

**DESIGN AND DEVELOPMENT OF 5-DoF
INDUSTRIAL ROBOTIC ARM**

A

REPORT

**SUBMITTED IN PARTIAL FULFILLMENT OF THE
MINOR PROJECT – II (4ME33) COURSE**

For the award of

Bachelor of Technology

(Mechanical Engineering)

Submitted by

Kirtan Gajjar	18ME070
Dipen Galathiya	18ME072
Parth Haria	18ME082
Prakhar Raval	18ME431

Under the guidance of

Dr. Haresh Patolia




Birla Vishvakarma Mahavidyalaya
Engineering Collage (An Autonomous Institution)
Vallabh Vidyanagar - 388 120
Affiliated to Gujarat Technological University
May 2022

CERTIFICATE

Date: 14-05-2022

This is to certify that the Minor Project II (4ME33) entitled “DESIGN AND DEVELOPMENT OF 5-DoF INDUSTRIAL ROBOTIC ARM” has been carried out by Kirtan Gajjar, Dipen Galathiya, Parth Haria and Prakhar Raval in the 8th semester under my guidance in partial fulfillment of the degree of Bachelor of Technology in Mechanical Engineering, Birla Vishvakarma Mahavidyalaya, Vallabh Vidyanagar, during the academic year 2021-22.



Dr. Haresh Patolia
Guide

Dr. P. M. George
Head of the Department

ORIGINALITY REPORT CERTIFICATE

It is certified that Minor Project II (4ME33) - Report entitled “DESIGN AND DEVELOPMENT OF 5-DoF INDUSTRIAL ROBOTIC ARM” has been carried out by Kirtan Gajjar, Dipen Galathiya, Parth Haria and Prakhar Raval has been examined by us. We undertake the following:

- [1] Project has significant new work / knowledge as compared already published or are under consideration to be published elsewhere. No sentence, equation, diagram, table, paragraph or section has been copied verbatim from previous work unless it is placed under quotation marks and duly referenced.
- [2] The work presented is original and own work of the author (i.e., there is no plagiarism). No ideas, processes, results or words of others have been presented as Author own work.
- [3] There is no fabrication of data or results which have been compiled / analyzed.
- [4] There is no falsification by manipulating research materials, equipment or processes, or changing or omitting data or results such that the research is not accurately represented in the research record.
- [5] The report has been checked using DUPLICHECKER (copy of originality report attached) and found within limits, (i.e., permitted similarity index $\leq 20\%$)

Name and signature of Students:

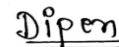
Kirtan Gajjar

18ME070



Dipen Galathiya

18ME072



Parth Haria

18ME082

P.M. Haria.

Prakhar Raval

18ME431




ACKNOWLEDGEMENT

We are thankful to Dr. Haresh Patolia, Professor, Mechanical Engineering Department, B. V. Mahavidyalaya Engineering college, our guide, for his guidance during our whole project by sharing their extensive knowledge in the field of robotics. We appreciate his motivation, encouragement, and support throughout this semester. We also appreciate his broad range of expertise and attention to every detail of our project; without him this work would have been impossible. We would like to take this opportunity to thank our Parents for their unconditional love, moral support, and encouragement for the timely completion of this project. We would like to thank Assistant Professor Dhruvesh Gajjar for helping us during 3D printing in Rapid Prototyping and Manufacturing laboratory at the college. We would like to thank our friends for constant support and help throughout our project work. So many people have contributed to our project, our education, and it is with great pleasure to take the opportunity to thank them. We apologize if we have forgotten anyone.

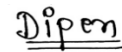
Kirtan Gajjar

18ME070



Dipen Galathiya

18ME072



Parth Haria

18ME082

P.M. Haria.

Prakhar Raval

18ME431



ABSTRACT

The importance of industrial robotic arm is increasing in the industry due to its characteristic like execution repeatability, position accuracy, adaptability to previously unknown situations and programmability for specific application. Industrial robotic arm kinematics is a vast field of study focused to know motion variables required to perform any desired operation by the robotic arm for required application. Numerical methods such as Newton-Raphson, Jacobian, Damped Least Square (DLS), etc. are commonly used for the inverse kinematic analysis. Trajectory planning is calculating the time sequence of position, velocity and acceleration for each joint or end effector of the robotic arm for performing a specific application. Moreover, as the industrial robotic arm work in hazardous environment, it is required to make an arrangement to control the robotic arm farther from the hazardous environment.

The kinematic analysis of 5-DoF industrial robotic is carried out using MATLAB[®] software. Adding to this, designing of the components of robotic arm is done using CREO[®] Parametric software. The components are then 3D printed at the college premises. A GUI is developed for controlling the robot as per the Forward and Inverse Kinematics which uses the App Designer and Hardware support for Arduino in MATLAB[®]. Furthermore, the GUI for trajectory planning in MATLAB[®] is made which takes input for both joint space and Cartesian space trajectory techniques. The analysis for a cubic trajectory is carried out using the MATLAB[®] for different values of joint angles and the results are then compared with same analysis done using CREO[®]. The trajectory is then viewed using simulation and can be directly used to make the robot follow the same path. An android application is made using MIT App Inventor 2 which is able to control the robot as per forward and inverse kinematics and is able to save position and run the robot continuously on the path formed by saved positions.

Table of Contents

Contents

ORIGINALITY REPORT CERTIFICATE.....	ii
ACKNOWLEDGEMENT	iii
ABSTRACT.....	iv
LIST OF FIGURES	vii
1. INTRODUCTION	1
1.1 Introduction to Robot	1
1.2 Anatomy of Robotic Arm.....	1
1.2.1 Links	2
1.2.2 End-Effector	2
1.2.3 Joints.....	2
1.3 Robotic Notations.....	3
1.4 Objectives.....	3
2. LITERATURE REVIEW	5
2.1 Need of Project.....	5
2.2 Literature Survey.....	5
3. DEVELOPMENT OF ROBOTIC ARM	8
3.1 Designing of Robotic Arm	8
3.2 Final Assembly (3D Modelling and Rendering in CREO®)	9
3.3 Manufacturing of Robotic Arm.....	10
3.4 Experimental Setup	12
3.5 Circuit Diagram for Control of Robotic Arm.....	13
3.5.1 Connections	13
4. KINEMATICS	14
4.1 Process of Kinematic Analysis.....	14
4.2 Frame Assignment.....	14
4.3 Kinematic Analysis using D-H Convention.....	15
4.4 Forward Kinematics	17
4.5 Flowchart for Forward Kinematics	17
4.6 Inverse Kinematics	18
4.6.1 Concept of Iterative Method.....	19
4.7 Jacobian Matrix	19

4.8 GUI for Control of Robotic Arm.....	20
5. TRAJECTORY PLANNING.....	21
5.1 Steps in Trajectory Planning	22
5.2 Joint Space Techniques	23
5.2.1 Cubic Polynomial Trajectories	23
5.2.2 Parabolic Trajectory	24
5.2.3 Cycloidal Trajectory	26
5.2.4 Circular Path	26
5.3 Position Planning	26
5.4 Orientation Planning	28
5.5 Results based on MATLAB®	28
5.6 Results of Joint Space Trajectory Technique	29
6. BLUETOOTH CONTROL.....	38
6.1 Working of Robotic Arm by Controlling with Bluetooth	38
6.2 Application for Bluetooth Control	38
6.2.1 MIT App Inventor 2	38
6.2.2 User Interface	39
6.2.4 Working.....	42
6.3 Flowchart for Working of Application.....	43
6.4 Application of Bluetooth after Inverse Kinematics.....	44
6.5 Performance of Pick-n-Place using Bluetooth	45
7. RESULT DISCUSSION, CONCLUSION AND FUTURE WORK.....	46
7.1 Results and Discussion.....	46
7.2 Conclusion.....	50
7.3 Future Work	50
8. REFERENCES	51
Plagiarism Report.....	53

LIST OF FIGURES

Figure 1.1: Robot Anatomy	2
Figure 1.2: Types of Joints.....	3
Figure 3.1: Components of Robotic Arm	9
(a) Base	8
(b) Waist.....	8
(c) Arm 01	8
(d) Arm 02.....	8
(e) Arm 03.....	9
(f) Gripper Base	9
(g) Gear 1	9
(h) Gear 2	9
(i) Grip Link.....	9
(j) Grip Finger	9
Figure 3.2: Rendered Assembly in CREO®	10
Figure 3.3: 3D printing of Arm-2	11
Figure 3.4: 3D Printing of Base	11
Figure 3.5: Assembled Model.....	12
Figure 3.6: Experimental Setup	12
Figure 3.7: Circuit Diagram.....	13
Figure 4.1: Process of Kinematic Analysis.....	14
Figure 4.2: Frame Assignment.....	15
Figure 4.3: D-H Parameter.....	15
Figure 4.4: Flowchart of Forward Kinematics.....	18
Figure 4.5: GUI developed using MATLAB®	20
Figure 5.1: Trajectory Planning Problem.....	21
Figure 5.2: Interface for Joint Space.....	29
Figure 5.3: Interface for Cartesian Space	29
Figure 5.4: Parameters for Joints	30
(a) Position Versus Time	30
(b) Velocity Versus Time.....	30
(c) Acceleration Versus Time	30
Figure 5.5: Parameters for Links in X-axis.....	32
(a) Position Versus Time	31
(b) Velocity Versus Time.....	31
(c) Acceleration Versus Time	32
(d) Angular Velocity Versus Time	32
(e) Angular Acceleration Versus Time	32
Figure 5.6: Parameters for Links in Y-axis.....	34
(a) Position Versus Time	33
(b) Velocity Versus Time.....	33
(c) Acceleration Versus Time	34
(d) Angular Velocity Versus Time	34
(e): Angular Acceleration Versus Time	34
Figure 5.7: Parameters for Links in Z-axis	36

(a) Position Versus Time	35
(b) Velocity Versus Time.....	35
(c) Acceleration Versus Time	36
(d) Angular Velocity Versus Time	36
(e) Angular Acceleration Versus Time	36
Figure 5.8: Simulation	37
Figure 6.1: User Interface of the App	39
Figure 6.2: Labels	40
Figure 6.3: Text Box	41
Figure 6.4: Functioning Buttons	41
Figure 6.5: Connectivity Interface	42
Figure 6.6: Flowchart of Android Application	43
Figure 6.7: Application Interface for Inverse	44
Figure 6.8: Pick-n-Place Operation.....	45
Figure 7.1: Position Versus Time	47
(a) Error in X- components	47
(b) Error in Y- components.....	47
(c) Error in Z- components	47
Figure 7.2: Velocity Versus Time.....	48
(a) Error in X- components	48
(b) Error in Y- components.....	48
(c) Error in Z- components	48
Figure 7.3: Acceleration Versus Time	49
(a) Error in X- components	49
(b) Error in Y- components.....	49
(c) Error in Z- components	49

LIST OF TABLES

Table 3.1: Connections of motors	13
Table 3.2: Connections of Bluetooth module	13
Table 4.1: D-H Parameters	16

1. INTRODUCTION

In this chapter, an idea about industrial robotic arm, anatomy of industrial robotic arm, basic components of industrial robotic arm and brief idea of notation used in context of the industrial robotic arm are given.

1.1 Introduction to Robot

Robotics is an interdisciplinary branch of engineering that comprises the formation, design, building, and action of robots. It deals with the computer systems for their control, sensory response, and information processing.

The definition of robot given by the International Standards Organization (ISO) is “An industrial robot is an automatic, servo-controlled, freely programmable, multipurpose manipulator, with several areas, for the handling of work pieces, tools, or special devices. Variably programmed operations make the execution of a multiplicity of tasks possible”.

A robotic arm is an artificial arm to achieve desired tasks. Now-a-days, there is a more and more purpose to develop artificial arms for various non-human situations where human communication is impossible.

Robotic arm has certain applications such as, military purpose, group task, in hospital to monitor different action; robotic arm can easily perform stitching and various surgical operations, can play role of a waiter in restaurant, can play sports with human, can be used for agricultural purpose, Swarm robotics.

1.2 Anatomy of Robotic Arm

The mechanical structure of an industrial robotic arm consists of links (rigid bodies) connected by means of joints (articulations), is segmented into an arm that ensures mobility and reachability, a wrist that confers orientation and end effector that performs the required task. The study of physical construction of the manipulator structure is called robot anatomy. The arrangement of base, arm, wrist, and end-effector is shown in Figure 1.1 which represents the anatomy of robotic arm.

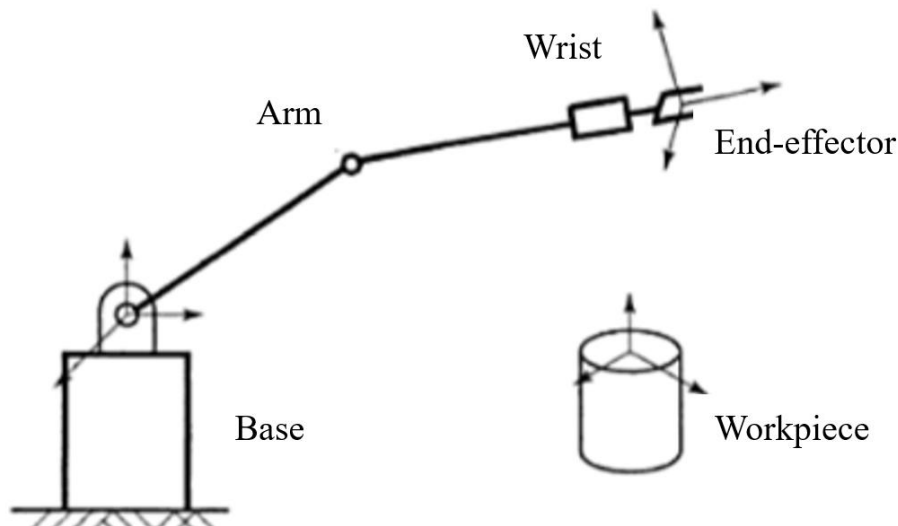


Figure 1.1: Robot Anatomy

1.2.1 Links

Mechanical structure of industrial robotic arm is formed from members such as rigid links or bars. Those links are connected with one another. It is considered as binary link when two links are connected with each other. Both links make a joint which is called binary joint. If links perform relative motion, then they are said to be revolute or rotary joint.

1.2.2 End-Effector

Various end effectors can be attached to the end of wrist considering the application of robotic arm. End effectors such as grippers and tools are mostly used. Grippers can be used as holding component and work piece. It can be used in other ways like magnetic vacuum bellows. The size of gripper is dependent on the object to be hold and task to be done. Tools are used in place of grippers where operations like welding, machining, boring are to be performed.

1.2.3 Joints

Prismatic Joint: The two links are joint together so that they can slide with respect to each other which is shown in Figures 1.2 (a) and 1.2 (b). Examples are Screw-Nut and Rack-Pinion.

Revolute Joint: The two links are joint together by a pin about the axis of which these links can rotate with respect to each other which is shown in Figures 1.2 (c), 1.2(d) and 1.2 (e).

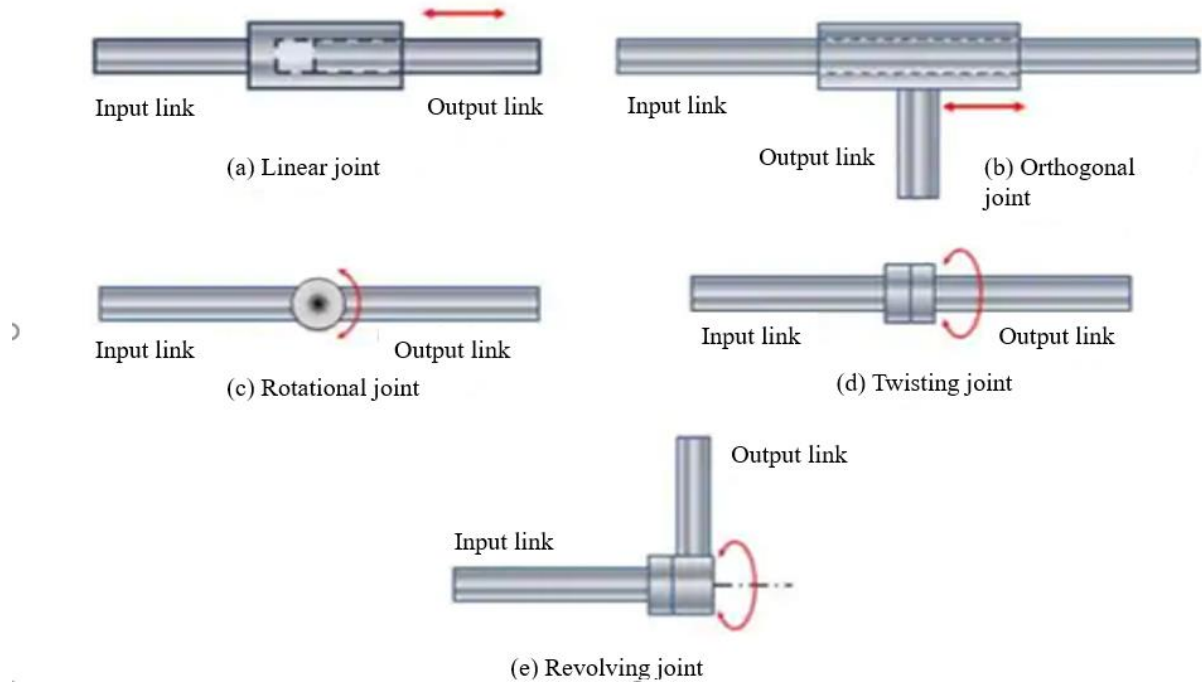


Figure 1.2: Types of Joints

1.3 Robotic Notations

The association of a vector to a coordinate frame is described by a leading superscript. For example, 0P is a position vector P in frame $\{0\}$. Transformation Matrices from one coordinate frame to another, have a leading superscript and a trailing subscript. For example, 0T_1 denotes the coordinate transformation matrix, which transforms coordinates from frame $\{1\}$ to frame $\{0\}$. Many trigonometric functions are necessary in building mathematical models.

