MARMARA UNIVERSITY

FACULTY OF MECHANICAL ENGINEERING

Design of 6 DOF Robot by Analytical and Numerical Methods

Berk BABUCOGLU, Hüseyin OZ, Mehmet KILIC

GRADUATION PROJECT REPORT

Department Of Mechanical Engineering

Supervisor

Prof. Dr Bülent EKICI

ISTANBUL,2022

MARMARA UNIVERSITY FACULTY OF MECHANICAL ENGINEERING

Design of 6 DOF Robot by Analytical and Numerical Methods

by

Berk BABUCOĞLU

Hüseyin OZ

Mehmet KILIC

June, 2022, Istanbul

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERINGIN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF BACHELOR OF SCIENCE

AT MARMARA UNIVERSITY

Berk Babuçoğlu Hüseyin Oz Mehmet Kılıç

Signature of Author(s).....
Department of Mechanical Engineering

Certified By.....
Supervisor, Department of Mechanical Engineering

Accepted By.....
Head of the Department of Mechanical Engineering

ACKNOWLEDGEMENT

This report represents the end of our undergraduate life at Marmara University, Mechanical Engineering Department and beginning of our engineering life. This report and preperation process have contributed us to understand basics and soft skills of mechanical engineering and our future. We would like to Show our appreciation to those who helped us to complete our first and hopefully many achievements in this area.

First of all, we would like to thank our supervisor Prof. Dr. Bülent EKİCİ, who encouraged us and shared his experiences with us. We also thank Mr. EKİCİ for leading us through in and out of school experiences and guidances. We all appreciate to his in-depth knowledge.

We would like to Asst. Prof. Uğur TÜMERDEM, for his attitude and forgiving and understanding behavior. His vast patience and leading us with the help of his expertise about robotics was very significant for us.

We would like to thank Nural Yılmaz, for her kind attitude and leading us in lab, also for her patience.

We would like to thank all of our respected lecturers that we had the chance to be a participant of their classes.

Finally and most importantly, we are grateful for our families for their tireless supports and guidance for our long education journey.

Berk BABUÇOĞLU

Hüseyin ÖZ

Mehmet KILIÇ

June, 2022 - İstanbul

TABLE OF CONTENTS

ABSTRACT

Robots surely have a significant share of the rising technological market. Robots, which have many varied applications today, have long been a part of our lives. Various sub-headings are used to group robots into distinct categories based on their intended use and application. Studies on the TQ MA2000 robot arm with six degrees of freedom were conducted in this study.

The forward and inverse kinematics equations for the TQ MA2000 robot arm, as well as the transformation matrices, were created in this thesis. With the help of these kinematic computations, the position is intended to be controlled. In the Solidworks program, a robot arm was completely developed and illustrated. The same software created the appropriate material allocations. The model was built using Simulink Simmechanics using design data imported from the Solidworks application into the Matlab program. The torque values determined in Matlab on the points of freedom are tabulated with this modeling. The Ansys application used Mechanical Rigid Body Dynamics to simulate the design created in the drawing program. The correctness of the torque values generated by the Ansys software was examined through comparison with other measurements.

The transformation matrix was developed together with the determination of the robot manipulator's Denavit-Hartenberg parameters. The location of the terminal operator is tested by providing the robot manipulator with forward and inverse kinematics parameters. Case studies are used to support these investigations.

The automation lab at our institute will conduct further research on the robot arm after this study. Our project is intended to be improved upon by our pals who will work on this robot arm after us.

Keywords:TQ MA2000 Robot Arm, Transformation matrices, Rigidy Body Dynamics, Simulink Simmechanics

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1.History of Robot

The automation lab at our institute will conduct further research on the robot arm after this study. Our project is intended to be improved upon by our pals who will work on this robot arm after us.

Increasing competition in the industry in recent years has made it necessary to switch to automation in production. This As a natural result, robots, which are an indispensable element of automation, are of great importance. won. Robots that can do all kinds of work with small additions made on them, they started to take a role. With the increasing number of robots in factories, less energy the need for robots that consume more energy and require longer maintenance periods. started. On top of that, robot manufacturers reduce energy consumption and increase life expectancy in robots. They started to do some work for it. These studies were carried out in two parts. one is a robot improvements made in the control of the robot and the other in the robot construction. are improvements. Better trajectory as a result of improvements in robot control Efforts were made to reduce energy consumption through planning. Made in robot construction improvements include lightening the design, increasing the quality of the material used, and is the optimization of the internal mechanisms of the robot. Load capacity/weight with such improvements by increasing the rate of energy consumption is reduced. lightening the design, It is done by providing optimum conditions in the arms and trunk with FEM analysis on the computer. Optimization of the internal mechanisms of the robot means transferring with the minimum number of parts. passes.[7]

One of the most important issues that companies compete on today is aesthetics in design. Regardless of the function of the product, customers attach great importance to aesthetics. did Whatever the machine, companies that have aesthetic concerns and value aesthetics wins. The aesthetics of the robot are important. Design so that production is not forced aesthetic by removing the sharp corners from the design and softening them, making straight arms easy It can be more attractive for customers to make it curvilinear.

In today's business, robots are incredibly potent forces. the accuracy with which they can carry out these responsibilities in a variety of operations. and they do. They have a comfort zone and safety area when working, just like humans. They do not require it. But they use a lot of money in order to carry out their duties effectively.[4]

Automation and robotics may boost productivity, security, and quality. It can function in any hazardous environment without the requirement for comfort, safety, or life support. Robots work with light, ventilation, fresh air, and noise. They don't require environmental comforts like security. works more effectively than people can. Process sensitivity can reach a thousandth of a millimeter. Low pay, societal issues, employee unhappiness, and robots are all issues. They replaced people because they were trouble-free. However, unsuitable or incorrect responses, hasty judgment, loss of authority, mechanical or Disadvantages of devices include malfunctioning. They need programming, peripherals, and training. [4]

The pace of automation technology development is directly correlated with global economic growth. Robots used in industry as a specialized tool Computer-aided design (CAD) and computer-aided manufacturing (CAM) were first utilized in the 1960s, and in North America they were first employed in the early 1980s. Robotic devices are widely used. A little return to the middle of the 1980s Despite this, towards the end, the number of robotics firms began to rise. [2]

Robots are frequently employed to save costs and enhance product quality. They are utilized in the sectors, such as chemical and nuclear energy, both of which are harmful to human health.

Karel Capek invented the phrase "robot" in 1920.[7]

In 1955, Denavit and Hartenberg introduced transformation matrices. The industrialisation of robots can be seen as beginning with this circumstance. first commercial The robot in question is the 1961 "Unimate" model. [5]

After 1961, interest in robots increased and studies on this subject started. The SCARA robot was developed in 1979. from the 1990s The development has continued rapidly until today. [6]

1.2. Robots

When necessary, robots can carry out activities including transporting, welding, and assembling. They are devices that can be programmed to do different tasks.

1.2.1. Robot Components

The robot is basically studied under 4 parts.

1.2.2. Lınk Description

A manipulator may be thought of as a set of bodies connected in a chain by joints. These bodies are called links. Joints form a connection between a neighboring pair of links. The term lower pair is used to describe the connection between a pair of bodies when the relative motion is characterized by two surfaces sliding over one another. shows the six possible lower pair joints [2]

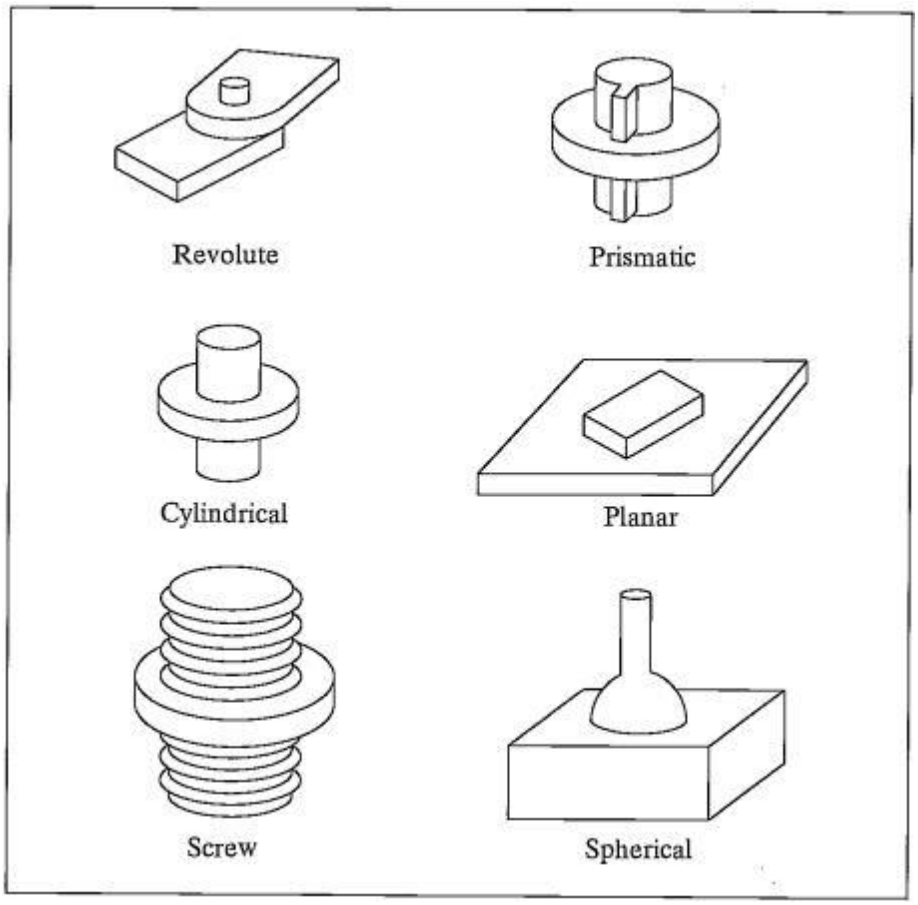


Figure 1: The six possible lower-pair joints. [1]

1.2.3. Link Connection Description

The problem of connecting the links of a robot together is again one filled with many questions for the mechanical designer to resolve. These include the strength of the joint, its lubrication, and the bearing and gearing mounting. However, for the investigation of kinematics, we need only worry about two quantities, which wifi completely specify the way in which links are connected together [2]

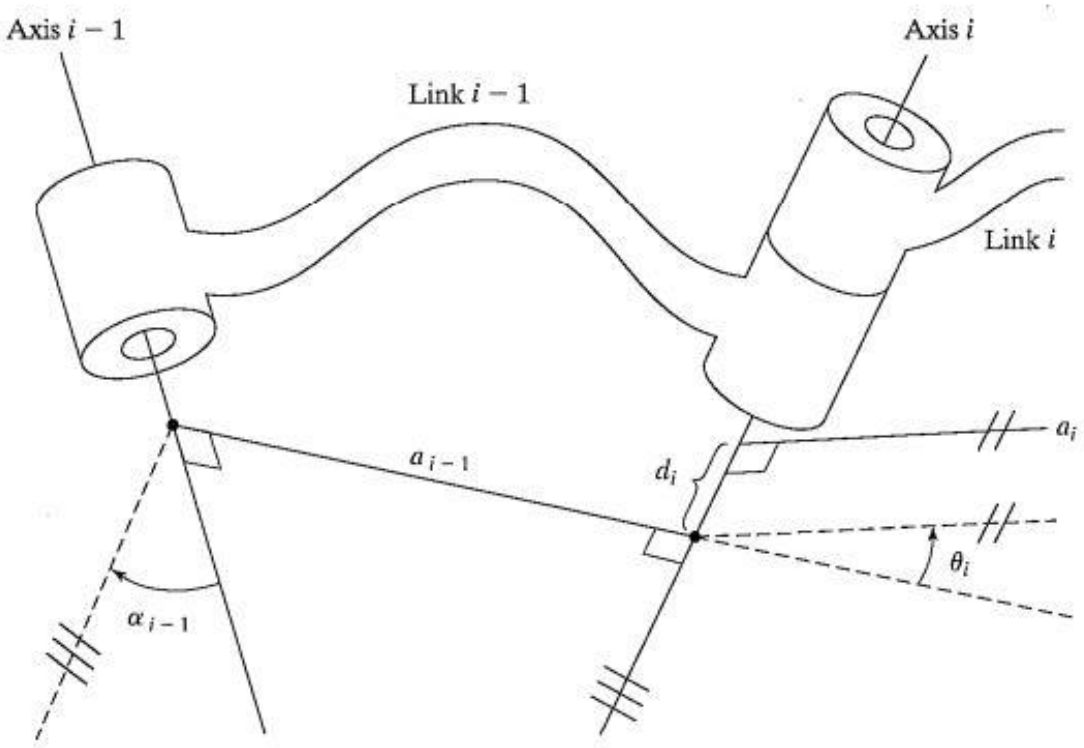


Figure 2 The link offset, d, and the joint angle, 9, are two parameters that may be used to describe the nature of the connection between neighboring links. [2]

Neighboring links have a common joint axis between them. One parameter of interconnection has to do with the distance along this common axis from one link to the next. This parameter is called the link offset. The offset at joint axis i is called d_i The second parameter describes the amount of rotation about this common axis between one link and its neighbor. This is called the joint angle, θ_i.[2]

1.2.4. Denavit-Hartenberg Demonstration

Step 1: The joints manipulators z axes are established. The axis of movement in prismatic axes is assumed to be z in rotary manipulators.

Step 2: The central coordinate axis is determined. Express the arm lengths of the robot on the z_0 axis. A zeroth coordinate system is placed at any location that can. According to the right hand rule y_0 and x_0 is written.

Step 3: Origin o_i , is placed at the intersection of z_i and z_{i-1}. If z_i and z_{i-1} are parallel then o_i origin z_i It is placed anywhere along its axis where arm lengths can be expressed.

Step 4: The a's are placed perpendicular to z_i and z_{i-1}.

Step 5: According to the right hand rule y_i are placed.

Step 6: The end-effector coordinate system is specified as $o_n\ x_n\ y_n\ z_n$.

Step 7: The DH table is extracted according to the following parameters:

r_i= Length along x_i from o_i to the intersection of the x_i and z_{i-1} axes.

d_i= It is the length along z_{i-1} from o_{i-1} to the point where the x_i and z_{i-1} axes intersect.

a_i= It is the angle measured along x_i between z_i and z_{i-1}.

Q_i= it is the angle measured along z_{i-1} between x_i and x_{i-1}.

Step 8: According to the parameters found, the DH table is created as shown below.

$$T_i = \begin{pmatrix} \cos(\theta_i) & -\sin(\theta_i)*\cos(\alpha_i) & \sin(\theta_i)*\sin(\alpha_i) & r_i*\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i)*\cos(\alpha_i) & -\cos(\theta_i)*\sin(\alpha_i) & r_i*\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Step 9: The desired transformation matrix is created as $T_i^0 = T_1 \dots T_i$.

1.3. Ansys Rigid Body Dynamics

We discussed solving dynamics problem by implicit and explicit methods to obtain the response of structures under dynamic loads. The outputs included deformation, stress, strain, etc. For large-scale and long-duration dynamics problems, the computation cost to solve them are significant. In some cases, when deformation of material is not the main concern, but the rigid motion is the response engineers are looking for, rigid dynamics is the right path. This applies when:

- Objects are very stiff.
- Deformation of material points is not important.
- You are dealing with a large-scale, complex system.
- The structural stiffness K is removed from the equation of motion.
- Different rigid parts are linked by joints/contacts.

1.3.1. Rigid Body Dynamics: Degrees of Freedom (DOFs)

Unlike dynamics for flexible bodies which looks for deformation at each material point, rigid body dynamics looks for the rigid motion of the system. Especially, it solves for relative displacement between different parts.

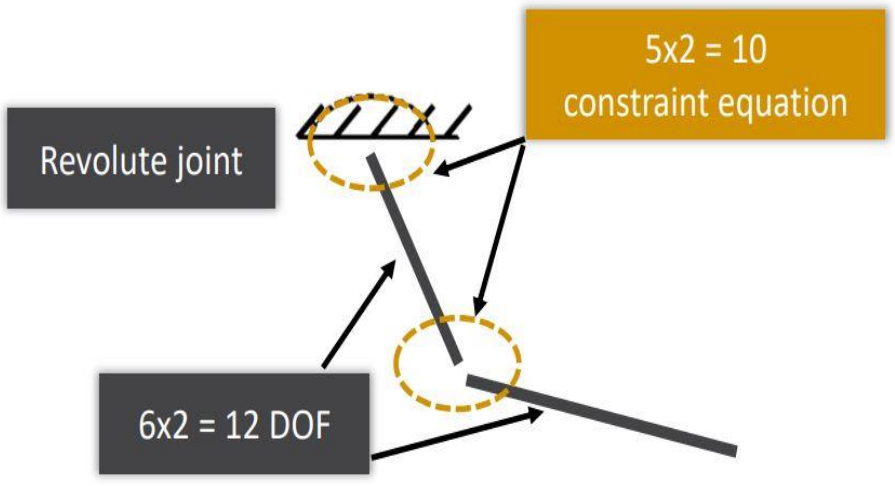


Figure 3 Degrees of Freedom (DOFs) [3]

- Because parts are rigid, relative DOFs are sufficient to describe the mechanism.
- The number of DOFs is largely reduced if using relative DOFs.
- One rigid body has at most 6 degrees of freedom in 3D space.

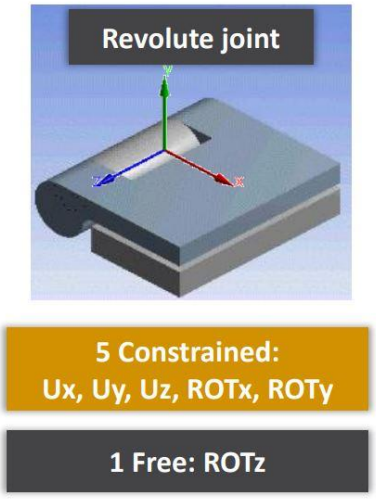


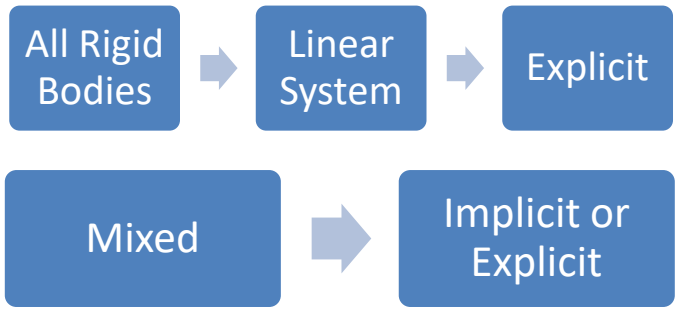
Figure 4 Revolute joint [3]

1.3.2. Rigid Body Dynamics: Time Integration

Usually, a dynamics problem that can be modeled as rigid bodies is not a short-duration problem. However, since structure stiffness is removed, the explicit method is used for most rigid body dynamics because of its simplicity.

In cases where one or more parts of an assembly is more flexible while the others are very stiff. It is wise to combine flexible bodies and rigid bodies in one analysis. In such cases, usually, the implicit method is needed.

- Explicit time integration is used for a pure rigid body assembly.
- Implicit or explicit time integration can be used for mixed rigid and flexible bodies.



1.3.3. Rigid Body Dynamics: Mixed Rigid and Flexible Bodies

Solve the normal assembly by different time integration methods.

