PID CONTROL OF 3-DOF ROBOTIC ARM

A thesis submitted in partial fulfillment of the requirements for
the award of the degree of

B.Tech.

in

INSTRUMENTATION AND CONTROL ENGINEERING

By

D. SUDHINDRA (110119039)

G. PRIYADARSHINI (110119040)

RISHIVANTHIYA G R (110119092)

DEPARTMENT OF
INSTRUMENTATION AND CONTROL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
TIRUCHIRAPPALLI-620015
MAY 2023

PID CONTROL OF 3-DOF ROBOTIC ARM

A thesis submitted in partial fulfillment of the requirements for the award of the degree of

B.Tech.

in

INSTRUMENTATION AND CONTROL ENGINEERING

By

D. SUDHINDRA (110119039)

G. PRIYADARSHINI (110119040)

RISHIVANTHIYA G R (110119092)

DEPARTMENT OF
INSTRUMENTATION AND CONTROL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
TIRUCHIRAPPALLI-620015
MAY 2023

BONAFIDE CERTIFICATE

This is to certify that the project titled **PID CONTROL OF 3-DOF ROBOTIC ARM** is a bonafide record of the work done by

D. SUDHINDRA (110119039)
G. PRIYADARSHINI (110119040)
RISHIVANTHIYA G R (110119092)

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **INSTRUMENTATION & CONTROL ENGINEERING** of the **NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI**, during the year 2022-2023.

Dr. RAHUL KUMAR SHARMA	**Dr. K. DHANALAKSHMI**
Project Guide	Head of the Department

Project Viva-voce held on

Internal Examiner	**External Examiner**

ABSTRACT

Dobot Magician is a multifunctional desktop robotic arm for practical training education, supporting teaching and research, graphic programming, script, etc. Installed with different end-effectors, Dobot Magician can realize interesting functions such as 3D printing, laser engraving, writing and drawing. It also supports secondary development by various extensible I/O interfaces. It is a 3 degrees of freedom (DOF) robot widely used nowadays in education, research, and industrial purposes. It is basically a robotic manipulator that can be utilized for multiple applications. Dobot Magician, a low-cost robotic manipulator, available in the market is used in this research.

In this paper, a mathematical model of the system is obtained by developing forward kinematics, inverse kinematics and dynamic equations of the Dobot magician robot. A non-linear approach is used for cartesian control of its end-effector position. Denavit-Hartenberg (DH) convention is used to obtain forward kinematics equations. An Algebraic approach is used to find inverse kinematics solution. An algorithm based on developed inverse kinematic is developed for the conversion of Dobot's Cartesian variables into joint variables. This conversion facilitates to obtain Cartesian control of Dobot Magician Robot. The simulation results show stability and efficacy of the developed Control approach in maneuvering the Dobot's end-effector position.

Keywords: Forward Kinematics, Inverse Kinematics, Dynamics, Cartesian Control, Trajectory Tracking.

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to our project guide **Dr. Rahul Kumar Sharma**, Department of Instrumentation and Control Engineering, National Institute of Technology, Tiruchirappalli for providing us with an opportunity to work with him. While providing constant support throughout the project. He greatly helped us understand the concepts involved and pushed us to think more critically and without his guidance, we would not have been able to successfully complete the project. His valuable feedback helped us a lot in avoiding potential pitfalls and dealing with possible errors. His patience and genial attitude will always be a source of inspiration to us.

My hearty thanks to all the members of the **Project Coordination Committee,** Dr. A. Ramakalyan (Chairman), Dr. D. Ezhilarasi, and Dr. C. Geetha, Department of Instrumentation and Control Engineering, National Institute of Technology, Tiruchirappalli for their precious suggestions and comments during the project reviews.

Our university **National Institute of Technology, Tiruchirappalli** for providing an opportunity to conduct research in addition to providing quality teaching and research facilities.

We would also like to thank the faculty and staff members of the **Department of Instrumentation and Control Engineering** and our individual parents and friends for their constant support and help.

D. Sudhindra (110119039)
G. Priyadarshini (110119040)
Rishivanthiya G R (110119092)

TABLE OF CONTENTS

Title **Page No.**

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Introduction

In the rapidly evolving technological landscape of the 21st century, robotic arms have emerged as a game-changing innovation with profound implications across various industries. These versatile and dexterous mechanical limbs have revolutionized manufacturing, healthcare, space exploration, and many other sectors. By combining precision, efficiency, and adaptability, robotic arms have become indispensable tools, enabling humans to accomplish tasks that were previously inconceivable.

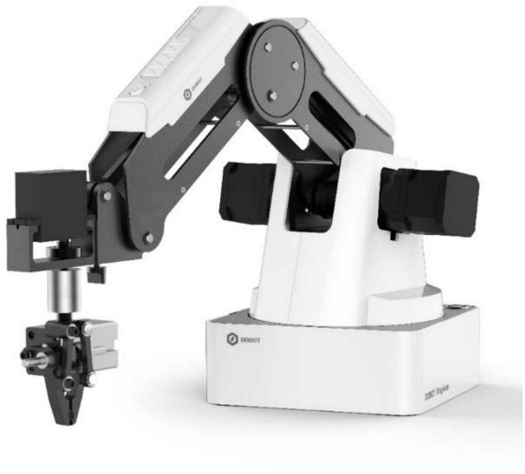


Figure1.1: Dobot Magician

Enhanced Productivity: One of the primary advantages of robotic arms lies in their ability to significantly enhance productivity across different sectors. With their exceptional precision and tireless work ethic, robotic arms excel at repetitive, monotonous, and physically demanding tasks. In manufacturing industries, they have streamlined production lines, increased efficiency, and reduced errors. The ability to perform intricate movements with consistent accuracy leads to improved product quality, lower production costs, and faster assembly times. Consequently, businesses can meet growing demands while maintaining competitiveness in an increasingly globalized economy.

Improved Safety: Robotic arms play a crucial role in improving safety conditions in hazardous work environments. By delegating dangerous tasks to robotic systems, human workers are protected from life-threatening risks associated with toxic substances, extreme temperatures, or high-pressure environments. Industries such as mining, nuclear power, and chemical manufacturing have seen substantial improvements in worker safety due to the utilization of robotic arms. These mechanical limbs can access confined spaces and handle materials that would otherwise put human lives at risk. Thus, they not only protect workers but also minimize the occurrence of accidents, ultimately saving lives.

Advancements in Healthcare: The healthcare industry has witnessed remarkable advancements through the integration of robotic arms. Surgical procedures have become more precise, less invasive, and safer, thanks to robotic-assisted surgeries. With the ability to perform complex and delicate maneuvers, robotic arms enhance surgical accuracy and reduce recovery times for patients. Surgeons can manipulate robotic arms with exceptional precision, making smaller incisions and reducing the risk of complications. Additionally, robotic arms enable telesurgery, allowing medical experts to perform procedures remotely, expanding access to specialized care in remote areas or during emergencies.

Space Exploration: Robotic arms have played a pivotal role in space exploration, enabling humans to reach unprecedented frontiers. They have become a crucial component of space missions, assisting in satellite deployment, maintenance, and repairs. For instance, the International Space Station (ISS) relies on robotic arms for various tasks, including docking spacecraft, moving equipment, and conducting spacewalks. These mechanical limbs offer a range of motion and strength that surpass human capabilities in the vacuum of space. They have been proven instrumental in advancing our understanding of the universe and preparing for future manned missions to celestial bodies.

1.1.1 Dobot Magician

Our project canters around the Dobot Magician. Dobot Magician is a multifunctional desktop robotic arm for practical training education, supporting teaching and playback, blockly graphic programming, script, etc. Installed with different end-effectors, Dobot Magician can realize interesting functions such as 3D printing, laser engraving, writing and drawing. It also supports secondary development by various extensible I/O interfaces.

Dobot Magician consists of A Base, Rear Arm, Forearm, and End-effector:

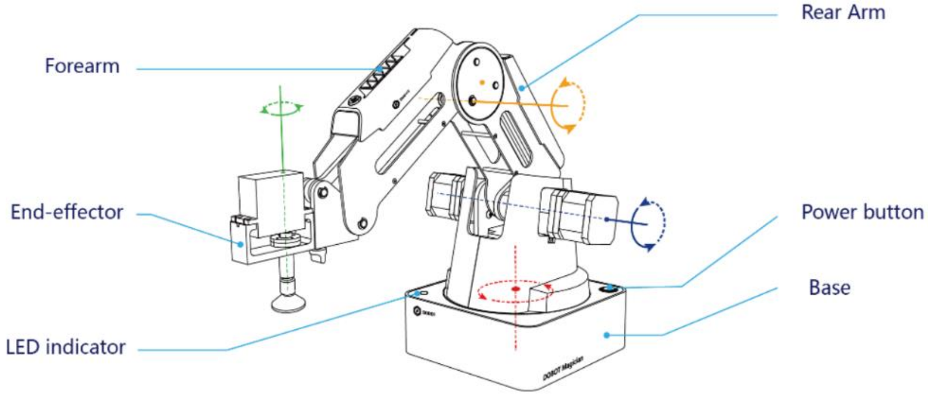


Fig 1.2: Parts of Dobot Magician

1.1.2 Workspace

The following figures show the workspace of Dobot

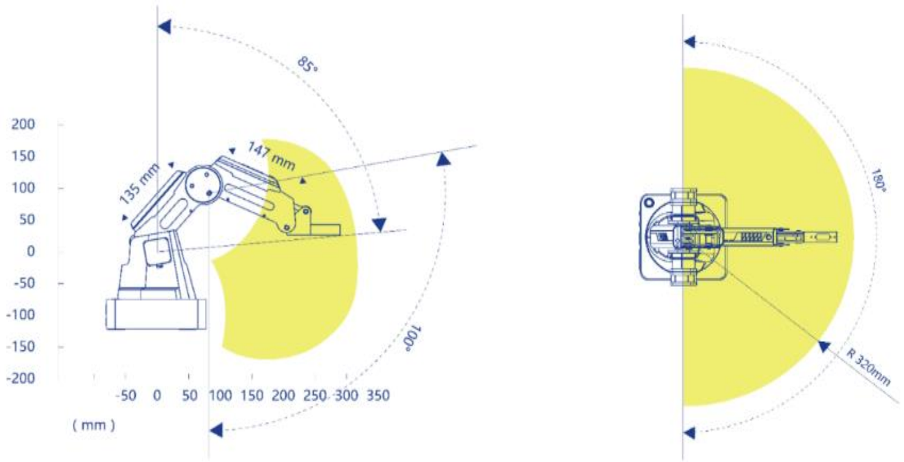


Fig 1.3 Workspace of Dobot Magician

1.1.3 Coordinate System

Dobot Magician has two types of coordinate system, the joint one and the Cartesian one, as shown below.

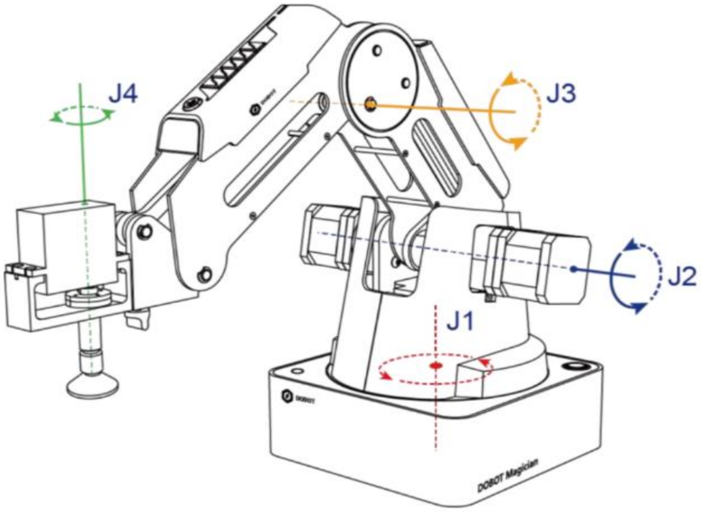


Fig 1.4: Joint Coordinate System of Dobot Magician

- Joint coordinate system: The coordinates are determined by the motion joints.
    - If the end-effector is not installed, Dobot Magician contains three joints: J1, J2, and J3, which are all the rotating joints. The positive direction of these joints is counterclockwise.
    - If the end-effector with servo is installed, such as suction cup kit, gripper kit, Dobot Magician contains four joints: J1, J2, J3 and J4, which are all the rotating joints. The positive direction of these joints is counterclockwise.
- Cartesian coordinate system: The coordinates are determined by the base.
    - The origin is the center of the three motors (Rear Arm, Forearm, base).
    - The direction of X-axis is perpendicular to the base forward.
    - The direction of Y-axis is perpendicular to the base leftward.
    - The direction of Z-axis is vertical upward, which is based on the right-hand rule.
    - The R-axis is the attitude of the servo center relative to the origin of the robotic arm, of which the positive direction is counter-clockwise. The R-axis only exists once the end-effector with servo is installed.

1.3 Objectives

The main purpose of this project is to model a 3-DOF Robotic arm and implement a PID controller for it.

- Mathematical Modelling of Robot
- Simulation Model of Robot in Simulink
- Implementing PID for the Simulink Model

CHAPTER 2
LITERATURE REVIEW

With the rise in significance of Robotic Arms across various Industries, Control Design of Robotic Arms is an important topic. Several researchers attempted to develop Mathematical Models, Simulation Models and Control Laws for similar Robotic Arms that are like Dobot Magician.

Md Rasedul Islam et al. (2019) developed forward kinematics and inverse kinematics of Dobot magician robot and built a non-linear approach for cartesian control of its end-effector position. An algorithm based on developed inverse kinematic was developed to conversion Dobot's Cartesian variables into joint variables. This conversion facilitated to obtain Cartesian control of Dobot Magician robot.

Mahmoud Khaled (2009) is a Robotic Arm that can identify different objects located in its workspace according to their shape and color and places the objects in a certain location with the help of a video camera as visual feedback. The arm has 4 Degrees of Freedom (DOF) with a DC Motor at each joint. The DC-Motor is position-controlled using a PID controller to position the link at the required angle; the feedback of the angle is measured using rotary-encoders.

