ME-452 Robotics

Complex Engineering Problem

First Report

Members:

Taha Mahmood	**2017472**
Sheikh Abdul Majid	**2017437**

Submitted to: Dr Abid Imran

Abstract

In this paper, a 3 DOF robotic arm with RPR joint configuration will be analyzed and simulated on the MATLAB software. The forward kinematic model will predicate the Denavit-Hartenburg (DH) parametric for the robotic arm position. The inverse kinematic will also be performed by analyzing the value of the joint parameter based on the desired position and orientation of the end-effector. The MATLAB software is utilized to define the transformation matrix, simulating the robotic manipulator for different parameters. Moreover, the dynamic analysis of the robot will also be performed to verify its structural strength and reliability.

CONTENTS

List of Figures

List of Table

1 INTRODUCTION

A robotic manipulator is a mechanical arm, where each link engages with other links to create a motion. A joint is used to ensure a proper connection with one link to another. The links are then connected to form a kinematic chain. At the end of the kinematic chain, an end-effector consists of a tool performing a proper function.

1.1 HISTORY

After World War 2, the world shifted towards the automation of various industrial processes. As a result, many researchers came up with the idea of designing a mechanism that can perform monotonous tasks with better accuracy. Hence, scientists started researching different the concept of an automated robotic arm in the late 1960s. With the advancement in technology, the popularity of the robotic arm increased.

One of the primary reasons for the rise in popularity of robotic arm Is because of lower cost. The prices of the parts required to construct a manipulator have reduced significantly. Furthermore, the motors used to control the joints also become more efficient in due time since they could produce a faster motion with more efficiency. The robots also could perform multiple tasks and perform the task which was difficult for the average human being. [1]

Nowadays, robotic arms can be found in almost all industries ranging from manufacturing automation to warehouse automation. The tasks performed by a robotic arm can be performed with more accuracy and efficiency. This has resulted in increased productivity for the particular process and an increase in product quality. However, a significant amount of the human workforce has been replaced, resulting in the rise of unemployment.

1.2 DESIGN METHODOLOGY

In this paper, a 3 DOF RPR manipulator will be designed, operating in a planer space. First, to conduct the position analysis, a DH Table will be designed for the manipulator, and then forward kinematics will be conducted to derive the series of equations defining the position of the end-effector with respect to a coordinate axis. Later on, the inverse kinematics will be performed to define the relation of the joint parameter with the position of the end-effector. [2]

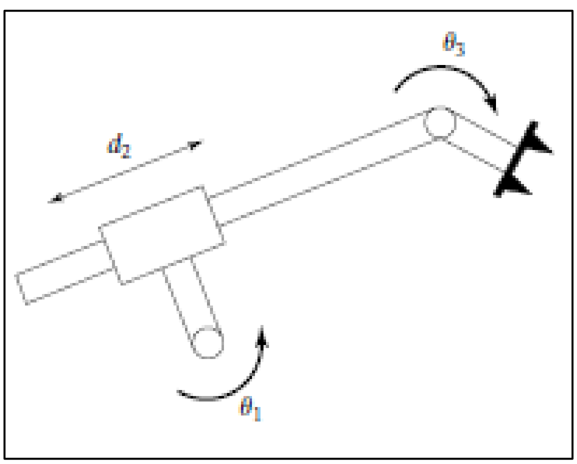

Figure 1: 3 DOF RPR manipulator

2 COMPLETE MATHEMATICAL MODEL

2.1 INTRODUCTION

In this paper, a 3 DOF RPR manipulator will be designed, operating in a planer space. First, to conduct the position analysis, a DH Table will be designed for the manipulator, and then forward kinematics will be conducted to derive the series of equations defining the position of the end-effector with respect to a coordinate axis. Later on, the inverse kinematics will be performed to define the relation of the joint parameter with the position of the end-effector.

First, the geometry of the robotic arm needs to be defined. To do so, we will construct the Denavit-Hartenberg (DH) table. With this, we will be able to define the joint parameters adapted in the manipulator. [3]

In mechanical engineering, the Denavit–Hartenberg parameters (also called DH parameters) are the parameters associated by attaching a reference frame to each link in the kinematic chain.