- All parts rigid – using explicit time integration
- One part flexible – using implicit time integration
- All parts flexible – using implicit time integration

2. FORWARD AND INVERSE KINEMATICS ANALYSIS

2.1 Introduciton

Depending on their geometrical dimensions, kinematics studies the position, velocity, and acceleration of components of mechanical systems. Robotic arms are open-chain systems consisting of limbs connected at the joints. Robot kinematics is concerned with deriving the relationships between the change in the position, velocity, and acceleration of the robot arm's end point and the joint variables' position, velocity, and acceleration. The relationships between the joint coordinate sets that are placed on each joint based on a fundamental set of coordinates known as the master coordinate set are subtracted to produce these relationships.

Each designed robot must undergo kinematic analysis. Robot kinematic analysis is classified into two categories: forward kinematics and inverse kinematics.

Researchers use a variety of kinematic analysis approaches to simulate systems. T the primary kinematic analysis techniques are simply outlined below.

- Geometric-based straight kinematic analysis method.
- Homogeneous transformation matrix method
- Denavit-Hartenberg method
- Quaternion and the rotating vector representation method*[10]*

The Denavit-Hartenberg method is appropriate for the analysis of multi-degree-of-freedom mechanisms, particularly the dynamic analysis of robot arms and inverse kinematics methods. This approach was applied in this study as well.

2.2 Denavit-Hartenberg Method

A general arm equation that represents the kinematic motion of the manipulator can be obtained by systematically assigning coordinate frames for each link. The parameters associated with joint k are defined with respect to z_{k-1}, which is aligned with the axis of joint k. The first joint parameter,θ_k is called the joint angle. It is the rotation about z_{k-1} needed to make x_{k-1} parallel to x_k. The second joint parameter, d_k is called the joint distance. It is the translation along z_{k-1} needed to make x_{k-1} intersect with x_k see Figure 5. Note that for each joint, it will always be the case that one of these parameters is fixed and the other is variable. The variable joint parameter depends on the type of joint.[10]

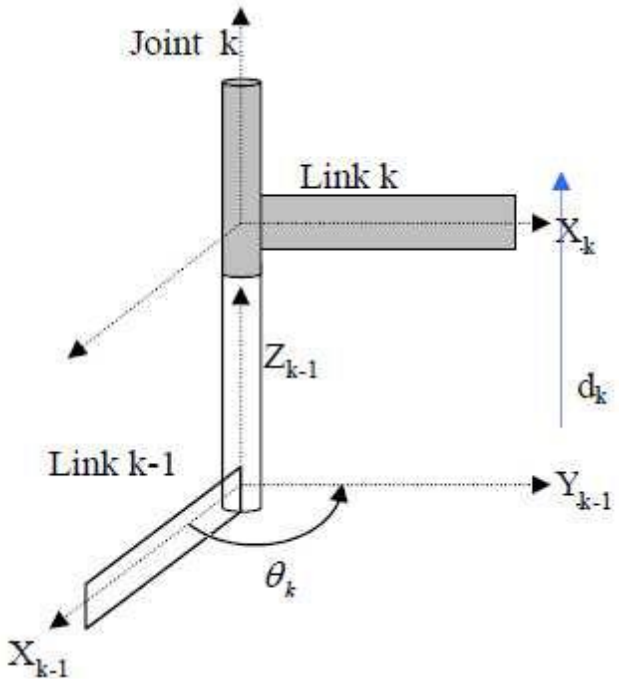


Figure 5 Joint angle θ and joint distance d [10]

For instance, for a revolute joint, the joint angle θ_k is variable while the joint distance d_k is fixed while for a prismatic joint, the joint distance d_k is variable and the joint angle θ_k is fixed shown in Figure 6. [10]

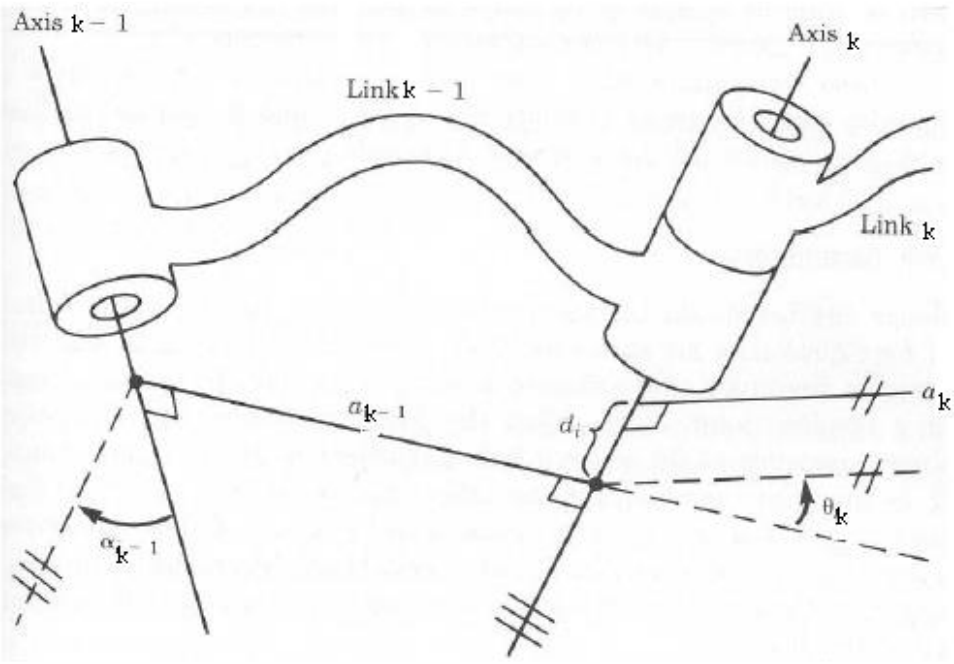


Figure 6 DH parameters defined for two links of a serial manipulator [10]

2.3 Forward Kinematic Analysis

In a robot manipulator, if the joint angles and link lengths are known, forward kinematics is the process of determining the location and orientation of the end functionalist. [10]

From its main structure to its end function, a robot is made up of a succession of links that are joined together by rotary or prismatic joints. All joints are given a coordinate system throughout the kinematic analysis, beginning with the main frame. The transformation matrix T expresses the connection between two nearby joints. Using this method, the transformation matrices between all ordered joints are obtained, and by successively multiplying these matrices, the position and orientation relationship between the main frame of the robot and the end functionalist is discovered. [10]

These industrial robots are mostly made of rigid links that are joined in a series by joints, with one end fixed (the base) and the other free to move and perform useful tasks when end-effectors are used properly. Robot manipulators, like human arms, use the first three joints (arm)

to position the structure and the remaining joints (in the case of industrial manipulators, the wrist, which has three joints) to orient the end-effecter. There are five different arm types that industrial robot manipulators often employ: Cartesian, cylindrical, polar, SCARA and revolution. The TQ MA2000 is considered in this work, it has 6 degree of freedom (DOF), one per each joint which means that each joint can be represented by a independent variable θ_n since every joint is activated, i.e. in this Each servo motor functioning on each link is independent. An open kinematic chain called a robot manipulator can have joints whose locations can be specified by a vector of six single variables θ_n and the number of joints and degrees of freedom are equal. Recall that the quantity of degrees of freedom (DOF) indicates the quantity of independent position variables that must be supplied in order to fully characterize a manipulator pose. It specifies the variety of motions the robot can make.. In practice, six degrees of freedom are enough to cover the full 3D workspace with a 3-component orientation vector. Table below shows the DH parameters whole set of Kinematic parameters for the TQ MA2000 Robot. The graphic representation of the TQ MA2000 robot is shown in Figure. [10]

Table 1 Complete DH parameter table fort he TQ MA 2000 6- DOF Manupilator

Link i	Length a_i (cm)	Twist Angle α_i	Offset d_i (cm)	Joint Angles θ_n
1	0	$\pi/2$	26	θ_1
2	23	0	0	θ_2
3	24	0	0	θ_3
4	0	$\pi/2$	5	θ_4
5	0	$\pi/2$	4.4	θ_5
6	0	0	8	θ_6

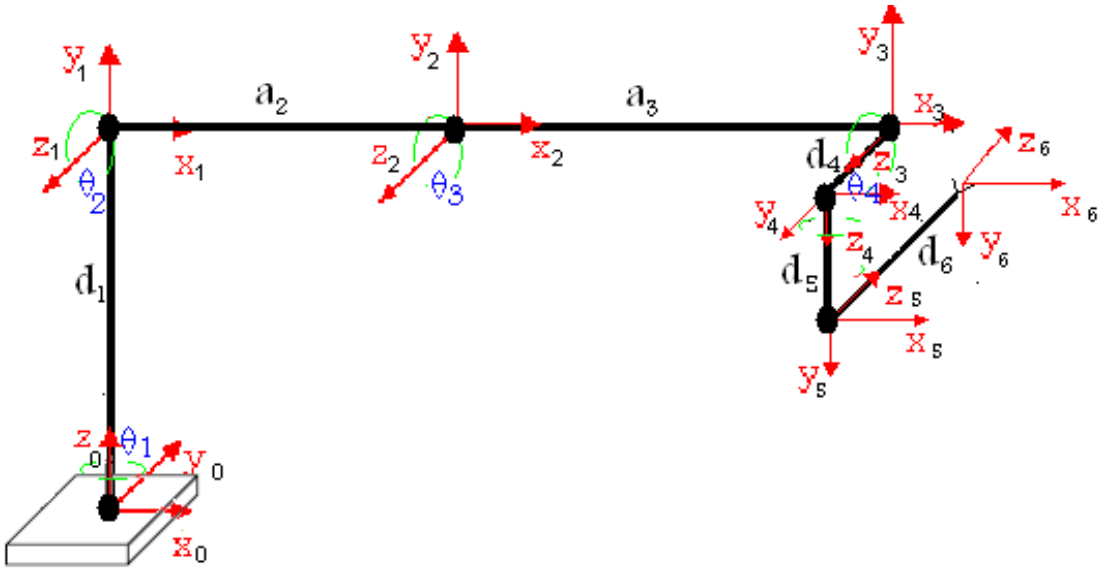


Figure 7 Graphic Represent of TQ MA2000 Robot *[10]*

The transformation matrix shown below can be used to represent the arm motion:

$$Motion = T_{base}^{end}(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$$

Where T_i^j represents the transformation matrix form i to j.

The Transformation Matix represents as:

$$T_{01} = \begin{pmatrix} \cos(\theta_1) & -\sin(\theta_1)*\cos(\alpha_1) & \sin(\theta_1)*\sin(\alpha_1) & a_1*\cos(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1)*\cos(\alpha_1) & -\cos(\theta_1)*\sin(\alpha_1) & a_1*\sin(\theta_1) \\ 0 & \sin(\alpha_1) & \cos(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{12} = \begin{pmatrix} \cos(\theta_2) & -\sin(\theta_2)*\cos(\alpha_2) & \sin(\theta_2)*\sin(\alpha_2) & a_2*\cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2)*\cos(\alpha_2) & -\cos(\theta_2)*\sin(\alpha_2) & a_2*\sin(\theta_2) \\ 0 & \sin(\alpha_2) & \cos(\alpha_2) & d_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{23} = \begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3)*\cos(\alpha_3) & \sin(\theta_3)*\sin(\alpha_3) & a_3*\cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3)*\cos(\alpha_3) & -\cos(\theta_3)*\sin(\alpha_3) & a_3*\sin(\theta_3) \\ 0 & \sin(\alpha_3) & \cos(\alpha_3) & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{34} = \begin{pmatrix} \cos(\theta_4) & -\sin(\theta_4)*\cos(\alpha_4) & \sin(\theta_4)*\sin(\alpha_4) & a_4*\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4)*\cos(\alpha_4) & -\cos(\theta_4)*\sin(\alpha_4) & a_4*\sin(\theta_4) \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{45} = \begin{pmatrix} \cos(\theta_5) & -\sin(\theta_5)*\cos(\alpha_5) & \sin(\theta_5)*\sin(\alpha_5) & a_5*\cos(\theta_5) \\ \sin(\theta_5) & \cos(\theta_5)*\cos(\alpha_5) & -\cos(\theta_5)*\sin(\alpha_5) & a_5*\sin(\theta_5) \\ 0 & \sin(\alpha_5) & \cos(\alpha_5) & d_5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{56} = \begin{pmatrix} \cos(\theta_6) & -\sin(\theta_6)*\cos(\alpha_6) & \sin(\theta_6)*\sin(\alpha_6) & a_6*\cos(\theta_6) \\ \sin(\theta_6) & \cos(\theta_6)*\cos(\alpha_6) & -\cos(\theta_6)*\sin(\alpha_6) & a_6*\sin(\theta_6) \\ 0 & \sin(\alpha_6) & \cos(\alpha_6) & d_6 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{06} = T_{01} * T_{12} * T_{23} * T_{34} * T_{45} * T_{56}$$

To evlauate faster and efficient, we used MATLAB to calculate desired equations, matrixes and values.

Note: With respect to Denavit-Hartenberg table, all α values placed in the codes to show the calculations efficient and more understandable.

```
T01 =

[cos(teta1), 0,  sin(teta1),  0]
[sin(teta1), 0, -cos(teta1),  0]
[         0, 1,           0, d1]
[         0, 0,           0,  1]


T12 =

[cos(teta2), -sin(teta2), 0, a2*cos(teta2)]
[sin(teta2),  cos(teta2), 0, a2*sin(teta2)]
[         0,          0, 1,             0]
[         0,          0, 0,             1]


T23 =

[cos(teta3), -sin(teta3), 0, a3*cos(teta3)]
[sin(teta3),  cos(teta3), 0, a3*sin(teta3)]
[         0,          0, 1,             0]
[         0,          0, 0,             1]
```

```
T34 =

[cos(teta4), 0,  sin(teta4),  0]
[sin(teta4), 0, -cos(teta4),  0]
[         0, 1,           0, d4]
[         0, 0,           0,  1]


T45 =

[cos(teta5), 0,  sin(teta5),  0]
[sin(teta5), 0, -cos(teta5),  0]
[         0, 1,           0, d5]
[         0, 0,           0,  1]


T56 =

[cos(teta6), -sin(teta6), 0,  0]
[sin(teta6),  cos(teta6), 0,  0]
[         0,           0, 1, d6]
[         0,           0, 0,  1]


T06=T01*T12*T23*T34*T45*T56
```

2.4 Inverse Kinematic Analysis

With the aid of end-function position and orientation information provided in accordance with the robot's main frame, inverse kinematics analysis determines joint variables in cartesian space. That is, the limb variables d and θ must be found in order for the robot to reach a desired point. It is possible to describe it as the opposite of forward kinematic analysis. Inverse kinematic analysis can be performed using both analytical and numerical techniques. It can take a very long time to compute numerical techniques using a computer. This occurs as a result of attempts to find the solution iteratively. On the other hand, analytical approaches provide quick and accurate answers but cannot be used with all kinds of robots.

Inverse kinematics is one of the most fundamental and persistent issues in robotics and computer animation (IK). Find the set of all configurations (joint angle vectors) of the robot that

satisfy the forward kinematics map, given a desired hand position and orientation (posture) and the forward kinematics map. The IK mapping is typically one to many, involves complicated inverse trigonometric functions, and lacks closed form solutions for the majority of manipulators. For real-time applications in quickly moving manipulators, computer animation requires incredibly rapid IK computation. Finding the necessary joint angles to generate a specific desired position and orientation of the end-effecter is an issue that is addressed by inverse kinematics. It might be very difficult to find the inverse kinematics solution for a general manipulator. They are typically non-linear equations. Multiple, infinite, or impossible solutions may occur instead of close-form solutions. Special instances, however, can be resolved and have a closed-form solution.

2.4.1 Algorithm for Inverse Kinematics

The suggested algorithm is immediately applicable to robots with offset shoulders and elbows. However, wrist offset affects more than just the first four joint angles; it also affects the hand position. This calls for simultaneously addressing position and orientation and necessitates the establishment of a 6-D workspace, which although technically feasible would significantly increase the amount of off-line processing and storage needs. It's common to think of manipulators as having two pieces. The wrist is positioned in space by the first three joints, which together make up a regional structure. The orienting structure, which is made up of the final three joints, is used to orient the hand or the object being held.

2.4.2 Decompose the Manipulator at the Wrist

The procedure's most crucial phase is this particular one. This is made feasible by the spherical wrist. Locate the wrist point so that you may use it to solve for the first three joint angles. After that, the wrist joint angles are solved to provide the proper orientation. *[10]*

Figure (Graphic Represent of TQ MA2000 Robot) shows the robot at reset position (all the joint angles equal zero), The suggested approach begins by identifying the necessary position in (x, y, z) coordinate and orientation angles of pitch, yaw, and roll $(\varphi_p, \varphi_y, \varphi_r)$ angles respectively. Then change the wrist joint angles $(\theta_4, \theta_5, \theta_6)$ from the zero (reset state) to replacement values that match the necessary orientation angles of pitch, yaw, and roll $(\varphi_p, \varphi_y, \varphi_r)$ angles respectively, the arm joint angles remain unchanged (at reset state, i.e. equal zero), then compute the foreword kinematics to compute the resulted position of the arm end point (O_4) and position of end-effecter (O_6) for the robot, find the vector between these two points as in equation below. *[10]*

$$V = O_6 - O_4$$

Use equation above to determine where the arm should finish up for it to achieve the desired position, with that the vector from the origin point to the required position is equal to the vector from origin base frame to the arm end point position sum with the vector V which is from to the required position as in equation below as shown in Figure. *[10]*

$$V_{04} = V_{0r} - V$$

Where V is the vector computed by equation for V, and V_{0r} is the vector from the origin O_0 to the required position. The position of the arm end point O_4 is the end of the vector V04. A planar two link manipulator often makes up the last two links of the regional structure. *[10]*

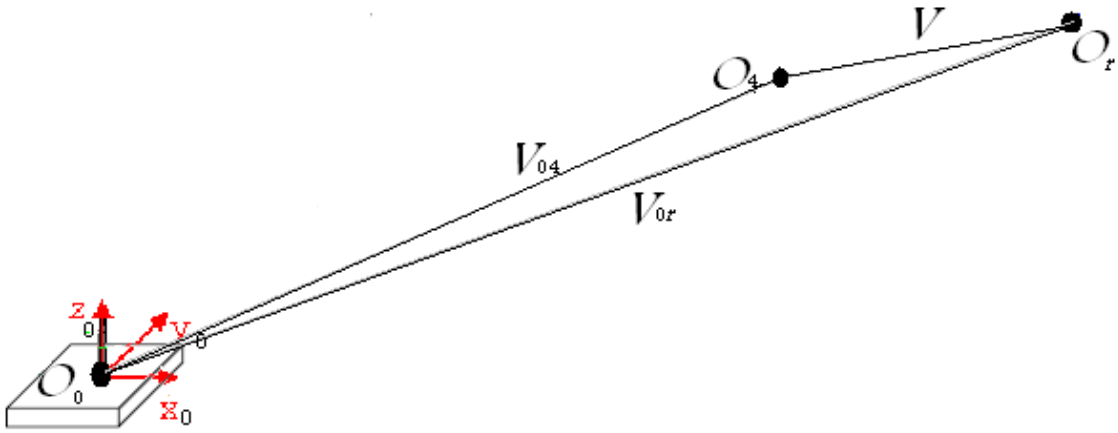


Figure 8 Wrist with setting its angles [10]

The link 4 parameters have been included in the regional structure because they locate the wrist point O_4, which will be known relative to O_0. Equation of V_{04}, as shown in figure below. *[10]*

