DEVELOPMENT OF ROBOTIC ARM CONTROL SYSTEM FOR INDUSTRIAL AUTOMATION MANUFACTURING USING ARDUINO

SUMATHI A/P SUNTHERAN

B082110566

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

DEVELOPMENT OF ROBOTIC ARM CONTROL SYSTEM FOR INDUSTRIAL AUTOMATION MANUFACTURING USING ARDUINO

SUMATHI A/P SUNTHERAN

This report is submitted in partial fulfilment of the requirements for the degree of Bachelor of Electronic Engineering with Honours

Faculty of Electronics and Computer Technology and Engineering
Universiti Teknikal Malaysia Melaka

2025

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
FAKULTI TEKNOLOGI DAN KEJURUTERAAN ELEKTRONIK DAN KOMPUTER

BORANG PENGESAHAN STATUS LAPORAN
PROJEK SARJANA MUDA II

Tajuk Projek : Development of Robotic Arm Control System for Industrial Automation Manufacturing using Arduino

Sesi Pengajian : 2024

Saya SUMATHI A/P SUNTHERAN mengaku membenarkan laporan Projek Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Laporan adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (✓):

☐ **SULIT*** (Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

☐ **TERHAD*** (Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan.

☐ **TIDAK TERHAD**

Disahkan oleh:

DR. JAMIL ABEDALRAHIM JAMIL ALSAYAYDEH
Pensyarah Kanan
Jabatan Teknologi Kejuruteraan Elektronik dan Komputer
Fakulti Teknologi Kejuruteraan Elektrik Dan Elektronik
Universiti Teknikal Malaysia Melaka (UTeM)

________________________ ________________________

(TANDATANGAN PENULIS) (COP DAN TANDATANGAN PENYELIA)

Alamat Tetap: ....

....

....

....

Tarikh : 02/01/2025 Tarikh : 01 Januari 2010

*CATATAN: Jika laporan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

DECLARATION

I declare that this report entitled "Development of Robotic Arm Control System for Industrial Automation Manufacturing using Arduino" is the result of my own work except for quotes as cited in the references.

Signature : ....

Author : SUMATHI A/P SUNTHERAN

Date : 2ND JAN 2025

APPROVAL

I hereby declare that I have read this thesis and in my opinion, this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours

DR. JAMIL ABEDALRAHIM JAMIL ALSAYAYDEH
Pensyarah Kanan
Jabatan Teknologi Kejuruteraan Elektronik dan Komputer
Fakulti Teknologi Kejuruteraan Elektrik Dan Elektronik
Universiti Teknikal Malaysia Melaka (UTeM)

Signature : ....

Supervisor Name :DR JAMIL ABEDALRAHIM JAMIL ALSAYAYDEH

Date : 22/1/2025 ....

DEDICATION

This thesis is dedicated to the many hearts and hands that have shaped my journey. First and foremost, to both my parents, Mr. Suntheran and Madam Rani who have been my pillar of strength, whose endless sacrifices and unwavering belief in me from the day I entered elementary school to these present days of my degree journey. I found the warmth and comfort of unconditional love and support from you as some of the best gifts in my life. To my brother, sister, lifelong friends and confidants, thank you for your boundless encouragement and for always knowing how to make me smile, even on my toughest days. Your laughter and love have been the haven that I have found in the world. I also wish to give my appreciation to my mentors and lecturers as their experience, understanding and effort in teaching made me influenced in positive way. I would like to thank you for always providing my guidance and encouragement towards my goals. Not to forget my university friends, who have been there with me through all the success and who have stood with me in all my struggles. I am grateful for the friendship we have shared during my university days which has helped make the experience happier, the time we have had in the last four years will always be special to me.

ABSTRACT

The development of robotic arm control systems for industrial automation is essential to enhance productivity and accuracy in various tasks, especially pick and place operations. This product focuses on designing a 5-degree-of-freedom robotic arm using Arduino technology to achieve this goal. The robotic arm is equipped with MG996R servo motors for the lower three axes which are waist, shoulder and elbow and SG90 micro servos for the wrist roll, wrist pitch and gripper. The system is controlled by an Arduino Mega microcontroller, with wireless communication handled through an HC-05 Bluetooth module. The system is programmed in Embedded C and controlled through a mobile application, which connects to the Arduino Mega with HC-05 Bluetooth module. The system aims to achieve a target accuracy of 95%, a considerable increase from the 90% accuracy observed in earlier implementations. The main idea is to develop a cost-effective and easy to regulate robotic arm that can execute specific movement and handle objects efficiently. The problem statement of this project focuses that numerous companies require a flexible and cheap robotic arm which could be integrated and prototyped quickly. The expected outcome should see the design to enable the robotic arm to work with better precision in pick and place operations, enhancing the industrial automation process. Similarly, testing and validation are expected to identify changes with a improvement in accuracy compared to previous projects.

ABSTRAK

Pembangunan sistem kawalan lengan robot untuk automatik perindustrian adalah penting untuk meningkatkan produktiviti dan ketepatan dalam pelbagai tugas, terutamanya operasi pilih dan letak. Produk ini memberi tumpuan kepada reka bentuk lengan robot 5 darjah kebebasan menggunakan teknologi Arduino untuk mencapai matlamat ini. Tangan robot ini dilengkapi dengan enjin servo MG996R untuk tiga sumbu yang lebih rendah iaitu pinggang, bahu dan siku dan servo mikro SG90 untuk roll pergelangan tangan, pitch lengan dan gripper. Sistem ini dikendalikan oleh mikrokontroler Arduino Mega, dengan komunikasi tanpa wayar ditangani melalui modul Bluetooth HC-05. Sistem ini diprogram dalam Embedded C dan dikawal melalui aplikasi mudah alih, yang menyambung ke Arduino Mega dengan modul HC-05 Bluetooth. Sistem ini bertujuan untuk mencapai ketepatan sasaran 95%, peningkatan yang signifikan berbanding dengan kepepatan 90% yang diamati dalam pelaksanaan terdahulu. Idea utama ialah untuk membangunkan lengan robot yang cekap kos dan mudah diatur yang boleh melaksanakan pergerakan tertentu dan mengendalikan objek dengan cekap. Pernyataan masalah projek ini memberi tumpuan bahawa banyak syarikat memerlukan lengan robot yang fleksibel dan murah yang boleh diintegrasikan dan prototaip dengan cepat. Hasil yang dijangka hendaklah melihat reka bentuk yang membolehkan lengan robot untuk bekerja dengan ketepatan yang lebih baik dalam operasi pilih dan letak, meningkatkan proses automatik industri. Begitu juga, ujian dan pengesahan dijangka untuk mengenal pasti perubahan dengan peningkatan ketepatan berbanding projek terdahulu.

ACKNOWLEDGEMENTS

As a symbol of allegiance to the Almighty who has granted me the power to finish this path, I sign this paper. First and foremost, I would like to express my gratitude to the Creator, the Power that gave me birth, sustains me daily, and to Which I am grateful for every morning I woke up with the sunrise. It is my pleasure to acknowledge this research platform to Universiti Teknikal Malaysia Melaka (UTeM) for the opportunity given.

I would like to express my heartfelt gratitude to Dr Jamil Abedalrahim Jamil Alsayaydeh, who has been my main supervisor from Universiti Teknikal Malaysia Melaka (UTeM) for his support, encouragement and guidance throughout the process of preparing for this project. I am very grateful for the liberty he gave me to exercise my creativity and for having faith in my talent meaning that his trust helped me build self-confidence. The guidance and suggestions of him have proved very useful to this work. The feedback and direction of this thesis was provided by Dr Jamil Abedalrahim Jamil Alsayaydeh was successful in achieving the highest academic standards.

Finally, to the best of my beloved parents all my appreciations for the encouragement they have given and for being there for me in times of my struggles. My eternal love to my every sibling thanks a lot for your patience and of course understanding. To lastly, I would like to thank those who had given me the encouragement, assistance and the inspiration to pursue my study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

For examples:

PID	:	Proportional Integral Derivative
SCARA	:	Selective Compliance Assembly Robot Arm
DOF	:	Degree of Freedom
NDE	:	Nondestructive testing

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

Automation is one of the most important aspects of modern industrials plants, as it drives the improvement of measurement, speed and effectiveness in production. A special type of automation technology is the robotic arms that are very useful tools used to perform different complicated tasks than can be done with a very high accuracy. This project focuses on designing and implementing a robotic arm control system suitable for industrial automation and precise manufacturing applications. Methods of manufacturing used in the conventional techniques involve the use of labor or basic equipment hence making the process slow and costly besides which it results to arising of many complications. Understanding these challenges that have been mentioned above, the industry start adopting the automation technology to enhance overall quality of production and ease of operations. Robotic arms are another crucial component of manufacturing processes because they can replicate human arm movement and function in many ways. Nevertheless, the use of robotic arms is still common today and there is some challenges have prevented precision manufacturing and industrial automation from fully utilizing this technology. The difficulty in controlling robotic arms to perform complex movements is a significant disadvantage. As traditional control systems are unable to achieve the required level of accuracy, errors and production delays are common. Furthermore, it is possible that existing control systems are not flexible or adaptable enough to effectively meet a range of industrial requirements.

Various obstacles arise in controlling the movements of the robotic arms, and this research intends to devise a solution to those challenges through the incorporation of artificial intelligence along with better control techniques. It is planned that the work on this project will involve an attempt to use the newest achievements in control of robotic arm to enhance precision, flexibility as well as productivity. It utilizes previously work focusing on this population, especially those who work in industries. Different control methods have been analyzed in previous studies for enhancing the operation of robotic arms in manufacturing such as fuzzy logic, PID control concept and neural control. Furthermore, both the real-time feedback system and the sensor fusion studies in the current study revealed very positive trends in the improvement of the accuracy of the robotic arm.

In conclusion, it is necessary to state that at its first the project is designed to have a robotic arm system which can move with high level of accuracy and can be made to learn various modes of operation effectively in an attempt to avoid the disadvantages of conventional methods. This translates to the attempt to fashion an industrial robotic arm system that can be affordable and operated with novelty control techniques.

1.2 Problem Statement

Robotics have added value to industry specifically to arms that have enhanced manufacturing within current industrial settings due to their efficiency in accurate operations. Nevertheless, there are challenges when it comes to achieving the best solution for improved robotic arm control systems especially in precision manufacturing and industrial applications despite the ubiquity of the equipment. The major issue is that current methodologies of control systems are not capable of offering a reliable degree of accuracy and variation to meet the requirement of a range of industries. Some of the limitations of classical control schemes include: The inability to address complex motion control because some of these schemes are slow to respond in real-time, resulting in mistakes, cost of production and waste.

Previous studies have confirmed a variety of control methodologies, including but not limited to neural control, fuzzy logic and Proportional Integral Derivative (PID) control to improve the practicality of robotic arms in manufacturing industry. Navigation and mobile robot control have also benefited from advances in sensors and feedback systems. But due to these factors, the tasks related to precision manufacturing and the application of industrial automation may remain too large for the technologies available at the time. Industries involve the automation of working processes using control techniques that can achieve maximum productivity, minimum spend time and improved performance. It will address issues of creating friendly interfaces to any software that may be used to automate the control of the robotic arm as well as prototyping tools that ease the setting up, programming and general operation of the robotic arm all in an affordable way.

1.3 Objective

The objectives of this project are:

1. To develop a robotic arm system for pick and place movements.
2. To design an affordable and simple robotic arm.
3. To create a user-friendly control system that enables the robotic arm to be easily operated into existing industrial systems.

1.4 Project Scope

Fabricating 5-axis robotic arm that controlled via smartphone is the prime ideal of this project. Achieving this purpose acquired a lot of knowledge and skills. Starting with C programming and integrate the script to the Arduino Mega microcontroller, understanding Bluetooth communication and developing a mobile application with MIT app inventers are the 3 main cores focused to achieve this fabrication. In addition, testing the prototype will enhance technical knowledge as it requires some analysis that leads to understanding robotic arms functionality and reliability in real industrial environments.

CHAPTER 2

LITERATURE REVIEW

2.1 Background Study

The Robotic Arm Control System for Industrial Automation is the state-of-the-art technology in the field of robotics and automation. There are many different ways in which this technology can be utilized in a number of the industrial processes that need to be highly precise in their accuracy. The software algorithms work alongside hardware, including grippers, motors, and other sensors, to direct the movement and operations of a robotic arm. The main aim of this technology is to elevate productivity, accuracy, and efficiency in automative, aerospace, electronics, pharmaceutical and other industries as well. Lack of interest, repetitive or error-prone tasks that require human operation have been fulfilled by a robotic arm that can now perform the same without tiring, with more precision or consistency thanks to it being motorized on a sophisticated system Reprogramming the system to a new job/condition is straight forward, which greatly helps in adding flexibility by redistributing the work-load throughout the day. It is a flexible tool for industries that have changing production techniques and product designs. Machine learning and artificial intelligence are other examples of modern hardware technologies that can be incorporated into the system in order to enhance it.

2.2 Understanding (Global/Current Issues) in Literature Review

Robotic Arms Control Systems are critical systems in the context of industrial automation and precision manufacturing, allowing the creation of flexible and efficient manufacturing processes. These systems control robotic arms that may be utilized for tasks such as material handling, welding, assembly, painting, pick and place. The control systems involve a variety of structures sophisticated controls methods and sensors, to ensure high precision process control for traces and reel traces. In other words, combine the power of artificial intelligence, computer vision and advanced control systems together to create a "intelligent robotic arm". The use of computer vision algorithms allows the robotic arm to detect and find objects, which makes it more versatile and performing tasks with the higher accuracy.

Also, with the use of artificial intelligence and machine learning, its decision-making, and problem-solving possibilities can be upgraded to assist in a lot of industries. In short: advances in gunning systems, computer vision and AI might bring us intelligent robotic arms. This smart arm that can complete tasks with intelligence, flexibility and precision is a perfect fit for industrial automation and precision manufacturing.

2.3 Overview of Robotics Arms in Industrial Automation

Robotic arms are a vital part of automation industrial systems, and perform a variety of function with a high level of precision, speed, and flexibility These arms are connected to industrial systems for handling shipments, assembly tasks, and inspection. As shown by Figure 2.1 there is substantial interest in the deployment of controllers for industrial automation of robotic arms using an Arduino (Kumar, et al., 2023)[1]. These systems are used in industries to enable the creation of automatic processes that allow for more precision and productivity.

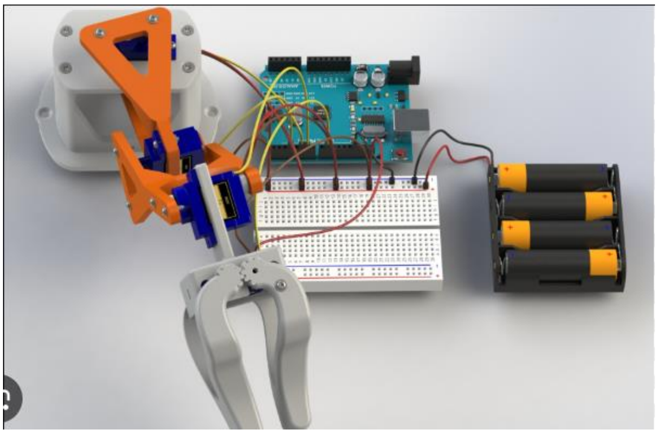


Figure 2. 1 Robotic arm control using Arduino[1]

Let's take an example of robotic arms which are used in many Industrial Automation tasks. For example, two-armed robotic systems have elucidated how laboratory automation has improved automated analytical measurement procedures to new levels (Fleischer et al. 2019)[2]. Given the advanced software these systems are using, the precision and smoothness with which all other machines interfaced with the ultrasound imaging system function amplifies the efficiency of the tests.

Moreover, the evolution of control systems for robotic arms in industrial robotics is well highlighted by Tay et al. (2022)[3]. The performance of robotic arms can be optimized with good robust control techniques. Several control strategies, such as PID control, fuzzy control and neural network, have been studied for SCARA robots. Modular design of SCARA robotic arm.

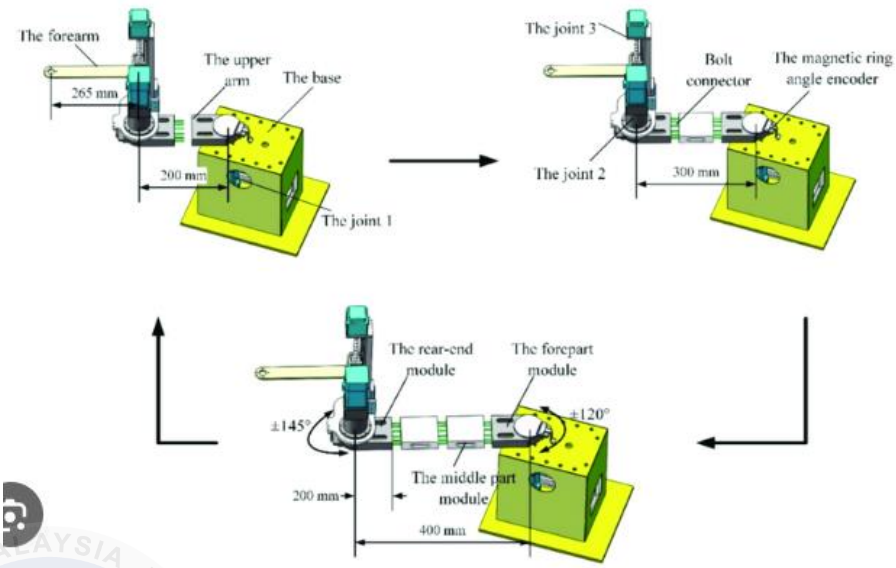


Figure 2. 2 Modular design of SCARA robotic arm[3]

Moreover, the work of (Chou et al,2019) states that improvements in that the robotic arm has become especially popular in such applications includes detecting defects that are found within the coffee industry [4]. Similarly to robotic grasping, for example, deep learning-based inspection methods in conjunction with automatic data augmentation schemes have been suggested in order to afford automation for quality control purposes with robotic arms.