These parameters are:

a_i *is the distance from* Z_{i-1} *to* Z_i *measured along* X_i

α_i *is the angle from* Z_{i-1} *to* Z_i *measured about* X_i

d_i *is the distance from origin* X_{i-1} *to* X_i *measured along* Z_{i-1}

θ_i *is the distance from* X_{i-1} *to* X_i *measured about* Z_{i-1}

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	$\pi/2$	l_1	d_2	0
3	$-\pi/2$	0	0	θ_3
4	0	l_3	0	0

Table 1: Denavit–Hartenberg parameters

It is evident from the figure above that the manipulator has 3 distinct links, and 4 coordinate frames, with the last one attached at the end-effector. Furthermore, there are two different joints. There is a revolute joint between the link 0 to 1, and between link 3 to 4, while there is one prismatic joint between link 2 to 3. Link 0 is associated with the reference frame, which is the base. While link 4 is associated with the end-effector. After defining the links, they have been modelled into a single manipulator.

After developing the DH table, the next step was to construct transformation matrices between each link. The transformation matrix ${}^{i-1}_{i}T$ represent the transformation from the (i-1) coordinate frame to the (i) coordinate frame. The formula for ${}^{i-1}_{i}T$ is defined as:

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

The DH Table is then incorporated into the formula.

$$
{}^{0}_{1}T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}
$$

$$
{}^{1}_{2}T = \begin{bmatrix} 1 & 0 & 0 & l_1 \\ 0 & 0 & -1 & -d_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}
$$

$$
{}^{2}_{3}T = \begin{bmatrix} c_3 & -s_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_3 & -c_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}
$$

$$
{}^{3}_{4}T = \begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}
$$

After multiplying each transformation matrix. The final transformation matrix is derived by multiplying each reference frame transformation matrix with each other.

$$
{}^{0}_{4}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T
$$

$$
{}^{0}_{4}T = \begin{bmatrix} c_1c_3 - s_1s_3 & -c_1s_3 - c_3s_1 & 0 & l_3(c_1c_3 - s_1s_3) + l_1c_1 + d_2s_1 \\ c_1s_3 + c_3s_1 & c_1c_3 - s_1s_3 & 0 & l_3(c_1s_3 + c_3s_1) + l_1s_1 - d_2c_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

where

$c_2 = \cos(\theta_2)$, $s_2 = \sin(\theta_2)$, $s_{23} = \sin(\theta_2 + \theta_3)$, $c_{23} = \cos(\theta_2 + \theta_3)$

2.2 FORWARD KINEMATICS

Forward Kinematic refers to computing the position of the end-effector using each joint's value. With this, we develop analytical equations which will relate the position of the end-effector with the joint parameters. To determine this equation, we will be using the DH table to correlate the x and y position of the end-effector.

$$ {}^{0}_{n}T = \begin{bmatrix} . & . & . & p_x \\ . & . & . & p_y \\ . & . & . & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Here, p_x, p_y, p_z define the position of the end-effector in the x, y, and z-axis respectively.

$$p_x = l_3(c_1c_3 - s_1s_3) + l_1c_1 + d_2s_1$$

$$p_y = l_3(c_1s_3 + c_3s_1) + l_1s_1 - d_2c_1$$

$$p_z = 0$$

These are the equations that define the position of the end-effector with respect to the joint's parameters. Here, d_2 is the position of the prismatic joint, while θ_1 and θ_3 are the values of the two revolute joints. Now, to find the translational vector of the transformation matrix, the joint parameters have been defined in terms of time. The equations of joint variables have been summarized below:

θ_1	d_2	θ_3
$(\pi/3)t$	$3t$	$(\pi/3)t$

Table 2: Joint Variable Equations

According to the equations shown in the table, the values of all the joint parameters are zero at the home position, as shown in Fig

2.3 INVERSE KINEMATICS

Inverse kinematics refers to finding the values of the joint parameters using the end-effector's position. It is the opposite of the forward kinematics. To calculate the joint parameters, we need to define a series of analytical equations.