The sines and cosines of an angle θ_i can take any of the forms such as:

$$\cos \theta_i = C_i$$

$$\sin \theta_i = S_i$$

$$\cos(\theta_i + \theta_j) = C_{ij}$$

$$\sin(\theta_i + \theta_j) = S_{ij}$$

1.4 Objectives

The main purpose of this project work is to make a prototype of 5-DoF industrial robotic arm and performing kinematic analysis on it. Also, in some of the industrial applications, there is a requirement of controlling the robotic arm wirelessly, so accordingly a Bluetooth module is used.

Following objectives are sorted from the above discussion:

- Forward and Inverse kinematics of the 5-DoF industrial robotic arm.
- 3D printing the parts of robot and assembling them as per the design.
- Controlling the robotic arm as per forward and inverse kinematics.
- Trajectory planning for the robotic arm using different trajectory techniques.
- Operating the robot with the help of android application and Bluetooth module.

2. LITERATURE REVIEW

In this chapter literature review related to kinematic analysis, trajectory planning for pick-n-place application and operating robotic arm via OPENCV is presented to fulfil the required objective assigned to it.

2.1 Need of Project

Kinematics is the motion geometry of robotic arm from reference position to the desired position with no regards to force or other factors that influence motion of robotic arm. To get desired motion of robotic arm for the specific task like pick-n-place, trajectory planning is required. Industrial robotic arm is widely used in the applications that are mentioned previously, so for the tasks where human arm cannot reach or conditions are not favorable for humans, the robotic arm controlled by Bluetooth, can be placed and desired task can be done.

2.2 Literature Survey

A. Oluwajobi and A. Oridate [1] designed and fabricated a 5-DoF robotic arm which can be used for educational and demonstrative purposes. A simulation of the robotic arm was achieved by using the MATLAB[®] Robotics Toolbox, to visualize the joint movements. A suitable servo controller was selected for the implementation and a control software for the robotic arm was developed using Microsoft's C# programming language. The software allows the robotic arm gripper to be positioned in space, by specifying the coordinates of its center position. The arm can be reprogrammed in order to adapt in variety of tasks like manufacturing, mining and packaging. V. Deshpande and P. M. George [13] developed a mathematical model for forward and inverse kinematics of a 5-DoF Robotic Arm for simple pick and place application. A complete analytical solution to the forward and inverse kinematic problem has been derived and the result of the forward kinematics can be crossed checked by the analytical method of inverse kinematic model. Y. Jadeja and B. Pandya [16] designed and controlled the 5-DoF robotic arm manipulator's angle by using Cortex ARM M3 LPC1768 Microcontroller including ultrasonic sensor and a digital controller using computer system. By using rotary-encoder the feedback of the angle can be measured. Main purpose of this paper is to introduce the level of intelligence that can be implemented to industries in order to reduce the human errors as well as enhance the quality and rapid production in manufacturing and processing. The arm was made successfully and helps to control the robot's movement easily. Based on

observation, it became clear that it is perfect and easily manageable as well as easy to operate. K. S. Nair *et al.* [7] described the mathematical formulation of complete kinematics of 5-DoF industrial robot (VRT-502) as well as theoretical analysis of the forward kinematics was done in MATLAB® to determine the end effector's position and orientation.

S. Manjaree *et al.* [10] provided a single platform for analytical analysis, simulation and experimental methods for a 5-DoF robotic manipulator with wrist in movement. A hybrid neuro-fuzzy intelligent technique with two different membership functions has been studied and their performances are comparatively evaluated with analytical solutions. The experimental validation had been performed for a 'circle' trajectory. S. Ghuffar *et al.* [9] presented a software architecture required to build a user-friendly interface for a robot manipulator. A prototype 5-DoF robotic arm was built and the end effector of this robotic arm is a two-fingered gripper. A user interface has been developed that provides powerful tools for efficient control of the robot manipulator.

J. Iqbal *et al.* [6] developed the kinematic models a 6-DoF robotic arm and analyzed its workspace. The proposed model makes it possible to control the manipulator to achieve any reachable position and orientation in an unstructured environment. The forward kinematic model has been validated using Robotics Toolbox for MATLAB® while the inverse kinematic model has been implemented on a real robotic arm. T. P. Singh *et al.* [12] proposed mathematical approach for solving the forward and inverse kinematic for 5-DoF and 6-DoF (PUMA 560) robotic manipulators. A movement flow planning is designed to evaluate all the DH parameter to calculate the desire position and orientation of the end effector. The experimental result is obtained by using RoboAnalyzer® and MATLAB® software and compare the result with the result calculate by using the D-H transformation.

A. Seemal and Philip Webb [2] used modified D-H convention to perform the forward kinematics for the manipulator kinematic. Inverse kinematics was performed using kinematics decoupling. The Comau NM45 Manipulator has been chosen for the kinematic model study. The manipulator contains six revolution joints. D. Constantin *et al.* [5] used the forward kinematics by two different techniques: analytically using a five-step algorithm derived from Denavit and Hartenberg notation and numerically using Robotic Toolbox. S. Panich *et al.* [11] described direct kinematic analysis was conducted to determine the parameter of industrial robot by using Denavit-Hartenberg (D-H) method and the calculated parameters of industrial

robot were implemented by direct kinematics and compared with the measured parameter by rotary encoder.

Y. D. Patel and P. M. George [15] used the mathematical formulation of complete kinematics and dynamics of two degrees of freedom industrial robot and the simulation of kinematics is performed using Pro/Engineer software and results are compared with analytical results.

R. R. Serrezuela *et al.* [8] used various methods such as the method of Pieper, which states that the inverse kinematic problem can be solved only for the position of the point of intersection of the axes of the robot wrist, i.e. for the first 3 joints using the algebraic method or geometric solution.

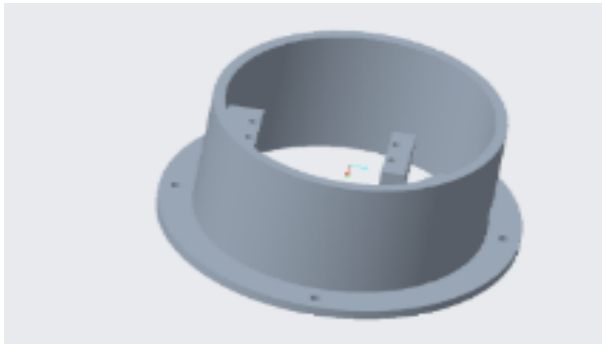
A. Srikanth *et al.* [3] introduced Jacobian matrix for simulation, the basic kinematic equations for use of the manipulator are derived and the influences of the physical constraints on the range of motion in the practical design and used RoboAnalyzer[®] software for determining the simulation results.

3. DEVELOPMENT OF ROBOTIC ARM

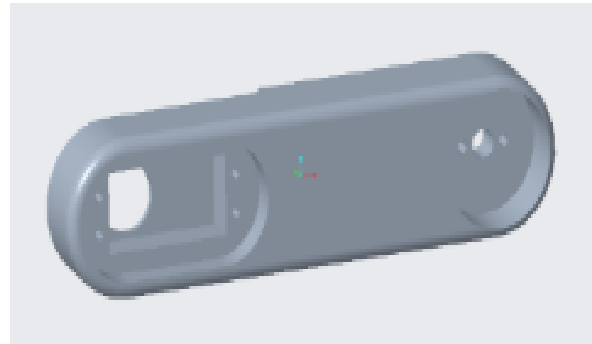
The present chapter discusses upon the designing of the components of robotic arm and creating its assembly in the CREO® Parametric software. After designing, the manufacturing of components is done with 3D printing. The developed components are then assembled and experimentation is carried out.

3.1 Designing of Robotic Arm

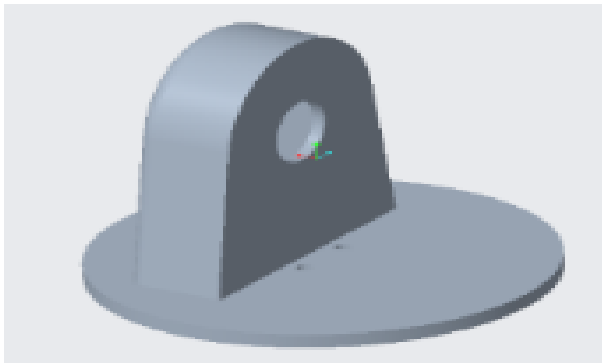
The industrial robotic arm developed here has 5-DoF and so it has 5 revolute joints. A gripper is attached at the end for pick and place application. The components of robotic arm like base, waist, arm 1, arm 2, arm 3, gripper base, gears, grip fingers and grip links have been designed using CREO® Parametric software, which is shown in Figures 3.1(a) to 3.1(j) respectively.



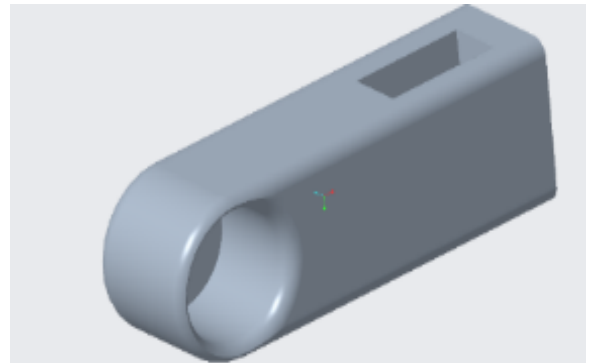
(a) Base



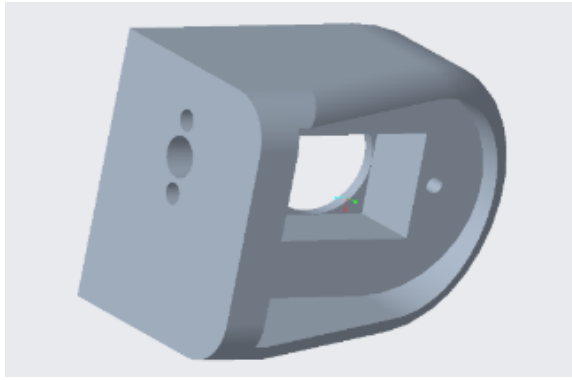
(c) Arm 01



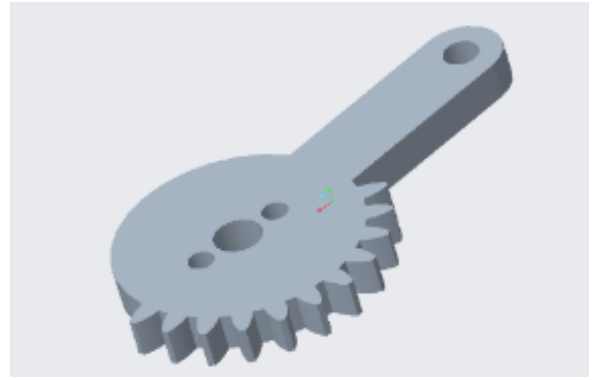
(b) Waist



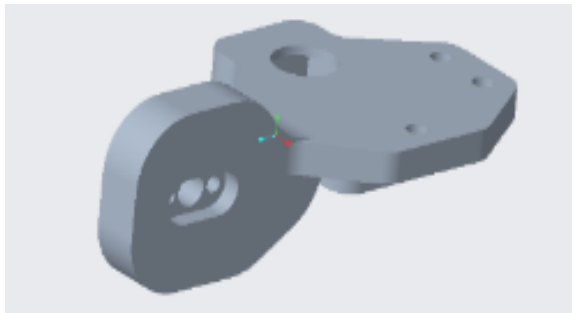
(d) Arm 02



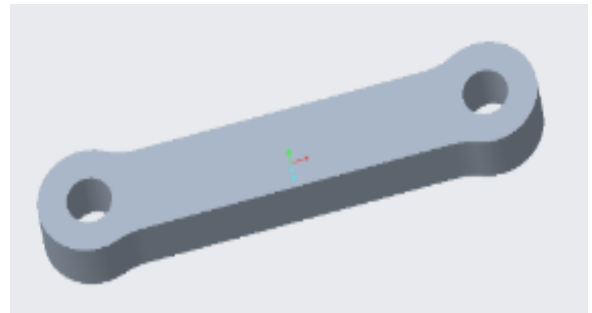
(e) Arm 03



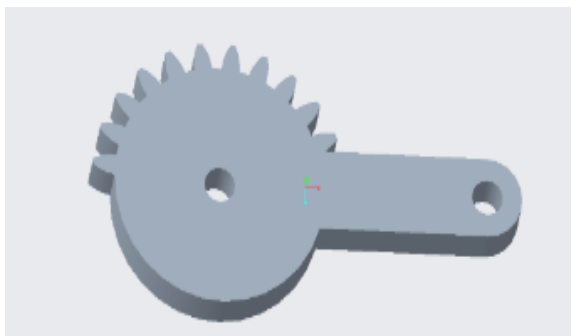
(h) Gear 2



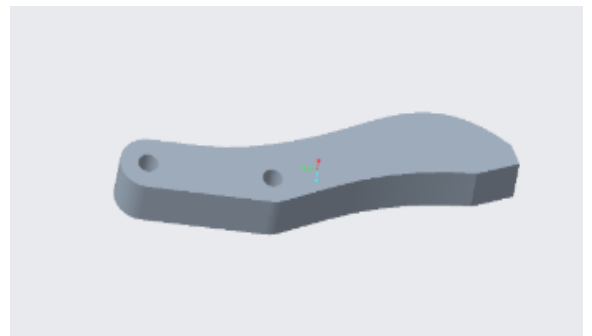
(f) Gripper Base



(i) Grip Link



(g) Gear 1



(j) Grip Finger

Figure 3.1: Components of Robotic Arm

3.2 Final Assembly (3D Modelling and Rendering in CREO®)

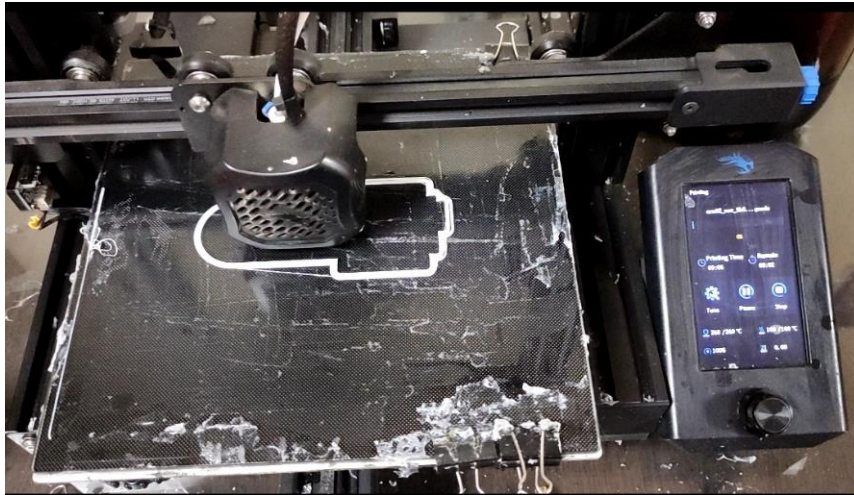
The assembly of all the designed components is created using CREO® and rendered using rendering tool in the software. The rendered model is shown in the Figure 3.2. The colour of the components while rendering is done keeping in mind, the colour of components being actually manufactured.



Figure 3.2: Rendered Assembly in CREO®

3.3 Manufacturing of Robotic Arm

The components of the robotic arm are 3D printed at Rapid Prototyping and Manufacturing laboratory at B. V. Mahavidyalaya, Engineering college. The components designed in CREO® are converted into STL files and a g-code is generated using slicing software which is then provided to the 3D printer and printing can be started. The base of 3D printer should be cleaned each time a new component is to be manufactured. The 3D printing of arm 3 is shown in Figures 3.3(a) and 3.3(b) and that of base and assembled model is shown in Figure 3.4 and 3.5 respectively.



(a) Initial stage



(b) Intermediate stage

Figure 3.3: 3D printing of Arm-2



Figure 3.4: 3D Printing of Base

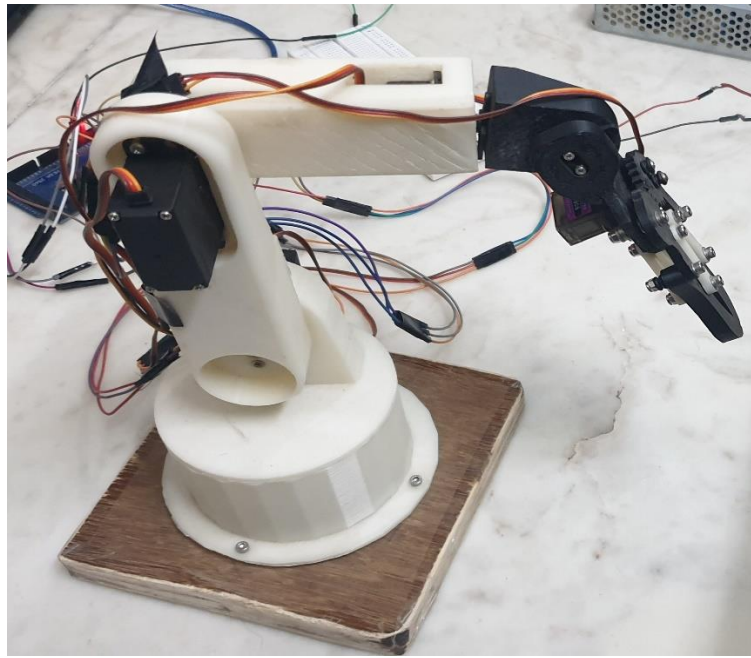


Figure 3.5: Assembled Model

3.4 Experimental Setup

The experimental set up consist of robot skeleton, 3 Mg995 servo motors, 2 Mg90 servo motors, 1 Sg90 servo motor, 1 Arduino Mega, 1 HC05 Bluetooth module, a breadboard and a SMPS module of 5V 10A output supply, which is shown in Figure 3.6.