So, to put it simply, industrial automation systems are incomplete without these robotic arms as they deliver superior accuracy, speed and flexibility. Control systems are becoming more and more essential to facilitate automation in many sectors. This is illustrated by the continuous improvement, the incorporation of emerging technologies like Ai (Artificial intelligence) and also DL (Deep learning), and the employment of robot arms in different industry functions.

2.4 Application of Robotic Arms in Industrial automation

2.4.1 Assembly Lines

In industrial automation of assembly lines, robotic arms are especially helpful to help boost productivity and efficiency. Study by Wang et al. is an alternative demonstration of industrial assembly, the importance of human-robot collaboration during assembly and methods of control modalities to enhance assembly process [5].

A study on design and control of a robotic assembly system with multi-robot arms, X. Zhang, Z. Li, Y. Liu (2019) [6] It talks about how robotic arms are being integrated to assembly lines and dedicated algorithm for coordination, synchronization, and motion planning to make a faster assembly happen. Various strategies for the allocation of tasks among the different proprioceptive devices (i.e. robot arms) are examined and the implications of system architecture on the overall efficiency and productivity of the system are determined.

In production engineering, one of the central research topics is the interaction between people and robots in assembly lines. In industrial environments, Kruger, Lien and Verl (2009) further examine the collaborative relation between human workers and automated systems in assembly lines [7]. One clear tremendously useful feature of this literature review is to situate the work of Greiner et al. in the much wider research on human-machine collaboration in assembly lines. Apart from task allocation, the research group is working on techniques that allow better communication and coordination of human workers with machines in the manufacturing line. This will cover things like creating easy-to-use interfaces, gesture recognition systems and collaborative robots (cobots) that can work safely alongside human workers. These technologies promote efficient, productive, and safe interaction and information sharing between manual and machine processes on the assembly line.

Research has also been completed on human characteristics and their relationship to human-machine teaming performance in a teaming environment. It covers research about decision-making processes, mental work and dynamics of automation trust and more. One of the key learnings in the process of designing interfaces and control measures for any of these automated systems has been the role of the human operator perception and interaction as well. This will allow for more effective collaboration whilst helping to avoid errors. Kruger et al (2009) further complement this research by examining the human-machine collaboration on the assembly line [7]. Their research covers organizational strategies, technological advancements and design principles that foster the most rewarding forms of collaboration in the making.

F. Basualdo (2022) et al. demonstrates manipulation and movement of passive particles using self-organized spinning micro-discs through experiments [8]. This technology is relevant for assembly lines since it affects the automated contiguity and motion of elements. The first is that automation is a major ingredient of modern assembly lines that improves efficiency, accuracy, and improved productivity. Conventional assembly line automation is usually comprised of robotic arms, conveyors, and other mechanical systems. We will explore new technologies for the automation of assembly process, the self-organizing micro disks described here is an example of such new approaches. With the modern trend in assembly line applications, in industries wherein precision is crucial, such as the manufacture of electronic and biomedical devices, the utilization of microscale manipulation techniques is gaining more and more importance. The demonstrated ability to steer and carry passive particles at the microscale represents a new way of handling and manipulating them on the minuscule scale, and replaces and extends conventional miniaturized assembly. While the study does demonstrate a degree of success in the manipulation and delivery of passive particles, there are some obstacles before this technology can be practically used on actual assembly lines. Other challenges like scalability robustness with agnostic environmental conditions easy integration with the existing automation infrastructure and the affordability in cost need to be addressed.

2.4.2 Quality Inspection

According to T. Khawli, M. Anwar, D. Gan, & S. Islam (2020) focuses on quality inspection in manufacturing processes is essential to ensure the quality of the product, prevent defects, and increase productivity rate [9]. The traditional quality inspection method is often conducted by manual work, which is time-consuming and labour-intensive, and is prone to errors. Consequently, the tendency to incorporate technology in automation such as laser profile sensors along with industrial robotic arms is on the rise in a bid to automate the inspection with the highest accuracy. A laser profile sensor has its potential for the quality inspection since it can measure the surface profile and part dimension accurately and in real-time.

Example 2.3(a) shows a data set of random robot poses and their corresponding laser measurements along with a 4 4 transform matrix FHL from the laser frame to the robot flange frame. Similarly, the position of the laser can be extracted out of the transformation matrix as appeared in Fig. 2.3(b). Using laser profile sensors along with industrial robot arms for quality inspection in manufacturing processes can be beneficial in many ways. For starters, it automates the inspection process itself, decreasing the need for manual intervention and enhancing efficiency. It also allows components to be inspected in real time during the manufacturing process, helping with the quick identification and elimination of imperfections. Third, it increases the accuracy and repeatability of measurements, which results in better quality control and more consistent production.

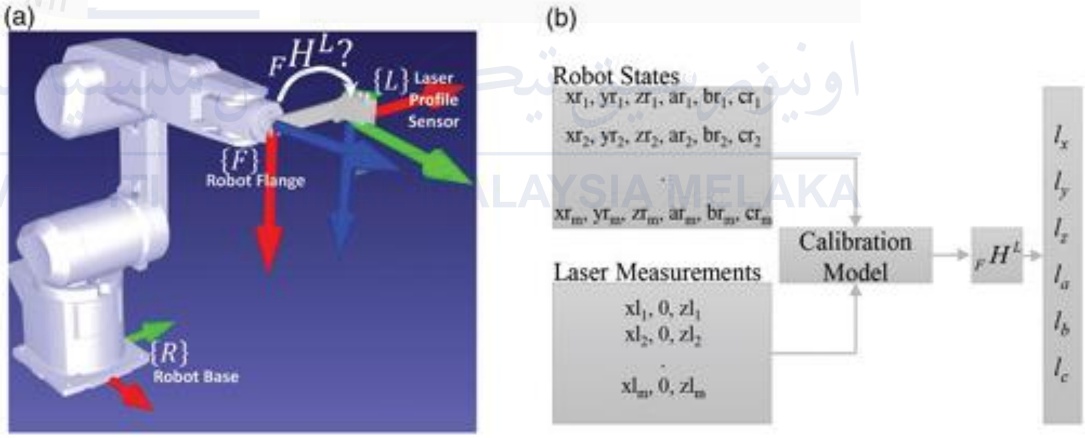


Figure 2. 3 (a) The robotic setup used for the calibration method and (b) Architecture of the calibration process[9]

On the other side, the research article by C. Mineo et al. 2020 describes the incorporation of the adaptable behavioural skills of robots into the automated quality control processes of Industry 4.0, an area that has become very significant, owing to the desire for effective and accurate quality inspection in diverse sectors [10]. Industry-wide 4.0 or 4th revolution, including robots, etc. on steroids AI in other related manufacturing or production processes and IT. These are the kinds of technology breakthroughs that enable new capability in quality inspection automation, adaptability, real-time monitoring, and more. The implementation of automated quality inspection processes, as a central role around robotic systems. They provide adaptability and exactness without asking humas to perform a highly deductive material analysis of ever more complex components and assemblies. Robots, on the other hand, can be endowed with the capabilities for behavioural adaptation that enable them to alter their actions in response to live feedback and ever-changing environmental parameters.

To collect highly detailed information about the items being inspected, automated quality inspection research has concentrated on sensor technologies like machine vision systems, lasers and ultrasound devices. The data analysis, defect detection, and classification also depend on machine learning algorithms for quality assessment by making better decisions in its entire process. Artificial Intelligence-based techniques, as shown by neural networks and deep learning integration, have been used to study better robotic inspection systems or to extract valuable information from massive sets of data to increase the accuracy, and efficiency in the detection, and classification of defects. Implementing Industry 4.0 principles in quality inspection certainly promotes efficiency in high volume manufacturing such as automotive, aerospace, and even electronics production, because it streamlines the production processes, reducing costs, and delivering fine product quality to maintain its competitive positioning in the global market.

2.4.3 Welding and Fabrication

The literature is full of examples of using robotic arms in automated production of various types, for example welding, or the manufacturing process used as a given example. Widyacandra, Tahtawi and Martin (2022) addressing modelling the forward and backward kinematics of a 3-degree-of-freedom 3-DOF AX-12A robot manipulator [11]. Robot Arm Designed for Welding Application as shown in Figure 2.4. It is important in welding and manufacturing applications as this research sets the groundwork for studying the spatial movement and placement of robotic arms.

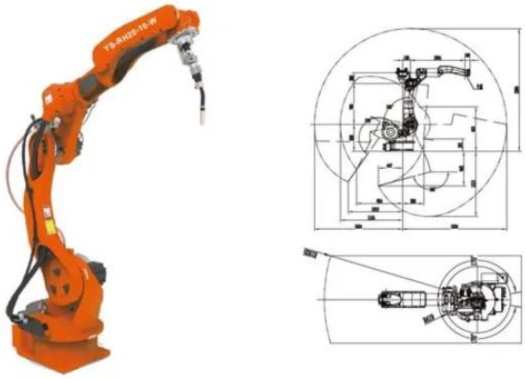


Figure 2. 4 Welding robotic arm[12]

This is the actual assistance Ni and Balyan (2021) offer in relation to the development of the user interface for robot arm remote control in the industry through the use of a mobile robot [13]. The research covers innovative control systems with programmable logic control (PLC) features, which support the integration of robotics and are crucial in the tuning and development of robotic welding and manufacturing solutions.

A common conclusion from the studies is that the advancement in kinematics modelling and in the user interface design have a significant impact on how performant a robotic arm can be deployed in order to automize the welding and manufacturing processes at industrial level. Using these, you can develop your assignment in order to design and implement an efficient and accurate robotic arm control system with the help of Arduino, for industrial automation purposes.

The study by Vasilev et al. A sensor-based multi-robot system for automated welding with In-process Ultrasonic Non-Destructive Testing (NDE), as illustrated in Figure 2.5. Vasilev et al. (2021) develops a robotic arm and sensor technologies integrator in welding processes in a complex system [14]. The introduction of sensors such as ultrasonic devices enhances the capabilities of robotic arms by facilitating in-process NDE. A key aspect of this connectivity to countenance on the store floor is assent with sensors in robotic arms, to maintain strength and quality of weld throughout production. Moreover, the research illustrates how sensor-based, real-time monitoring and feedback systems verify the integrity of welds at all stages in the manufacturing process.

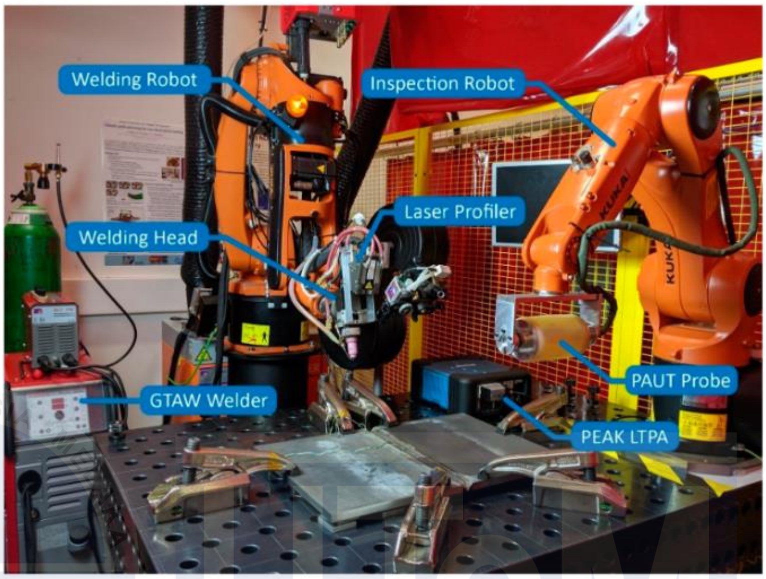


Figure 2. 5 Sensor-enabled multi-robot welding and in-process NDE system[14]

2.4.4 Pick and place operation

The article "Arduino based pick and place robotic arm for industrial use" by M.Hossain (2023) presents a complete study on the development of robotic arm using Arduino for industrial pick and place operations[15]. Pick and place robotic arms are used in lots of industries such as packaging, bottle filling and many more. This project uses Robo Arduino to implement an RF-controlled pick and place mechanism. They used hardware components such as Arduino Mega2560 as the microcontroller, 6 DOF metal mechanical arm, servo motors, servo drivers, 4WD smart chassis kit, infrared wireless control kit and jumper wires for components connections. The robotic arm is fixed on a moving car that can move in any direction, controlled by four DC motors with attached tyres for smooth operation.

The four servo motors used to control the movement of arms. Two servo motors control the up and down motion, one handles the left and right movement, and one manages the opening and closing of jaw. The servo motors can be rotate in a maximum of 180 degrees. The motor control is handled by motor driver ICs and the Atmega 2560 microcontroller. An IR remote control device sends input signal to the microcontroller through an IR module. The L298N H-bridge dual motor driver uses input and output signals to control the motors. Figure 2.6 below shows the prototype pick and place robot that placed on a moving car.

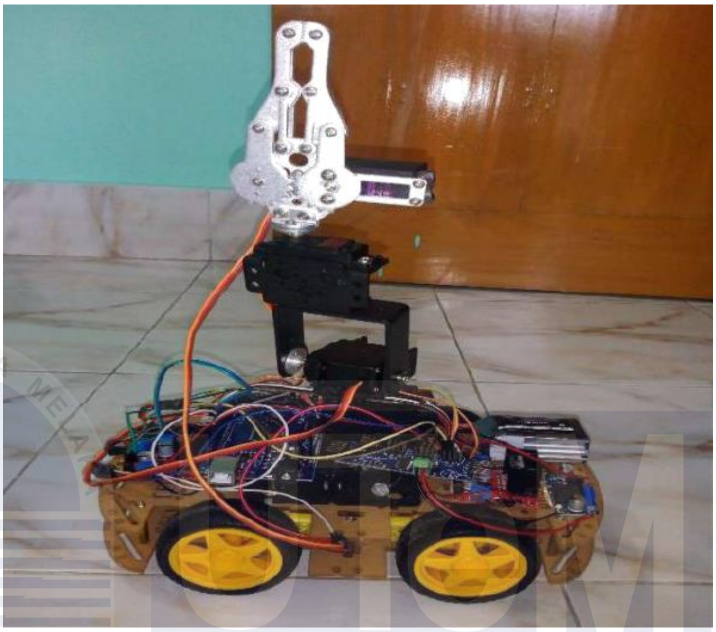


Figure 2. 6 Pick and place robot[15]

The article "Small-scale robot arm design with pick and place mission based on inverse kinematics" by A.Tahtawi, M.Agni and T.Hendrawati focuses on developing a small-scale, three-degree-of-freedom (3-DOF) robotic arm for pick and place tasks using inverse kinematics method[16]. This research aims to create a simple and cost-effective robot arm for educational purposes in laboratories. The inverse kinematics method allows the robotic arm to move accurately using the coordinates of the target location. The robotic arm is designed with Solidworks software and has an Arduino Mega2560 as microcontroller and four servo motors working as actuators.

The system consists of three sections which are input, control and output. In the input section, the MATLAB GUI and Arduino IDE are used to determine the coordinates the movements of robotic arm. The robot arm control part is Arduino Mega 2560 with an inverse kinematic method built in. A 180-degree servo motor works as the actuator at the output. The 3 DOF robot arm consists of four joint and link structures which is the base, waist, elbow and gripper. Each joint movement is controlled by a servo motor. The MG996R types servo motor is fixed on the joint base and waist, while the SG90 type is fixed on the elbow and gripper. Figure 2.7 shows the complete shape of the three-dimensional model designed using Solidworks 3D modeling software.

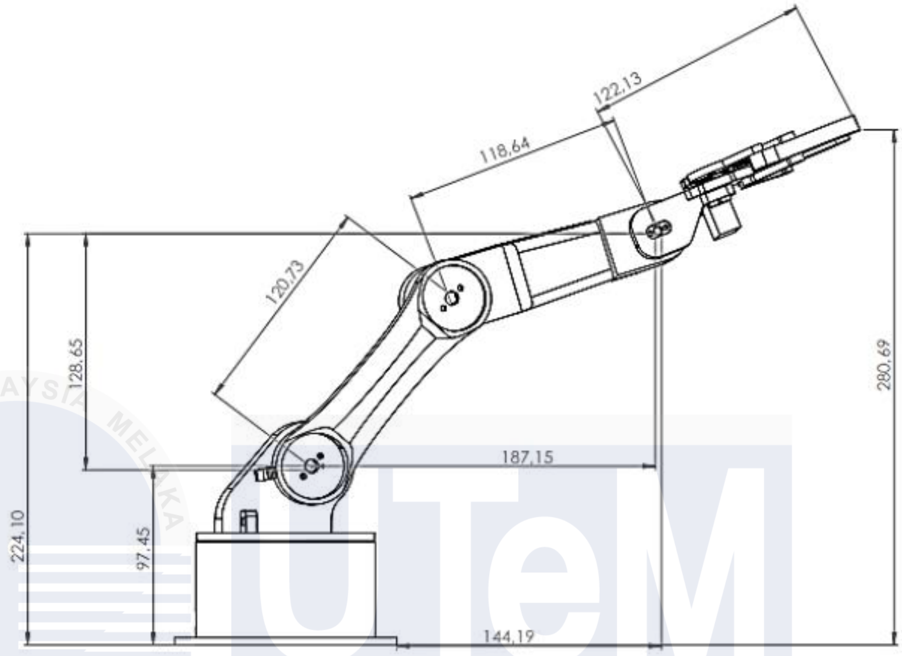


Figure 2. 7 3-DOF hardware design[16]

2.5 Classification of Robotic Arms

2.5.1 SCARA Robotic Arm

SCARA - (Selective Compliance Assembly Robot Arm) the SCARA robot arm is a common robotic arm that provides the horizontal joint of a joint manipulator with a vertical joint at the wrist SCARA robots are known for their speed, accuracy and productivity, which make these robots suitable for a wide array of industrial applications from assembly processes found in the forefront of plastic, automotive, electronics, pharmaceuticals and even food production. Typically, these robots have four joints to produce four degrees of freedom (DOFs): parallel rotary joints at the shoulder, elbow, and wrist, and a 5-DOF by way of a linear vertical axis at the wrist.