Mohammad Rezwan Sheikh (2019) analyzed forward kinematics and inverse kinematics, besides studying about PID and computed torque control approaches. In this research, alphabets and numbers are coded using an object-oriented programming language (C#) to make the learning of alphabet, numbers, words, and sentence writing more fun to the children.

Jun Wu et al., proposes a novel method to evaluate the dynamic performance of the robot along joint-space directions. Based on the given ranges of end-effector velocity and acceleration in task space, the required torque along all joint-space directions is obtained based on the velocity term and the acceleration term in dynamic model, respectively.

Sheng Dong et al., presents a simplified method of dynamic equations in a generalized coordinate system, which decouples the relative motion of the front and back links of the robot joint, and maps generalized angular variables to the same angular datum. Because of the decoupling between absolute angle variables, the complexity of the coefficient matrix of the dynamic equation of the system is reduced, which facilitates the application of the actual series robot system.

Mohammad Mahdavian et al., presented trajectory generation for a 4 DOF arm of SURENA III humanoid robot with the purpose of optimizing energy and avoiding a moving obstacle. The main goal of optimization is to reduce the consumed energy by the arm in a movement between two known points in a specified time frame to avoid the moving obstacle. Initial and final velocities of the arm are taken as zero.

Thomas Lens et al., presents the modeling of the lightweight BioRob robot arm with series elastic actuation for simulation and controller design.

War War Naing et al., presented a robotic arm with three joints, two links and three Dc motors. An arduino microcontroller is used to generate the required angular position of the robot joints. In this research, the link length of the robot arm is calculated to enable carrying the desire object weight. The position of the robot arm end effector is calculated with kinematic modeling method which include forward and inverse kinematic. Robotic tool box is used to task the position of the robot arm using forward and inverse kinematic. PID control method is used for accurate position of the end effector. In this research the gain of the PID controller is tuned by using the Ziegler-Nichol method. The output position of the robot arm are shown in MATLAB simulation. Forward and inverse kinematics result are also analyzed in matlab.

K.Renuka et al., has done kinematic and dynamic modelling and PID control of 3-DOF robotic arm. The kinematic model is obtained by relating the end effector’s position and orientation. Manipulator dynamics was used to get the relationship between joint actuator torques and motion of links. The control of the robot requires the knowledge of the mathematical model and is obtained from basic physical laws governing robot dynamics. Both linear control schemes and nonlinear controllers have been employed for the manipulator control, and the control is done using PID controller.

Hossein Sadegh Lafmejani1 et al., has studied the modeling, simulation and control of 3 degrees of freedom articulated robotic manipulator. Firstly, the kinematics and dynamics equations of the manipulator were obtained by using the Lagrange method. In order to validate the analytical model of the manipulator, the model simulated in the simulation environment of Matlab was compared with the model simulated with the SimMechanics toolbox. A sample path has been designed for analyzing the tracking subject. The system has been linearized with feedback linearization and then a PID controller was applied to track a reference trajectory. Finally, the control results have been compared with a nonlinear PID controller.

E. A. Sallam and W. M. Elawady proposed a fuzzy pre-compensation of a fuzzy self tuning fuzzy PID controller to enhance the PID-type fuzzy controller performance for the control of rigid planar robot manipulators. The proposed control scheme consisted of a fuzzy logic-based pre-compensator followed by a fuzzy self tuning fuzzy PID controller. In the fuzzy self tuning fuzzy PID controller, a supervisory hierarchical fuzzy controller (SHFC) is used for tuning the input scaling factors of the fuzzy PID controller according to the actual tracking position error and the actual tracking velocity error. Numerical simulations using the dynamic model of a three DOF planar rigid robot manipulator with uncertainties showed the effectiveness of the approach in set point tracking problems. Results showed that the proposed controller had superior performance compared to a conventional fuzzy PID controller.

In Alassar et al. introduced a fuzzy logic controller (FLC) for manipulating a 5-DOF robotic manipulator based on the independent joint control approach. The proposed controller was developed to overcome the shortcomings of the linear PID controller. Simulation was performed using MATLAB/Simulink to compare the performance of the controller in terms of time response. The results obtained were promising as the FLC was used to control complex nonlinear dynamic systems. However, the design of the FLC highly depended upon expert knowledge or trial and error. Furthermore, it didn’t guarantee stability and robustness due to the linguistic expressions of fuzzy control.

Changxiang Fan, Fan Zeng, Shouhei Shirafuji, Jun Ota developed a three-mobile-robot system to cooperatively lift, support, and transport a large object on a mobile robot system. To facilitate the manipulation involved with loading an object onto the robots, where the robots must maintain firm contact with the object and anchor at its location, they designed an adaptable mechanism for the object-loading platform and a liftable brake to ensure that the robot remains stationary when necessary. Furthermore, each robot is designed to act as an omnidirectional wheel; therefore, all three robots can work as an omnidirectional block when collectively loading an object. The kinematic constraints of the object–robot system and forward kinematics of the robots' cooperative motion are proposed, and experiments are conducted to confirm the designed mechanisms and confirm that the robots can load an object and cooperatively transport it along the expected trajectory.

A. Vivas, P. Poignet, F. Marquet, F. Pierrot, and M. Gauti, dealt with the experimental identification of the dynamic parameters of parallel machines. The dynamic parameters are estimated by using the weighted least squares solution of an over determined linear system obtained from the sampling of the dynamic model along a closed loop exciting trajectory. Experimental results are exhibited for the H4 robot, a fully parallel structure providing 3 degrees of freedom (DOF) in translation and 1 DOF in rotation. A comparative study is performed depending on the available measurements, i.e., different sensor locations (motor, end effector).

In Ohri et al., presented a comparison of the conventional PID controller and sliding mode control (SMC) for robotic manipulators. Both techniques have good performance, but the SMC is insensitive to parametric variations and has good disturbance rejection. The simulation results of the study showed that the performance of the SMC is better than PID under both sine and cosine trajectory. As the payload is changed or during uncertainty, the tracking error in the PID controller is increased, but in the case of SMC, the performance remained the same, which clearly shows that the SMC is more robust than the PID controller. In this paper, the authors assumed that the actuators do not have dynamics of their own, and arbitrary forces can be commanded at the joint of the robotic manipulator for simplicity. The results obtained were promising, but the authors didn't consider the actuator dynamics and ignored the effects of joint friction which brought about low controller performance.

CHAPTER 3
MATHEMATICAL MODELING

3.1 KINEMATIC MODELING

A mathematical model usually describes a system by a set of variables and a set of equations that establish relationships between the variables. Mathematical models are an essential part for simulation and design of control systems. The purpose of the mathematical model is to be a simplified representation of reality, to mimic the relevant features of the system being analyzed. Through mathematical modeling phenomena from real world are translated into a conceptual world. This process is initiated by observing the phenomena, applying a mathematical model to it and predict its behavior through simulation. It helps in analyzing the dynamic characteristics of the system. Mathematical models may assume many different forms. Depending on the system and the particular circumstances, one mathematical model may be better suited than other models. For example, in optimal control problems, it is advantageous to use state-space representations. Mathematical Modeling of Control Systems transient-response or frequency-response analysis of single-input, single-output, linear, time-invariant systems, the transfer-function representation may be more convenient than any other. Once a mathematical model of a system is obtained, various analytical and computer tools can be used for analysis and synthesis purposes.

Robotic manipulators are highly coupled multi-input multi-output (MIMO) nonlinear systems with uncertainties and highly time-varying dynamic capabilities. These characteristics make the trajectory control of a robotic manipulator very challenging. In any control system, stability and robustness are the two important characteristics that a robot manipulator requires. Consequently, it is very important to model and evaluate robot manipulators by utilizing proper control methods. In industrial applications, typical PID controllers are used to control such complex systems. PID controllers are easy to implement and require the tuning of only three parameters and have relatively acceptable tracking performance for a robot.

The first important step in controlling any physical system is to acquire a complete and accurate mathematical model of the system. This model is useful for simulation of the system before implementing the system into a real-time application. In this section, the mathematical equations for the kinematic and dynamic model of 3 DOF industrial robotic manipulator is discussed. The kinematic analysis has been classified into two types, forward and inverse kinematics, and the dynamic behavior of the manipulator concerned with the relationship between joint actuator torques and motion of links for the design of the control algorithms. Finally, the overall nonlinear system model or manipulator dynamic and actuator dynamics are developed.

Kinematics studies the motion of bodies without considering the forces or moments that cause the motion. Robot kinematics refers to the analytical study of the motion of a robot manipulator. Formulating the suitable kinematics models for a robot mechanism is very crucial in analyzing the behaviour of an industrial manipulator. There are mainly two different spaces used in kinematics modelling namely, Cartesian space and joint space. The transformation between two Cartesian coordinate systems can be decomposed into rotation and translation.

Robot kinematics can be divided into forward kinematics and inverse kinematics. Forward kinematics problem is straightforward and there is no complexity deriving the equations. Hence, there is always a forward kinematics solution of a manipulator. Inverse kinematics is a much more difficult problem than forward kinematics. The solution of the inverse kinematics problem is computationally expensive and generally takes a very long time in the real time control of manipulators. Singularities and nonlinearities make the problem more difficult to solve. Hence, only for a very small class of kinematically simple manipulators have complete analytical solutions. The relationship between forward and inverse kinematics is illustrated in Figure.

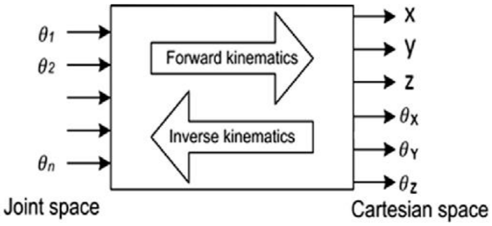


Fig 3.1: Relationship between forward and inverse kinematics

3.1.1 Forward Kinematics

Forward kinematics calculates the position and orientation of the robot end-effector from the joint coordinates by using a transformation matrix. This part is very important to calculate the position and/or orientation error to calculate the controllers qualify. There are many ways to represent rotation: Euler angles, Gibbs vector, Cayley-Klein parameters, Pauli spin matrices, axis and angle, orthonormal matrices, and Hamilton 's quaternions. Of all these transformations homogeneous transformation of matrices is the most followed method. DH method has been used to assign the co-ordinate frame to the robotic arm. showed that a general transformation between two joints requires four parameters. These parameters are known as Denavit-Hartenberg (DH) parameters. They are:

1.Link Length (a_{i-1}): the length measured along Xi-1, from axis Zi-1 to axis Zi

2.Link Twist (α_{i-1}): the angle measured about Xi-1, from axis Zi-1 to axis Zi

3.Link Offset (d_i): the distance measured along the axis Zi; from Xi-1 to axis Xi

4.Joint Angle (θ_i): the angle measured about Zi, from Xi-1 to axis Xi

To obtain the DH parameters, co-ordinate frames (i.e., the link frames which map between the successive axes of rotation) are assumed to have coincided with the joint axes of rotation and have the same order.

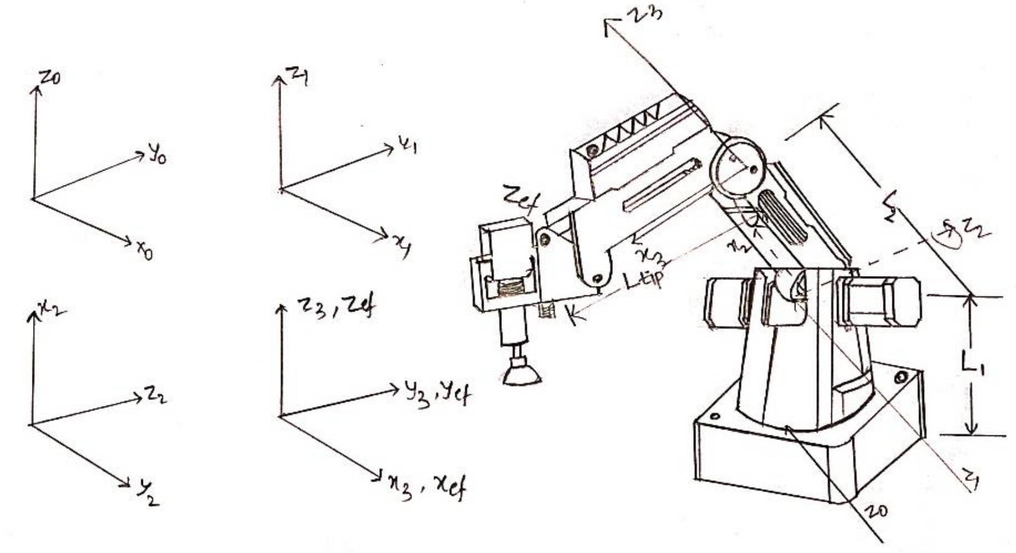


Fig 3.2: Assigning co-ordinate frames to Dobot Magician

Frame (i)	***Link length (a_{i-1})***	***Link Twist (α_{i-1})***	***Link Offset (d_i)***	***Joint Angle (θ_i)***
1	*0*	*0*	$\boldsymbol{L_1}$	$\boldsymbol{\theta_1}$
2	*-π/2*	*0*	*0*	$\boldsymbol{\theta_2}$*- π/2*
3	*0*	$\boldsymbol{L_2}$	*0*	$\boldsymbol{\theta_3}$
4	*0*	$\boldsymbol{L_{tip}}$	*0*	*0*

Table (1) DH parameters

General form of a link transformation that relates frame $\{i\}$ relative to the frame $\{i-1\}$ is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} {}^{i-1}_{i}R & {}^{i-1}_{i}P \\ 0 & 1 \end{bmatrix} $$

Where ${}^{i-1}_{i}R$ is the rotation matrix (3*3) that describes frame $\{i\}$ relative to frame $\{i-1\}$ and can be expressed as:

$$ {}^{i-1}_{i}R = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} \end{bmatrix} $$

${}^{i-1}_{i}P$ is the vector that locates the origin of frame $\{i\}$ relative to frame $\{i-1\}$ and can be expressed as:

$$ [\alpha_{i-1} \quad -\sin\alpha_{i-1}\, d_i \quad \cos\alpha_{i-1}\, d_i]^T $$

$$ T_1^0 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_2^1 = \begin{bmatrix} \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ T_3^2 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{ef}^3 = \begin{bmatrix} 1 & 0 & 0 & L_{tip} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The homogenous transformation matrix that represents frame {ef} with respect to the frame {0} can be obtained by multiplying individual transformation matrices.

$$\boldsymbol{T_{ef}^{0}} = \begin{bmatrix} \boldsymbol{T_1^0} & \boldsymbol{T_2^1} & \boldsymbol{T_3^2} & \boldsymbol{T_{ef}^3} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r_{11}} & \boldsymbol{r_{12}} & \boldsymbol{r_{13}} & \boldsymbol{P_x} \\ \boldsymbol{r_{21}} & \boldsymbol{r_{22}} & \boldsymbol{r_{23}} & \boldsymbol{P_y} \\ \boldsymbol{r_{31}} & \boldsymbol{r_{32}} & \boldsymbol{r_{33}} & \boldsymbol{P_z} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

The equation obtained from this transformation matrix is known as the forward kinematic equation and is given below.