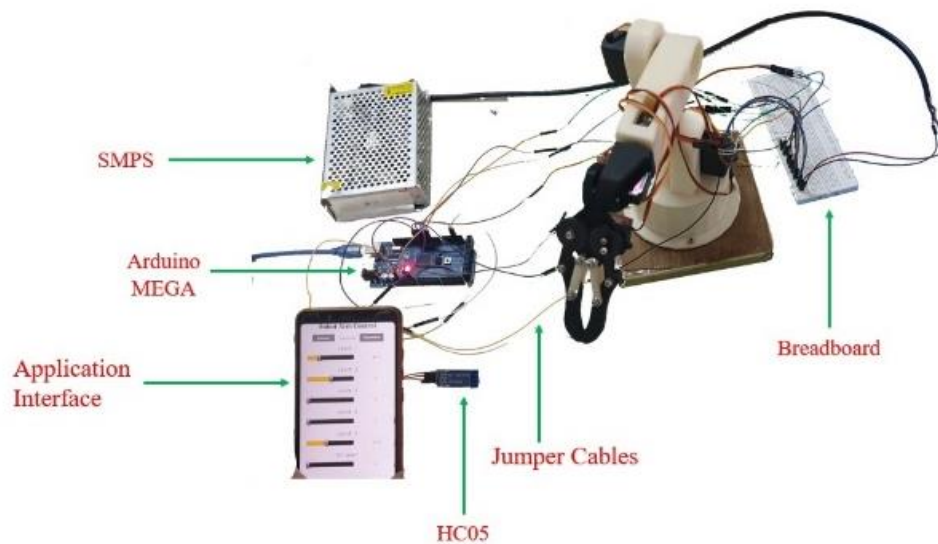


Figure 3.6: Experimental Setup

3.5 Circuit Diagram for Control of Robotic Arm

The connections between the servo motors of each joint, Arduino and Bluetooth module are shown in the circuit diagram in Figure 3.7, which is made using Fritzing software.

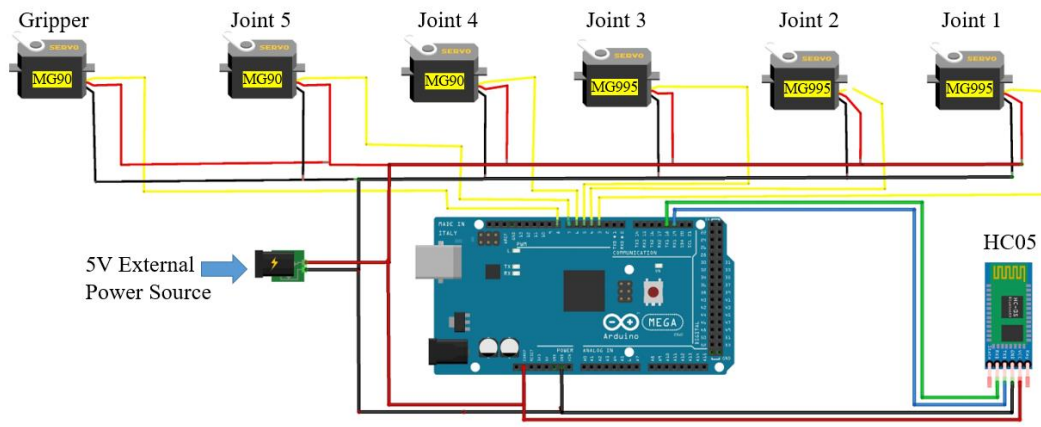


Figure 3.7: Circuit Diagram

3.5.1 Connections

The connections between motors and joints is done using jumper wires, breadboard, and SMPS. Also the Bluetooth module is connected with Arduino Mega 2560. These connections are depicted in Table 3.1.

Table 3.1: Connections of motors

Joints	Motors	GND	VCC	Signal
1	Tower Pro MG995 with stall torque 10 kg-cm at 4.8V and 180° rotation	GND of SMPS	5V of SMPS	D3
2				D4
3				D5
4	Tower Pro MgG90 with stall torque 2.2kg-cm at 4.8V and 180° rotation			D6
5				D7
6				D8

Table 3.2: Connections of Bluetooth module

Arduino Mega 2560	GND	VCC	Tx	Rx
HC05 Bluetooth module	GND	5V	Rx19	Tx18

4. KINEMATICS

Robot kinematics is the study of the motion (kinematics) of robots. Robot kinematics deals with collision avoidance, singularity avoidance and aspects of redundancy. While dealing with the kinematics used in the robotic arm, each part of the robotic arm is dealt by providing a frame of reference to it and hence a robotic arm with many parts may have many individual frames assigned to each parts having motion.

4.1 Process of Kinematic Analysis

In the kinematic analysis of industrial robotic arm position, there are two separate problems to solve: forward kinematics and inverse kinematics. Basic process for kinematic analysis which includes forward kinematics, inverse kinematics and the target achieved are represented in Figure 4.1. It involves solving the forward transformation equation to find the location of the end-effector in terms of the angles and displacements between the links. Inverse kinematics involves solving the inverse transformation equation to find the relationships between the links of the industrial robot from the location of the end effector in space.

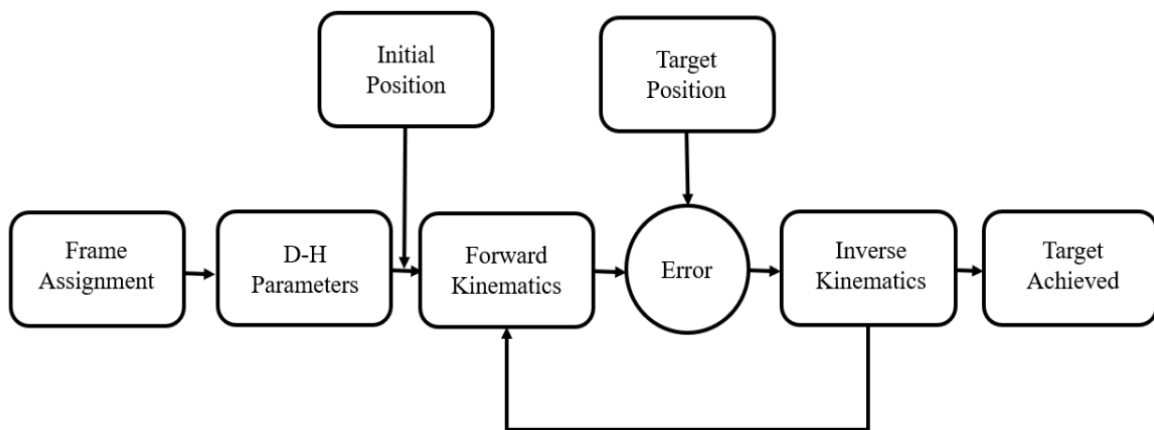


Figure 4.1: Process of Kinematic Analysis

4.2 Frame Assignment

Frame assignment of the robotic arm is carried out according to the frame assignment algorithm which is shown in the Figure 4.2.

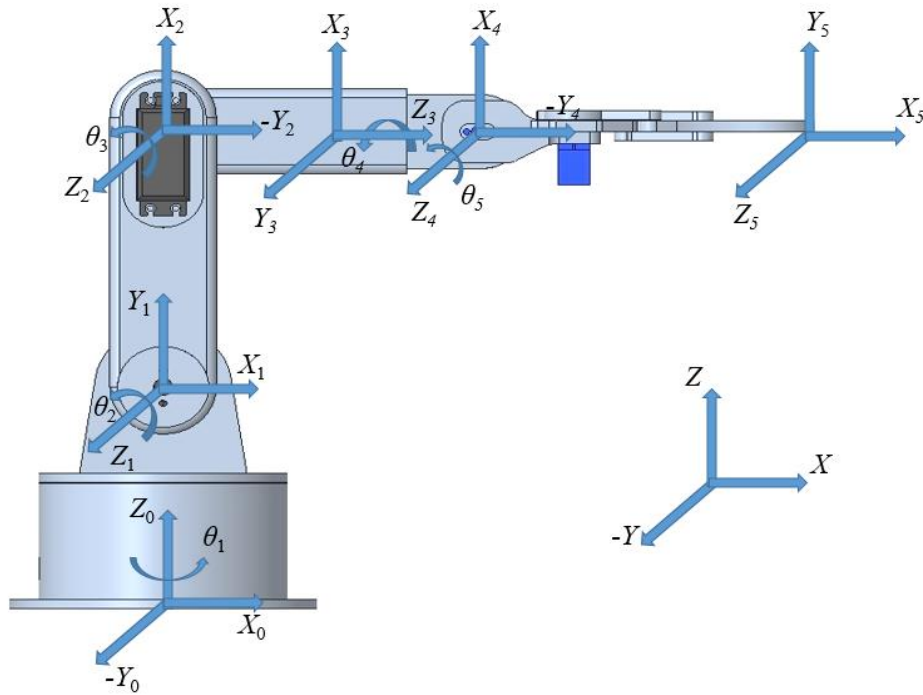


Figure 4.2: Frame Assignment

4.3 Kinematic Analysis using D-H Convention

The definition of an industrial robotic arm with four joint-link parameters for each link and a systematic procedure for assigning right-handed orthonormal coordinate frame, one to each link in an open kinematic chain is known as Denavit-Hartenberg (D-H) notation. The D-H Convention for assigning frames to links and identifying joint-link parameters is shown in Figure 4.3. It shows a pair of adjacent links, their associated joints and axes. Line segment AB in the figure is common normal to $(i-2)$ - and $(i-1)$ - axes and line segment CD is the common normal to $(i-1)$ - and i - axes.

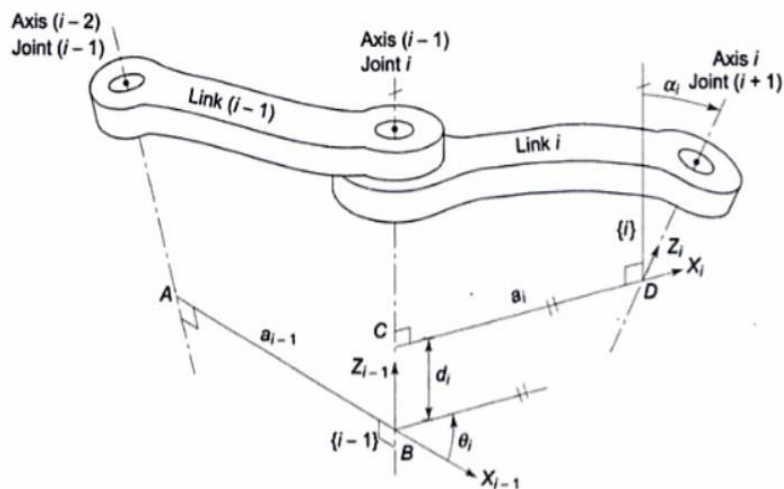


Figure 4.3: D-H Parameters

The four joint-link parameters which are used for defining the robotic arm are:

1. Link length(a_i): Distance measured along x_i -axis from the point of intersection of x_i -axis and z_{i-1} -axis (point C) to the origin of frame $\{i\}$, that is distance CD.
2. Link twist(α_i): Angle between z_{i-1} - and z_i -axes measured about x_i -axis in the right hand sense.
3. Joint distance(d_i): Distance measured along z_{i-1} -axis from the origin of frame $\{i-1\}$ (point B) to the point of intersection of x_i -axis and z_{i-1} -axis (point C), that is distance BC.
4. Joint angle(θ_i): Angle between x_{i-1} - and x_i -axes measured about z_{i-1} -axis in the right hand sense.

The D-H parameters of the robotic arm according to CAD design and frame assignment are shown in Table 4.1.

Table 4.1: D-H Parameters

Link i	Link Length (a_i) mm	Link Twist (α_i) deg	Joint Distance (d_i) mm	Joint Angle (θ_i) deg	Displacement Variable (q_i) deg
1	0	90°	99.15	θ_1	θ_1
2	123.75	0°	0	θ'_2	$\theta'_2 = \theta_2 + 90^\circ$
3	-5.1	90°	-17.05	θ_3	θ_3
4	5.1	-90°	128.1	θ_4	θ_4
5	132.39	0°	17.05	θ'_5	$\theta'_5 = \theta_5 - 90^\circ$

Transformation matrix for frame $\{i\}$ with respect to frame $\{i-1\}$ is:

$${}^{i-1}T_i = T_z(\theta_i) \times T_z(d_i) \times T_x(a_i) \times T_z(\alpha_i)$$

$${}^{i-1}T_i = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^{i-1}T_i = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \dots (4.1)$$

The transformation matrix obtained for frame $\{i\}$ with respect to frame $\{i-1\}$ given 5-DoF industrial robotic arm is indicated by Eq. 4.1.

4.4 Forward Kinematics

By using the D-H convention, calculations are performed using the homogeneous transformation matrix. The D-H conventions uses a product of four basic transformations to represent the homogeneous transformation and denoted by ${}^{i-1}T_i$. The T matrix is a homogenous 4x4 transformation matrix which describe the position of a point on an object and the orientation of the object in a three-dimensional space. The homogeneous transformation matrix from frame to frame which can be derived by using D-H parameters. The transformation matrix for each joint is given in Eq. 4.2.

$$\left. \begin{aligned} {}^0T_1 &= \begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^2T_3 &= \begin{bmatrix} C_3 & 0 & S_3 & a_3 C_3 \\ S_3 & 0 & -C_3 & a_3 S_3 \\ 0 & 1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^4T_5 &= \begin{bmatrix} C_4 & 0 & -S_4 & a_4 C_4 \\ S_4 & 0 & C_4 & a_4 S_4 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \right\} \begin{aligned} {}^1T_2 &= \begin{bmatrix} -S_2 & -C_2 & 0 & -a_2 S_2 \\ C_2 & -S_2 & 0 & a_2 C_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^3T_4 &= \begin{bmatrix} C_4 & 0 & -S_4 & a_4 C_4 \\ S_4 & 0 & C_4 & a_4 S_4 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad \dots (4.2)$$

4.5 Flowchart for Forward Kinematics

The working of kinematic analyses in the robotic arm is represented in Figure 4.4.

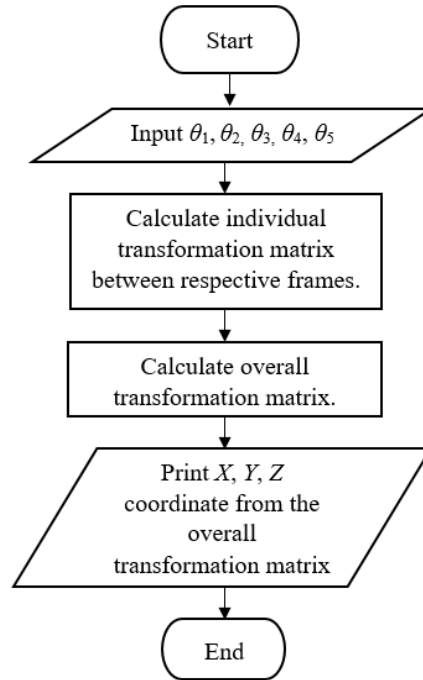


Figure 4.4: Flowchart of Forward Kinematics

4.6 Inverse Kinematics

Inverse kinematics is the application of kinematic equations to determine the motion of a robotic arm to get to a desired position. For example, to perform automated trash picking, an industrial robotic arm used in a manufacturing line needs precise motion from an initial position to a desired position between trash and manufacturing machines. So, the importance of inverse kinematics is in major condition which can be calculated by different methods. Methods for inverse kinematics are as follows:

- a) Closed Form Solution method: Closed form solution is also known as algebraic solution which uses the algebraic expression. The closed form method is significantly faster than iterative solution. In the closed form solution, joint displacements are determined as explicit function of the position and orientation of the end effector.
- b) Iterative method: Iterative methods are very well known for their applicability to any robotic configuration as it uses numerical mathematics to solve. These methods are computationally extensive and slow as compared to the closed formed method. All the iterative methods are Jacobian based which maps the joint variable to the Cartesian space or vice-versa on its inversion.

4.6.1 Concept of Iterative Method

For the inverse kinematic solution of any robot, closed form solution is not always possible with explicit equations and from forward kinematics implicit equation (nonlinear) need to be solved by the numerical approximation methods. The aim is to reduce the error (e) between the current position and target position. It gives two type of error i.e. approximated error in each iteration and linearization error. More iteration will minimize or eliminate approximated error.

$$F(q_t) = F(q) + J(q) \cdot (q_t - q_0) + L \| q_t - q_0 \|^2 \quad \dots (4.3)$$

By neglecting linearized error, it can be written as Eq. 4.4,

$$F(q_t) = F(q) + J(q) \cdot (q_t - q_0) \quad \dots (4.4)$$

So the objective function of the problem is to minimize $|(q_t - q_0)|$ that is subject to the constrain,

$$e = q \cdot (q_t - q_0) \quad \dots (4.5)$$

By inverse mapping the Jacobian, Eq. 4.6 is obtained

$$\Delta q = (q_t - q_0) = \int q^{-1} \cdot e \quad \dots (4.6)$$

So, step change in end effector position during each iteration is given by Eq. 4.7,

$$q_{i+1} = q_i + \Delta q \quad \dots (4.7)$$

4.7 Jacobian Matrix

The transformation from joint velocities to the end effector velocity is described by a matrix, called the Jacobian. The Jacobian matrix, which is dependent on manipulator configuration is a linear mapping from velocities in joint space to velocities in Cartesian space.