Since the robotic manipulator is a planar robot, the manipulator's position will be defined in the x and y-axis. However, three joint parameters are to be evaluated. To account for the redundancies, another variable, phi φ, will be introduced. The equation that defines phi will be as follows: -

$$\varphi = \theta_1 + \theta_3$$

Initially, in the forward kinematics, the position of the end-effector with respect to the x and y-axis was defined as follows: -

$$l_3 \cos(\theta_1 + \theta_3) + l_1 \cos\theta_1 + d_2 \sin\theta_1 = x$$

$$l_3 \sin(\theta_1 + \theta_3) + l_1 \sin\theta_1 + d_2 \cos\theta_1 = y$$

Performing inverse kinematics for each joint

Joint 2: d_2

$$l_3 \cos(\theta_1 + \theta_3) + l_1 \cos\theta_1 + d_2 \sin\theta_1 = x$$

$$l_3 \cos(\varphi) + l_1 \cos\theta_1 + d_2 \sin\theta_1 = x$$

$$l_1 \cos\theta_1 + d_2 \sin\theta_1 = x - l_3 \cos(\varphi) = x'$$

$$l_3 \sin(\theta_1 + \theta_3) + l_1 \sin\theta_1 + d_2 \cos\theta_1 = y$$

$$l_3 \sin(\varphi) + l_1 \sin\theta_1 + d_2 \cos\theta_1 = y$$

$$l_1 \sin\theta_1 + d_2 \cos\theta_1 = y - l_3 \sin(\varphi) = y'$$

Squaring x' and y' and summing them together

$$x'^2 + y'^2 = (l_1 \cos\theta_1 + d_2 \sin\theta_1)^2 + (l_1 \sin\theta_1 + d_2 \cos\theta_1)^2$$

$$x'^2 + y'^2 = l_1^2 + d_2^2$$

$$d_2 = \sqrt{x'^2 + y'^2 - l_1^2}$$

Joint 1: θ_1

$$l_1 \cos\theta_1 + d_2 \sin\theta_1 = x - l_3 \cos(\varphi) = x'$$

$$\theta_1 = 2\tan^{-1}\left(\frac{d_2 \pm \sqrt{d_2^2 - l_1^2 - x'^2}}{l_1 + x'}\right)$$

Joint 3: θ_3

$$\theta_3 = \varphi - \theta_1$$

3 RESULTS

3.1 FORWARD KINEMATICS

This is the MATLAB code that has been compiled to perform the forward kinematics of our manipulator

```
clc
clear all
format short
L1=0.8;L3=0.8;
q1=0;d2=2;q3=0;

q=q1;d=0;a=0;alpha=0; %for i=1
T01=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;d=d2;a=L1;alpha=pi/2; %for i=2
T12=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=q3;d=0;a=0;alpha=-pi/2; %for i=3
T23=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;a=L3;d=0;alpha=0; %for i=4
T34=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];

T02=T01*T12;
T03=T01*T12*T23;
T04=T01*T12*T23*T34

axis([-5 5 -5 5]);

Ax1 = [T01(1,4),T02(1,4)];
Ay1 = [T01(2,4),T02(2,4)];
Ax2 = [T02(1,4),T03(1,4)];
Ay2 = [T02(2,4),T03(2,4)];
Ax3 = [T03(1,4),T04(1,4)];
Ay3 = [T03(2,4),T04(2,4)];
Ax4 = [-.1,.1];
Ay4 =  [0,0];
p1  = line(Ax1,Ay1,'LineWidth',[3],'Color','k');
p2  = line(Ax2,Ay2,'LineWidth',[3],'Color','M');
p3  = line(Ax3,Ay3,'LineWidth',[3],'Color','R');
p4  = line(Ax4,Ay4,'LineWidth',[12],'Color','B');

drawnow
xlabel('x (m)')
ylabel('y (m)')

pause()
```

```
for t=0:.01:1
    q1=pi/3*t;
    d2=2+3*t;
    q3=pi/3*t;

    q=q1;d=0;a=0;alpha=0; %for i=1
    T01=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha)
-sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
    q=0;d=d2;a=L1;alpha=pi/2; %for i=2
    T12=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha)
-sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
    q=q3;d=0;a=0;alpha=- pi/2; %for i=3
    T23=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha)
-sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
    q=0;a=L3;d=0;alpha=0; %for i=4
    T34=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha)
-sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];

    T02=T01*T12;
    T03=T01*T12*T23;
    T04=T01*T12*T23*T34

    Ax1 = [T01(1,4),T02(1,4)];
    Ay1 = [T01(2,4),T02(2,4)];
    Ax2 = [T02(1,4),T03(1,4)];
    Ay2 = [T02(2,4),T03(2,4)];
    Ax3 = [T03(1,4),T04(1,4)];
    Ay3 = [T03(2,4),T04(2,4)];
    Ax4 = [-.1,.1];
    Ay4 =  [0,0];

    set(p1,'X', Ax1, 'Y',Ay1)
    set(p2,'X', Ax2, 'Y',Ay2)
    set(p3,'X', Ax3, 'Y',Ay3)
    set(p4,'X', Ax4, 'Y',Ay4)
    set(p4,'X', Ax4, 'Y',Ay4)

    drawnow

    pause(.01)
end
```