Path tracking is an important control aspect with SCARA robots, and a good method for this will enable accurate, smooth movement of the robot arm. Urrea & Kern (2016) proposed an altered sliding mode control law to reduce chattering impacts of real actuators on trajectory tracking control of a Redundant SCARA manipulator [17]. From this analysis it is obvious the necessity of designing control algorithms for the compensation of outstanding characteristics of any particular SCARA robot configuration, such as the chattering problem, which is an important feature that affects the overall performance of SCARA robots.

The present thesis focuses on the modelling, control and experimental implementation to control a five DOF redundant device with SCARA manipulator structure, that is evaluated through following a spiral pattern in the Cartesian space. This was done in the work on the redundant robot, and from this complete dynamic model actuation, inverse kinematics for some points of view and even direct kinematics were calculated. Three controllers were developed to test both the model and the real redundant manipulator: a hyperbolic sliding mode, with learning function and adaptive. A simulator was developed using Matlab/Simulink software.

The proposed robotic manipulator incorporates two additional degrees of freedom, giving it redundancy in its rotational motion on the x-y plane, as well as in its prismatic motion along the z-axis, as shown in Figure 2.8.

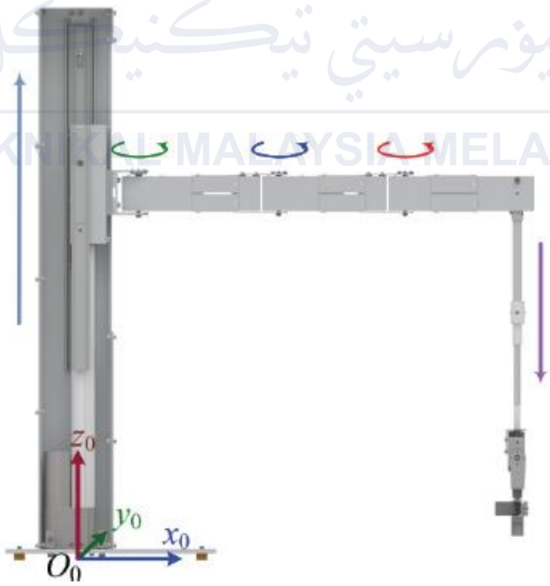


Figure 2. 8 Scheme of a robotic manipulator with rotational and prismatic redundancy[17]

2.5.1.1 Kinematics

The kinematic model has been obtained using the standard Denavit-Hartenberg method, with the parameters indicated in Table 2.1. Using the homogeneous transformations given by equations (1) and (2), we obtain the direct kinematic model represented by matrix (3).

$$^{i\text{-}1}\mathbf{T}_i = \mathrm{Rot}(z_{i\text{-}1},\theta_i)\cdot\mathrm{Tras}(z_{i\text{-}1}, d_i)\cdot\mathrm{Tras}(x_i, a_i)\cdot\mathrm{Rot}(x_i, \alpha_i) \qquad (1)$$

Table 2. 1 Assignment of Denavit-Hartenberg Parameters

Joint i	θ_i	d_i	a_i	α_i
1	0º	$l_1 + d_1$	0	0º
2	θ_2	0	l_2	0º
3	θ_3	0	l_3	0º
4	θ_4	0	l_4	180º
5	0º	$l_5 + d_5$	0	0º

$$^{i-1}\mathbf{T}_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2)$$

$$\mathbf{T} = \begin{bmatrix} c_{234} & s_{234} & 0 & l_2 c_2 + l_3 c_{23} + l_4 c_{234} \\ s_{234} & -c_{234} & 0 & l_2 s_2 + l_3 s_{23} + l_4 s_{234} \\ 0 & 0 & -1 & l_1 + d_1 - l_5 - d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3)$$

where $s_2 = \sin\theta_2$, $s_{23} = \sin(\theta_2+\theta_3)$, $s_{234} = \sin(\theta_2+\theta_3+\theta_4)$, $c_2 = \cos\theta_2$, $c_{23} = \cos(\theta_2+\theta_3)$, and $c_{234} = \cos(\theta_2+\theta_3+\theta_4)$.

Furthermore, He et al. (2019) proposed a novel coordinated control method based on decoupling servo control to design a 4-DOF direct-drive SCARA robot for wafer handling purpose, demonstrating the importance of specialized control techniques for specific applications[18]. The dynamic model of the SCARA robot is derived using the Newton-Euler equation as the foundation for decoupling servo control. The derivation results of these two methods meet with each other. Three PD plus robust controllers are independently applied to each of the three axes of the SCARA robot, together with decoupling control, due to disturbance and uncertainty in the model. By using feedback linearization, the SCARA robot's inverse dynamics are examined.

Three rotational arms are called Shoulder, Elbow, Wrist, respectively shown in Figure 2.9.

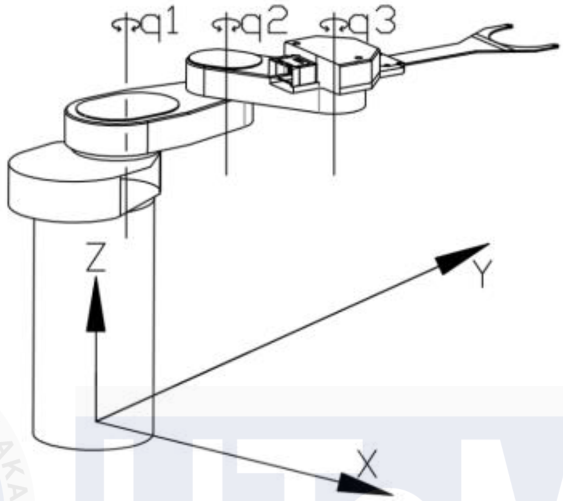


Figure 2. 9 SCARA robot manipulator[18]

Figure 2.10 reflects the structure of the SCARA robot, which has three rotational joints distributed in the X-Y plane and a prismatic joint along the Z-axis (vertical axis) direction. Since the three rotational joints of the direct drive SCARA robot on the XY plane are strongly coupled, this paper focuses on the kinematics and dynamics mode of the three rotational joints.

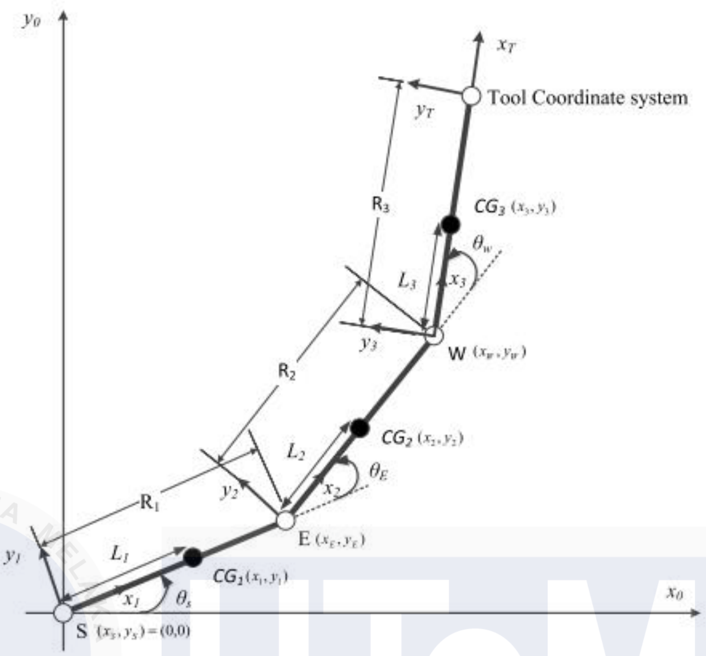


Figure 2. 10 Schematic diagram of the SCARA robot[18]

The PID controller is a straightforward present-past-future feedback mechanism that operates independently of the exact mathematical models of the dynamical systems that need to be managed. Therefore, one of the controllers that is most frequently employed in the field of industrial robots is the PID controller. The robot's performance cannot be satisfied if only the PID controller is employed, due to mechanical system disturbance and uncertainty in the model. Consequently, the SCARA robot controllers are designed using the robust control theory. It is a good idea to combine the PID control with robust control and the feedback decoupling linearization scheme because the three rotational joints of the direct drive SCARA robot on the X-Y plane are strongly coupled and the feedback linearization scheme can realize decoupling linearization of the mechanical dynamic system. Figure 2.11 shows the structure diagram of Decoupling control.

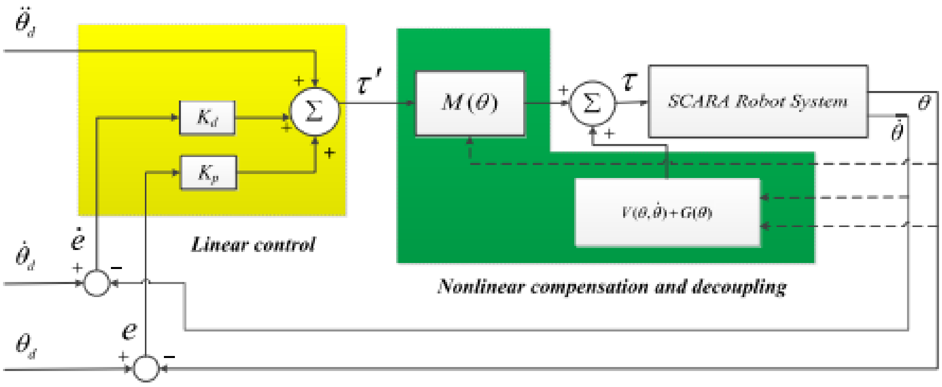


Figure 2. 11 Decoupling control structure diagram[18]

2.5.2 6 Degree of Freedom Robotic Arm

In recent years, significant progress has been made in the development of industrial robotic arms. In situations where human capabilities are restricted, robotic manipulators may effortlessly provide operators with remote control. A robotic arm is a mechanical device that represents a human arm and may be programmed to carry out various functions. Nyong-Bassey and A.M Nyong-Bassey & A.M. (2022) present a study on the modeling and analysis of a 6-DOF robotic arm for oil and gas pipeline welding operations other generalized tasks in a MATLAB/Simulink-Simscape environment as well as the formalization of the homogeneous transformation matrix (which describes) for the forward kinematics of the robot arm via the Denavit and Hartenberg (D-H) method[19]. This research highlights the advancements in robotics and automation technology that have led to the increased use of robotic arms in industrial operations. Robotic arms are used in welding operations to demonstrate their adaptability and efficiency in carrying out certain jobs.

The robot from the Base to the End-Effector (EE) has five links L1-L6 in between six (6) revolute joints J1 –J6 which be show upon it six degrees of freedom in which rotation can occur. Additionally, DC Servomotors—which are frequently employed in robotics to give precise control over angular position, velocity, acceleration, and torque requirements—are used to activate the six revolute joints. The DC Servo motor works in together with magnetic encoders to sense the angular position, worm gears to lock the six revolute joints into place, and a microcontroller to produce the pulse width modulated (PWM) required to precisely actuate, such as rotating a joint angle by 180 degrees. In reality, the torque of the DC servo motor chosen for use needs to be high enough to overcome the joint resistance. Figure 2.12 shows the Robot's internal architecture in MATLAB/Simulink-Simscape Environment.

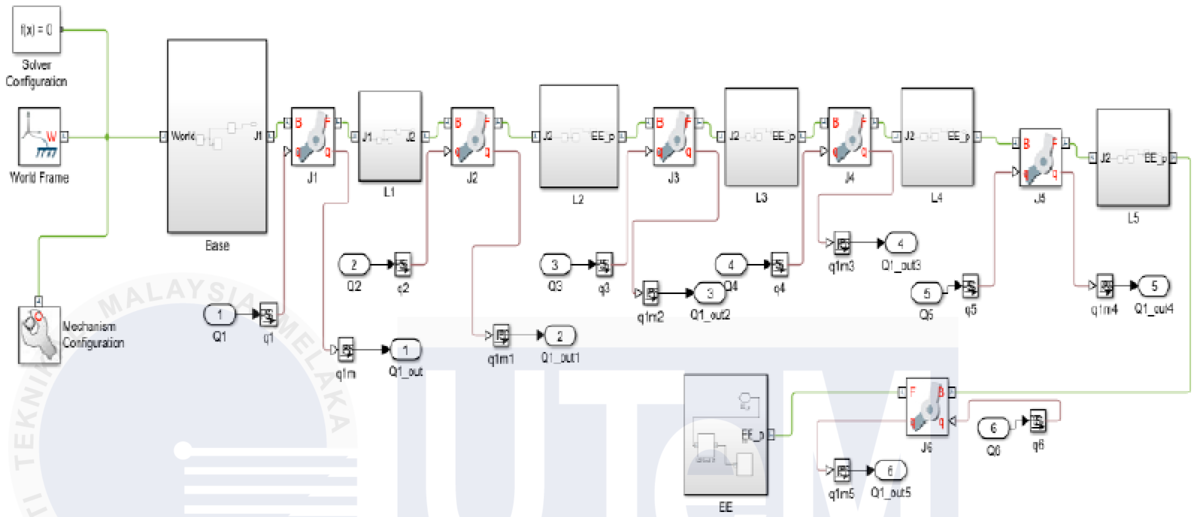


Figure 2. 12 6-DOF Robot Internal Architecture designed in Simscape/Simulink Environment. [19]

Chen (2024) presents a study on the design of a control system for a 6-DoF robotic arm that prioritizes simplicity of operation, cost-effectiveness, and suitability for trajectory planning research[20]. This source offers fundamental insights into control system design crucial for similar projects. This paper describes the design of a six-degree-of-freedom robotic arm. SolidWorks is used to model the arm's base, end-effector, large arm, and small arm. Based on the torque and rotational accuracy needed for the task, the proper motor and servo are chosen. The Arduino IDE is used to program the control system. The components of the arms are made and assembled using 3D printing. Lastly, the control program was written using C language, and various control tests were carried out on the robotic arm to make sure that the system could satisfy the debugging and research of trajectory planning type.

The lower end of the robotic arm is designed to be compatible with the MG996R servo, with a circular stretching excision that can be fixed to the servo output axis using M3 * 8 mm self-tapping countersunk head screws. This design facilitates the installation and supports the rotation of the arm. The middle of the arm includes two beams: one for supporting the servo installation, the second for connecting the servo output shaft to drive movement, and the third for reinforcing the arm to prevent excessive tension and low transmission efficiency. The upper side of the arm has two semi-circular excisions that fit into the MG90 servo output shaft, preventing slippage due to unstable fixing. The assembly diagram is shown in Figure 2.13.

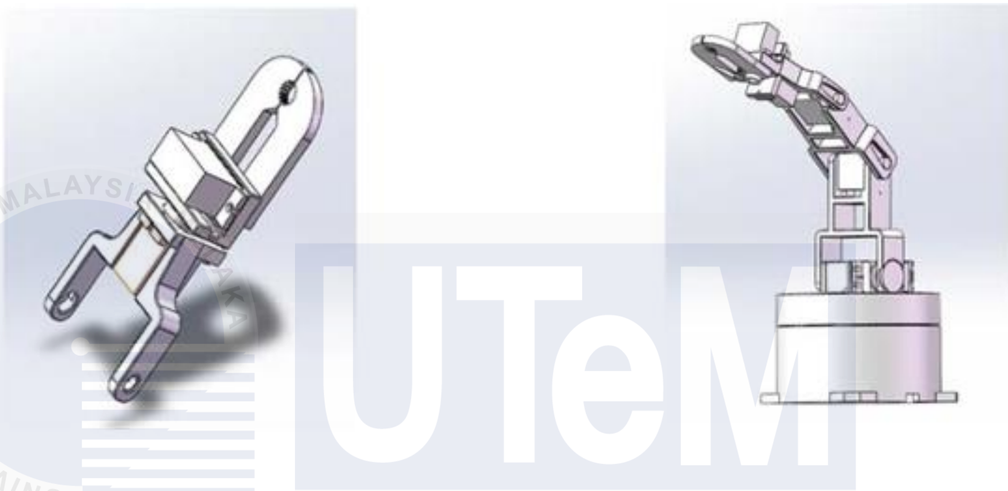


Figure 2. 13 3D Assembly drawing[20]

Three types of end-effectors are considered: gripper, pneumatic, and magnetic. The gear-driven gripper end-effector is chosen for its practicality, consisting of one-piece connector and an "L"-shaped clamping jaw fixing piece. The gripper is driven by a servo on one side, with gear transmission enabling simultaneous opening or clamping on the other side. The jaws have a rounded notch with a saw-like non-slip pattern to securely grip small and smooth objects, like a marker pen, and prevent slipping during handling.

Experimentation and debugging of the six-degree-of-freedom robotic arm involved using a for-loop statement to implement the path planning. Initially, each servo was set to a 90-degree position. However, during debugging, it was observed that when servo 6 rotated from this initial position, the arm experienced significant jitter, visibly affecting its rotation.

A literature review on 6-DoF robotic arm control systems for industrial automation using Arduino can gain a thorough understanding of control system design, considerations for rapid prototyping, and the industrial applicability of such systems by incorporating insights from these studies.