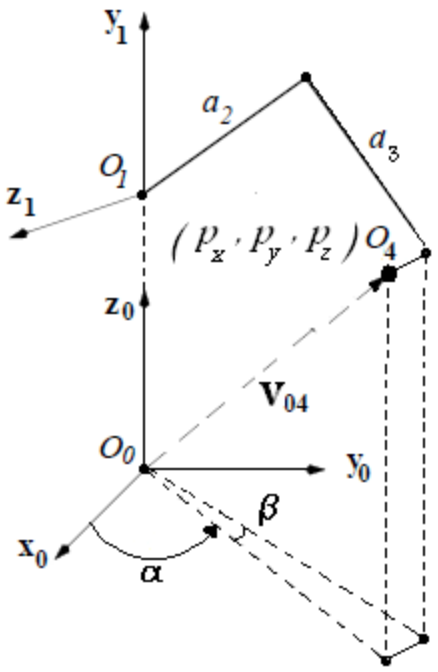


Figure 9 Vector from O_0 to O_4[10]

Project the position of the wrist point into the x_0, y_0 plane, i.e., just select the p_x, p_y coordinates of O_4

Solution for θ_1

The wrist offset makes the calculation of θ_1 more complex, because the x_1 axis is no longer in the vertical plane of the upper arm and forearm Figure. When x_1 and the wrist point are projected onto the x_0, y_0 plane, then the angle θ_1 from x_0 to x_1 is no longer the same as the angle from x_0 to the projected wrist point. The projection on the x_0, y_0 plane is redrawn in Figure for clarity. The position of the arm end point at (p_x, p_y, p_z) see Fig.5. Then compute α and β as in equations*[10]*

$$\alpha = a\tan 2(p_y, p_x)$$
$$\beta = a\sin(d_4 / L)$$

Where $L = \sqrt{p_x{}^2 + p_y{}^2}$, and hence for right move when θ_1 will be as follow

$$\theta_1 = \alpha + \beta$$

And for left move

$$\theta_1 = (\alpha - \beta) - 180$$

Solution for θ_2 and θ_3

For each solution of θ_2 find two set of solution for θ_2 and θ_3, elbow up and elbow down, therefore will compute four set of solution of the first three angles that is (left, right, elbow up, elbow down).

$$\theta = a\tan(pz_0 / r_0)$$

Where $pz_0 = p_z - d_1$ and $r_0 = \sqrt{p_x{}^2 + p_y{}^2}$

The cosine law for link 2 and link 3:

$$\alpha_2 = \pm a\cos\left(\frac{r^2 + a_2{}^2 - a_3{}^2}{2 * r * a_2}\right)$$
$$\alpha_3 = \pm a\cos\left(\frac{r^2 + a_3{}^2 - a_2{}^2}{2 * r * a_3}\right)$$

Where, $r = \sqrt{p_x{}^2 + p_y{}^2 + p_{z_0}{}^2}$

To compute θ_2 and θ_3

$$\theta_2 = \theta - \alpha_2$$
$$\theta_3 = \alpha_2 + \alpha_3$$

The Solution of the Last Three Joints $\theta_4, \theta_5, \theta_6$

The following equations can be derived from the forward kinematics problem.

$$T_0^6 = T_0^3 * T_3^6$$
$$T_0^6 = \begin{pmatrix} R_0^3 & P_a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R_3^6 & P_w \\ 0 & 1 \end{pmatrix}$$
$$T_0^6 = \begin{pmatrix} R_0^3 R_3^6 & R_0^3 * P_w + P_a \\ 0 & 1 \end{pmatrix}$$
$$R = R_0^3 * R_3^6$$
$$P = R_0^3 * P_w + P_a$$
$$R_3^6 = (R_0^3)^{-1} * R$$

Since the calculation numerically is too much to apply, we used MATLAB to get results faster and reliable.

```
T06=vpa(T01*T12*T23*T34*T45*T56,3)

T36=vpa(T34*T45*T56,3)

R36=vpa(T36(1:3,1:3),3)
```

$$R_3^6 = \begin{pmatrix} C_4C_5C_6 + S_4S_6 & S_4C_6 - C_4C_5S_6 & C_4S_5 \\ S_4C_5C_6 - C_4S_6 & C_4C_6 - S_4C_5S_6 & S_4S_5 \\ S_5C_6 & S_5S_6 & C_5 \end{pmatrix}$$ where

$$R_3^6 = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

$$\theta_4 = \frac{r_{23}}{r_{13}} ,$$

$$\theta_5 = a\tan 2(r_{33}, \pm\sqrt{{r_{13}}^2 + {r_{23}}^2}) ,$$

$$\theta_6 = a\tan 2(r_{31}, -r_{32})$$

2.5 Case Study

The MA2000 robotic manipulator, a 6DOF manipulator with a complicated wrist construction, is used to evaluate the suggested method practically.

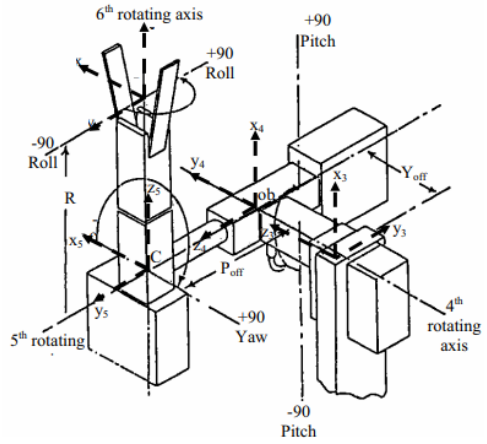


Figure 10 Schematic of the complex wrist [12]

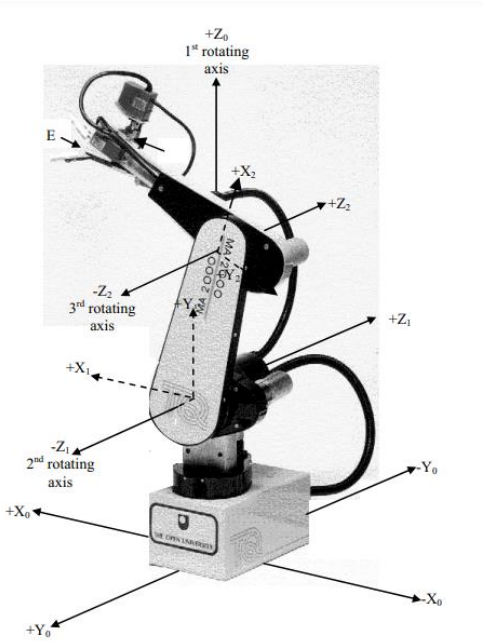


Figure 11 MA2000 with the major coordinate frames [12]

The schematic diagram of the MA2000 at the reset position is shown in the Figure 1. If it is required to move the gripper to a new position [-25,-10,50] cm with respect to the base (global) coordinate frame. The desired orientation of the gripper can be obtained by rotates the gripper 180 degrees about the pitch rotating axis, 225 degrees about the rotated yaw rotating axis and finally 135 degrees about the rotated roll rotating axis. The problem is to determine the values of the six joint variables of the MA2000 that attain the central point of the gripper to the desired position with the desired orientation. By applying the steps in previous section, four possible sets of solutions of the inverse kinematics of the MA2000 manipulator can be found. Thus there are four sets of the joints variables [θ_1, θ_2, θ_3,θ_4,θ_5,θ_6] that make the MA2000 attains the desired position with the desired orientation which are given in table. The sketch of MA2000 with these joints variables (solutions) are shown in figures.

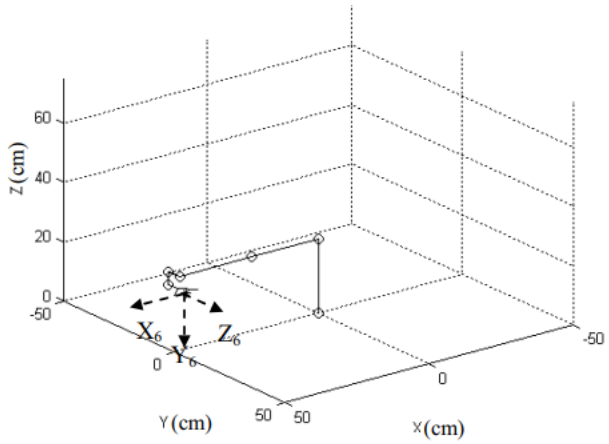


Figure 12 MA2000 at its reset position

Solutions:

Table 2 Solutions for case study

	θ_1(deg.)	θ_2(deg.)	θ_3(deg.)	θ_4(deg.)	θ_5(deg.)	θ_6(deg.)
1st set	197.35	-7.89	79.77	108.11	27.64	135
2nd set	197.35	75.87	-79.77	183.90	27.64	135
3rd set	-1.22	187.89	-79.77	71.88	226.22	135
4th set	-1.22	104.12	79.77	356.09	226.22	135

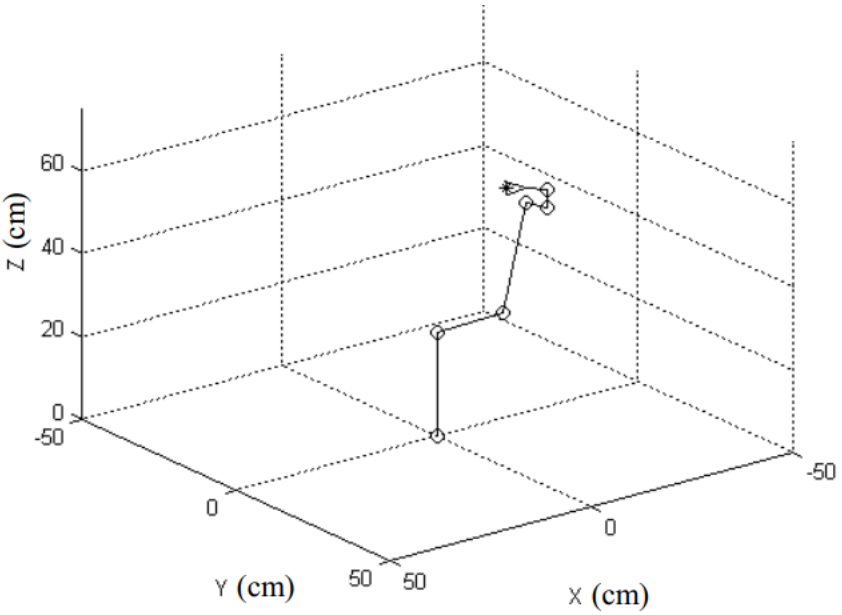


Figure 13 First Set

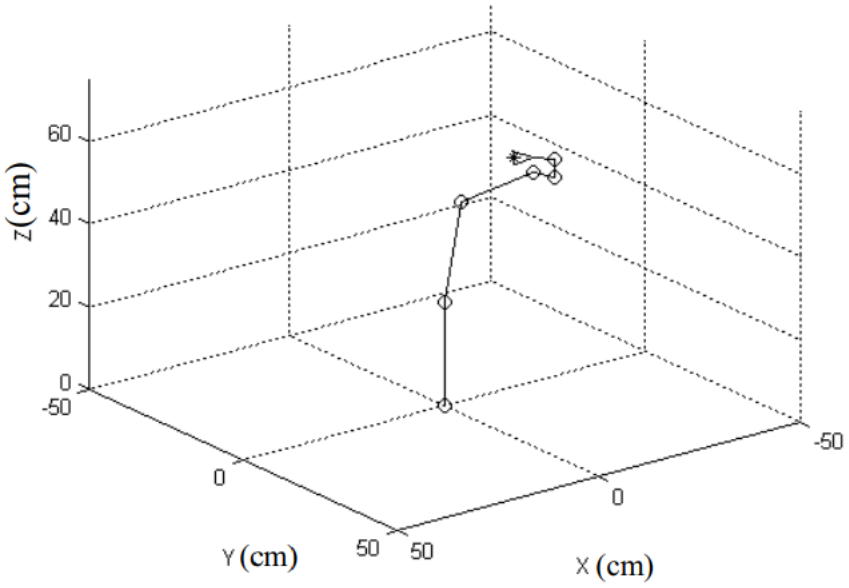


Figure 14 Second Set

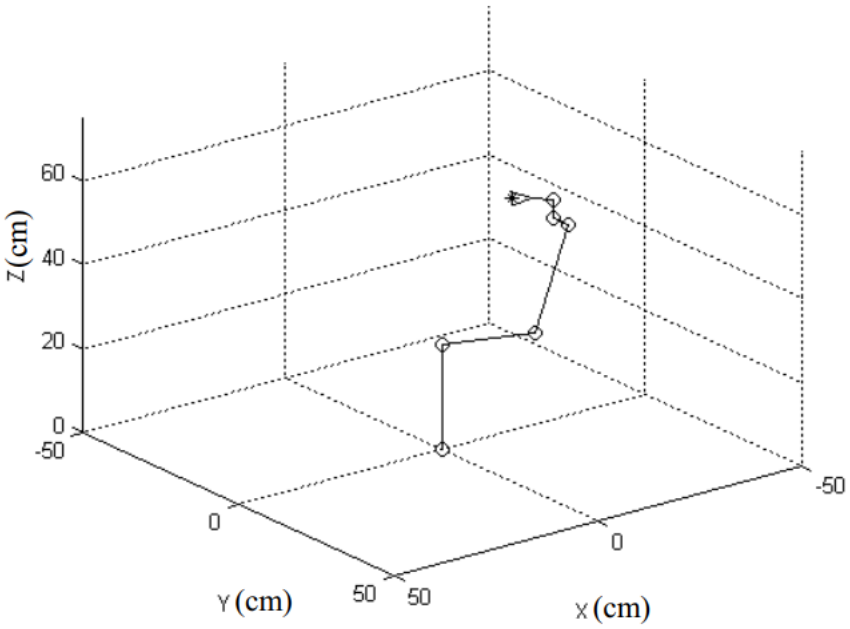


Figure 15 Third Set

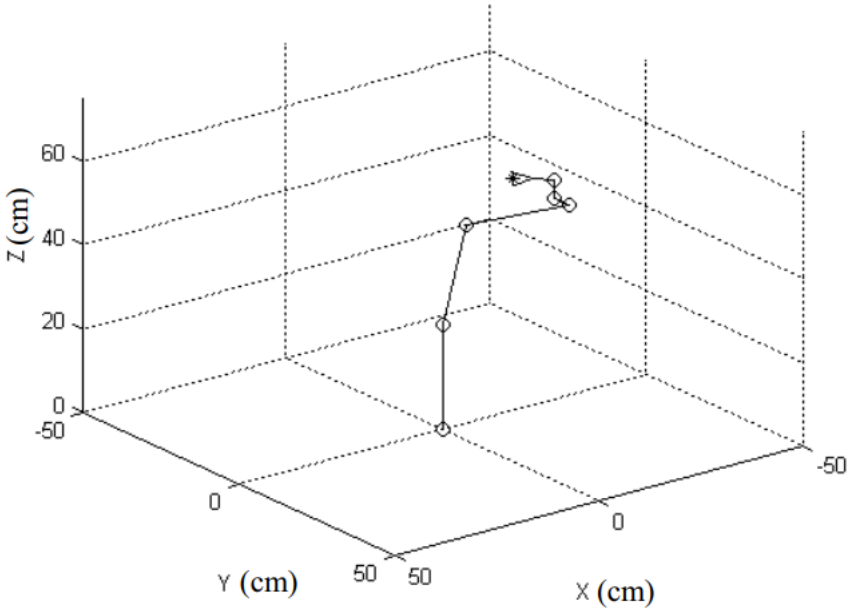


Figure 16 Fourth Set

3. SOLIDWORKS DRAWING AND DESIGN

Subject of our final project " Design of 6 DOF Robot By Analytical and Numeric Methods ". We continue this work on the TQ MA2000 model robot in the automation lab of our school.

Primarily, due to the fact that we could not access the robot's catalog, we completed the drawing on solidworks 2018 by taking the dimensions of the robot arm with the help of caliper and other measuring instruments.After making the drawing, we took care of that the measurements were taken with a high accuracy in order to obtain the minimum margin of error when taking the moment and torque values at the joint points.As can be seen from the figures, the actual image of the robot arm and the solidworks drawing are available.

Figure 17 TQ Ma2000 Robot Arm Real View

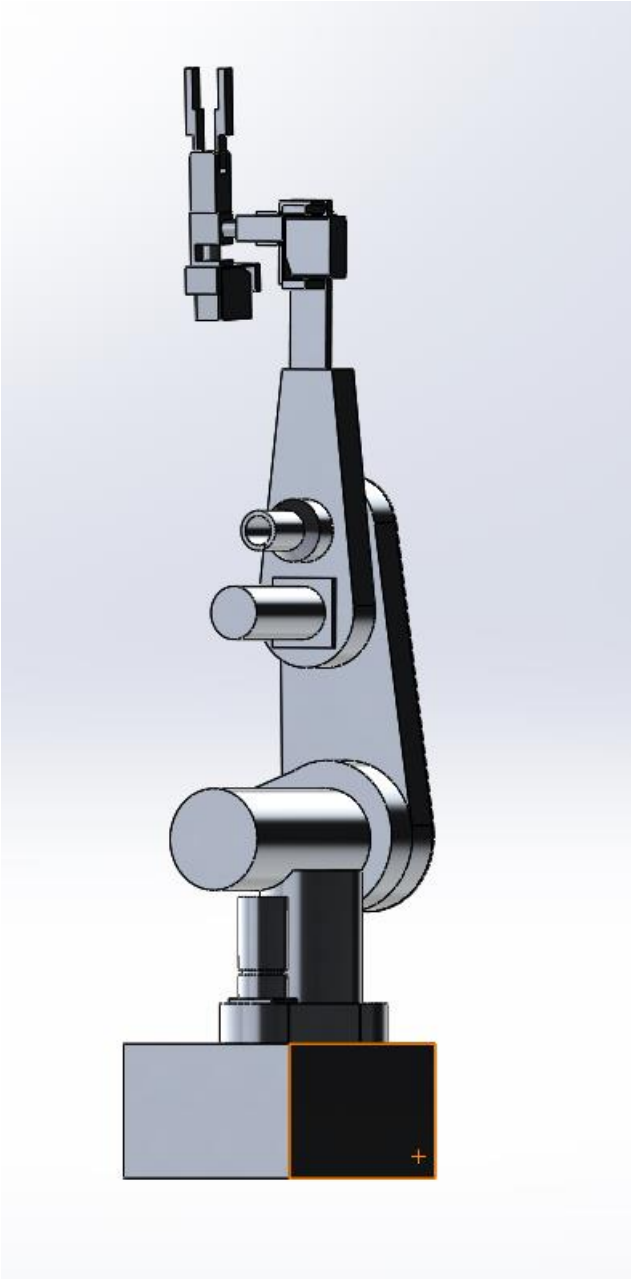


Figure 18 TQ Ma2000 Robot Arm Solidworks Design

This robot consists of 3 main joints (waist, shoulder, elbow) and three auxiliary joints (tilt, deflection, rotation).

There are 6 motors to control the holder at the end of the robot arm and rotate the joints. These are, respectively; waist, shoulder, elbow, tilt, deflection and rotation engines

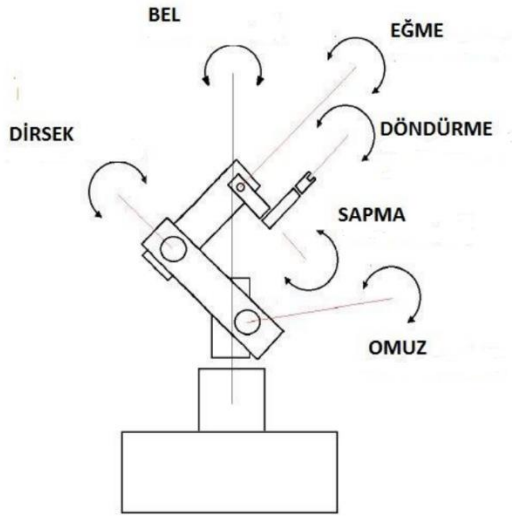


Figure 19 Potentiometer Relationship of TQ Ma2000 Robot Arm (Quinn,2013)

3.1 Exploded Views in Drawings

Exploded drawing view from an existing exploded assembly view has performed. The actual view is a model view, usually in the isometric orientation.