Initially, the joint parameters were defined as

$$q_1 = 0\,rad$$

$$d_2 = 2m$$

$$q_3 = 0\,rad$$

After executing the code, the following transformation matrix was found

$$ {}^{0}_{4}T = \begin{bmatrix} 1 & 0 & 0 & 1.6 \\ 0 & 1 & 0 & -2.0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Here the position of the end-effector has been identified, where

$$ x = 1.6\ m $$

$$ y = -2.0\ m $$

And this is the plot that has been obtained

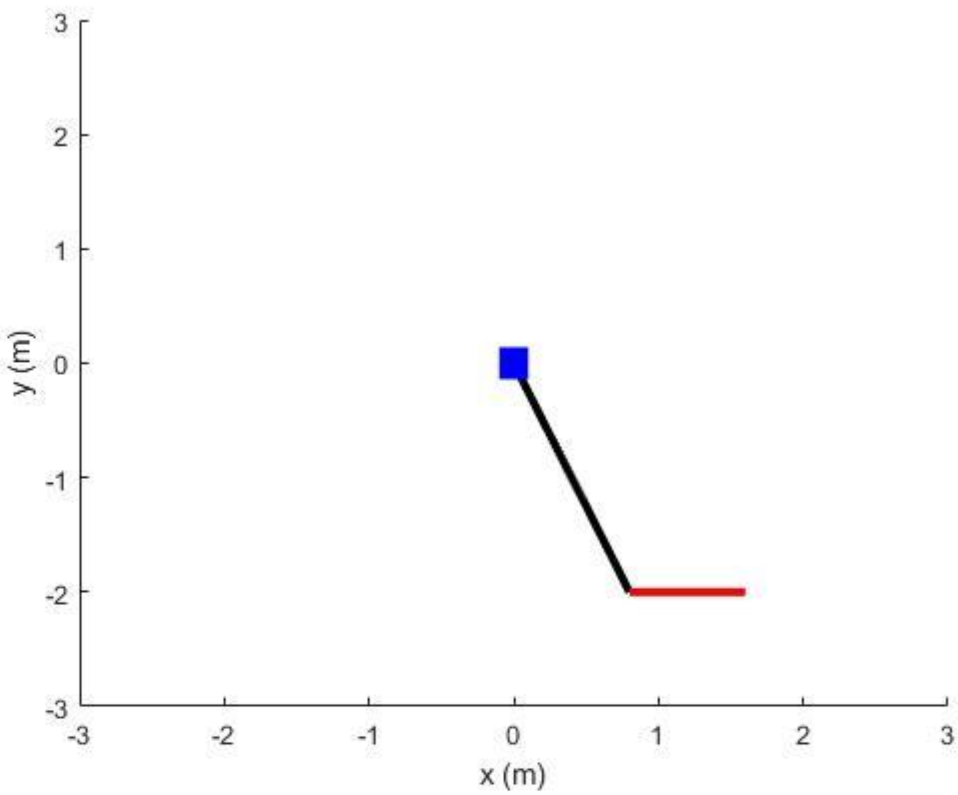

Figure 2: Forward Kinematic plot

Later the joint parameters were defined as a function of time

$$ q_1 = \frac{\pi}{3} t $$

$$ d_2 = 3t $$

$$ q_3 = \frac{\pi}{3} t $$

Where,

$$t \rightarrow 0 \; to \; 1 \; sec$$

This was the simulation result that was derived after executing the code.