$r_{11} = \cos\theta_1 \cos\theta_2 \cos\theta_3 + \cos\theta_1 \cos\theta_2 \sin\theta_3$

$r_{21} = \cos\theta_1 \cos\theta_2 \cos\theta_3 - \cos\theta_1 \sin\theta_2 \sin\theta_3$

$r_{13} = -\sin\theta_1$

$r_{21} = \sin\theta_1 \sin\theta_2 \cos\theta_3 + \sin\theta_1 \cos\theta_2 \sin\theta_3$

$r_{22} = \sin\theta_1 \cos\theta_2 \cos\theta_3 - \sin\theta_1 \sin\theta_2 \sin\theta_3$

$r_{23} = \cos\theta_1$

$r_{31} = \cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3$

$r_{32} = -(\cos\theta_2 \sin\theta_3 - \sin\theta_2 \cos\theta_3)$

$r_{33} = 0$

$P_x = L_2 \cos\theta_1 \sin\theta_2 + L_{tip} \cos\theta_1 (\cos\theta_2 \sin\theta_3 + \sin\theta_2 \cos\theta_3)$

$P_y = L_2 \sin\theta_1 \sin\theta_2 + L_{tip} \sin\theta_1 (\cos\theta_3 \sin\theta_2 + \sin\theta_3 \cos\theta_2)$

$P_z = L_1 + L_{tip}(\cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3) + L_2 \cos\theta_2$

With the joint variable of each joint $(\theta_1, \theta_2, \theta_3)$, using this forward kinematic equation, the position and orientation of frames were determined with respect to the reference frame.

3.1.2 Inverse Kinematics

The conversion of the position and orientation of a manipulator end-effector from Cartesian space to joint space is called inverse kinematics. Cartesian space includes orientation matrix and position vector, and joint space is represented by joint angles. Tasks to be performed by a manipulator are in cartesian space, whereas actuators work in joint space. Therefore, to know the joint angles we convert the position and orientation of end effector from cartesian space to joint space with the help of inverse kinematics equations. The inverse kinematics solution for a robotic manipulator is computationally complex compared to direct kinematics. It is often difficult to find a closed-form solution due to the nonlinear nature of the equations to solve. Furthermore, an inverse kinematics problem for a redundant robotic manipulator is much more complex since it gives infinite solutions.

From the forward kinematics equations, the position of end effector with respect to base is:

$$P_{ef}^{0} \;=\; \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$L_2 \cos\theta_1 \sin\theta_2 + L_{tip} \cos\theta_1 \,(\cos\theta_2 \sin\theta_3 + \sin\theta_2 \cos\theta_3) = x$$

$$L_2 \sin\theta_1 \sin\theta_2 + L_{tip} \sin\theta_1 \,(\cos\theta_3 \sin\theta_2 + \sin\theta_3 \cos\theta_2) = y$$

$$L_1 + L_{tip}(\cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3) + L_2 \cos\theta_2 = z$$

From trigonometric identities,

$$\cos\theta_2 \sin\theta_3 + \sin\theta_2 \cos\theta_3 = \sin(\theta_2 + \theta_3) = S_{23}$$

$$\cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3 = \cos\,(\,\theta_2 + \theta_3) = C_{23}$$

From the above identities we have,

$$L_2 \sin\theta_2 + L_{tip} \sin(\theta_2 + \theta_3) = \frac{x}{\cos\theta_1} \quad \text{.....(1)}$$

$$L_2 \sin\theta_2 + L_{tip} \sin(\theta_2 + \theta_3) = \frac{y}{\sin\theta_1} \text{........(2)}$$

$$L_1 + L_{tip} \cos\,(\,\theta_2 + \theta_3) + L_2 \cos\theta_2 = z\text{.....(3)}$$

By doing (2)-(1), we have

$$0 = \text{y}/\sin\theta_1 - \text{x}/\cos\theta_1$$

$$\text{x}/\cos\theta_1 = y/\sin\theta_1 \quad \text{.....(4)}$$

$$\frac{\sin\theta_1}{\cos\theta_1} = y/x = \tan\theta_1$$

$$\theta_1 = \tan^{-1} y/x$$

Rearranging (3) we get,

$$L_2 \cos\theta_2 + L_{tip} \cos\,(\,\theta_2 + \theta_3) = z - L_1 \;\ldots.(5)$$

squaring both equation (1) & (5)and adding we get,

$$L_2^2(S_2^2 + C_2^2) + L_{tip}^2(S_{23}^2 + C_{23}^2) + 2L_2L_{tip}(S_2S_{23} + C_2C_{23}) = \frac{x^2}{C_1^2} + \;(z - L_1)^2$$

Where,

$$S_2^2 + C_2^2 = 1$$

$$S_{23}^2 + C_{23}^2 = 1$$

$$p^2 = \;(x/C_1)^2 + \;+ (z - L_1)^2 = (x + y)^2 + (z - L_1)^2$$

$$C_3 = \frac{p^2 - L_2^2 - L_{tip}^2}{2L_2L_{tip}}$$

$$S_3 = \mp\sqrt{1 - C_3^2}$$

$$\theta_3 = \tan^{-1}\left(\frac{\mp\sqrt{1 - C_3^2}}{C_3}\right)$$

Multiplying (1) by S_2& (5) by C_2we have,

$$L_2S_2^2 + L_{tip}S_2S_{23} = \left(\frac{x}{C_1}\right)S_2 \; \ldots\ldots (6)$$

$$L_2C_2^2 + L_{tip}C_2C_{23} = (z - L_1)C_2 \ldots . (7)$$

Adding equation (6) & (7) , we get,

$$L_2S_2^2 + L_2C_2^2 + L_{tip}C_2C_{23} + L_{tip}S_2S_{23} = \left(\frac{x}{C_1}\right)S_2 + (z - L_1)C_2$$

$$L_2 + L_{tip}C_3 = \left(\frac{x}{C_1}\right)S_2 + (z - L_1)C_2$$

Let

$L_2 + L_{tip}C_3 = r$; $(z - L_1) = a;$ $\frac{x}{C_1} = b$

$r = bS_2 + aC_2$

$r = acos\,\theta_2 + b\,sin\,\theta_2$ $\ldots .(8)$

Equation (8) is a transcendental equation; therefore we can solve it using half angle property of trigonometry.

Let, $u = tan\frac{\theta_2}{2}$ $cos\,\theta_2 = \frac{1-u^2}{1+u^2}$ $sin\,\theta_2 = \frac{2u}{1+u^2}$

Therefore equation (8) can be written as,

$$r = a\left(\frac{1-u^2}{1+u^2}\right) + b\left(\frac{2u}{1+u^2}\right)$$

$$(r + a)u^2 - 2bu + (r-a) = 0$$

$$u = \frac{2b \pm\sqrt{4b^2-4(r+a)(r-a)}}{2(r+a)}$$

$$u = \frac{2b\pm2\sqrt{b^2+a^2-r^2}}{2(r+a)}$$

$$tan\frac{\theta_2}{2} = \frac{b \pm \sqrt{b^2 + a^2 - r^2}}{r + a}$$

$$\theta_2 = 2\,tan^{-1}\left(\frac{b \pm \sqrt{b^2 + a^2 - r^2}}{r + a}\right)$$

Therefore, we have a total four solutions.

$$[\theta_1, \theta_2, \theta_3] = \left[tan^{-1}\frac{y}{x}\,, \quad 2\,tan^{-1}\left(\frac{b + \sqrt{b^2 + a^2 - r^2}}{r + a}\right)\;,\quad tan^{-1}\left(\frac{+\sqrt{1 - C_3^2}}{C_3}\right)\right]$$

$$[\theta_1, \theta_2, \theta_3] = \left[tan^{-1}\frac{y}{x}\,, \quad 2\,tan^{-1}\left(\frac{b + \sqrt{b^2 + a^2 - r^2}}{r + a}\right)\;,\quad tan^{-1}\left(\frac{-\sqrt{1 - C_3^2}}{C_3}\right)\right]$$

$$[\theta_1, \theta_2, \theta_3] = \left[tan^{-1}\frac{y}{x}\,, \quad 2\,tan^{-1}\left(\frac{b - \sqrt{b^2 + a^2 - r^2}}{r + a}\right)\;,\quad tan^{-1}\left(\frac{+\sqrt{1 - C_3^2}}{C_3}\right)\right]$$

$$[\theta_1, \theta_2, \theta_3] = \left[\tan^{-1}\frac{y}{x}\,, \quad 2\tan^{-1}\left(\frac{b - \sqrt{b^2 + a^2 - r^2}}{r + a}\right)\;,\quad \tan^{-1}\left(\frac{-\sqrt{1 - C_3^2}}{C_3}\right)\right]$$

Using these four solutions and workspace information of Dobot Magician, an algorithm for inverse kinematic solutions were developed in MATLAB.

3.2 DYNAMIC MODELING

Dynamic equation is a mathematical formulation that describes the relationship between the forces acting on a robot and its resulting motion. It is derived from the principles of dynamics and allows engineers to model and analyze the behavior of robots in a dynamic environment.

The dynamic equation of a robot is typically represented as:

$$M(q)\ddot{q} + C(q, \dot{q}) + f(q) = \tau \tag{3.1}$$

In this equation:

- *M(q)* represents the mass matrix, which describes the inertia properties of the robot's links and joints. It relates the acceleration of the robot's joints ($\ddot{q}$) to the forces acting on them. The mass matrix is often a symmetric and positive definite matrix.
- *$C(q,\dot{q})$* denotes the Coriolis and centrifugal forces. These forces arise from the motion of the robot and its joints. They account for the interaction between the robot's links and joints and depend on both the joint positions *(q)* and velocities *($\dot{q}$)*.

- *G(q)* corresponds to the gravitational forces acting on the robot. It captures the effect of gravity on the robot's motion and depends solely on the joint positions *(q).*
- τ represents the vector of external forces and torques applied to the robot. These forces can come from interactions with the environment, input commands, or other external factors.

For Dobot magician we have considered a simplified 3-DOF Robot like below:

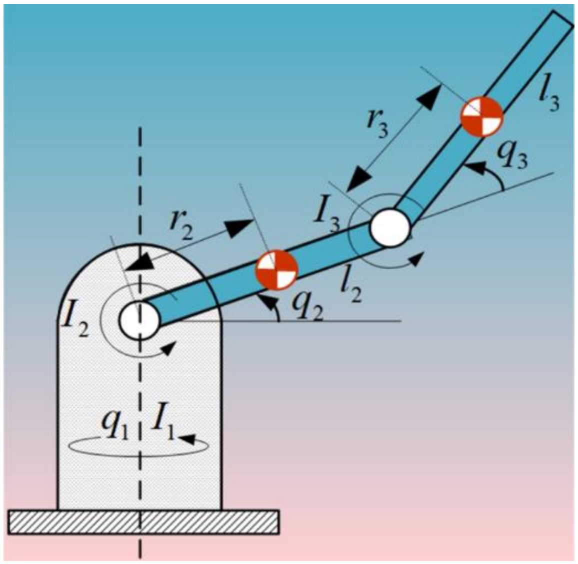


Fig 3.3 A simplified 3-DOF Robot

The dynamic equation for the above robot is as follows:

$$M(q) = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \qquad b(\theta, \dot{\theta}) = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

$$f(q) = \begin{bmatrix} 0 \\ b_1 \cos q_2 + b_2 \cos(q_2 + q_3) \\ b_2 \cos(q_2 + q_3) \end{bmatrix}$$

$$m_{11} = a_1 \cos {q_2}^2 + a_2 \cos(q_2 + q_3)^2 + 2a_3 \cos q_2 \cos(q_2 + q_3) + I_1$$

$$m_{22} = a_1 + a_2 + 2a_3 \cos q_3 + I_2$$

$$m_{33} = a_2 + I_3$$

$$m_{23} = m_{32} = a_2 + a_3 \cos q_3$$

$$m_{12} = m_{21} = m_{13} = m_{31} = 0$$

$$b_{11} = \frac{-1}{2}\, a_1\dot{q_2}\, sin\, 2q_2 - \frac{1}{2} a_2(\dot{q_2} + \dot{q_3})\, sin(2q_2 + 2q_3) - a_3\, \dot{q_2}\, sin(2q_2 + q_3) - a_3\, \dot{q_3}\, cos\, q_3 + I_2$$

$$b_{12} = \frac{-1}{2}\, a_1\dot{q_1}\, sin\, 2q_2 - \frac{1}{2} a_2\dot{q_1}\, sin(2q_2 + 2q_3) - a_3\, \dot{q_1}\, sin(2q_2 + q_3)$$

$$b_{13} = \frac{-1}{2}\, a_2\dot{q_1} \sin(2q_2 + 2q_3) - a_3\, \dot{q_1} \cos q_2 \sin(q_2 + q_3)$$

b_{21}=$-b_{12}$

$b_{22} = -a_3\dot{q_3} \sin q_3$

b_{23} = -$a_3(q_2{}^{\cdot} + \ \dot{q_3}) \sin q_3$

b_{31} = -b_{13}

$b_{32} = \ a_3\dot{q_2} \sin q_3$

$b_{33} = 0$

$a_1 = \ m_2r_2^2$+$m_3l_2^2$

a_2 = $m_3r_3^2$

a_3 = $m_3r_3l_2$

b_1 = $(m_2r_2 + m_3l_2)$g

$b_2 = \ m_3r_3g$

3.3 SIMULATION

We used *Robotics System Toolbox* and *SimScape Multibody Toolbox* to create a Simulink model for the Dobot Magician Robot.