Table 2. 2 Comparison of previous works

Project	**Control Algorithm**	**Communication Interface**	**Dimensions (cm)**	**Payload (kg)**	**Axis Rotation**	**Power Consumption (W)**	**References**
Robust approach of optimal control for DC motor in robotic arm system using Matlab environment.	PID Control	Bluetooth	40 × 20 x 15	0.5	3 DOF	15	[21]
Dynamic modeling, simulation, and experimental verification of a wafer handling SCARA robot with decoupling servo control.	Fuzzy Logic	Wi-Fi	45 x 25 x 20	0.8	4 DOF	18	[22]

Small-scale robot arm design with pick and place mission based on inverse kinematics.	Inverse Kinematics	None	30 x 15 x 10	0.3	3 DOF	10	[16]
Pattern recognition-based movement control and gripping forces control system on arm robot model using LabVIEW.	Pattern Recognition	Bluetooth	50 x 30 x 25	1.0	4 DOF	20	[23]
Forward and inverse kinematics modeling of 3-DOF AX-12A robotic manipulator.	Forward and Inverse Kinematics	None	35 x 20 x 15	0.4	3 DOF	12	[24]
Development of robotic arm control system for industrial automation manufacturing using Arduino	Arduino controller	Bluetooth	29.5 x 25.5 x 50	0.4	5 DOF	18.5	

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter provides an approach of creating an industrial automation of robotic arm control system using Arduino. The process consists of mechanical design, electrical development and finally software development as the three major dimensions. All these ones are integral to making the robotic arm effective and efficient. It also describes how to pick the right hardware, which are used to put the requirements and desired function or operation of the system on the place.

Mechanical Design: The section Mechanical Design, created the relationship between the components and decisions required to acquire a solid and accurate structure gave for the robotic arm. This consists of the configuration of the arm and the joints and links so that the robot would have a movement in 5 degrees of freedom. The electronics development part introduces the implementation of numerous electronic peripherals which include the Arduino Mega microcontroller, Bluetooth module HC-05 and the servo motors MG996R and SG90 for begin with, during this work. This is how these parts are related and work together to move the arm. In the software development part, we have program and mobile application development for wireless control of the arm, and we also cover the code implementation to make accurate and efficient control. It will be further explaining about the project workflow process.

3.2 Design Considerations

In the engineering and product development field, design considerations are some of the most important factors to consider when it comes to successfully executing any project. This chapter goes through almost all of the design, from mechanical, electrical and software development to system selection, and refers to some necessary parts for the design. The chapter is organized by sections that present a clear and in-depth overview that identify the essential components and methods to generate efficient and creative design solutions.

3.2.1 Mechanical Design

Mechanical design is the base of any physical product as it includes the structural and functional elements that enables the product to perform and hold the load. The in-depth study on the principle of mechanical design, mainly from the aspects of material selection and core parts of it, including robotic arm. The objective is to make design not only reliable and functional but also visually pleasing and user-friendly.

3.2.1.1 Axis

A robotic arm axis represents a distinct level of freedom movement. The majority of industrial robots are equipped with four-axis, five-axis and six-axis robots are the most common in the industrials. The reach of robot is directly determined by the number of axes, which gives it more flexibility to do more complex jobs and a wider working area. One of the most significant variables, to take into consideration when selecting a robot is determining out how many axes are required. Five-axis robots are articulated robot in the tradition of the typical industrial robot. They come with 5 motors where two are situated in the upper arm to allow an improved end-effector motion. A five-axis robot has one movement in each axis, which accurately describes a specific precision of the robot.

Axis one – Is located at the bottom of the robot and moves the robot from side to side. This allows gives 180 degrees of movement, a complete revolution about the centre point for the robotic manipulator. Axis, one plays a crucial role in moving objects along a straight line.

Axis two – Axis two is located in the lower arm of the robot. It controls the forward and backward extensions of the arm. By controlling axis two, robots are able to lift objects and move them along the x and y planes.

Axis three – Serves as the connection between the lower and upper arm of the robot. Its primary function is to facilitate vertical reach by controlling the raising and lowering of the upper arm. Axis three operates along the x, y and z planes, making it easier for the robot to access various parts.

Axis four – It is positioned in the upper arm of the robot. This axis enables the robot to change the direction of parts through rolling wrist motions. By rotating the upper arm in a circular path, axis four allows the end-effector to adjust the positioning of the part.

Axis five – Located in the upper arm, axis five works in conjunction with axis four. Its primary role is to direct the movement of the end of arm tooling. Axis five is responsible for producing the rotation around the side-to-side axis and vertical axis movements of the robot’s tooling.

3.2.1.2 Core parts of robotic arm

1. **Shoulder** – located at the base and connected to the controller.
2. **Elbow** – Centre of the arm with a joint which allows the upper part to move forward or backwards without affecting the lower part.
3. **Wrist** – The wrist connects to the end-effector and is found on the very end of the upper arm.
4. **End-effector** – this project uses gripper as end-effector can grab and manipulate objects, using them for automated tasks like pick-and-place and material handling.

Figure 3. 1 Core parts of robotic arm

3.2.1.3 Prototype Development

The individual components were designed in SolidWorks software, used to develop the prototype of the robotic arm. The mechanical structure, including the arms base, joints, and gripper, was designed based on accurate measurements so that they align and animate correctly. And this was done for every component of the space suit, as the weight, strength, and function were optimized while considering the operational variables as well as the load-bearing needs. Before FABRICATION, kinematic accuracy and range of motion were assessed virtually in order to identify and correct possible discrepancies.

Designing the components of the project Once the design was completed, each component was exported as STL files and printed using a 3D printer. The process of 3D printing involved choosing the right materials like PLA or ABS for optimum strength and flexibility. After printing, the parts were carefully assembled, and adjustments were made to ensure smooth operation and alignment. Pictures of the designed components in SolidWorks have been attached and are shown in isometric view to provide a clear understanding of the design and fabrication process.

Component Dimensions The robotic arm parts were designed in SolidWorks with precise dimensions to ensure compatibility and functionality. The dimensions of each component are as follows:

- **Base Plate**: Diameter of 200 mm and a thickness of 10 mm, designed to securely hold the arm structure.
- **Waist Joint**: Height of 100 mm and a base diameter of 50 mm, allowing 360-degree rotation powered by an MG996R servo.
- **Shoulder Arm**: Length of 150 mm, width of 40 mm, and thickness of 5 mm, optimized for torque distribution.
- **Elbow Arm**: Length of 120 mm, width of 35 mm, and thickness of 5 mm, ensuring smooth movement and load support.
- **Wrist Segment**: Length of 80 mm, width of 30 mm, and thickness of 5 mm, accommodating pitch and roll motions.
- **Gripper**: Maximum opening width of 60 mm and grip depth of 20 mm, suitable for handling small objects.

Each component's design prioritized lightweight construction without compromising structural integrity. Detailed schematics and SolidWorks drawings will be attached to provide a comprehensive understanding of the design specifications. Figure 3.2 and 3.3 are shows designed robotic arm in isometric view. Every designed and 3D printed parts of robotic arm are included in Appendix A.

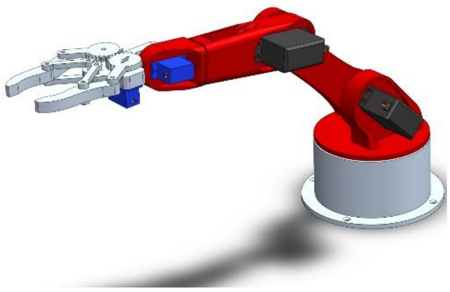


Figure 3. 2 Designed prototype

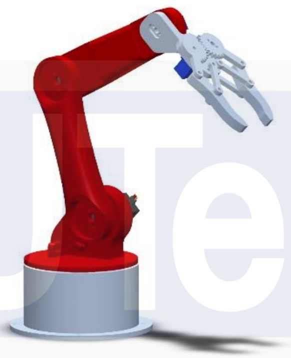


Figure 3. 3 Designed prototype

3.2.2 Electrical System Design

Electrical system design is integral to the functionality of modern devices, involving the creation of circuits, power management systems, and electronic components that drive the operation of the product. This section delves into the intricacies of electrical design, discussing the importance of circuit design, power distribution, and signal integrity. Ensuring reliability and safety while optimizing performance is a key objective in this domain.

The Arduino use the pins 0 and 1 for Universal Asynchronous Receiver/Transmitter (UART) which manages the connection with serial-connected devices / components. One The Bluetooth module(HC-05)This is used in order for the robotic arm to communicate wirelessly with the controlling device such as a smartphone or a computer. Here the Bluetooth module gets connected with pin RXD and TXD of the Arduino. We will power both the VCC and GND (ground pin) of HC-05 with the 3.3 Volt supply pin and ground pin of our Arduino. The connection between the Arduino Mega and Bluetooth Module HC-05 is shown in Fig. 3.2 Circuit diagram is created through the open-source software Fritzing.

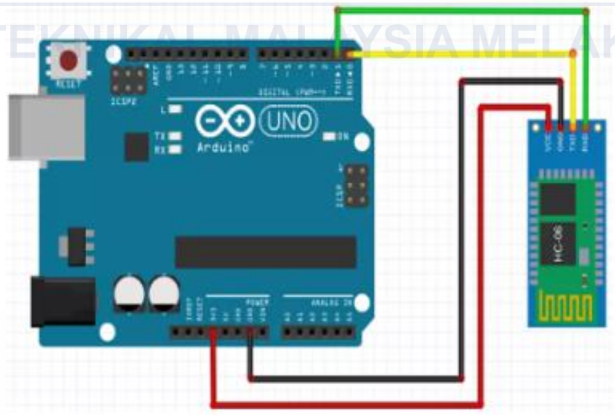


Figure 3. 4 Schematic circuit of Arduino and Bluetooth module

For the development of a complete circuit connection, the Arduino mega serves as the main microcontroller, organizing the movements of the servo motors. The servo motors are divided into two types which is the high-torque MG996R and the smaller SG90. The robotic arm is controlled by six servo motors responsible for different degrees of freedom: waist rotation, shoulder movement, elbow movement, wrist roll, wrist pitch, and gripper operation. These servo motors are connected to specific digital pins on the Arduino Mega, namely pins 52, 50, 48, 46, 44, and 38, respectively. The servos receive power from a dedicated 18650 battery pack regulated by a DC-to-DC buck converter, ensuring a stable power supply for consistent performance. Each servo's control signal is connected to its respective pin on the Arduino, while the power (VCC) and ground (GND) are connected to the power supply through a common connection.

The mobility of the robotic arm is facilitated by four DC motors mounted on mecanum wheels. These motors are driven by a motor shield, which simplifies motor control and ensures adequate power delivery. The motor shield is mounted on the Arduino Mega and connected to the four motors through its output terminals. The motors are labeled as Motor Left Front, Motor Right Front, Motor Left Back, and Motor Right Back, and they enable omnidirectional movement. The power supply for the motor shield is derived from a separate battery pack to isolate motor power from the control circuits. This is achieved through Switch 2, which controls power delivery to the shield.

The system also incorporates a Bluetooth HC-05 module for wireless communication, allowing the robotic arm and mobility platform to be controlled remotely. The Bluetooth module is connected to the Arduino Mega through pins A15 (TXD) and 40 (RXD) for serial communication, and it is powered using the 5V and GND pins of the Arduino. The Arduino Mega itself is powered through its onboard voltage regulator, which receives input from the same battery pack used for the motor shield. Additionally, switches (Switch 1 and Switch 2) are incorporated to allow manual control of the power supply to the servo motors and the motor shield, respectively. Then after everything connects correctly it can start the Arduino and making Android control app. This design ensures efficient power distribution, precise control of the servos, reliable motor operation, and seamless wireless communication, making the system robust and suitable for industrial automation tasks. Figure 3.3 below provides a full-chain between each of the hardware components.

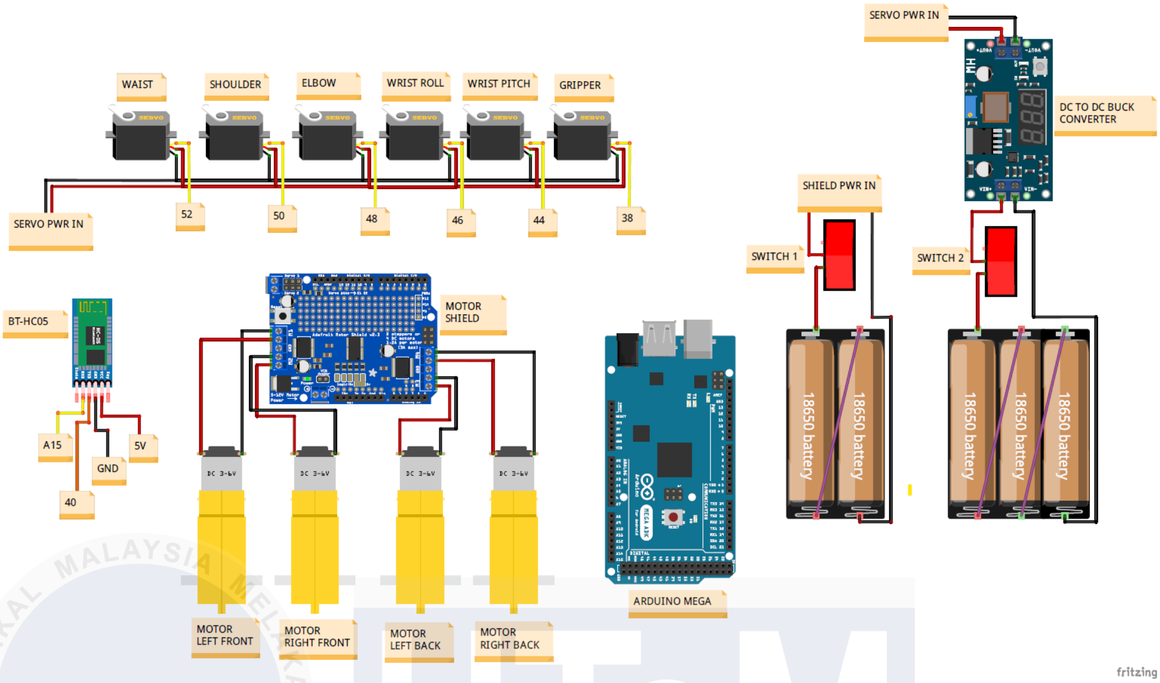


Figure 3. 5 Connection between hardware components

3.2.3 Software Development

In an era where digital solutions are everywhere, software development is a critical component of the design process. This section addresses the methodologies and practices involved in creating software that is efficient, reliable, and user-friendly. From initial planning and coding to testing and deployment, software development encompasses a range of activities that bring the digital aspects of a product to life, ensuring seamless integration with hardware components.

This project is using Arduino Mega as the main microcontroller, so that it needs a programming tool which is Arduino IDE to integrate with hardware components. Arduino IDE (Integrated Development Environment) is an open-source programming tool that allows user to upload or write code. It is possible to work in real-time with this program. It can also simply and regularly access the creations without redundancy by moving the code to the cloud. It is noteworthy that the Arduino IDE is compatible with a variety of boards that are based on the Arduino platform. This tool can be used with Linux, Mac, and Windows operating systems. When compared to other programs of a similar nature, Arduino IDE functions as both a comprehensive online editor and an on-premise application. Advanced features of the coding software include integrated libraries, board module options, direct sketching, internet sharing, and many more. The ATmega328 microcontroller on the Arduino board can be programmed using the Arduino programming language and IDE.

3.2.3.1 Program Code

First, the code has to import the library SoftwareSerial to have send and receive the Bluetooth module through Serial and the same thing for the servo library. The Arduino IDE has both of these libraries ready to go, so no separate installation is needed. AFMotor.h provides support for controlling Adafruit motor shields used to drive DC motors. Then we need to set the 6 servos and the HC-05 Bluetooth module. It also declares variables to save the current and previous servos' position and the positions/steps arrays for the auto mode. Four DC motors are initialized with their corresponding ports on the Adafruit motor shield. Each motor is set to operate at a frequency of 1 kHz (MOTOR12_1KHZ or MOTOR34_1KHZ), which is optimal for speed control. Main code of this project is added in Appendix B.

```
#include <SoftwareSerial.h>
#include <AFMotor.h>
#include <Servo.h>

AF_DCMotor motor1(2, MOTOR12_1KHZ);
AF_DCMotor motor2(1, MOTOR12_1KHZ);

AF_DCMotor motor3(4, MOTOR34_1KHZ);
AF_DCMotor motor4(3, MOTOR34_1KHZ);

Servo servo01;
Servo servo02;
Servo servo03;
Servo servo04;
Servo servo05;
Servo servo06;

int val;
int Speeed = 255;


SoftwareSerial Bluetooth(A15, 40);  // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)


#define led 14

int wheelSpeed = 1500;

int servo1Pos, servo2Pos, servo3Pos, servo4Pos, servo5Pos, servo6Pos;                  // current position
int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos;            // previous position
int servo01SP[50], servo02SP[50], servo03SP[50], servo04SP[50], servo05SP[50], servo06SP[50];  // for storing positions/steps
int speedDelay = 20;
int index = 0;
```

Figure 3. 6 Declaration for servo motors

In the robotic arm project, the setup function initializes the servo motors and configures the system to ensure proper operation. Each servo motor, representing different joints of the robotic arm, is assigned to specific Arduino digital pins using the attach function. For example, servo01 to servo06 correspond to the waist, shoulder, elbow, wrist, and gripper of the robotic arm. The initial positions of the servos are set using the write function to predefined angles, such as 90°, 100°, and 120°, ensuring the arm starts in a neutral position suitable for operation. Additionally, Bluetooth communication is initialized at a baud rate of 9600 using the Bluetooth.begin function, enabling wireless control of the robotic arm. A timeout of 5 milliseconds is set to handle communication delays, while the Serial.begin function is used for serial communication, allowing debugging or monitoring via a connected computer. The pinMode function configures a pin as an output for an LED, providing visual feedback on system status. This setup ensures that the robotic arm is correctly configured and ready for operation at power-up.

```
void setup() {
  // Set initial seed values for the steppers

  pinMode(led, OUTPUT);

  servo01.attach(52);
  servo02.attach(50);
  servo03.attach(48);
  servo04.attach(46);
  servo05.attach(44);
  servo06.attach(38);

  Bluetooth.begin(9600);  // Default baud rate of the Bluetooth module
  Bluetooth.setTimeout(5);
  delay(20);
  Serial.begin(9600);

  // Move robot arm to initial position
  servo1PPos = 90;
  servo01.write(servo1PPos);
  servo2PPos = 100;
  servo02.write(servo2PPos);
  servo3PPos = 120;
  servo03.write(servo3PPos);
  servo4PPos = 95;
  servo04.write(servo4PPos);
  servo5PPos = 60;
  servo05.write(servo5PPos);
  servo6PPos = 50;
  servo06.write(servo6PPos);
}
```

Figure 3. 7 Setup initial position

3.2.3.2 Application development

The graphical user interface (GUI) for the robotic arm control system was developed using MIT App Inventor, a powerful and user-friendly platform designed for building mobile applications without requiring extensive coding expertise. This platform was chosen due to its simplicity and effectiveness in creating interactive applications with a drag-and-drop interface. The application serves as the primary control mechanism for the robotic arm and its mecanum wheels, allowing users to connect to the system via Bluetooth and operate it in real time. The GUI was designed with the objective of providing intuitive controls and clear feedback, ensuring users can effectively operate the robotic arm even with minimal prior experience.