To create an exploded drawing view:

1. **In the assembly:**

a. Create a new configuration.

b. Create an exploded view.

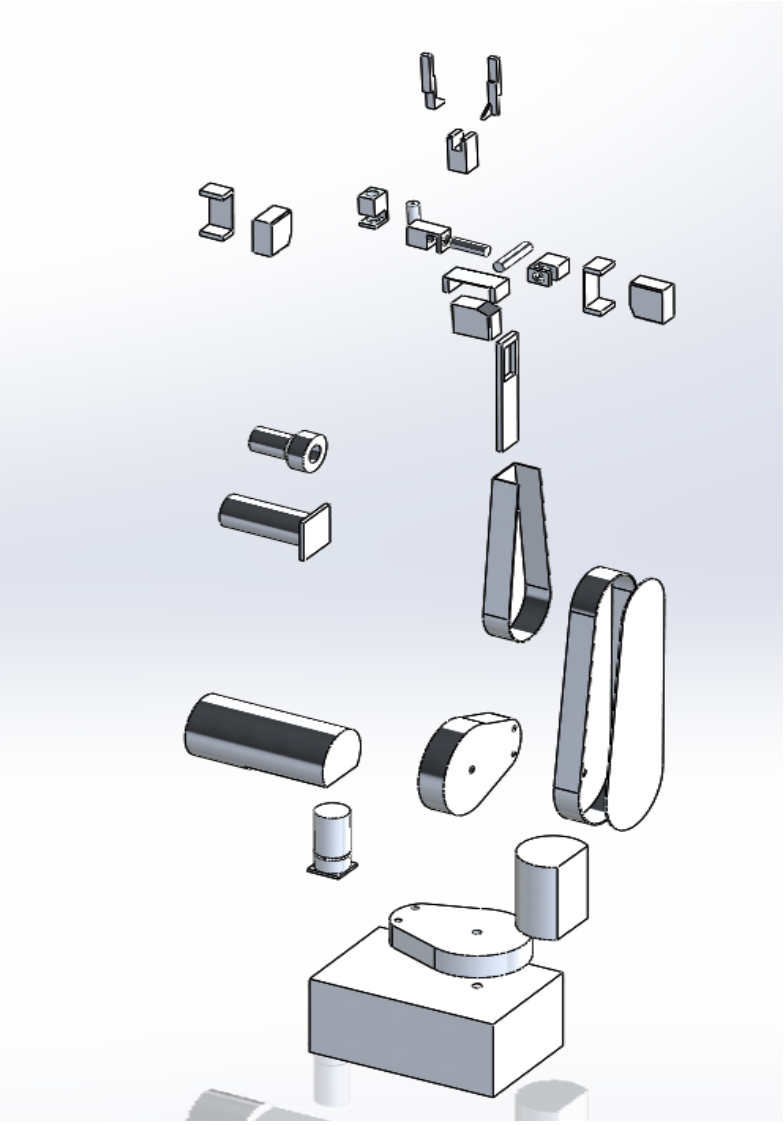


Figure 20 Solidworks Exploded View Drawing

2. In the drawing:

a. Insert a model view of the assembly using the orientation needed for the exploded view.

b. Right-click the drawing view and click Properties.

c. In the Drawing View Properties dialog box, under Configuration information, select Show in exploded or model break state.

4. ANALYTICAL SOLUTION WITH ANSYS AND MATLAB

4.1 Motor Selection with ANSYS and MATLAB

One of the aims of this project is to find the torque values of the motors in the 6 revolution joints of the robot arm. ANSYS Rigid Body Dynamics tool and MATLAB Simmechanics tool were used to find engine torque values. Then the torque results from the two applications were compared. Modeling of the robot arm with ANSYS and MATLAB is explained below.

4.2 ANSYS Section

Firstly, the geometry designed and assembled in Solidworks is transferred to Ansys rigid body dynamics tool. The ANSYS version used is ANSYS 2020 R2. Aluminum material is assigned to the transferred geometry from the engineering data section. The density of aluminum used is 2.7 g/m^3.

All contacts in the connection part of the robot arm have been deleted and mates have been given over ANSYS from the very beginning. The base box is fixed to the ground. 6 revolute joints have been assigned to the 6 jointed parts. The other parts are fixed from each other's connection points. Then, 6 joints were assigned to 6 joint points from the analysis settings and angle values were given to these joints for the movement of the robot arm.

The analysis time was decided to be 5 s. In this way, the moments that will occur in the revolute joints as a result of the rapid movements (aggressive) of the robot arm will also be found.

The movements of the robot arm have a certain complexity. This complexity is a representation of the actual movements of the robot arm. While angles are given to the revolute joints, the angles are chosen in such a way that maximum torque can be created for these joints. In this way, the maximum torque values that will occur in the revolve joints of the robot arm will be seen. Below are the angle values given to the revolute joints.

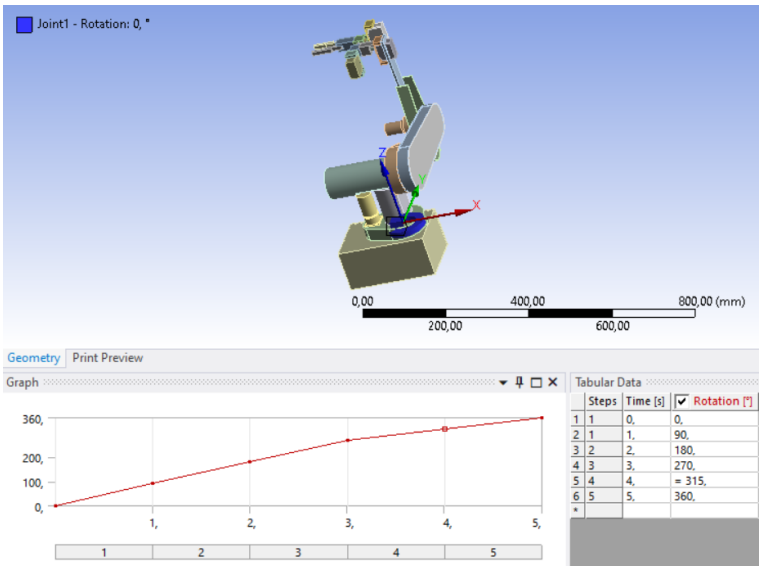


Figure 21 1. Revolute joint and angle values

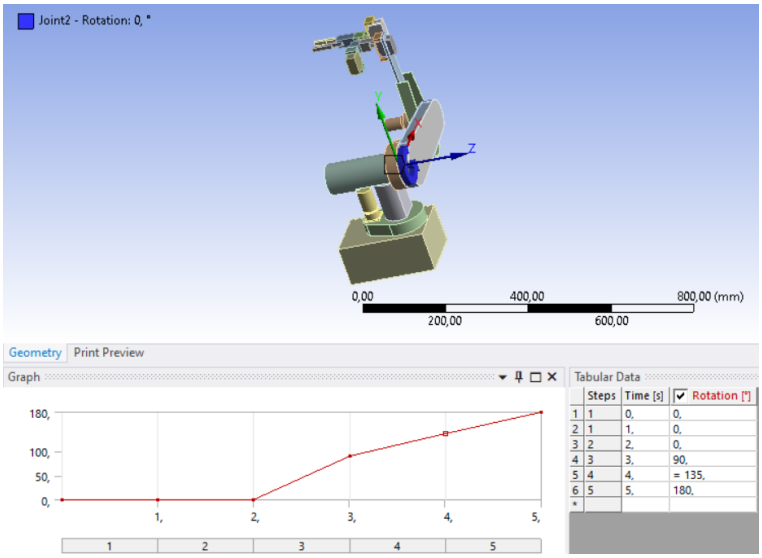


Figure 22 2. Revolute joint and angle values

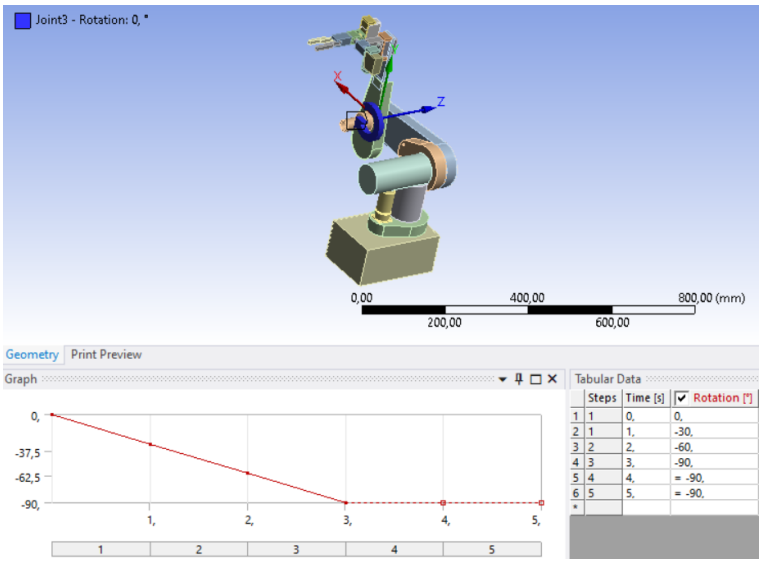


Figure 23 3. Revolute joint and angle values

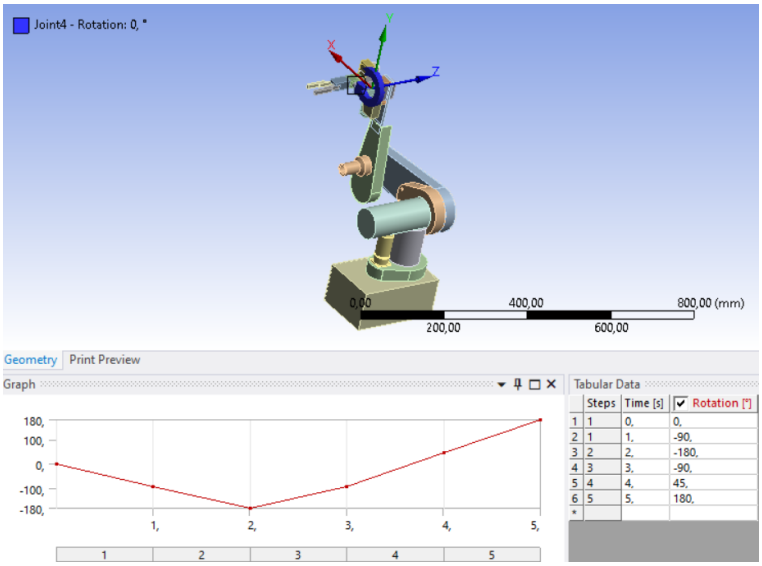


Figure 24 4. Revolute joint and angle values

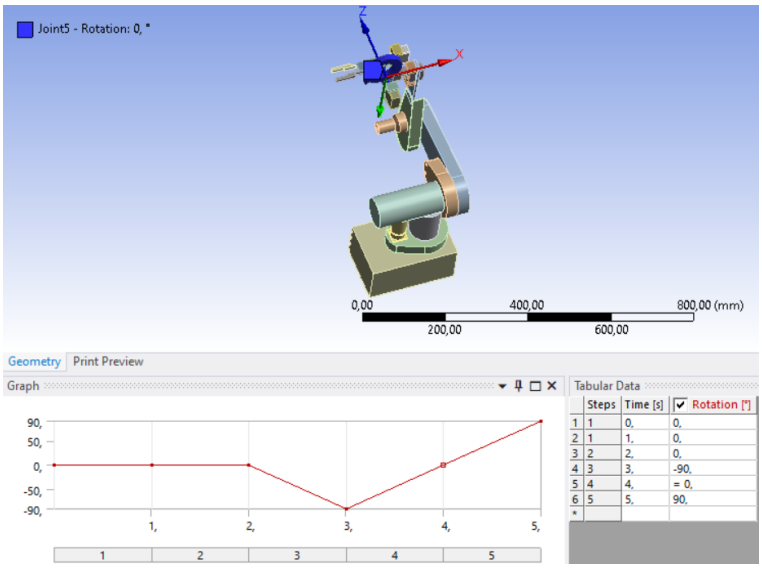


Figure 25 5. Revolute joint and angle values

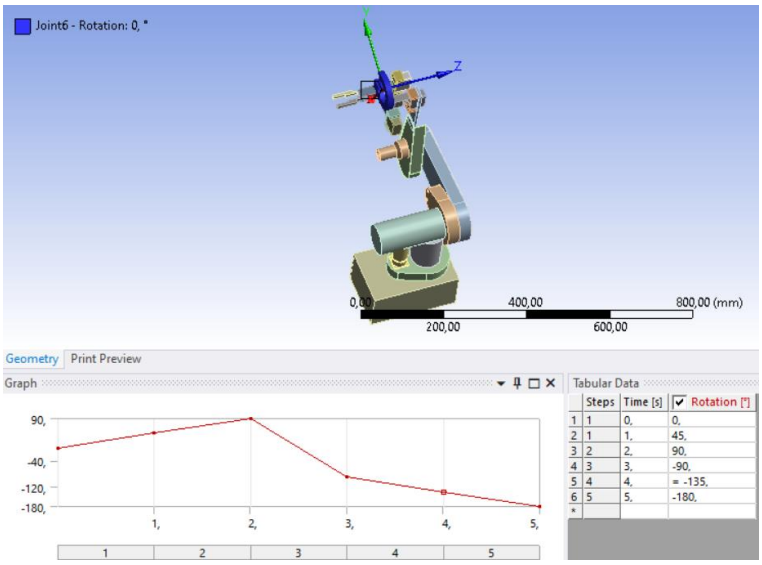


Figure 26 6. Revolute joint and angle values

Probe is assigned for each revolute joint by clicking on insert>probe>joint from the solution section.

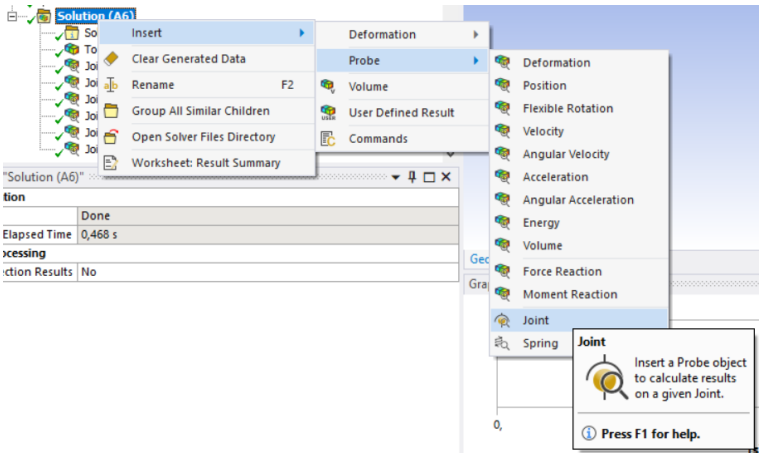


Figure 27 Probe assigned

For each joint probe, the result type is selected as the total moment in the option section. The torque required for the motors is given by the torque in the z axis. Therefore, the z axis is selected in the result selection.

Definition	
Type	Joint Probe
Boundary Condition	Revolute - sorunlu gövdenin parçasi To Taban Kutusu
Orientation Method	Joint Reference System
Suppressed	No
Options	
Result Type	Total Moment
Result Selection	Z Axis
Display Time	End Time
Results	
Maximum Value Over Time	
Minimum Value Over Time	
Information	

Figure 28 Result type and result selection

Finally, the prepared simulation was run and the results were found.

4.3 MATLAB Section

The MATLAB version used is MATLAB R2022a. Geometry designed with Solidworks is saved as an xml file by clicking Tools>Simscape Multibody Link>Export>Simscape Multibody. If Simscape is not installed before the file is opened in MATLAB, it is loaded and installed. Once installed, Simulink and Solidworks connection are made with the smlink_linksw command. The xml file is opened with the smimport '...xml' command.

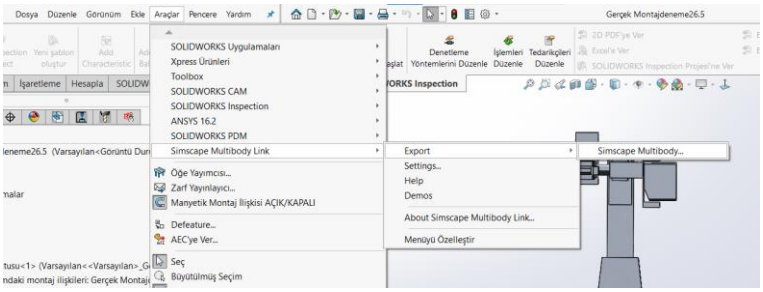


Figure 29 Exporting of Geometric from Solidworks Matlab Simmechanics

Before converting the geometry to an xml file, care was taken to ensure that the assembly was organized. If the assembly is complicated, it causes the Simulink to be complicated. Below is the Simulink block diagram.

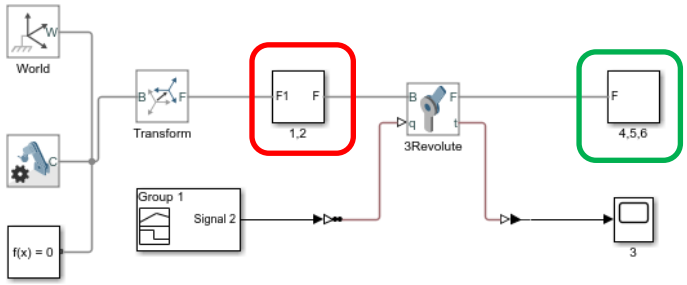


Figure 30 Main (Joint 3)

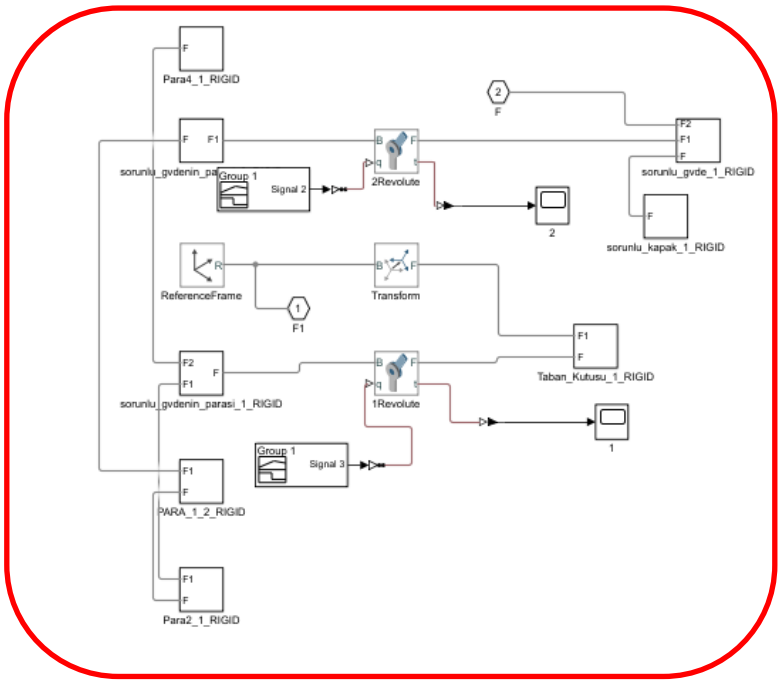


Figure 31 Left side(joint 1,2)

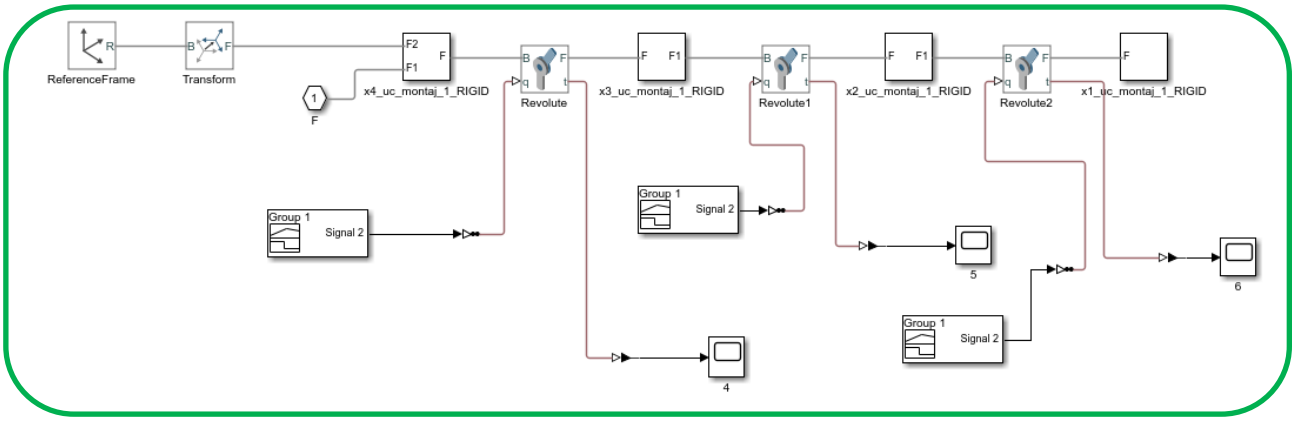


Figure 32 Right side (joint 4,5,6)

In simulink, signal block is connected for each revolute joint angle. The same angles given in ANSYS are entered into this signal block. In this way, the movement of the robot arm in ANSYS was the same as in Simulink.