The inverse problem, where the joint velocities are to be determined for a given end-effector velocity is of practical importance and requires the inverse of Jacobian.

Jacobian is one of the most important tools for characterization of differential motions of the manipulator. Also, the Jacobian is useful for describing the mapping between forces applied to the end-effector and resulting torques at joints.

The Jacobian of a manipulator for prismatic and a revolute joint is represented by Eq. 4.8,

$$J_i(q) = \begin{cases} \begin{bmatrix} P_{i-1} \\ 0 \end{bmatrix} & \text{for a prismatic joint} \\ \begin{bmatrix} P_{i-1} \times {}^{i-1}P_n \\ P_{i-1} \end{bmatrix} & \text{for a revolute joint} \end{cases} \quad \dots (4.8)$$

4.8 GUI for Control of Robotic Arm

To control the robotic arm, Graphic User Interface (GUI) has been developed using MATLAB® App Designer. The user interface is displayed by Figure 4.5.

Figure 4.5: GUI developed using MATLAB®

Robotic arm can be controlled by forward and inverse kinematics. When the joint angles are provided by sliding or by typing the values in the box, it calculates the coordinates of the end effector (gripper), and gives motion to the robotic arm and displays the values of the coordinates in respective blocks.

Also, when the coordinates are given as input, it calculates the joint angles of the end effector (gripper), and gives motion to the robotic arm and displays the values of the joint angles in respective blocks.

5. TRAJECTORY PLANNING

The main aspect of trajectory planning is to describe the requisite motion of the manipulator as a time sequence of each joint, link and end effector locations and derivatives of locations, which are generated by “interpolating” the desired path by a polynomial function. The trajectory planning consists of generating a time sequence of values of different parameters attained by the polynomial interpolating the trajectory.

Every manipulator has at least capability to move from an initial location to a final desired location. The kinematics and dynamics of the manipulator govern the transition of posture with actuators exerting the desired torques. As no limits must be violated and no unmodelled behaviour should be executed, it is necessary to devise planning algorithms that generate smooth trajectories.

One of the main objectives of any trajectory planning is to achieve a smooth motion of the manipulator. The parameters involved in trajectory planning problem are shown in Figure 5.1. In trajectory planner, a user inputs a list of parameters and constraints describing the desired trajectory and trajectory planner generates a time sequence of intermediate configurations expressed in either joint or Cartesian coordinate system.

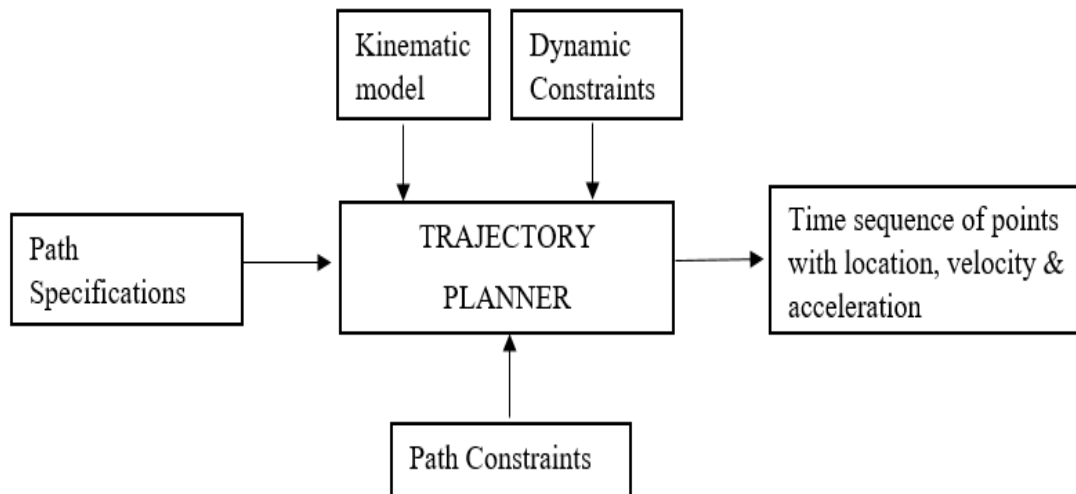


Figure 5.1: Trajectory Planning Problem

5.1 Steps in Trajectory Planning

- a) Task Description – The first step in the motion-planning problem is to identify the kind of motion required. This specification of the requisite trajectory is the input to trajectory planning algorithm. There are 3 different categories here:
- i. Point-to-point (PTP): Point-to-point motion is used in applications such as pick-n-place operation, the task is specified as initial and final end-effector. Here, except the constraint that motion be smooth, no particular specification about the intermediate location of the end cases, the user specifies only the goal point in Cartesian space for a known initial point.
 - ii. Continuous Path (CP): In addition to start and finish points, a specific path between them is required to be traced by end effector in Cartesian space, this is called Continuous Path motion and trajectory.
 - iii. Path Points (PP): This is a third type of task description, where more than two points of the path are specified. This is done to ensure better monitoring of the executed trajectory in case of applications similar to those of PTP motion.

Here, first point on path is called initial point and last point is called the goal point. The intermediate points locations are the via points. Together, these are referred to as path points.

- b) Selecting and employing a trajectory planning technique – There are 2 techniques for trajectory planning:
- i. Joint Space: In case of point-to-point motion, joint space techniques are employed in which motion planning is done at the joint level. The joint space planning schemes generate time dependent functions of all joint variables and their first two derivatives to describe the desired motion of the manipulator.
 - ii. Cartesian Space: It is used in applications where continuous path motion is required. The Cartesian Space planning techniques provide time history of location, velocity and acceleration of the end-effector with respect to the base.
- c) Computing the Trajectory – The final step is to compute the time sequence of values attained by the functions generated from the trajectory planning technique. These

values are computed at a particular path update. The path update in real time, lies between 20Hz and 200Hz in typical industrial manipulator systems.

5.2 Joint Space Techniques

The first step in joint space techniques is to obtain corresponding set of joint variable values for each specified path point, either by employing the inverse kinematics algorithm or the variable can be directly recorded if trajectory planning is performed by teaching-by-showing technique. In general, following features are required in a joint space trajectory-planning algorithm:

- An important temporal constraint is that the travelling between two path points is the same for each joint so as to make all the joints reach their corresponding path-point locations simultaneously. This guarantees the attainment of the desired location by tool with respect to the base at each point.
- Joint locations and velocities be continuous functions of time so as to ensure smooth motion.
- The interpolating function should not be computationally intensive.
- Non-smooth trajectories and other similar undesirable effects be minimized.

Here, with four constraints a 3-degree polynomial with four coefficients can be used, that is,

$$Q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad \dots (5.1)$$

5.2.1 Cubic Polynomial Trajectories

The problem of obtaining cubic polynomial trajectories is considered first, for a point-to-point motion without via points and then it is extended to case with via points.

A cubic trajectory is obtained for each joint variable, $q(t)$, of the n-DoF manipulator. The determination of the interpolation cubic polynomial is carried out for each of the n-joint variables.

Therefore, to describe the joint motion, assume that the cubic polynomial is:

$$Q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad \dots (5.2)$$

where the four constraints are:

$$q(0) = q^s$$

$$q(t_g) = q^g \quad \dots (5.3)$$

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

$$\dot{q}(t_g) = 0$$

Thus, the cubic polynomial to interpolate the path connecting the initial joint position to final joint position is:

$$Q(t) = q^s + \frac{3}{t_g^2} (q^g - q^s)t^2 - \frac{2}{t_g^3} (q^g - q^s)t^3 \quad \dots (5.4)$$

5.2.2 Parabolic Trajectory

For the given start and target points, a smooth trajectory, which consists of linear segment with parabolic blends near the end points, is used. During the blend segment with parabolic blends near the end-points, is used. During the blend portion of the trajectory, constant acceleration is used to achieve a smooth velocity transition. The two parabolic blends are assumed to be identical, that is, they have same constant acceleration. Therefore, parabolic blends near the path points are of same duration (t_b) and the whole trajectory is symmetric about the halfway point in time and position (t_m, q^m). Thus, the trajectory gives continuity of position, velocity and acceleration throughout.

Here, both constraints initial and final velocities are zero and the joint position profile is symmetrical about the point (q^m, t_m) with:

$$q^m = \frac{q^s + q^g}{2} \text{ and } t_m = \frac{t_g}{2} \quad \dots (5.5)$$

The trajectory has to satisfy the additional constraint for smooth transition from parabolic blend to linear segment. For joint velocity to be continuous, the velocity at the end of the first blend must be equal to the velocity of the linear segment, that is, at the blend point (q^m, t_b)

$$\ddot{q}^c t_b = \frac{q^m - q^b}{t_m - t_b} \quad \dots (5.6)$$

where q^b is the value of the joint variable at the end of the parabolic segment, at time t_b . For constant acceleration \ddot{q}^c , the joint position q^b , at the end of the first blend is given by:

$$q^b = q^s + \frac{1}{2}\ddot{q}^c t_b^2 \quad \dots (5.7)$$

Combining equations 5.5, 5.6, 5.7 gives us quadratic equation in t_b as:

$$\ddot{q}^c t_b^2 - \ddot{q}^c t_g t_b + (q^g - q^s) = 0 \quad \dots (5.8)$$

To predict the time history of q , \dot{q} , and \ddot{q} , with given q^s , q^g , and t_g , the only unknown parameter is t_b , the blend duration. Solving Eq. 5.8 for t_b , gives:

$$t_b = \frac{1}{2}t_g - \frac{1}{2}\sqrt{\frac{t_g^2 - 4(q^g - q^s)}{\ddot{q}^c}} \quad \dots (5.9)$$

There are many solutions to Eq. 5.9 depending on value of \ddot{q}^c but the answer is always symmetric about point (q^m, t_b) . The constraint on choice of acceleration during the blend \ddot{q}^c , is:

$$|\ddot{q}^c| \geq \frac{4|q^g - q^s|}{t_g^2}; \text{ and } \ddot{q}^c \neq 0 \quad \dots (5.10)$$

The trapezoidal velocity profile trajectory using equation generates the following sequence of polynomials for the time history of joint position, velocity and acceleration.

$$q(t) = \begin{cases} q^s + \frac{1}{2}\ddot{q}^c t^2 & 0 \leq t \leq t_b \\ q^s - \frac{1}{2}\ddot{q}^c t_b^2 + \ddot{q}^c t_b t & t_b < t \leq t_g - t_b \\ q^s - \frac{1}{2}\ddot{q}^c (t_g - t)^2 & t_g - t_b < t \leq t_g \end{cases} \quad \dots (5.11)$$

$$\dot{q}(t) = \begin{cases} \ddot{q}^c t & 0 \leq t \leq t_b \\ \ddot{q}^c t_b & t_b < t \leq t_g - t_b \\ \ddot{q}^c (t_g - t) & t_g - t_b < t \leq t_g \end{cases} \quad \dots (5.12)$$

$$\ddot{q}(t) = \begin{cases} \ddot{q}^c & 0 \leq t \leq t_b \\ 0 & t_b < t \leq t_g - t_b \\ -\ddot{q}^c & t_g - t_b < t \leq t_g \end{cases} \quad \dots (5.13)$$

5.2.3 Cycloidal Trajectory

The equation for cycloidal trajectory are as follows:

$$P(t) = P_0 + \left(\frac{t}{t_f} - \frac{1}{2\pi} \sin \frac{2\pi t}{t_p} \right) (P_f - P_0) \quad \dots (5.14)$$

$$\frac{d}{dt} P(t) = \left(\frac{1}{t_f} - \frac{1}{t_p} \cos \frac{2\pi t}{t_p} \right) (P_f - P_0) \quad \dots (5.15)$$

$$\frac{d^2}{dt^2} P(t) = \frac{2\pi}{t_f^2} (P_f - P_0) \sin \frac{2\pi t}{t_p} \quad \dots (5.16)$$

5.2.4 Circular Path

The circular path $P(c)$ is specified using the following parameters:

- Radius of circular path is ρ in plane $x'y'$.
- Unit vector of the circle axis \hat{r} at the centre of circle directed according to right-hand rule.
- A point D on the axis of circle passing through the centre of circle.

$${}^0C = {}^0D + ({}^0\delta^T \cdot \hat{r}) \hat{r} \quad \dots (5.17)$$

Any point P on the circle $P(c)$ can easily be described in terms of frame $\{x' y' z'\}$ as:

$$P' = \begin{bmatrix} \rho \cos(c/\rho) \\ \rho \sin(c/\rho) \\ 0 \end{bmatrix} \quad \dots (5.18)$$

Finally, the parametric description of circle w.r.t frame $\{0\}$ is obtained by mapping, P' to base frame as:

$${}^0P(c) = {}^0C + R P'(c) \quad \dots (5.19)$$

where R is rotation matrix of frame $\{x' y' z'\}$.

5.3 Position Planning

For a path 0P to be traversed by given n-DoF manipulator in a time t_g , first the extreme values of path coordinate c are determined. For the given path, assume $c = 0$ at time $t = 0$ is the initial

position and $c = c_g$ at $t = t_g$ is the goal position. Next, a function $c(t)$, which interpolates c between $c = 0$ and $c = c_g$ in a smooth way, is obtained. This smooth function $c(t)$ could either be a cubic polynomial or a linear segment with parabolic blends and is obtained as discussed in the case of joint-space schemes for point-to-point motion without via points. Now that a time law is imposed on the path coordinate c , it is possible to obtain time history of 0P by substituting the values attained by $c(t)$, at each instant of time,

The linear velocity of the end-effector is determined as:

$${}^0\dot{P} = \dot{c} \frac{d({}^0P)}{dc} = \hat{a}\dot{c} \quad \dots (5.20)$$

Linear acceleration of the tool is obtained by differentiating above equation w.r.t to time

$${}^0\ddot{P} = \dot{c}^2 \frac{d^2({}^0P)}{dc^2} + \dot{c} \frac{d({}^0P)}{dc} \quad \dots (5.21)$$

Thus, for a Cartesian straight-line motion, 0P is obtained from Eq. 5.20 and 5.21 is:

$${}^0\dot{P} = \frac{\dot{s}}{|{}^0P_g - {}^0P_s|} ({}^0\dot{P}_g - {}^0\dot{P}_s) = \hat{a}\dot{c} \quad \dots (5.22)$$

$${}^0\ddot{P} = \frac{\dot{s}}{|{}^0P_g - {}^0P_s|} ({}^0\ddot{P}_g - {}^0\ddot{P}_s) = \hat{a}\ddot{c} \quad \dots (5.23)$$

Similarly, for circular motion, from Eq. 5.20 and 5.21, the velocity and acceleration of point P on the circle are as:

$${}^0\dot{P} = R\dot{c} \begin{bmatrix} -\sin\left(\frac{c}{\rho}\right) \\ \cos\left(\frac{c}{\rho}\right) \\ 0 \end{bmatrix} \quad \dots (5.24)$$

$${}^0\ddot{P} = R \begin{bmatrix} -\frac{\dot{c}^2}{\rho} \cos\left(\frac{c}{\rho}\right) - \ddot{c} \sin\left(\frac{c}{\rho}\right) \\ -\frac{\dot{c}^2}{\rho} \sin\left(\frac{c}{\rho}\right) + \ddot{c} \cos\left(\frac{c}{\rho}\right) \\ 0 \end{bmatrix} \quad \dots (5.25)$$

5.4 Orientation Planning

Now the time sequence of orientations obtained by the end-effector is needed to be obtained. These orientations cannot be specified by rotation matrices because interpolation of two-rotation matrices does not always yield a valid rotation matrix.

Supposing, ZYX-Euler angle representation is used to express the end-effector orientation w.r.t the base, the angles ϕ, θ and ψ are defined as the Euler angles.

$$\alpha = [\phi \quad \theta \quad \psi]^T \quad \dots (5.26)$$

because α is similar to 0P , discussed in position planning, any path on the end-effectors orientation along with temporal constraints can be imposed.

$$\alpha_s = [\phi_s \quad \theta_s \quad \psi_s]^T \text{ to } \alpha_g = [\phi_g \quad \theta_g \quad \psi_g]^T \text{ in time } t = t_g.$$

Following steps should be incorporated to generate a smooth trajectory

1. Obtain the parametric description of given path joining as to α_s to α_g .
2. Impose a time law on path coordinate c , as done in position planning in Cartesian space.
3. Obtain the time sequence of values attained by $\alpha(t)$ based on $c(t)$.
4. As explained earlier for position planning, obtain the history of $\dot{\alpha}(t)$ and $\ddot{\alpha}(t)$.

For example, a linear segment joint α_s and α_g has the following equation for $\dot{\alpha}(t)$ and $\ddot{\alpha}(t)$

$$\dot{\alpha}(t) = \frac{\dot{c}}{\|\alpha_g - \alpha_s\|} (\alpha_g - \alpha_s) \quad \dots (5.27)$$

$$\ddot{\alpha}(t) = \frac{\ddot{c}}{\|\alpha_g - \alpha_s\|} (\alpha_g - \alpha_s) \quad \dots (5.28)$$

5.5 Results based on MATLAB®

The Graphical User Interface (GUI) for performing trajectory planning is developed in MATLAB® software for Joint space and Cartesian space. Initially angles for joint 1, 2, 3, 4, 5 are taken as 0 and final angles are 45°, -40°, 20°, 10°, -70° respectively after 3 seconds.

