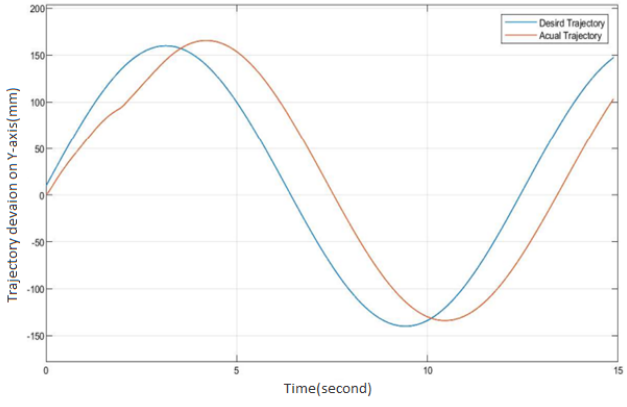
The following set of plots shows the end effector trajectory tracking along with its respective error between its desired and actual position. Beside end effector trajectory error on X-axis and Y-axis are provided to readily determine the severity of the its accuracy.



(a) End effector trajectory tracking



(b) End effector X-axis trajectory



(c) End effector Y-axis trajectory

Figure 4.11: (a) portray the end effector actual trajectory against the desired and Fig4.10 (b) and (c) illustrate the tracking error on X and Y-axis

The forgoing portrait shows angle error of the three active joints of the Delta Robot in response to the controlled torque.

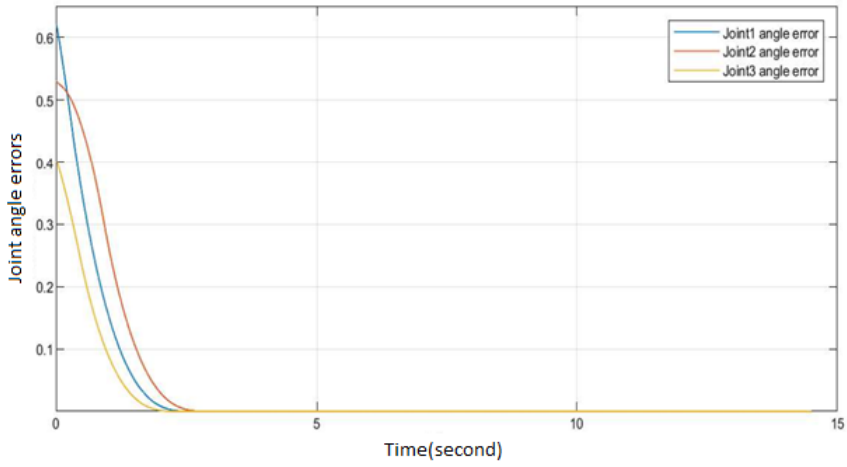


Figure 4.12: Joint angle errors

The joint angle error, end effector trajectory and end effector trajectory error on X and Y-axis figures illustrate that the designed controller, Passivity Based controller has made the error to converge in less than 3 second with joint angle root mean square error of 0.0017rad and end effector trajectory rms error of 1.1mm on X-axis and 1.33mm on Y-axis .

4.7.2 Result With the Effect of External Disturbance

The above simulation result is dedicated for undisturbed system. Even though the robustness of the designed controller, PBC, has been proofed in chapter three section 3.4, late elaborate it by adding external disturbance on the model. In the same way as before, the 3-dof Delta Robot inverse kinematics problem is carried out first having the Cartesian position of the desired path trajectory in order to find the required actuated revolute joint angels θ_1 , θ_2 and θ_3. The values of the joint angles of θ_1 , θ_2 and θ_3 are vital in determining the exact position of the end effector after introducing external torque disturbance to the system.

A time-varying external torque disturbance with sin wave amplitude of $[0.3\ ,05,0.3]^T$ is imposed at the three revolute joints that are also configured to have an internal mechanics of damping coefficient of one, which is the torque necessary to maintain a constant joint primitive angular velocity between the base and follower frames, to test the robustness of the Passivity Based Control (PBC) in this research. Because of the designed controller, PBC, is model-based, the robustness of the designed Passivity Based Controller could not be tested by varying various designing factors of the robot unevenly, such as mass, upper and lower leg length of the three symmetrical legs.