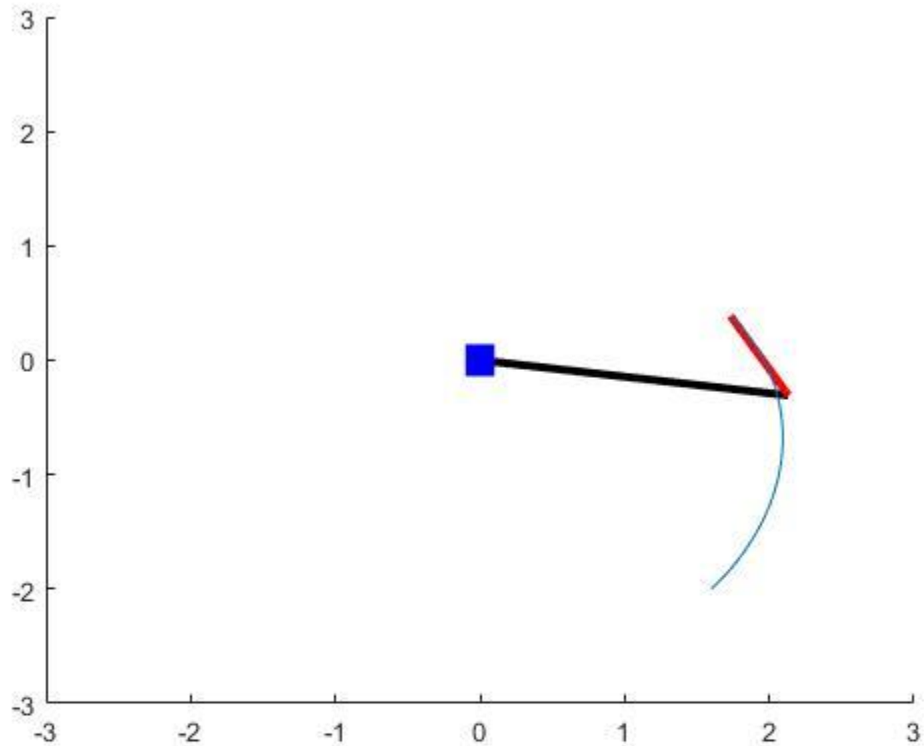

Figure 3: Forward Kinematics Simulation plot

3.2 Inverse Kinematics

After analyzing the inverse kinematics mathematical model, the following MATLAB code was executed.

```
clc;clear all;
format short

L1=0.8;L3=0.8;


x=1.5;
y=2.0;
PHIH=pi/2;
xi = x-L3*cos(PHIH);
yi = y-L3*sin(PHIH);
d2=sqrt(xi^2+yi^2-L1^2);
q1=2*atan2((d2+sqrt((d2^2)-(L1^2)-(xi^2))),(L1+xi));
q3=PHIH-q1;

q=q1;d=0;a=0;alpha=0; %for i=1
```

```
T01=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;d=d2;a=L1;alpha=pi/2; %for i=2
T12=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=q3;d=0;a=0;alpha=-pi/2; %for i=3
T23=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;a=L3;d=0;alpha=0; %for i=4
T34=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];

T02=T01*T12;
T03=T01*T12*T23;
T04=T01*T12*T23*T34
axis([-2 3 -2 3]);

  Ax1 = [T01(1,4),T02(1,4)];
  Ay1 = [T01(2,4),T02(2,4)];
  Ax2 = [T02(1,4),T03(1,4)];
  Ay2 = [T02(2,4),T03(2,4)];
  Ax3 = [T03(1,4),T04(1,4)];
  Ay3 = [T03(2,4),T04(2,4)];
  Ax4 = [-.1,.1];
  Ay4=  [0,0];
  p1  = line(Ax1,Ay1,'LineWidth',[3],'Color','k');
  p2  = line(Ax2,Ay2,'LineWidth',[3],'Color','M');
  p3  = line(Ax3,Ay3,'LineWidth',[3],'Color','R');
  p4  = line(Ax4,Ay4,'LineWidth',[12],'Color','B');

  drawnow

n=1;
r=.3;
  for t=0:.01:1

phi=t*2*pi;
x=1.5+r*cos(phi);
y=2.5+r*sin(phi);

PHIH=pi/2;
xi = x-L3*cos(PHIH);
yi = y-L3*sin(PHIH);
d2=sqrt(xi^2+yi^2-L1^2);
q1=2*atan2((d2+sqrt((d2^2)-(L1^2)-(xi^2))),(L1+xi));
q3=PHIH-q1;