Trajectory Planning

Choose the trajectory technique

☒ Joint Space
☐ Cartesian Space

Start Point	$\theta 1$	<input type="text" value="0"/>	$\theta 2$	<input type="text" value="0"/>	$\theta 3$	<input type="text" value="0"/>	$\theta 4$	<input type="text" value="0"/>	$\theta 5$	<input type="text" value="0"/>
Goal Point	$\theta 1$	<input type="text" value="0"/>	$\theta 2$	<input type="text" value="0"/>	$\theta 3$	<input type="text" value="0"/>	$\theta 4$	<input type="text" value="0"/>	$\theta 5$	<input type="text" value="0"/>

☒ Cubic ☐ Cycloidal ☐ Parabolic

Total Time

Choose

☒ Joint
☐ Link

☒ Position
☐ Velocity
☐ Acceleration

☐ Joint1
☐ Joint2
☐ Joint3
☐ Joint4
☐ Joint5

Figure 5.2: Interface for Joint Space

Trajectory Planning

Choose the trajectory technique

☐ Joint Space
☒ Cartesian Space

X(s)	<input type="text"/>	R(r)	<input type="text"/>	s(t)	<input type="text"/>
Y(s)	<input type="text"/>	P(r)	<input type="text"/>	r(t)	<input type="text"/>
Z(s)	<input type="text"/>	Y(r)	<input type="text"/>		

Total Time

Choose

☐ Joint
☒ Link

☒ Position ☐ Angular Velocity
☐ Velocity ☐ Angular Acceleration
☐ Acceleration

Axis

☒ X
☐ Y
☐ Z

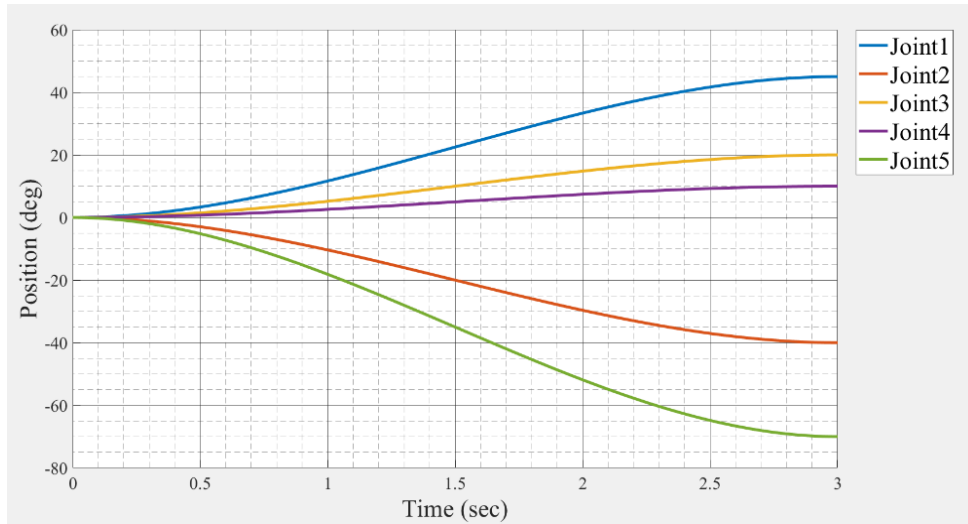
☐ Link1
☐ Link2
☐ Link3
☐ Link4
☐ Link5

Figure 5.3: Interface for Cartesian Space

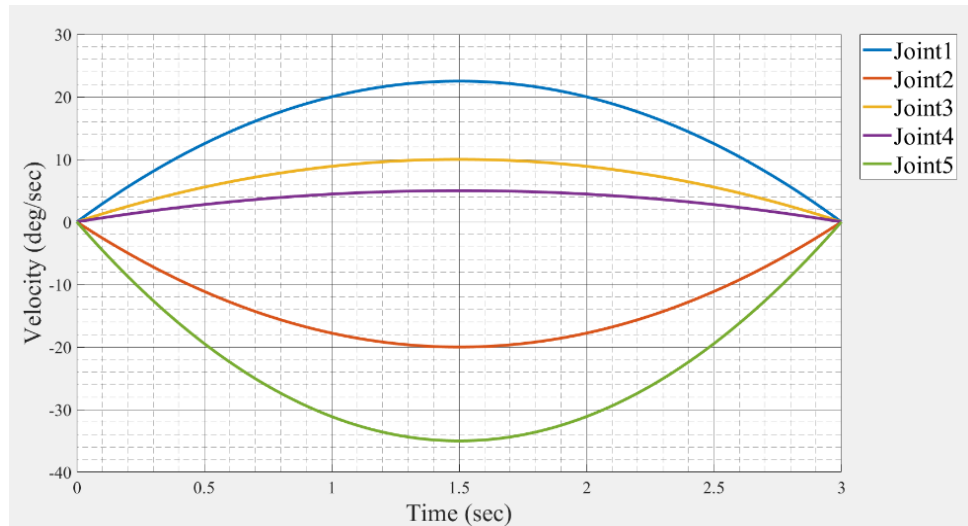
The GUI's for joint and Cartesian spaces are represented by Figures 5.2 and 5.3 respectively. In joint space, the initial and final joint angles are given as input while in Cartesian space, the path constraints are explicitly specified to get the desired output.

5.6 Results of Joint Space Trajectory Technique

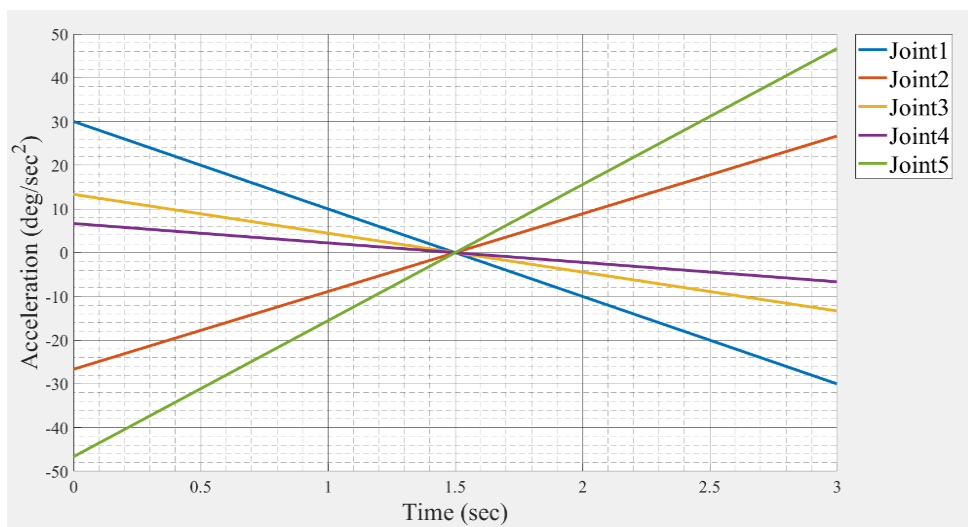
The initial and final joint angles of the robotic arm are given as input and the parameters of each joint is plotted against time which is represented below. The position, velocity and acceleration of joint with respect to time is shown by Figures 5.4(a), 5.4(b) and 5.4(c).



(a) Position Versus Time



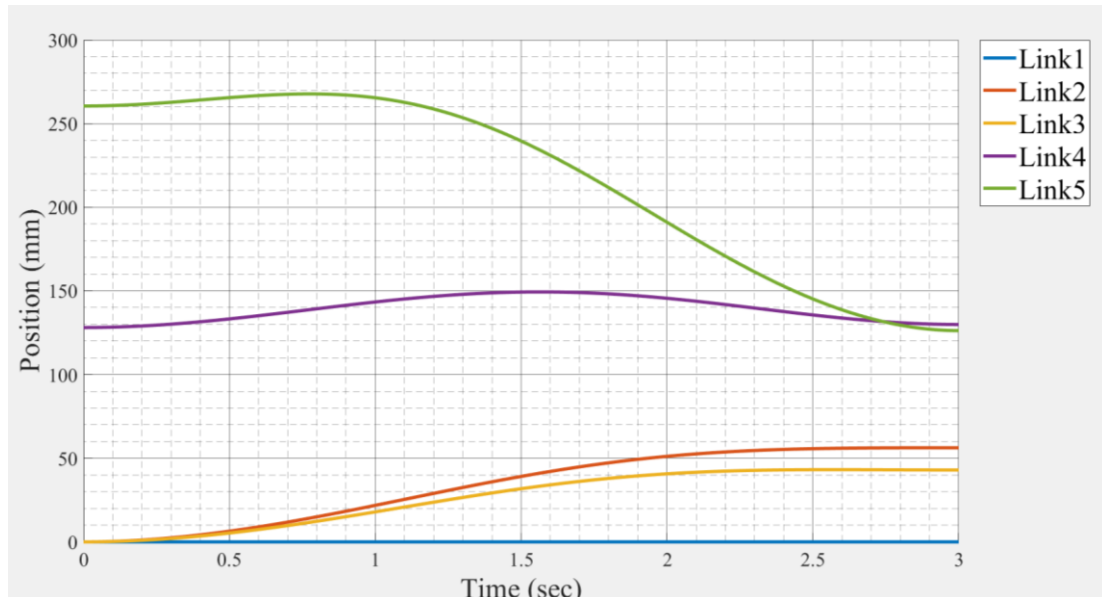
(b) Velocity Versus Time



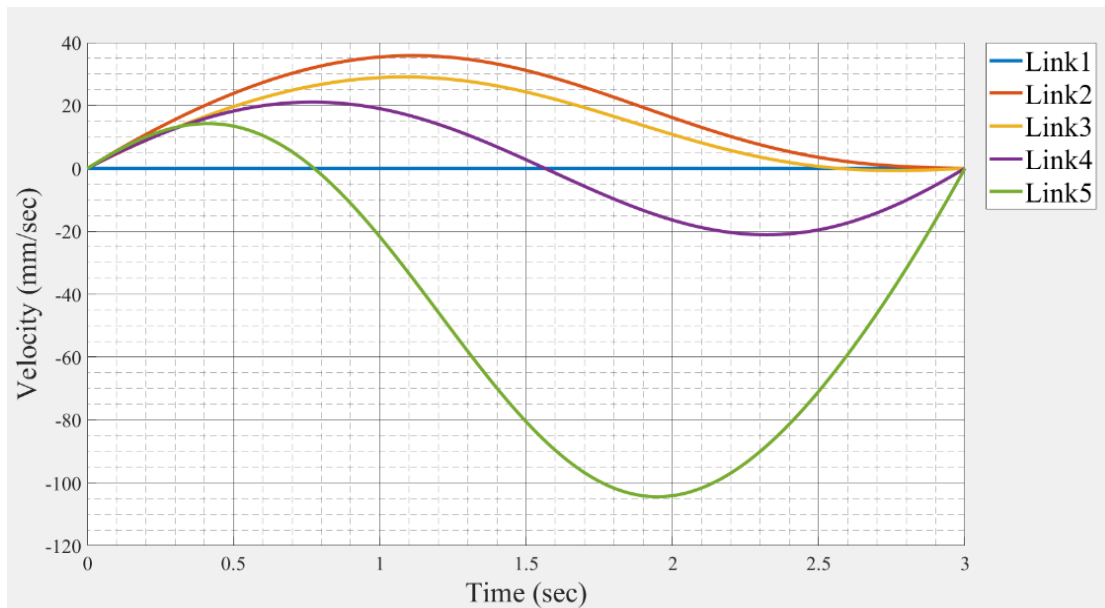
(c) Acceleration Versus Time

Figure 5.4: Parameters for Joints

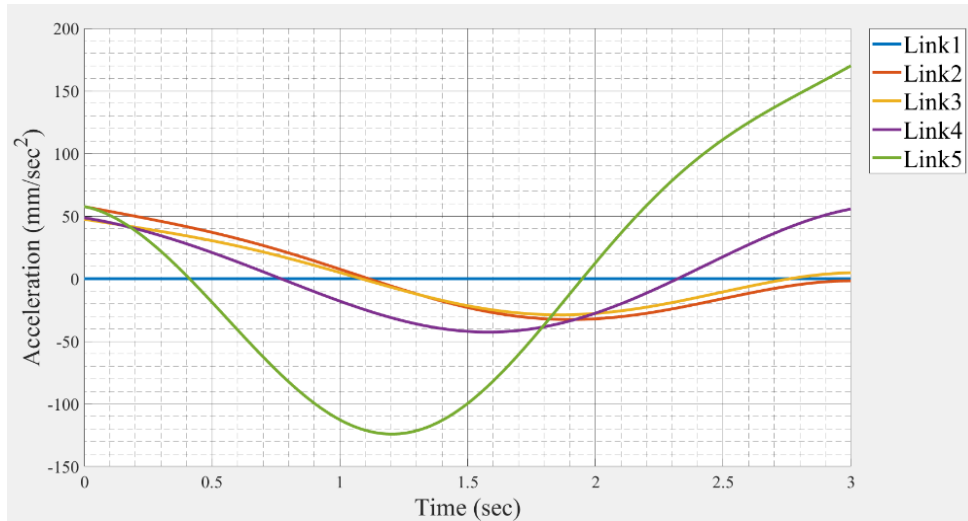
The position, velocity, acceleration, angular velocity and angular acceleration of each link of the robotic arm are plotted against time with respect to X-axis which is shown by Figures 5.5(a) to 5.5(e) respectively.