For the whole course of simulation all internal mechanics and external torque disturbances are taken into account. As in the previous portion, the simulation results are given for trajectory tracking of a circular path, which is a circle on the X-Y plane centered at (15,15, Z_0) with a radius of 160mm.

As in the previous section the computed torque, based on the philosophy of PBC, imposed on each active joint of the three DOF Delta Robot and then actual position of the end effector is sensed. Which in return fed back to the Passivity Based Controller against the desired position to get the controlled torque. If the strategy is effective, this controlled torque should enforce the system to be in stable state and the end effector must track the desired circular trajectory irrespective of disturbance within acceptable time. The below figure shows the required control torques needed to achieve the desired circular trajectory for each of the three active revolute joints after introducing external disturbance to the system.

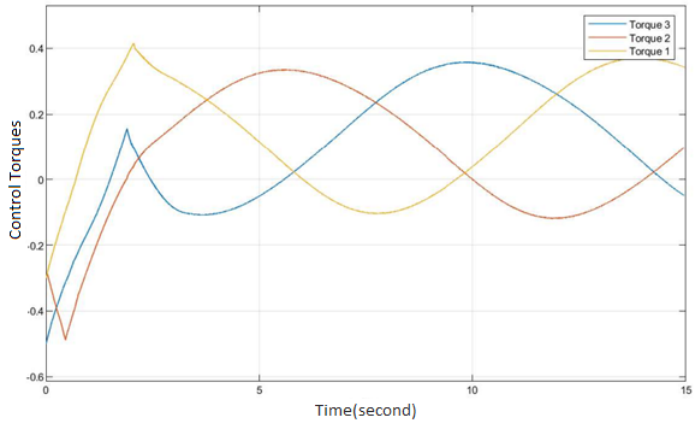


Figure 4.13: Passivity based control torque for the three active joints

The forgoing set of picture shows the end effector trajectory tracking with the effect of internal mechanics and external torque disturbance. To make a stress the plot portrait the end effector trajectory along with its respective error between its desired and actual position. In addition, end effector trajectory error on X-axis and Y-axis are provided to readily determine the severity of the its accuracy.

