FACULTY OF ENGINEERING AND ARCHITECTURE
MECHATRONICS ENGINEERING

GRADUATION/DESIGN PROJECT

Analysis and Simulations of Dynamics of Various Serial Robot Manipulators by Lagrange method

BURAK BERKE CANAKLI
160412023
TLEULENOVA AYANA
160412059
CIHAN NAZLI
150412007

SUPERVISOR: DUYGU ATCI

JUNE 2021

ABSTRACT

Manipulators are the mechanical basis of most robots with any control system: with a rigid program of actions, controlled by a human operator or with artificial intelligence, acting purposefully without human intervention. Due to their simplicity and reliability, manipulators with two rotary and one translational kinematic pairs are widely are used, along with traditional automation tools, in small-scale production and in industries with harmful working conditions, when performing auxiliary operations of the technological cycle, such as transportation, installation or removal of parts. In particular, in mining, such manipulators are the main mechanism of the drilling carriage, they are used to lift support elements, etc.

Keywords: Dynamics, Lagrange, Robotics, 3-DOF robot arm

ÖZET

Manipülatörler, herhangi bir kontrol sistemine sahip çoğu robotun mekanik temelidir: bir insan operatör tarafından veya yapay zeka ile kontrol edilen katı bir eylem programı ile, insan müdahalesi olmadan amaçlı hareket eden. Basitlikleri ve güvenilirlikleri nedeniyle, iki döner ve bir öteleme kinematik çifti olan manipülatörler, geleneksel otomasyon araçlarıyla birlikte, küçük ölçekli üretimde ve zararlı çalışma koşullarına sahip endüstrilerde, teknolojik döngünün yardımcı işlemlerini gerçekleştirirken yaygın olarak kullanılmaktadır. parçaların taşınması, takılması veya çıkarılması gibi. Özellikle madencilikte, bu tür manipülatörler sondaj arabasının ana mekanizmasıdır, destek elemanlarını vb. kaldırmak için kullanılırlar.

Anahtar Kelimeler: Dinamik, Lagrange, Robotik, 3-DOF robot kolu

TABLE OF CONTENTS

ABSTRACT....................i
ÖZET....................ii
LIST OF TABLES.................... **Error! Bookmark not defined.**
LIST OF FIGURES....................iv
SYMBOLS....................v
ACRONYMS.................... **Error! Bookmark not defined.**
I. INTRODUCTION....................1
1.1. Serial Manipulators.................... **Error! Bookmark not defined.**
1.2. Previous Reserches.................... **Error! Bookmark not defined.**
II. CHAPTER (DYNAMIC ANALYSIS)....................5
2.1. References to Equations....................16
2.2. Workspace Analysis with MATLAB.................... **Error! Bookmark not defined.**
III. CHAPTER(MODELLING CONCEPT).................... **Error! Bookmark not defined.**
3.1. Modelling of the 2-DOF robotic arm....................16
3.2. Modelling of the 3-DOF robotic arm....................16
IV. SUMMARY AND CONCLUSION....................31
3.1. Graphical Analysis of 2-DOF Robotic Arm.................... **Error! Bookmark not defined.**
3.2. Graphical Analysis of 3-DOF Robotic Arm.................... **Error! Bookmark not defined.**
REFERENCES....................40

LIST OF FIGURES

Figure 1 *Serial Robot Manipulator*.. **Error! Bookmark not defined.**
Figure 1.2. *Rirst Industrial Robotic Arm* .... ***Error! Bookmark not defined.***
Figure 2.1.2. *Inverse Kinematics Representation for 2-DOF* .. ***Error! Bookmark not defined.***
Figure 3.1.1. *Design of Link 1 in SolidWorks*.......................... **Error! Bookmark not defined.**
Figure 3.1.2. *Design of Link 2 in SolidWorks*.......................... **Error! Bookmark not defined.**
Figure 3.1.3. *Design of Base in SolidWorks* .... **Error! Bookmark not defined.**
Figure 3.1.4. *Improved Robotic Design of 2-DOF Robotic Arm*............**Error! Bookmark not defined.**
Table 3.1. *Properties of the 2-DOF robotic arm* .... **Error! Bookmark not defined.**
Figure 3.2.1. *Improved Robotic Design of 3-DOF Robotic Arm*............**Error! Bookmark not defined.**
Figure 4.1.1. *Torque graphic of Joint 1* from SIMULINK................ **Error! Bookmark not defined.**
Figure 4.1.2 . *Torque graphic of Joint 1* from MATLAB................. **Error! Bookmark not defined.**
Figure 4.1.3. *Torque graphic of Joint 2* from SIMULINK................ **Error! Bookmark not defined.**
Figure 4.1.4. *Torque graphic of Joint 1* from MATLAB.................. **Error! Bookmark not defined.**
Figure 4.1.5. *Block Diagram of 2-DOF arm* .... **Error! Bookmark not defined.**
Figure 4.2.1. *Torque graphic of Joint 1 from MATLAB* .... **Error! Bookmark not defined.**
Figure 4.2.2. *Torque graphic of Joint 1 from SIMULINK*................ **Error! Bookmark not defined.**
Figure 4.2.3. *Torque graphic of Joint 1 from MATLAB* .... **Error! Bookmark not defined.**
Figure 4.2.4. *Torque graphic of Joint 2 from SIMULINK*................ **Error! Bookmark not defined.**
Figure 4.2.5. *Torque graphic of Joint 2 from MATLAB*.................. **Error! Bookmark not defined.**
Figure 4.2.6. *Torque graphic of Joint 3 from SIMULINK*................ **Error! Bookmark not defined.**

SYMBOLS

G: vector of gravitational forces

I_i: inertia matrix of link I about its center of mass and expressed in the base link frame

J_i: link i Jacobian matrix

J_{vi}: Jacobian submatrix associated with the linear velocity of the center of mass of link i

$J_{\omega i}$: Jacobian submatrix associated with the angular velocity vector of link i

K: kinetic energy of a mechanical system

L: Lagrange function, L=K-U

M: manipulator inertia matrix

M_{ij}: (I,j) element of M

n: number of generalized coordinates

${}^{k}p_{ci}^{*}$: position vector of the center of mass of the ith link with respect to the kth link frame and expressed in the fixed base frame

Q_i: generalized active force corresponding to the ith generalized coordinate

Q: vector of generalized forces, $Q = |Q_1, Q_2 \ldots\ldots Q_n|^T$

q_i: ith generalized coordinate

q: vector of generalized coordinates, $|q_1, q_2 \ldots\ldots q_n{}^T|$

U: potential energy of a mechanical system

V: Velocity coupling vector

δW: virtual work

I. INTRODUCTION

1.1. Serial Manipulators.

Over the last few decades, use of industrial robots can be seen workdwide and has significantly increased with a faster increasingly trend. Mostly these are being used for material handling. welding, painting, asembling of parts, packaging. handling hazardous materials, undersea operations, cte. Robot manipulator implicates an electromechanical device that requires human dexterity to perform a variety of tasks Although few manipulaton are anthropomorphie and humanoid, most of these robots can be treatcd as clectromechanical devices from their structure point of view. On the other hand, there are autonomous and sстiautonomou robots that have a broad range of applications such as planetary space cxploration, surgical robotic, rehabilitation, and household applications.

The modern industrial robotic arm is applicable to replace human labor. So, the robot can use the tool gripper to hold the tool and process the workpiece, or hold the workpiece itself to feed it into the working area for the further processing. This type of robot is in the greatest demand. Any industrial robotic arm is a multipurpose mechanism, often with several degrees of freedom (axes of motion). The most common are remote-controlled "mechanical arms" that are mounted on a fixed or movable base. However, the specificity of the various applications of industrial robots forces manufacturers to develop specialized robots for specific tasks.

Fig. 1 *Serial Robot Manipulator*

Robot manipulators move along pre-specified trajectories which are sequence of points were end effector position, and orientations are known. Trajectories may be joint space or Cartesian spaces that are a function of time. The industrial robots can be explicitly considered as open chain mechanisms that are systems of rigid bodies connected by various joints. Joints allow particular types of relative motions between the connected bodies. For example, a rotational joint acts as a hinge and allows only a relative rotation between the connected bodies about the axis of the joint. A system of rigid bodies interconnected by joints is called a kinematic chain.

The operating mechanism of the manipulator is an open kinematic chain, the links of which are connected in series with each other by joints of various types (rotational or translational). The combination and relative position of the links and joints determines the number of degrees of mobility, as well as the scope of the robot's grip. It is often assumed that the first three joints in the operating mechanism of the manipulator provide transport degrees of mobility (ensuring the movement of the working body to the required position), and the rest of the joints realize the orienting degrees of mobility (orienting the working body according to the task)

1.2. Previous Researches

The concept of the robot Was evidently recognized by the Czech playwright Karel Capek during the twentieth century in his play "Rossum's Universal Robots (R.U.R.)". The term "robot" is derived from "robota" which means subordinate labour in Slave languages. In 1940, the ethics of the interaction berween robots and humans was envisioned to be governed by the well-known three fundamental laws of Isaac Asimov, the Russian science-fiction writer in his novel "Run-around".

BY 1961, THE UNIMATE 1900 SERIES BECAME THE FIRST MASS PRODUCED ROBOTIC ARM FOR FACTORY AUTOMATION.

Mindful of the uphill battle he would face from manufacturers, and motivated by Asimov’s Three Laws of Robotics that relate a “first do no harm” philosophy similar to the Hippocratic Oath, Engelberger focused on employing the robots in tasks harmful to humans. His strategy worked and in 1959 the 2,700 pound Unimate #001 prototype was installed on an assembly line for the first time at a General Motors die-casting plant in Trenton, New Jersey.

Fig. 1.2. *First Industrial Robotic Arm*

Currently, the most common control systems for manipulation robots are manufactured by ABB, KUKA, Yaskawa Motoman, Fanuc. In their developments, to solve these problems, they use closed proprietary solutions. That is, the user receives a system that includes a manipulation robot and a control system from one manufacturer. This approach allows the manufacturer to guarantee the performance of the final solution, but limits the possibilities on the part of the user.

II. CHAPTER (Forward and Inverse Kinematics)

1. Forward Kinematics.

Table-1 DH-Parameters

Link	α	a	d	θ
1	0	L1	0	Θ1
2	0	L2	0	Θ2

$$ {}_1^0T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & l_1c\theta_1 \\ s\theta_1 & c\theta_1 & 0 & l_1s\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}_2^1T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & l_1c\theta_1 \\ s\theta_1 & c\theta_1 & 0 & l_1s\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}_2^0T = \begin{bmatrix} c\theta_{12} & -s\theta_{12} & 0 & l_1c\theta_1 + l_2c\theta_{12} \\ s\theta_{12} & c\theta_{12} & 0 & l_1s\theta_1 + ls\theta_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ \theta_1 = Joint\ rates\ of\ first\ actuator $$

$$ \theta_2 = Joint\ rates\ of\ second\ actuator $$

$$ c\theta_{12} = co\,\mathrm{s}(\theta_1 + \theta_2) $$

$$ s\theta_{12} = si\,\mathrm{n}(\theta_1 + \theta_2) $$

$$ L1 = link\ 1\ length $$

$$ L1 = link\ 2\ length $$

2. Workspace Analysis with Matlab for Forward Kinematics

So, our next step of the project is to use MATLAB to do large calculations. It can be noticed that calculations will be more complicated with each degree of freedom, hence it`s easier to write a code. Due to the equations from the book we need the first and second derivative of Theta, that`s why we need to give definition function depending on time. So first we start with the code for 2-DOF robotic arm. Here we provide our code with the description:

```
syms L1 L2  Q1 Q2 D1 D2 a1 a2 Q1 Q2
alphaa=[0,0]; % this is the alpha value for all  the link
a=[a1,a2]; % Length of the Link
d=[0,0]; %Offset
Q=[Q1,Q2]; % joint angle variation
%%Transformation Matrices
for i=1:2
    switch i
        case 1
            T01= [cos(Q(1,i)),-
sin(Q(1,i))*cosd(alphaa(1,i)),sind(alphaa(1,i))*sin(Q(1,i)),a(
1,i)*cos(Q(1,i));sin(Q(1,i)),cos(Q(1,i)).*cosd(alphaa(1,i)),-
sind(alphaa(1,i))*cos(Q(1,i)),sin(Q(1,i))*a(1,i);0,sind(alphaa
(1,i)),cosd(alphaa(1,i)),d(1,i);0,0,0,1];
        case 2
            T12= [cos(Q(1,i)),-
sin(Q(1,i))*cosd(alphaa(1,i)),sind(alphaa(1,i))*sin(Q(1,i)),a(
1,i)*cos(Q(1,i));sin(Q(1,i)),cos(Q(1,i)).*cosd(alphaa(1,i)),-
sind(alphaa(1,i))*cos(Q(1,i)),sin(Q(1,i))*a(1,i);0,sind(alphaa
(1,i)),cosd(alphaa(1,i)),d(1,i);0,0,0,1];
    end
end
T01 % first link with respect to base
T12 % second link with respect first link
T02= T01*T12
```

