DESIGN AND CONTROL OF SERIAL ROBOT MANIPULATOR REPORT

ABSTRACT

In today's industrial production facilities, robot arms are produced with hydraulic, pneumatic, or electric systems and are used in many different sectors. These programmable mechanical components reduce need of human labour, minimize error rates, and increase productivity. Robot arms can either execute pre-programmed commands or be controlled by an operator. The industrial and medical sectors are the areas where robot arms are most commonly used.

In this project, a robot arm was designed and built with the ability to move in 5 axes using 5 servo motors. The gripper located at the front of the arm allows for the transportation of materials, and the movements of the robot can be stored in memory using the save command. The robot can be controlled using an Android application through an Arduino Uno and Bluetooth module.

ÖZET

Bugünün endüstriyel üretim tesislerinde, robot kollar hidrolik, pnömatik veya elektrikli sistemlerle üretilerek, birçok farklı sektörde kullanılmaktadır. Bu programlanabilir mekanik parçalar, insan gücünü azaltarak hata payını azaltır ve üretkenliği artırır. Robot kollar, önceden programlanmış komutları yerine getirebilir veya bir kullanıcı tarafından kontrol edilebilir. Endüstri ve tıp sektörleri, robot kollarının en çok kullanıldığı alanlardır.

Bu projede, 5 adet servo motor kullanarak 5 eksen hareket kabiliyetine sahip bir robot kol tasarlandı ve inşa edildi. Ön kısımdaki tutucu sayesinde malzemeler taşınabilir ve kaydetme komutu kullanılarak robot hareketleri hafızaya alınabilir. Arduino Uno ve Bluetooth modülü aracılığıyla Android uygulaması ile robot kontrol edilebilir.

TABLE OF CONTENTS

ABSTRACT ..... **ii**

ÖZET ..... **iii**

LIST OF FIGURES ..... **vi**

LIST OF TABLES ..... **vii**

LIST OF SYMBOLS ..... **viii**

1. INTRODUCTION **1**
- 1.1. History Of Robotic Arms 2
- 1.2. Functions Of Robotic Arms 2
- 1.3. Types Of Robotic Arms 2
- 1.4. Advantages And Disadvantages Of Robotic Arms 4

2. METHODS **5**
- 2.1. ROBOTIC ARM CONTROL 5
- 2.2. ROBOTIC ARM KINEMATICS 6
  - 2.2.1. General Features of Robot Arm Kinematics 6
  - 2.2.2. Coordinate Systems and Transformation Matrices for a Robotic Arm 6
  - 2.2.3. D & H (Denavit and Hartenberg) Coordinate Frames 9
  - 2.2.4. Forward Kinematics 9
  - 2.2.5. Inverse Kinematics 14
- 2.3. THEORETICAL PART 14
- 2.4. Servo Motors 14
  - 2.4.1. Technical Specifications of Servo Motors 15
- 2.5. Arduino Uno 16
- 2.6. HC-06 Bluetooth Module 17
  - 2.6.1. Technical Specifications of HC-06 18
- 2.7. Circuit Schematic 19
- 2.8. Power 20
  - 2.8.1. HW-319 Voltage Regulator 20

3. IMPLEMENTATION AND RESULTS **21**

3.1. Designed System 21

3.2. Assembled System 23

3.3. Circuit 24

3.4. MATLAB Modelling and Simulation 24

3.5. Android Programming 26

4. CONCLUSION 30

REFERENCES 31

APPENDIX A 33

LIST OF FIGURES

Figure 1. Example of Serial Robotic Arm 2

Figure 2. Example of Parallel Robotic Arm 3

Figure 3. Example of Hybrid Robotic Arm 3

Figure 4. Block Diagram of System 5

Figure 5 n-jointed serial robot arm represented by relative angles. 8

Figure 6 D & H Parameters 9

Figure 7. MATLAB Code1 11

Figure8. MATLAB Code2 12

Figure 9. MATLAB Code3 13

Figure 10. SG90 Servo Motor 15

Figure 11. ES 3005 Servo Motor 15

Figure 12. Arduino Uno 16

Figure 13. HC-06 Bluetooth Module 18

Figure 14. Circuit Schematic 19

Figure 15. Power Adaptor 20

Figure 16. HW-319 Voltage Regulator 21

Figure 17. Waist Part 22

Figure 18. Base Part 22

Figure 19. Shoulder Part 22

Figure 20. Elbow Part 22

Figure 21. Wrist Roll Part 22

Figure 22. Gripper Part 22

Figure 23. Assembled System 23

Figure 24. Robot Arm Circuit 24

Figure 25. MATLAB Modelling 25

Figure 26. Simulink System Dynamic Model 26

Figure 27. Android Application 27

Figure 28. APP Inventor Blocks 29

LIST OF TABLES

Table 1. SG90 Technical Specifications 15

Table 2. ES 3005 Technical Specifications 15

Table 3 Technical Specifications of HC-06 18

Table 4. Technical Specifications of HW-319 20

LIST OF SYMBOLS

A	Matrix
${}^{R}\mathbf{T}_{H}$	Transformation matrix
V	Voltage
gr	Gram
kg	Kilogram
cm	Centimeter
A	Ampere
$\boldsymbol{l_n}$	Joint Length
$\boldsymbol{a_i}$	distance from Z_i to Z_{i+1} measured along X_i
$\boldsymbol{\alpha_i}$	angle from Z_i to Z_{i+1} measured along X_i
$\boldsymbol{d_i}$	distance from X_{i-1} to X_i measured along Z_i
$\boldsymbol{\theta_i}$	angle from X_{i-1} to X_i measured along Z_i

1. INTRODUCTION

Throughout history, humans have needed additional assistant systems for a more efficient and effective production process. With easy access to information, increasing competition, and the need to reduce costs, automation systems have become increasingly widespread today. Standardized automation systems, in addition to well-trained employees, help reduce error rates and enable the production of high-quality products. Moreover, the use of auxiliary machines has become inevitable when humans' physical strength is insufficient. These auxiliary machines can operate without the need for human labour as technology advances. Robotic systems have emerged as a result of these developments and have been developed through the joint efforts of mechatronics, mechanical, electrical-electronics, and computer engineering disciplines.

In this project, research has been conducted on the mechanical and software aspects of the robot arm designed to perform the predetermined task, and applications have been implemented. Thus, the process of building a robot arm that can perform its task according to specific commands is described.

In the first step of the project, the intended function and range of motion for the robotic arm were determined. The robotic arm, which can be controlled via an Android phone or tablet, is capable of carrying, mixing, and executing pre-set commands. The assigned task for this project is for the robotic arm to pick up a material and move it to a desired position, record its movements, and repeat the same action until stopped. To ensure the proper execution of these actions, a precise and high torque motor was necessary, and thus a servo motor was chosen. The robotic arm consists of 5 servo motors, enabling it to move in 5 axes.

In the project, the servo motor control was achieved by programming the Arduino Uno microcontroller. This eliminated the need for any additional circuitry beyond the one containing the servo motor inputs, as the necessary components were already available on the Arduino board. The mechanical design of the robotic arm was created using SolidWorks software, specifying the dimensions of the robot's limbs. Finally, a 12V 2A power supply was chosen to enable the robot's functionality.

1.1. History Of Robotic Arms

Robot arms were first used in industrial automation applications in the 1950s. The introduction of the first industrial robot arm called Unimate by General Motors in 1961 marked a turning point in the development of robot arms. Since then, robot arms have been widely used in various sectors, particularly in the automotive industry.

1.2. Functions Of Robotic Arms

Robot arms can be used to fulfil a range of different functions, including lifting heavy loads, material handling, assembly operations, welding, packaging, precise manipulation, surgical operations, and more. Equipped with precision sensors, actuators, and control systems, robot arms can automate complex movements and perform a wide variety of tasks.

1.3. Types Of Robotic Arms

Robot arms can be categorized into different types based on their structural features, degrees of freedom, and application areas. Here are some commonly used types of robot arms:

- Serial Robot Arms: This type of robot arms operates using multiple interconnected linkages. They typically consist of a base and sequential linkages. Serial robot arms provide high precision and flexibility, making them commonly used in assembly, packaging, welding, and other operations.

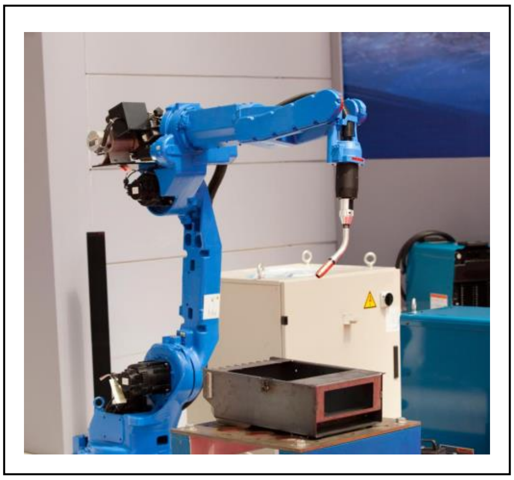


Figure 1. Example of Serial Robotic Arm

- Parallel Robot Arms: Parallel robot arms operate using multiple parallel linkages. This structure provides high carrying capacity, speed, and rigidity. Parallel robot arms are used in applications requiring precise assembly, surgical operations, and simulations.