Each of the revolute joints is set as follows. The motion value is input and is given as an angle. Torque value is output and the result is taken as Nmm.

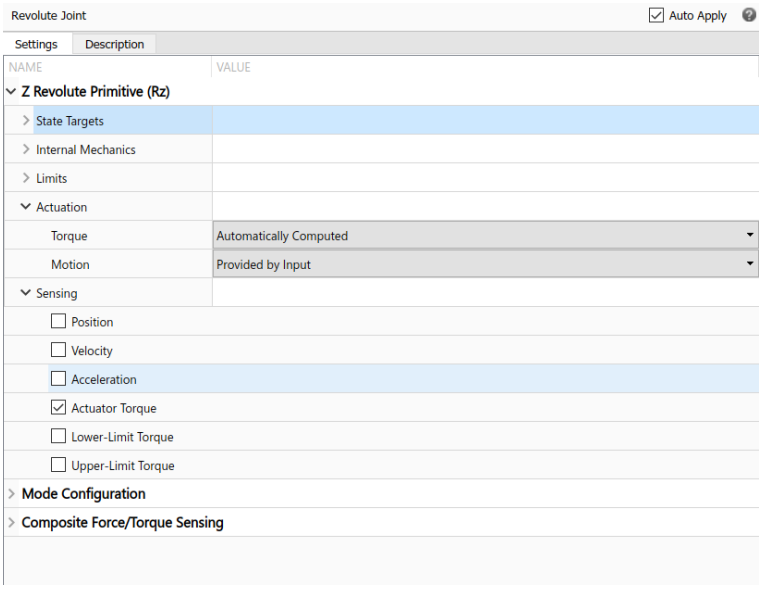


Figure 33 Determination of revolute joint input and output

The signal block cannot be directly connected to the revolute joint block. For this, Simulink PS converter is placed between the signal block and the revolute joint block. This block converts Simulink input signal to physical signal. Likewise, the PS Simulink Converter block is used before the scope block. This block converts the input physical signal to a Simulink output signal.

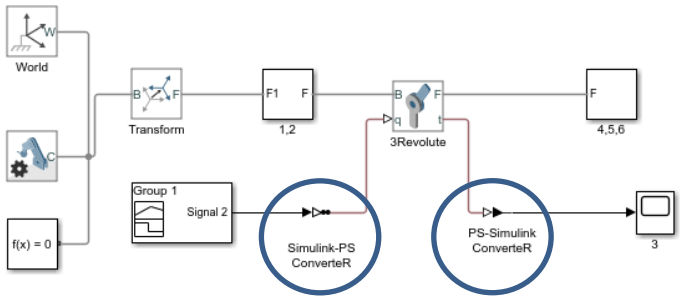


Figure 34 PS Simulink Conventer and Simulink PS converter

4.4 ANSYS and MATLAB Results and Comparison

Data from ANSYS and data from MATLAB are compatible with each other. However, there are fluctuations in the data received from MATLAB. Butterworth low pass filter is used to remove fluctuations. In addition, there are instantaneous high torque values at the angle change places in the joints. This is because the angle value changes instantaneously. That is, no matter how small the angle change is in a very small time interval, the torque value increases instantaneously because the time is very short.

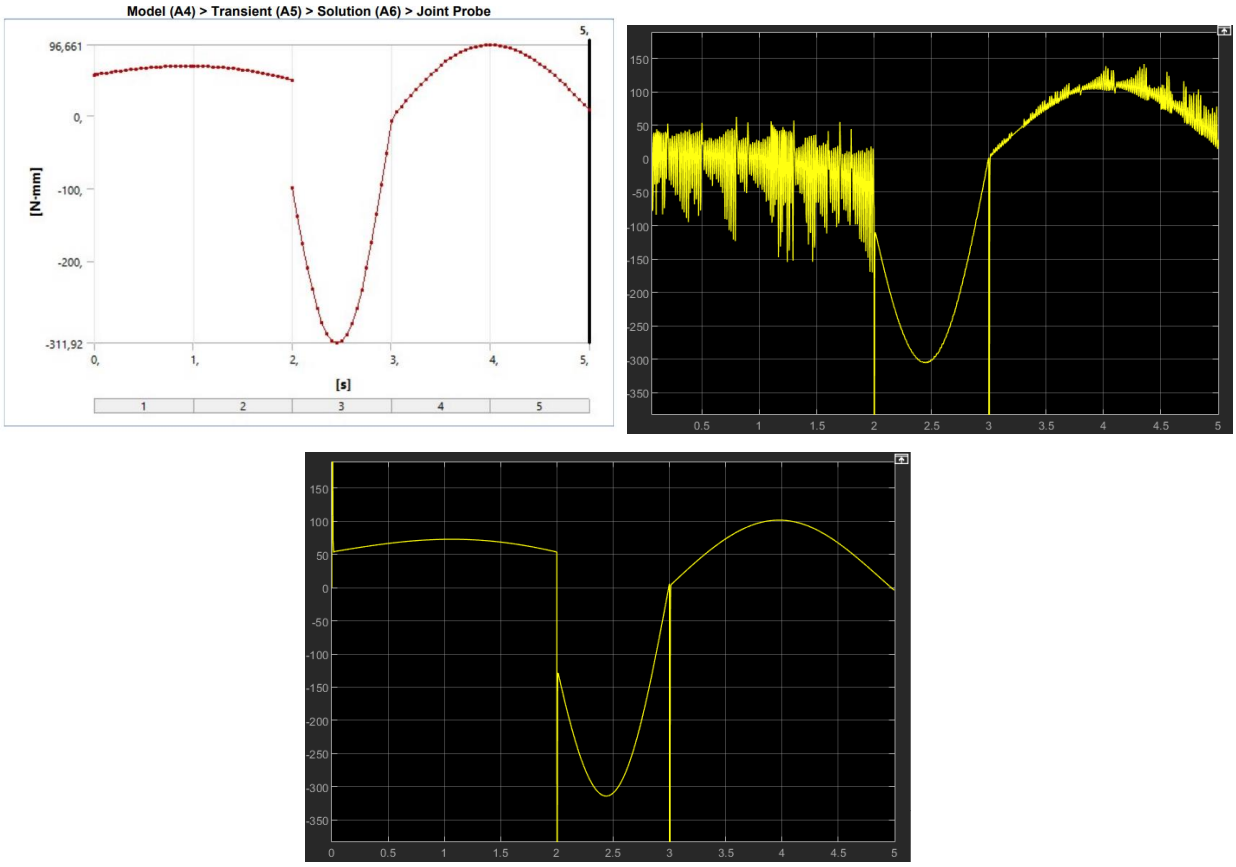


Figure 35 Results of ANSYS and MATLAB(without filter and with filter) of Joint 1

The torque value of the first revolute joint is approximately 312 Nmm. This joint is located at the bottom of the robot arm and carries the entire robot arm. Therefore, the largest torque value has occurred in this joint.

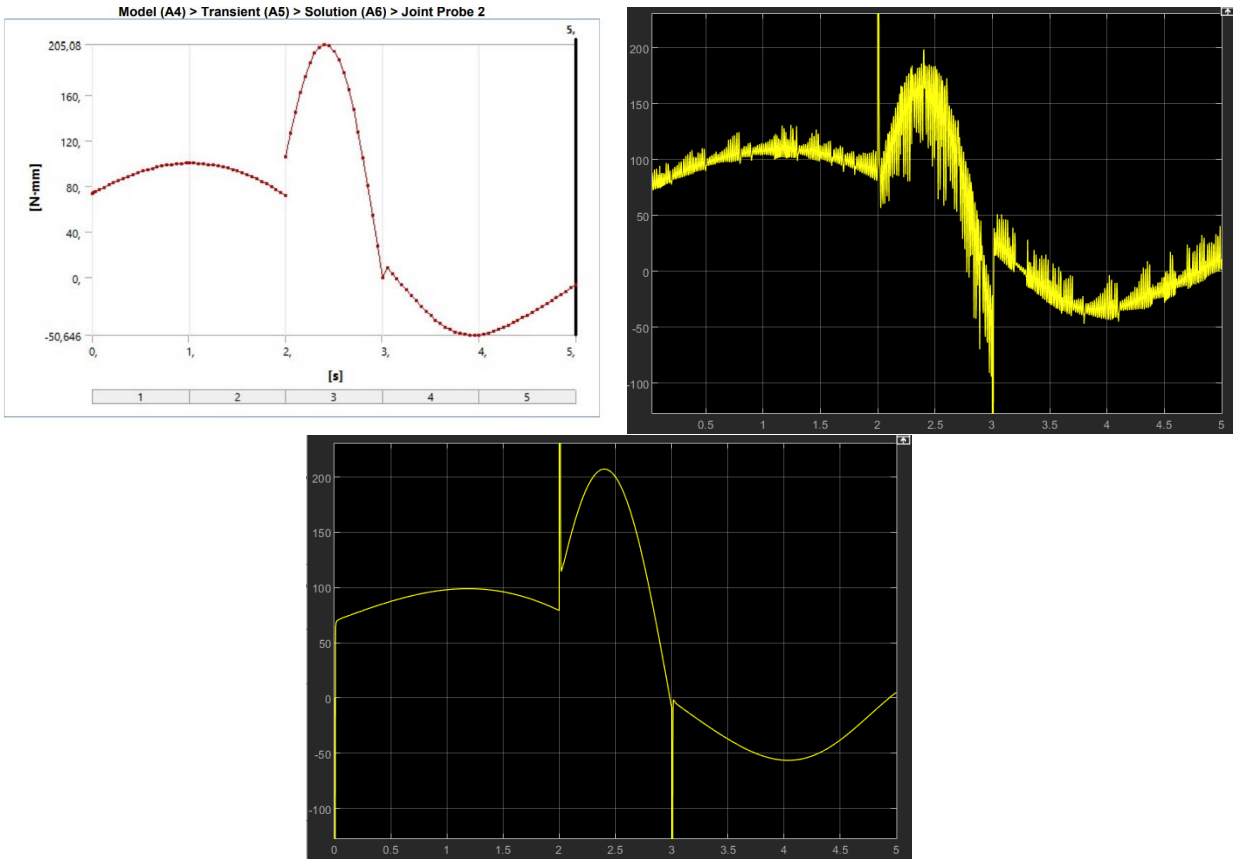


Figure 36 Results of ANSYS and MATLAB(without filter and with filter) of Joint 2

The 2nd revolute joint is the joint that carries the highest load after the 1st joint. According to the data obtained from ANSYS and MATLAB, the maximum approximate torque value is 205 Nmm.

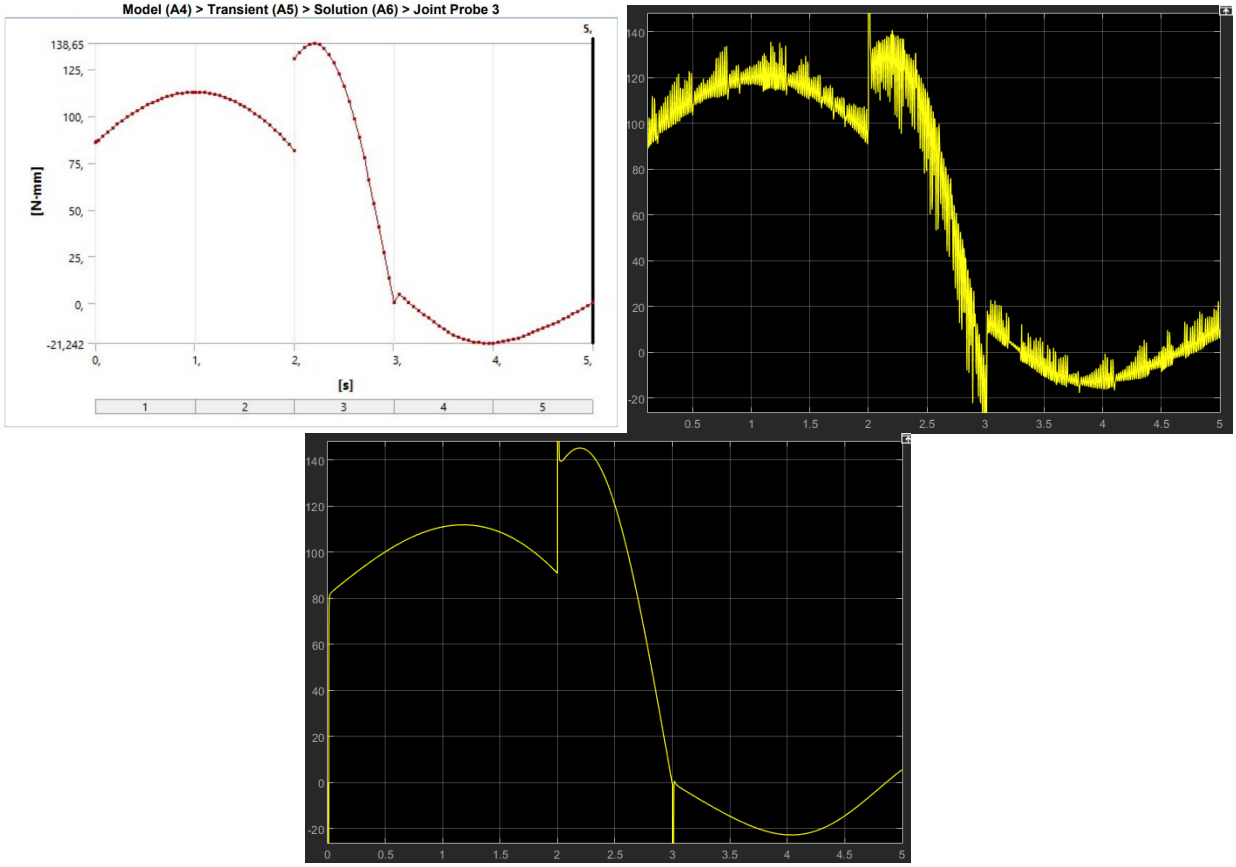


Figure 37 Results of ANSYS and MATLAB(without filter and with filter) of Joint 3

According to the data obtained from ANSYS and MATLAB, the maximum approximate torque value of the 3rd revolution joint is 139 Nmm.

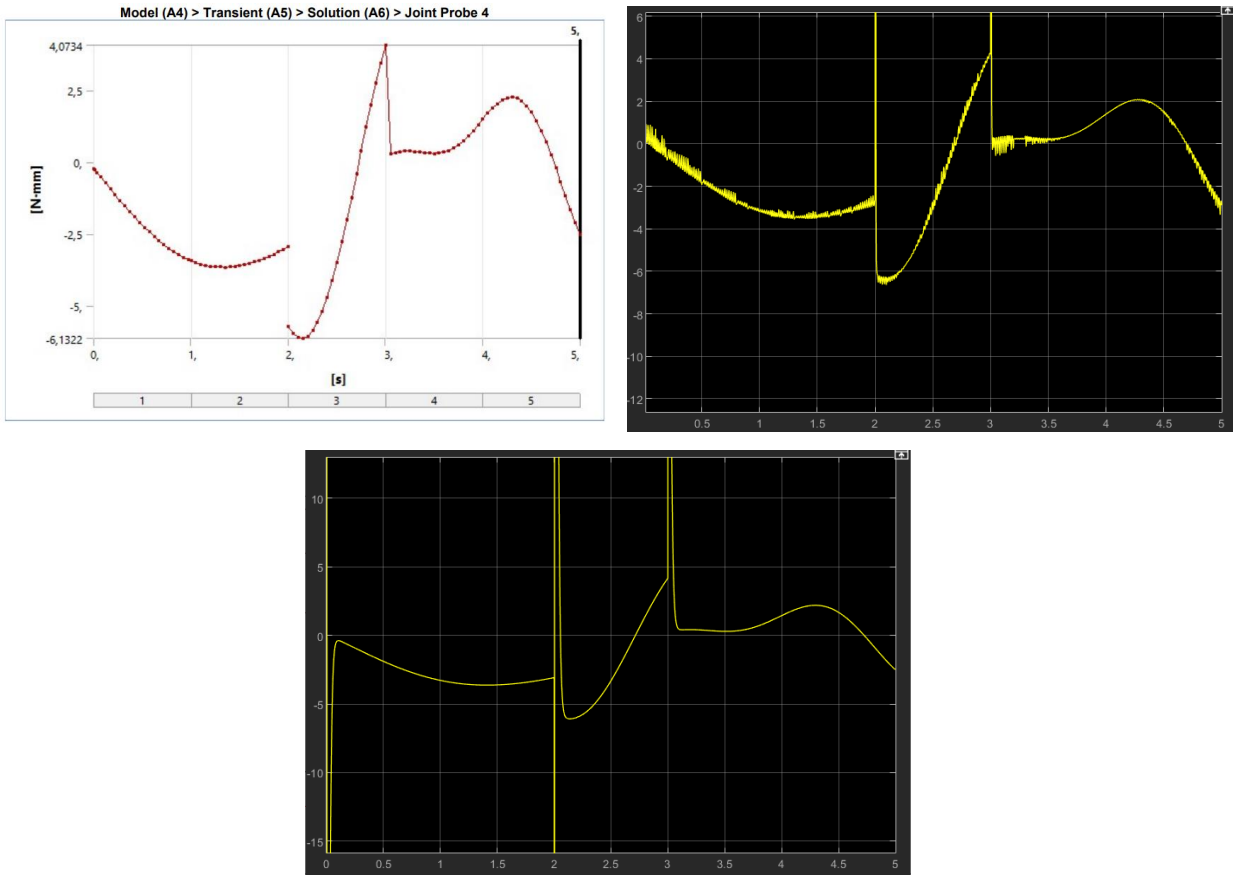


Figure 38 Results of ANSYS and MATLAB(without filter and with filter) of Joint 4

The torque value of the ends of the robot is quite low compared to the lower parts of the robot. The reason for this is that the load of the robot arm does not affect these joints too much. According to the data obtained from ANSYS and MATLAB, the maximum approximate torque value of the 4th rotation joint is 6.2 Nmm.