Matlab Part Answer

T01 =

```
[cos(Q1), -sin(Q1), 0, a1*cos(Q1)]

[sin(Q1),  cos(Q1), 0, a1*sin(Q1)]

[      0,        0, 1,          0]

[      0,        0, 0,          1]
```

T12 =

```
[cos(Q2), -sin(Q2), 0, a2*cos(Q2)]

[sin(Q2),  cos(Q2), 0, a2*sin(Q2)]

[      0,        0, 1,          0]

[      0,        0, 0,          1]
```

T02 =

```
[cos(Q1)*cos(Q2) - sin(Q1)*sin(Q2), - cos(Q1)*sin(Q2) -
cos(Q2)*sin(Q1), 0, a1*cos(Q1) + a2*cos(Q1)*cos(Q2) -
a2*sin(Q1)*sin(Q2)]

[cos(Q1)*sin(Q2) + cos(Q2)*sin(Q1),   cos(Q1)*cos(Q2) -
sin(Q1)*sin(Q2), 0, a1*sin(Q1) + a2*cos(Q1)*sin(Q2) +
a2*cos(Q2)*sin(Q1)]

[                                0,
0, 1,                                                    0]

[                                0,
0, 0,                                                    1]
```

3. Workspace Analysis with Matlab for Inverse Kinematics

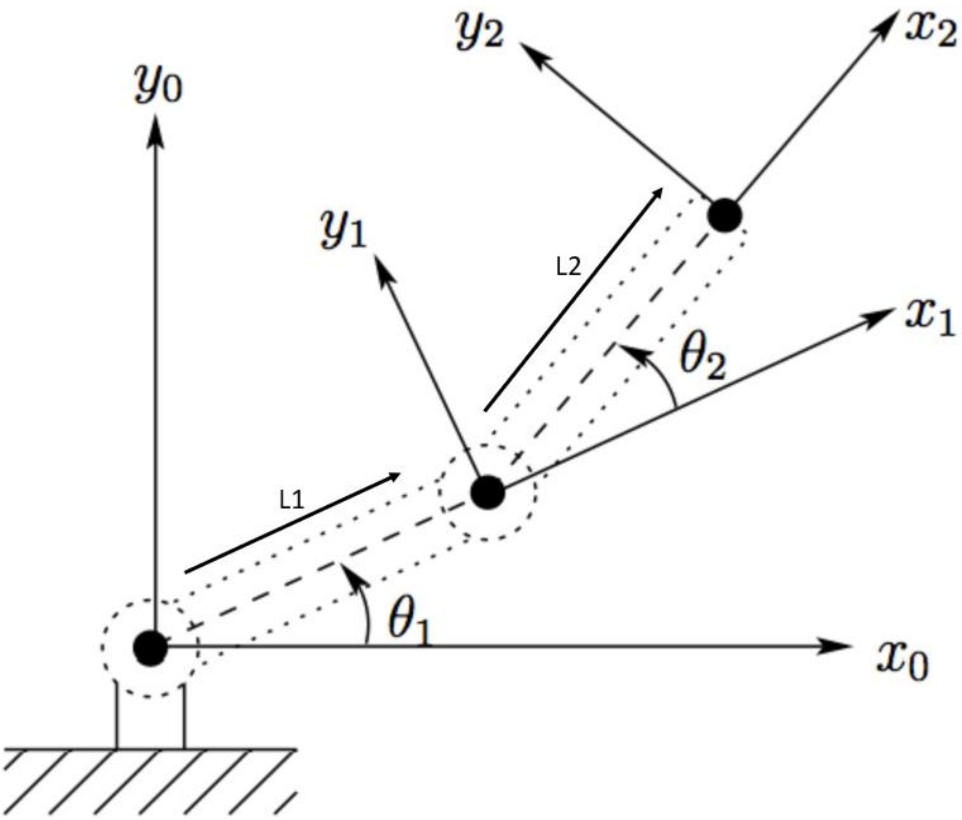

Fig. 2.1.1. *Inverse Kinematics Representation for 2-DOF*

Our design properties :

L1=0.310;	length of link 1
L2=0.152;	length of link 2
theta1=41	orientation of joint 1
theta2=36	orientation of joint 2

$$Px = l1 \times \cos(\theta_1) + l2\ \times \cos\,(\theta_1 + \theta_2)$$

$$Py = l1 \times \sin(\theta_1) + l2\ \times \sin\,(\theta_1 + \theta_2)$$

When we put the values, we get the location of the serial manipulator.

Matlab Part

```
syms L1 L2 theta1 theta2 X Y
L1=0.310;    % length of link 1
L2=0.152;    % length of link 2
theta1=41*(pi/180); %   orientation of joint 1
```

```
theta2=36*(pi/180); %   orientation of joint 2

% Define X and Y Coordinates of End Effector
X_RHS=L1*cos(theta1)+L2*cos(theta1+theta2);
Y_RHS=L1*sin(theta1)+L2*sin(theta1+theta2);

X_and_Y=[X_RHS Y_RHS]
```

Matlab Answer

```
X_and_Y =

    0.2682    0.3515
```

We calculate the position of the robotic arm in our matlab code by entering the lengths of the angles and links. We see that the position calculated from Matlab and the Solidworks drawing of the robotic arm are almost exactly the same.

It can be seen here:

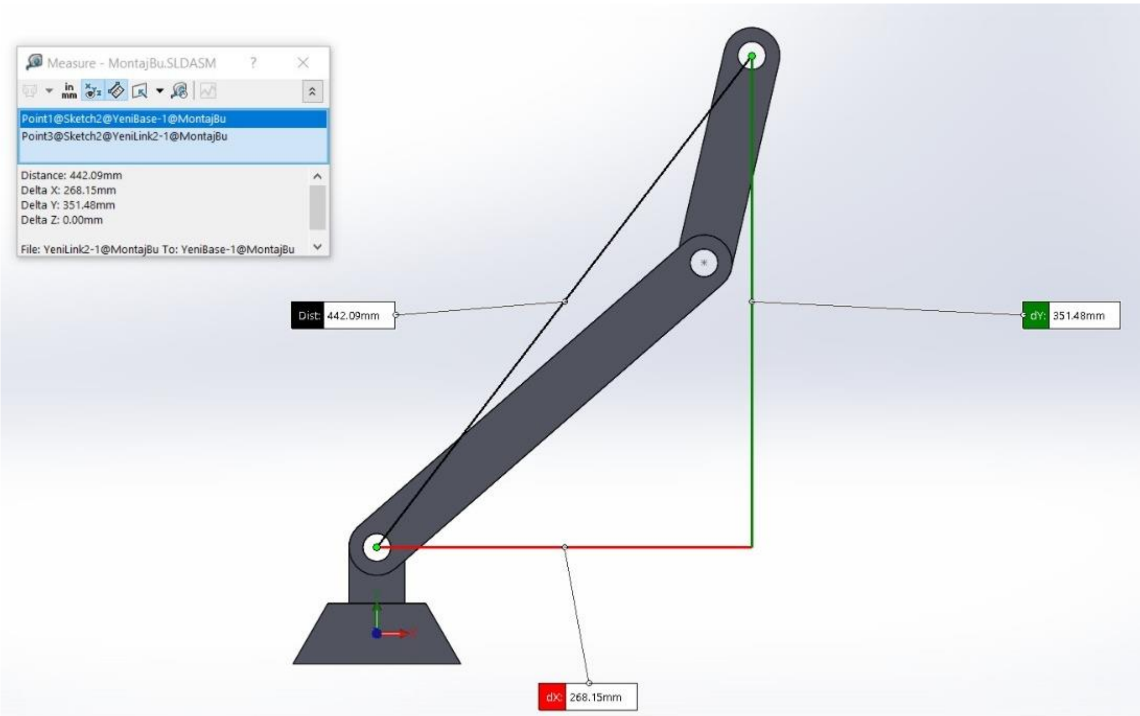

Fig.2.1.2. *Inverse Kinematic Proof between Link 2 and Link 1*

Matlab Part

```
syms L1 L2 theta1 theta2 X Y K1 K2
L1=0.310;  % length of link 1
L2=0.152;  % length of link 2
X= 0.2682; % position of actuator x direction
Y= 0.3515; % position of actuator y direction
%Show the pair of solutions for theta2
theta2=atan2(+(1-((X^2+Y^2-L1^2-
L2^2)/(2*L1*L2))^2)^0.5,(X^2+Y^2-L1^2-L2^2)/(2*L1*L2));
%Show the pair of solutions for theta1
K1=L1 + L2*cos(theta2);
K2=L2*sin(theta2);
theta1=atan2(Y,X)- atan2(K2,K1);
%Show the pair of solutions for Joint angles
JointAngles = [theta1*(180/pi) theta2*(180/pi)]
```

Matlab answer

```
JointAngles =


   41.0087   35.9610
```

In this section, we calculate our angle values from matlab by entering the position values and link lengths that we calculated in the previous matlab. The calculated angle and the angles in Solidworks are almost the same .