3.3.1 Building the Robot in SimScape

A Basic 3D Configuration of the Robot, containing information for the dimensions of the links and the positions of the joints was constructed using *SimScape Multibody Toolbox* in the file "RobotModel.slx".

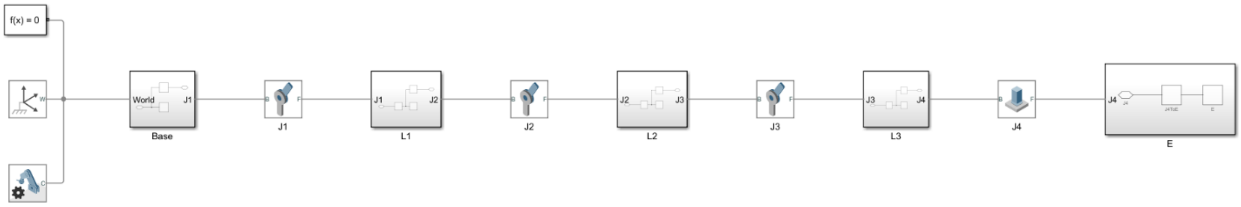


Fig 3.4 : Simple 3-DOF Robot Made using SimScape Multibody Toolbox

The world reference frame starts from Base. From there we need to translate the frame from Base to J1, J1 to L1, L1 to J2 and so on, in the following way:

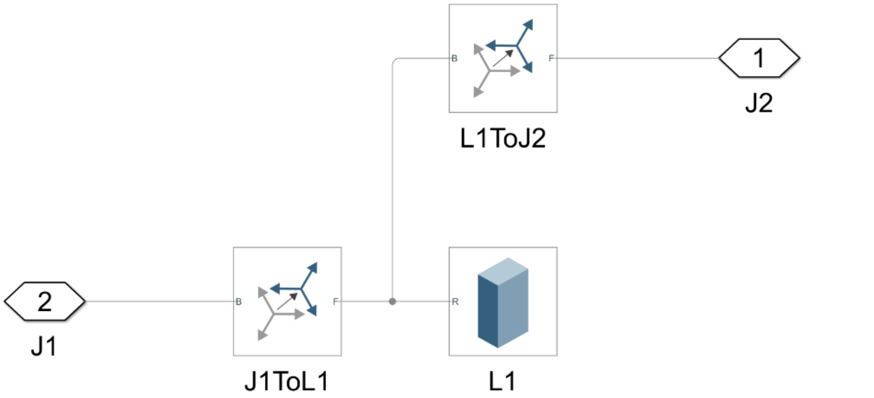


Fig 3.5: Shifting of Reference Frame from J1 to L1, L1 to J2

When run, this SimScape configuration outputs the following Robot:

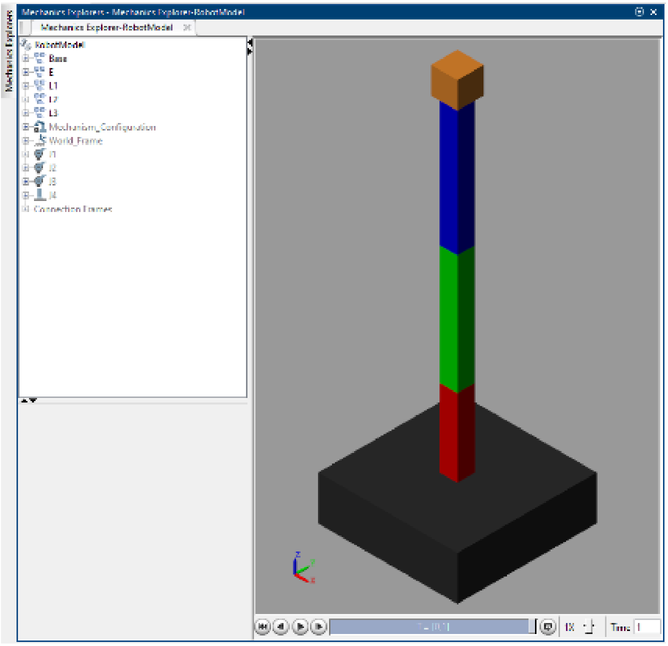
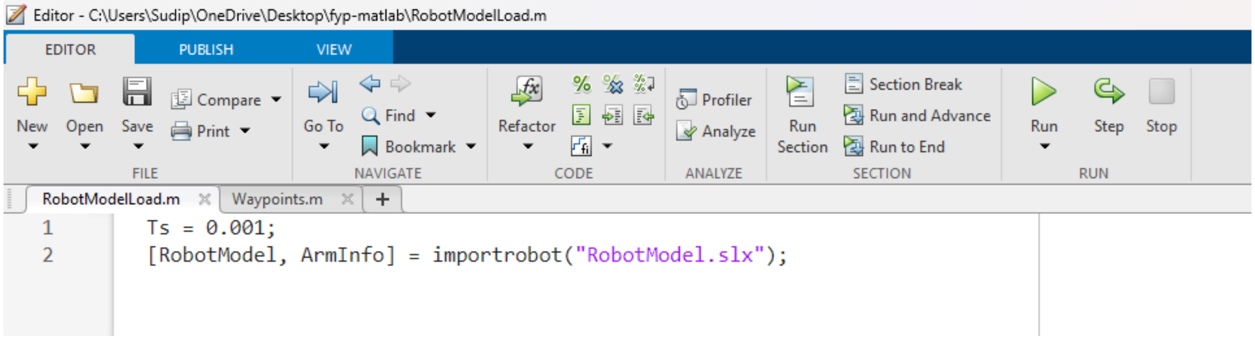


Fig 3.6: Render of the Robot in SimScape

The model was loaded into MATLAB workspace with the following code:

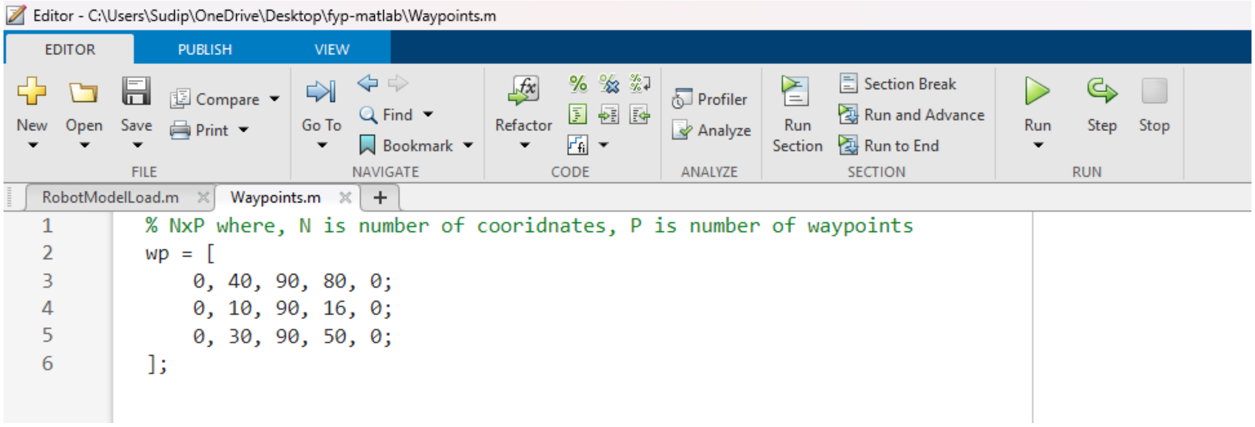


```
Ts = 0.001;
[RobotModel, ArmInfo] = importrobot("RobotModel.slx");
```

Fig 3.7: Loading RobotModel into Workspace

3.3.2 Driving the Robot through Dynamics

A set of waypoints are saved into workspace in the workspace in the variable “wp”. It is a NxP matrix where N is the number of coordinates (3 joint angles in our case), and P is the number of waypoints.



```
% NxP where, N is number of cooridnates, P is number of waypoints
wp = [
    0, 40, 90, 80, 0;
    0, 10, 90, 16, 0;
    0, 30, 90, 50, 0;
];
```

Fig 3.8: Adding waypoints "wp" to workspace

The waypoints "wp" is provided to a *Trapezoidal Velocity Profile Trajectory* block, which create a trajectory passing through our waypoints (This trajectory is also the 1st input in the Scope). The trajectory block outputs q(Angle), q_d(Angular Velocity), q_{dd}(Angular Acceleration).

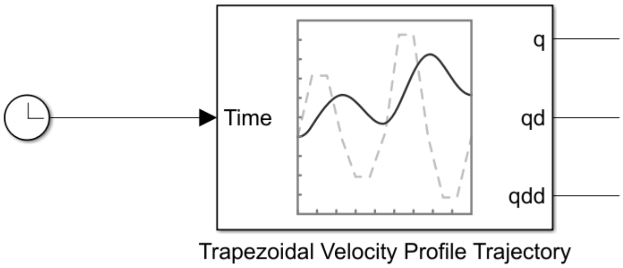


Fig 3.9: Clock connected with Trapezoidal Velocity Profile Trajectory Block

This trajectory is then supplied to the *Inverse Dynamics* block. This block is provided the *RobotModel* variable as the RigidBodyTree parameter, which describes the configuration of the robot. Using this RigidBodyTree object this block will figure out the Inverse Dynamics Equation and using the given trajectory it will output the Joint Torques to be provided to the Robot.

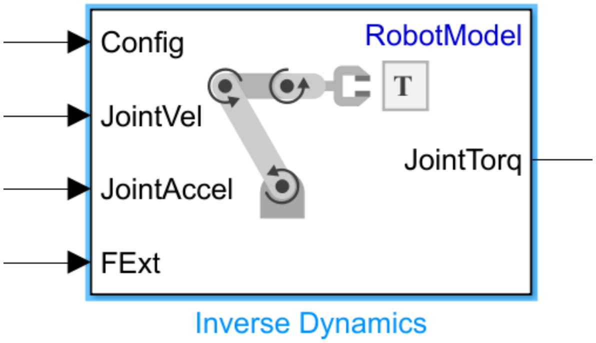


Fig 3.10: Inverse Dynamics Block

The Joint Torques are then provided to two blocks:

ForwardDynamics: It is provided with *RobotModel* as RigidBodyTree param, same as *Inverse Dynamics* block. It will generate the trajectory using the Forward Dynamics Equation. This trajectory is the desired trajectory, which will be the 2nd input to the Scope. The output of ForwardDynamics is JointAccel, so it is integrated two times to get Joint Angles.

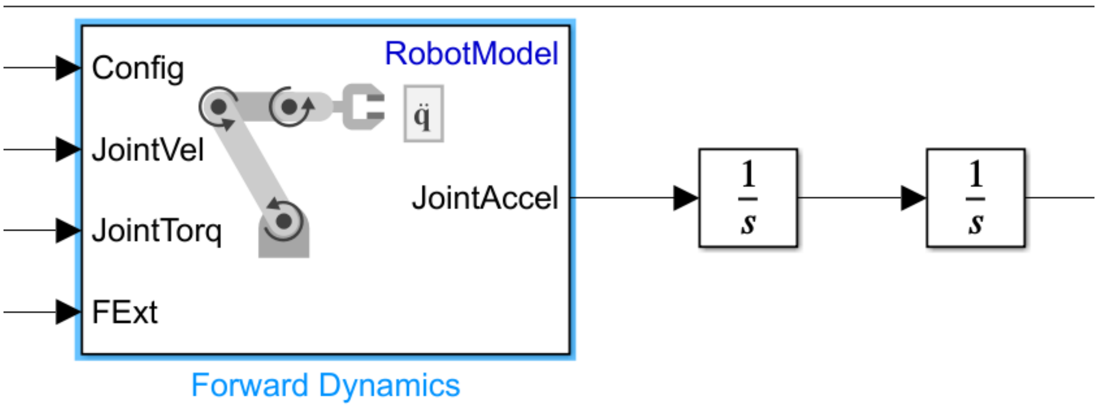


Fig 3.11: 2 Integrators connected with Forward Dynamics Block

RobotModel (with a small modification): We are providing the Joint Torques directly to each of the 3 joints. And the angles of each joint are muxed together and taken as output. This is the real trajectory. It will differ from desired trajectory because SimScape models have disturbances like friction between the links and joints, like in real world. This trajectory is the 3rd input to the scope.