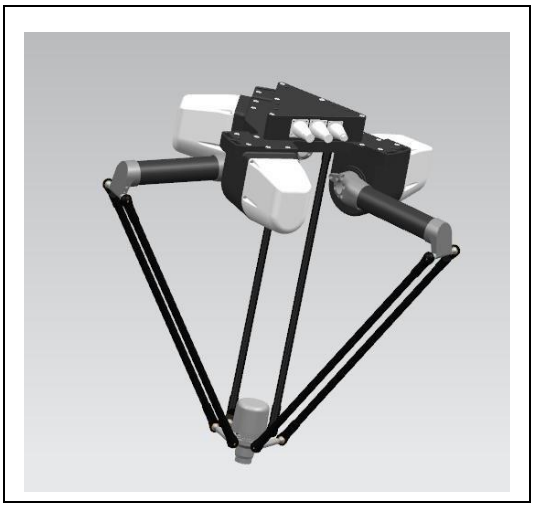


Figure 2. Example of Parallel Robotic Arm

- Hybrid Robot Arms: Hybrid robot arms combine elements of both serial and parallel structures. These robot arms offer both high precision and high carrying capacity, making them useful for applications that require both capabilities. For example, they can be used in industrial assembly lines to lift heavy loads and perform precise assembly tasks.

Figure 3. Example of Hybrid Robotic Arm

1.4. Advantages And Disadvantages Of Robotic Arms

Robot arms bring several advantages:

- Automation and Efficiency: Robot arms can perform repetitive tasks at high speed and precision, leading to automation and increased efficiency in production processes.
- Human Safety: Robot arms can be used to protect humans in dangerous or challenging working conditions. For example, in nuclear power plants or environments with hazardous chemicals, robot arms can reduce the risk of human exposure.
- High Precision: Equipped with precision sensors and control systems, robot arms can operate with micron-level precision. This is a significant advantage for tasks requiring precise assembly or surgical operations.
- Flexibility: Robot arms are programmable and can be easily adapted to different tasks. This provides great flexibility in production processes when changes are needed.

Robot arms also have some disadvantages:

- High Cost: Robot arms are often costly investments. The hardware itself, software, and training expenses contribute to the overall cost. As a result, they may not be accessible for small-scale businesses.
- Human Skill Requirement: Programming and maintaining robot arms may require specialized skills. This can be a challenge if there is a lack of human expertise in these areas.
- Limited Flexibility: Some robot arms may be designed for specific tasks and may not be suitable for different operations. In such cases, reconfiguration or replacement may be necessary for new tasks.

2. METHODS

Firstly, historical research on robot arms was conducted and it was determined that the robot arm to be used in the project is of articulated type and can move in 5 axes, as well as being able to perform holding and swinging movements with the help of its gripper. The Arduino Uno microcontroller was chosen as the most suitable option to control the robot arm, as it is easier to use compared to other microcontrollers and has an open-source code. The high number of users of this microcontroller makes it easier to get help in case of any errors. Additionally, detailed information was obtained about the servo motors to be used in the project, and it was decided that the robot arm will consist of 5 servo motors to ensure its precise and high torque operation. Thanks to these efforts, it was planned that the robot can perform its tasks in the project properly and efficiently.

2.1. ROBOTIC ARM CONTROL

The mechanical part of the robot arm was designed by assembling 3D-printed parts appropriately. An Arduino microcontroller was selected and software was developed to move the robot arm according to its intended purpose. Subsequently, experiments were conducted with the Bluetooth module and servo motors to gather information about the system's operation. The software was developed using the suitable Arduino microcontroller to control the robot arm and make it move according to its intended purpose.

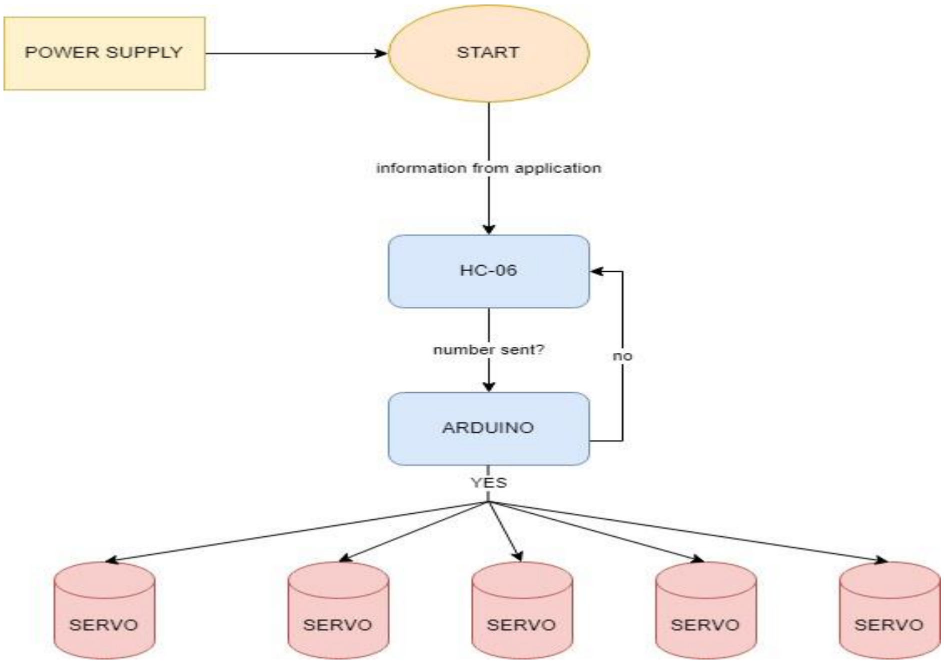


Figure 4. Block Diagram of System

2.2. ROBOTIC ARM KINEMATICS

2.2.1. General Features of Robot Arm Kinematics

The study of motion is a component of kinematics in robotics. Relative to a reference coordinate frame, a robot arm's parts can rotate or translate. Denavit and Hartenberg created a methodical and comprehensive method that establishes the link between the displacements of the robot's arm's individual parts, represented by joint coordinates determined by joint variables, and the end effector's overall displacement. The A matrices that reflect the rotation and translation of each component are successively multiplied to provide the end effector's rotational and translational quantities relative to the reference coordinate system. Given the coordinates of the end effector, the joint variables may be found by going back through the procedure. The terms "forward" and "inverse" kinematics" refer to these processes.(Koivo,1989)

2.2.2. Coordinate Systems and Transformation Matrices for a Robotic Arm

Homogeneous coordinate representation is the representation of an n-dimensional position vector with an n+1-dimensional vector. A location vector is expressed in homogeneous coordinates between coordinate frames in the 4x4 matrix below.(Fu,1987)

$$ {}^{R}\mathbf{T}_{H} = \begin{bmatrix} x_x & y_x & z_x & p_x \\ x_y & y_y & z_y & p_y \\ x_z & y_z & z_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1) $$

The 4th column of a transformation matrix consists of 3 elements that correspond to the translation round the x, y, and z axes.

$$ Trans(p_x, p_y, p_z) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2) $$

There are three rotation transformations corresponding to rotations by q angle around the x, y, and z axes, since rotation is possible around any of the three coordinate axes. The following matrix can be written for the x, y and z-axis:

$$ Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (3) $$

$$Rot(y,\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

$$Rot(z,\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (5)$$

Pure rotation and translation matrices can be multiplied repeatedly to calculate the elements of the transformation matrix. A sequence of rotations around the axes of the fixed reference frame may be used to determine the orientation of the end effector in the reference frame within Cartesian space. Although there are various ways to do this, one option that is frequently employed is called the "roll-pitch-yaw" transformation. To do this, you must first determine the rotation around the x-axis, then the rotation around the y-axis, and lastly the rotation around the z-axis.

$$RPY(\phi,\theta,\psi) = Rot(z,\phi)Rot(y,\theta)Rot(x,\psi) \qquad (6)$$

$$= \begin{bmatrix} C(\phi)C(\theta) & C(\phi)S(\theta)S(\psi) - S(\phi)C(\psi) & C(\phi)S(\theta)C(\psi) + S(\phi)S(\psi) & 0 \\ S(\phi)C(\theta) & S(\phi)S(\theta)S(\psi) + C(\phi)C(\psi) & S(\phi)S(\theta)C(\psi) - C(\phi)S(\psi) & 0 \\ -S(\theta) & S(\psi)C(\theta) & C(\psi)C(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (7)$$

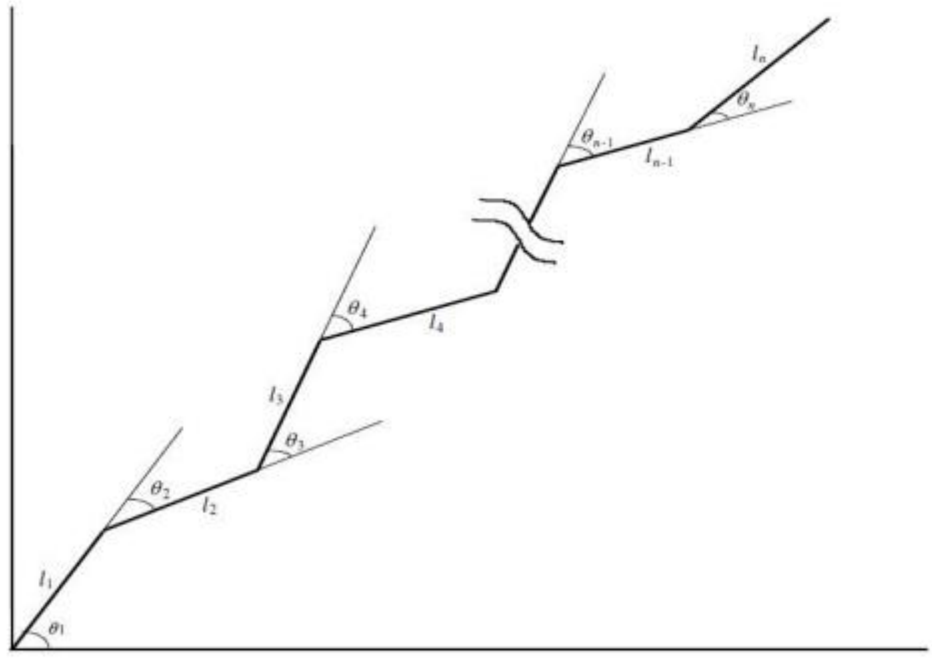


Figure 5 n-jointed serial robot arm represented by relative angles.