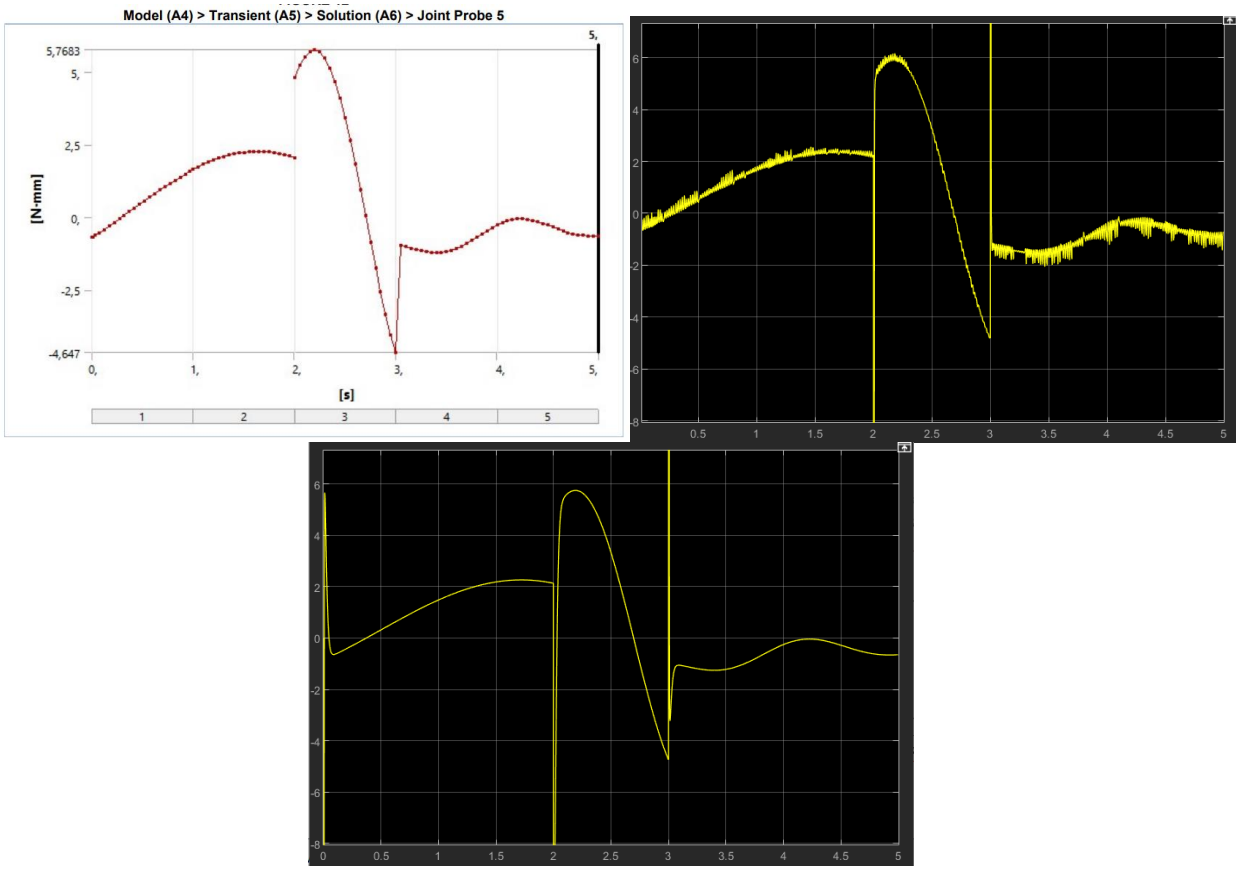


Figure 39 Results of ANSYS and MATLAB(without filter and with filter) of Joint 5

According to the data obtained from ANSYS and MATLAB, the maximum approximate torque value of the 5th revolution joint is 5.8 Nmm.

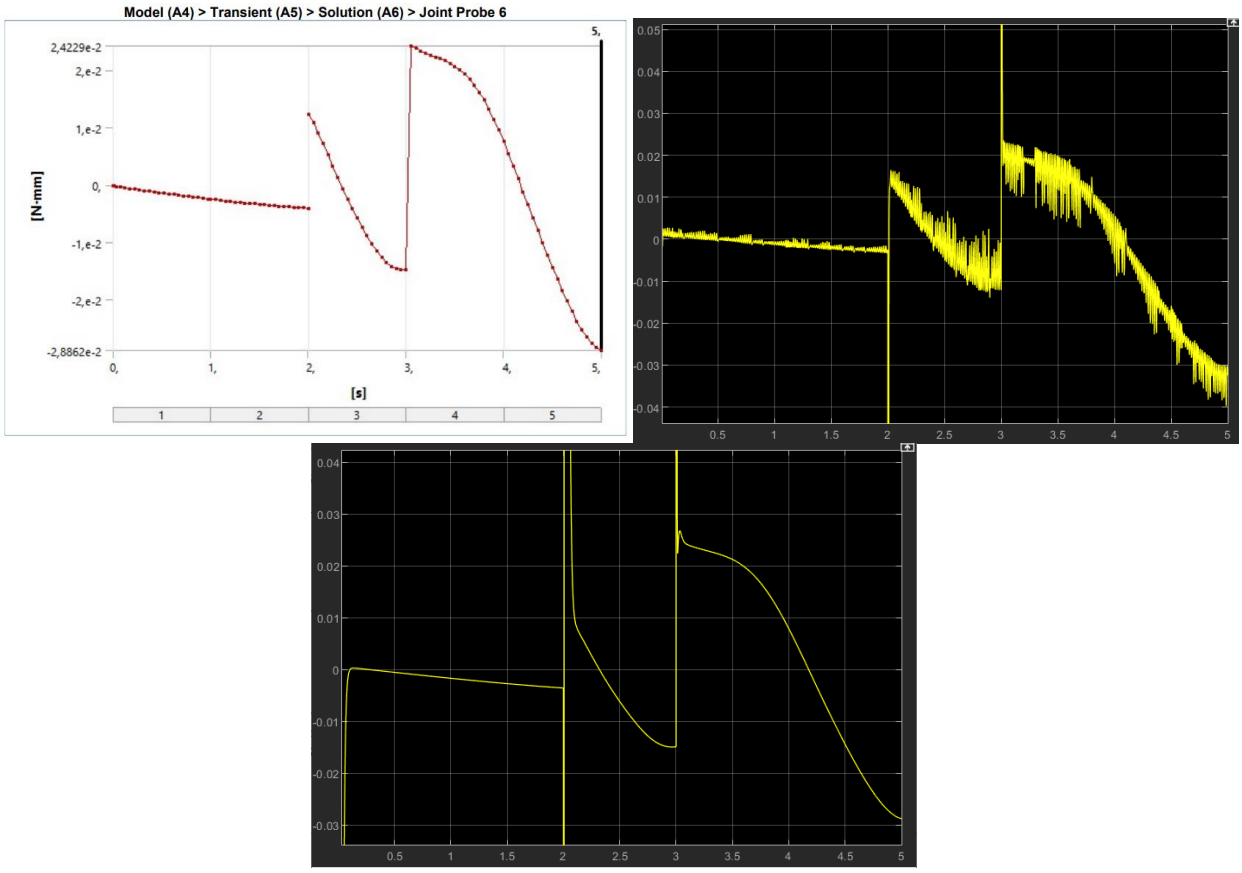


Figure 40 Results of ANSYS and MATLAB(without filter and with filter) of Joint 6

The revolute joint of the robot arm with the smallest torque is the extreme joint. According to the data obtained from ANSYS and MATLAB, the maximum approximate torque value of the 6th revolution joint is 0.03 Nmm.

From the bottom up, the torque in the revolute joints decreases. The reason for this is that more loads are placed on the joints down.

These simulations are done without any additional load on the robot arm. In other words, if the robot arm will hold and lift a part, these results will change, the torque values will increase in direct proportion to the load. In these simulations, it is assumed that no additional load will be placed on the robot arm. For example, this robot arm can be used as a welding machine, or something can be printed and drawn by holding a pen. These are not an excessive load on the robot arm and the results from the simulations are appropriate and accurate for these situations. In reality, the fingers at the end of the robot arm are very weak and will not be able to lift a heavy load anyway. Therefore, this assumption is correct for the TQ MA2000.

The torque values above can be used in the selection of motors to be used in revolute joints. If the motor is to be controlled more agressively (faster movement), the torque value can be selected higher.

5. PID CONTROLLER

5.1 What is the PID Controller?

Proportional Integral Derivative (PID) controller is an automatically optimized and accurate control system used to regulate different parameters like temperature, pressure and speed at desired values. The basic mechanism used in PID controllers is control loop feedback. A PID calculates the error by calculating difference between actual value and desired value and then sets the deciding parameters accordingly. This error is continuously being calculated until the process stops.

Proportional is used to find out the error between the desired value and actual value and is responsible for the corrective response. Integral is applied to calculate all the past values of error and then integrate them to find out the Integral term. When error is expelled from the system this integral stops increasing. The derivative is used to predict the expected error values in future based on the present values. Controlling effect can be increased if the system has a rapid rate of change, which is also based on Derivative. Combining all these three operations gives the name Proportional Integral Derivative (PID) Controller.

5.2 Schematic

In our project, we added PID Controllers for each joint. The main objective was calculating the error and decreasing with the help of the controller on Simulink. The models are specified below.

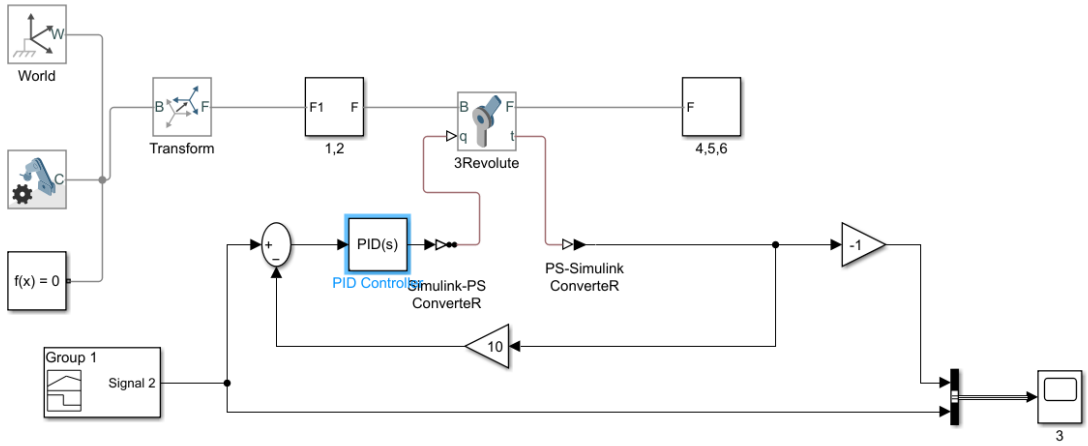


Figure 41 Main Schematic of the Robotic Manupilator

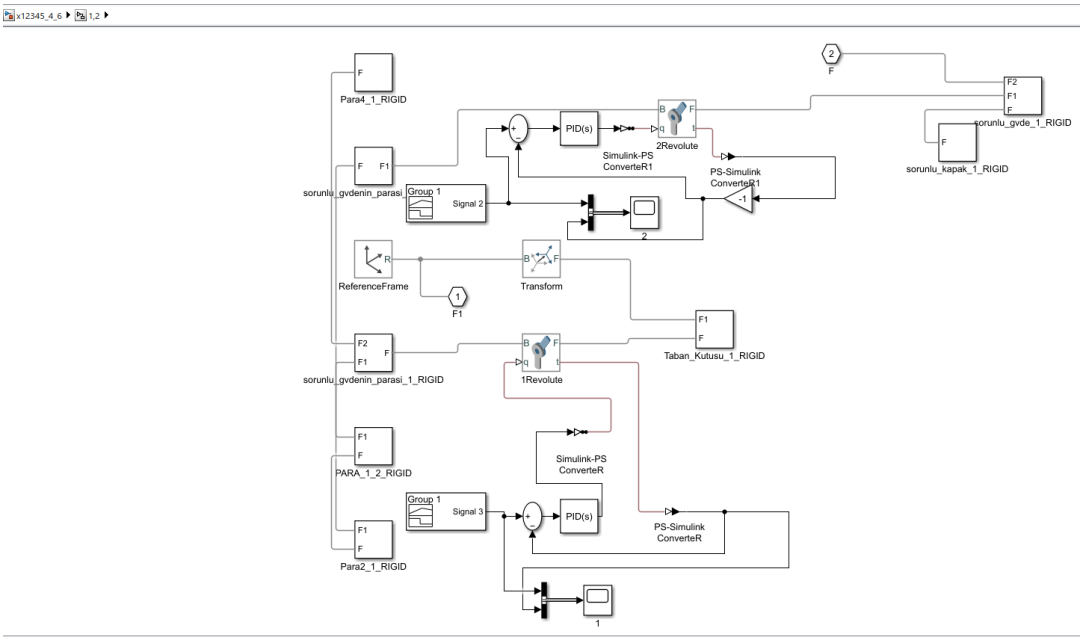


Figure 42 Schematic of Joints 1 and 2

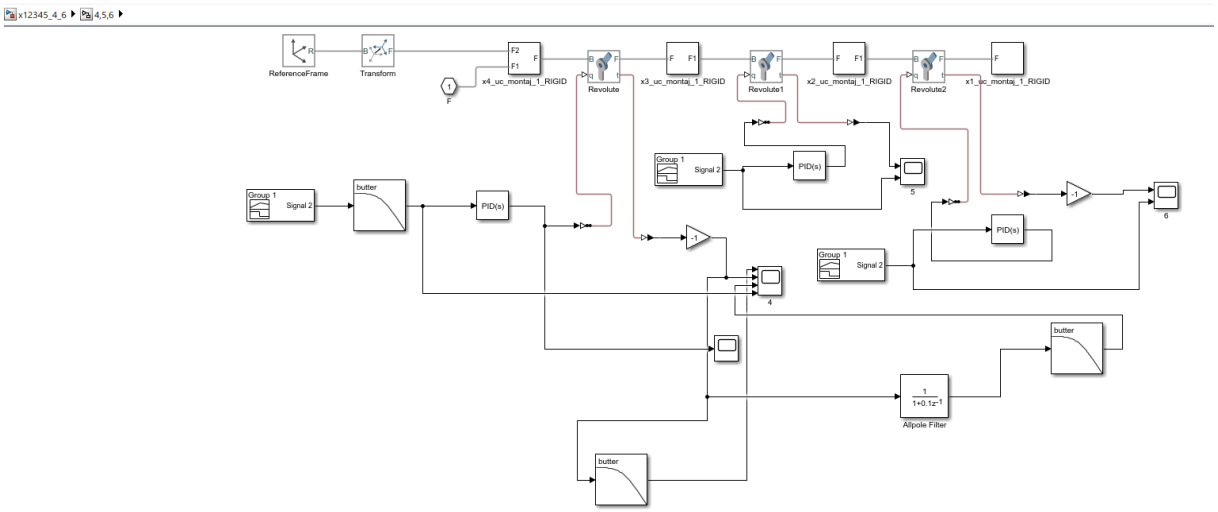
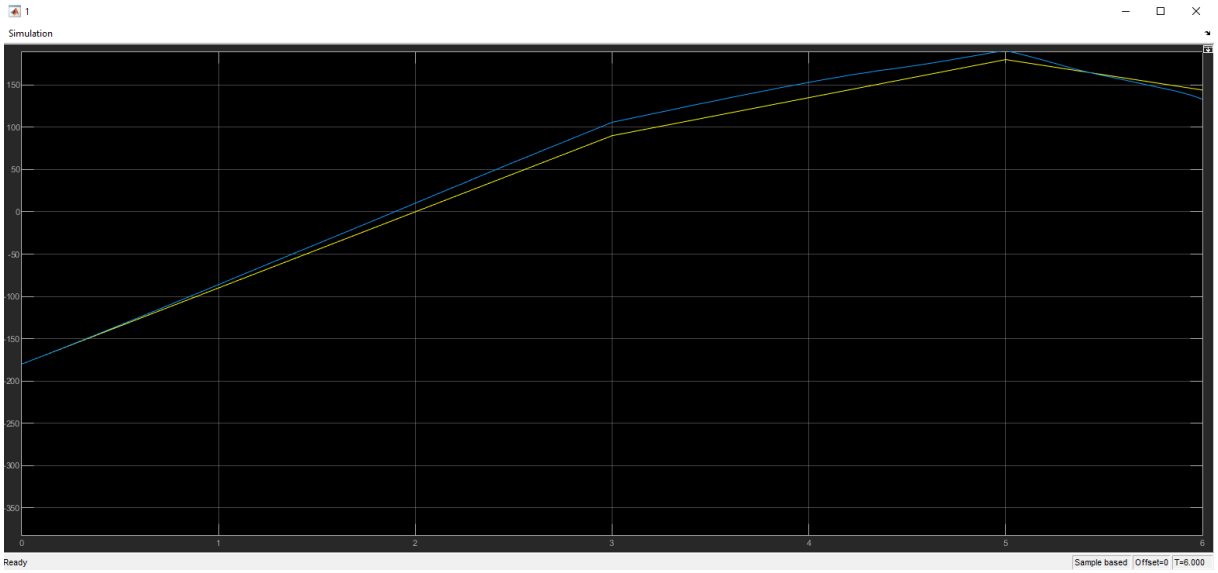


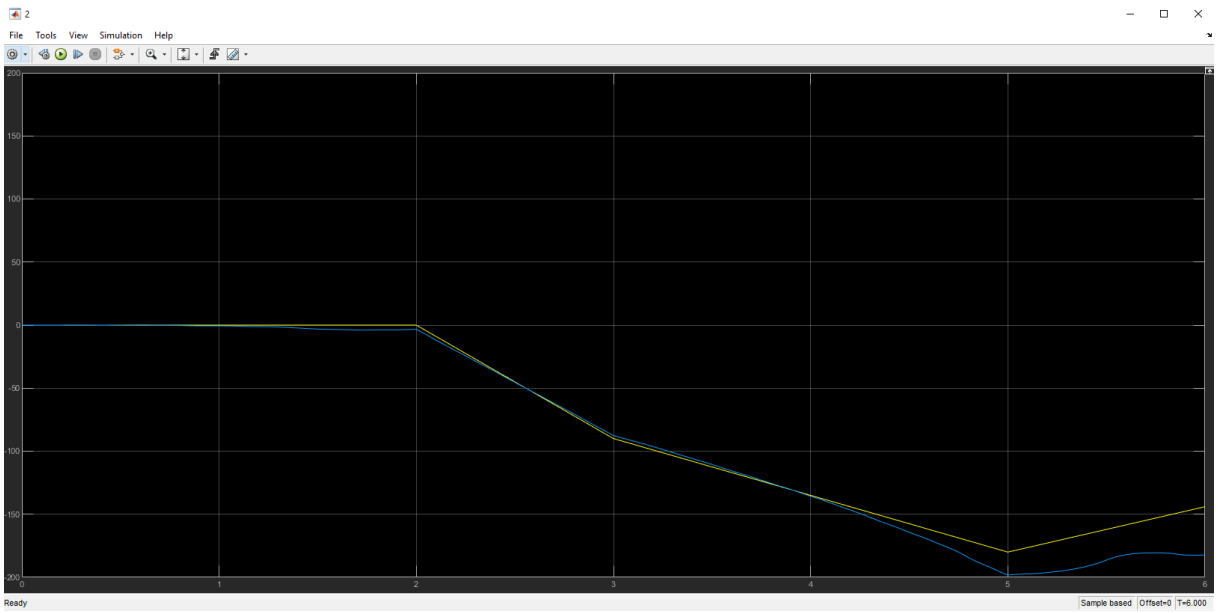
Figure 43 Schematic of Joints 4,5,6

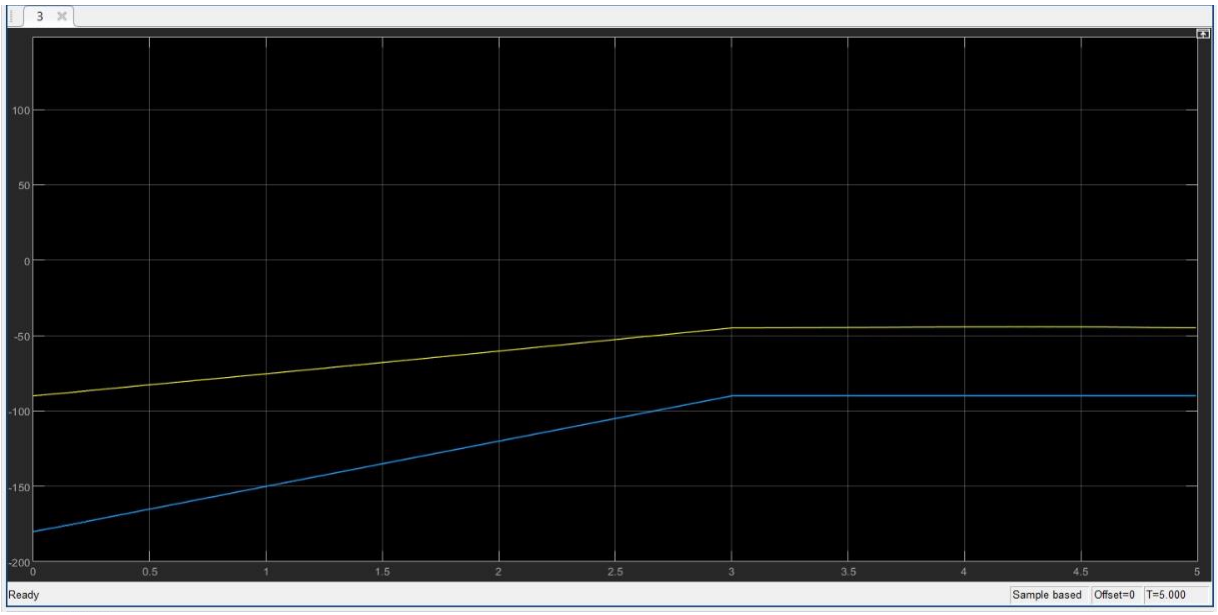
5.3 Results

In addition to ease of construct and computationally efficiency, PID proved to be an effective control algorithm for complex nonlinear systems like robot manipulators, in addition to its simple implementation and computational efficiency.