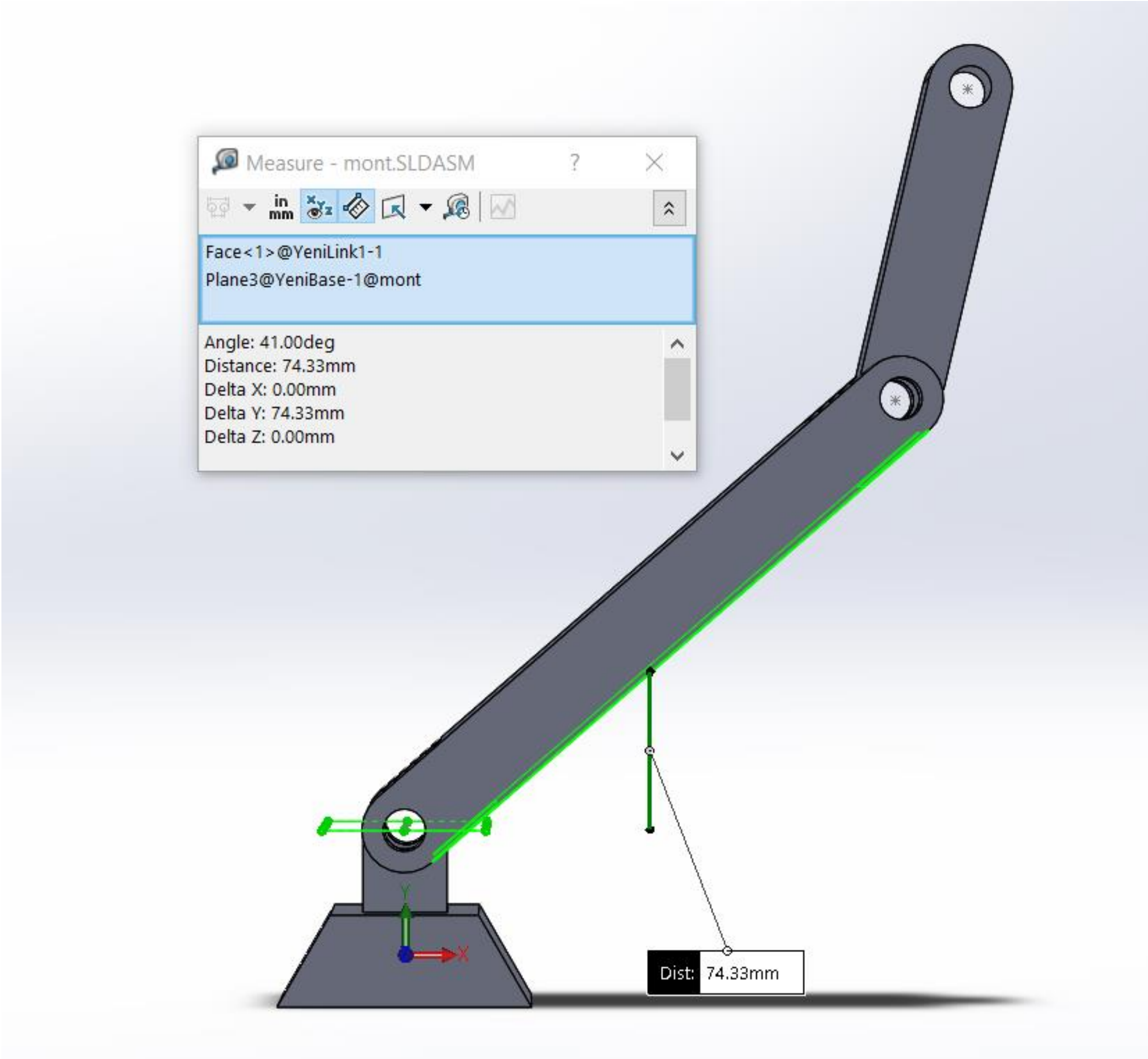

Fig.2.1.3. *Inverse Kinematic Proof between Link 1 and base*

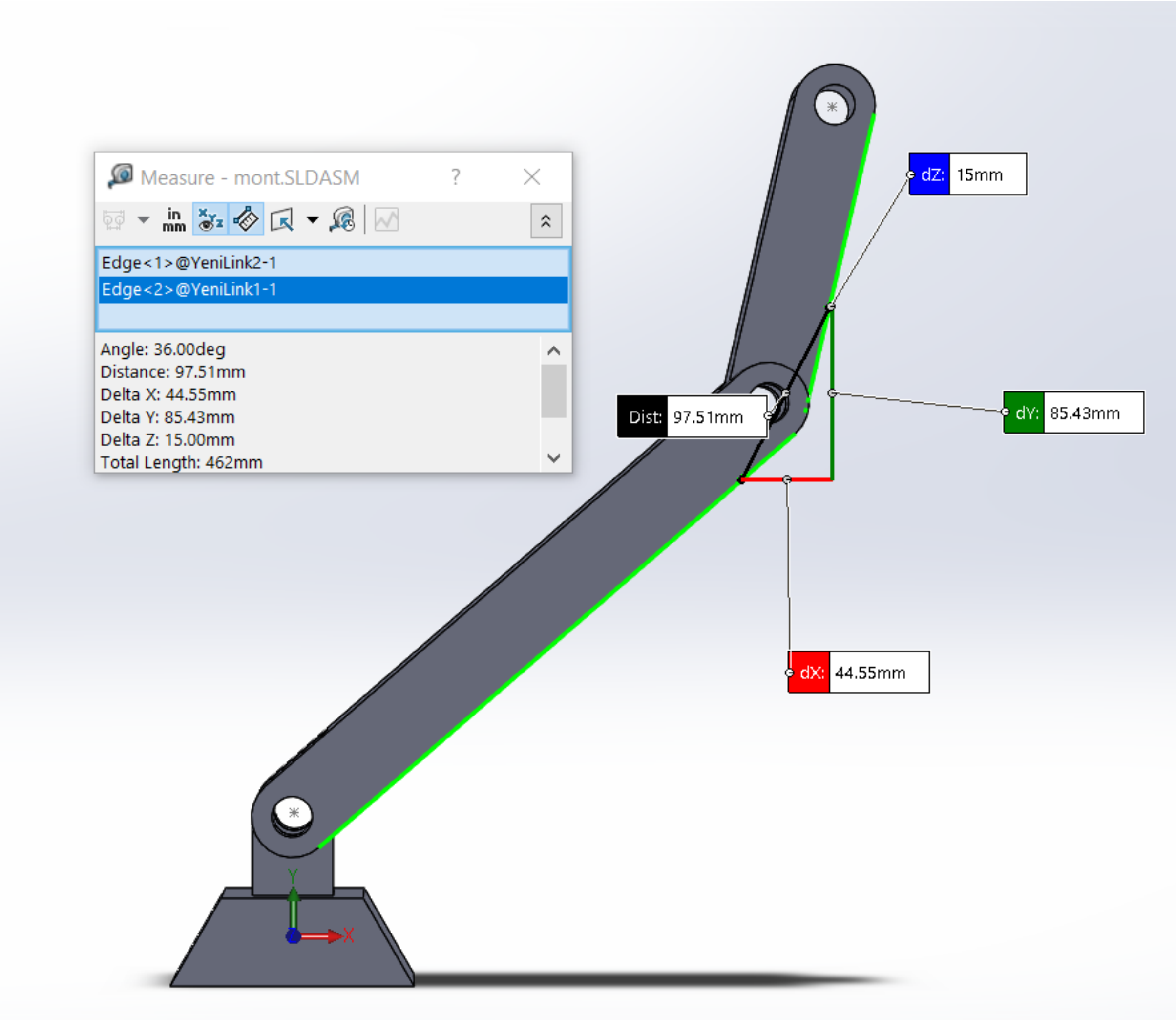

Fig.2.1.4. *Inverse Kinematic Proof between Link 1 and link 2*

Jacobian matris matlab calculation

```
%inverse kinematics
syms L1 L2 theta1 theta2 X Y

 % Define X and Y Coordinates of End Effector
 X_RHS=L1*cos(theta1)+L2*cos(theta1+theta2)
 Y_RHS=L1*sin(theta1)+L2*sin(theta1+theta2)

X_EQ = X == X_RHS;
Y_EQ = Y == Y_RHS;

S=solve([X_EQ Y_EQ],[theta1 theta2])
%Show the pair of solutions for theta1
simplify(S.theta1)
%Show the pair of solutions for theta1
simplify(S.theta2)
%Compute System Jacobian
J = jacobian([X_RHS Y_RHS],[theta1 theta2])
```

```
X_RHS =

L2*cos(theta1 + theta2) + L1*cos(theta1)

Y_RHS =

L2*sin(theta1 + theta2) + L1*sin(theta1)

S =

  struct with fields:


    theta1: [2×1 sym]

    theta2: [2×1 sym]
```

Matlab answer

```
ans =


2*atan((2*L1*Y + (- L1^4 + 2*L1^2*L2^2 + 2*L1^2*X^2 +
2*L1^2*Y^2 - L2^4 + 2*L2^2*X^2 + 2*L2^2*Y^2 - X^4 -
```

```
2*X^2*Y^2 - Y^4)^(1/2))/(L1^2 + 2*L1*X - L2^2 + X^2 + Y^2))

2*atan((2*L1*Y - (- L1^4 + 2*L1^2*L2^2 + 2*L1^2*X^2 + 2*L1^2*Y^2 - L2^4 + 2*L2^2*X^2 + 2*L2^2*Y^2 - X^4 - 2*X^2*Y^2 - Y^4)^(1/2))/(L1^2 + 2*L1*X - L2^2 + X^2 + Y^2))

ans =


-2*atan(((- L1^2 + 2*L1*L2 - L2^2 + X^2 + Y^2)*(L1^2 + 2*L1*L2 + L2^2 - X^2 - Y^2))^(1/2)/(- L1^2 + 2*L1*L2 - L2^2 + X^2 + Y^2))

 2*atan(((- L1^2 + 2*L1*L2 - L2^2 + X^2 + Y^2)*(L1^2 + 2*L1*L2 + L2^2 - X^2 - Y^2))^(1/2)/(- L1^2 + 2*L1*L2 - L2^2 + X^2 + Y^2))


J =


[- L2*sin(theta1 + theta2) -  L1*sin(theta1),    -L2*sin(theta1 + theta2)]

[  L2*cos(theta1 + theta2) + L1*cos(theta1),    L2*cos(theta1 + theta2)]
```

III. CHAPTER (Dynamic Analysis)

In robotics, several methods have been proposed for the mathematical description of the movement of robots. These methods are based on the use of classical principles and equations of mechanics like Lagrange– Euler, Newton-Euler and Gauss. The purpose is to develop a set of equations that describe the dynamical behavior of a manipulator. It`s a mandatory part of the process, since a dynamical model can be used for computer simulation of a robotic system. By examining the behavior of the model under various operating conditions, it is possible to predict how a robotic system will behave when it is built because it calculates the trajectory of movement, speed and acceleration of each parts of the manipulator when solving the inverse problem of dynamics. The developed module allows for an arbitrary robotic manipulator to automatically compose equations and solve both direct and inverse problems of dynamics. There are two types of dynamical problems: direct dynamics and inverse dynamics. The direct dynamics problem is to find the response of a robot arm corresponding to some applied torques and/or forces. That is, given a vector of joint torques or forces, we wish to compute the resulting motion of the manipulator as a function of time. Various manufacturing automation tasks can be examined without the need of a real.

Dynamic analysis is a branch of the theory of mechanisms and machines, which studies the movement of the links of a mechanism under the action of a given forces. The main goal of dynamic analysis is to establish general relationships between the joint actuator torques and the motion of the structure. These dependencies are determined from the equations of motion of the mechanism. So in this project we explain the dynamic analysis of a mechanism until the 6th degree of freedom by using programs like Simulaink Matlab. We followed the method and equations from book to create a code for MATLAB. They analyzed the static problem of robot arm using Lagrange laws of motion which can be observed in the next part.

References to equations

So, we start calculations with the 2nd degree of freedom:

The Lagrangian description of a "system" is based on a quantity, L, called the "Lagrangian", which is defined as:

$$L = K - U$$

where K is the kinetic energy of the system, and U is its potential energy. A "system" can be a rather complex collection of objects, although we will illustrate how the Lagrangian formulation is implemented for a single object of mass m moving in one dimension under the influence of gravity.

In the modern formulation of classical mechanics, the motion of the system will be such that the following integral is minimized:

$$S = \int L dt$$

Where, L can depend on time explicitly or implicitly (through the fact that position and velocity, on which the Lagrangian depends, are themselves time-dependent). The requirement that the above integral be minimized is called the "Principle of Least Action"1, and is thought to be the fundamental principle that describes all of the laws of physics. The condition for the action to be minimized is given by the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial v_x}\right) - \frac{\partial L}{\partial x} = 0$$

Thus, in the Lagrangian formulation, one first writes down the Lagrangian for the system, and then uses the Euler-Lagrange equation to obtain the "equations of motion" for the system.

Here is an example for Lagrangian Dynamics of a Planar R-DOF Manipulator In this example we formulate Lagrange's equations of motion for the planar 2- dof manipulator:

For a slender rod, a and b are much smaller than c. The inertia matrix can be approximated by

Inertia Matrices:

$$I_{ii} = \frac{1}{12} m_i a_i^2 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \rightarrow \quad I_1 = \frac{1}{12} m_1 a_1^2 \begin{bmatrix} s^2\theta_1 & -s\theta_1 c\theta_1 & 0 \\ -s\theta_1 c\theta_1 & c^2\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$I11 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.0056 & 0 \\ 0 & 0 & 0.0056 \end{bmatrix}$$

$$I22 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.3870 & 0 \\ 0 & 0 & 0.3870 \end{bmatrix}$$

$${}^0_1R = \begin{bmatrix} 0.7547 & -0.6561 & 0 \\ 0.6561 & 0.7547 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

$${}^1_2R = \begin{bmatrix} 0.8090 & -0.5878 & 0 \\ 0.5878 & 0.8090 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

$${}^0_2R = \begin{bmatrix} 0.2250 & -0.9744 & 0 \\ 0.9744 & 0.2250 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

$${}^0_1RT = \begin{bmatrix} 0.7547 & 0.6561 & 0 \\ -0.6561 & 0.7547 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

$${}^0_2RT = \begin{bmatrix} 0.2250 & 0.9744 & 0 \\ -0.9744 & 0.2250 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

The inertia matrices of links 1 and 2 about their respective centers of mass and expressed in the base frame are obtained by substituting previous Equation for i=1,2

$$I_2 = \frac{1}{12} m_2 a_2^2 \begin{bmatrix} s^2\theta_{12} & -s\theta_{12} c\theta_{12} & 0 \\ -s\theta_{12} c\theta_{12} & c^2\theta_{12} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$I1 = \begin{bmatrix} 0.0024 & -0.00280 & 0 \\ -0.0028 & 0.0032 & 0 \\ 0 & 0 & 0.0056 \end{bmatrix}$$

$$I2 = \begin{bmatrix} 0.3674 & -0.0848 & 0 \\ -0.0848 & 0.0196 & 0 \\ 0 & 0 & 0.3870 \end{bmatrix}$$