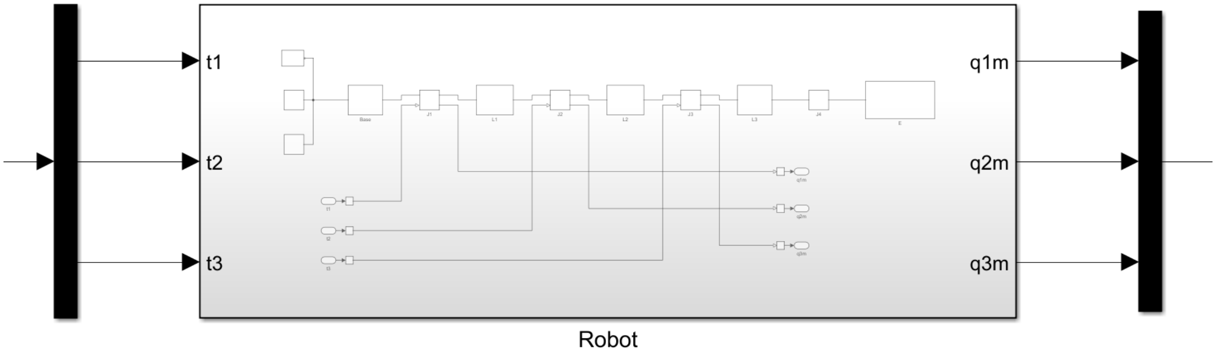


Fig 3.12: Giving Torque input and getting Angle output from the RobotModel

The final Robot Dynamics Model:

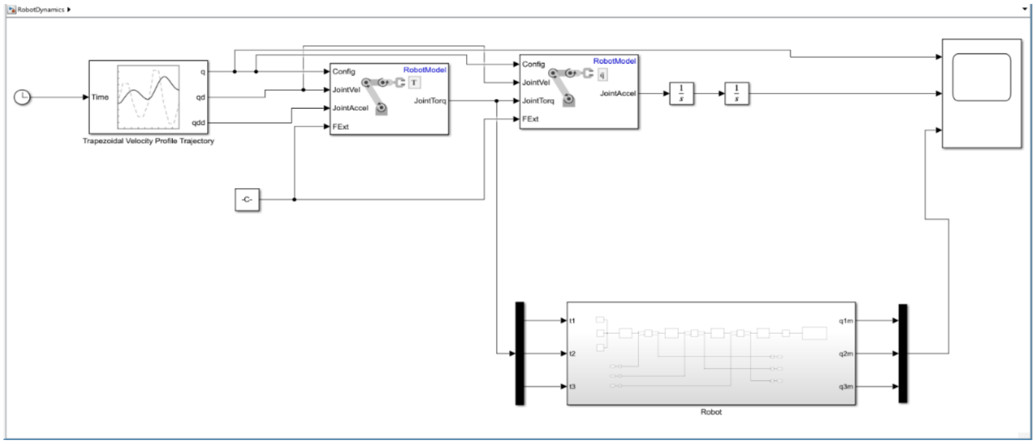


Fig 3.13: Robot Dynamics Model

CHAPTER 4
PROPORTIONAL INTEGRAL DERIVATIVE (PID) CONTROL

4.1 PROPORTIONAL (P) CONTROL

In a proportional controller, the output is directly proportional to the input. This is the most common controller action in which the output of the controller is directly proportional to its input, the input being the error signal. The basic equation for proportional controller is

Error = Set point (r)-measured value (c)

$$m = K_p e + m_0 \tag{4.1}$$

Where m is the controller output, m_0 is controller output with no error in percent and K, is a constant called the proportional gain. The output from the controller depends only on the size of the error at the instant of time concerned. The transfer function G_c(s) for the controller for system with proportional controller shown is given by

$$G_c(s) = K_p \tag{4.2}$$

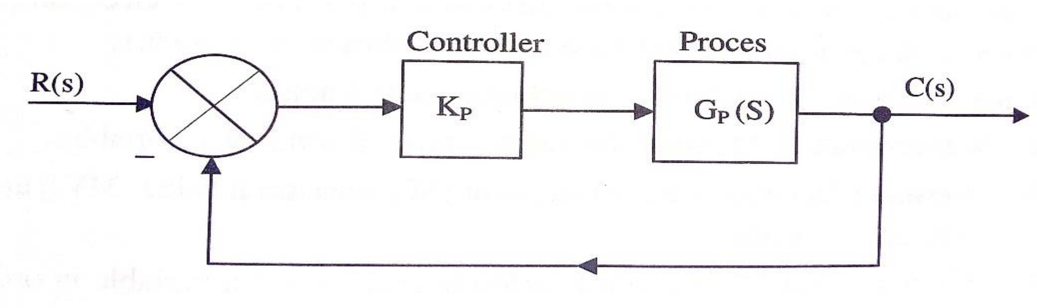


Fig 4.1 : System with Proportional Control

The controller is effectively just an amplifier with a constant gain. A high gain means a large response to an error. However, the constant gain tends to exist only over a certain range of errors, this range being called the proportional band. It is usual to express controller output as a percent of the total controller output possible. Thus, a 100% change in controller output corresponds to an error change from one extreme of the proportional band to the other. A graph of output against error would be a straight line with a slope of K_p.

Proportional gain is given by

$$K_p = \frac{100}{Proportional\ Band} \tag{4.3}$$

Characteristics of the Proportional Controller:

There are two characteristics:

1.For increasing gain, the closed-loop time constant becomes smaller, and the system response speeds up.

2. As proportional gain increases, the offset error decreases.

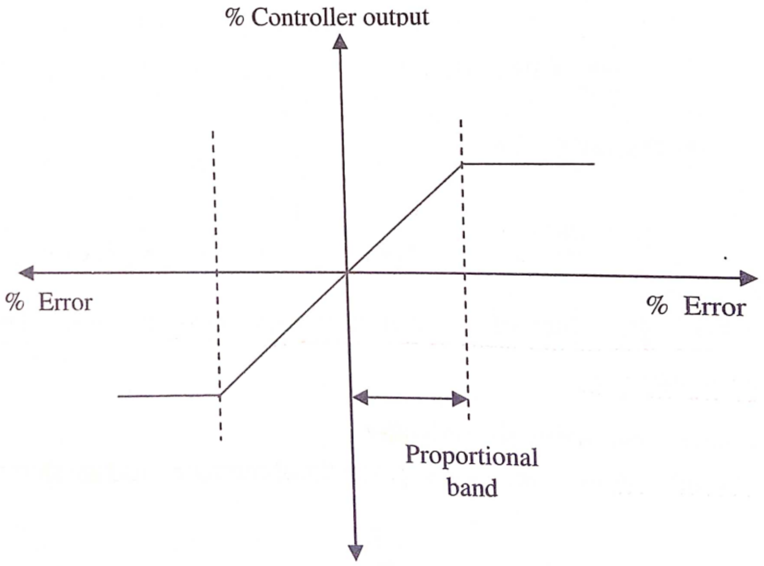


Fig 4.2: Proportional Control

4.2 INTEGRAL (I) CONTROL

Integral control is used when we want the controller action to correct from any steady and continuing offset to a desired reference signal.

In a proportional controller the steady state error cannot be unless until $K_p \neq \infty$ which is practically impossible. So at any case if error is constant if we integrate it we get ramp i.e.,without excessive increase of K_p the controller output can be increased by Integral Controller

$$\text{Output} = K_i \int_0^t e \, d\tau \qquad (4.4)$$

Where K_i, is a constant called the integral gain. The system with integral control is shown below.

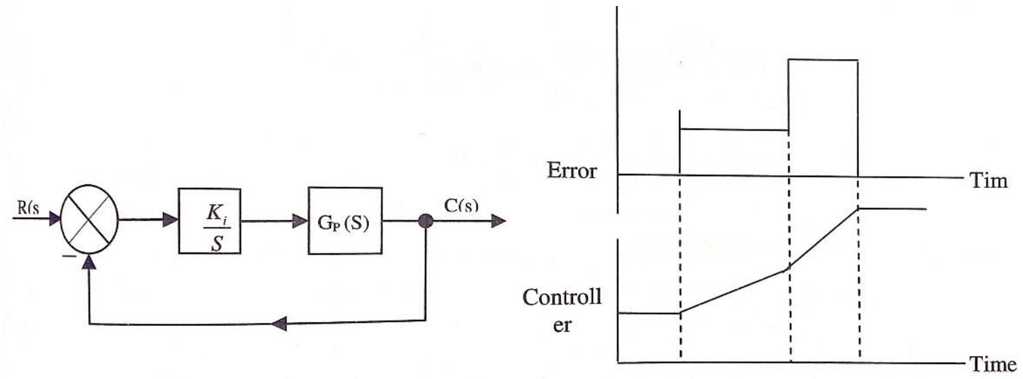


Fig 4.3: Integral Control and Response of Integral Controller

The integral action is provided by summing the error over time, multiplying that m by a gain and adding the result to the present controller output. Figure 66 shows the response of an integral controller when error is in the form of a step. The output from the controller increases at a regular rate after the start of the error. The output at any time is thus proportional to the accumulation of the effects of past errors. An advantage of an integral control is that the type number of the system is increased by 1. However, the relative stability of the system will be reduced.

4.3 DERIVATIVE (D) CONTROL

A differential controller is also known as rate or anticipatory controller. If we want the controller to have the rate of change of the error signal in the control action, we use derivative control. With the derivative form of the controller, the controller output is proportional to the rate of change of error e with time, that is

$$\text{Output} = K_d \frac{de}{dt} \tag{4.5}$$

where K_d is the derivative gain and has the units of time.

The output of a derivative controller is not dependent on the present or past error, but rather on the rate at which the error is changing. figure 6.8 illustrates derivative response to step and ramp error inputs. With a derivative control, as soon as the error signal begins, the derivative action differentiates the change and maintains a level as long as the measurement continues to change at the given rate. A derivative controller can provide a large corrective action before a large error occurs. However, if the error is constant, there is no corrective action even if the error is.
how fast the error is changing and ug the rate of change to produce a con action that will reduce the expected error. Derivative control must be used with great care and usually with small gain as a rapid change of error can case large, sudden changes of controller output.

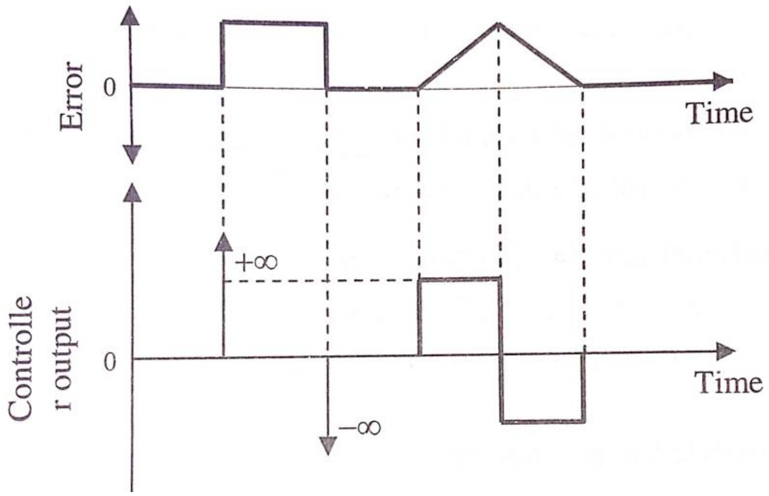


Fig 4.4: Derivative control

The essential characteristics of a derivative controller are

1. the controller output is zero if the error is zero or constant
2. the controller gives an output if there is an error changing in time

4.4 PROPORTIONAL INTEGRAL DERIVATIVE (PID) CONTROL

The PID controller mode is a combination of the proportional, integral, and derivative control modes. The most powerful control action combines all three modes as shown

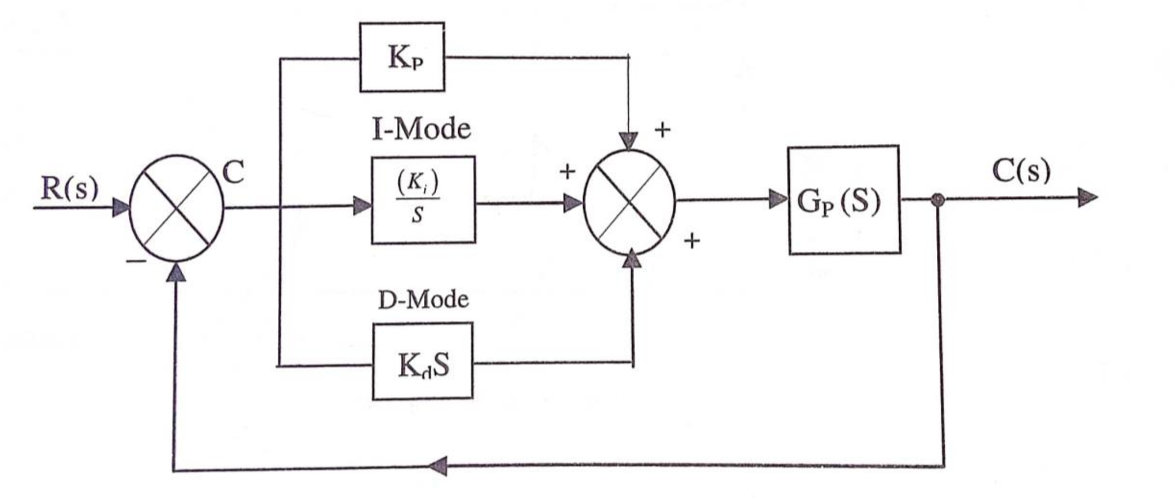


Fig 4.5: PID controller

The analytical expression can be written as

$$V_0 = K_p e + K_i \int_0^t e \, d\tau + K_d \frac{de}{dt} \quad (4.6)$$

The output is given by,

$$\text{Output} = K_p e + \int_0^t e \, d\tau + K_d \frac{de}{dt} \quad (4.7)$$

Therefore, the transfer function is

$$\frac{V_0(s)}{V_e(s)} = K_p + \frac{K_i}{s} + K_d s = G_c(s) \tag{4.8}$$

Let us take integral time constant as τ

$$\tau_i = \frac{K_p}{K_i} \tag{4.9}$$

And derivative time constant as

$$\tau_d = \frac{K_d}{K_p} \tag{4.10}$$

Therefore,

$$G_c(s) = K_p\left[1 + \frac{K_i}{K_p s} + \frac{K_d}{K_p} s\right] \tag{4.11}$$

Transfer function is

$$G_0(s) = G_c(s)G_p(s) = K_p\left[1 + \frac{1}{\tau_1 s} + \tau_d s\right] G_p(s) \tag{4.12}$$

Open loop transfer function is

$$G_0(s) = K_p\left[\frac{\tau_d\tau_1 s^2 + \tau_1 s + 1}{\tau_1 s}\right] G_p(s) \tag{4.13}$$