The results which running the Simulink model have been specified using tables for each joint.




2
File Tools View Simulation Help
200
150
100
50
0
-50
-100
-150
-200
0 1 2 3 4 5 6
Ready
Sample based Offset=0 T=6.000


3
100
50
0
-50
-100
-150
-200
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
Ready
Sample based Offset=0 T=5.000

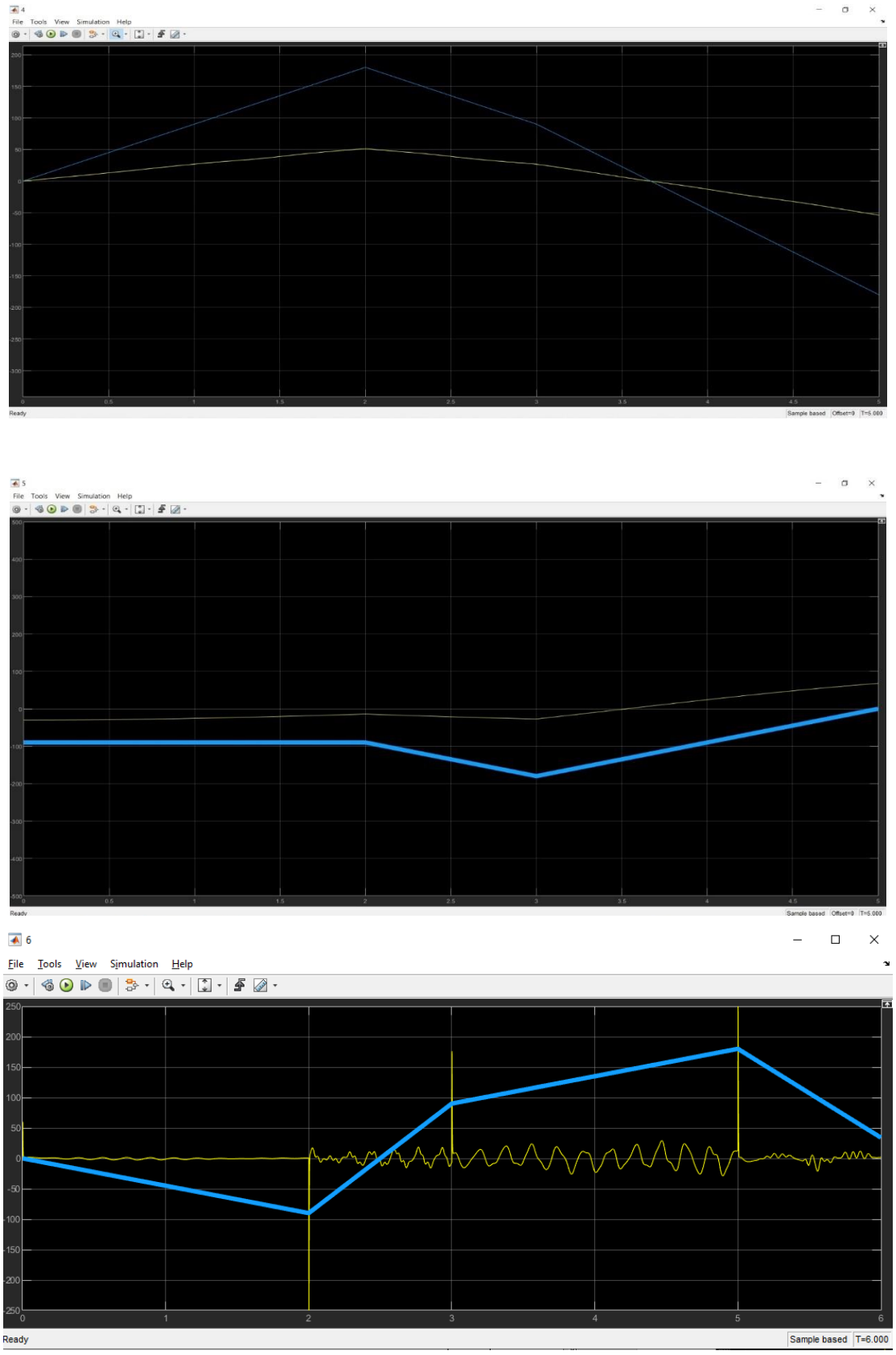


Figure 44 All results for PID Controller of Each Joint

6. CONCLUSION

6.1 Conclusion for Inverse Kinematics

The algorithm which we worked on to solve inverse kinematics of the manupilator has some features which:

- It can be applied with the majority of robotic manipulators, including those with complex structures and high degrees of freedom.

- For desired position and orientation, it gives all the possible sets of the variables of manupilator that attain the end-effector.

- It is less computationally complex than other approaches that determine the inverse kinematics problem's solution algebraically utilizing trigonometric (non linear) equations or numerical methods. Thus, the suggested algorithm's execution time and computational rate are lower than those of other approaches, such as the numerical method, making it more appropriate for usage in real-time applications.

In this Project, the algorithm has used to solve the inverse kinematics problem of the MA2000 robotic manupilator. Our result has 4 sets but sometimes there might be one, two or no sets. It depends about the manupilator and its structural capability.

6.2Conclusion for ANSYS and MATLAB Comparison

The selection of the motors to be used in the joints is very important in order to ensure the movement of the joints of the robots correctly. When choosing these motors, it should be determined how much torque affects the joints. In this way, the torque value of the motor required for that joint is found. One of the aims of this project is to find the torque values of the motors in the revolute joints of the TQ MA2000 robot arm. These values were found using ANSYS Rigid Body Dynamics tool and MATLAB Simmechanics tool. Torque values were found by giving the same angle values to the joints of the same angle robot arm in both applications. Torque values found from ANSYS and MATLAB are quite compatible with each other.

The highest torque value was naturally found in the lowest revolute joint of the robot arm. This is because the bottom joint carries the entire robot. It has been observed that the required motor torque value decreases as the robot arm reaches the end points.

6.3Conclusion for PID Controller

For each joints, it is significant to control step by step to find the errors or mistakes by ourselves. While choosing the values which we used in the PID Controller, the main purpose was nothing but efficiency and fitness to the robot's dynamics. We determined to get results closer and appropriate, that is how the Proportional, Integral and Derivative parameters has chosen.

The errors are also could be observed by checking graphs. However, we determined to get as much as close to the calculated values. The values found at the examined part, was the closest value which we might found for MA 2000 robot.

7. BIBLIOGRAPHY

[1]Craig, J.J. (2005). Introduction to robotics, mechanics and control (3. Baskı). USA: Pearson. (page-63)

[2]Craig, J.J. (2005). Introduction to robotics, mechanics and control (3. Baskı). USA: Pearson. (page-66)

[3]Rigid Body Dynamics, Time Integration Method – Lesson 4 (Ansys, 2021)

[4]Niku, S.B. (2001). Introduction to robotics, analysis, systems, applications. USA: Prentice Hall.

[5]Hagele, M., Nilson, K., ve Pires, J. N. (2008). Handbook of robotics. USA: Springer

[6]Kurfess, T. R. (Ed). (2005). Robotics and automation handbook. USA: CRC Press

[7]Hüseyin Havusoğlu (2014) Robot kol tasarımı, Kinematik Analizi ve etkileşimli kontrolü, Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü – (Haziran 2014 /İzmir, 1-2)

[8]https://courses.ansys.com/wp-content/uploads/2021/02/Lesson4_RigidBodyDynamics.pdf

[9]Control of 3 DOF (3R) Robot Manipulator and Moving Objects Based on Image Processing, Ekrem Yavuz, Musa Alcı, Erol Uyar

[10]Inverse Kinematics Analysis for Manipulator Robot With Wrist Offset Based On the Closed-Form Algorithm Mohammed Z. Al-Faiz , Mohammed S.Saleh

[11]Kinematics Analysis of 6-DoF Articulated Robot with Spherical Wrist, Seemal Asif and Philip Webb

[12]An Algorithm to Solve the Inverse Kinematics Problem of a Robotic Manipulator Based on Rotation Vectors, Mohamad Z. Al-Faiz, Mazin Z. Othman, and Baker B. Al-Bahri

[13]https://www.youtube.com/watch?v=xpA8TKEMpMk

[14]https://www.mathworks.com/help/physmod/sm/ug/import-robot-arm-model.html

[15]Design and Implementation of PSO-PID Controller for MA2000 Robotic Manipulator, Firas Abdullah Thweny Al-Saed