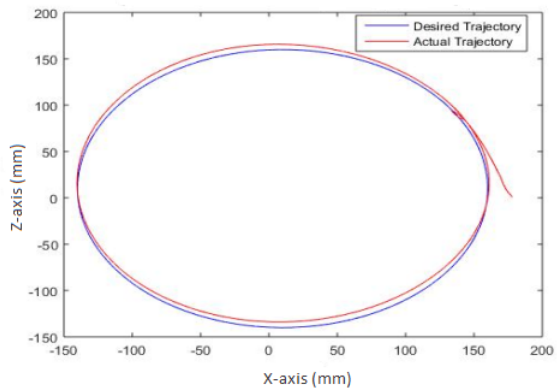


Figure 4.14: End effector trajectory tracking

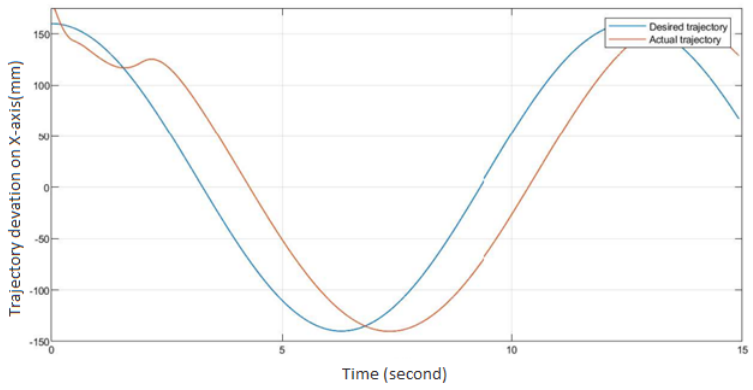


Figure 4.15: End effector trajectory error on X-axis

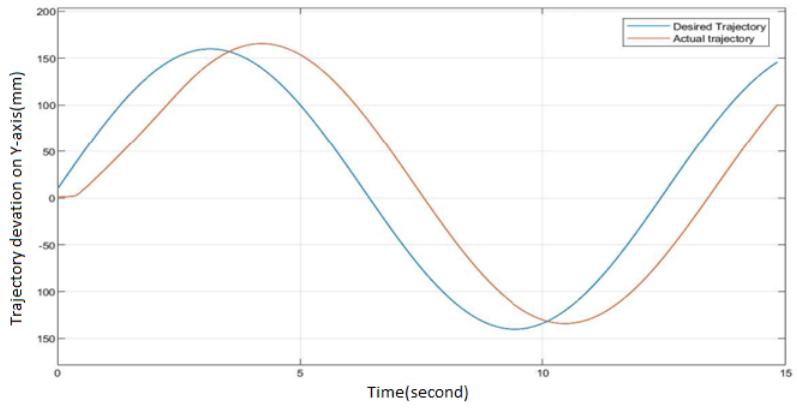


Figure 4.16: End effector trajectory error on Z-axis

The above portraits shows that even after introducing external disturbance to the system, the controller is also able to produce the required amount of control torque to maintain the stability of the system and track the desired trajectory within time. PBC has made the error to converge with joint angle rms error of 0.00173rad and end effector trajectory rms error of 2.88 mm on X and Y-axis.

4.8 Discussion

Beside the mathematical proof, the simulation output figures illustrate that the Passivity based control scheme, controlled torque, is able to preserve the passivity of the robot closed-loop system and maintain the stability of the system and track the desired circular trajectory in a finite time, less than three seconds with and without external disturbance.

Passivity based controller is used for achieving stability of nonlinear systems by making the system passive, because passive systems are by nature asymptotically stable systems. Since it strongly exploits the system properties, it does not produce unnecessarily large control effort and has inherent robustness against plant uncertainty and disturbance. However, the problem of PBC is its model dependence. In order to apply the designed passivity based control strategy, the exact value of the dynamic parameters such as: Inertial matrix $M(\theta)$, the Centrifugal and Coriolis forces matrix $C(\theta, \dot{\theta})$ and Gravity $G(\theta)$ as well must be known explicitly . Thus, it requires computation of matrixes, Jacobean matrix. So, we can say that somehow it fades one of the benefits can be gained from Simulink Multibody and makes the work a bit challenging.

One of the cons of designing robot model on CAD (Solidworks) environment and export it into Simulink Multibody is to ease a complex computation carried out to obtain the required output and input that can be position, angles, forces or torque which mainly involves large mathematical analysis and singularity in its computation. Solidworks models can be exported and simulated in the Simulink environment to evaluate forces and torques in mechanical joints, plot accelerations and displacements of each component of the system, run various tests, and acquire needed output without any necessity of motion equations derivation and Simscape physical modeling of the mechanical system.

Simulink Multibody tool introduced an easy way to actuate Robot joints by joint actuator block. Sensing computed torques, angles, angular speeds, and accelerations are possible by joint sensor. Beside Bod sensor is used to get the Cartesian Coordinates of the moving platform. Thus, it facilitates the overall simulation process.

Delta robots are applied in various application like Packing industry, High-precision assembly operations, Food processing, Medical/Pharmaceutical operations, Soldering, Adhesive dispensing, 3D printing machine and in so many other aspects. Therefore, based on the application and preference of either position precision or workspace, the upper

and the lower leg length of the Delta Robot can be customized. Although there is no set rule, the work space widens as the leg length increases but small error on the actuator may cause bigger position accuracy gap on the end effector. On the contrary, when the leg length lowers, so does the work space but precision increase. Specifically for this thesis the 3-DOF Delta Robot parameters has been taken from a specific commercial Delta Robot, ABB FlexPicker IRB 360- 1/1600.