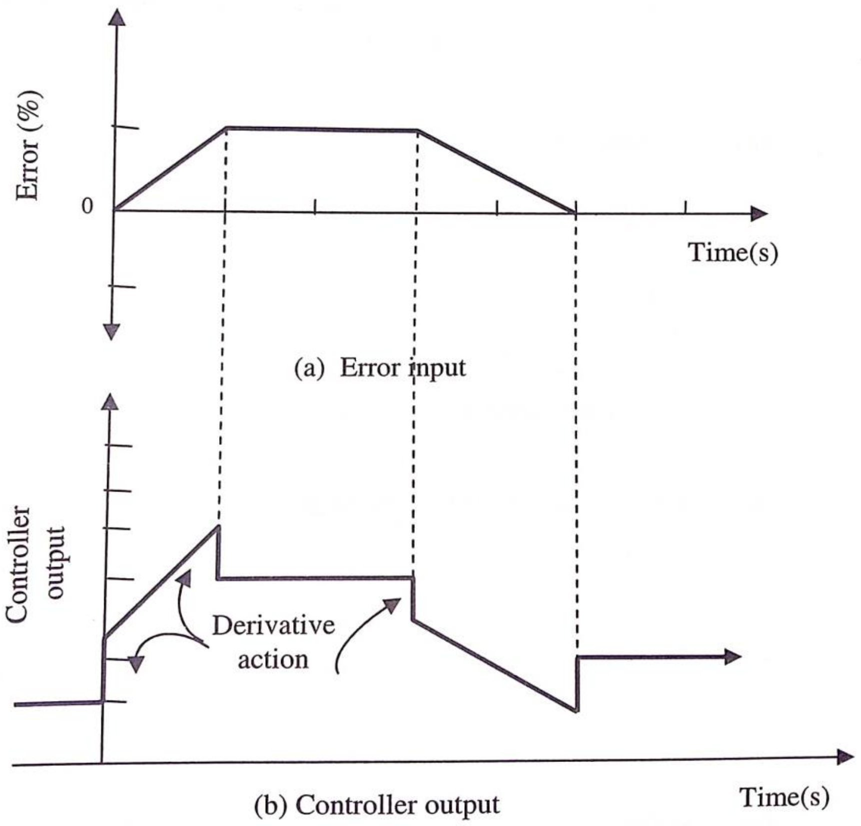
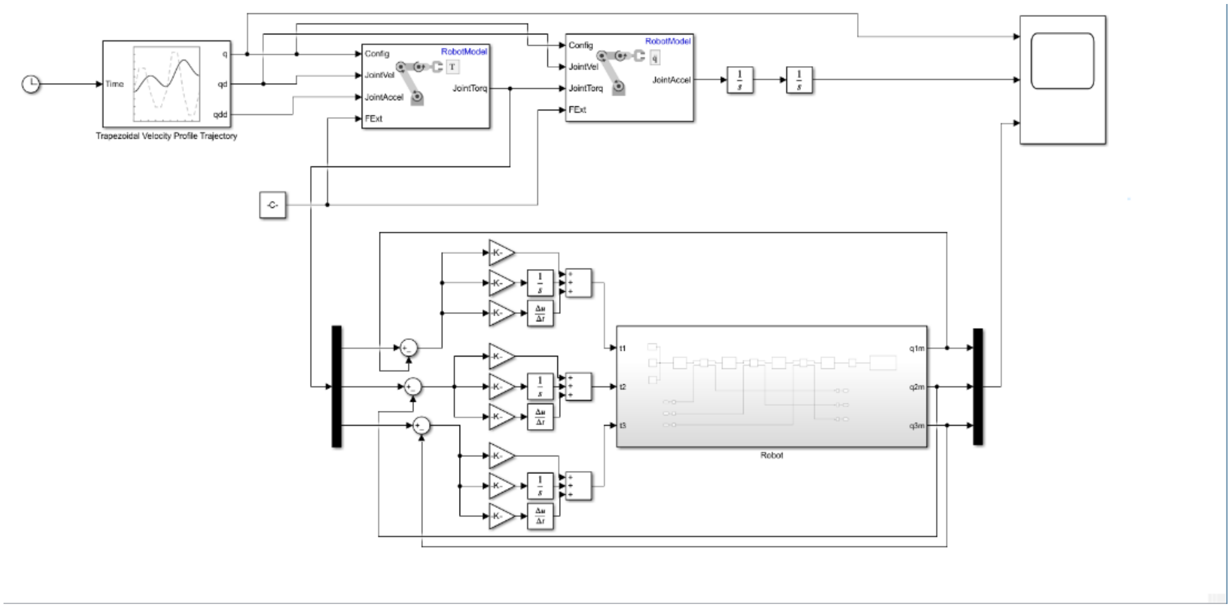


Fig 4.6: Response of PID Controller

Note that the open loop transfer function of a PID controller has increased the number of zeros by "2" and number if poles by "1". The system type is increased by "1". The section of variables K_p, K_i and K_d enables the locations of the poles and zeros introduced by the controller to be determined and hence affect the stability of the control system.

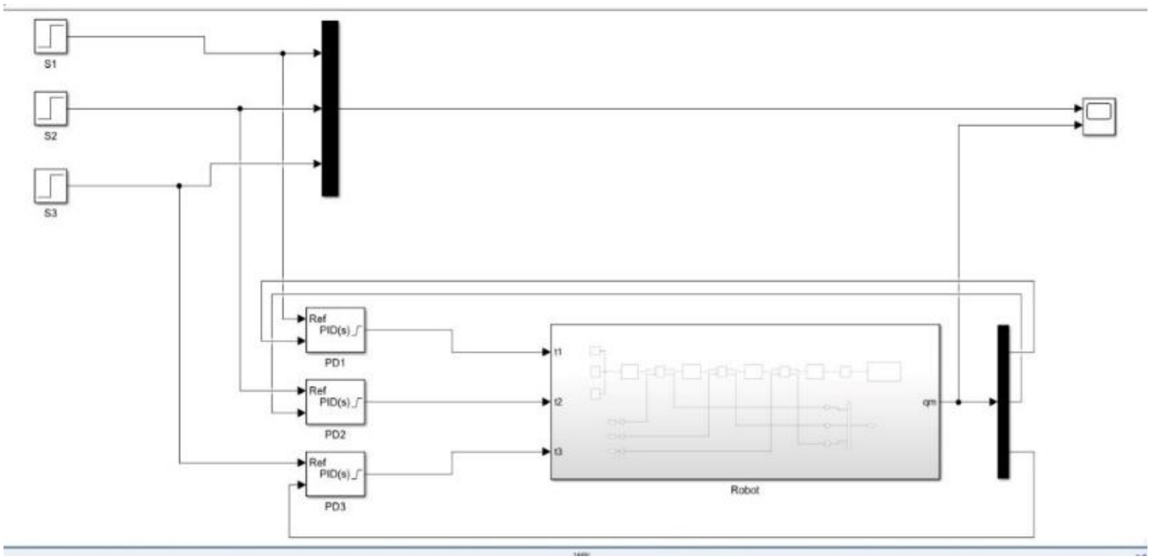
This mode eliminates the offset of the proportional mode and provides the fastest response.

Design and Implementation of PID controller in Simulink:

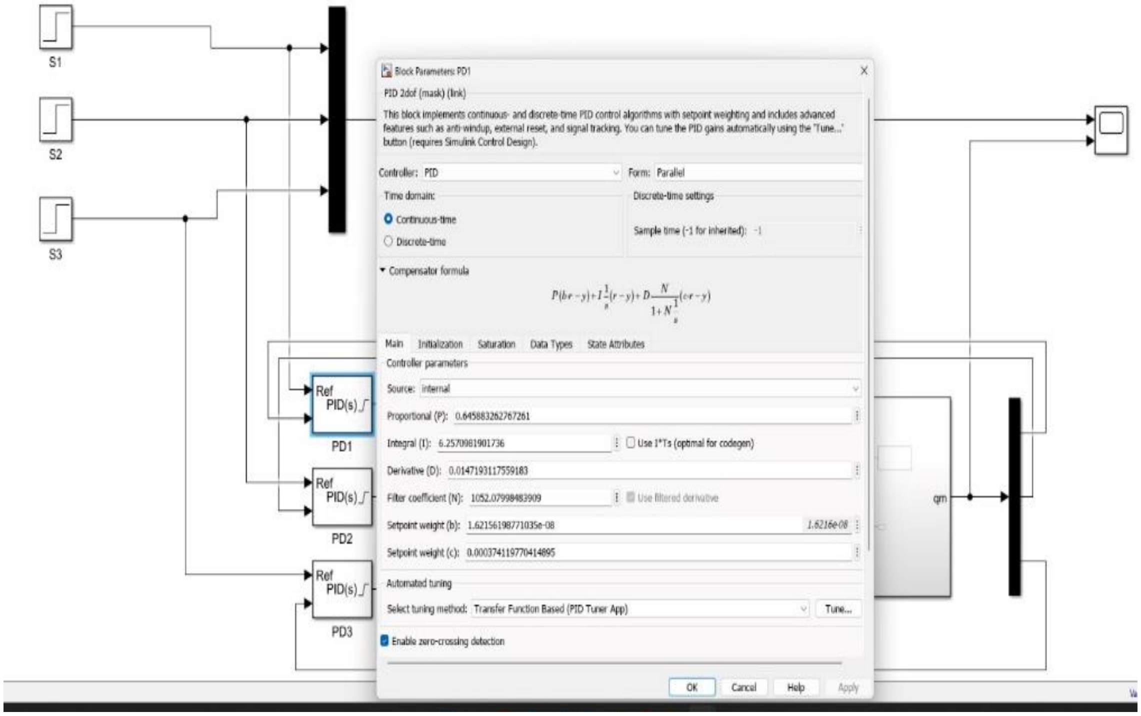


Implementation of PID in the Robot Model with a step input:

For tuning the Controller using sITuner, Step inputs have been given to the PID controller which had been attached to the Robot Model.



The tuning results came as given below:



Block Parameters: PD1

PID 2dof (mask) (link)

This block implements continuous- and discrete-time PID control algorithms with setpoint weighting and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID   Form: Parallel

Time domain:
● Continuous-time
○ Discrete-time

Discrete-time settings
Sample time (-1 for inherited): -1

Compensator formula

$$P(br-y)+I\frac{1}{s}(r-y)+D\frac{N}{1+N\frac{1}{s}}(cr-y)$$

Main | Initialization | Saturation | Data Types | State Attributes

Controller parameters

Source: internal

Proportional (P): 0.645883262767261

Integral (I): 6.2570981901736   □ Use I*Ts (optimal for codegen)

Derivative (D): 0.0147193117559183

Filter coefficient (N): 1052.07998483909   ☑ Use filtered derivative

Setpoint weight (b): 1.62156198771035e-08   1.6216e-08

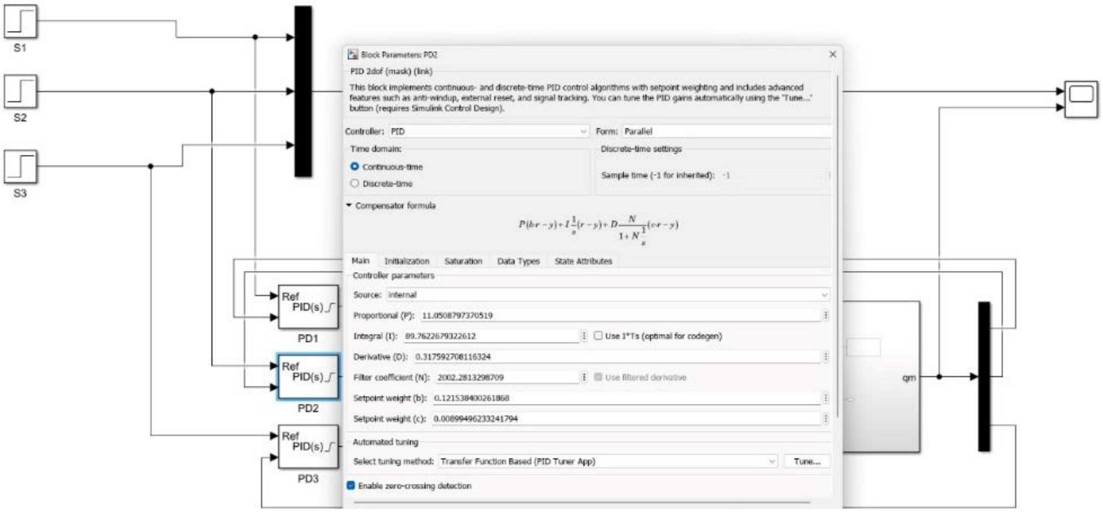
Setpoint weight (c): 0.000374119770414895

Automated tuning

Select tuning method: Transfer Function Based (PID Tuner App)   Tune...

☑ Enable zero-crossing detection

OK   Cancel   Help   Apply



Block Parameters: PD2

PID 2dof (mask) (link)

This block implements continuous- and discrete-time PID control algorithms with setpoint weighting and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID   Form: Parallel

Time domain:
● Continuous-time
○ Discrete-time

Discrete-time settings
Sample time (-1 for inherited): -1

Compensator formula

$$P(br-y)+I\frac{1}{s}(r-y)+D\frac{N}{1+N\frac{1}{s}}(cr-y)$$

Main | Initialization | Saturation | Data Types | State Attributes

Controller parameters

Source: internal

Proportional (P): 11.0508797370519

Integral (I): 89.7622679322612   □ Use I*Ts (optimal for codegen)

Derivative (D): 0.317592708116324

Filter coefficient (N): 2002.2813298709   ☑ Use filtered derivative

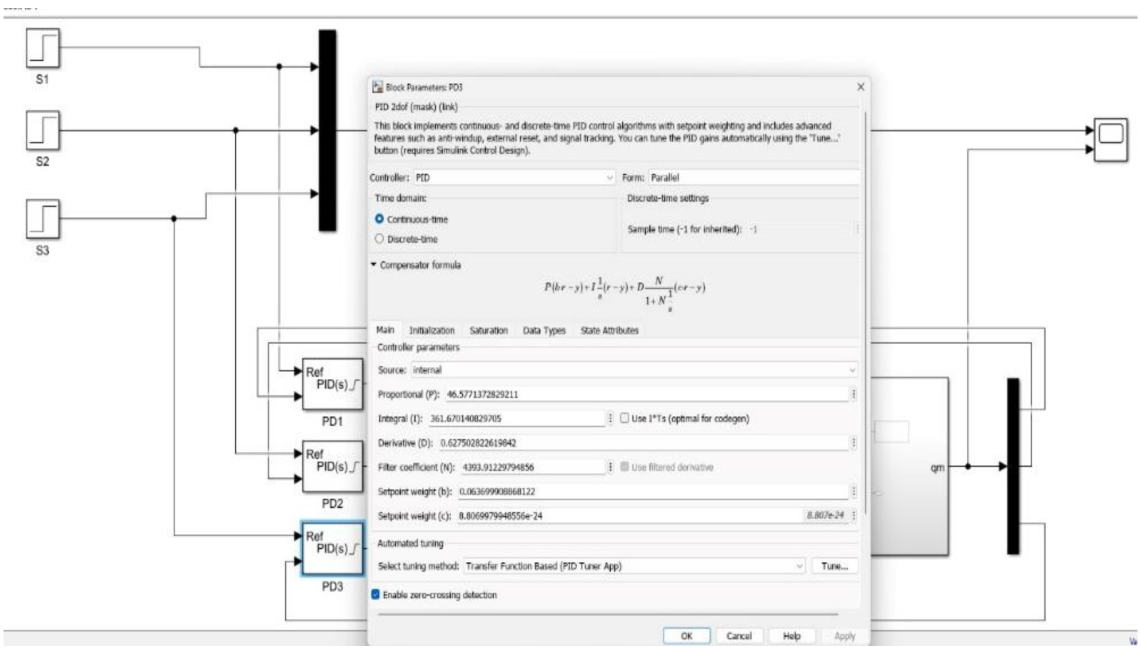
Setpoint weight (b): 0.121538400261868

Setpoint weight (c): 0.00899496233241794

Automated tuning

Select tuning method: Transfer Function Based (PID Tuner App)   Tune...