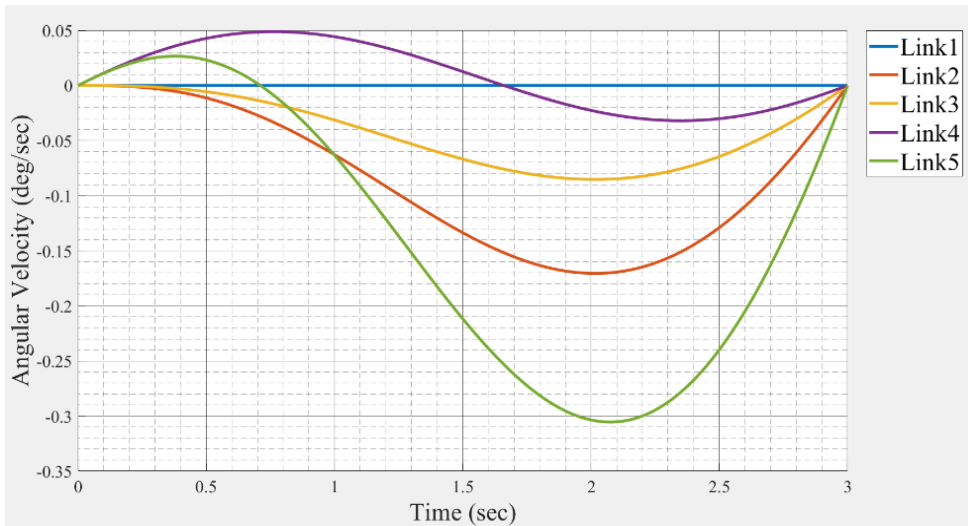
(a) Position Versus Time



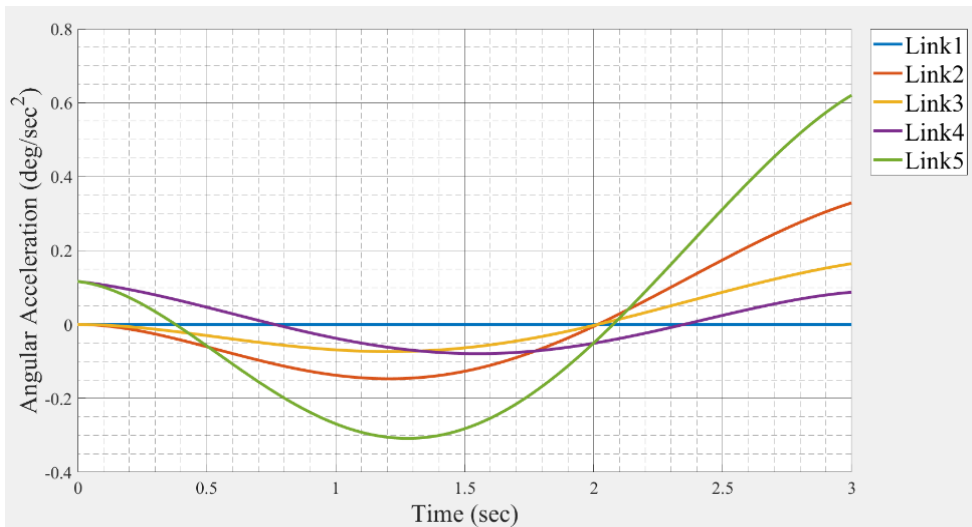
(b) Velocity Versus Time



(c) Acceleration Versus Time



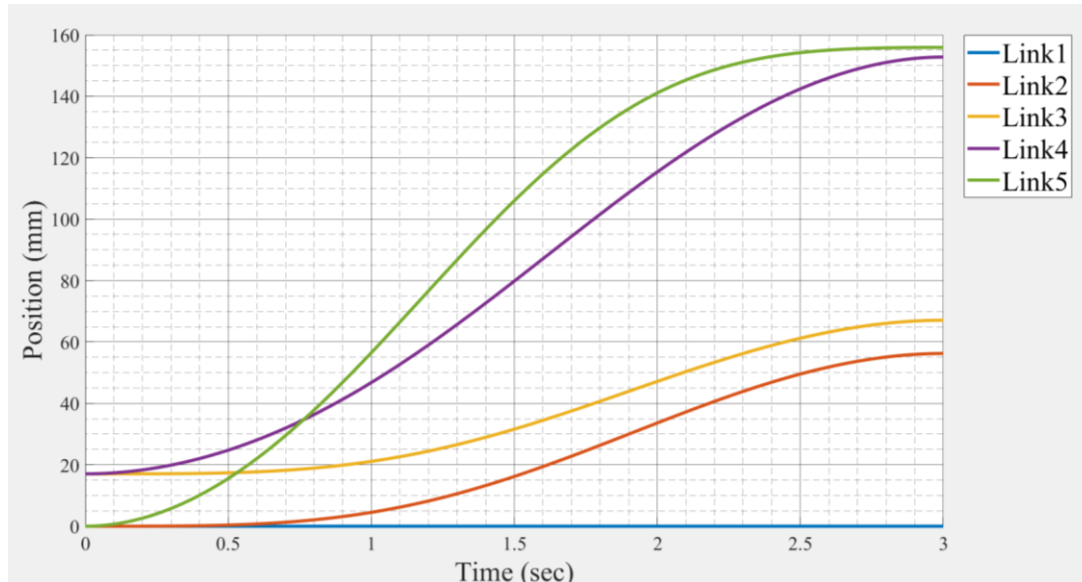
(d) Angular Velocity Versus Time



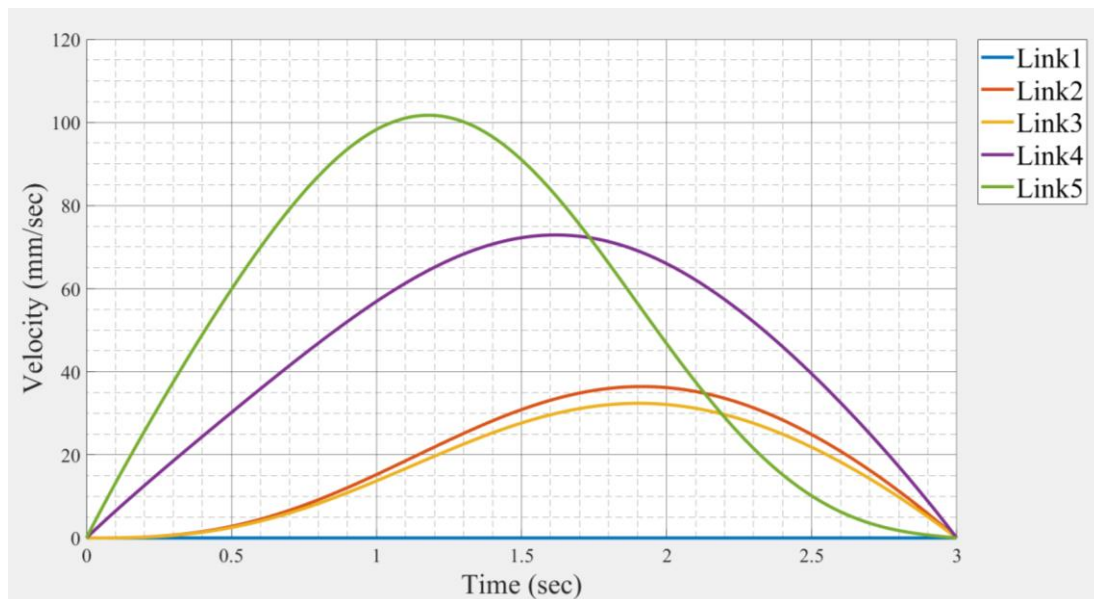
(e) Angular Acceleration Versus Time

Figure 5.5: Parameters for Links in X-axis

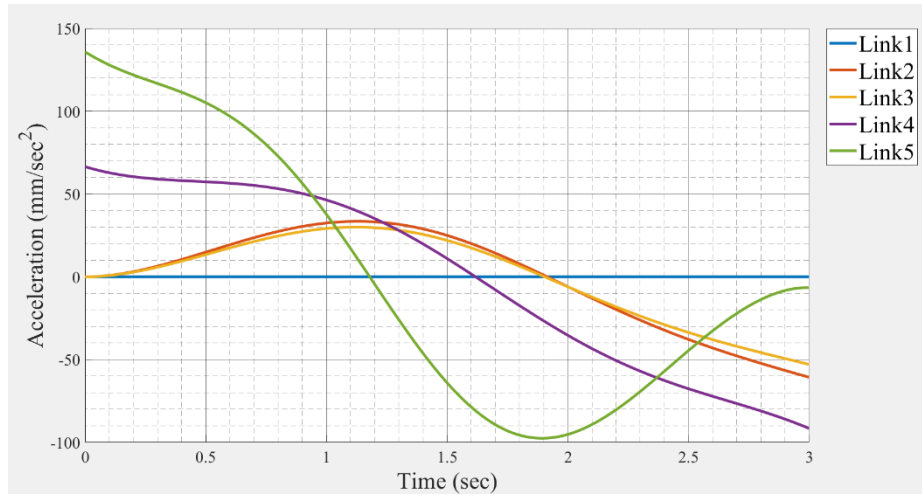
The position, velocity, acceleration, angular velocity and angular acceleration of each link of the robotic arm are plotted against time with respect to Y-axis which is shown by figures 5.6(a) to 5.6(e) respectively.



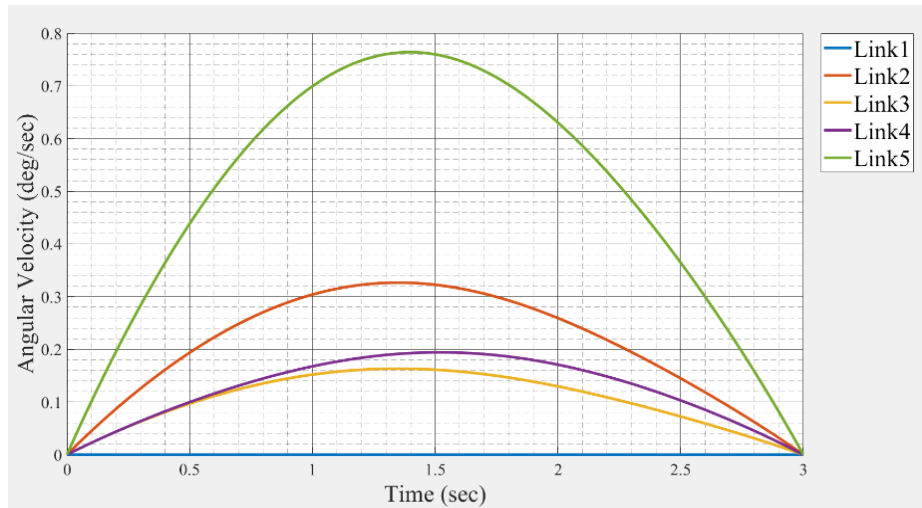
(a) Position Versus Time



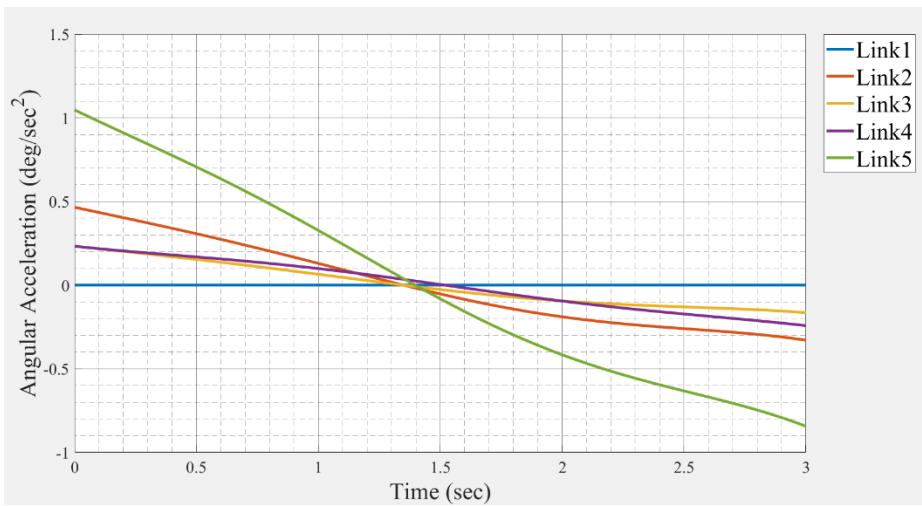
(b) Velocity Versus Time



(c) Acceleration Versus Time



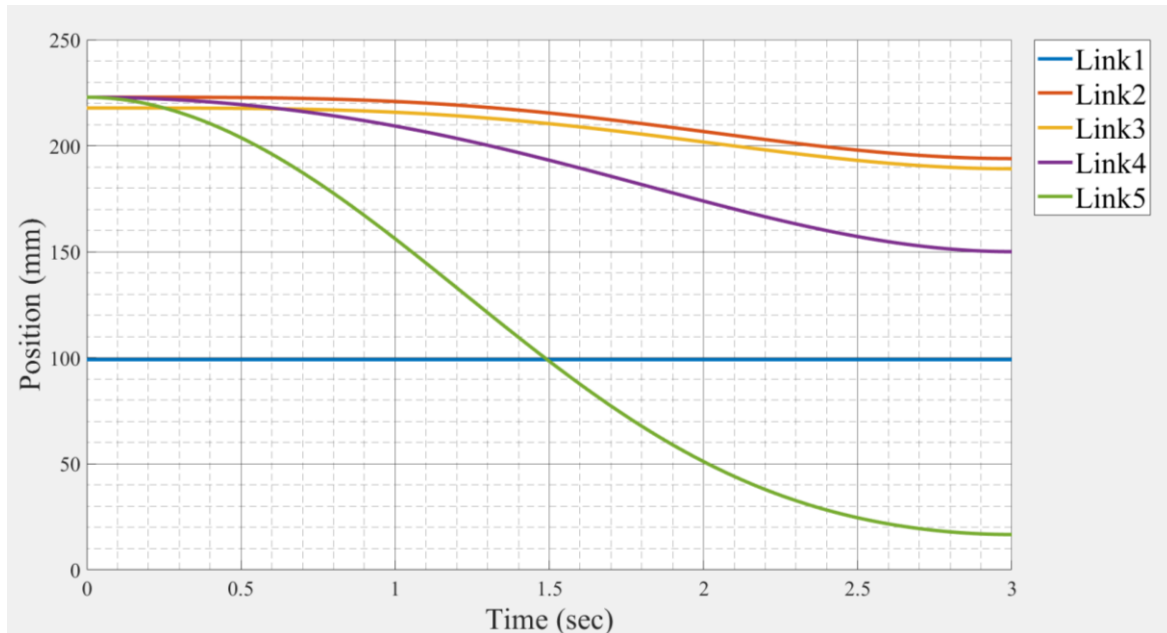
(d) Angular Velocity Versus Time



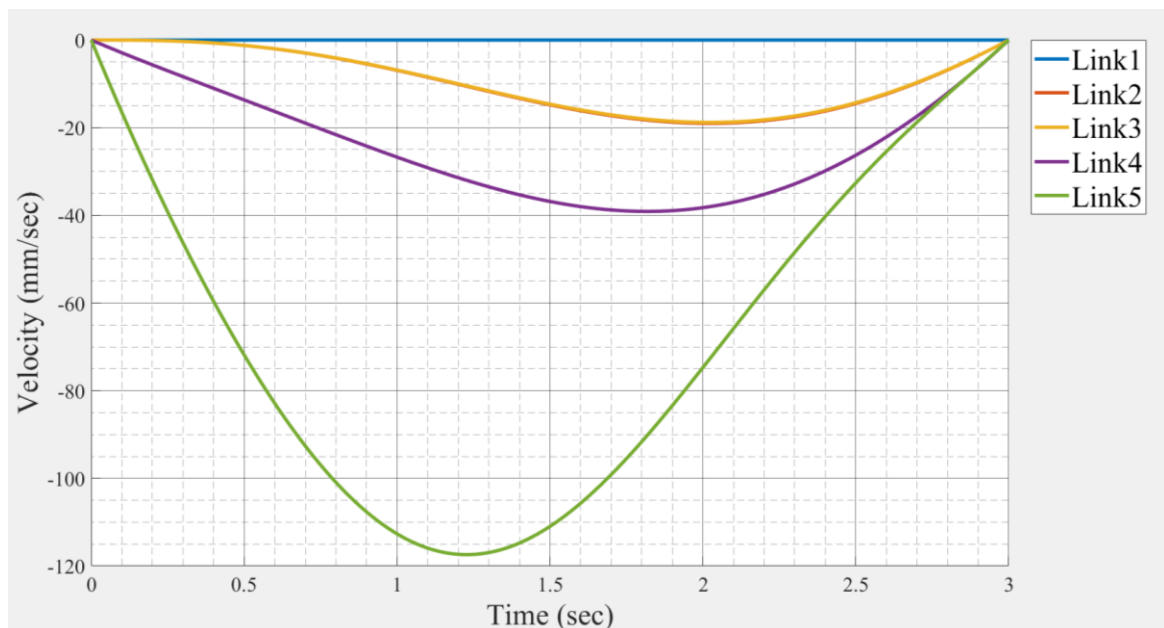
(e) Angular Acceleration Versus Time

Figure 5.6: Parameters for Links in Y-axis

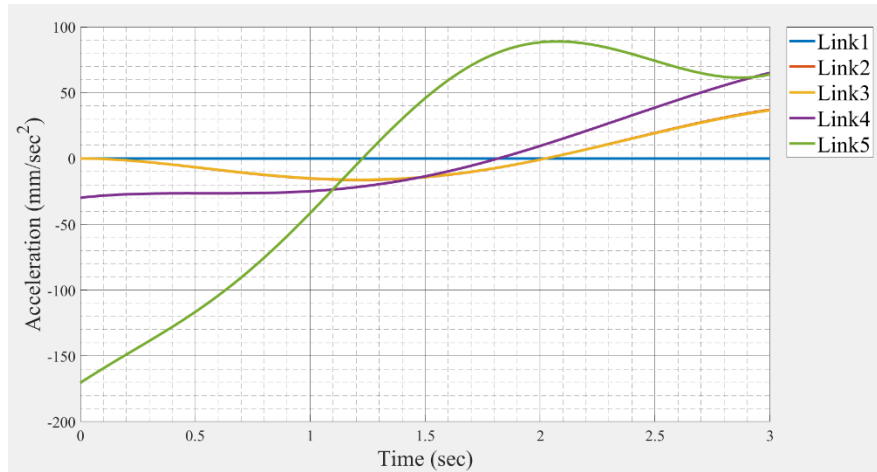
The position, velocity, acceleration, angular velocity and angular acceleration of each link of the robotic arm are plotted against time with respect to Z-axis which is shown by Figures 5.7(a) to 5.7(e) respectively.



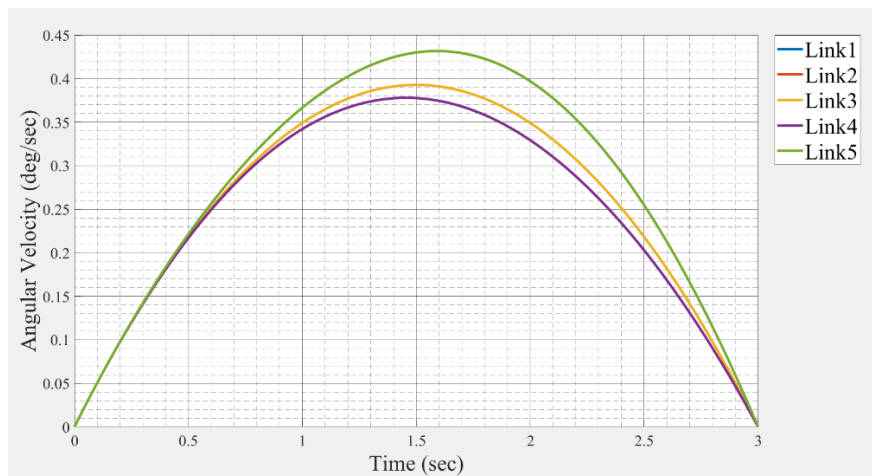
(a) Position Versus Time



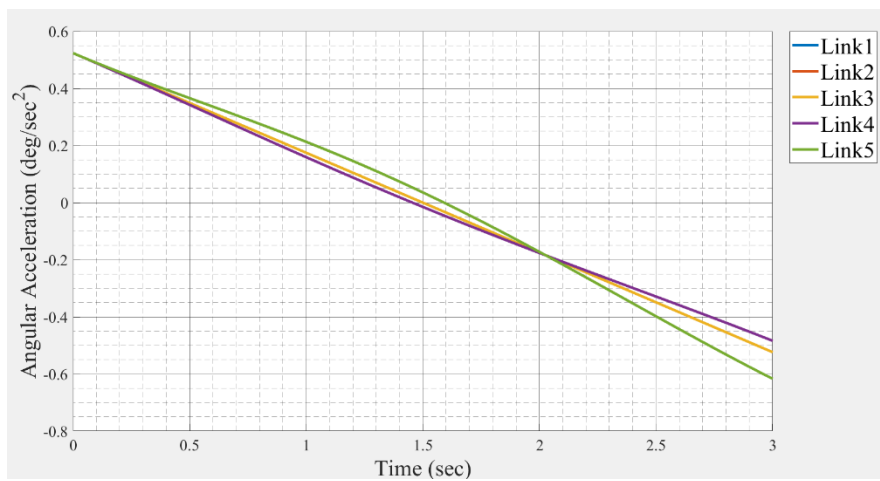
(b) Velocity Versus Time



(c) Acceleration Versus Time



(d) Angular Velocity Versus Time



(e) Angular Acceleration Versus Time

Figure 5.7: Parameters for Links in Z-axis

The snapshot of simulation of a robotic arm performing motion on a defined trajectory is shown in the Figure 5.8. The robotic arm is performing motion on a motion with a cubic trajectory.