(GEREĞİNDEN ÇOK SERBESTLİK DERECELİ ROBOT KOLU KONTROL SİSTEMİ TASARIMI VE UYGULAMASI YÜKSEK LİSANS TEZİ)

When examining the serial-link robot arm shown in Figure 5, it becomes clear that changing one link's location and orientation also alters the positions and orientations of the other links that are located between that link and the end effector. Typically, it is necessary to ascertain the location and orientation of the end effector in relation to the reference frame of the base when the positions and orientations of the connections change.(Koivo ,1989)

2.2.3. D & H (Denavit and Hartenberg) Coordinate Frames

Denavit-Hartenberg (D&H) coordinate frames are commonly used in robotics to define the position and orientation of each link in a robotic arm or manipulator. The D&H method is based on assigning a coordinate frame to each joint in the robot and defining the transformation between adjacent frames using a set of parameters. This allows for a systematic and consistent method of describing the kinematics of a robot, which is important for tasks such as inverse kinematics and trajectory planning.

In order to represent the position of each link in relation to its neighboring links, it is essential to establish a frame that is attached to each link. These link frames, as depicted in Figure 6, are named based on the corresponding link number; in other words, frame i is rigidly attached to link i. Furthermore, a Cartesian frame $X_i\ Y_i\ Z_i$should be assigned to each joint to define the position and orientation between the interconnected links. (Forward Kinematics Analysis of a 5-Axis RV-2AJ Robot Manipulator. Journal of Robotics,3(2),45-56.)

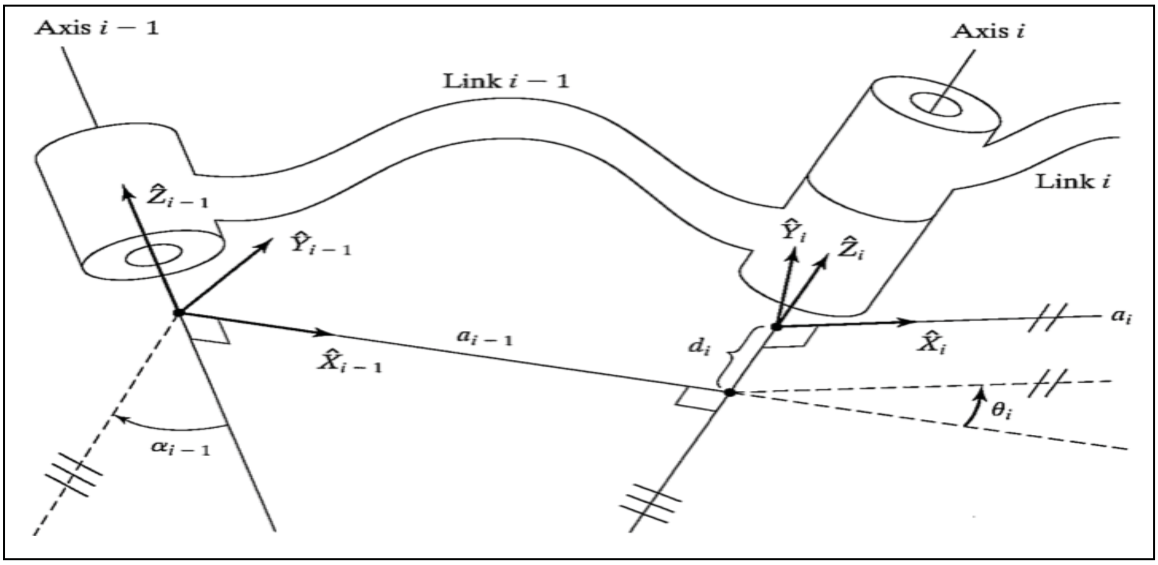


Figure 6 D & H Parameters **(Forward Kinematics Analysis of a 5-Axis RV-2AJ Robot Manipulator. Journal of Robotics,3(2),45-56.)**

2.2.4. Forward Kinematics

Forward kinematics is a mathematical model that calculates the position and orientation of the end effector of a robotic arm starting from the joint configuration. These calculations are typically performed using a 4x4 homogeneous transformation matrix. The transformation matrix contains rotation and translation information for each joint. RPY (roll, pitch, yaw) angles, on the other hand, are a system used to express the orientation of an object. RPY angles are associated with a rotation matrix and are used in conjunction with it. In summary, forward kinematics is a mathematical model used to find the position and orientation of a robotic arm, utilizing a transformation matrix and RPY angles. (GEREĞİNDEN ÇOK SERBESTLİK DERECELİ ROBOT KOLU KONTROL SİSTEMİ TASARIMI VE UYGULAMASI YÜKSEK LİSANS TEZİ)

Transformation matrix is shown below:

$$ {}^{R}\mathbf{T}_{H} = \begin{bmatrix} x_x & y_x & z_x & p_x \\ x_y & y_y & z_y & p_y \\ x_z & y_z & z_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8) $$

The necessary D & H parameters for the geometric analysis of the robot were determined by examining the geometric structure of the robot based on our available information.

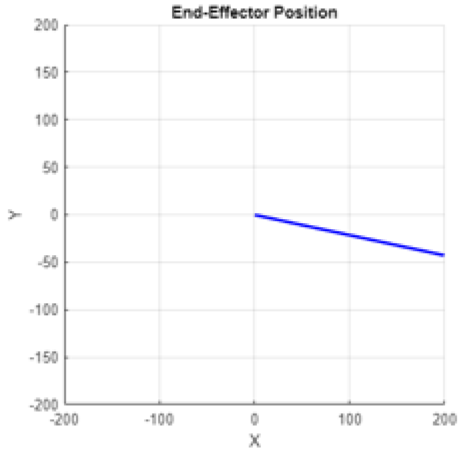
Using the known parameters, position data was obtained using MATLAB. As a result, the forward kinematic analysis of the robot was conducted.

The MATLAB codes are shown below:

```
% Given DH parameters
a = [0, 162, 115, 140, 0];
alpha = [pi/2, 0, 0, 0, 0];
d = [39, 120, 86, 65, 0];
theta = [base_rad, shoulder_rad, elbow_rad, wristroll_rad, gripperucu_rad];

% Initial State matrix
T_0_1 = [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];

% Total transformation matrix calculation
T_total = T_0_1;
for i = 2:length(a)
    A = [cos(theta(i)),  -sin(theta(i))*cos(alpha(i)),
sin(theta(i))*sin(alpha(i)),  a(i)*cos(theta(i));
         sin(theta(i)),  cos(theta(i))*cos(alpha(i)),   -
cos(theta(i))*sin(alpha(i)), a(i)*sin(theta(i));
         0,              sin(alpha(i)),                cos(alpha(i)),
d(i);
         0,              0,                            0,
1];

    T_total = T_total * A;
end

% End-effector location
end_effector_position = T_total(1:3, 4);

% Plotting
figure;
hold on;
grid on;
axis equal;
xlabel('X');
ylabel('Y');
zlabel('Z');
title('End-Effector Position');
% Drawing the positions of the joints
P0 = [0; 0; 0];
P1 = T_0_1(1:3, 4);
P2 = T_total(1:3, 4);
plot3([P0(1), P1(1)], [P0(2), P1(2)], [P0(3), P1(3)], 'r', 'LineWidth', 2);
plot3([P1(1), P2(1)], [P1(2), P2(2)], [P1(3), P2(3)], 'b', 'LineWidth', 2);
plot3(P2(1), P2(2), P2(3), 'ko', 'MarkerSize', 6, 'MarkerFaceColor', 'g');
```

Figure 7. MATLAB Code1

```
% Plot the end-effector position
plot3(end_effector_position(1), end_effector_position(2),
end_effector_position(3), 'ro', 'MarkerSize', 8, 'MarkerFaceColor', 'r');
% Determining the dimensions of the arm
axis([-200, 200, -200, 200, -200, 200]);
```




```
% Given joint angles (degrees)
base = 90;
shoulder = 150;
elbow = 90;
wristroll = 120;
gripper = 180;

% Converting joint angles in radians
base_rad = deg2rad(base);
shoulder_rad = deg2rad(shoulder);
elbow_rad = deg2rad(elbow);
wristroll_rad = deg2rad(wristroll);
gripper_rad = deg2rad(gripper);
% DH parameters
a = [0, 162, 115, 140, 0];
alpha = [pi/2, 0, 0, 0, 0];
d = [39, 120, 86, 65, 0];
theta = [base_rad, shoulder_rad, elbow_rad, wristroll_rad, gripper_rad];
% Forward Kinematics Matrix Create
T = eye(4); % Initial State
T_total = zeros(4, 4, length(a));

for i = 1:length(a)
    A = [cos(theta(i)), -sin(theta(i))*cos(alpha(i)),
sin(theta(i))*sin(alpha(i)), a(i)*cos(theta(i));
          sin(theta(i)), cos(theta(i))*cos(alpha(i)), -
cos(theta(i))*sin(alpha(i)), a(i)*sin(theta(i));
          0, sin(alpha(i)), cos(alpha(i)), d(i);
```