Link Jacobian Matrices. The position vectors of the centers of mass of links 1 and 2 with respect to be various link frames and expressed in the base frame are given by:

$$p_{0c1}^{*} = \begin{bmatrix} \frac{1}{2} a_1 c\theta_1 \\ \frac{1}{2} a_1 s\theta_1 \\ 0 \end{bmatrix}, \quad p_{1c2}^{*} = \begin{bmatrix} \frac{1}{2} a_2 c\theta_{12} \\ \frac{1}{2} a_2 s\theta_{12} \\ 0 \end{bmatrix}, \quad p_{0c2}^{*} = \begin{bmatrix} \frac{1}{2} a_1 c\theta_1 + \frac{1}{2} a_2 c\theta_{12} \\ \frac{1}{2} a_1 s\theta_1 + \frac{1}{2} a_2 s\theta_{12} \\ 0 \end{bmatrix}$$

$$Pc_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad Pc_{01} = \begin{bmatrix} 0.11700 \\ 00.1017 \\ 0 \end{bmatrix} \quad Pc_{12} = \begin{bmatrix} 0.0171 \\ 0.0741 \\ 0 \end{bmatrix} \quad Pc_{02} = \begin{bmatrix} 0.2511 \\ 0.2774 \\ 0 \end{bmatrix}$$

Link Jacobian Submatrices:

$$J_{v1} = \begin{bmatrix} -\frac{1}{2} a_1 s\theta_1 & 0 \\ \frac{1}{2} a_1 c\theta_1 & 0 \\ 0 & 0 \end{bmatrix}, \quad J_{w1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad J_{v2} = \begin{bmatrix} -a_1 s\theta_1 - \frac{1}{2} a_2 s\theta_{12} & -\frac{1}{2} a_2 s\theta_{12} \\ a_1 c\theta_1 + \frac{1}{2} a_2 c\theta_{12} & \frac{1}{2} a_2 c\theta_{12} \\ 0 & 0 \end{bmatrix},$$

$$J_{w2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$Jv_1 = \begin{bmatrix} -0.1017 & 0 \\ 0.1170 & 0 \\ 0 & 0 \end{bmatrix} \quad Jw_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \quad Jv_2 = \begin{bmatrix} -0.2774 & -0.0741 \\ 0.2511 & 0.0171 \\ 0 & 0 \end{bmatrix} \quad Jw_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$Jv_1T = \begin{bmatrix} -0.1017 & 0.1170 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$Jv_2T = \begin{bmatrix} -0.2774 & 0.2511 & 0 \\ -0.0741 & 0.0171 & 0 \end{bmatrix}$$

Manipulator Inertia Matrix:

$$M = J_{v1}^T m_1 J_{v1} + J_{w1}^T I_1 J_{w1} + J_{v2}^T m_2 J_{v2} + J_{w2}^T I_2 J_{w2}\ ,\ M$$

$$= \begin{bmatrix} \frac{1}{3} m_1 a_1^2 + m_2 \left(a_1^2 + a_1 a_2 c\theta_2 + \frac{1}{3} a_2^2 \right) & m_2 \left(\frac{1}{2} a_1 a_2 c\theta_2 + \frac{1}{3} a_2^2 \right) \\ m_2 \left(\frac{1}{2} a_1 a_2 c\theta_2 + \frac{1}{3} a_2^2 \right) & \frac{1}{3} m_2 a_2^2 \end{bmatrix}$$

$$Jv_1T = \begin{bmatrix} -0.1017 & 0.1170 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$Jv_2T = \begin{bmatrix} -0.2774 & 0.2511 & 0 \\ -0.0741 & 0.0171 & 0 \end{bmatrix}$$

Velocity Coupling Vector:

$$V_1 = \sum_{j=1}^{2} \sum_{k=1}^{2} \left(\frac{\partial M_{1j}}{\partial \theta_k} - \frac{1}{2} \frac{\partial M_{jk}}{\partial \theta_1} \right) \dot{\theta}_j \dot{\theta}_k = -m_2 a_1 a_2 s\theta_2 \left(\dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} \dot{\theta}_2^2 \right)$$

$$V_2 = \sum_{j=1}^{2} \sum_{k=1}^{2} \left(\frac{\partial M_{2j}}{\partial \theta_k} - \frac{1}{2} \frac{\partial M_{jk}}{\partial \theta_2} \right) \dot{\theta}_j \dot{\theta}_k = -m_2 a_1 a_2 s\theta_2 \dot{\theta}_1^2$$

V1 =	V2=
-1.2693	0.4704

Gravitational Vector:

$$G_1 = \frac{1}{2} m_1 g_c a_1 c\theta_1 + m_2 g_c a_1 c\theta_1 + \frac{1}{2} m_2 g_c a_2 c\theta_{12}$$

$$G_2 = \frac{1}{2} m_2 g_c a_2 c\theta_{12}$$

$$g = \begin{bmatrix} 0 \\ 9.8100 \\ 0 \end{bmatrix} \quad gT = [0 \quad 9.8100 \quad 0] \quad G1 = 1.3029 \quad G2 = 0.0337$$

Torque Vectors:

$$T_1 = \left[\left(\frac{1}{3}m_1 + m_2\right)a_1^2 + m_2a_1a_2c\theta_2 + \frac{1}{3}m_2a_2^2\right]\ddot{\theta}_1 + \left(\frac{1}{2}m_2a_1a_2c\theta_2 + \frac{1}{3}m_2a_2^2\right)\ddot{\theta}_2 - m_2a_1a_2s\theta_2\left(\dot{\theta}_1\dot{\theta}_2 + \frac{1}{2}\dot{\theta}_2^2\right) + g_c\left[\left(\frac{1}{2}m_1 + m_2\right)a_1c\theta_1 + \frac{1}{2}m_2a_2c\theta_{12}\right]$$

$$T_2 = \left[\frac{1}{2}m_2a_1a_2c\theta_2 + \frac{1}{3}m_2a_2^2\right]\ddot{\theta}_1 + \frac{1}{3}m_2a_2^2\ddot{\theta}_2 + \frac{1}{2}m_2a_1a_2s\theta_2\dot{\theta}_1^2 + \frac{1}{2}m_2g_ca_2c\theta_{12}$$

$T1 = 1.3029$

$T2 = 0.0337$

3.1 Workspace with Matlab For Dynamic Analysis

Here is our code for 2-Dof Robotic Arm:

```
syms  a Q  x I11 I22 I1 I2 m1 m2 a1 a2 R01 R12 Q1 Q2 R01T
R12T PC00 PC01 PC12 PC02 X Y L1 L2 X_DOT Y_DOT Z0 Z1 O1
O2  M   Z2 gc g gt G1 G2 Q1DOT Q2DOT j k Q1DOT2 Q2DOT2
T1=zeros(1,5)
T2=zeros(1,5)

for i=1:5
   x=i

% x=1;
%time
gc=9.81;     %gravity
m={0.704,0.201};    %link  mass
a1=0.310;    %link 1 length
a2=0.152;    %link 2 length

q1=Q1(x)
q2=Q2(x)
I11=(1/12*m{1}*a1^2)*[0 0 0; 0 1 0; 0 0 1;]        %link 1
inertia matrix
I22=(1/12*m{2}*a2^2)*[0 0 0; 0 1 0; 0 0 1;]        %link 2
inertia matrix
R01=[cos(Q1(x)) -sin(Q1(x)) 0 ; sin(Q1(x)) cos(Q1(x)) 0 ;
0 0 1 ;]
```

```
R12=[cos(Q2(x)) -sin(Q2(x)) 0 ; sin(Q2(x)) cos(Q2(x)) 0 ;
0 0 1 ;]
R02=R01*R12
R01T=transpose(R01)
R02T=transpose(R02)
I={R01*I11*R01T,R02*I22*R02T}

% The position vectors of the centers of mass of link 1
and 2.
PC00=[0;0;0]
PC01=[a1/2*cos(Q1(x)); a1/2*sin(Q1(x)); 0;]
PC12=[a2/2*(cos(Q1(x))*cos(Q2(x)) -
sin(Q1(x))*sin(Q2(x))); a2/2*(cos(Q1(x))*sin(Q2(x)) +
cos(Q2(x))*sin(Q1(x))); 0;]
PC02=[a1*cos(Q1(x)) + a2/2*(cos(Q1(x))*cos(Q2(x)) -
sin(Q1(x))*sin(Q2(x))); a1*sin(Q1(x)) +
a2/2*(cos(Q1(x))*sin(Q2(x)) + cos(Q2(x))*sin(Q1(x))); 0;]

Z0=[0;0;1]
Z1=[0;0;1]

% The link Jacobian Submatrices Jvi Jwi
Jv={[cross(Z0,(PC01-PC00))
cross(Z1,PC00)],[cross(Z0,(PC02-PC00)) cross(Z1,(PC12-
PC00))]}
Jw={[0 0; 0 0; 1 0;],[0 0; 0 0;1 1;]}

JvT = {transpose(Jv{1}),transpose(Jv{2})};
JwT = {transpose(Jw{1}),transpose(Jw{2})};

%Manipulator inertia matrix
n=2;
M=msum(n,Jv,Jw,m,I,JwT,JvT)

%Velocity coupling vector
V = {-m{2}*a1*a2*sin(Q2(x))*(Q1DOT(x)*Q2DOT(x) +
0.5*Q2DOT(x)^2),(1/2)*m{2}*a1*a2*sin(Q2(x))*Q1DOT(x)^2}


%Gravitational vector
g=[0;gc;0]
gt=transpose(g)
G1=gt*[m{1}*Jv{1}(:,1) + m{2}*Jv{2}(:,1)]
G2=gt*[m{1}*Jv{1}(:,2) + m{2}*Jv{2}(:,2)]

%Lagrange's Equations of motion
T1(i)=M(1,1)*Q1DOT2 + M(1,2)*Q2DOT2 + V{1}+ G1
T2(i)=M(2,1)*Q1DOT2 + M(2,2)*Q2DOT2 + V{2} + G2
```

```
n=2;
mont_DataFile;
mont;
%functions

end
t=1:5
figure(1)
plot(t,T1,'bo-')
title('Joint 1')
xlabel('Time(Sec)')
ylabel('Torque(Nm)')
figure(2)
plot(t,T2,'ro-')
title('Joint 2')
xlabel('Time(Sec)')
ylabel('Torque(Nm)')

function Mi=Minertia(jv,jw,m,I,jwT,jvT)

Mi=(jvT*m*jv + jwT*I*jw)

end

function Msum=msum(n,Jv,Jw,m,I,JwT,JvT)
Msum=0;
for i=1:n
    Msum=Msum +
Minertia(Jv{i},Jw{i},m{i},I{i},JwT{i},JvT{i})
end
end


function Q = Q1(x)
    Q = (x^2+3*x+1)*(pi/180);
end

function Q = Q1DOT(x)
    Q = 2*x+3;

end

function Q = Q1DOT2(x)
    Q = 2;

end

function Q = Q2(x)
```

```
    Q = (x^2+2*x+1)*(pi/180);
end

function Q = Q2DOT(x)
    Q = 2*x+2;
end

function Q = Q2DOT2(x)
    Q = 2;
end
```