Xp = L3*(cos(q1+q3))+L1*cos(q1)+d2*sin(q1);
Yp = L3*(sin(q1+q3))+L1*sin(q1)-d2*cos(q1);
q=q1;d=0;a=0;alpha=0; %for i=1
```

```
T01=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;d=d2;a=L1;alpha=pi/2; %for i=2
T12=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=q3;d=0;a=0;alpha=-pi/2; %for i=3
T23=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
q=0;a=L3;d=0;alpha=0; %for i=4
T34=[cos(q) -sin(q) 0 a;sin(q)*cos(alpha) cos(q)*cos(alpha) -sin(alpha) -
sin(alpha)*d;sin(q)*sin(alpha) cos(q)*sin(alpha) cos(alpha) cos(alpha)*d;0 0
0 1];
T02=T01*T12;
T03=T01*T12*T23;
T04=T01*T12*T23*T34;

  Ax1 = [T01(1,4),T02(1,4)];
  Ay1 = [T01(2,4),T02(2,4)];
  Ax2 = [T02(1,4),T03(1,4)];
  Ay2 = [T02(2,4),T03(2,4)];
  Ax3 = [T03(1,4),T04(1,4)];
  Ay3 = [T03(2,4),T04(2,4)];
  Ax4 = [-.1,.1];
  Ay4=  [0,0];

  set(p1,'X', Ax1, 'Y',Ay1)
  set(p2,'X', Ax2, 'Y',Ay2)
  set(p3,'X', Ax3, 'Y',Ay3)
  set(p4,'X', Ax4, 'Y',Ay4)
  o1(n,1)=Xp;
  o2(n,1)=Yp;


  drawnow
  pause(.01)
  n=n+1;
  end
hold on
plot(o1(:,1),o2(:,1))
```

Initially, the end-effector position was defined as

$$x = 1.5\ m$$

$$y = 2.0\ m$$

After executing the following code, the joints parameters were evaluated as follows: -

θ_1	d_2	θ_3
1.5017	1.7464	0.0691

Table 3: Joint Parameters Values

Similarly, the transformation matrix was also evaluated as following:-

$$ {}^{0}_{4}T = \begin{bmatrix} 0 & -1 & 0 & 1.7975 \\ 1 & 0 & 0 & 1.4776 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The plot of the entire manipulator is shown in the figure below:-