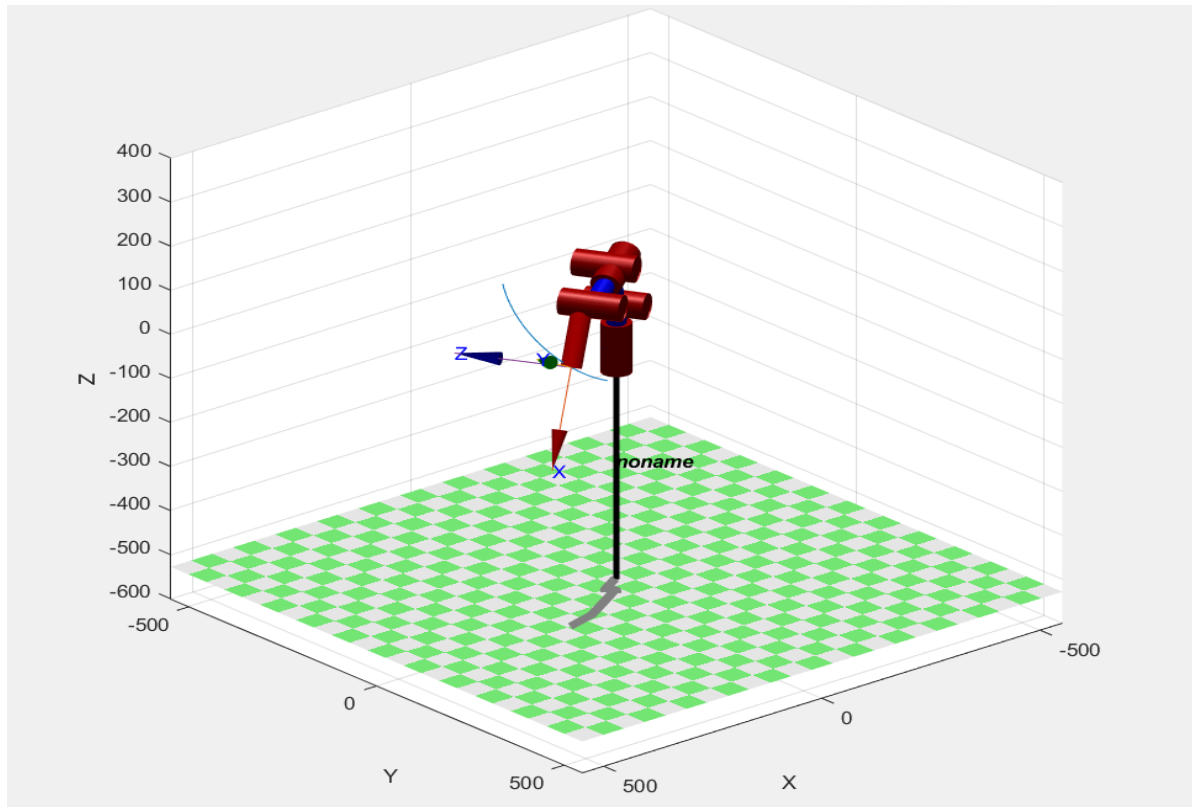


Figure 5.8: Simulation

6. BLUETOOTH CONTROL

Bluetooth is a wireless communications protocol running at 2.4 GHz, with ISM band, suitable for forming personal area networks. It is designed for low power devices such as mobile phones. Bluetooth comes as standard on all the mobile phones, and desktop computers. It can be easily fitted with a module to allow Bluetooth communication.

6.1 Working of Robotic Arm by Controlling with Bluetooth

In industrial robotic environments, a variety of tasks are performed by robotic arm. Now, the handling and controlling of these robotic arm can be done by various techniques. The user must obtain access to the teach pendant or terminal to monitor the changes to the programming of the robot. One of the ways in which access can be obtained is by using Bluetooth. A Bluetooth module can be attached with the controller and by developing an android application or software, the robotic arm can be controlled manually.

6.2 Application for Bluetooth Control

The application to control the robot with the help of the Bluetooth was made using MIT App Inventor 2. The making of the app includes first designing the app in the designer and then block programming in blocks view.

6.2.1 MIT App Inventor 2

MIT App Inventor is an intuitive, visual programming environment that allows everyone to build fully functional apps for Android phones, iPhones, and Android/iOS tablets. Those who are new to MIT App Inventor can have a simple first app running in less than 30 minutes. The blocks-based tool in the software facilitates the creation of complex, high-impact apps in significantly less time than traditional programming environments. The MIT App Inventor project seeks to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation.

Block-based coding programs inspire intellectual and creative empowerment. MIT App Inventor goes beyond this to provide real empowerment for kids to make a difference – a way to achieve social impact of immeasurable value to their communities.

6.2.2 User Interface

The interface of the application developed for performing forward kinematics on the robotic arm is shown in Figure 6.2.



Figure 6.1: User Interface of the App

Simple Text can be written inside a label. The text encircled in Figure 6.3 are labels.



Figure 6.2: Labels

Sliders: The sliders are to move the respective joints through some angle and slider limits are set according to the joint limits of the robot.

Positions: A user can save 10 different positions in the application and run them in the saved fashion.

The text box shown in Figure 6.4 are used to write input by the user but here they are used as read only.

Robot Arm Control

Connect
Disconnected
Disconnect

Joint 1

0

Joint 2

0

Joint 3

0

Joint 4

0

Joint 5

0

Gripper

0

X

260.49

Y

0

Z

222.90

SAVE
RUN
RESET

Positions saved : 0

Figure 6.3: Text Box

Then comes the buttons to send signals when pressed which are highlighted in Figure 6.5.

Robot Arm Control

Connect
Disconnected
Disconnect

Joint 1

0

Joint 2

0

Joint 3

0

Joint 4

0

Joint 5

0

Gripper

0

X

260.49

Y

0

Z

222.90

SAVE
RUN
RESET

Positions saved : 0

Figure 6.4: Functioning Buttons

6.2.4 Working

When the connect button is pressed a list opens which shows all paired devices of the mobile phone as shown in Figure 6.5. Then you have to select the Bluetooth module to connect it to the application.



Figure 6.5: Connectivity Interface

Disconnected button: When pressed, it disconnects the Bluetooth.

Slider: The slider when moved sends the appropriate signal to move the respective joints.

SAVE button: It saves the current position of the robot.

RUN button: It runs the robot through the saved position in an infinite loop and turns into a **PAUSE button** which can be used to pause the robot loop.

RESET button: It is used to stop the run infinite loop and delete all the saved position.

6.3 Flowchart for Working of Application

The working of the application developed is shown in the flowchart given in Figure 6.6.

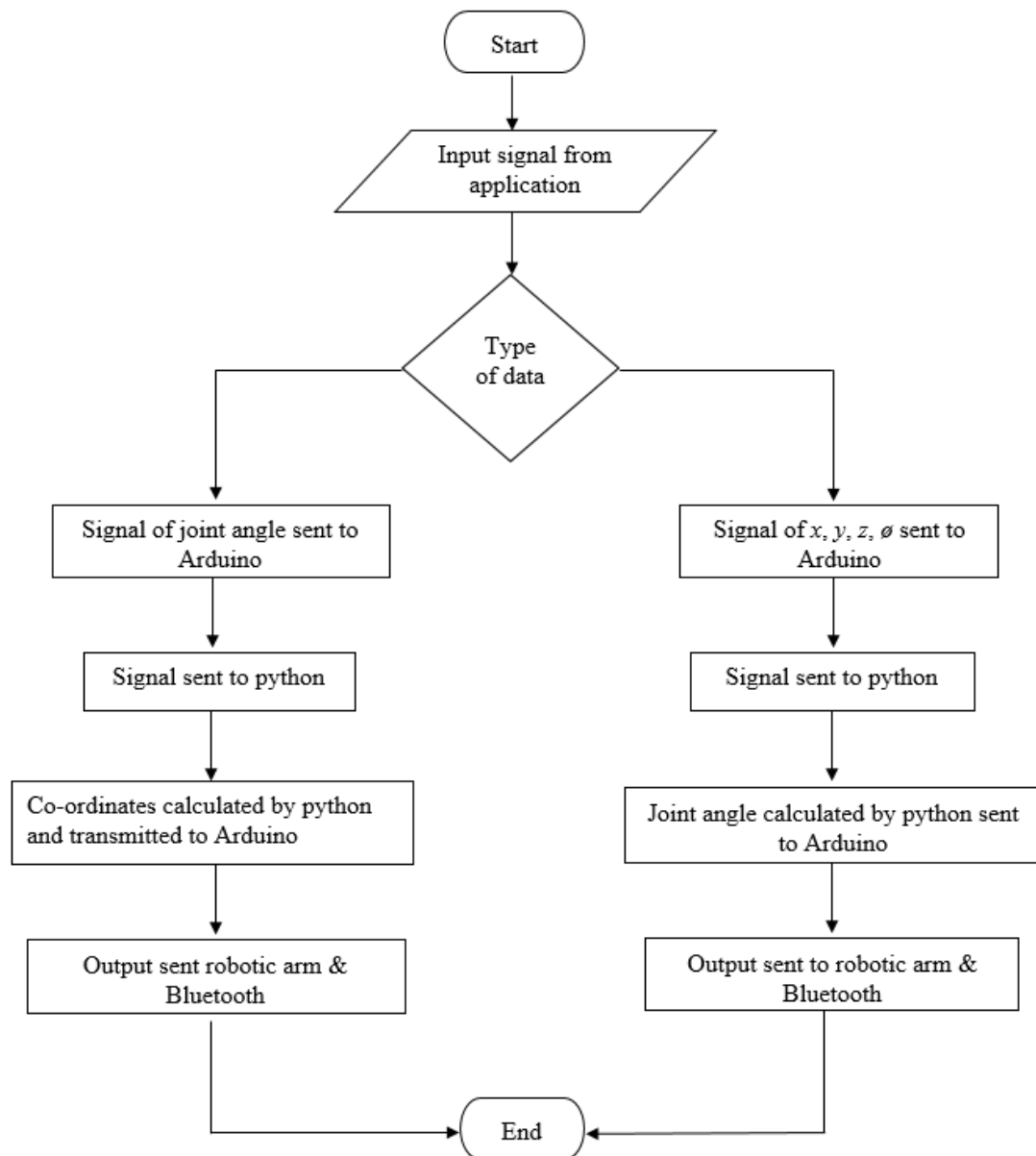


Figure 6.6: Flowchart of Android Application

6.4 Application of Bluetooth after Inverse Kinematics

The interface of application with inclusion of inverse kinematics control is shown by Figure 6.7. Forward and inverse kinematics can be computed in same interface.

Robot Arm Control

Connect Disconnected Disconnect

Forward

Joint 1

0

Joint 2

0

Joint 3

0

Joint 4

0

Joint 5

0

Gripper

Gripper close ☐

Send

Inverse

X

Y

Z

Ø

Send

SAVE RUN RESET

Positions saved : 0

Figure 6.7: Application Interface for Inverse

6.5 Performance of Pick-n-Place using Bluetooth

The manufactured prototype of robotic arm is being controlled by a Bluetooth module HC-05, and commands to this module is given from the application developed in MIT App Inventor 2. The initial and final positions of an object were saved and then processed and run for picking the object from initial position and placing it to the final position. The gripper holding a paper ball, is shown in Figure 6.8.

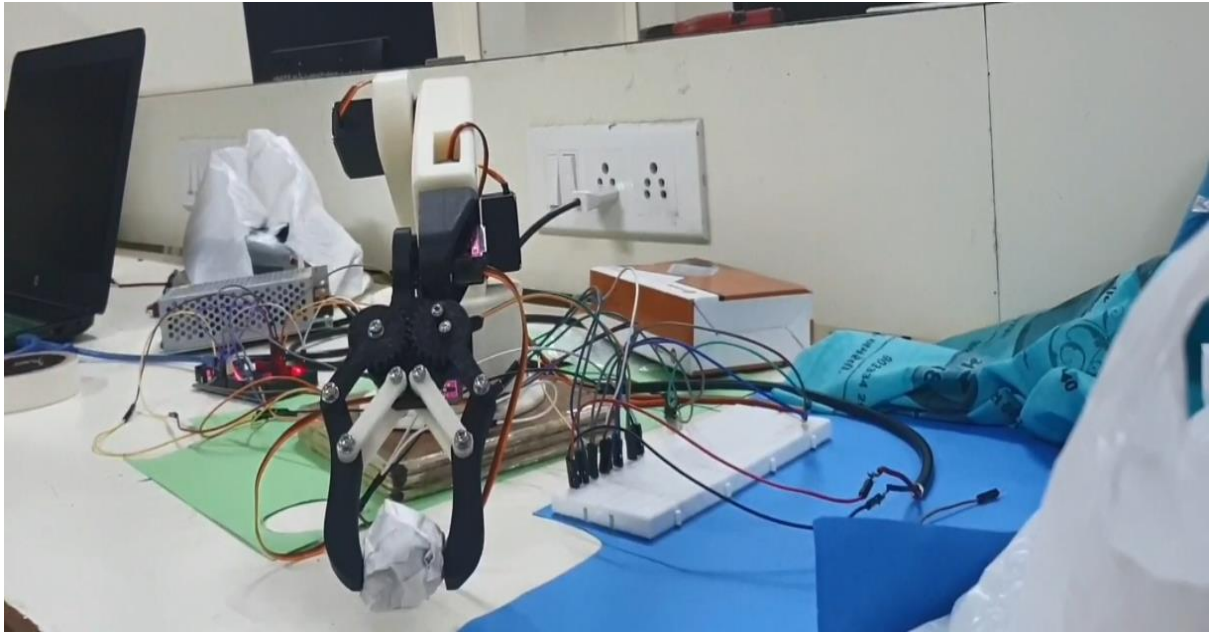


Figure 6.8: Pick-n-Place Operation

7. RESULT DISCUSSION, CONCLUSION AND FUTURE WORK

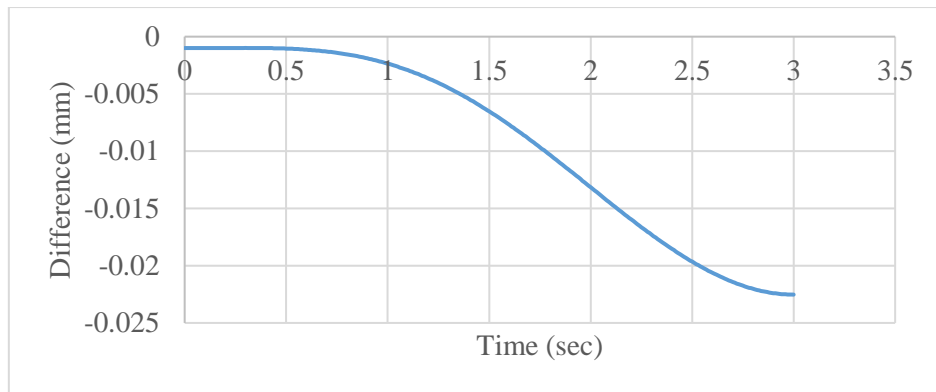
The final chapter includes comparison of the cubic trajectory analysis done in previous chapter with same trajectory done using CREO®. The results of comparison are shown using error graphs of position, velocity and acceleration of end effector for respective axis.

7.1 Results and Discussion

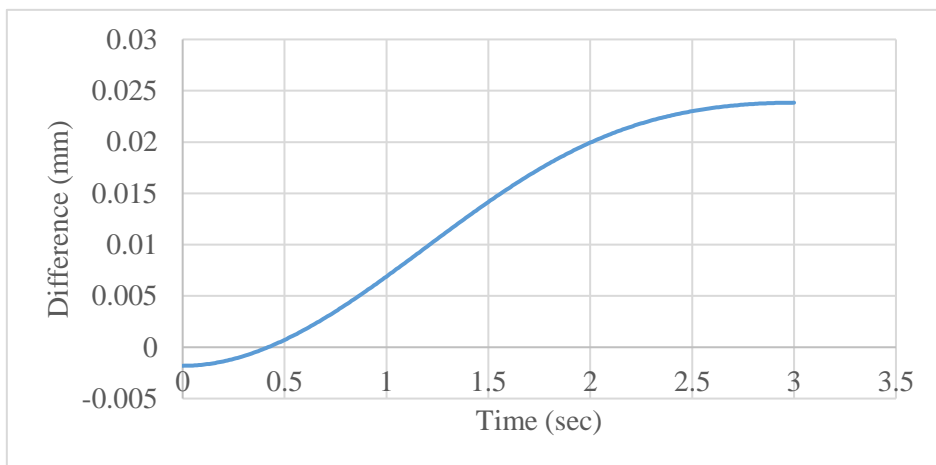
The presented project work is concentrated upon the kinematic analysis, Bluetooth control and trajectory planning of 5-DoF industrial robotic arm. It is concluded that D-H parameters provides a simple way for relating joint and link parameters. The 3D model of the robotic arm was made using CREO® parametric software. Further the designed components were manufactured by 3D printing. An application was also developed for controlling the robotic arm using Bluetooth. Forward and inverse kinematics were successfully tested by using the application. An object was picked and placed at the given coordinates which can be useful in assembly lines. Trajectory planning was done by joint space trajectory technique for cubic trajectory. Results of the parameters like position, velocity, acceleration, angular velocity, angular acceleration for joints and links are shown in the graphs in previous chapter. A GUI was made in MATLAB® for operating the robotic arm by connecting Arduino with MATLAB®, which gives ease to access joint angles and coordinates of end effector. It also provides simulation of the robotic arm performing on a given trajectory.