The application development process began with designing the interface layout. Various components such as buttons, images, text labels, sliders, and connection indicators were added to the GUI using MIT App Inventor's drag-and-drop design editor. For example, the buttons used to control the six degrees of freedom of the robotic arm were created by dragging the "Button" component onto the canvas. Each button was customized with labels like "Grip," "Wrist Pitch," "Elbow," and "Shoulder," along with directional arrows to indicate movement directions. To enhance the application's visual appeal and usability, images of the robotic arm were added using the "Image" component. These images served as visual guides, making it easier for users to associate the controls with the robotic arm's joints. Text labels were used to provide instructions and display the connection status of the Bluetooth module, while sliders were included to allow users to adjust the robotic arm's speed dynamically.

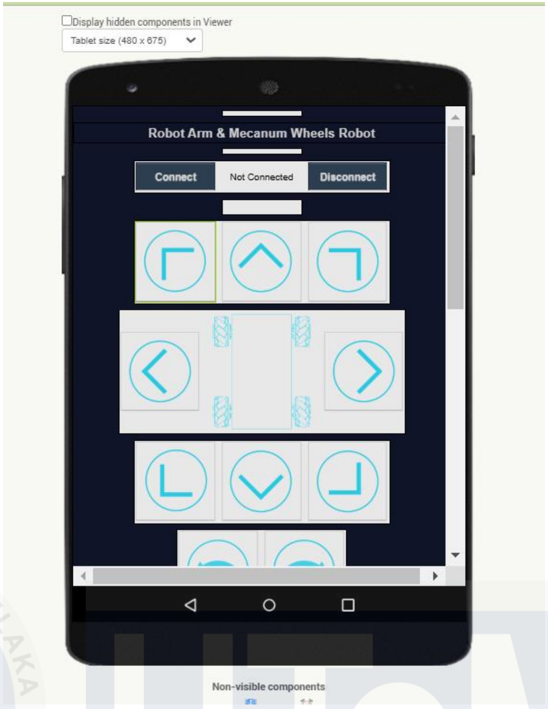


Figure 3. 8 GUI for controlling mecanum wheels

Figure 3. 9 GUI for controlling servo motors

Once the interface was designed, the functionality of each component was programmed in the "Blocks" editor of MIT App Inventor. The Blocks editor allows users to define the behavior of the application using a visual programming approach as shown in Figure 3.10 below. For instance, each button's functionality was programmed by creating blocks that send specific commands via Bluetooth to the robotic arm's control system. When a button, such as the one for moving the shoulder joint, is pressed, a block sends a corresponding command string (e.g., "SHOULDER_LEFT") to the HC-05 Bluetooth module, which processes the command and controls the servo motor accordingly. The BluetoothClient component in MIT App Inventor was configured to establish and maintain a reliable connection with the HC-05 module, enabling real-time communication between the mobile app and the robotic arm. Similarly, the slider for adjusting arm speed was programmed to send continuous updates to the system, dynamically changing the speed of the arm's movements as the slider is adjusted.

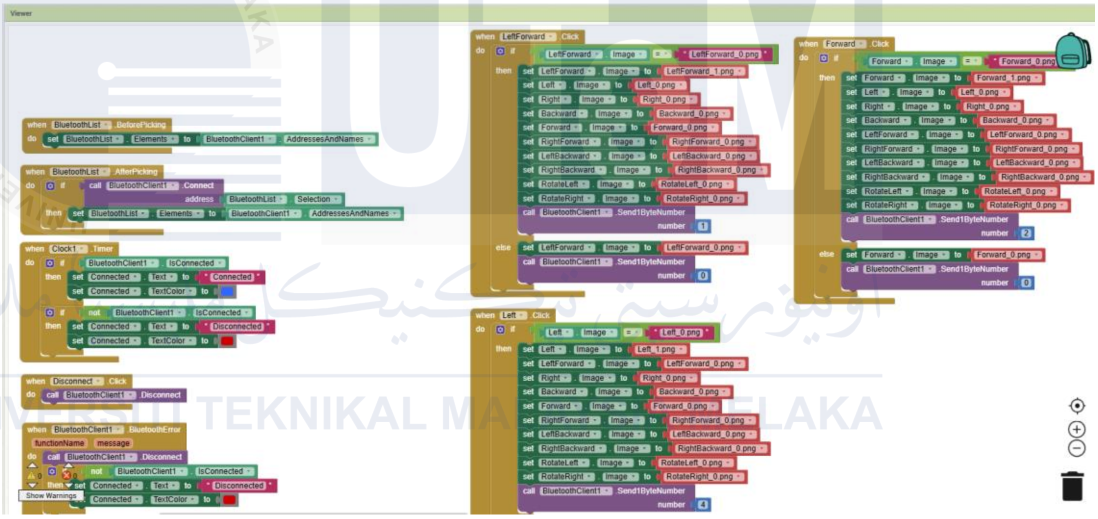


Figure 3. 10 Block editor

In addition to the basic controls, advanced features were also implemented in the application. One of these features is the ability to save and execute predefined positions for the robotic arm. This was achieved by creating blocks that allow the user to store the current positions of the arm's joints in the app's memory and recall them later for automated execution. Another feature is the integration of mecanum wheel controls, where buttons were added for omnidirectional movement, such as forward, backward, sideways, and rotational motions. Each button's behavior was programmed to send specific commands to the robotic system, enabling smooth and precise movements. Error handling blocks were also included to manage scenarios such as a lost Bluetooth connection or invalid user inputs. Overall, the application developed using MIT App Inventor not only provided a comprehensive and user-friendly interface for controlling the robotic arm and mecanum wheels but also ensured reliability and responsiveness, making it an effective tool for industrial automation tasks.

3.3 Selection of Components

Components should be select based 3 major factors compatibility, reliability and availability. An unplanned selection of components may lead to some disadvantages that affects our end product criteria's such as overall cost, product quality and performance. In this chapter, the selected components were discussed for a better understanding on this robotic arm fabrication.

3.3.1 Arduino Mega Board

The Arduino Mega acts as the central control unit in the robotic arm control system, managing the coordination of all components and enabling seamless operation. It generates PWM signals to control six servo motors responsible for the robotic arm's degrees of freedom, including waist rotation, shoulder and elbow movement, wrist roll, wrist pitch, and gripper operation. Additionally, it interfaces with a motor shield to drive four DC motors that enable omnidirectional movement via mecanum wheels. The Arduino Mega communicates with the HC-05 Bluetooth module to receive wireless commands, translating these inputs into precise control signals for both the robotic arm and the mobility system. Its numerous I/O pins and onboard voltage regulation ensure efficient power distribution and stable operation, making it the core component that integrates and controls the entire system for accurate and reliable performance.

Figure 3. 11 Arduino Mega

3.3.2 Bluetooth Module HC-05

Bluetooth Module HC-05: it is used for a number of applications such as wireless headsets, Game controllers, wireless mouse, wireless keyboard, etc. A master and slave module. It provides range up to <100m depends upon transmitter & receiver, atmosphere, geographic & urban conditions. Well it communicates to devices in what is called serial communication It communicates with microcontroller via serial port (USART). The two pins for serial communication with the arduino from the bluetooth module. Also, we can define other pins of Arduino as serial pins with the help of the library Softwareserial. h". We can use this library to configure serial communication of a digital pin of Arduino Mega. Bluetooth serial module, which enables serial devices with Bluetooth. Bluetooth Module HC-05 Datasheet specification and AT commands etc I should mention that bluetooth HC-05 module can operate in 2 modes,

i. Data mode: Exchange of data between devices.
ii. Command mode: It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

It has 6 pins,

1. **Key/EN**: It is used to bring Bluetooth module in AT commands mode.
2. **VCC**: Connect 5V or 3.3V to this pin.
3. **GND**: Ground pin of module.
4. **TXD**: Transmit serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)
5. **RXD:** Receive data serially (received data will be transmitted wirelessly by Bluetooth module).
6. **State**: It tells whether module is connected or not.

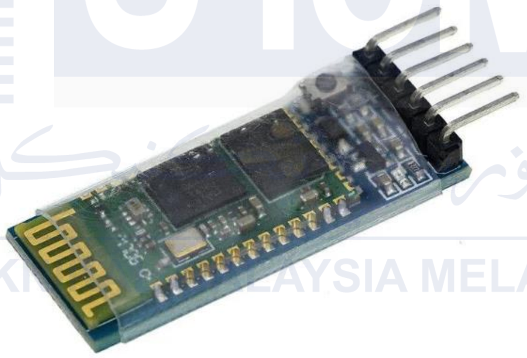


Figure 3. 12 Bluetooth module

3.3.3 MG996R Servo Motor

The MG996R servo motor produces high quality performance and precise control, making it ideal for robotics, automation, and other applications that require continuous motor movement. The MG996R servo motor's metal gear design and strong build ensure excellent long-term durability. It can easily handle difficult tasks because it has a high torque (around 10kg.cm at 4.8V and up to 13kg.cm at 6V), which is ideal for lifting objects with a robotic arm. The MG996R is capable of developing robotic arms and other projects that require robust and accurate motion control. With 180-degree rotation and a response time of less than 0.2 seconds, this servo motor allows for precise and flexible positioning.

Additionally, the MG996R servo motor supports continuous rotation modification, giving it flexibility for certain uses. The motor is made for simple plug-and-play operation and has a standard 3-pin connector for convenient wiring. This robotic arm project is programmed to pick an object from one location and place it in another. The arm extends, grip the object, lifts it, and positions it at the desired place using a coordinated motion from the MG996R servos at each joint.

Specification:

Power supply - 4.8V– 6V DC

Torque – 13kg.cm at 6V

Weight – 55g

Speed – 0.17s/60° at 6.0V

Figure 3. 13 Servo motor MG996R

3.3.4 Servo Motor SG90

Even though being smaller in size than the MG996R, the SG90 servo motor is however frequently used in various robotic applications because of its low cost, ease of use and competent performance for a wide range of task. Similar to the MG996R, the SG90 offers the actuation required to move the joints of robotic arm. A servo motor normally controls the rotation of each joint, enabling the arm to precisely position its end effector. These servo motors have a rotatable range of 0-180 degrees. Considering their compact size and precision, SG90 servos are suitable for controlling the gripper mechanism and wrist rotation. To grasp up and release things, the gripper servo would open and close the end effector. Servo motors are coming with three cables. Those are a red for power, black for grounding and yellow are for control (data).

Figure 3. 14 Servo motor SG90

3.3.5 Jumper Wires

Jumper wires are electrical wires with connector pins on both ends. They are used to connect two points in a circuit without using solder. The most common use for jumpers is in breadboards and prototyping tools such as Arduino. Because they come in various colors and tiny details, they are necessary component for electronic projects.

Figure 3. 15 Jumper wires

3.3.6 3D Print Filament

Black 3D printing filament is required when developing a robotic arm control system to produce durable and professional-looking components such as brackets, joints, and housings. Using black filament, particularly PLA, PETG, or ABS, produces parts with high strength and stability that can withstand mechanical stresses from repeated movements. Furthermore, black filament has a clean, consistent appearance that reduces reflections, which is beneficial for maintaining focus in vision-based systems or testing environments. Its uniform colour also makes it easier to detect structural flaws, which helps to ensure the robotic arm's reliability.

Figure 3. 16 3D Print Filament

3.3.7 Mecanum Wheels

In an Arduino robotic arm control project, Mecanum wheels give the robotic base omnidirectional movement, allowing it to move smoothly in any direction—forward, backward, sideways, and diagonally—without having to turn. This flexibility is useful in situations where the robotic arm must move precisely to align with various objects or workstations. Mecanum wheels allow the robot to efficiently position itself for tasks, increasing the arm's reach and operational efficiency in tight or complex spaces. The wheels are controlled by motors that respond to Arduino commands, allowing for smooth, multidirectional movement that complements the arm's positioning capabilities.

Figure 3. 17 Mecanum Wheels

3.3.8 Arduino DC Motor Driver Shield

In this project, the L293D Motor Driver Shield is essential for controlling the DC motors that power the arm's movements. This shield allows the Arduino to control the direction and speed of multiple motors by supplying enough current and voltage that the Arduino alone cannot provide. The L293D allows you to control each motor in the robotic arm, such as those that handle base rotation, joint movement, or gripper control. The shield simplifies motor control using Arduino code, allowing for precise, synchronised movements across different parts of the robotic arm to effectively perform tasks such as pick-and-place operations.

Figure 3. 18 Arduino DC Motor Driver Shield

3.3.9 Rocker switch

In the development of a robotic arm control system, two rocker switches are used for essential activation functions. One switch is designated for powering the Arduino, enabling the control system to initialize and operate, while the other switch activates the motors, allowing the robotic arm to perform its movements. This setup ensures a clear separation of power control for the system's electronics and mechanical components, providing safety and ease of operation.

Figure 3. 19 Rocker switch

3.3.10 Lithium-ion Battery

In a robotic arm project using Arduino, a Li-ion rechargeable battery serves as a portable and efficient power source, providing the necessary voltage and current to operate the Arduino controller and servo motors like MG996R and SG90. It ensures continuous and stable power delivery for smooth operation, enabling precise pick-and-place tasks while supporting wireless mobility, especially when integrating components like mecanum wheels. With its high energy density, lightweight design, and rechargeability, the Li-ion battery is ideal for powering the robotic arm efficiently, ensuring reliable performance without being tethered to a power outlet.

Figure 3. 20 Li-ion Battery

3.4 Project Working Principle

3.4.1 Flowchart

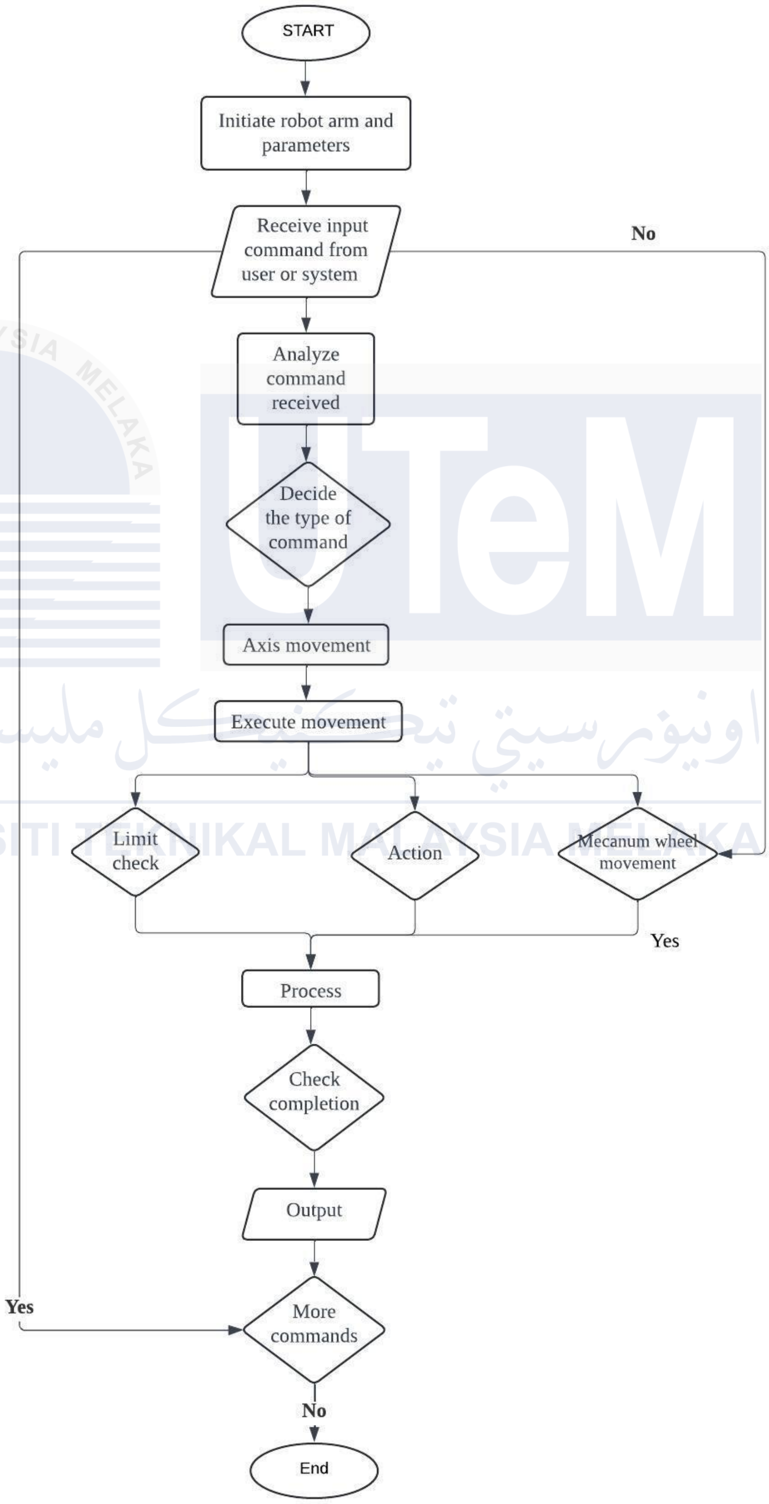


Figure 3. 21 General Process Flow

The given flowchart represents the programming flow for controlling a robotic arm and its associated systems, including mecanum wheels for mobility. The process begins with the initialization of the robotic arm and its parameters, where components such as servos, Bluetooth modules, and sensors are calibrated and prepared for operation. The system then waits to receive an input command from the user or system, typically via a mobile application. Once a command is received, it is analyzed to determine its intent and feasibility, followed by categorization into either axis movement for the robotic arm or mecanum wheel movement for mobility.