The robot (3-DOF Delta robot) real but not approximated parameters have been introduced via CAD dynamic modeling which is one of the advantages of CAD(Solidworks) modeling environment. On the hand, the simulation speed is low. The simulation speed of mathematical dynamic model is more rapid than CAD dynamic model.

5 Conclusion, Recommendation and Future Work

5.1 Conclusion

The 3-DOF Delta Robot real parameters is taken from a specific commercial Delta Robot, ABB FlexPicker IRB 360- 1/1600 and designed using 3D-CAD SolidWorks 2019 modeling environment. XML file is generated for the designed CAD model and then exported into Simscape Multibody. And finally, the generated Simulink model of Delta Robot has been combined with the Simulink model of the control architecture (Passivity based Control scheme) and the rest system. Through this process the Simulink-PS Converter block play a good role in converting Simulink signal into physical signal and vice versa.

In this thesis, the kinematics, dynamics and non-singularity analysis of the 3-DOF Delta robot has been done for 3-DOF Delta Robot, a Passivity Based controller has been developed and the desired circular trajectory, which is a circle on X-Y plane centered at (15, 15,Z_0) with 160mm radius has been tracked. The simulation result shows that the steady state tracking is reached in about 3 seconds for a circular trajectory and X & Y rms error of 2.88mm even though the simulation speed of CAD dynamic model is slower than mathematical dynamic model. The robustness of the designed controller is also demonstrated through mathematical computation and via simulation as well.

5.2 Recommendation

For both set point regulation and trajectory tracking control problem, the amount of torque or force required to accomplish the desired trajectory or destination point must be computed. Which in turn require sophisticated and involved mathematical commutation. Besides, singularity may also a bigger concern throughout the computation. Therefore, in order to ease such complex mathematical computation, it is recommended to use software packages which have modeling capability. In this thesis work, the model of the Delta robot has been modeled using CAD (Solidworks) modeling environment and Simscape

Multibody ease the complex mathematical computation. Hence, the required torque is automatically computed.

5.3 Future Work

The use of robotics and automation in healthcare and related fields is growing by the day. Specially after the outbreak of the novel coronavirus disease 2019 (COVID-19), the evolving role of robotics has enormously increased to minimize person-to-person contact and to ensure cleaning, sterilization and support in hospitals and similar facilities such as quarantine. According to the paper released by International journal of environmental research and public health in May 2020, the International Federation of Robots (IFR) predicts an ever-increasing trend in the demand of medical robots within the coming years with an estimation of 9.1 billion USD market by 2022. Most of the service robots in hospitals are variants of mobile robots with a high payload capacity but with limited degrees of freedom. However, surgical robots with multi DOF are flexible, precise, and reliable systems offering similar performance to that of a well-trained human surgeon with a minimum error margin typically within millimeters [31, 32]. Therefore, stand on this big market demand, my future work plan is to extend the degree of freedom of Delta robot and test the simulation result on the real robot. Apart from this, the designed controller of the thesis is model based but to make its application wide, I recommend to integrate the passivity based control philosophy with others adaptive and robust control scheme so as to make it non model based.

Reference

[1] Mark W. Spong. **Motion control of robot manipulators** (1996), 1339–1350 (see page 1).

[2] Vjekoslav Damic, Maida Cohodar, and Avdo Voloder. **Modelling and Path Planning of Delta Parallel Robot in Virtual Environment**. In: *Proceedings of the Annals of DAAAM and International DAAAM Symposium, Zadar, Croatia*. 2018, 21–28 (see pages 1, 2).

[3] László Szűcs, Péter Galambos, and Dániel András Drexler. **Kinematics of Delta-type Parallel Robot Mechanisms via Screw Theory: A tutorial paper**. In: *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE. 2021, 000293–000298 (see page 1).

[4] Luis Angel Castañeda, Alberto Luviano-Juárez, and Isaac Chairez. **Robust trajectory tracking of a delta robot through adaptive active disturbance rejection control**. *IEEE Transactions on Control Systems Technology* 23:4 (2014), 1387–1398 (see pages 1, 11).