Figure8. MATLAB Code2

```
        0, 0, 0, 1];

    T = T * A; % Total transformation matrix
    T_total(:,:,i) = T; % Storing the transformation matrix
end
% End-effektors location ve orientation
end_effector_position = T(1:3,4);
end_effector_orientation = T(1:3,1:3);
% Printing the results to the screen
disp("End-effector Location:");
```

```
End-effector Location:
```

```
disp(end_effector_position);
```

```
  271.0000
  -57.7961
   20.4071
```

```
disp("End-effector Orientation:");
```

```
End-effector Orientation:
```

```
disp(end_effector_orientation);
```

```
   -0.0000    0.0000    1.0000
   -1.0000   -0.0000   -0.0000
    0.0000   -1.0000    0.0000
```

Figure 9. MATLAB Code3

We calculated the necessary D&H parameters on Figure 7 by defining them as variables in MATLAB for our own robot. Then we defined the initial condition matrix and calculated the total transformation matrix. As a result of these calculations, we determined the position of the end effector. On Figure 8 and Figure 9, using the same D&H parameters, we performed the forward kinematics calculations of our robot in MATLAB. Thus, we found the position and orientation of the end effector.

In conclusion, by practically using the mentioned A matrices, transformation matrices, and RPY angles in MATLAB, we approximately determined the position and orientation of the end effector of our robot. This method prevents potential errors that could have been made manually during the forward kinematics calculations of the robot. However, while these calculations have advantages, they can also have disadvantages. Since these calculations were done theoretically in a simulation environment, errors may be encountered when applied in real-life scenarios. This method can be validated by performing it in a simulation environment for verification after manual calculations are done in real-life applications.

2.2.5. Inverse Kinematics

Inverse kinematics is the process of determining the joint parameters (angles, positions, etc.) required to position the end-effector (or tool) of a robotic manipulator at a desired location and orientation in the workspace. It is the opposite of forward kinematics, which calculates the position and orientation of the end-effector based on the joint parameters. Inverse kinematics is an important concept in robotics, particularly in applications such as motion planning and control, path optimization, and robotics simulation.

2.3. THEORETICAL PART

This section contains technical information about the electronic components used for the robotic arm. It includes the circuit diagram and details about the power supply.

2.4. Servo Motors

Servo motors are commonly used devices for controlling mechanical movements without the need for complicated feedback systems. They are widely employed in applications that require precise control. A typical servo motor consists of a motor, a feedback sensor, and a control system.

Servo motors find application across a wide range of industrial and commercial sectors. They are commonly used in robotics systems, automation equipment, CNC machines, autonomous vehicles, camera stabilization systems, and many other fields. Additionally, they are prevalent in toys, remote-controlled (RC) vehicles, model aircraft, and drones.

PWM (Pulse Width Modulation) is often used for controlling servo motors. PWM works by varying the pulse width and pulse period to control the motor's position and speed. The control signal is typically generated by a microcontroller or a control board and transmitted to the control circuitry on the servo motor.Tower Pro SG90 Mini and ES 3005 Servo motors is used in this project. SG90 servo motors for Gripper part and ES 3005 servo motors is used for the rest (base, waist, arm).

Figure 10. SG90 Servo Motor

Figure 11. ES 3005 Servo Motor

The SG90 and ES 3005 are popular and widely used servo motor models. The SG90 servo motor is compact in size and commonly used in hobby projects. It operates at a voltage range of typically 4.8V to 6V and offers a rotation angle of 180 degrees. Its speed can reach approximately 60 degrees in 0.1 seconds, and it can generate a torque of approximately 1.5 kg/cm.

The ES 3005 servo motor, on the other hand, features a more powerful design. It operates at a voltage range of 4.8V to 6V. It has a 360-degree rotation capability, allowing it to be used in continuous rotation mode. Its speed can reach approximately 60 degrees in 0.16 seconds, and it can generate a torque of approximately 7.2 kg/cm.

Both servo motors typically come with 3-pin connectors, enabling easy connectivity. Due to their ability to provide precise positioning with appropriate control signals, these servo motors are widely used and preferred models in various applications.

2.4.1. Technical Specifications of Servo Motors

Weight	9gr
Operating Voltage	4.8V~6.0V
Gear Type	Plastic
Stall Torque @4.8V	1.2kg-cm
Stall Torque @6.0V	1.8kg-cm

Table 1. SG90 Technical Specifications

Weight	42gr
Operating Voltage	4.8V~6.0V
Gear Type	Metal
Stall Torque @4.8V	10kg-cm
Stall Torque @6.0V	12kg-cm

Table 2. ES 3005 Technical Specifications

2.5. Arduino Uno

Arduino is a popular platform for hobby electronics and prototyping projects. It is known for its ease of use, extensive community support, and a rich ecosystem. Arduino is a development board based on a microcontroller and allows users to interact with electronic components and develop various projects.

Arduino can be used for a wide range of applications. It can perform basic tasks such as reading sensor data, controlling motors, and turning on LEDs, as well as more complex project development. It is preferred in fields such as robotics, automation, home automation, wearable technologies, and art projects.

Arduino Uno, one of the most recognized and widely used models in the Arduino family, has the following advantages and features:

- Easy to Use: Arduino Uno features a user-friendly interface and a simple programming language, enabling users to quickly start their projects.
- Extensive Support and Resources: The Arduino community is vast and provides numerous resources, forums, and example projects. This makes it easy for users to find information and assistance.
- Various Input/Output Pins: Arduino Uno has digital and analog input/output pins, allowing flexibility in interacting with various sensors, motors, and other components.
- Wide Connectivity Options: Arduino Uno offers different connectivity options such as USB connection and support for external power sources, enabling projects to be configured based on different needs.

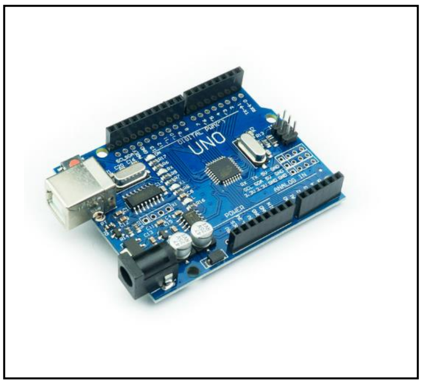


Figure 12. Arduino Uno

Arduino Uno utilizes the ATmega328P microcontroller. This microcontroller features an 8-bit RISC architecture and includes 32 KB of flash memory, 2 KB of SRAM, and 1 KB of EEPROM. It also provides various features such as different input/output pins, timers, serial communication interfaces, and analog-to-digital converters.

With its user-friendly design, extensive community support, flexible connectivity options, and a powerful microcontroller, Arduino Uno provides an ideal platform for developing advanced electronic projects.

2.6. HC-06 Bluetooth Module

The HC-06 Bluetooth module is a popular device used for wireless communication. It is known for its low cost and ease of use. The HC-06 module operates through a serial communication protocol called UART and enables wireless communication via a Bluetooth connection.

The HC-06 Bluetooth module can be integrated with Arduino and other microcontrollers to provide wireless communication in projects. The module can easily communicate with Arduino using the serial communication interface. Data exchange occurs through the TX (transmit) and RX (receive) pins.

Some advantages of the HC-06 module include:

- Easy Integration: The HC-06 module uses a simple serial communication protocol and can be easily integrated with microcontrollers like Arduino.
- Wireless Connectivity: It provides wireless connectivity through Bluetooth technology, eliminating the need for wired connections for data transmission.
- Cost-effective: The HC-06 module is a low-cost Bluetooth module, making it economically feasible for various projects.

The working principle of the HC-06 Bluetooth module is as follows: The module connects to Arduino or a microcontroller through the serial communication protocol. Data is sent to the module using the UART protocol and transmitted wirelessly via the Bluetooth connection. The module can wirelessly exchange data with paired devices.

The HC-06 Bluetooth module typically operates at a voltage of 3.3V and requires a power supply between 3.3V and 6V. The communication baud rate can be set to 9600 bps (bits per second), and the maximum communication range is generally up to 10 meters.

With this information, you can understand the basic working principles, advantages, and how the HC-06 Bluetooth module can be integrated with Arduino. This module expands the wireless connectivity capabilities of Arduino-based applications, particularly in projects requiring wireless communication.

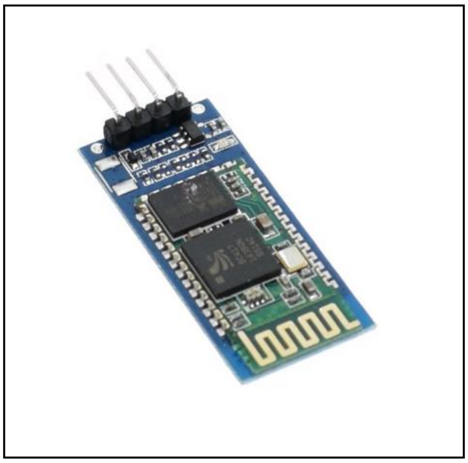


Figure 13. HC-06 Bluetooth Module

2.6.1. Technical Specifications of HC-06

Chipset	CSR
Working Voltage	3.3V
Current	Pairing 20~30mA, connected 8mA
Weight	5g

Table 3 Technical Specifications of HC-06

2.7. Circuit Schematic

The circuit consists of servo inputs, Arduino pin inputs, and a Bluetooth module input. The connections between the servo motors, Bluetooth module, and Arduino Uno, as well as the power supply connections, are shown in the diagram.