☑ Enable zero-crossing detection



Block Parameters: PD3

PID 2dof (mask) (link)

This block implements continuous- and discrete-time PID control algorithms with setpoint weighting and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID

Form: Parallel

Time domain:

● Continuous-time

○ Discrete-time

Discrete-time settings

Sample time (-1 for inherited): -1

Compensator formula

$$P(br-y)+I\frac{1}{s}(r-y)+D\frac{N}{1+N\frac{1}{s}}(cr-y)$$

Main | Initialization | Saturation | Data Types | State Attributes

Controller parameters

Source: internal

Proportional (P): 46.5771372829211

Integral (I): 361.670140829705 □ Use I*Ts (optimal for codegen)

Derivative (D): 0.627502822619842

Filter coefficient (N): 4393.91229794856 ☑ Use filtered derivative

Setpoint weight (b): 0.063699908868122

Setpoint weight (c): 8.8069979948556e-24 8.807e-24

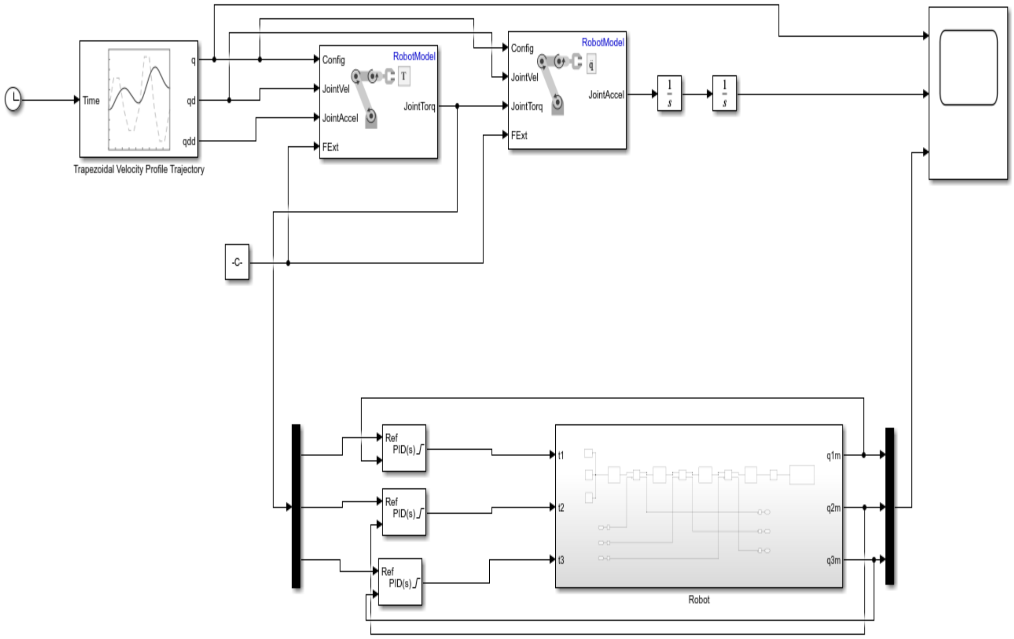
Automated tuning

Select tuning method: Transfer Function Based (PID Tuner App) Tune...

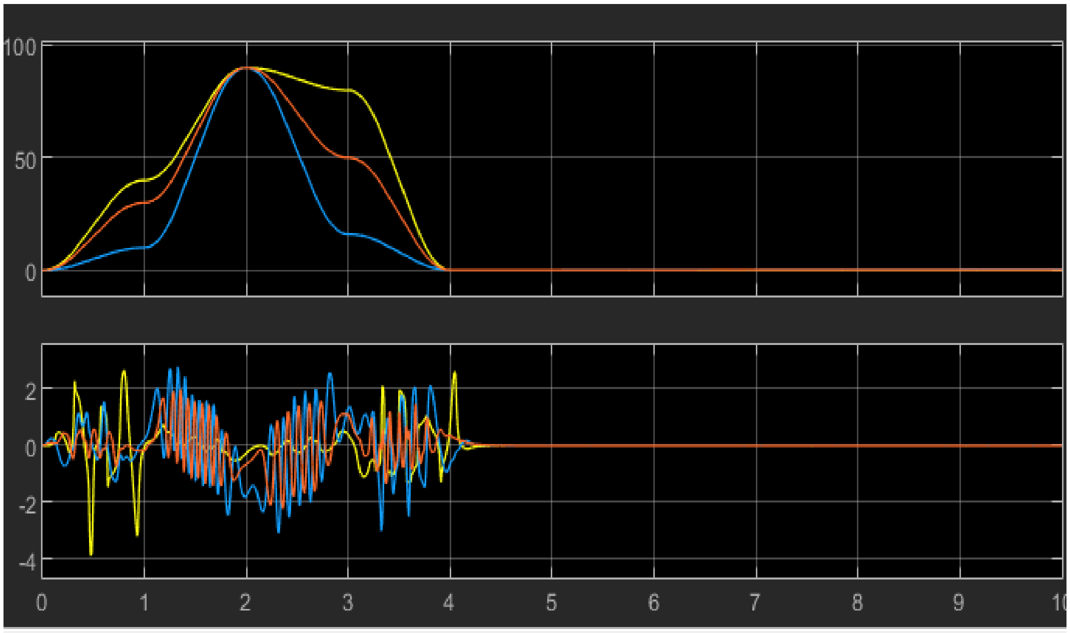
☑ Enable zero-crossing detection

OK Cancel Help Apply

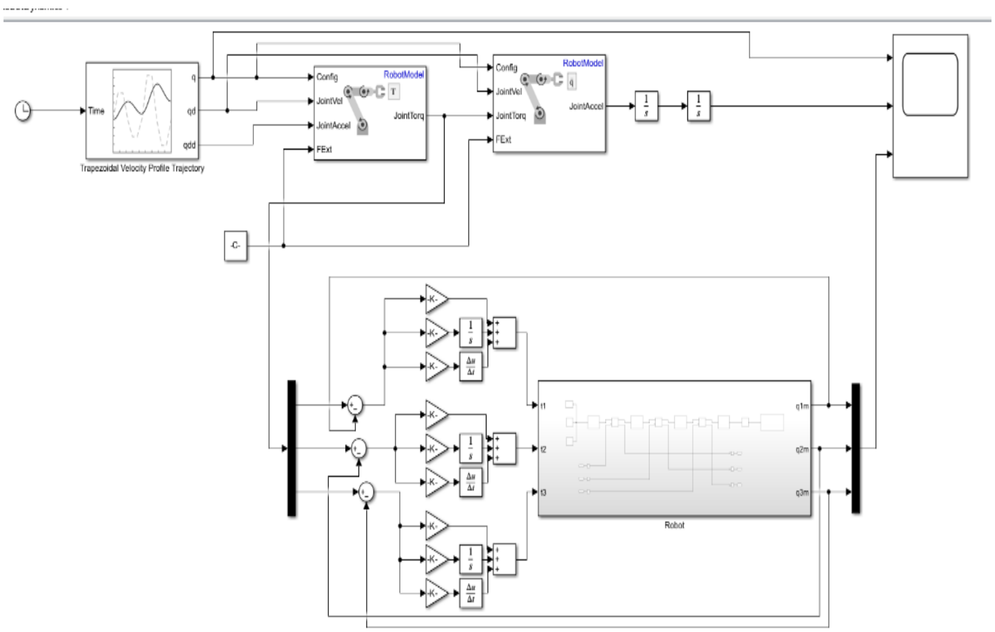
Implementing the Gains obtained from the above Tuning in the DOBOT Magician Model



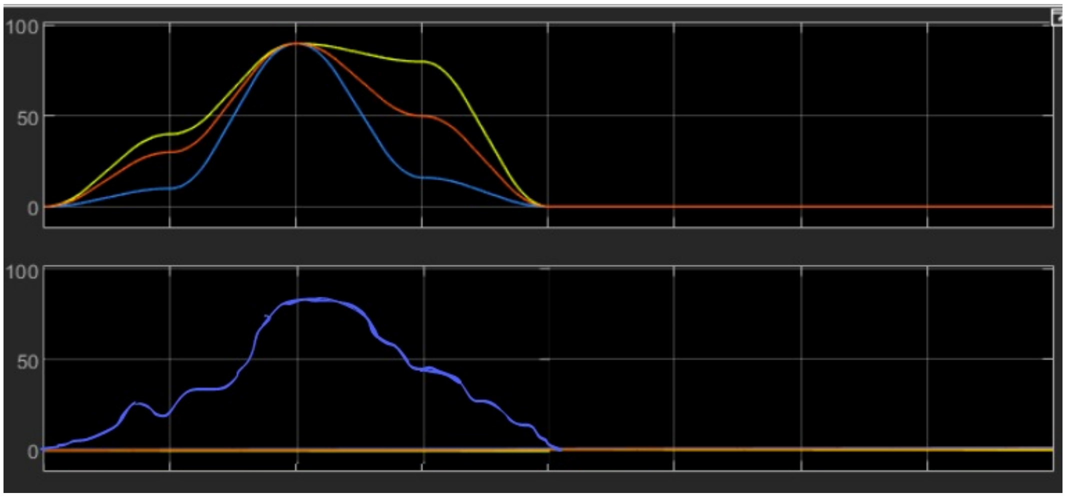
The result has been shown in the figure below.



As the output did not track the trajectory properly, it has been decided to build the PID controller from basic matlab blocks and tune it using Ziegler Nichols method.

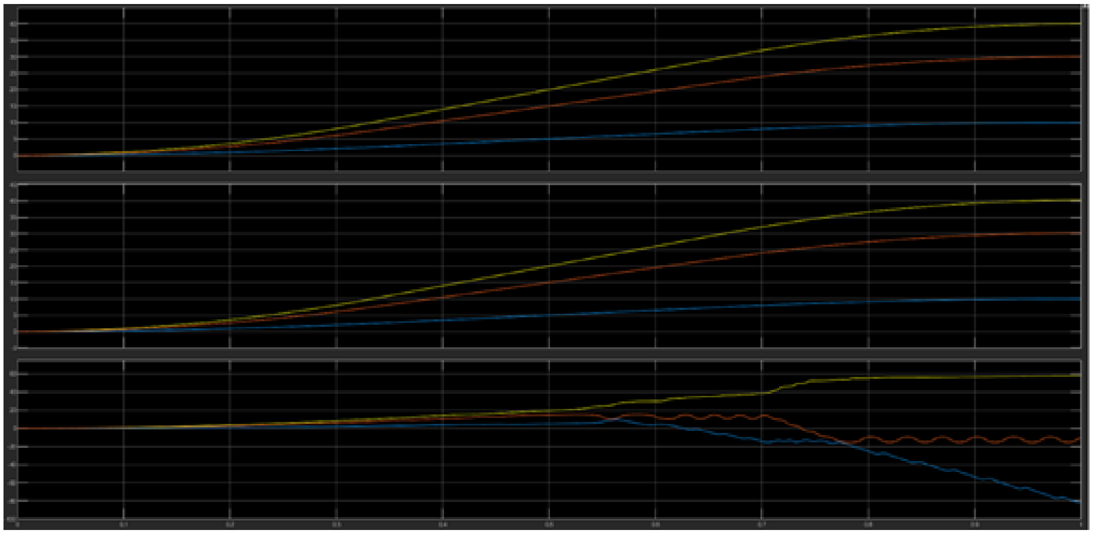


Theta 1 and Theta 2 have been made as 0, so that it will be easy to tune the theta 3. The result of the above implementation has been shown in the figure given below.


100
50
0
100
50
0

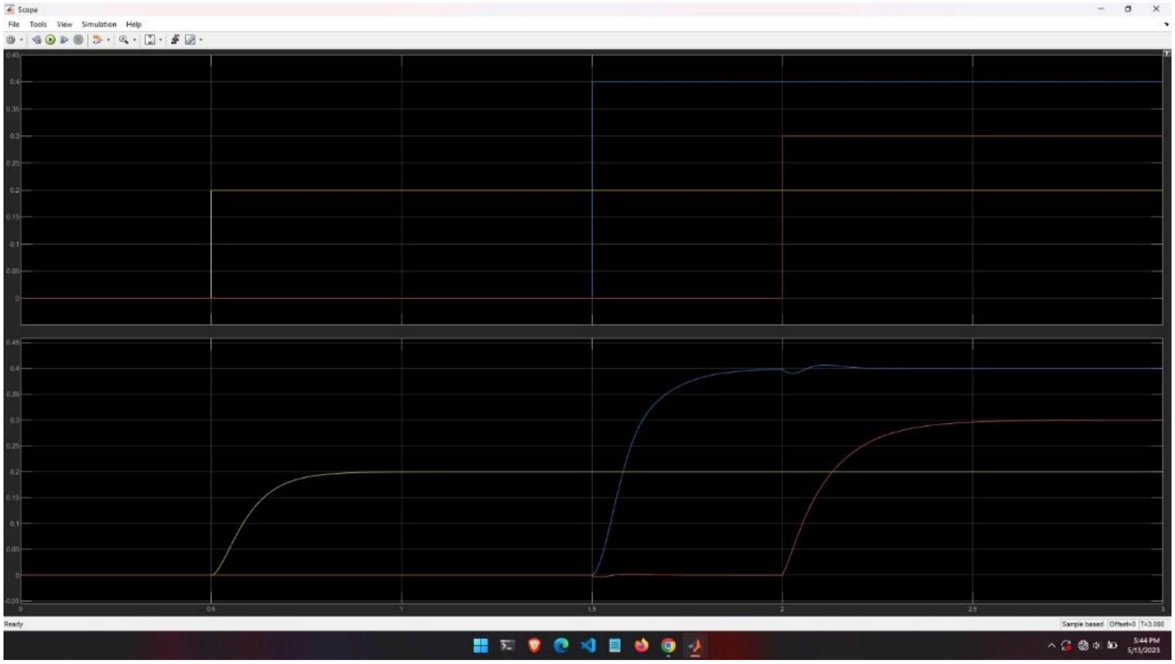
CHAPTER 5
RESULTS

The result of the Scope when we run RobotDynamics.slx without controller is:



The 1st Graph is the reference signal, the second graph is the signal that went through the system's kinematics part alone and the 3rd graph is the signal that went through both the kinematics and dynamics part of the system. The 3rd Graph is the Trajectory of SimScape Robot. It Deviates from the Desired Trajectory because each joint in SimScape has stiffness constant, damping constant. So even when we apply the required torque to each joint, there will be deviations in the Joint Angles.