[5] YD Patel and PM George. *Parallel Manipulators Applications—A Survey. Modern Mechanical Engineering, 02 (03), 57–64*. 2012 (see pages 1, 2).

[6] Yong-Lin Kuo and Peng-Yu Huang. **Experimental and simulation studies of motion control of a Delta robot using a model-based approach**. *International Journal of Advanced Robotic Systems* 14:6 (2017), 1729881417738738 (see pages 1, 2).

[7] Mauro Maya, Eduardo Castillo, Alberto Lomelí, Emilio González-Galván, and Antonio Cárdenas. **Workspace and payload-capacity of a new reconfigurable delta parallel robot**. *International Journal of Advanced Robotic Systems* 10:1 (2013), 56 (see page 1).

[8] Amir Hashemi Dastjerdi, Mohammad Morad Sheikhi, and Mehdi Tale Masouleh. **A complete analytical solution for the dimensional synthesis of 3-DOF delta parallel robot for a prescribed workspace**. *Mechanism and Machine Theory* 153 (2020), 103991 (see pages 3, 4).

[9] Hualin Yang, Long Chen, Zhibin Ma, Miaoting Chen, Yan Zhong, Fang Deng, and Maozhen Li. **Computer vision-based high-quality tea automatic plucking robot using Delta parallel manipulator**. *Computers and Electronics in Agriculture* 181 (2021), 105946 (see pages 3, 4).

[10] Yuancan Huang and Zonglin Huang. **Neural Network based dynamic trajectory tracking of Delta parallel robot**. In: *2015 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE. 2015, 1938–1943 (see pages 3–5, 12, 15, 44).

[11] Robert L Williams. **The delta parallel robot: Kinematics solutions**. *Mechanical Engineering, Ohio University* (2016), 1–15 (see pages 3, 4, 11, 15, 16, 19, 25).

[12] Yinyan Zhang, Shuai Li, Jianxiao Zou, and Ameer Hamza Khan. **A passivity-based approach for kinematic control of manipulators with constraints**. *IEEE Transactions on Industrial Informatics* 16:5 (2019), 3029–3038 (see pages 4, 42).

[13] Dongjun Lee and Kent Yee Lui. **Passive configuration decomposition and passivity-based control of nonholonomic mechanical systems**. *IEEE Transactions on Robotics* 33:2 (2016), 281–297 (see pages 4, 42).

[14] Romeo Ortega, Julio Antonio Loría Perez, Per Johan Nicklasson, and Hebertt J Sira-Ramirez. **Passivity-based control of Euler-Lagrange systems: mechanical, electrical and electromechanical applications**. Springer Science & Business Media, 2013 (see pages 4, 11, 42–44).

[15] JE Lavin-Delgado, S Chavez-Vazquez, JF Gomez-Aguilar, G Delgado-Reyes, and MA Ruiz-Jaimes. **Fractional-order passivity-based adaptive controller for a robot manipulator type scara**. *Fractals* 28:08 (2020), 2040008 (see pages 4, 12, 42).

[16] Chao Ren, Yutong Ding, Shugen Ma, Liang Hu, and Xinshan Zhu. **Passivity-based tracking control of an omnidirectional mobile robot using only one geometrical parameter**. *Control Engineering Practice* 90 (2019), 160–168 (see pages 4, 42, 44).

[17] Ollin Peñaloza-Mejía, CP Ojeda-Perez, and Héctor Javier Estrada-García. **Passivity-based tracking control of robot manipulators with torque constraints**. In: *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE. 2016, 947–952 (see pages 4, 42, 44).

[18] Chems Eddine Boudjedir and Djamel Boukhetala. **Adaptive robust iterative learning control with application to a Delta robot**. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 235:2 (2021), 207–221 (see pages 5, 12, 13).

[19] János Somló, Gábor Dávid Varga, Márk Zenkl, and Balázs Mikó. **The `Phantom" Delta Robot A New Device for Parallel Robot Investigations**. *Acta Polytechnica Hungarica* 15:4 (2018), 143–160 (see page 11).