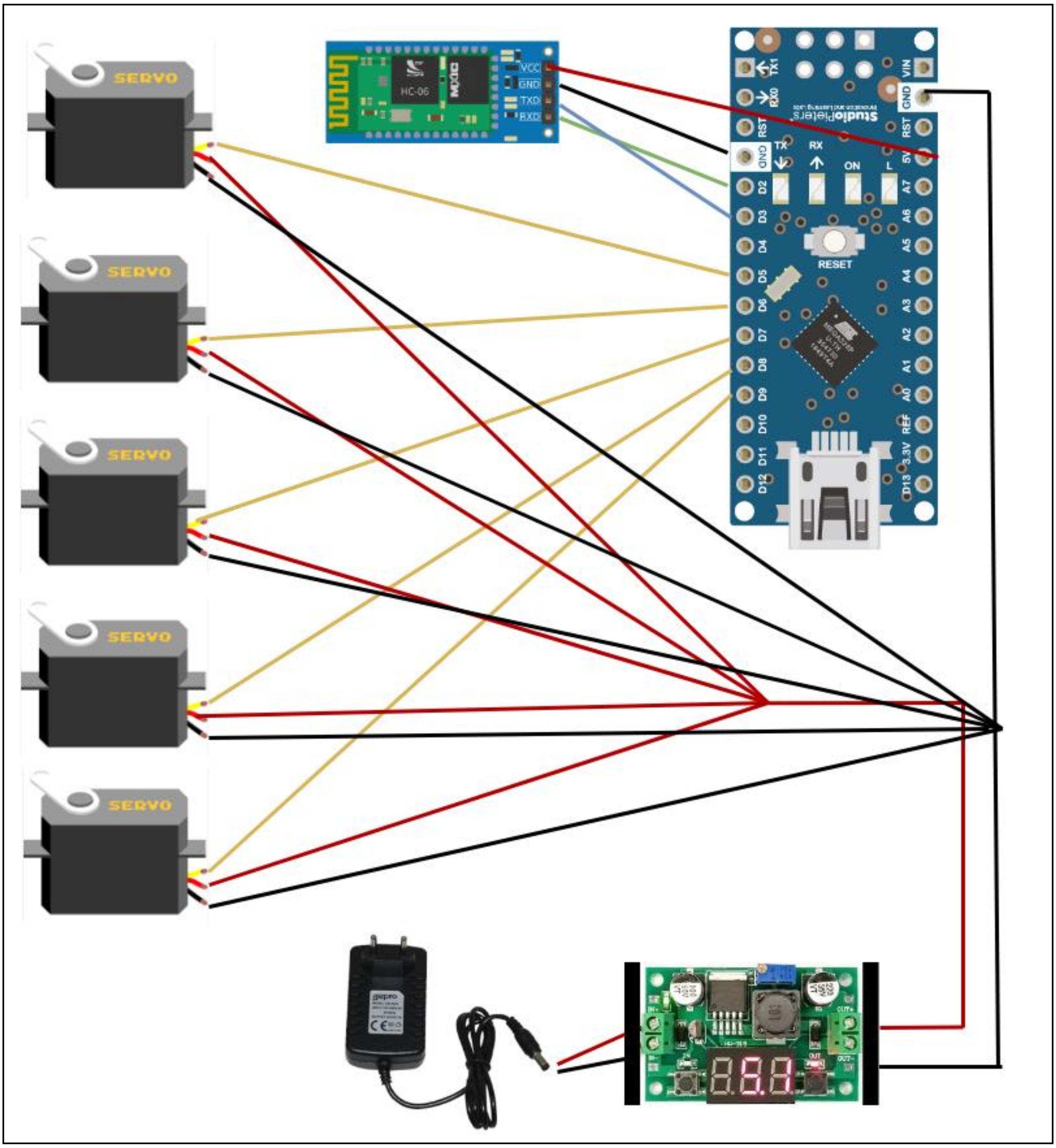


Figure 14. Circuit Schematic

2.8. Power

12V 2A power supply is used in this project because servo requires 4.8-6.0V. Also, Arduino is not enough to provide necessary current for all servos so that external power supply is used for project.

To step down 12V to 5V , used HW-319 Voltage Regulator Card.

Figure 15. Power Adaptor

2.8.1. HW-319 Voltage Regulator

Since the power adapter used is 12V and the stable operating range for the servos is 5V, an HW-319 voltage regulator board was used to reduce the voltage from 12V to 5V. Additionally, considering that the servos can draw a current of up to 2A when under load, this board was chosen accordingly.

Input Voltage	4-40V
Output Voltage	1.25-37V
Output Current	3A(max)
Dimensions	56x35x14mm

Table 4. Technical Specifications of HW-319

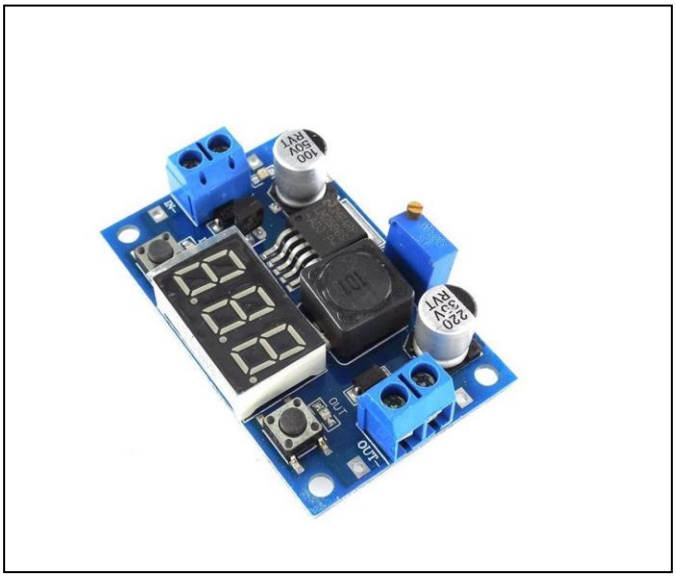


Figure 16. HW-319 Voltage Regulator

3. IMPLEMENTATION AND RESULTS

This section involves the design, control, and Android programming of a robotic arm, as well as the modelling process using MATLAB. The design of the robotic arm includes elements such as the selection and assembly of mechanical components and the kinematic analysis of moving parts. In the control part, it involves the integration of motors that move the arm and motion learning. With Android programming, a user interface for the robotic arm can be designed, enabling remote control of movements, and recorded movements can be looped. Additionally, the modelling work done in MATLAB includes simulating the behaviour of the robotic arm and conducting kinematic analysis.

3.1. Designed System

Using 3D modeling software, the robotic arm's design is initially designed. The arm's structural elements, joint locations, and motion mechanism are all included in this model. Then, a 3D printer receives the design. A thermoplastic filament, a kind of meltable plastic substance, is frequently used in 3D printers. The printer works in stages, building up each component of the design piece by piece with tiny layers of melted plastic.

This procedure is frequently carried out via heating, in which a controlled nozzle is used to heat the plastic filament into the required form.

As a consequence, the robotic arm's component pieces are created. Following their assembly, the required electronic parts and actuators are installed, and programming is then completed. As a result, the robotic arm created with a 3D printer is made usable and functioning. Due to benefits like quick prototyping, flexibility for modification, and affordability, this approach offers a lot of potential for the development of robotic systems.

The design parts of robotic arm are given below:

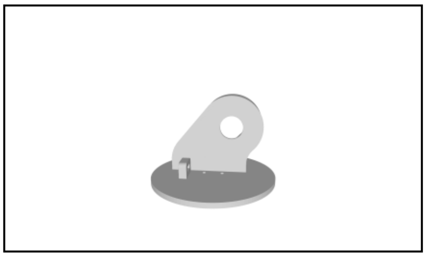


Figure 17. Waist Part

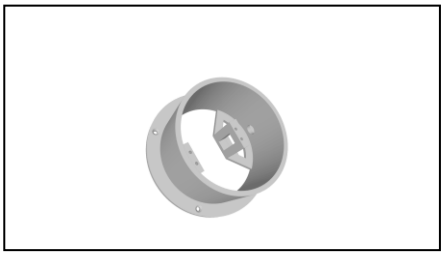


Figure 18. Base Part

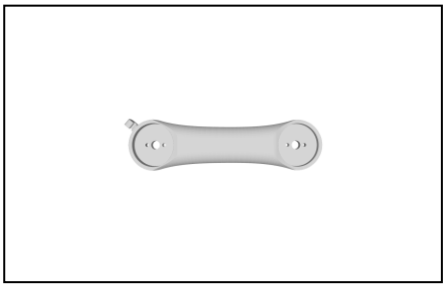


Figure 19. Shoulder Part

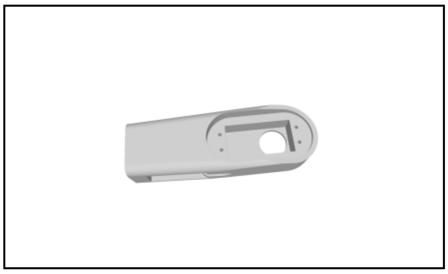


Figure 20. Elbow Part

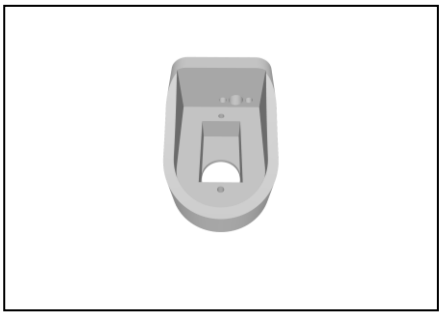


Figure 21. Wrist Roll Part

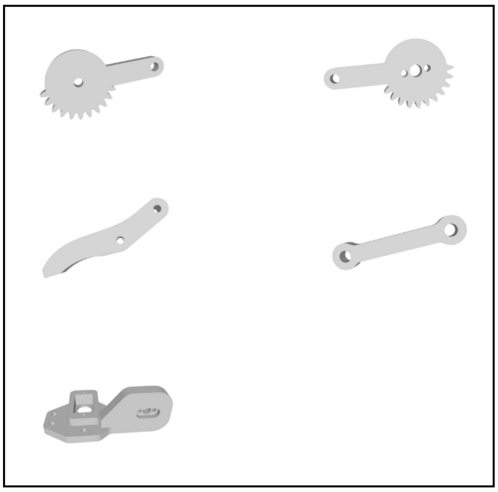


Figure 22. Gripper Part

3.2. Assembled System

The assembly of the robot arm was completed by combining the parts produced with 3D printing with servo motors. The robot arm is a mechanism that can move in 5 axes and this movement is achieved through 5 servo motors.

The drawings of the parts were made using the SolidWorks program and the necessary parts were produced with a 3D printer and then the assembly phase was started. A base was used for the robot arm to remain stationary and rotate left and right, and one servo motor is located on this base. The "waist" section was mounted on top of the base, and a servo motor was also installed in this section to move the robot arm's "shoulder". A servo motor was also placed on this part by combining the Shoulder and Elbow.

A "wrist roll" section was added to the end of the Waist section to allow the gripper to rotate left and right orientationally, and its movement was provided by a servo motor. In the final section, the gripper part was combined and fixed to the wrist roll section. To enable the gripper to open and close, a servo motor was placed on one of the gears of the gripper.

Together with the assembly stages mentioned above, the system was successfully completed and the motion capabilities of the robot arm were provided.

In real life assembly is given below:

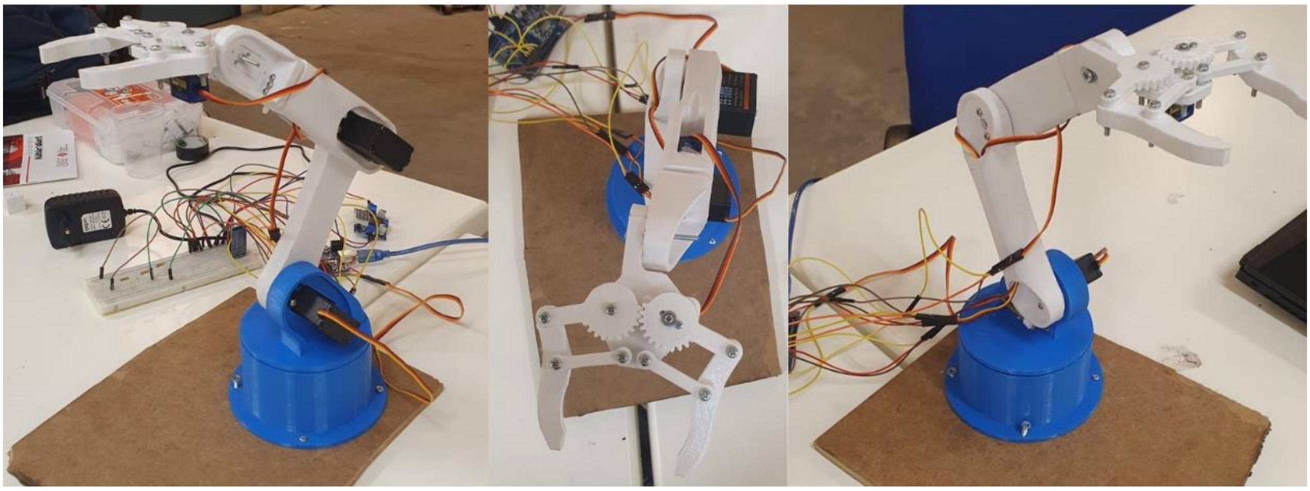


Figure 23. Assembled System

3.3. Circuit

In the robot arm circuit, a 12V 2A adapter has been used. However, since the servos operate at 5V, a voltage regulator has been employed to lower the voltage level to 5V. The GND and Voltage lines have been connected on a breadboard, and the power connections for both Bluetooth and servos have been established. To reduce the Bluetooth RX pin voltage from 5V to 3.3V, three 1K resistors have been used to achieve the necessary drop.

The robot arm circuit is given below:

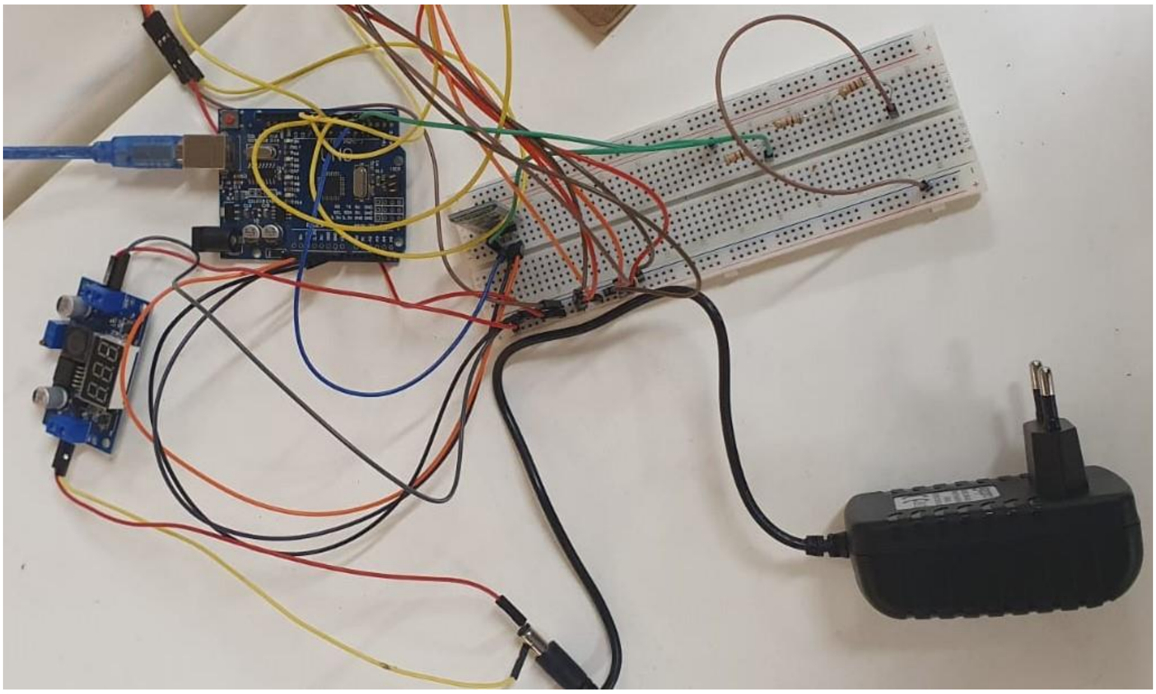


Figure 24. Robot Arm Circuit

3.4. MATLAB Modelling and Simulation

The purpose of MATLAB modelling is to simulate and perform kinematic analysis of 3D components designed before their real-life implementation. By combining these components in a simulation environment, the necessary motion controls can be established and analyzed.

In the Simulink part, the simulation model and working logic of the robot arm are explained using kinematic equations.

Thus, MATLAB modelling allows for a professional approach to simulate and analyze the movements and kinematics of components designed for real-life applications.

Simulink, on the other hand, performs the simulation of robot arms and explains their working principles through kinematic and dynamic equations.