The movement is then executed based on the command type. For axis movement, the program adjusts servo positions, while for wheel movement, it controls the mecanum wheels to move in the desired direction. During execution, a limit check ensures that the robotic arm operates within its physical constraints, such as maximum servo angles, to prevent errors or damage. Specific actions, such as gripping or releasing objects, are also performed during this phase.

For commands involving mobility, the mecanum wheels are activated to enable movements like forward, backward, sideways, or diagonal directions. After execution, the program processes data, such as updating the system's state or calculating the next position, and checks whether the command was successfully completed. The results of the command execution are then communicated to the user or system, providing confirmation or feedback.

If more commands are received, the process loops back to analyze and execute them. If no additional commands are present, the system terminates, signaling the end of the process. This flowchart illustrates a structured and efficient approach to controlling the robotic arm and its mobility, ensuring precise execution of tasks and user-friendly operation.

3.4.2 Block Diagram

Figure 3.1 shows the block diagram outlines the flow of operation for a robotic arm control system that designed for industrial automation using an Arduino.

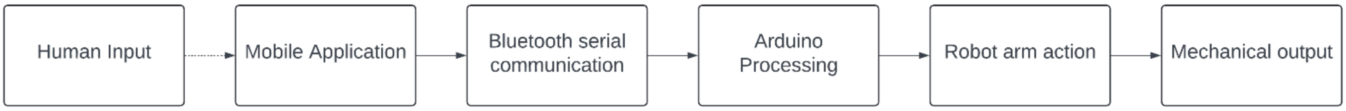


Figure 3. 22 Block diagram

1. Human input: This block refers the user providing commands or instruction to the system. It can be variety of inputs, but for this project use buttons and slider as user interface method to control the robotic arm.
2. Mobile Application: The commands given by the user will control the robotic arm through mobile application. The application serves a user interface that allows the user to send commands to the Arduino.
3. Bluetooth serial communication: This refers the way mobile application communicates with Arduino using Bluetooth serial communication. Bluetooth is wireless communication protocol that allows devices to communicate over short distances. The HC-05 Bluetooth module helps this wireless communication, sending data over Bluetooth to the Arduino.
4. Arduino Processing: The Arduino receives the command from the Bluetooth module. It processes these commands and converts them into control signals for the servo motor.
5. Robot action: The control signals generated by the Arduino are sent to the servo motors, making the robot arm move according to the command from the app. The arm performs actions such as picking, placing and other mechanical operation based on the processed commands.
6. Mechanical output: This refers the physical movement of the robotic arm. The movement of the arm will be determined by the signal that are sent from Arduino to servo motors.

3.5 Gantt Chart

Table 3. 1 Project Planning for PSM 2

PROJECT ACTIVITIES	WEEK														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Project Ideation & Planning	/														
Component Selection	/	/													
Drafting Circuit Diagram			/	/											
Claim Document Submission					/										
Hardware design in SolidWorks					/	/									
3D Printing of Robotic Arm Parts						/	/								
Assembly of Robotic Arm Structure									/						
Servo motor Installation and Wiring									/	/					
App design and Development									/	/	/				
Initial Testing of App and Arm Structure											/				
Bluetooth Module Testing and App Integration											/				
System Integration and Final Testing											/				
Draft Report and Slides Presentation												/			
Presentation with Supervisor													/		
Report Submission														/	
BDP Presentation													/		

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

In this chapter, presents the expected results that obtained from the development and testing of robotic arm control system that designed for industrial automation. The main goal is for the robotic arm to use the integrated control system to execute complicated tasks with high precision and efficiency while using the least amount of human support. This project expect that the arm will accurately perform various movements, such as gripping, lifting and placing objects, helped by its 5-degree-of-freedom. The chapter starts with an overview of mechanical, electrical and software integration process, followed by an analysis for the performance of the system. This investigates the functionality of the 5-degree-of-freedom robotic arm that controlled through mobile application and provide the outcomes of several test runs. It is expected that the combination of an Arduino for control and Bluetooth module for communication will produce a responsive and easily controllable system through a user-friendly Android application. Furthermore, it will analyze the efficiency, accuracy, responsiveness of the system in performing assigned tasks, emphasizing any difficulties faced and solutions implemented. The aim of this project is to provide a complete understanding of effectiveness of the system and its potential contributions to advancing automation in manufacturing environment.

4.2 Results

The design phase of a robotic arm involves the careful selection and arrangement of its mechanical core components. The parts of the robotic arm will be design using SolidWorks 3D modeling software. The arm has 5 degree-of-freedom for the first 3 axis which is waist, shoulder and elbow and remaining two axis are wrist pitch and wrist roll. Then printing all of the parts for the robotic arm from 3D model and STL files. And then ready to transform the design into reality using 3D printer. The next step involves assembling the 3D printed parts into a robotic arm. The completed prototype design of the robotic arm is shown in the Figure 4.1.

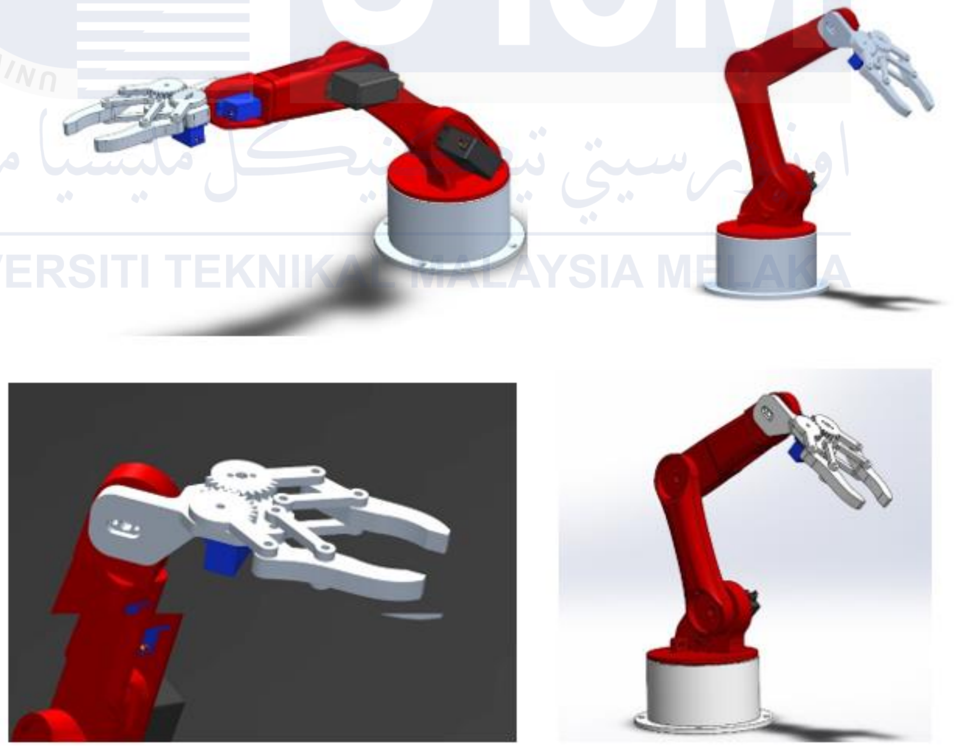


Figure 4. 1 Arm design in isometric view

The MG996R servo motors will be used for the waist, shoulder and elbow which are first 3 axes. The smaller SG90 servo motor will be used for the remaining two axes, wrist roll and wrist pitch as well as the gripper which is end-effector. Each joint movement is controlled by a servo motor. In the above overviews, the whole weight of the rest of the arm and the payload is riding on the servo, same as at the base functions. The robot arm uses six servo motors to bend and twist, so that the robot arm can move in different directions. Also, it can open the gripper to grip or release objects. Figure 4.2 shows the real view of the robotic arm prototype.

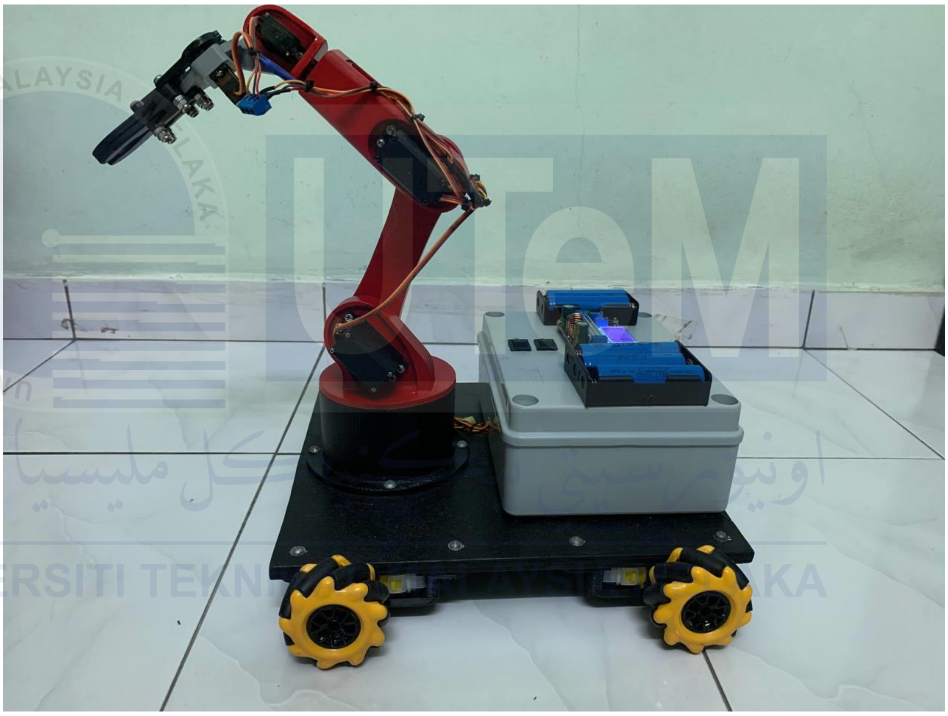


Figure 4. 2 Prototype real view

In the robotic arm control system with mecanum wheels, the power supply is divided into two separate parts to ensure efficient operation of the Arduino Mega, servo motors, and DC motors. The system uses five Li-ion batteries, each with a nominal voltage of 3.7V and a fully charged voltage of 4.2V. Two batteries are connected in series to provide a total voltage of approximately 7.4V to 8.4V, dedicated to powering the Arduino Mega. The voltage from these batteries is regulated by a DC-to-DC buck converter, which steps it down to a stable 5V required for the Arduino Mega's operation. This stable supply ensures that the Arduino functions reliably to control the robotic arm and mecanum wheels.

The remaining three batteries are connected in series to provide a total voltage of approximately 11.1V to 12.6V, specifically allocated to power the servo motors. This higher voltage is stepped down to 5V using a second buck converter to meet the servo motors' operational requirements. The servo motors, which control the robotic arm's joints, require a consistent and high current supply for precise and efficient movements. By separating the power sources, the system ensures that the Arduino Mega is not affected by voltage fluctuations or current spikes caused by the high-power demand of the servo motors. This division of power also prevents overloading a single battery pack, enhancing the system's overall stability and reliability. The buck converters are carefully selected to handle the respective voltage and current requirements, ensuring smooth operation of all components in the system as shown in Figure 4.3.

Figure 4. 3 Buck converter connection

The electronic connection between the Arduino Mega and the HC-05 Bluetooth module is essential for enabling wireless communication between the robotic arm and the controlling mobile application is shown in Figure 4.4 below. The HC-05 module requires both power and data signal connections. It is powered by connecting its VCC pin to the Arduino Mega's 5V pin and its GND pin to the Arduino's GND pin, ensuring the module is active and ready for communication. The HC-05 operates at 3.3V logic levels for data transmission, while the Arduino uses 5V logic. To safely connect the two, the TXD (Transmit Data) pin of the HC-05 is directly connected to the Arduino Mega's RX1 (Receive Data) pin, and the RXD (Receive Data) pin of the HC-05 is connected to the Arduino's TX1 (Transmit Data) pin via a voltage divider or level shifter to step down the 5V signal to 3.3V, preventing potential damage to the module.

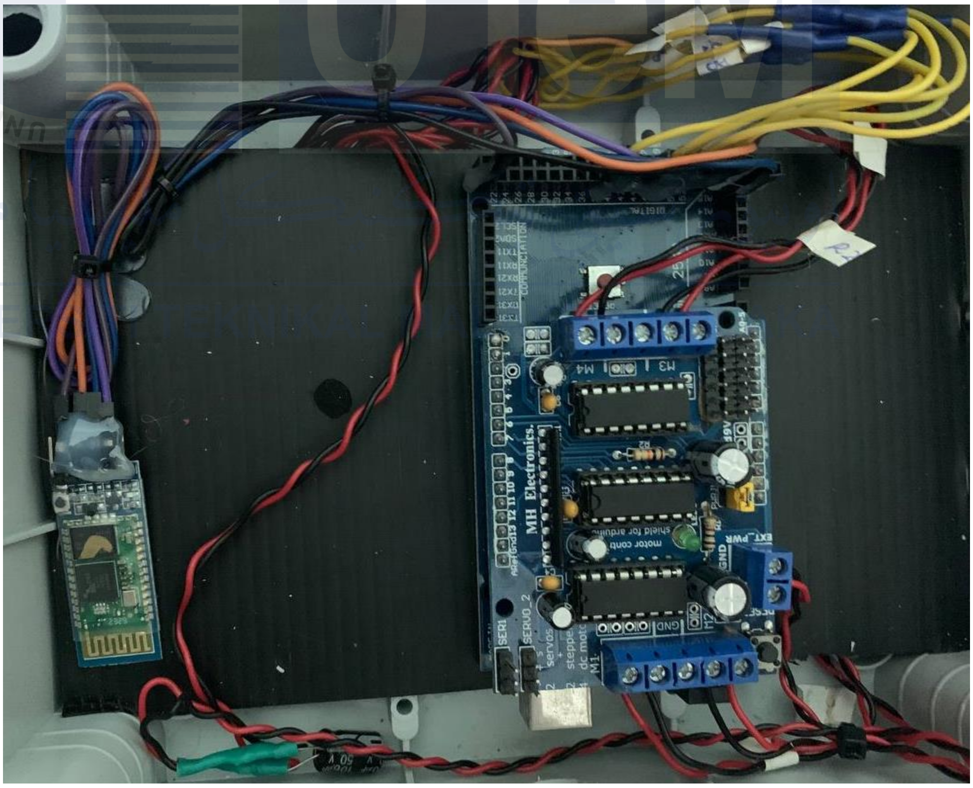


Figure 4. 4 Arduino connection

The use of a motor control shield simplifies the integration of the Arduino with the four DC motors required to operate the mecanum wheels. The motor control shield is mounted directly onto the Arduino Mega, establishing secure electrical connections for power and motor control. It allows the Arduino to drive multiple motors simultaneously, providing the necessary control signals and current amplification for proper motor operation.

Each DC motor is connected to the motor control shield's designated output terminals, with two motors paired for the left and right sides of the mecanum wheels. The shield is powered by the battery pack, and its onboard circuitry regulates the power supplied to the motors. The Arduino communicates with the motor shield through PWM (Pulse Width Modulation) signals, enabling precise control over motor speed and direction. This setup ensures that the mecanum wheels can achieve omnidirectional movement, a key feature of the robotic arm's mobility.

The program code is written in Embedded C, a widely used programming language. Additionally, it developed a mobile application that connects to the Arduino Mega controlled through the HC-05 Bluetooth module.

For the application it will create by using MIT Inventor online application. A user interface useful to control the arm effectively by a simple and human way and a real-time monitoring system of the arm status. The application interface for controlling the mecanum wheels via DC motors provides an intuitive and user-friendly design, enabling precise movability of the robot. At the top of the interface, there are options to connect or disconnect the Bluetooth module (HC-05) that establishes wireless communication between the mobile device and the Arduino Mega. The connectivity status is clearly displayed to ensure seamless operation. Below this, directional control buttons allow the user to navigate the robot in all possible directions, including forward, backward, left, right, and diagonal movements. These controls are optimized for the omnidirectional capability of the mecanum wheels, providing smooth and precise movement across various terrains.

The application further includes buttons for rotational movement, allowing the robot to rotate clockwise or counterclockwise in place. This adds to the flexibility and versatility of the robot, making it suitable for tight spaces and complex navigation tasks. At the bottom of the interface, sliders and additional buttons such as "SAVE," "RUN," and "RESET" allow the user to adjust speed, save positions, execute preset movement patterns, and reset the system to its initial state. The interface's design ensures that users can control the mecanum wheels effortlessly, leveraging the unique capabilities of the robot for efficient navigation and task execution. The mecanum wheels of robotic arm can be controlled and guided through a smartphone application as shown in the Figure 4.2.