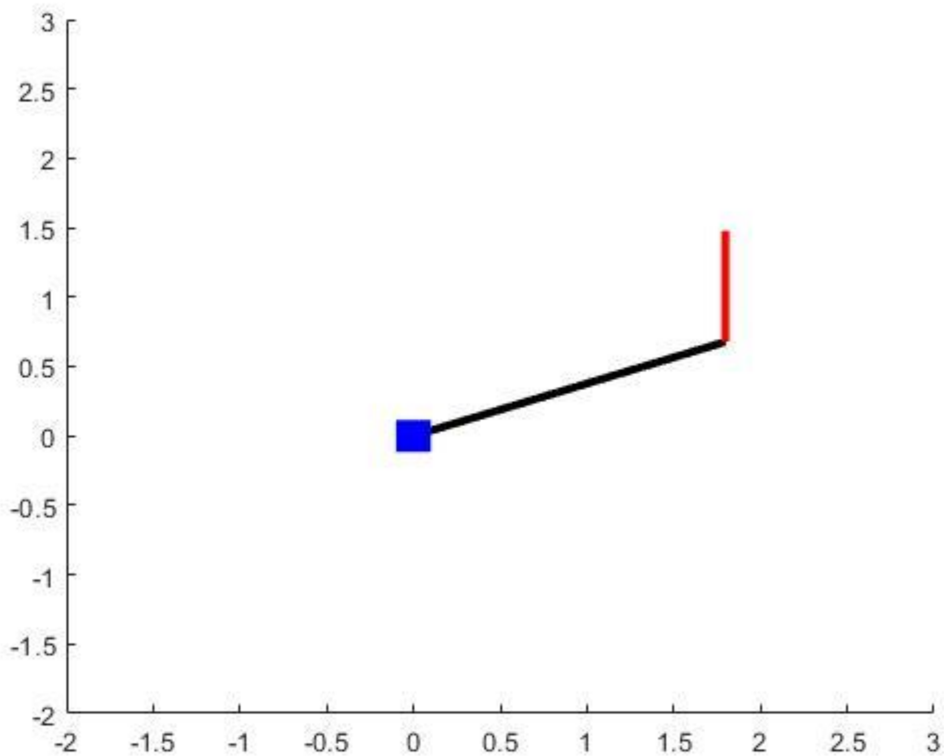

Figure 4: Inverse Kinematics Plot

Later, the trajectory of the end-effector position was defined as follows:-

$$\varphi = 2\pi t$$

$$x = 1.5 + r\cos\varphi$$

$$y = 2.5 + r\sin\varphi$$

Where,

$$r = 0.3\ m$$

$$t \rightarrow 0\ to\ 1\ sec$$

This was the simulation result that was derived after executing the code.

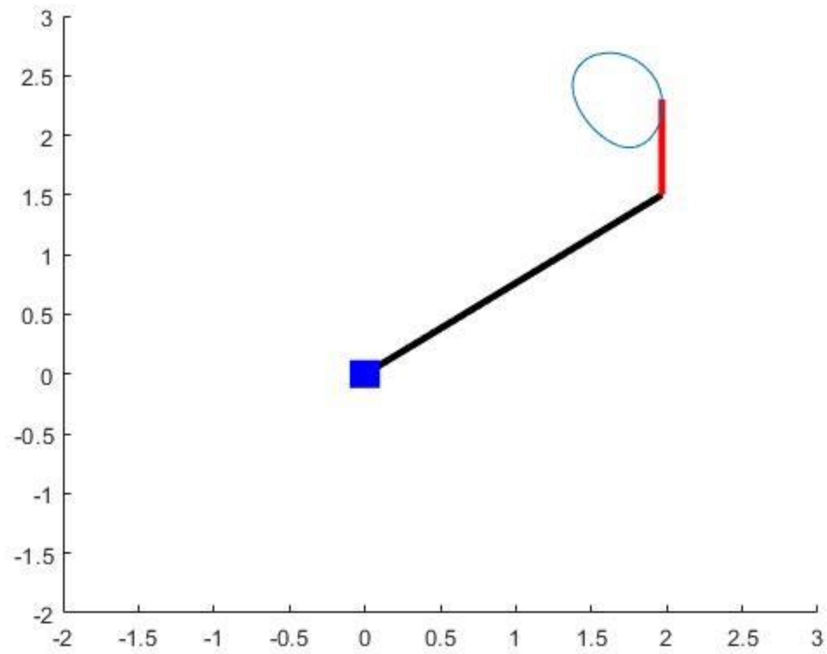

Figure 5: Inverse Kinematics Simulation plot

4 CONCLUSION

In this report, an RPR manipulator has been designed based on the requirements of dexterous and reachable workspace. First, the coordinate frames were attached to each of the joints of the links and to the end-effector. Based on the coordinate frames, the DH table was constructed which was further analyzed to build the transformation matrix, defining the task space coordinates in the inertial frame. The transformation matrices were then used to construct a forward kinematics code on MATLAB, where the joint variables were defined in terms of time, and using those joint parameters, the task space coordinates were calculated. Moreover, the home position and final position of the manipulator were also determined through the joint parameter equations.

After performing the forward kinematics, the manipulator was analyzed through inverse kinematics approach. In this case, first the joint parameters were defined in terms of the cartesian coordinates determined from the final transformation matrix. Then, the cartesian coordinates were defined as functions of φ and radius of the circle. Finally, the joint parameters' values were determined, and the desired trajectory was drawn on MATLAB. After performing a detailed kinematic analysis, the values of the parameters will be further utilized in performing the dynamic analysis, and in determining the closed form solution of the manipulator.

5 REFERENCES

1. https://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html
2. *"Robot Arm and Computer Vision". Retrieved 29 July 2016*
3. *Craig, John J. Introduction to robotics mechanics and control*