The MATLAB and Simulink parts are given below:

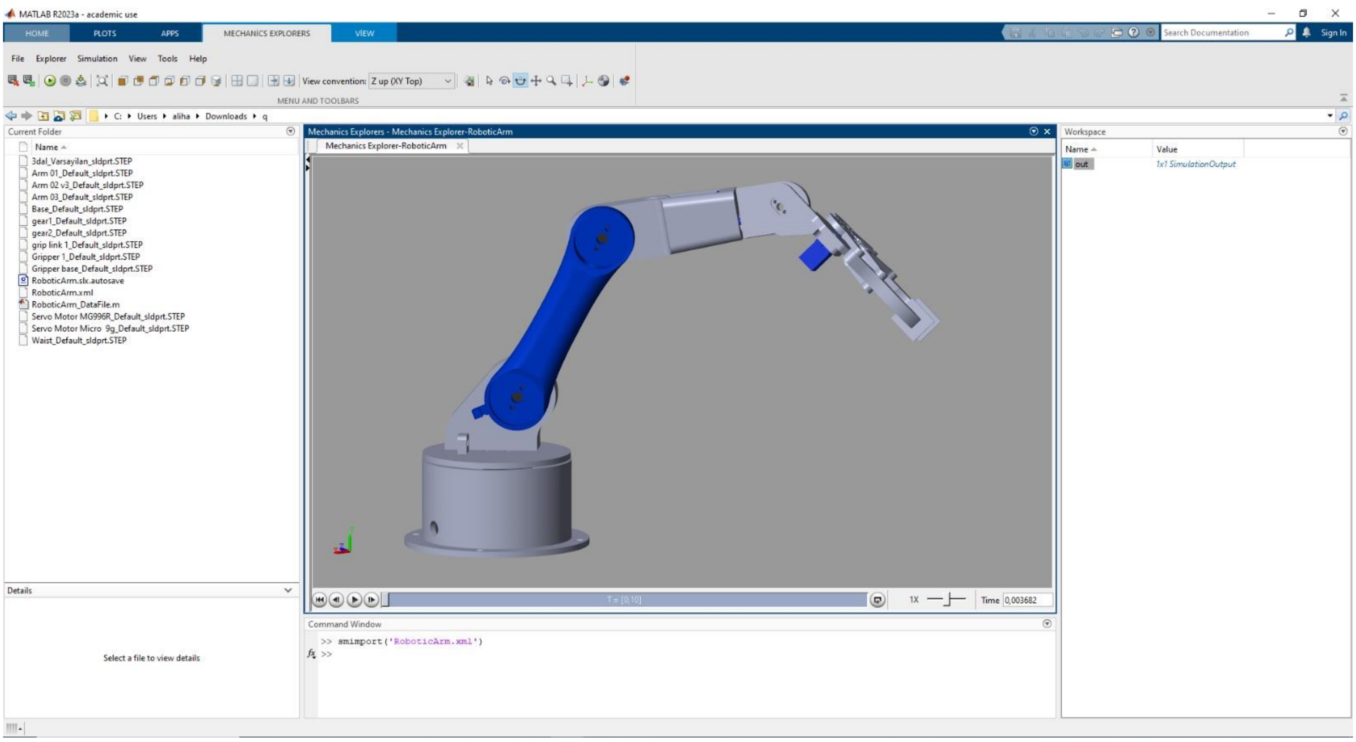


Figure 25. MATLAB Modelling

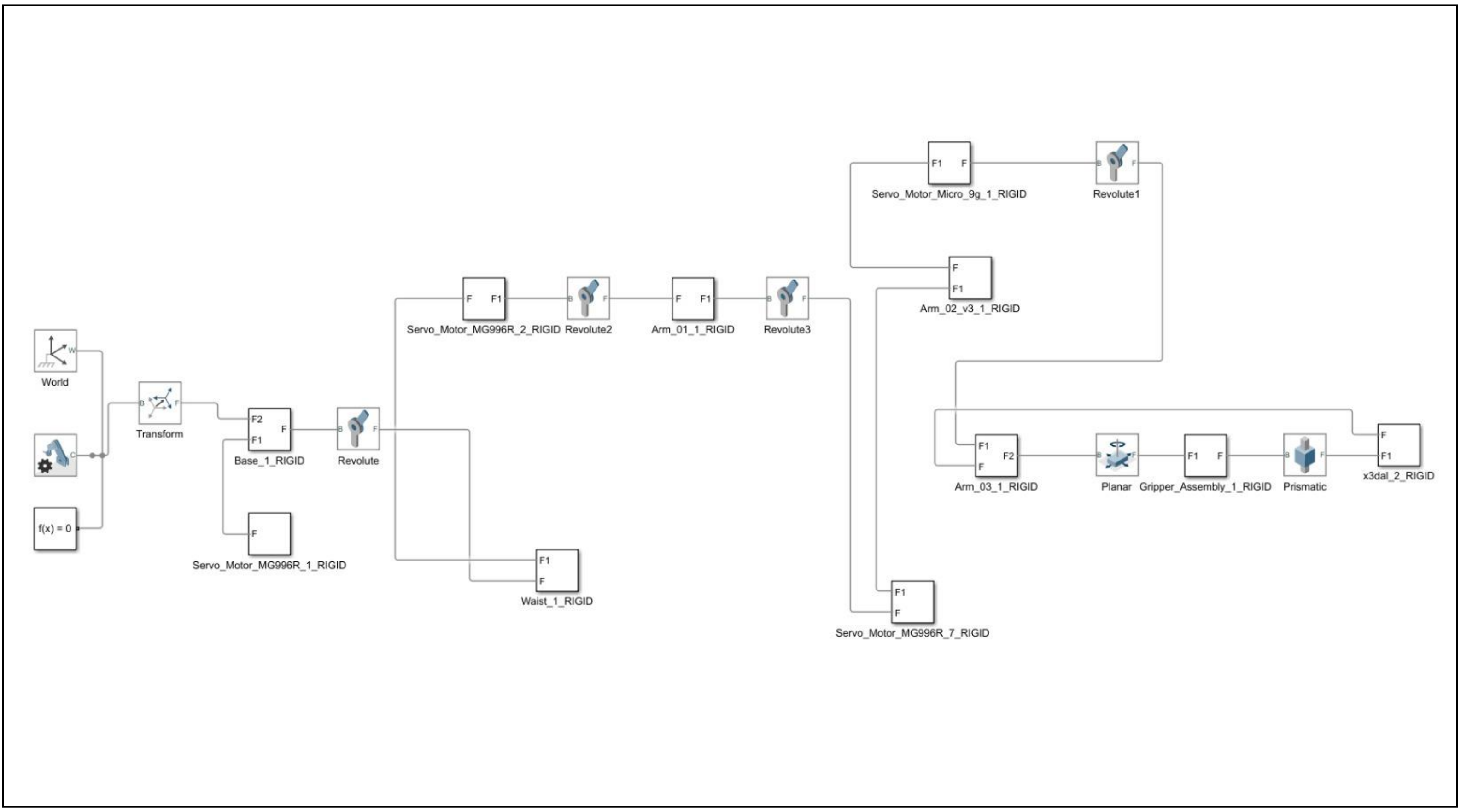


Figure 26. Simulink System Dynamic Model

3.5. Android Programming

During the design process of our project, we incorporated a mobile application that would serve as a means to remotely control our robot. To accomplish this, we utilized the MIT App Inventor software, which provided us with the necessary tools and resources. This powerful platform allowed us to design and customize the user interface of our application according to our specific requirements.

Using the MIT App Inventor, we were able to create a visually appealing and intuitive interface for our mobile application. We designed various screens and elements such as buttons, sliders, and menus, which would enable users to interact with the robot remotely. The software provided us with a range of customization options, allowing us to choose different colors, fonts, and layouts to match our desired aesthetic.

To make our application functional, we integrated blocks of code into the background of the design. These blocks served as the logic and functionality behind the user interface. We programmed them to establish a connection between the mobile device and the robot, enabling users to send commands and receive feedback in real-time. The MIT App Inventor offered a visual programming environment that made it relatively easy for us to create these connections and define the behavior of the application.

Once the design and coding of the application were complete, we were able to build and deploy the application on our mobile devices. This allowed us to control the robot remotely by simply launching the application and utilizing the interface we had designed. The mobile application acted as a bridge between the user and the robot, facilitating seamless and convenient control from a distance.

Overall, the use of the MIT App Inventor greatly facilitated the development of our mobile application for remote control. Its user-friendly interface design tools and visual programming environment allowed us to create a functional and aesthetically pleasing application that met our project requirements.

The android application design that used in the project is given below:

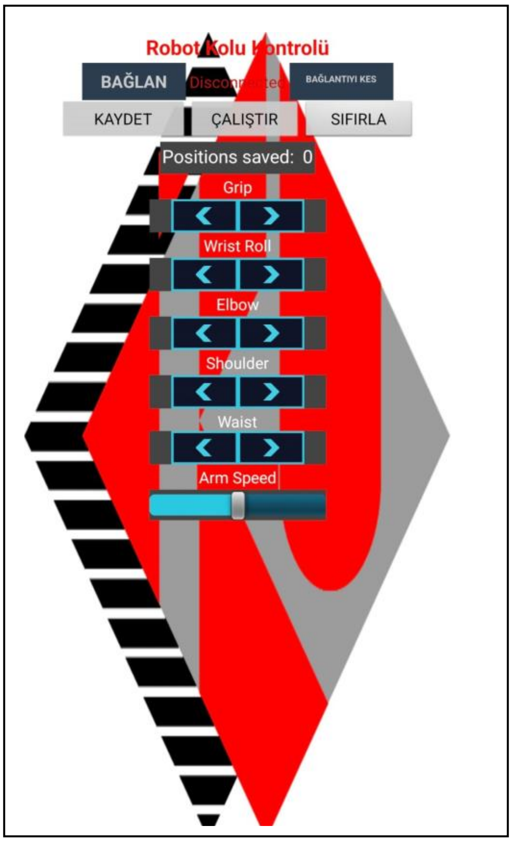


Figure 27. Android Application

Program developers use code blocks coupled on a block-based interface to generate the coding that will execute in the background of our program. With the use of this method, users may more readily create apps since the coding process is made more approachable.

Combining several blocks results in the code that the program is running in the background. These blocks include bits of code that represent certain functions and are coupled to carry out intricate actions. This method makes it easier for users with or without coding skills to comprehend application logic and carry out difficult activities.

Users may then call and run code built on an Arduino platform using the buttons that have been added to the program. When these buttons are pressed, the Arduino code is activated, enabling the execution of pertinent operations. Users may perform desired operations on Arduino-based hardware in this way by interacting with the user interface of their apps.