The result of the Scope after implementing the PID controller in the robot system without giving the joint torque as the input:

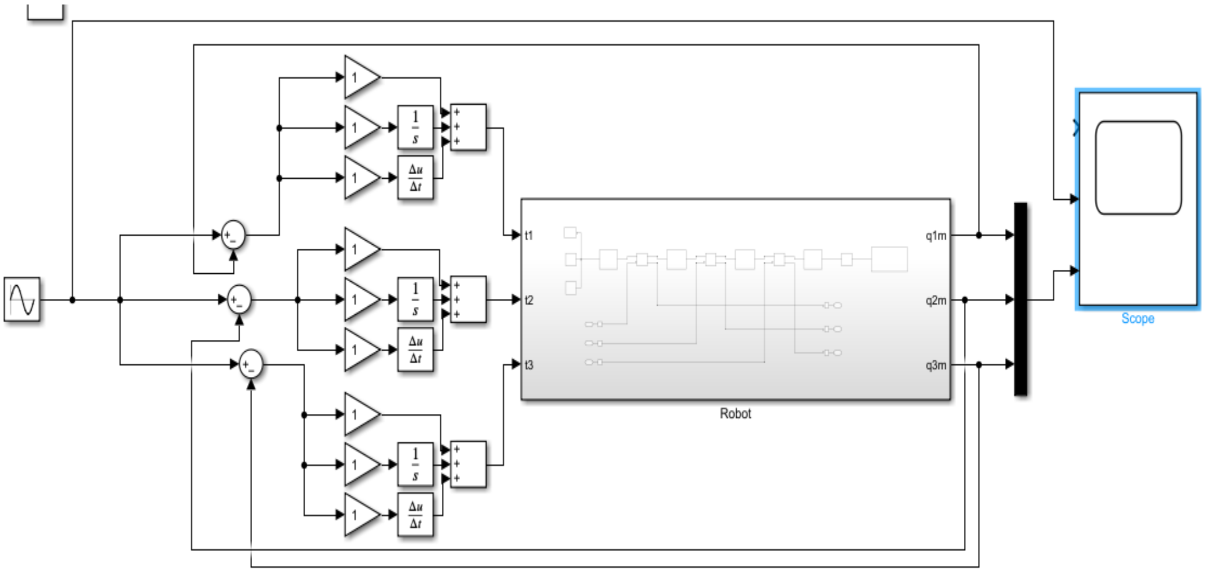


We implemented the PID controller and tuned it using MATLAB sITuner. We tested it using step input.

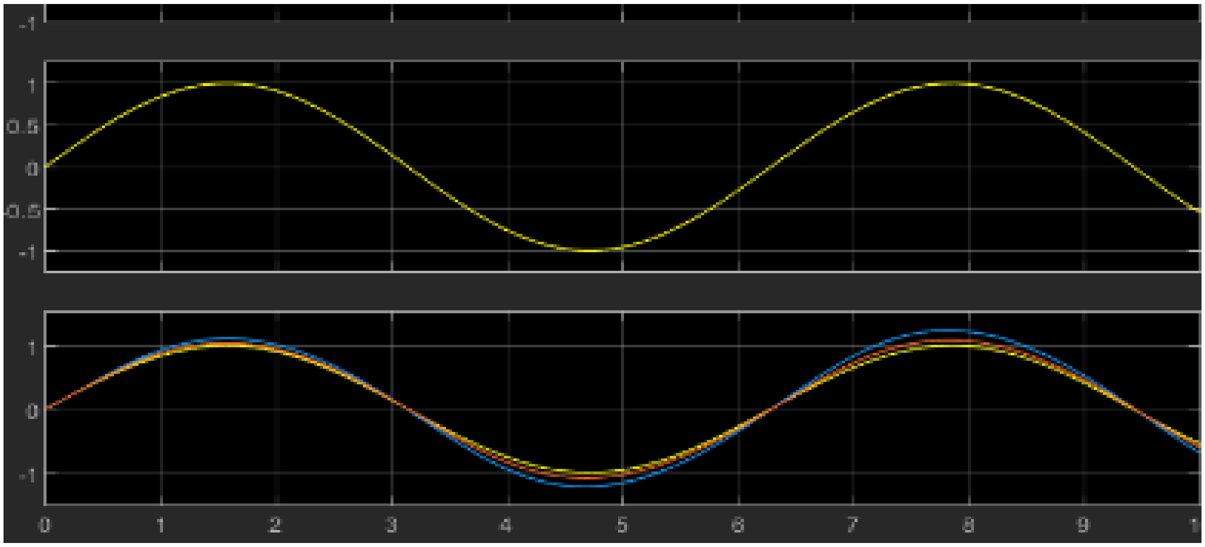
The first graph is the reference signal. The second graph is the tuned outputs.

Sine wave tracking:

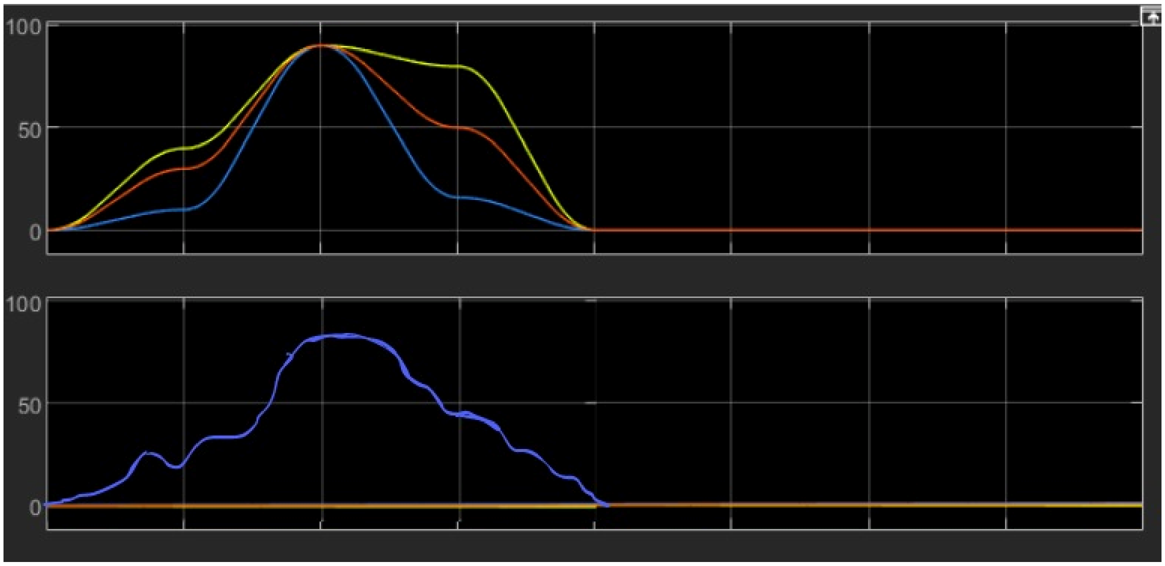
Simulink model:



Scope:



The result of the Scope after implementing the PID controller in the robot system with the joint torques as input to the controller:



Here, the first graph is the reference signal and the second graph is the output we got from the simulation after tuning theta 3 alone.

CHAPTER 6
CONCLUSIONS

In this work, a 3-DOF robotic arm, Dobot Magician, has been considere. The Dobot Magician had been analyzed completely for Mathematical modelling of the system. In this the forward kinematics of the Dobot is derived using modified Denavit-Hartenberg parameters and inverse kinematics is solved using algebraic approach. Also, the dynamic analysis of the dobot is performed. For the dynamics of the robot, the mass, link length, dimensions had been taken from the user manual of Dobot Magician.

A Simulink Model of the robot using *SimScape Toolbox* and *Robotics System Toolbox* has been created. The model is run for the objective of the robot trajectory tracking using Dynamic Equations and a comparison is made between the responce trajectory and the reference trajectory. A PID controller had been designed and implemented in the robotic system to obtain the desired trajectory tracking response.

CHAPTER 7
FUTURE WORKS

In the future, the design of advanced control methodologies such as adaptive control, sliding mode control, etc. can be performed for the obtained mathematical model of the robotic arm. The effect of model uncertainties and disturbances may also be considered. The designed control laws remain to be implemented on the experimental setup of Dobot Magician.

After implementing such control schemes, their responses can be compared to get the most efficient control algorithm for the 3-DOF Robotic arm. Applications of such techniques can be performed for industrial robot systems and other such devices used across various fields.

REFERENCES

[1] Md Rasedul Islam, Mohammad Arifur Rahman, Mohammad Rahman, “Cartesian Trajectory Based Control of Dobot Robot*”, International Conference on Industrial Engineering and Operations Management*, 2019.

[2] Mohammad Rezwan Sheikh. “Trajectory Tracking of a Four Degree our Degree of Freedom Robotic Manipulator”, *Journal of Physics*, 1345 (4), 2019.

[3] Sheng Dong, Zhaohui Yuan and Fuli Zhang, “A Simplified Method for Dynamic Equation of Robot in Generalized Coordinate System”, *International Journal of Physics*, 2019.

[4] Mohammad Mahdavian, Aghil Yousefi-Koma, Masoud Shariat-Panahi, Amirmasoud Ghasemi-Toudeshki, “Optimal Trajectory Generation for Energy Consumption Minimization and Moving Obstacle Avoidance of a 4DOF Robot Arm”, *3rd International Conference on Robotics and Mechatronics*, 2015.

[5] Thomas Lens, J¨urgen Kunz, and Oskar von Stryk, “Dynamic Modeling of the 4 DoF BioRob Series Elastic Robot Arm for Simulation and Control”, *2nd International Conference on Simulation, Modeling and Programming for Autonomous Robots*, 2010.

[6] G.Field,Y.Stepanenko.. “Iterative Dynamic Programming: An Approach to Minimum Energy Trajectory Planning for Robotic Manipulators”, *Intenational Conference on Robotics and Automation. Minneapolis*, Minnesota, April 1996.

[7] A.R. Hirakawa, A.Kawamura, “Trajectory Planning of Redundant Manipulators for Minimum Energy Consumption without Matrix Iniversion”, *International Conference on Robotics and Automation. Albuquerque*, New Mexico, April 1997.

[8] N.Baba, N.Kubota. “Collision Avoidance Planning of a Robot Manipulator by Using Genetic Algorithm - A Consideration for the Problem in Which Moving Obstacles and/or Several Robots Are Included in the Workspace”, *IEEE World Congress on Computational Intelligence*, 1994.

[9] M. Kawato, Y. Maeda, Y. Uno , and R. Suzuki, “Trajectory Formation of Arm Movement by Cascade Neural Network Model Based on Minimum Torque-Change Criterion”, *Biological Cybernetics*, 1990.

[10] A.S.Deo, I.D.Walker, “Minimum Effort Inverse Kinematics for Redundant Manipulators”, *IEEE Transactions on Robotics and Automation,* 1997.

[11] S.F.P. Saramago, V.Steffen Jr., “Dynamic Optimization for the Trajectory Planning of Robot Manipulators in the Presence of Obstacles” *Journal of the Brazilian Society of Mechanical Sciences*, 1999.

[12] Z. H. Zhu, R. V. Mayorga and A.K.C. Wong, “Dynamic robot manipulator trajectory planning for obstacle avoidance”, *Mechanics Research Communications*, 1999.

[13] Y.Zhang, S.Ma, “Minimum-Energy Redundancy Resolution of Robot Manipulators Unified by Quadratic Programming and its Online Solution”, *International Conference on Mechatronics and Automation*, 2007.

[14] G.S.Sharma, A. Kaur, “Optimization of Energy in Robotic arm using Genetic Algorithm”, *International Journal of Computer Science and Technology*, 2011.

[15] S.F.P. Saramagoa, V.Stefen Jr., “Optimal trajectory planning of robot manipulators in the presence of moving obstacles”, *Mechanism and Machine Theory.*, 2000.

[16] Serdar Kucuk, “Energy minimization for 3-RRR fully planar parallel manipulator using particle swarm optimization”, *Mechanism and Machine Theory*, 2012

[17] Dong Hui, Du Zhijiang. “Obstacle Avoidance Path Planning of Planar Redundant Manipulators using Workspace Density”, *International Journal of Advanced Robotic Systems* ,2015.

[18] F. Marquet, “Contributional’ ` etude de la redondance en robotique par- ´ allele” *Ph.D. Dissertation, Univ. Montpellier,* Montpellier, France, 2002.

[19] V. Nabat, F. Pierrot, M. Rodriguez, J. Azcoita, R. Bueno, O. Company, and K. Florentino, “High-speed parallel robot with four degrees of freedom”, *European Patent*, EP1870214, Dec. 26, 2007.

[20] A. Vivas, P. Poignet, F. Marquet, F. Pierrot, and M. Gautier, “Experimental dynamic identification of a fully parallel robot”, *IEEE ICRA*, Taipei, Taiwan, 3278-3283, 2003.

[21] J. Angeles, “The degree of freedom of parallel robots: A group-theoretic approach”, *IEEE ICRA*, 1005-1012, 2005.