[20] Xingguo Lu, Yue Zhao, and Ming Liu. **Self-learning interval type-2 fuzzy neural network controllers for trajectory control of a Delta parallel robot**. *Neurocomputing* 283 (2018), 107–119 (see page 13).

[21] Meryam Rachedi, Mohamed Bouri, and Boualem Hemici. **Application of an H221e control strategy to the parallel Delta**. In: *CCCA12*. IEEE. 2012, 1–6 (see page 13).

[22] Yu Li, Deyong Shang, and Yue Liu. **Kinematic modeling and error analysis of Delta robot considering parallelism error**. *International Journal of Advanced Robotic Systems* 16:5 (2019), 1729881419878927 (see pages 15, 16).

[23] Mark W Spong and Mathukumalli Vidyasagar. **Robot dynamics and control**. John Wiley & Sons, 2008 (see pages 18, 40).

[24] **Abacus, Delta robot kinematics tutorial , https://hypertriangle.com/ alex/delta robot-tutorial/**. 2009 (see pages 19, 22, 25, 27, 28, 30).

[25] Krystian Łygas, Piotr Wolszczak, Tomasz Klepka, and Daniel Ghiculescu. **Kinematic design of parallel delta system in Matlab**. In: *Applied Mechanics and Materials*. Vol. 844. Trans Tech Publ. 2016, 7–12 (see page 20).

[26] Lung Wen Tsai. **Robot Analysis the mechanical of serial and parallel manipulators', Department of Mechanical Engineering and Institute for Systems Research university of Maryland** (see pages 21, 27).

[27] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. **Robot modeling and control**. 2020 (see page 31).

[28] Mohsen Asgari, Mahdi Alinaghizadeh Ardestani, and Mersad Asgari. **Dynamics and control of a novel 3-DoF spatial parallel robot**. In: *2013 First RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)*. IEEE. 2013, 183–188 (see pages 31, 37, 39, 40, 44).

[29] M López, E Castillo, G García, and A Bashir. **Delta robot: inverse, direct, and intermediate Jacobians**. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 220:1 (2016), 103–109 (see pages 32, 34, 44).

[30] Heinz D Unbehauen. **CONTROL SYSTEMS, ROBOTICS AND AUTOMATION–Volume XXII: Robotics**. EOLSS Publications, 2009 (see pages 42, 44).

[31] Zeashan Hameed Khan, Afifa Siddique, and Chang Won Lee. **Robotics utilization for healthcare digitization in global COVID-19 management**. *International journal of environmental research and public health* 17:11 (2020), 3819 (see page 73).

[32] Shivaramakrishnan Balasubramanian, Jayakumar Chenniah, Gurumurthy Balasubramanian, and Venkataram Vellaipandi. **The era of robotics: Dexterity for surgery and medical care: Narrative review**. *International Surgery Journal* 7:4 (2020), 1317–1323 (see page 73).

A Inverse Kinematics

```
function [theta1, theta2, theta3] = ...
    Deltarobotinversekinematics(x0, z0,y0)

    %Calculates the inverse kinematics of 3-DOF Delta Robot

    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 by ...
        +120 deg
    theta3= Delta_calcAnglezy(x02, z02, y0);  % rotate by ...
        -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;       %upper leg length

        z1 = -0.5 * 0.57735 * f; % f/2 * tg 30(from geometry ...
            of the Delta)
        z0 =z0 - 0.5 * 0.57735 * e; % shifting the center to ...
            the 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

    % choosing outer point
    zj = (z1 - a*b - sqrt(d))/(b*b + 1);
    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=5; % exit if the point does not exist
end
```

B Direct Kinematics

```
function [x0,z0,y0,d]= Delta_calcForward( theta1,theta2,theta3)

%calculates the Direct kinematics of 3-DOF Delta Robot

    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);

% The 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
```