This strategy caters to a larger user base while speeding up and improving the application development process. It promotes coding and hardware integration as a professional strategy, giving application developers great flexibility and creative options.

Blocks that used in the application on APP Inventor are given below:

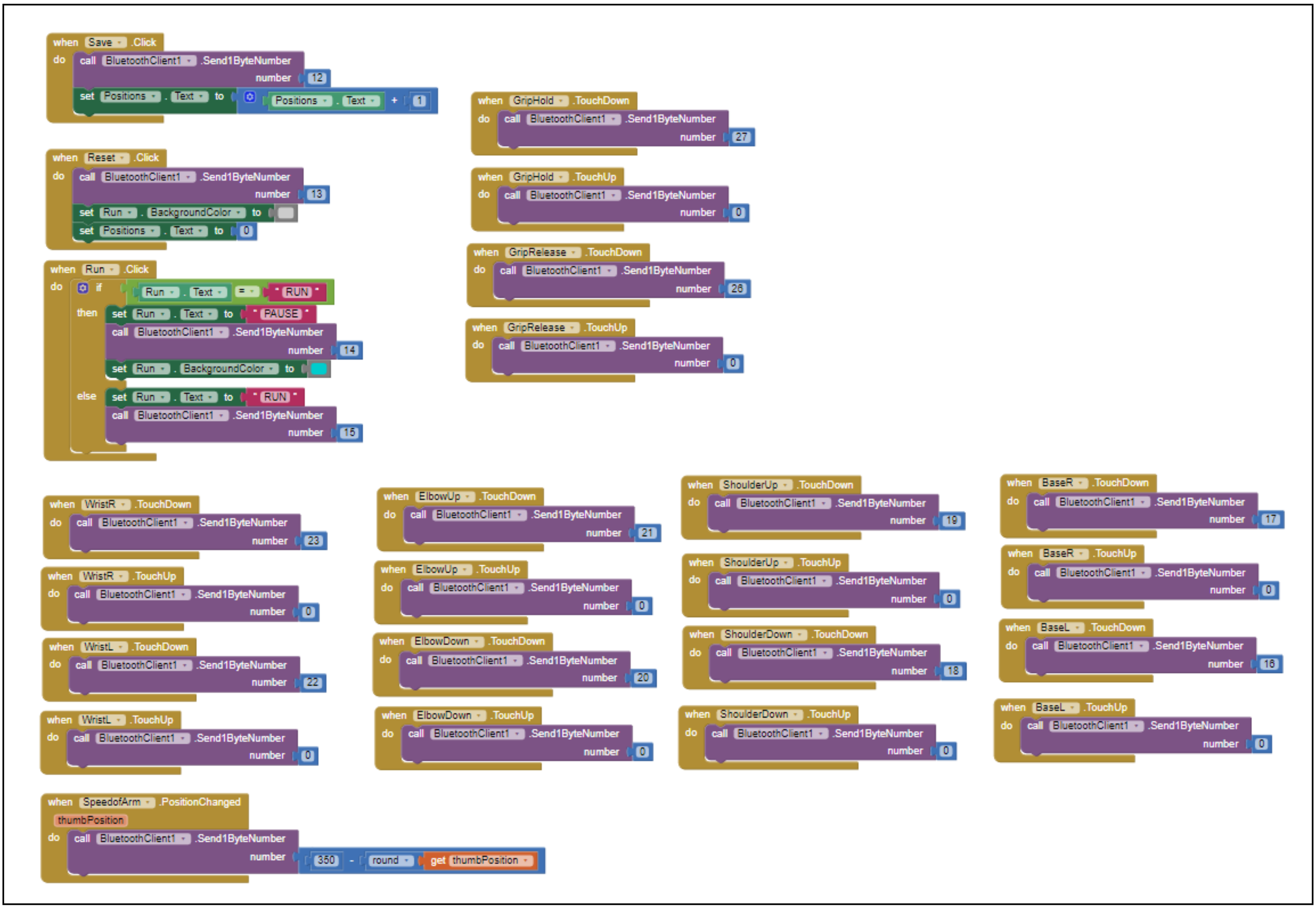


Figure 28. APP Inventor Blocks

There are blocks running in the background of our program. The working principle is as follows: We send numbers as bytes to the Arduino microcontroller through the HC-06 Bluetooth module. For example, when we press the left arrow in the "waist" section of the application interface, we send the corresponding number "22" to the Arduino. By defining these numbers as variables in the Arduino program, we enable the robot to perform the required movement when that number is received. To turn the "waist" section to the left, after sending the "22" number to Arduino, Arduino executes the code we defined for "22" to rotate the robot's body to the left. In this way, by pressing the relevant section on the interface to move the desired part of the robot, we send a number to Arduino, and Arduino executes the corresponding code block based on the received number to move the relevant section.

4. CONCLUSION

The utilization of MATLAB in calculating the forward kinematics and optimizing the design of our robot played a crucial role in the project. By employing MATLAB's powerful tools and algorithms, we were able to determine the Denavit-Hartenberg (D&H) parameters, which describe the kinematic properties of our robot's joints and links. This allowed us to accurately simulate the robot's position and movement in a virtual environment, providing valuable insights into its capabilities and performance.

The flexibility of our robot is a standout feature that sets it apart from other robots available in the market. With the ability to be remotely controlled, it offers users a convenient and intuitive way to interact with the robot. This versatility makes our robot suitable for a wide range of applications across various industries and scenarios. Whether it's performing delicate tasks in a laboratory, assisting in manufacturing processes, or even exploring hazardous environments, our robot's adaptability ensures it can meet the diverse needs of different users.

An important consideration during the design process was cost-effectiveness. By carefully selecting servos based on their affordability, we aimed to create a robot that offers a budget-friendly solution without compromising on performance. This approach not only helps reduce the overall cost of the robot but also makes it more accessible to a wider range of users, including hobbyists, students, and small businesses.

In summary, this project represents a significant step forward in the development of a remotely controllable and cost-effective robot. MATLAB's capabilities in calculating forward kinematics and assisting in design optimization have been instrumental in realizing our goals. The combination of remote control functionality, high flexibility, and affordability makes our robot a compelling choice for various applications, ultimately bridging the gap between users and advanced robotics technology.

REFERENCES

[1] Krishna and Gupta, "Mechanics and Control of Robotics", Mechanical Engineering Series, University of Illinois, Chicago, pp 30-158, 1997.

[2] J. Denavit and R.S. Hartenberg, "A kinematic notation for Lower-pair mechanisms based on matrices", ASME Jappl. Mechan. pp. 215-221, June 1955.

[3] J.J. Craig, "Introduction to Robotics: Mechanics and Control", USA, 1989. [10] H. Asada and J.J.E. Slotine, "Robot Analysis and Control", WileyInterscience Publication, USA, pp 40-56, 1986.

[4] E.A. Maxwell, „General homogeneous coordinates in space of three dimensions", Cambridge. U. K.: Cambridge Unv. Press, pp 102-104, 1900.

[5] M. W. Spong., S. Hutchinson, M. Vidyasagar, "Robot Modelling and Control", Wiley, 2006.

[6] HC-06 Bluetooth Module Technical Specifications Webpage:

https://wiki.mikrokopter.de/en/HC-06

[7] ES 3005 Servo Motor Technical Specifications Webpage:

https://www.robotistan.com/emax-es3005de-su-gecirmez-servo

[8] SG90 Servo Motor Technical Specifications Webpage:

https://www.robotistan.com/tower-pro-sg90-rc-mini-servo-motor

[9]. Bolt, W. (2006) Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering (4. Baskı). New Jersey: Prentice Hall.

[10] Koiva, A.J. (1989) Fundamentals for Control of Robotic Manipulators (1. Baskı). New York: John Wiley & Sons INC.

[11] Forward Kinematics Analysis of a 5-Axis RV-2AJ Robot Manipulator. Journal of Robotics,3(2),45-56.
https://www.researchgate.net/publication/325059301_Forward_Kinematics_Analysisof_a_5-Axis_RV-2AJ_Robot_Manipulator

[12] What is APP Inventor and How to Use:
https://appinventor.mit.edu/
https://www.techlearning.com/how-to/what-is-mit-app-inventor-and-how-does-it-work-tips-and-tricks
https://en.wikipedia.org/wiki/MIT_App_Inventor

[13]HW-319 Voltage Regulator Specifications WebPage:
https://tr.aliexpress.com/i/1005002333521082.html?gatewayAdapt=glo2tur,

[14] GEREĞİNDEN ÇOK SERBESTLİK DERECELİ ROBOT KOLU KONTROL SİSTEMİ TASARIMI VE UYGULAMASI YÜKSEK LİSANS TEZİ
İlker Eren
Pamukkale Üniversitesi 2006

[15]ROBOTIC ARM DESIGN BY UNIVERSAL ROBOTS
18 NOVEMBER 2022
https://www.universal-robots.com/in/blog/robotic-arm-design/

[16]DIY ROBOTIC ARM BY MOHAMED SOLIMAN
2016
https://www.instructables.com/DIY-Robotic-Arm/

[17] The Best DIY & 3D Printed Robot Arms of 2022 by Emmett Grames, Pranav Gharge
11 MARCH 2022
https://all3dp.com/2/3d-printed-robot-arm-diy-robotic/

APPENDIX A

The project codes and MATLAB Simulation files can be found at Github page:

https://github.com/AlihanKanber/robotarmcodeandsimulation.git