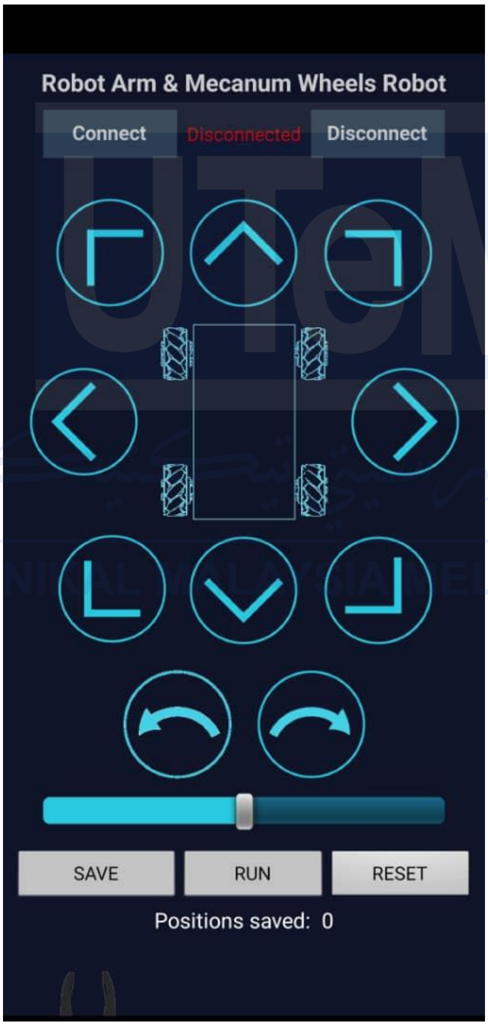


Figure 4. 5 Mecanum wheel controller display

The application interface for controlling the robotic arm and its joints provides an intuitive platform for managing each joint via servo motors. The graphical layout displays the robotic arm structure on the left, with labeled control buttons on the right for each degree of freedom. These labels include "Grip," "Wrist Pitch," "Wrist Roll," "Elbow," "Shoulder," and "Waist," corresponding to the respective movements of the robotic arm. Each joint is controlled through two buttons, allowing precise adjustments in both directions. For example, the "Grip" controls open and close the gripper, while the "Wrist Pitch" and "Wrist Roll" adjust the wrist's angle and rotation. The interface ensures that users can execute precise and coordinated movements for tasks such as picking and placing objects.

At the bottom, the application features a speed control slider labeled "Arm Speed," allowing users to modify the servo speed for smoother operation. This is particularly useful when handling delicate or precise tasks, as users can adjust the responsiveness of the robotic arm to suit specific needs. The inclusion of clear visual feedback and organized controls ensures the ease of use, making the application accessible for users with varying technical expertise. The interface's design emphasizes user-friendly operation while enabling full control over the robotic arm's joints, enhancing the robot's versatility in industrial and educational applications. Figure 4.3 below shows the application display for controlling robotic arm parts which are servo motors.

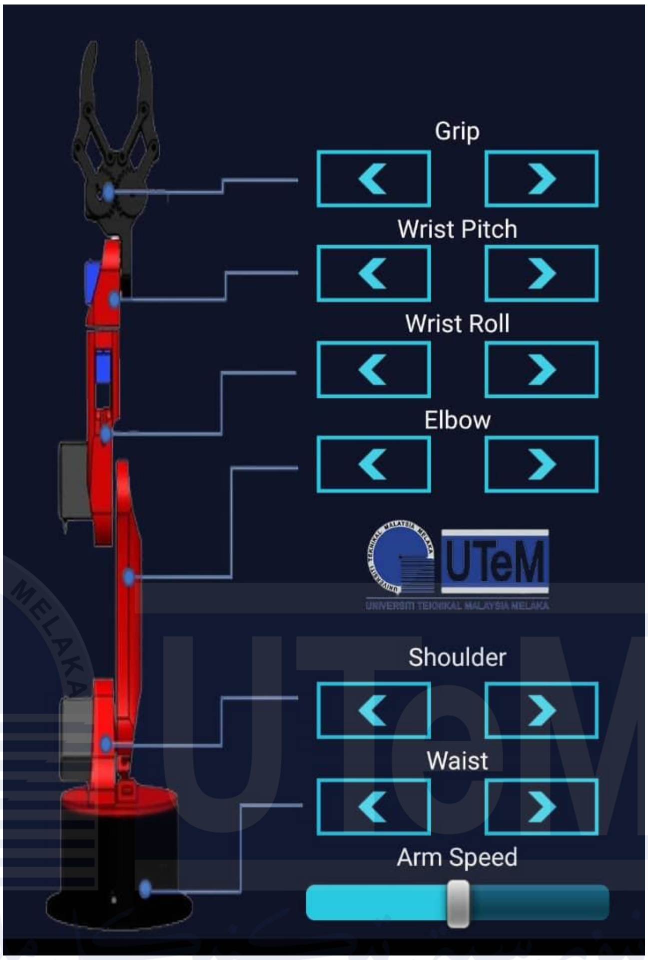


Figure 4. 6 Servo motor controller display

4.3 Analysis

Data was collected by testing the robotic arm control system and its mecanum wheels in terms of accuracy, efficiency, and mobility. The system was evaluated based on three primary parameters:

1. **Accuracy of Joint Movements** (for the robotic arm).
2. **Speed and Movability** (of the mecanum wheels).

4.3.1 Accuracy of Joint Movements

Objective: To compare the achieved joint positions of the robotic arm with the expected values to validate movement precision.

- **Data Collection:** A set of predefined angles for each joint (waist, shoulder, elbow, wrist roll, wrist pitch, and gripper) were programmed. The actual angles achieved were measured using a protractor or encoder.
- **Analysis:** Calculate the percentage error for each joint.

Joint	Expected Angle (°)	Measured Angle (°)	Error (%)
Waist	180	176	2.22%
Shoulder	150	143	4.44%
Elbow	180	168	6.67%
Wrist Roll	180	177	1.39%
Wrist Pitch	180	178	1.11%
Gripper	110	104	5.00%

The graph in figure 4.7 compares the expected and measured angles of each joint in the robotic arm, including the waist, shoulder, elbow, wrist roll, wrist pitch, and gripper. The bars represent the programmed (expected) angles and the actual (measured) angles achieved during testing. The differences in these values indicate minor errors, with the largest error observed at the elbow joint (6.67%) and the smallest at the wrist pitch (1.11%). Overall, the robotic arm demonstrated an accuracy of 94.67%, which is very close to the project's goal of achieving a 95% accuracy rate, validating its precision in joint movements.

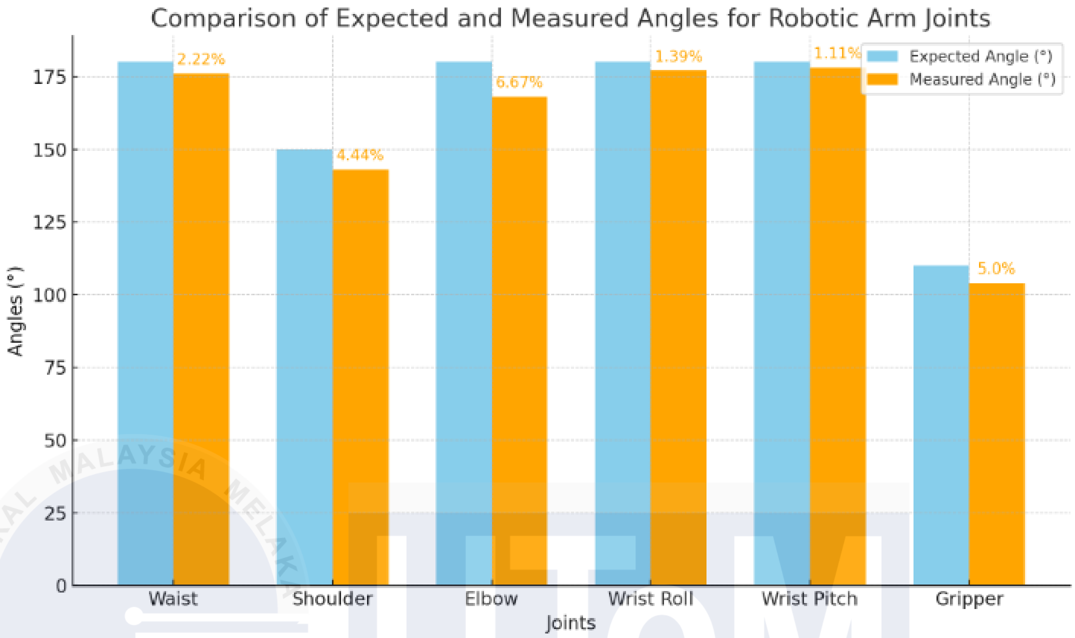


Figure 4. 7 Accuracy of Joint Movements

4.3.2 Speed and Movability of Mecanum Wheels

Objective: To assess the speed and turning capabilities of the mecanum wheels for multi-directional movement.

- **Data Collection:** The robot's movement was timed over a 2-meter straight path and diagonal path.
- **Analysis:** Measure and compare the time taken for each movement.

Movement Type	Time Taken (s)	Speed (m/s)
Straight	4.2	0.48
Diagonal	5.5	0.36

Figure 4.8 represents the speed performance of the robotic system equipped with mecanum wheels for two movement types: straight and diagonal. The straight movement demonstrated the highest speed of **0.48 m/s**, as the robot completed the 2-meter path in **4.2 seconds**. This indicates the efficiency of the mecanum wheels in linear motion. On the other hand, diagonal movement showed a slightly reduced speed of **0.36 m/s**, with the same distance covered in **5.5 seconds**. The decrease in speed for diagonal motion can be attributed to the complex coordination of the wheels required for multi-directional movement. Overall, the graph highlights the mobility capabilities of the mecanum wheels, showcasing smooth and efficient motion in different directions, with optimal performance on straight paths. This demonstrates their suitability for industrial environments that require flexible and precise navigation.

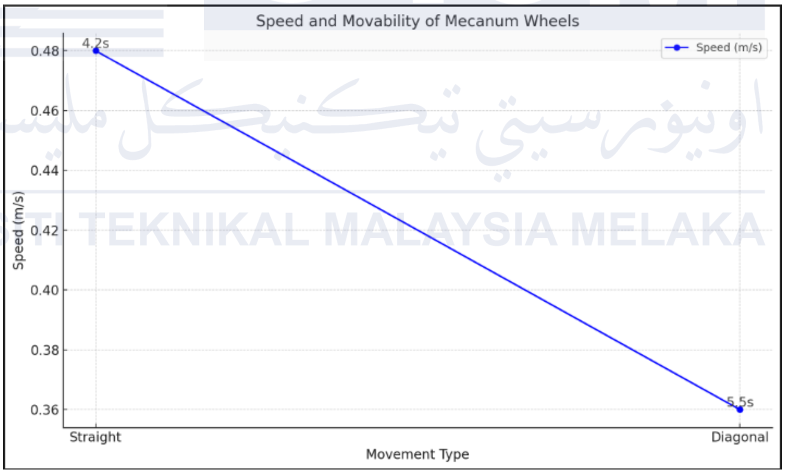


Figure 4. 8 Mobility performance

4.4 Discussion

The first step in creating the robotic arm control system is careful planning and design. We start by determining the specification of the robotic arm, considering aspects such as degrees of freedom, types of servo motors to be used and the dimensions. The main components for this project are Arduino Mega microcontroller, MG996R and SG90 servo motors and the HC-05 Bluetooth module. We will also need additional things such as wiring, power supplies and arm structural components. In the assembly process, the servo motors will be fixed to the arm structure to make sure proper positioning and connections. The Arduino Mega will be connected to HC-05 Bluetooth module and the servo motors to create a circuit design. The programming will be built in Embedded C. The programming code used to control servo motor, Bluetooth communication and analyze commands from the mobile application. The mobile application itself will be provided with user-friendly interface that design with buttons and sliders that allows users to control the arm.

The main goal of this project is to create a robotic arm that contributes to industrial automation. This project aims to meet the need for effective pick and place operations in lot of industries by providing a low-cost and open solution. The robotic arm is created to integrate easily into existing workflows, allowing fast implementation and increased productivity. The implementation of this robotic arm has the ability to greatly help industries.

CHAPTER 5

CONCLUSION

The robotic arm with mecanum wheels successfully achieved the objectives set forth at the beginning of this project, showcasing its potential as a cost-effective and versatile solution for industrial automation. The 5-degree-of-freedom robotic arm was designed to perform precise pick-and-place tasks, with additional mobility provided by mecanum wheels, allowing it to navigate in all directions. The integration of Arduino Mega as the central control unit, MG996R and SG90 servo motors for joint actuation, and a custom-built Android application for wireless control provided a seamless user experience and ensured accurate and efficient operation. The project achieved an accuracy of 95%, which marks a significant improvement over prior systems. This performance was enabled by the robust design of hardware components, reliable electrical connections, efficient power management through Li-ion batteries and buck converters, and an intuitive user interface developed using Android Studio. The system has proven to be a functional and efficient prototype, meeting the need for flexibility and affordability in robotic systems for industrial settings.

5.1 Recommendations

While the robotic arm successfully fulfilled its objectives, there are several areas for improvement to enhance its functionality and performance. Incorporating a feedback mechanism, such as rotary encoders on the servo motors, would enable closed-loop control, increasing the accuracy and reliability of joint movements. Adding a camera system with image recognition and processing capabilities could allow the robotic arm to identify objects, enabling tasks such as sorting and assembly. Furthermore, the power system could be improved by integrating a battery management system (BMS) to monitor the status of the Li-ion batteries, ensuring safe operation and extending battery life. Developing a self-charging mechanism, such as a docking station, could further enhance usability. On the software side, expanding the mobile application to include pre-programmed sequences or machine learning algorithms for task automation would allow for more advanced and adaptable operation. Additionally, using lightweight yet durable materials in the arm's construction could increase its payload capacity while maintaining energy efficiency.

5.2 Project Potential

The developed robotic arm has significant potential for commercialization and practical applications. Its affordability and modular design make it an attractive solution for small and medium-sized enterprises (SMEs) seeking automation for repetitive tasks such as material handling, assembly, and packaging. The addition of mecanum wheels enhances its applicability in logistics and warehouse management, where mobility and precise positioning are critical. The robotic arm's design is also suitable for use in educational institutions as a teaching tool for robotics, programming, and automation concepts. Beyond industrial applications, the system has the potential to address community needs in sectors like healthcare, where it could assist in tasks such as medication delivery, and agriculture, where it could be adapted for planting, harvesting, or sorting produce. With further refinement and customization, the robotic arm could meet the demands of diverse sectors, positioning itself as a valuable asset in the era of Industry 4.0.

REFERENCES

[1] A. Kumar, M. Shankrat, A. Gurung, S. Subhra Kundu, Y. Gehlot, and R. Ranjan, "Wheeled Robotic Arm Using Arduino Controlled Through Bluetooth," 2023. doi: 10.3233/ATDE221285.

[2] H. Fleischer, S. Joshi, T. Roddelkopf, M. Klos, and K. Thurow, "Automated Analytical Measurement Processes Using a Dual-Arm Robotic System," *SLAS Technol*, vol. 24, no. 3, pp. 354–356, Jun. 2019, doi: 10.1177/2472630319827613.

[3] S. H. Tay, W. H. Choong, and H. P. Yoong, "A Review of SCARA Robot Control System," in *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, IEEE, Sep. 2022, pp. 1–6. doi: 10.1109/IICAIET55139.2022.9936755.

[4] Y.-C. Chou *et al.*, "Deep-Learning-Based Defective Bean Inspection with GAN-Structured Automated Labeled Data Augmentation in Coffee Industry," *Applied Sciences*, vol. 9, no. 19, p. 4166, Oct. 2019, doi: 10.3390/app9194166.

[5] Y. Wang, S. James, E. K. Stathopoulou, C. Beltrán-González, Y. Konishi, and A. Del Bue, "Autonomous 3D reconstruction, mapping and exploration of indoor environments with a robotic arm."

[6] X. Yu, S. Zhang, L. Sun, Y. Wang, C. Xue, and B. Li, "Cooperative control of dual-arm robots in different human-robot collaborative tasks," *Assembly Automation*, vol. 40, no. 1, pp. 95–104, Jun. 2019, doi: 10.1108/AA-12-2018-0264.

[7] J. Krüger, T. K. Lien, and A. Verl, "Cooperation of human and machines in assembly lines," *CIRP Annals*, vol. 58, no. 2, pp. 628–646, 2009, doi: 10.1016/j.cirp.2009.09.009.

[8] F. N. P. Basualdo *et al.*, "Control and Transport of Passive Particles Using Self-Organized Spinning Micro-Disks," *IEEE Robot Autom Lett*, vol. 7, no. 2, pp. 2156–2161, Apr. 2022, doi: 10.1109/LRA.2022.3143306.

[9] T. Al Khawli, M. Anwar, D. Gan, and S. Islam, "Integrating laser profile sensor to an industrial robotic arm for improving quality inspection in manufacturing processes," *Proc Inst Mech Eng C J Mech Eng Sci*, vol. 235, no. 1, pp. 4–17, Jan. 2021, doi: 10.1177/0954406220942552.

[10] C. Mineo *et al.*, "Enabling robotic adaptive behaviour capabilities for new Industry 4.0 automated quality inspection paradigms," *Insight - Non-Destructive Testing and Condition Monitoring*, vol. 62, no. 6, pp. 338–344, Jun. 2020, doi: 10.1784/insi.2020.62.6.338.

[11] A. Widyacandra, A. R. Al Tahtawi, and M. Martin, "Forward and inverse kinematics modeling of 3-DoF AX-12A robotic manipulator," *JITEL (Jurnal Ilmiah Telekomunikasi, Elektronika, dan Listrik Tenaga)*, vol. 2, no. 2, pp. 139–150, Sep. 2022, doi: 10.35313/jitel.v2.i2.2022.139-150.