C Syntax for Jacobian Matrix and Dynamic Constants

```
% Jacobian matrix Generation
% took sample @ time =[0.0881887617649640]

    e = 199.01;      % end effector
    f = 841.77;      %base
    re = 800;        % lower leg length
    rf = 300;        % upper leg length
    ma =  0.0811;    %mass of upper leg
    mb = 0.1170;     %mass of lower leg
    g = 9.81;        %gravity
    mp = 0.0452;     %mass of moving platform

    px = 173.215491332598;
    py = -0.974493902620554;
    pz = 1.20094005347463;
    q11 = 0.876272316341518;
    q12 = 0.449713147816531;
    q13 = 0.910373602925376;

    cx1 = px + f - e;
    cy1 = py;
    cz1 = pz;

    cx2 = py + f - e;
    cy2 = -px;
    cz2 = pz;

    cx3 = -px + f - e;
    cy3 = -py;
    cz3 = pz;

    q31 = acos(cy1/re);
    q32 = acos(cy2/re);
```

```
    q33 = acos(cy3/re);

    q21 =  acos((cx1*cx1 + cy1*cy1 + cz1*cz1 - rf*rf - ...
       re*re)/(2*re*rf*sin(q31)));
    q22 = acos((cx2*cx2 + cy2*cy2 + cz2*cz2 - rf*rf - ...
       re*re)/(2*re*rf*sin(q32)));
    q23 = acos((cx3*cx3 + cy3*cy3 + cz3*cz3 - rf*rf - ...
       re*re)/(2*re*rf*sin(q33)));

theta = [q11 ,q12, q13 ; q21 ,q22 , q23 ; q31 ,q32 , q33]; % ...
   all angles value

    j1x = sin(q31)*cos(q21 + q11);
    j2x = cos(q32);
    j3x = -1*sin(q33)*cos(q23 + q13);

    j1y = cos(q31);
    j2y = -sin(q32)*cos(q22 + q12);
    j3y = - cos(q33);

    jlz = sin(q31)* sin(q21 + q11);
    j2z = sin(q32)* sin(q22 + q12);
    j3z = sin(q33)* sin(q23 + q13);


jpos = [j1x , j1y , jlz ; j2x , j2y ,j2z ; j3x , j3y , j3z ];

jtheta = rf.* [sin(q21)*sin(q31), 0 , 0 ; 0 , ...
   sin(q22)*sin(q32) ,0 ; 0 , 0 ,sin(q23)*sin(q33) ];
J = jtheta\jpos  % Jacobian
%c = jtheta\jtheta should be identity matrix

%calculation of gravitational force
I = [1 , 0, 0 ;0 , 1 , 0; 0, 0, 1];


 y =(0.5*ma+mb)*g *a;
 Mga = y*[cos(q11); cos(q12); cos(q13)];
 fgp = [0 ; 0 ; -1*(mp + 3* mb)*g];

jinvr=inv(J);   %inverse of Jacobian
```

```
Jit= jinvr.' ;  % transpose of inverse of
G = -Mga - Jit * fgp
GF =  G.*[1,0,0;1,0,0;1,0,0]
%Calculation of inertia matrix

Ia = (1/3 * ma*a*a +  mb*a*a)*I ;
Mp = [mp, 0,0;0,mp,0;0,0,mp];
Msub = Jit*Mp/J;%Jit*Mp*jinvr;
M = Ia + Msub
%e = eig(M)% to check all the eigen value of M is positive (M ...
    should be symmetric positive definite)

% Calculation of the centrifugal and Coriolis forces
% Calculation of derivative of Jacobean inverse

%Pdotdot= [xdotdot,ydotdot,z dotdot]
%qdotdot= [q1dotdot,q2dotdot,q3dotdot]
%qdot= [q1dot,q2dot,q3dot]

pdotdot = [-1173.46391323112 ; -0.686716185423232 ; ...
    -31.3586256600572];
qdot = [-1173.46391323112 ; -0.147364080555540 ...
    ;-0.375225178665328];
qdotdot = [-2.72575151997963 ; -0.00516352026225666 ; ...
    -2.62748745589489];

jdotinv = (pdotdot - (J\qdotdot))/qdot;
C = Jit * Mp * jdotinv
```

D Syntax for Circular Trajectory Generation

```
function [x,y] = fcn(t)

    x = 15+160*cos(t);
    Y = 15+160*sin(t);
```