The second code for 3-DOF robotic arm:

```
syms  I11  x m I22 I1 I2 Jw3 m1 m2 m3 a1 a2 a3 R01 R12 Q1
Q2 Q3 R01T R12T PC00 Q3 PC03 PC01 PC12 PC02 X Y L1 L2
X_DOT Y_DOT Z0 Z1 O1 O2 Jvi Jwi M Jv1T Jv2T Jw1T Jw2T  Z2
gc g gt G1 G2 Q1DOT Q2DOT j k Q1DOT2 Q2DOT2

T1=zeros(1,5)
T2=zeros(1,5)
T3=zeros(1,5)
for i=1:5
   x=i

% x=1;
% %time
gc=9.81;     %gravity
m={0.704,0.704,0.201};    %link  mass
a1=0.310;    %link 1 length
a2=0.310;    %link 2 length
a3=0.152;    %link 3 length
q1=Q1(x)
q2=Q2(x)
q3=Q3(x)
I11=(1/12*m{1}*a1^2)*[0 0 0; 0 1 0; 0 0 1;]        %link 1
inertia matrix
I22=(1/12*m{2}*a2^2)*[0 0 0; 0 1 0; 0 0 1;]        %link 2
inertia matrix
I33=(1/12*m{3}*a3^2)*[0 0 0; 0 1 0; 0 0 1;]        %link 3
inertia matrix
R01=[cos(Q1(x)) -sin(Q1(x)) 0 ; sin(Q1(x)) cos(Q1(x)) 0 ;
0 0 1 ;]
R12=[cos(Q2(x)) -sin(Q2(x)) 0 ; sin(Q2(x)) cos(Q2(x)) 0 ;
0 0 1 ;]
R23=[cos(Q3(x)) -sin(Q3(x)) 0 ; sin(Q3(x)) cos(Q3(x)) 0 ;
0 0 1 ;]
R02=R01*R12
```

```
R03=R02*R23
R01T=transpose(R01)
R02T=transpose(R02)
R03T=transpose(R03)
I={R01*I11*R01T,R02*I22*R02T,R03*I33*R03T}
% The position vectors of the centers of mass of link 1,2
and  3.
PC00=[0;0;0]
PC01=[a1/2*cos(Q1(x)); a1/2*sin(Q1(x)); 0;]
PC12=[a2/2*(cos(Q1(x)+Q2(x))); a2/2*(sin(Q1(x)+Q2(x)));
0;]
PC02=[a1*cos(Q1(x)) + a2/2*(cos(Q1(x)+Q2(x)));
a1*sin(Q1(x)) + a2/2*(sin(Q1(x)+Q2(x))); 0;]
PC23=[1/2*(a3*cos(Q3(x))*cos(Q2(x)+Q1(x)) -
a3*sin(Q3(x))*sin(Q1(x)+Q2(x)));
1/2*(a3*sin(Q1(x)+Q2(x))*cos(Q3(x)) -
a3*sin(Q3(x))*cos(Q1(x)+Q2(x)));0;]
PC03=[a1*cos(Q1(x)) + a2*cos(Q1(x)+Q2(x)) +
1/2*(a3*cos(Q3(x))*cos(Q2(x)+Q1(x)) -
a3*sin(Q3(x))*sin(Q1(x)+Q2(x)));  a1*sin(Q1(x)) +
a2*sin(Q1(x)+Q2(x)) + 1/2*(a3*sin(Q1(x)+Q2(x))*cos(Q3(x))
- a3*sin(Q3(x))*cos(Q1(x)+Q2(x)));0;]
Z0=[0;0;1]
Z1=[0;0;1]
Z3=[0;0;1]
% The link Jacobian Submatrices Jvi Jwi
Jv={[cross(Z0,(PC01-PC00)) cross(Z1,PC00)
cross(Z1,PC00)],[cross(Z0,(PC02-PC00)) cross(Z0,(PC12-
PC01)) cross(Z1,PC00)],[cross(Z0,(PC03-PC00))
cross(Z1,(PC23-PC00)) cross(Z1,PC00)]}
Jw={[0 0 0; 0 0 0; 1 0 0;],[0 0 0; 0 0 0;1 1 0;],[0 0 0;
0 0 0;1 1 0;]}

%Transpose
JvT =
{transpose(Jv{1}),transpose(Jv{2}),transpose(Jv{3})};
JwT =
{transpose(Jw{1}),transpose(Jw{2}),transpose(Jw{3})};

%Manipulator inertia matrix
n=3;
M=msum(n,Jv,Jw,m,I,JwT,JvT)


%Velocity coupling vector
```

```
V = {-
(m2+2*m3)*a1*a2*sin(Q2(x))*(Q1DOT(x)*Q2DOT(x)+1/2*Q2DOT2(
x)^2),(1/2*m2 + m3)*a1*a2*sin(Q2(x))*Q1DOT(x)^2,Q3(x)}


%Gravitational vector
g=[0;gc;0]
gt=transpose(g)
G1=gt*[m{1}*Jv{1}(:,1) + m{2}*Jv{2}(:,1) +
m{3}*Jv{3}(:,1)]
G2=gt*[m{1}*Jv{1}(:,2) + m{2}*Jv{2}(:,2) +
m{3}*Jv{3}(:,2)]
G3=gt*[m{1}*Jv{1}(:,3) + m{2}*Jv{2}(:,3) +
m{3}*Jv{3}(:,3)]
%
% %Lagrange's Equations of motion
T1(i)=M(1,1)*Q1DOT2 + M(1,2)*Q2DOT2 + M(1,3)*Q3DOT2 +
V{1} + G1
T2(i)=M(2,1)*Q1DOT2 + M(2,2)*Q2DOT2 + M(2,3)*Q3DOT2 +
V{2} + G2
T3(i)=M(3,1)*Q1DOT2 + M(3,2)*Q2DOT2 + M(3,3)*Q3DOT2 +
V{3} + G3
n=3;
% dof3_DataFile;
% dof3;
end
t=1:5
figure(1)
plot(t,T1,'bo-')
title('Joint 1')
xlabel('Time(Sec)')
ylabel('Torque(Nm)')
figure(2)
plot(t,T2,'ro-')
title('Joint 2')
xlabel('Time(Sec)')
ylabel('Torque(Nm)')
figure(3)
plot(t,T3,'go-')
title('Joint 3')
xlabel('Time(Sec)')
ylabel('Torque(Nm)')

function Mi=Minertia(jv,jw,m,I,jwT,jvT)

Mi=(jvT*m*jv + jwT*I*jw)
```

```
end

function Msum=msum(n,Jv,Jw,m,I,JwT,JvT)
Msum=0;
for i=1:n
    Msum=Msum +
Minertia(Jv{i},Jw{i},m{i},I{i},JwT{i},JvT{i})
end
end

function Q = Q1(x)
    Q = (x^2+3*x+1)*(pi/180);
end

function Q = Q1DOT(x)
    Q = 2*x+3;

end

function Q = Q1DOT2(x)
    Q = 2;

end

function Q = Q2(x)
    Q = (x^2+2*x+1)*(pi/180);
end

function Q = Q2DOT(x)
    Q = 2*x+2;
end

function Q = Q2DOT2(x)
    Q = 2;
end

function Q = Q3(x)
    Q = (x^2+x+1)*(pi/180);
end

function Q = Q3DOT(x)
    Q = 2*x+1;
end

function Q = Q3DOT2(x)
    Q = 2;
end
```

3.2. Modelling of the 2-DOF robotic arm.

Dynamic analysis of the mechanism was carried out in accordance with the simulation concept of the SolidWorks. Before conducting a dynamic analysis, it is important to define the goal of the analysis prior to the formulation of the finite element model. The primary steps in performing a dynamic analysis are summarized as follows:

I. Define the dynamic environment.
II. Formulate the proper finite element model.
III. Select and apply the appropriate analysis approach to determine the behavior of the structure.
IV. Evaluate the results.

However, in order to represent the mechanism in Matlab, first we need to create it by using SolidWorks. Therefore, we designed a robotic arm from 3 parts.

Which are Link 1:

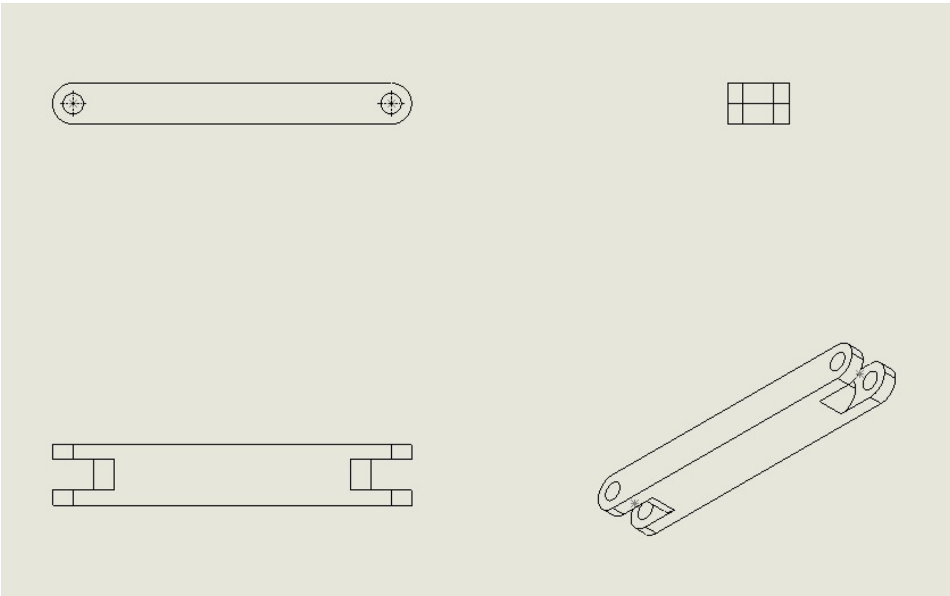

Fig 3.1.1 *Design of Link 1 in SolidWorks*

Link 2:

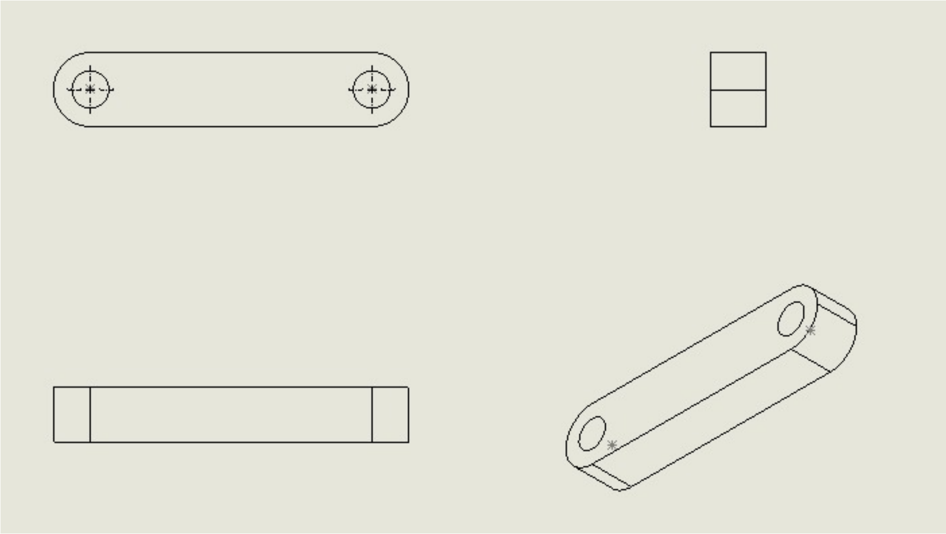

Fig 3.1.2 *Design of Link 2 in SolidWorks*

Base:

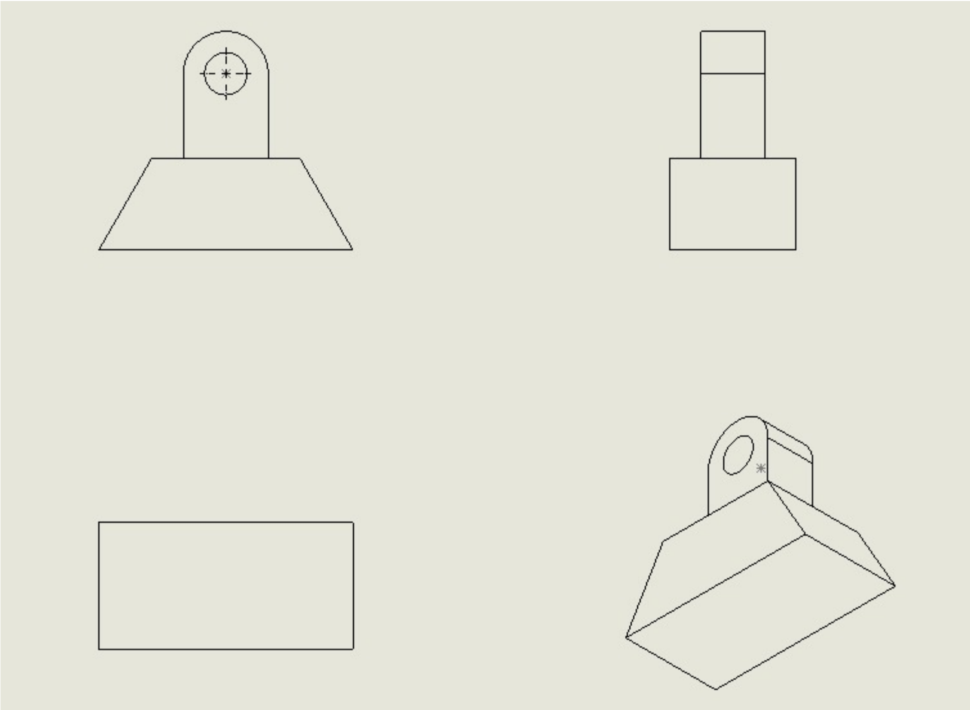

Fig 3.1.3 *Design of Base in SolidWorks*

Finally, we did the assembly:

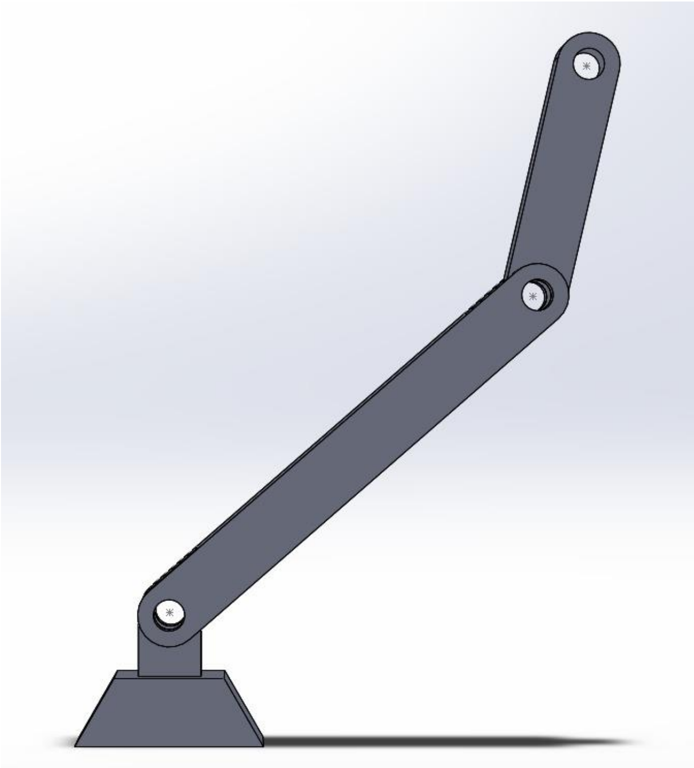

Fig. 3.1.4 *Improved Robotic Design of 2-DOF Robotic Arm*

As a result, the final mechanism had 2 degree of freedom.

Also an important part of this analysis is mass properties which can be obtained from Solid Works.

Properties of the 2-DOF robotic arm:

Proporties	Link 1	Link 2
Mass(kg)	0.704	0.201
Mass of Inertia (kg*m^2)	$7{,}7389*10^{-2}$	$6{,}7183*10^{-3}$
Length(m)	0,309	0,152
Center of mass (m)	0,134	0,085

Table 3.1 *Properties of the 2-DOF robotic arm*

At last we can carry out our mechanism to Simulink in order to compare the results from Matlab and simulation values. First of all, we had to create an arm in SolidWorks and then transfer to Matlab by using Toolbox.

First of all, we had repair the geometry of the shape, since there were gaps, we need to get rid of. Secondly, the functions were added to each Link depending to the time, so we obtained this table:

Insomuch, Link 1 and Link 2 have different rotation angles, since they have different equations for the Theta that we mentioned earlier. However, the time for them is the same and equals 5 seconds.

For Instance if t=5:

$$theta1(5) = x^2 + 3x + 1 = 41°$$

$$theta2(5) = x^2 + 2x + 1 = 36°$$

Moreover, gravity was added to the simulation in –y direction. By the end, after all this steps were done, final torque and force of each joint can be achieved.

3.2 Modelling of the 3-DOF robotic arm.

The next step of the project is to design a robotic arm with 3 degree of freedom. We had to add one more Link and joint.

Here is the design in SolidWorks:

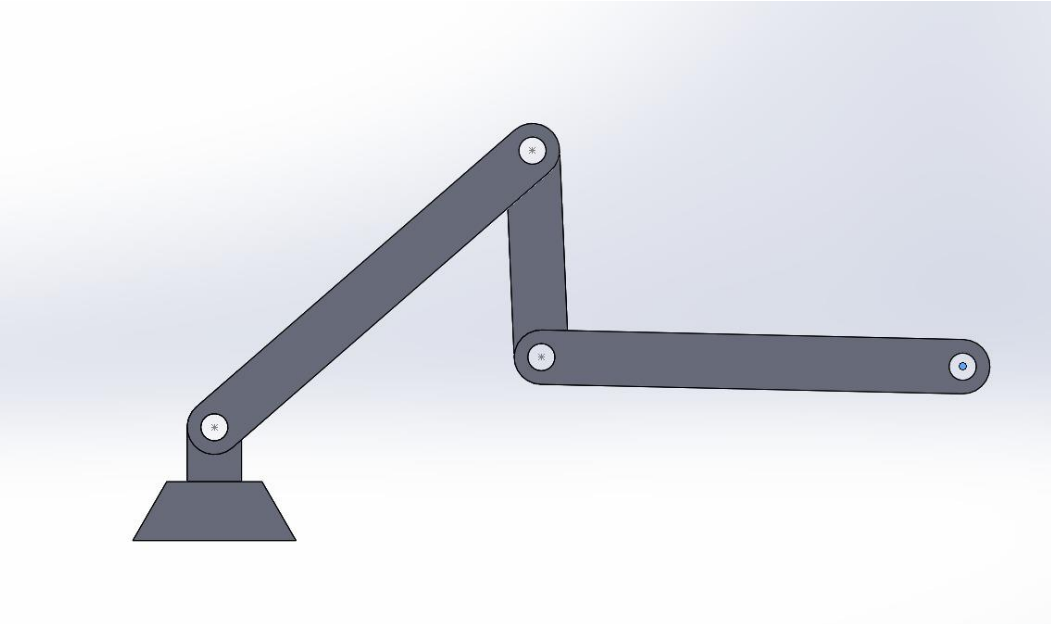

Fig. 3.2.1 *Improved Robotic Design of 3-DOF Robotic Arm*

SUMMARY AND CONCLUSION

4.1. Graphical Analysis of 2-DOF Robotic Arm.

So here we provide the graphics we obtained from SMULINK and MATLAB. Firstly, we will discuss the 2-DOF Robotic Arm:

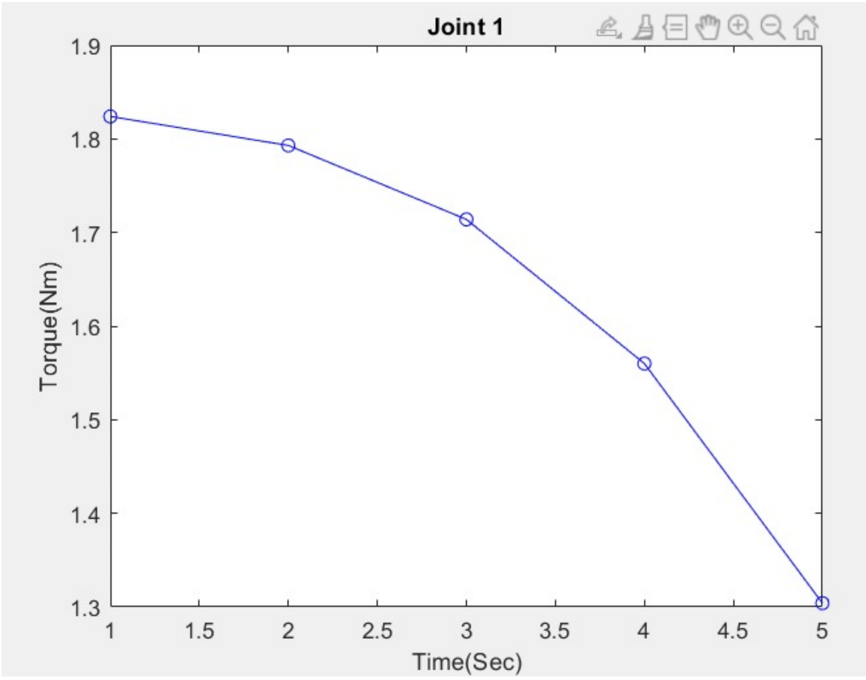

Fig. 4.1.1. *Torque graphic of Joint 1* from SIMULINK

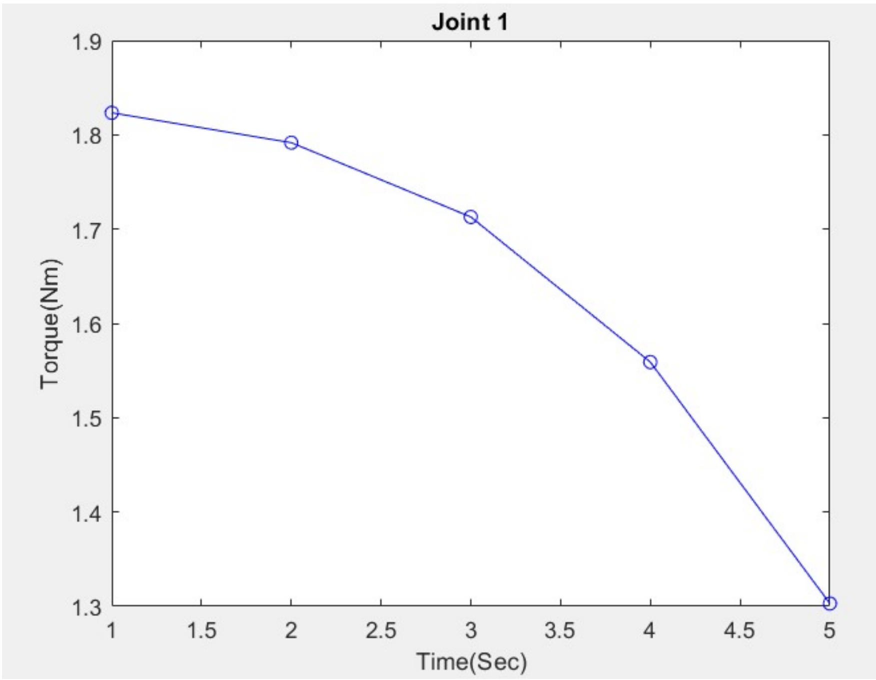

Fig. 4.1.2. *Torque graphic of Joint 1 from MATLAB*

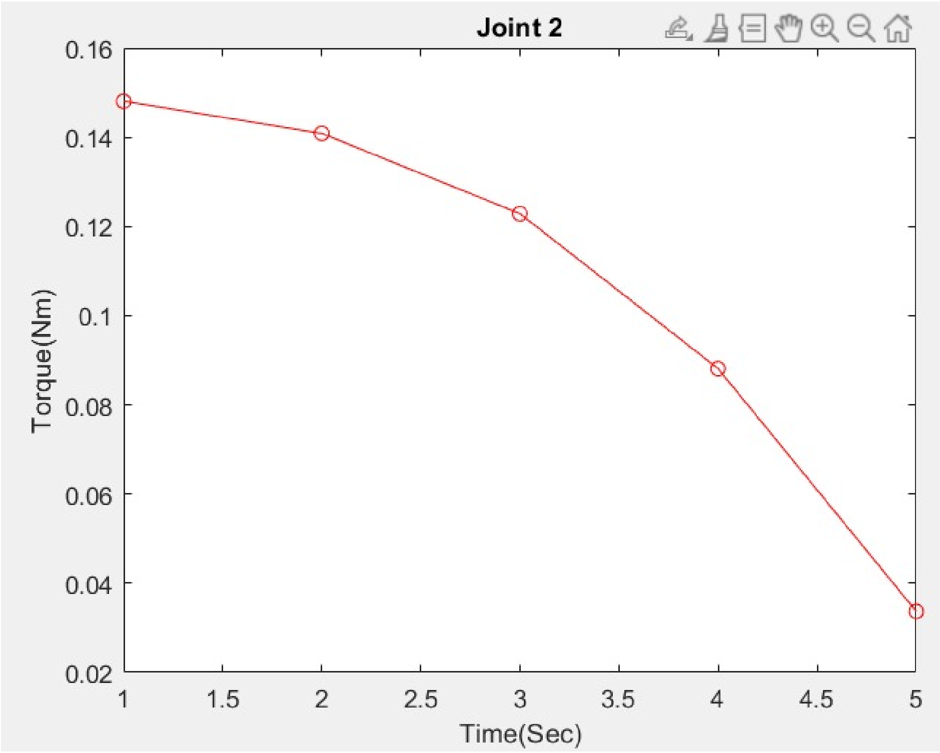

Fig. 4.1.3. *Torque graphic of Joint 2 from SIMULINK*

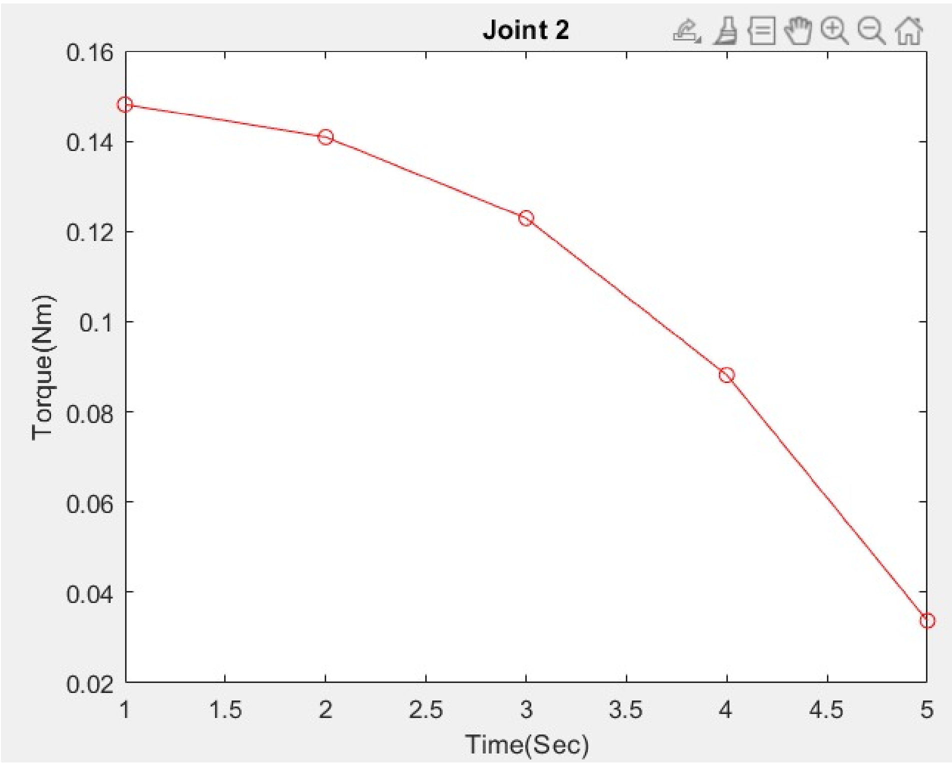

Fig. 4.1.4. *Torque graphic of Joint 2 from MATLAB*

Also, we provide the block diagram of 2-DOF arm below:

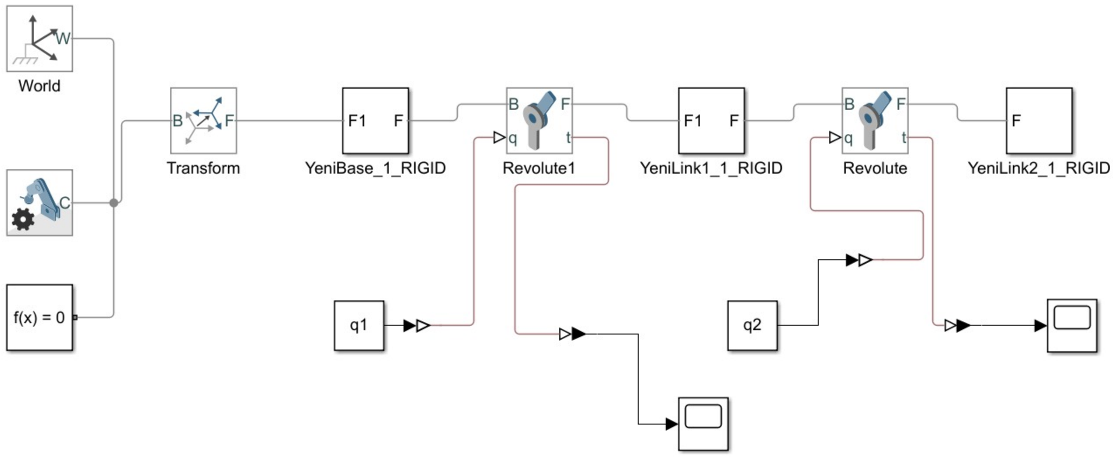

Fig. 4.1.5. *Block Diagram of 2-DOF arm*

4.2. Graphical Analysis of 3-DOF Robotic Arm.

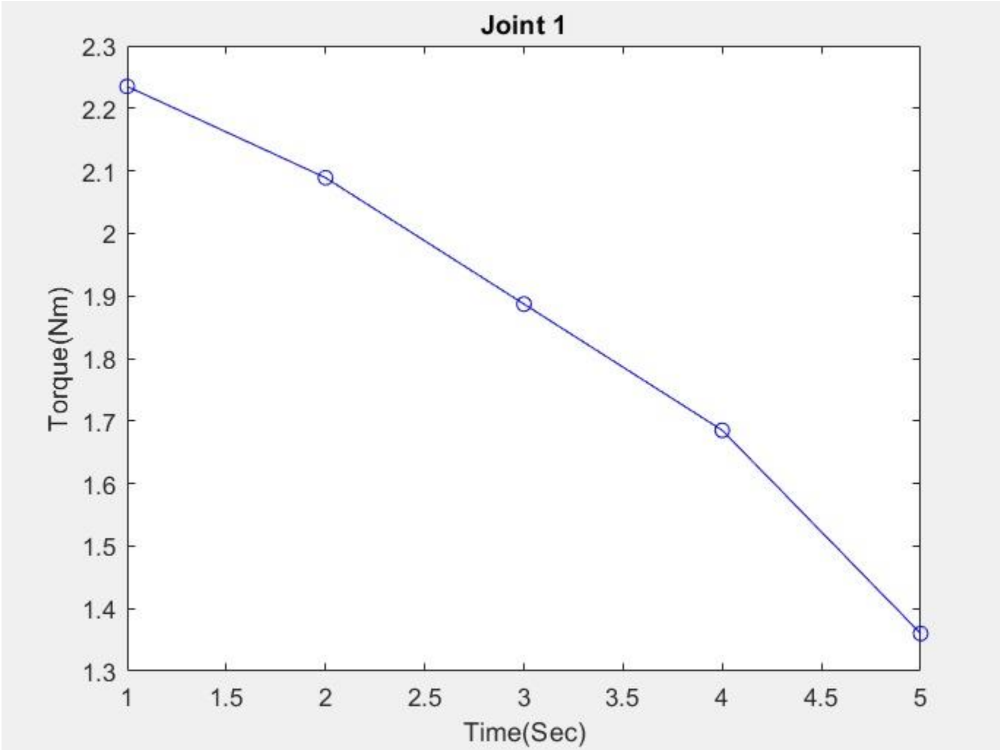

Fig. 4.2.1 *Torque graphic of Joint 1 from MATLAB*

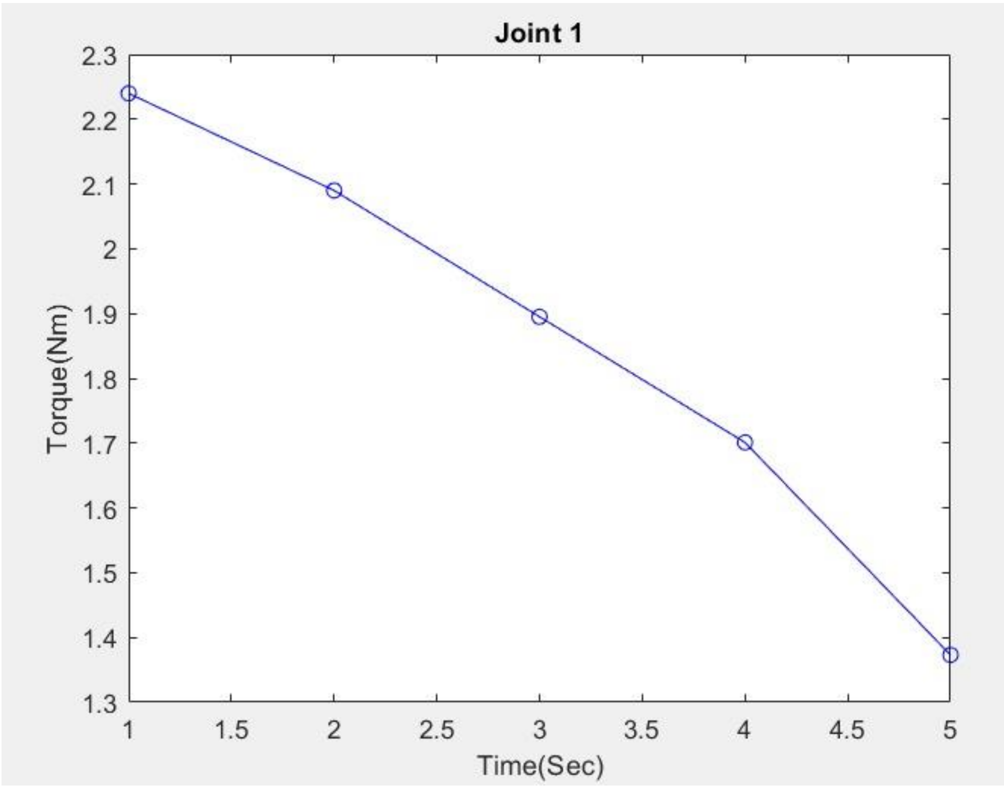

Fig. 4.2.2. *Torque graphic of Joint 1 from SIMULINK*

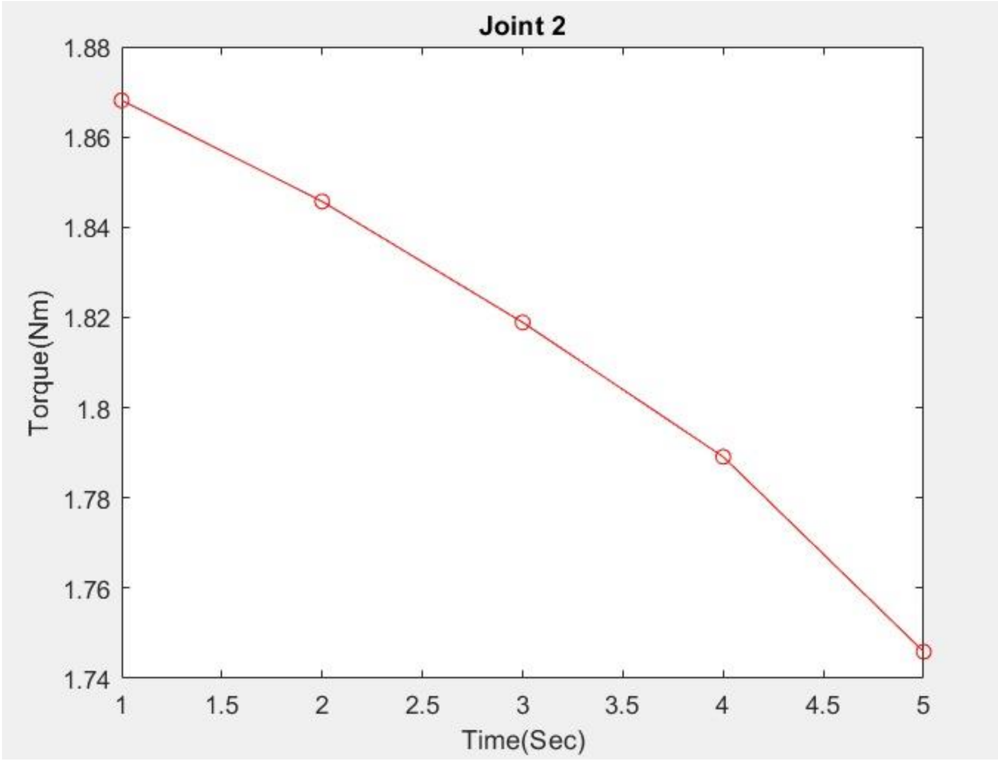

Fig. 4.2.3. *Torque graphic of Joint 2 from MATLAB.*

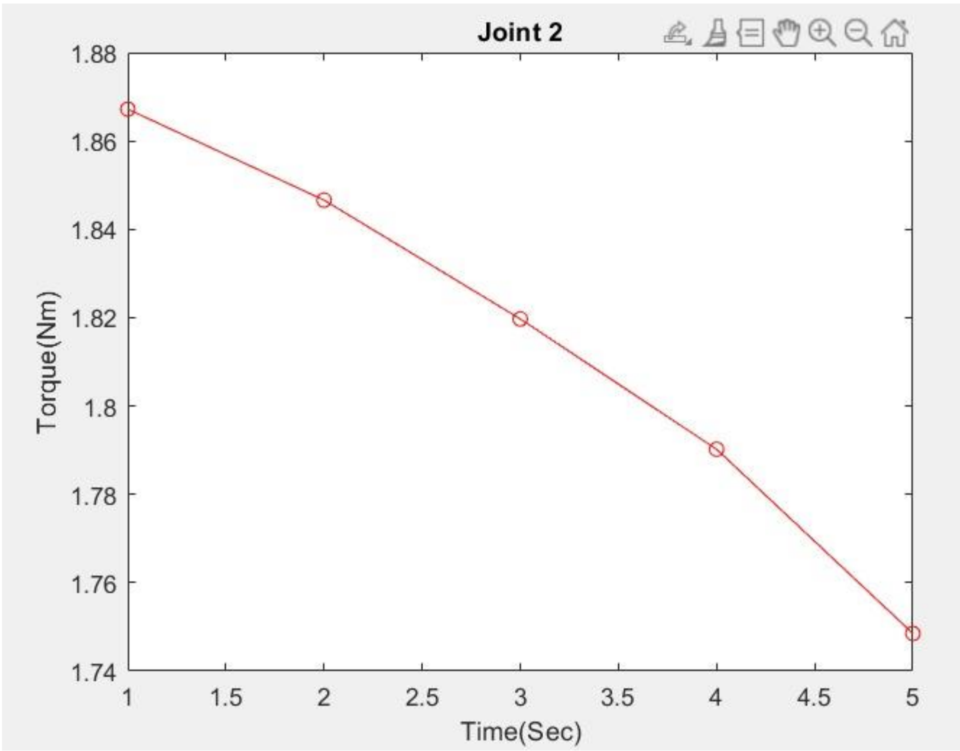

Fig. 4.2.4. *Torque graphic of Joint 2 from SIMULINK.*

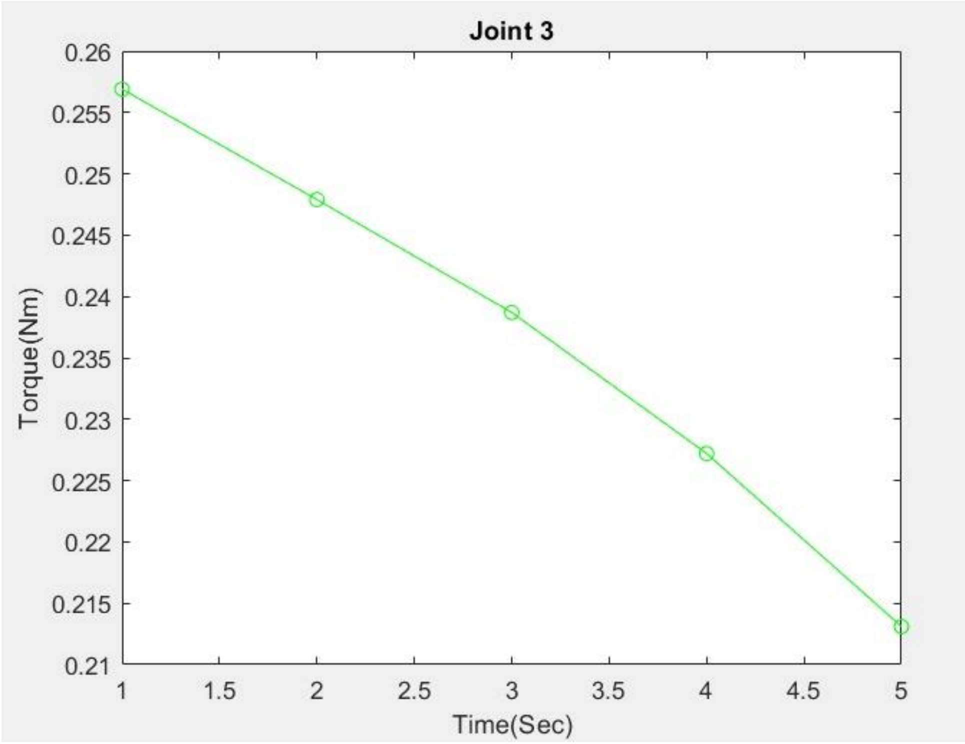

Fig. 4.2.5. *Torque graphic of Joint 3 from MATLAB.*

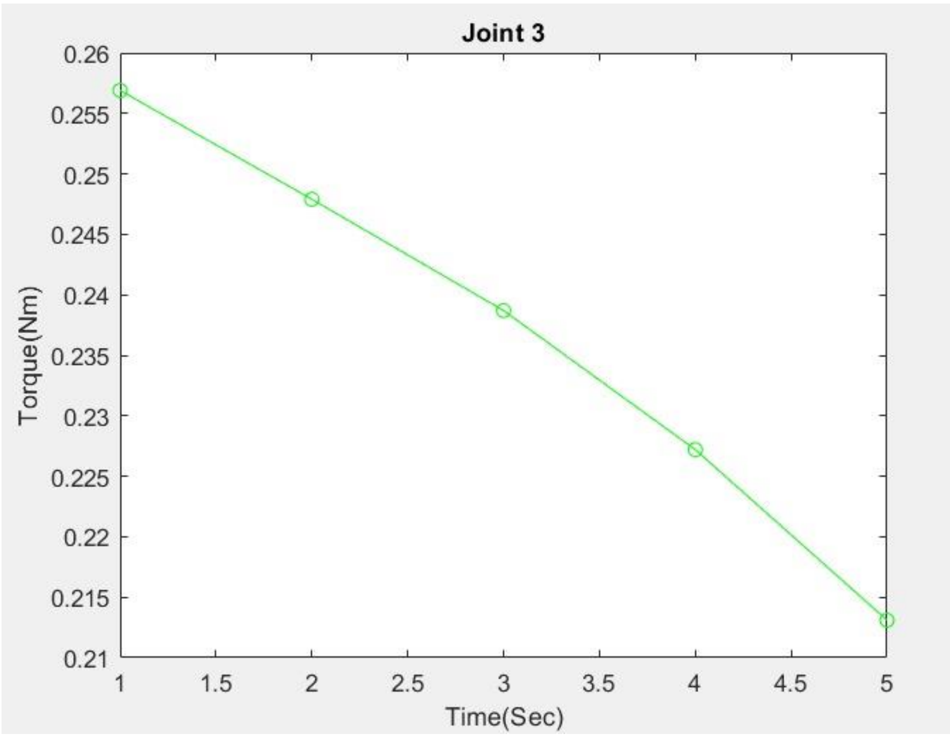

Fig. 4.2.6. *Torque graphic of Joint 3 from SIMULINK.*

The graphics of Joint 3 have also the same shape and values.

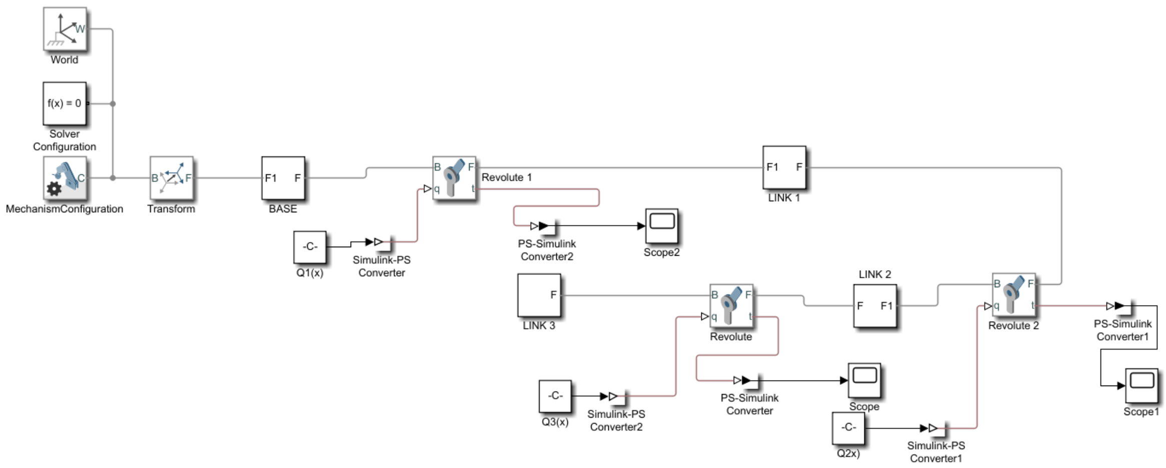

Fig. 4.2.7. *Block Diagram of 3-DOF arm*

4.3. Error Calculations

Here, we provide the error calculations of Matlab and Simulink values:

$$Error = \frac{Simulink - Matlab\ calculated}{Matlab\ calculated} x100$$

MATLAB Calculated Torque Value depending on time :

```
T1 = [1.8233 1.7917 1.7129 1.5590 1.3029];
T2 = [0.1480 0.1408 0.1228 0.0881 0.0337];
```

Simulink Torque Vale depending on time

```
T1 = [1.824  1.793  1.714  1.56    1.304];
T2 = [0.1481 0.1409 0.1229 0.08816 0.03374];
```

For Torque 1 time 1

$$Error = \frac{1.824 - 1.8233}{1.8233} x100 = 0.03839192673$$

For Torque 1 time 2

$$Error = \frac{1.793 - 1.7917}{1.7917} x100 = 0.07255678964$$

For Torque 1 time 3

$$Error = \frac{1.714 - 1.7129}{1.7129} x100 = 0.06421857668$$

For Torque 1 time 4

$$Error = \frac{1.56 - 1.5590}{1.5590} x100 = 0.06414368185$$

For Torque 1 time 5

$$Error = \frac{1.304 - 1.3029}{1.3029} x100 = 0.08442704736$$

For Torque 2 time 1

$$Error = \frac{0.1481 - 0.1480}{0.1480} x100 = 0.06756756757$$

For Torque 2 time 2

$$Error = \frac{0.1409 - 0.1408}{0.1408} x100 = 0.07102272727$$

For Torque 2 time 3

$$Error = \frac{0.1229 - 0.1228}{0.1228} x100 = 0.08143322476$$

For Torque 2 time 4

$$Error = \frac{0.8816 - 0.881}{0.881} x100 = 0.06810442679$$

For Torque 2 time 5

$$Error = \frac{0.3374 - 0.337}{0.337} x100 = 0.118694362$$

4.4. Conclusion

By the end, we come to the conclusion that in this project, we learned how to calculate forward kinematics, inverse kinematics and lagrange dynamics of serial robotic arms and analyze them in Matlab Simulink.

We wanted to make calculations using our previous drawing. However, our old drawing was parabolic, that`s why our calculations did not match. Hence, we drew a new robotic arm by using SolidWorks. In this new version of robotic arm, the results we obtained from forward and inverse kinematic and dynamic analysis were almost the same as the results we obtained in Matlab Simulink. We also observed that the values we calculated in forward kinematics and inverse kinematics were almost the same in our drawing in Solidworks.

When we make the error calculation, we observed that the error margin is less than 0.01%. So, at this point, we can claim that, this project was done successfully, since all the values are the same, and calculations were done very accurately.

REFERENCES

[1] Lung Wen Tsai, "Robot Analysis", New York, United states, John Wiley & Sons. Inc.

[2] Kinematic and Dynamic Analysis of Two Link Robot Arm using PID with Friction Compensator, Kaung Khant Ko Ko Han, Aung Myo Thant Sin, Theingi

[3] Performance measurement and dynamic analysis of two dof robotic arm manipulator Y. D. Patel1 P. M. George