[16]Essam Hashem humonoid arm robot simulation on sımulink matlab 2020

8. APPENDIX

Matlab Data File

```
% Simscape(TM) Multibody(TM) version: 7.5

% This is a model data file derived from a Simscape Multibody Import XML file using
the smimport function.
% The data in this file sets the block parameter values in an imported Simscape
Multibody model.
% For more information on this file, see the smimport function help page in the
Simscape Multibody documentation.
% You can modify numerical values, but avoid any other changes to this file.
% Do not add code to this file. Do not edit the physical units shown in comments.

%%%VariableName:smiData


%============= RigidTransform =============%

%Initialize the RigidTransform structure array by filling in null values.
smiData.RigidTransform(43).translation = [0.0 0.0 0.0];
smiData.RigidTransform(43).angle = 0.0;
smiData.RigidTransform(43).axis = [0.0 0.0 0.0];
smiData.RigidTransform(43).ID = "";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(1).translation    =    [198.92966365986962    29.09103609341819
113.95460822200664];  % mm
smiData.RigidTransform(1).angle = 2.0943951023931957;  % rad
smiData.RigidTransform(1).axis     =     [0.57735026918962584     -0.57735026918962573
0.57735026918962573];
smiData.RigidTransform(1).ID = "B[1234:4.uc montaj-1:-:1234:3. uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(2).translation    =    [-1.743989398728798    -11.224810451062424
16.40000002579654];  % mm
smiData.RigidTransform(2).angle = 2.0943951023931957;  % rad
smiData.RigidTransform(2).axis     =     [0.57735026918962595     -0.57735026918962573
0.57735026918962562];
smiData.RigidTransform(2).ID = "F[1234:4.uc montaj-1:-:1234:3. uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(3).translation    =    [-1.7439893791877381    1.4851895489771678
4.4999999999936762];  % mm
smiData.RigidTransform(3).angle = 0;  % rad
smiData.RigidTransform(3).axis = [0 0 0];
smiData.RigidTransform(3).ID = "B[1234:3. uc montaj-1:-:1234:2. uc montaj-1]";

%Translation Method - Cartesian
```

```
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(4).translation    =    [-1.743989386857882    -28.014810433302834
16.399999997122443];  % mm
smiData.RigidTransform(4).angle = 2.0943951024161573;  % rad
smiData.RigidTransform(4).axis     =     [0.57735026919727961     -0.57735026917439425
0.57735026919720345];
smiData.RigidTransform(4).ID = "F[1234:3. uc montaj-1:-:1234:2. uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(5).translation    =    [-1.7439893791877807    1.485189548977166
4.4999999999936406];  % mm
smiData.RigidTransform(5).angle = 0;  % rad
smiData.RigidTransform(5).axis = [0 0 0];
smiData.RigidTransform(5).ID = "B[1234:2. uc montaj-1:-:1234:1.uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(6).translation    =    [-35.632947896202914    -0.9367292664918736
56.810000033528745];  % mm
smiData.RigidTransform(6).angle = 2.0943951023931962;  % rad
smiData.RigidTransform(6).axis     =     [-0.57735026918962595     -0.57735026918962595
0.57735026918962551];
smiData.RigidTransform(6).ID = "F[1234:2. uc montaj-1:-:1234:1.uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(7).translation    =    [232.72319130434789    41.610986879908552
29.999999999999986];  % mm
smiData.RigidTransform(7).angle = 2.0570847705651816;  % rad
smiData.RigidTransform(7).axis     =     [-0.6024975919017066     -0.56435655916833494
0.56435655916833494];
smiData.RigidTransform(7).ID  =  "B[5.uc  montaj:sorunlu  gövde-1:-:5.uc  montaj:sorunlu
kapak-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(8).translation    =    [232.72319130434818    -41.610986879908324
1.7000000000003652];  % mm
smiData.RigidTransform(8).angle = 2.057084770565182;  % rad
smiData.RigidTransform(8).axis    =    [0.60249759190170649    -0.56435655916833483    -
0.56435655916833505];
smiData.RigidTransform(8).ID  =  "F[5.uc  montaj:sorunlu  gövde-1:-:5.uc  montaj:sorunlu
kapak-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(9).translation    =    [111.47728715514386    -233.04669233914851
30.200000000000003];  % mm
smiData.RigidTransform(9).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(9).axis = [1 0 0];
smiData.RigidTransform(9).ID  =  "B[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Taban Kutusu-1]";

%Translation Method - Cartesian
```

```
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(10).translation       =       [157.5       71.849999999999682
2.8421709430404007e-14];  % mm
smiData.RigidTransform(10).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(10).axis = [1 0 0];
smiData.RigidTransform(10).ID  =  "F[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Taban Kutusu-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(11).translation = [-29.292853031174722 24.518743061054309 0];
% mm
smiData.RigidTransform(11).angle = 2.0700838523544851;  % rad
smiData.RigidTransform(11).axis     =     [0.59367312786777549     0.56901327631598009
0.56901327631598009];
smiData.RigidTransform(11).ID = "B[5.uc montaj:Parça2-1:-:5.uc montaj:PARÇA 1-2]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(12).translation   =   [29.565349707425703   26.509302037512281
86.918266418902334];  % mm
smiData.RigidTransform(12).angle = 1.5709965873346654;  % rad
smiData.RigidTransform(12).axis    =    [0.014149927285200106    -0.99979975950969668
0.014149927285200132];
smiData.RigidTransform(12).ID = "F[5.uc montaj:Parça2-1:-:5.uc montaj:PARÇA 1-2]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(13).translation = [0 0 0];  % mm
smiData.RigidTransform(13).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(13).axis = [1 0 0];
smiData.RigidTransform(13).ID  =  "B[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Parça2-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(14).translation   =   [237.61158291575697   102.64962932864117
100.00000000000003];  % mm
smiData.RigidTransform(14).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(14).axis = [0.69195387261684882 -0.72194171383190331 0];
smiData.RigidTransform(14).ID  =  "F[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Parça2-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(15).translation = [0 0 0];  % mm
smiData.RigidTransform(15).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(15).axis = [1 0 0];
smiData.RigidTransform(15).ID  =  "B[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Parça4-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(16).translation  =  [233.04926255229441  -182.34828703317106  -
35.365329779414921];  % mm
```

```
smiData.RigidTransform(16).angle = 1.5707963267948966;  % rad
smiData.RigidTransform(16).axis = [0 0 1];
smiData.RigidTransform(16).ID  =  "F[5.uc  montaj:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj:Parça4-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(17).translation  =  [-0.068178002347610289  2.4081642122125886
86.918266418901624];  % mm
smiData.RigidTransform(17).angle = 0;  % rad
smiData.RigidTransform(17).axis = [0 0 0];
smiData.RigidTransform(17).ID  =  "B[5.uc  montaj:PARÇA  1-2:-:5.uc  montaj:sorunlu
gövdenin parçasi-2]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(18).translation  =  [111.54534951976086  -235.45485982242465  -
1.0516032489249483e-12];  % mm
smiData.RigidTransform(18).angle = 3.1415446345831204;  % rad
smiData.RigidTransform(18).axis = [4.4408920997806151e-16 5.5511151247257689e-17 1];
smiData.RigidTransform(18).ID  =  "F[5.uc  montaj:PARÇA  1-2:-:5.uc  montaj:sorunlu
gövdenin parçasi-2]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(19).translation  =  [111.47728715514393  -233.04669233914859
30.200000000000003];  % mm
smiData.RigidTransform(19).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(19).axis = [1 0 0];
smiData.RigidTransform(19).ID  =  "B[5.uc  montaj:sorunlu  gövdenin  parçasi-2:-:5.uc
montaj:sorunlu gövde-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(20).translation  =  [-3.7664166914279077e-12  -
1.2648436865521227e-14 -1.3446233020860312e-12];  % mm
smiData.RigidTransform(20).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(20).axis  =  [1  -1.5525374011869744e-33  -2.8040436165284465e-
17];
smiData.RigidTransform(20).ID  =  "F[5.uc  montaj:sorunlu  gövdenin  parçasi-2:-:5.uc
montaj:sorunlu gövde-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(21).translation  =  [230.00000000000003  0  1.7000000000000071];
% mm
smiData.RigidTransform(21).angle = 3.1415926535897931;  % rad
smiData.RigidTransform(21).axis = [1 0 0];
smiData.RigidTransform(21).ID  =  "B[5.uc  montaj-1:sorunlu  gövde-1:-:1234-1:4.uc
montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(22).translation  =  [-26.055116401709732  -62.70896390658217
90.368098159512996];  % mm
```

```
smiData.RigidTransform(22).angle = 2.0943951023931953;  % rad
smiData.RigidTransform(22).axis   =   [-0.57735026918962584   -0.57735026918962584   -
0.57735026918962584];
smiData.RigidTransform(22).ID   =   "F[5.uc   montaj-1:sorunlu   gövde-1:-:1234-1:4.uc
montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(23).translation   =   [-11.293989379187769   -11.214810451022823
4.4999999999936406];  % mm
smiData.RigidTransform(23).angle = 0;  % rad
smiData.RigidTransform(23).axis = [0 0 0];
smiData.RigidTransform(23).ID   =   "AssemblyGround[1234-1:2.   uc   montaj-1:Silindir
yatagi-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(24).translation   =   [-46.793989379187799   -12.214810451022823
44.799999999993645];  % mm
smiData.RigidTransform(24).angle = 0;  % rad
smiData.RigidTransform(24).axis = [0 0 0];
smiData.RigidTransform(24).ID  =  "AssemblyGround[1234-1:2.  uc  montaj-1:Motor  yatagi-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(25).translation   =   [-38.393989379187779   12.885189548977188
46.199999999993651];  % mm
smiData.RigidTransform(25).angle = 0;  % rad
smiData.RigidTransform(25).axis = [0 0 0];
smiData.RigidTransform(25).ID = "AssemblyGround[1234-1:2. uc montaj-1:Servo 3-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(26).translation   =   [-1.7439893791877659   -11.214810451022823
16.399999999993643];  % mm
smiData.RigidTransform(26).angle = 1.5707963267948968;  % rad
smiData.RigidTransform(26).axis = [1 0 0];
smiData.RigidTransform(26).ID = "AssemblyGround[1234-1:2. uc montaj-1:Ust servo mili-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(27).translation   =   [-11.293989379187741   -11.214810451022835
4.4999999999936762];  % mm
smiData.RigidTransform(27).angle = 0;  % rad
smiData.RigidTransform(27).axis = [0 0 0];
smiData.RigidTransform(27).ID   =   "AssemblyGround[1234-1:3.   uc   montaj-1:Silindir
yatagi-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(28).translation   =   [-38.393989379187751   12.885189548977154
46.199999999993665];  % mm
smiData.RigidTransform(28).angle = 0;  % rad
```

```
smiData.RigidTransform(28).axis = [0 0 0];
smiData.RigidTransform(28).ID = "AssemblyGround[1234-1:3. uc montaj-1:Servo 3-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(29).translation = [-1.7439893791877381 -51.524810451022837
16.399999999993696]; % mm
smiData.RigidTransform(29).angle = 2.0943951023931953; % rad
smiData.RigidTransform(29).axis = [-0.57735026918962584 -0.57735026918962584 -
0.57735026918962584];
smiData.RigidTransform(29).ID = "AssemblyGround[1234-1:3. uc montaj-1:Alt servo mili-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(30).translation = [-46.793989379187771 -12.214810451022833
44.799999999993652]; % mm
smiData.RigidTransform(30).angle = 0; % rad
smiData.RigidTransform(30).axis = [0 0 0];
smiData.RigidTransform(30).ID = "AssemblyGround[1234-1:3. uc montaj-1:Motor yatagi-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(31).translation = [72.677052105886716 3.9132707370860551
74.109999999999005]; % mm
smiData.RigidTransform(31).angle = 1.5707963267948968; % rad
smiData.RigidTransform(31).axis = [1 0 0];
smiData.RigidTransform(31).ID = "AssemblyGround[1234-1:1.uc montaj-1:Parmak-2]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(32).translation = [-35.622947894113324 -11.886729262913953
45.709999999998999]; % mm
smiData.RigidTransform(32).angle = 0; % rad
smiData.RigidTransform(32).axis = [0 0 0];
smiData.RigidTransform(32).ID = "AssemblyGround[1234-1:1.uc montaj-1:Parmak tutucu-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(33).translation = [-75.932947894113312 -0.93672926291391012
56.809999999999029]; % mm
smiData.RigidTransform(33).angle = 1.5707963267948968; % rad
smiData.RigidTransform(33).axis = [0 1 0];
smiData.RigidTransform(33).ID = "AssemblyGround[1234-1:1.uc montaj-1:Alt servo mili-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(34).translation = [72.677052105886716 -5.7867292629139033
39.509999999999017]; % mm
smiData.RigidTransform(34).angle = 1.5707963267948968; % rad
smiData.RigidTransform(34).axis = [-1 -2.0675735429298567e-16 3.6049722235143015e-
18];
```

```
smiData.RigidTransform(34).ID = "AssemblyGround[1234-1:1.uc montaj-1:Parmak-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(35).translation = [220.37237433542046 -34.908963906581818
118.34239341326445]; % mm
smiData.RigidTransform(35).angle = 1.6834260659133025; % rad
smiData.RigidTransform(35).axis = [-0 -1 -0];
smiData.RigidTransform(35).ID = "AssemblyGround[1234-1:4.uc montaj-1:Servo Gövde
Baglanti- ic-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(36).translation = [158.64946393537639 -11.208963906581815
97.271742456349614]; % mm
smiData.RigidTransform(36).angle = 1.5739643455272028; % rad
smiData.RigidTransform(36).axis = [0.99683698886154026 -0.056196163736760583
0.05619616373676059];
smiData.RigidTransform(36).ID = "AssemblyGround[1234-1:4.uc montaj-1:Motor yatagi-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(37).translation = [164.11901216486751 -12.608963906581813
123.65363098172382]; % mm
smiData.RigidTransform(37).angle = 1.5739643455272028; % rad
smiData.RigidTransform(37).axis = [0.99683698886154026 -0.056196163736760583
0.05619616373676059];
smiData.RigidTransform(37).ID = "AssemblyGround[1234-1:4.uc montaj-1:Servo 3-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(38).translation = [-86.509503661166221 -30.208963906581822
83.530822650913635]; % mm
smiData.RigidTransform(38).angle = 1.5739643455272025; % rad
smiData.RigidTransform(38).axis = [-0.99683698886154026 -0.056196163736760438 -
0.056196163736760424];
smiData.RigidTransform(38).ID = "AssemblyGround[1234-1:4.uc montaj-1:Servo Govde
Baglanti-Dis-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(39).translation = [-26.055116401708545 4.7910360934181888
90.368098159509188]; % mm
smiData.RigidTransform(39).angle = 0; % rad
smiData.RigidTransform(39).axis = [0 0 0];
smiData.RigidTransform(39).ID = "AssemblyGround[1234-1:4.uc montaj-1:Part1-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(40).translation = [185.23678985993124 29.091036093418186
122.0167237693652]; % mm
smiData.RigidTransform(40).angle = 2.0306356409308468; % rad
smiData.RigidTransform(40).axis = [0.62066855940211363 0.55442336682796167 -
0.55442336682796156];
```

```
smiData.RigidTransform(40).ID = "AssemblyGround[1234-1:4.uc montaj-1:Silindir yatagi-
1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(41).translation   =   [-85.956459361473563   12.695145101578671
355.28353674545036];  % mm
smiData.RigidTransform(41).angle = 0;  % rad
smiData.RigidTransform(41).axis = [0 0 0];
smiData.RigidTransform(41).ID = "AssemblyGround[1234-1:4.uc montaj-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(42).translation   =   [-103.17068782299957   -79.136640221248996
167.49999999999997];  % mm
smiData.RigidTransform(42).angle = 0;  % rad
smiData.RigidTransform(42).axis = [0 0 0];
smiData.RigidTransform(42).ID = "AssemblyGround[5.uc montaj-1:Taban Kutusu-1]";

%Translation Method - Cartesian
%Rotation Method - Arbitrary Axis
smiData.RigidTransform(43).translation   =   [-38.929073665529991   82.979328845298028
396.75013043916886];  % mm
smiData.RigidTransform(43).angle = 0;  % rad
smiData.RigidTransform(43).axis = [0 0 0];
smiData.RigidTransform(43).ID = "RootGround[5.uc montaj-1]";


%============= Solid =============%
%Center of Mass (CoM) %Moments of Inertia (MoI) %Product of Inertia (PoI)

%Initialize the Solid structure array by filling in null values.
smiData.Solid(17).mass = 0.0;
smiData.Solid(17).CoM = [0.0 0.0 0.0];
smiData.Solid(17).MoI = [0.0 0.0 0.0];
smiData.Solid(17).PoI = [0.0 0.0 0.0];
smiData.Solid(17).color = [0.0 0.0 0.0];
smiData.Solid(17).opacity = 0.0;
smiData.Solid(17).ID = "";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(1).mass = 0.029611521877560418;  % kg
smiData.Solid(1).CoM = [9.5500000000000007 13.636645962645135 17.214326629396808];  %
mm
smiData.Solid(1).MoI = [6.1678592341509839 5.1116774683379624 3.2545287983298739];  %
kg*mm^2
smiData.Solid(1).PoI = [-0.35126164639421559 0 0];  % kg*mm^2
smiData.Solid(1).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(1).opacity = 1;
smiData.Solid(1).ID = "Silindir yatagi*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
```

```
smiData.Solid(2).mass = 0.024937093626133791;  % kg
smiData.Solid(2).CoM = [31.299725712380564 13.800024102947988 8.2287796100296688];  %
mm
smiData.Solid(2).MoI  = [2.6393250032155158  17.7359367850807  18.262657181675095];  %
kg*mm^2
smiData.Solid(2).PoI     =     [4.9429641704708252e-06     -5.6250126587197862e-05     -
1.4726452518397729e-05];  % kg*mm^2
smiData.Solid(2).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(2).opacity = 1;
smiData.Solid(2).ID = "Motor yatagi*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(3).mass = 0.10898104098391122;  % kg
smiData.Solid(3).CoM  =  [23.597416795035901  -11.549999077612325  19.984876011071382];
% mm
smiData.Solid(3).MoI  = [18.0656792821067  32.013286684154934  23.587173360831567];  %
kg*mm^2
smiData.Solid(3).PoI       =       [-4.2364785833159239e-08        0.9616809098647966
1.1354518453149676e-06];  % kg*mm^2
smiData.Solid(3).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(3).opacity = 1;
smiData.Solid(3).ID = "Servo 3*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(4).mass = 0.020149443843819292;  % kg
smiData.Solid(4).CoM = [0 0 28.549598342017649];  % mm
smiData.Solid(4).MoI = [5.6840363475008298 5.684035864191336 0.41913882372073769];  %
kg*mm^2
smiData.Solid(4).PoI = [0 0 0];  % kg*mm^2
smiData.Solid(4).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(4).opacity = 1;
smiData.Solid(4).ID = "Ust servo mili*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(5).mass = 0.014220992127361045;  % kg
smiData.Solid(5).CoM = [0 0 20.149618880238148];  % mm
smiData.Solid(5).MoI  =  [2.0725177584191332  2.0725180311808038  0.29581948545532211];
% kg*mm^2
smiData.Solid(5).PoI = [0 0 0];  % kg*mm^2
smiData.Solid(5).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(5).opacity = 1;
smiData.Solid(5).ID = "Alt servo mili*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(6).mass = 0.013306118400000002;  % kg
smiData.Solid(6).CoM  =  [-36.742174980195585  -5.502260301546694  5.2999337688142028];
% mm
```

```
smiData.Solid(6).MoI  =  [0.34956964965716403  5.8558925385555645  5.9743807930014645];
% kg*mm^2
smiData.Solid(6).PoI      =      [0.040724186663958578      0.13186433579196655      -
0.75845444851204413];  % kg*mm^2
smiData.Solid(6).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(6).opacity = 1;
smiData.Solid(6).ID = "Parmak*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(7).mass = 0.049557719648926024;  % kg
smiData.Solid(7).CoM = [19.775628696769431 10.949999999999999 11.1];  % mm
smiData.Solid(7).MoI = [4.3247633659477227 9.5317753472770015 9.8031238268440024];  %
kg*mm^2
smiData.Solid(7).PoI = [0 0 0];  % kg*mm^2
smiData.Solid(7).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(7).opacity = 1;
smiData.Solid(7).ID = "Parmak tutucu*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(8).mass = 0.065926374045396363;  % kg
smiData.Solid(8).CoM = [0 0.00024304066873698794 87.25972555757896];  % mm
smiData.Solid(8).MoI = [111.21233122511325 117.23976214726383 6.5061271038451389];  %
kg*mm^2
smiData.Solid(8).PoI = [-0.0005941193484483786 0 0];  % kg*mm^2
smiData.Solid(8).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(8).opacity = 1;
smiData.Solid(8).ID = "Servo Gövde Baglanti- ic*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(9).mass = 0.10986908781097952;  % kg
smiData.Solid(9).CoM      =      [61.130309923186118      1.1982908528436866e-08      -
8.4292777519373132];  % mm
smiData.Solid(9).MoI = [85.98122049231749 474.48181990836292 539.47304883227582];  %
kg*mm^2
smiData.Solid(9).PoI = [-1.1104142995566452e-08 -0.83703523041988137 0];  % kg*mm^2
smiData.Solid(9).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(9).opacity = 1;
smiData.Solid(9).ID = "Servo Govde Baglanti-Dis*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(10).mass = 0.12417296284264098;  % kg
smiData.Solid(10).CoM       =       [-7.999742189883559e-06       -12.073756296884802
3.4201033478617196e-07];  % mm
smiData.Solid(10).MoI  =  [61.705600823917685  34.851613439548395  61.705613827575242];
% kg*mm^2
smiData.Solid(10).PoI      =      [-1.9993611858752645e-06      5.5558524126061304e-07
4.6765762143538896e-05];  % kg*mm^2
```

```
smiData.Solid(10).color       =       [0.89803921568627454       0.91764705882352937
0.92941176470588238];
smiData.Solid(10).opacity = 1;
smiData.Solid(10).ID = "Part1*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(11).mass = 0.65906529497504329;  % kg
smiData.Solid(11).CoM = [109.00015605113818 71.85000000000008 36.145507438347551];  %
mm
smiData.Solid(11).MoI  =  [2753.1218537032082  4697.7230130517401  5995.1899347756289];
% kg*mm^2
smiData.Solid(11).PoI = [0 0.0037180033958148797 0];  % kg*mm^2
smiData.Solid(11).color       =       [0.89803921568627454       0.91764705882352937
0.92941176470588238];
smiData.Solid(11).opacity = 1;
smiData.Solid(11).ID = "Taban Kutusu*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(12).mass = 0.13985651270084418;  % kg
smiData.Solid(12).CoM = [99.476323951255623 0 0.84999999999999998];  % mm
smiData.Solid(12).MoI  =  [110.51259487210191  1101.4717612060012  1211.9169918578186];
% kg*mm^2
smiData.Solid(12).PoI = [0 0 0];  % kg*mm^2
smiData.Solid(12).color       =       [0.89803921568627454       0.91764705882352937
0.92941176470588238];
smiData.Solid(12).opacity = 1;
smiData.Solid(12).ID = "sorunlu kapak*:*Default";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(13).mass = 1.8712967589946508;  % kg
smiData.Solid(13).CoM       =       [0.0075753335941806119       -0.26757380135695452
1.4535259534553766];  % mm
smiData.Solid(13).MoI = [4875.681130500695 5005.3575573085063 1293.4746505899889];  %
kg*mm^2
smiData.Solid(13).PoI  =  [177.57549747028159  -5.0273742222435267  -3.674239366853123];
% kg*mm^2
smiData.Solid(13).color       =       [0.89803921568627454       0.91764705882352937
0.92941176470588238];
smiData.Solid(13).opacity = 1;
smiData.Solid(13).ID = "PARÇA 1*:*Default";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(14).mass = 0.23911124611642323;  % kg
smiData.Solid(14).CoM = [101.17264109817837 0 7.1217080785631106];  % mm
smiData.Solid(14).MoI  =  [322.86884883442696  2229.0126633569653  2512.2910479122047];
% kg*mm^2
smiData.Solid(14).PoI = [0 -3.7081569651287096 0];  % kg*mm^2
smiData.Solid(14).color       =       [0.89803921568627454       0.91764705882352937
0.92941176470588238];
smiData.Solid(14).opacity = 1;
smiData.Solid(14).ID = "sorunlu gövde*:*Default";
```

```
%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(15).mass = 1.0670394545782083;  % kg
smiData.Solid(15).CoM        =        [-0.19908778784832953        -4.6912800100078895
49.999999999999993];  % mm
smiData.Solid(15).MoI  =  [1167.8076666545851  1307.3426175926397  696.75119328354413];
% kg*mm^2
smiData.Solid(15).PoI = [0 0 5.9322456423589456];  % kg*mm^2
smiData.Solid(15).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(15).opacity = 1;
smiData.Solid(15).ID = "Parça2*:*Default";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(16).mass = 1.0655970141421975;  % kg
smiData.Solid(16).CoM  =  [127.46831606461944  -233.11922291267189  15.100000000000001];
% mm
smiData.Solid(16).MoI = [876.4475018350206 1712.1134051590711 2426.5830568643828];  %
kg*mm^2
smiData.Solid(16).PoI = [0 0 2.0632427905452233];  % kg*mm^2
smiData.Solid(16).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(16).opacity = 1;
smiData.Solid(16).ID = "sorunlu gövdenin parçasi*:*Varsayilan";

%Inertia Type - Custom
%Visual Properties - Simple
smiData.Solid(17).mass = 0.23556943226446572;  % kg
smiData.Solid(17).CoM = [0 0 4.4206662224268634];  % mm
smiData.Solid(17).MoI  =  [149.99729820411102  149.99729820411102  41.254013438012514];
% kg*mm^2
smiData.Solid(17).PoI = [0 0 0];  % kg*mm^2
smiData.Solid(17).color        =        [0.89803921568627454        0.91764705882352937
0.92941176470588238];
smiData.Solid(17).opacity = 1;
smiData.Solid(17).ID = "Parça4*:*Default";


%============= Joint =============%
%X Revolute Primitive (Rx) %Y Revolute Primitive (Ry) %Z Revolute Primitive (Rz)
%X Prismatic Primitive (Px) %Y Prismatic Primitive (Py) %Z Prismatic Primitive (Pz)
%Spherical Primitive (S)
%Constant Velocity Primitive (CV) %Lead Screw Primitive (LS)
%Position Target (Pos)

%Initialize the RevoluteJoint structure array by filling in null values.
smiData.RevoluteJoint(6).Rz.Pos = 0.0;
smiData.RevoluteJoint(6).ID = "";

smiData.RevoluteJoint(1).Rz.Pos = 6.0725495509634104;  % deg
smiData.RevoluteJoint(1).ID = "[1234-1:4.uc montaj-1:-:1234-1:3. uc montaj-1]";

smiData.RevoluteJoint(2).Rz.Pos = -89.809922964412692;  % deg
```

```
smiData.RevoluteJoint(2).ID = "[1234-1:3. uc montaj-1:-:1234-1:2. uc montaj-1]";

smiData.RevoluteJoint(3).Rz.Pos = 0.90028121367675518;  % deg
smiData.RevoluteJoint(3).ID = "[1234-1:2. uc montaj-1:-:1234-1:1.uc montaj-1]";

smiData.RevoluteJoint(4).Rz.Pos = -179.91770412217841;  % deg
smiData.RevoluteJoint(4).ID  =  "[5.uc  montaj-1:sorunlu  gövdenin  parçasi-1:-:5.uc
montaj-1:Taban Kutusu-1]";

smiData.RevoluteJoint(5).Rz.Pos = -0.58879599498323687;  % deg
smiData.RevoluteJoint(5).ID  =  "[5.uc  montaj-1:sorunlu  gövdenin  parçasi-2:-:5.uc
montaj-1:sorunlu gövde-1]";

smiData.RevoluteJoint(6).Rz.Pos = -176.34523330177896;  % deg
smiData.RevoluteJoint(6).ID  =  "[5.uc  montaj-1:sorunlu  gövde-1:-:1234-1:4.uc  montaj-
1]";
```