The results obtained from the analysis of robotic arm in MATLAB® are compared with that of CREO® and the plot of difference between parameters have been obtained. A total of 9 plots have been obtained in which the 3 plots are for the difference of position in X, Y, Z- axes, 3 plots are for the difference of velocity in X, Y, Z- axes and 3 plots are for the difference of acceleration in X, Y, Z- axes.

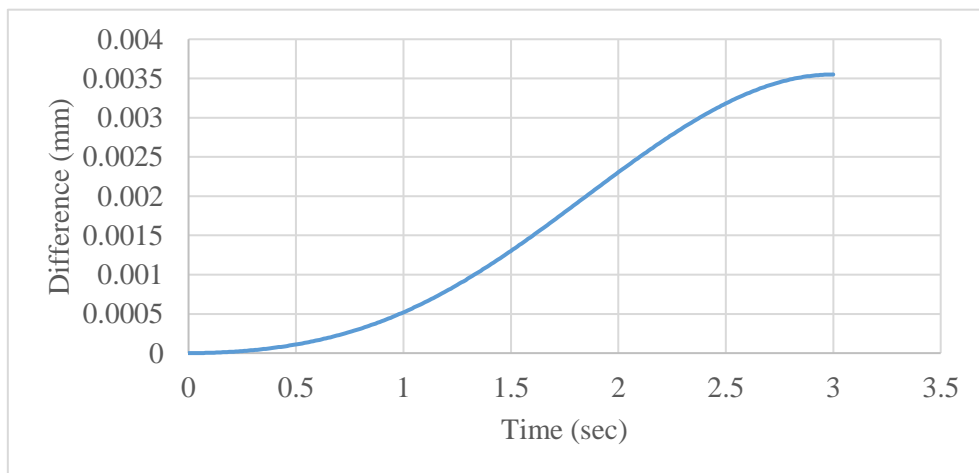
The error curve for position of robotic arm with respect to time in X, Y and Z- axes is represented in Figures 7.1 (a), 7.1(b) and 7.1(c) respectively. The error curve for velocity of robotic arm with respect to time in X, Y and Z- axes is represented in Figures 7.2(a), 7.2(b) and 7.2(c) respectively. The error curve for acceleration of robotic arm with respect to time in X, Y and Z- axes is represented in Figures 7.3(a), 7.3(b) and 7.3(c) respectively.



(a) Error in X-components

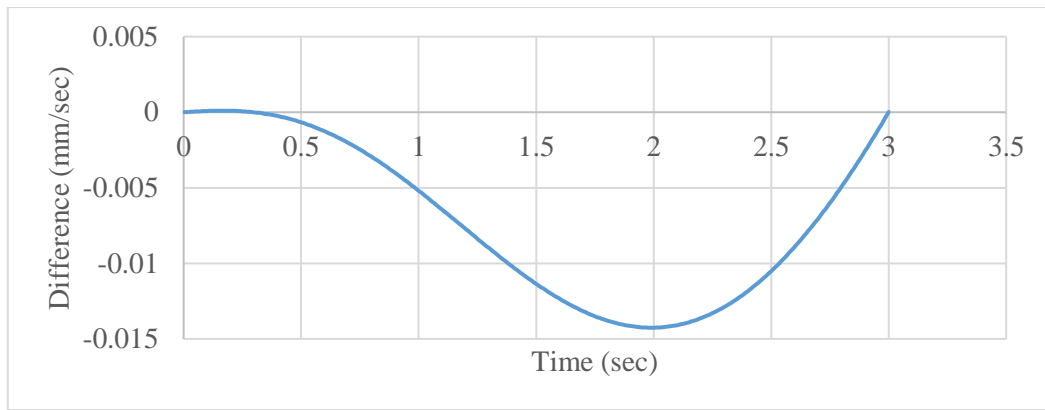


(b) Error in Y- components

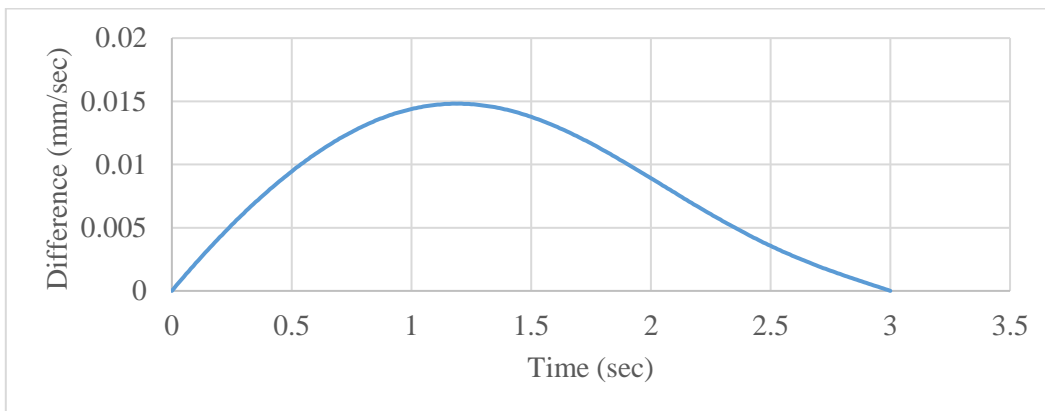


(c) Error in Z- components

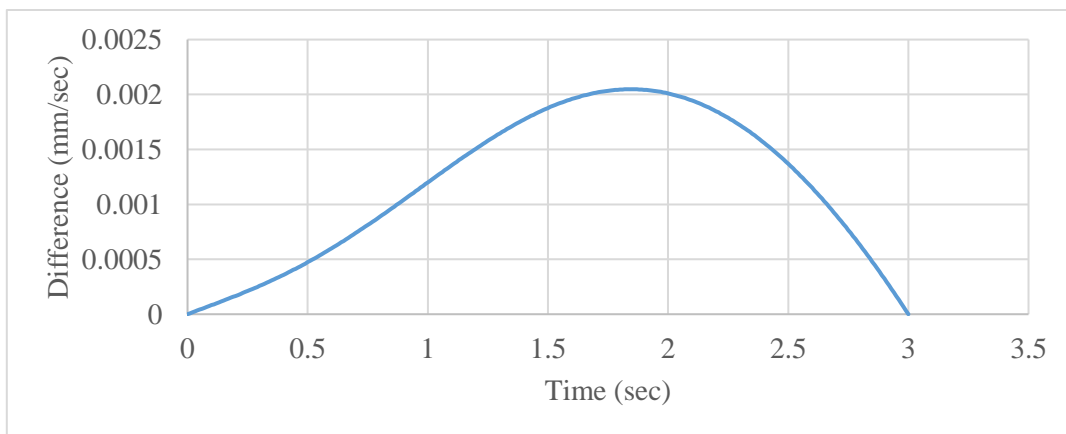
Figure 7.1: Position Versus Time



(a) Error in X- components

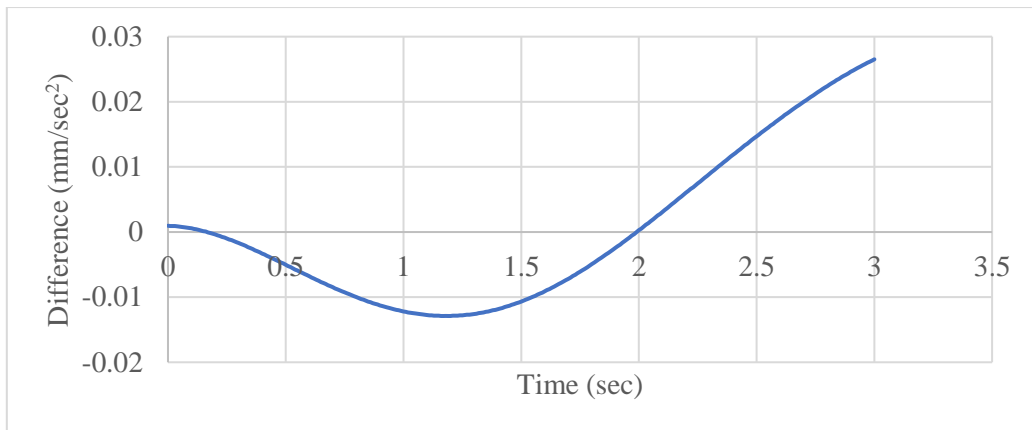


(b) Error in Y- components

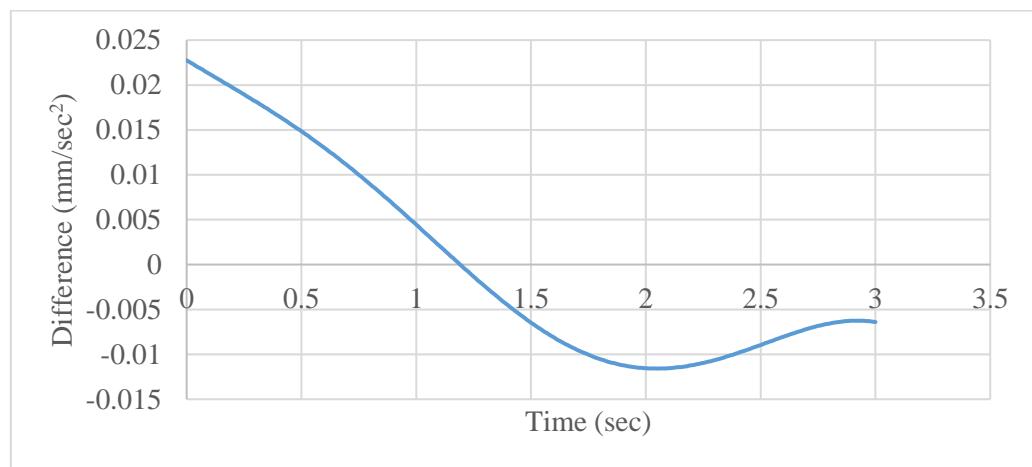


(c) Error in Z- components

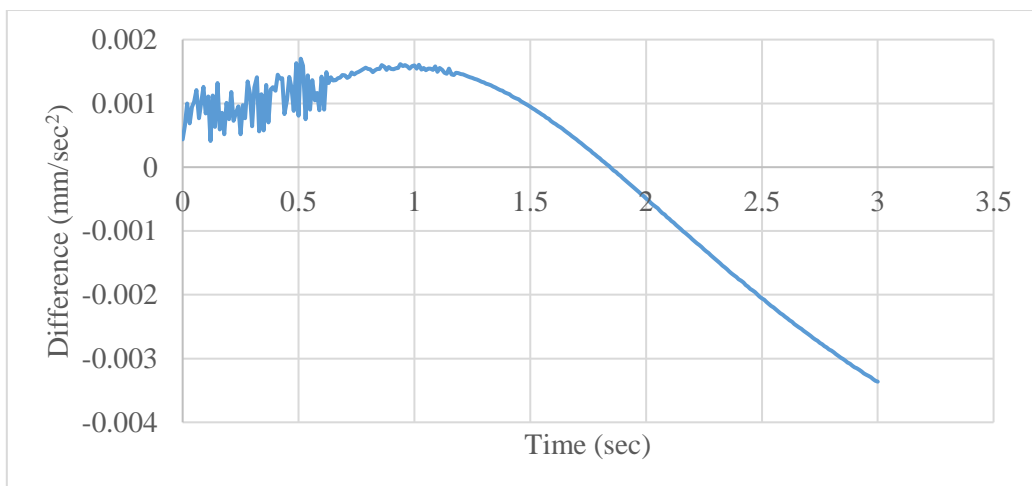
Figure 7.2: Velocity Versus Time



(a) Error in X- components



(b) Error in Y- components



(c) Error in Z- components

Figure 7.3: Acceleration Versus Time

7.2 Conclusion

It can be concluded that the robotic arm successfully performs according to forward and inverse Kinematics. The GUI developed for connecting Arduino with MATLAB® helps in successful operation of the robotic arm. It is also able to calculate and analyze time sequence of position, velocity and acceleration for both Joint Space and Cartesian Space trajectory techniques based on the given inputs in the GUI for trajectory planning. The maximum absolute percentage error for the position, velocity and acceleration in X, Y and Z direction is 0.018%, 0.015%, 0.21%, 0.014%, 0.015%, 0.003%, 0.016%, 0.017% and 0.005%. The graph of error in Z-component for acceleration takes some time to reach to a transient response due to error in estimating the real coordinates of Z-component from D-H parameters. It is also operated by controlling with Bluetooth module and can be used to teach and show robot to perform a specific task.

7.3 Future Work

- An arm having more than 5-DoF should be designed and developed for more applicable aspects.
- Dynamic analysis of the robotic arm should be carried out. Advanced algorithms and certain sensors can be used for sorting of objects based on color, shape and sizes.
- Collision avoidance can be implemented in the project.

SPECIAL ACKNOWLEDGEMENT

With deep gratitude, we would like to acknowledge the financial support given by the Alumni Association, BVM Engineering College. We would also like to show deep appreciation to supervisors who helped finalize our project.

8. REFERENCES

-
- [1] A. Oluwajobi and A. Oridate, "Design and Development of an Educational 5-DoF Robotic Arm", *International Journal of Robotics and Automation Technology*, 6, pp 55-65, 2019.
 - [2] A. Seemal and Philip Webb, "Kinematics Analysis of 6-DoF Articulated Robot with Spherical Wrist", *Mathematical Problems in Engineering*, vol. 2021, pp 1-11, 2021.
 - [3] A. Srikanth, Y.Ravithej, V.Sivaraviteja, V.Sreechand, "Kinematic analysis and simulation of 6 DOF of robot for industrial applications", *International Journal of Engineering and Science*, vol. 3, issue 8, pp 01-04, 2013.
 - [4] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, "Robotics Modelling, Planning and Control", Springer-Verlag London Limited, 2nd Edition.
 - [5] D. Constantin and Lupoae, Marin and Baci, Catalin and Ilie, Buliga, "Forward kinematic analysis of an industrial robot", in *Proceedings of the International Conference on Mechanical Engineering*, 2015.
 - [6] J. Iqbal, R. Islam and H. Khan "Modeling and Analysis of a 6 DoF Robotic Arm Manipulator", *Canadian Journal on Electrical and Electronics Engineering*, vol. 3, no. 6, 2012.
 - [7] K. S. Nair, and B. Prasanna Kumari, "Kinematic modelling and analysis of a 5-axis articulated robot arm model VRT-502", *International Journal of Engineering Research and Technology* vol. 4 issue 07, 2015.
 - [8] R. R. Serrezuela, A. F. C Chavarro , M. A. T Cardozo , A. L. Toquica and L. F. O Martinez "Kinematic modelling of a robotic arm manipulator using MATLAB[®]", *ARPJ Journal of Engineering and Applied Sciences*, vol. 12, no. 7, 2017.
 - [9] S. Ghuffar, I. Javaid, U. Mehmood, and M. Zubair, "Design and Fabrication of a Programmable 5-DoF Autonomous Robotic Arm", *Proceedings of the 6th WSEAS Int. Conf. on Systems Theory and Scientific Computation*, Elounda, Greece, August 21-23, pp 167-173, 2006.

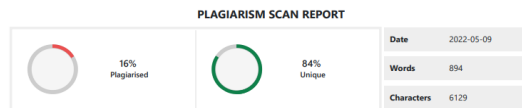
- [10] S. Manjaree, B. C. Nakra and V. Agarwal, "Comparative Analysis for Kinematics of 5-DoF Industrial Robotic Manipulator", vol. 9, no. 4, 2015.
- [11] S. Panich, "Mathematic Model and Kinematic Analysis for industrial robot", *Global Journal of Researches in Engineering: J General Engineering* vol. 15 issue 6 version 1.0, 2015
- [12] T. P. Singh, Dr. P. Suresh and Dr. S. Chandan, "Forward and inverse kinematic analysis of robotic manipulators", *International Research Journal of Engineering and Technology*, vol. 04. issue 02 ,2017
- [13] V. Deshpande and P. M. George, "Kinematic Modelling and Analysis of 5 DoF Robotic Arm", *International Journal of Robotics Research and Development*, vol. 4, issue 2, 2014, pp 17-24.
- [14] X. Zhao, M. Wang, N. Liu and Y. Tang, "Trajectory Planning for 6-DoF industrial robot Based on Quintic Polynomial", In *Proceedings of the 2017 2nd International Conference on Control, Automation and Artificial Intelligence*, 2017.
- [15] Y. D. Patel and P. M. George, "Performance Measurement and Dynamic Analysis of 2-DoF industrial robot Manipulator", *International Journal of Research in Engineering and Technology*, vol. 2, no. 9, pp 77-84, 2013.
- [16] Y. Jadeja and B. Pandya, "Design and Development of 5-DoF Robotic Arm Manipulators ", *International Journal of Scientific and Technology Research*, vol. 8, issue 11, 2019.
- [17] R. K. Mittal, I. J. Nagrath, "Robotics and Control", McGraw Hill, 2nd Edition.
- [18] CREO® Parametric 7.0.
- [19] MATLAB® Peter Corke ROS toolbox.
- [20] MIT App Inventor 2.

Plagiarism Report

Chapter 1



Chapter 2



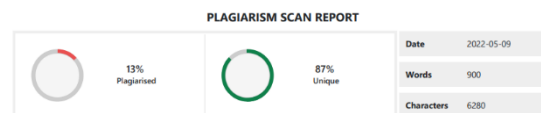
Chapter 3



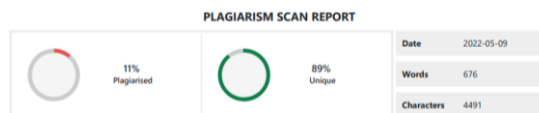
Chapter 4



Chapter 5



Chapter 6



Chapter 7