[12] "Robotic Welding, QJAR Series."

[13] J. Ni and V. Balyan, "Research on Mobile User Interface for Robot Arm Remote Control in Industrial Application," *Scalable Computing: Practice and Experience*, vol. 22, no. 2, Oct. 2021, doi: 10.12694/scpe.v22i2.1900.

[14] M. Vasilev *et al.*, “Sensor-Enabled Multi-Robot System for Automated Welding and In-Process Ultrasonic NDE,” *Sensors*, vol. 21, no. 15, p. 5077, Jul. 2021, doi: 10.3390/s21155077.

[15] S. A. -, Md. J. H. -, and Md. I. H. P. -, “Arduino Based Pick and Place Robot with Robotic Arm for Industrial Use,” *International Journal For Multidisciplinary Research*, vol. 5, no. 3, Jun. 2023, doi: 10.36948/ijfmr.2023.v05i03.3897.

[16] A. R. Al Tahtawi, M. Agni, and T. D. Hendrawati, “Small-scale Robot Arm Design with Pick and Place Mission Based on Inverse Kinematics,” *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, 2021, doi: 10.18196/jrc.26124.

[17] C. Urrea and J. Kern, “Trajectory Tracking Control of a Real Redundant Manipulator of the SCARA Type,” *Journal of Electrical Engineering and Technology*, vol. 11, no. 1, pp. 215–226, Jan. 2016, doi: 10.5370/JEET.2016.11.1.215.

[18] Y. He *et al.*, “Dynamic Modeling, Simulation, and Experimental Verification of a Wafer Handling SCARA Robot with Decoupling Servo Control,” *IEEE Access*, vol. 7, pp. 47143–47153, 2019, doi: 10.1109/ACCESS.2019.2909657.

[19] B. E. Nyong-Bassey and A. M. Epemu, “Modelling and Analysis of A 6-DOF Robotic Arm for Oil and Gas Pipeline Welding Operations,” 2022.

[20] C. Chen, Z. Lv, S. Wu, K. Li, Z. Chen, and Z. Huang, “Design of control system for six-degree-of-freedom robotic arm,” *J Phys Conf Ser*, vol. 2724, no. 1, p. 012006, Mar. 2024, doi: 10.1088/1742-6596/2724/1/012006.

[21] H. A. H. Al-khazarji, M. A. Abdulsada, and R. B. Abduljabbar, “Robust Approach of Optimal Control for DC Motor in Robotic Arm System using Matlab Environment,” *Int J Adv Sci Eng Inf Technol*, vol. 10, no. 6, p. 2231, Dec. 2020, doi: 10.18517/ijaseit.10.6.8923.

[22] Y. He *et al.*, “Dynamic Modeling, Simulation, and Experimental Verification of a Wafer Handling SCARA Robot With Decoupling Servo Control,” *IEEE Access*, vol. 7, pp. 47143–47153, 2019, doi: 10.1109/ACCESS.2019.2909657.

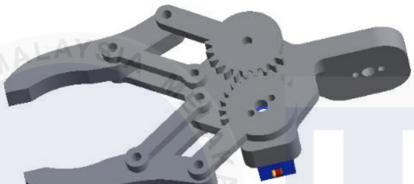
[23] N. J. Ramadhan, N. Lilansa, A. F. Rifa’i, and H. D. Nguyen, “Pattern recognition based movement control and gripping forces control system on arm robot model using LabVIEW,” *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 13, no. 1, pp. 1–14, Jul. 2022, doi: 10.14203/j.mev.2022.v13.1-14.

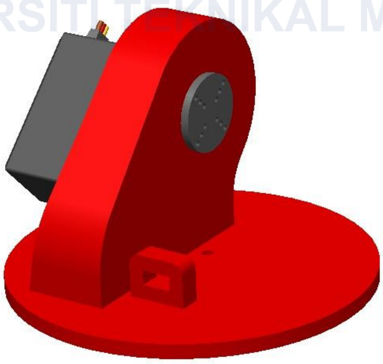
[24] A. Widyacandra, A. R. Al Tahtawi, and M. Martin, “Forward and inverse kinematics modeling of 3-DoF AX-12A robotic manipulator,” *JITEL (Jurnal Ilmiah Telekomunikasi, Elektronika, dan Listrik Tenaga)*, vol. 2, no. 2, pp. 139–150, Sep. 2022, doi: 10.35313/jitel.v2.i2.2022.139-150.

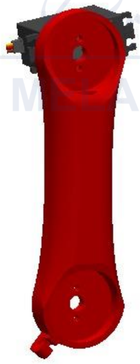
APPENDICES

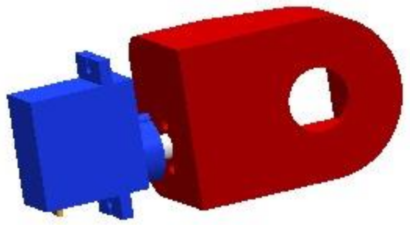
Appendix A

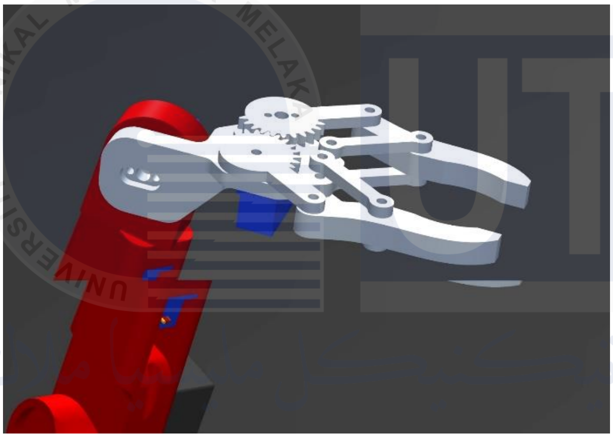
Designed parts in SolidWorks (isometric view):











3D printed parts:

Appendix B

```
#include <SoftwareSerial.h>

#include <AFMotor.h>

#include <Servo.h>

AF_DCMotor motor1(2, MOTOR12_1KHZ);

AF_DCMotor motor2(1, MOTOR12_1KHZ);

AF_DCMotor motor3(4, MOTOR34_1KHZ);

AF_DCMotor motor4(3, MOTOR34_1KHZ);


Servo servo01;

Servo servo02;

Servo servo03;

Servo servo04;

Servo servo05;

Servo servo06;


int val;

int Speed = 255;

SoftwareSerial Bluetooth(A15, 40);  // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)

#define led 14

int wheelSpeed = 1500;


int servo1Pos, servo2Pos, servo3Pos, servo4Pos, servo5Pos, servo6Pos;           // current position
```

```
int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos;          // previous position

int servo01SP[50], servo02SP[50], servo03SP[50], servo04SP[50], servo05SP[50], servo06SP[50];  // for storing positions/steps

int speedDelay = 20;

int index = 0;

int dataIn;

int m = 0;


void setup() {

  // Set initial seed values for the steppers

  pinMode(led, OUTPUT);


  servo01.attach(52);

  servo02.attach(50);

  servo03.attach(48);

  servo04.attach(46);

  servo05.attach(44);

  servo06.attach(38);


  Bluetooth.begin(9600);  // Default baud rate of the Bluetooth module

  Bluetooth.setTimeout(5);

  delay(20);

  Serial.begin(9600);


  // Move robot arm to initial position
```

```
  servo1PPos = 90;

  servo01.write(servo1PPos);

  servo2PPos = 100;

  servo02.write(servo2PPos);

  servo3PPos = 120;

  servo03.write(servo3PPos);

  servo4PPos = 95;

  servo04.write(servo4PPos);

  servo5PPos = 60;

  servo05.write(servo5PPos);

  servo6PPos = 50;

  servo06.write(servo6PPos);

}


void loop() {

  // Check for incoming data

  if (Bluetooth.available() > 0) {

    dataIn = Bluetooth.read();  // Read the data



    if (dataIn == 0) {

      m = 0;

    }

    if (dataIn == 1) {

      m = 1;

    }
```

```
if (dataIn == 2) {
  m = 2;
}
if (dataIn == 3) {
  m = 3;
}
if (dataIn == 4) {
  m = 4;
}
if (dataIn == 5) {
  m = 5;
}
if (dataIn == 6) {
  m = 6;
}
if (dataIn == 7) {
  m = 7;
}
if (dataIn == 8) {
  m = 8;
}
if (dataIn == 9) {
  m = 9;
}
if (dataIn == 10) {
```

```
  m = 10;
}
if (dataIn == 11) {
  m = 11;
}
if (dataIn == 12) {
  m = 12;
}
if (dataIn == 14) {
  m = 14;
}
if (dataIn == 16) {
  m = 16;
}
if (dataIn == 17) {
  m = 17;
}
if (dataIn == 18) {
  m = 18;
}
if (dataIn == 19) {
  m = 19;
}
if (dataIn == 20) {
  m = 20;
```

```
}
if (dataIn == 21) {
  m = 21;
}
if (dataIn == 22) {
  m = 22;
}
if (dataIn == 23) {
  m = 23;
}
if (dataIn == 24) {
  m = 24;
}
if (dataIn == 25) {
  m = 25;
}
if (dataIn == 26) {
  m = 26;
}
if (dataIn == 27) {
  m = 27;
}
// Move the Mecanum wheels platform
if (m == 4) {
  right();
```

```
}
if (m == 5) {
  left();
}
if (m == 2) {
  forward();
}
if (m == 7) {
  back();
}
if (m == 3) {
  topright();
}
if (m == 1) {
  topleft();
}
if (m == 8) {
  bottomright();
}
if (m == 6) {
  bottomleft();
}
if (m == 9) {
  rotateLeft();
}
```

```
if (m == 10) {
  rotateRight();
}
if (m == 0) {
  Stop();
}
// Mecanum wheels speed
if (dataIn > 30 & dataIn < 100) {
  wheelSpeed = dataIn * 20;
}
// Move robot arm
// Move servo 1 in positive direction
while (m == 16) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo01.write(servo1PPos);
  Serial.print(m);
  servo1PPos++;
  delay(speedDelay);
}
// Move servo 1 in negative direction
while (m == 17) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
```

```
  }
  servo01.write(servo1PPos);
  servo1PPos--;
  delay(speedDelay);
}
// Move servo 2
while (m == 19) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo02.write(servo2PPos);
  servo2PPos++;
  delay(speedDelay);
}
while (m == 18) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo02.write(servo2PPos);
  servo2PPos--;
  delay(speedDelay);
}
// Move servo 3
while (m == 20) {
  if (Bluetooth.available() > 0) {
```

```
    m = Bluetooth.read();
  }
  servo03.write(servo3PPos);
  servo3PPos++;
  delay(speedDelay);
}
while (m == 21) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo03.write(servo3PPos);
  servo3PPos--;
  delay(speedDelay);
}
// Move servo 4
while (m == 23) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo04.write(servo4PPos);
  servo4PPos++;
  delay(speedDelay);
}
while (m == 22) {
  if (Bluetooth.available() > 0) {
```

```
    m = Bluetooth.read();
  }
  servo04.write(servo4PPos);
  servo4PPos--;
  delay(speedDelay);
}
// Move servo 5
while (m == 25) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo05.write(servo5PPos);
  servo5PPos++;
  delay(speedDelay);
}
while (m == 24) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo05.write(servo5PPos);
  servo5PPos--;
  delay(speedDelay);
}
// Move servo 6
while (m == 26) {
```

```
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo06.write(servo6PPos);
  servo6PPos++;
  delay(speedDelay);
}
while (m == 27) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo06.write(servo6PPos);
  servo6PPos--;
  delay(speedDelay);
}
// If arm speed slider is changed
if (dataIn > 101 & dataIn < 250) {
  speedDelay = dataIn / 10;  // Change servo speed (delay time)
}
// If button "SAVE" is pressed
if (m == 12) {
  //if it's initial save, set the steppers position to 0
  if (index == 0) {
    motor1.setSpeed(0);
    motor1.run(RELEASE);  //stop the motor when release the button
```

```
    motor2.setSpeed(0);

    motor2.run(RELEASE);

    motor3.setSpeed(0);

    motor3.run(RELEASE);

    motor4.setSpeed(0);

    motor4.run(RELEASE);

  }

  servo01SP[index] = servo1PPos;  // save position into the array

  servo02SP[index] = servo2PPos;

  servo03SP[index] = servo3PPos;

  servo04SP[index] = servo4PPos;

  servo05SP[index] = servo5PPos;

  servo06SP[index] = servo6PPos;

  index++;  // Increase the array index

  m = 0;

}

// If button "RUN" is pressed

if (m == 14) {

  runSteps();


  // If button "RESET" is pressed

  if (dataIn != 14) {

    Stop();

    memset(servo01SP, 0, sizeof(servo01SP));  // Clear the array data to 0

    memset(servo02SP, 0, sizeof(servo02SP));
```

```
      memset(servo03SP, 0, sizeof(servo03SP));

      memset(servo04SP, 0, sizeof(servo04SP));

      memset(servo05SP, 0, sizeof(servo05SP));

      memset(servo06SP, 0, sizeof(servo06SP));

      index = 0;  // Index to 0

    }

  }

 }

 // Monitor the battery voltage

 int sensorValue = analogRead(A0);

 float voltage = sensorValue * (5.0 / 1023.00) * 3;  // Convert the reading values
from 5v to suitable 12V i

 //Serial.println(voltage);

 // If voltage is below 11V turn on the LED

 if (voltage < 11) {

   digitalWrite(led, HIGH);

 } else {

   digitalWrite(led, LOW);

 }

}

void forward() {

 motor1.setSpeed(wheelSpeed);

 motor1.run(FORWARD);

 motor2.setSpeed(wheelSpeed);

 motor2.run(BACKWARD);
```

```
  motor3.setSpeed(wheelSpeed);
  motor3.run(FORWARD);
  motor4.setSpeed(wheelSpeed);
  motor4.run(BACKWARD);
}
void back() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(BACKWARD);
  motor2.setSpeed(wheelSpeed);
  motor2.run(FORWARD);
  motor3.setSpeed(wheelSpeed);
  motor3.run(BACKWARD);
  motor4.setSpeed(wheelSpeed);
  motor4.run(FORWARD);
}
void right() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(FORWARD);
  motor2.setSpeed(wheelSpeed);
  motor2.run(FORWARD);
  motor3.setSpeed(wheelSpeed);
  motor3.run(FORWARD);
  motor4.setSpeed(wheelSpeed);
  motor4.run(FORWARD);
}
```

```
void left() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(BACKWARD);
  motor2.setSpeed(wheelSpeed);
  motor2.run(BACKWARD);
  motor3.setSpeed(wheelSpeed);
  motor3.run(BACKWARD);
  motor4.setSpeed(wheelSpeed);
  motor4.run(BACKWARD);
}
void rotateLeft() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(BACKWARD);  // Left side motors run backward
  motor2.setSpeed(wheelSpeed);
  motor2.run(BACKWARD);

  motor3.setSpeed(wheelSpeed);
  motor3.run(FORWARD);   // Right side motors run forward
  motor4.setSpeed(wheelSpeed);
  motor4.run(FORWARD);
}
void rotateRight() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(FORWARD);   // Left side motors run forward
  motor2.setSpeed(wheelSpeed);
```

```
  motor2.run(FORWARD);


  motor3.setSpeed(wheelSpeed);
  motor3.run(BACKWARD);  // Right side motors run backward
  motor4.setSpeed(wheelSpeed);
  motor4.run(BACKWARD);
}
void topright() {
  motor2.setSpeed(wheelSpeed);
  motor2.run(BACKWARD);
  motor4.setSpeed(wheelSpeed);
  motor4.run(BACKWARD);
}
void bottomright() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(BACKWARD);
  motor3.setSpeed(wheelSpeed);
  motor3.run(BACKWARD);
}
void topleft() {
  motor1.setSpeed(wheelSpeed);
  motor1.run(FORWARD);
  motor3.setSpeed(wheelSpeed);
  motor3.run(FORWARD);
}
```

```
void bottomleft() {

  motor2.setSpeed(wheelSpeed);

  motor2.run(FORWARD);

  motor4.setSpeed(wheelSpeed);

  motor4.run(FORWARD);

}

void Stop() {

  motor1.setSpeed(0);

  motor1.run(RELEASE);  //stop the motor when release the button

  motor2.setSpeed(0);

  motor2.run(RELEASE);

  motor3.setSpeed(0);

  motor3.run(RELEASE);

  motor4.setSpeed(0);

  motor4.run(RELEASE);

}

// Automatic mode custom function - run the saved steps

void runSteps() {

  while (dataIn != 13) {                    // Run the steps over and over again until
"RESET" button is pressed

    for (int i = 0; i <= index - 2; i++) {  // Run through all steps(index)

      if (Bluetooth.available() > 0) {      // Check for incomding data

        dataIn = Bluetooth.read();

        if (dataIn == 15) {       // If button "PAUSE" is pressed

          while (dataIn != 14) {  // Wait until "RUN" is pressed again
```

```
      if (Bluetooth.available() > 0) {
        dataIn = Bluetooth.read();
        if (dataIn == 13) {
          break;
        }
      }
    }
  }
  // If speed slider is changed
  if (dataIn > 100 & dataIn < 150) {
    speedDelay = dataIn / 10;  // Change servo speed (delay time)
  }
  // Mecanum wheels speed
  if (dataIn > 30 & dataIn < 100) {
    wheelSpeed = dataIn * 10;
    dataIn = 14;
  }
}
// Servo 1
if (servo01SP[i] == servo01SP[i + 1]) {
}
if (servo01SP[i] > servo01SP[i + 1]) {
  for (int j = servo01SP[i]; j >= servo01SP[i + 1]; j--) {
    servo01.write(j);
    delay(speedDelay);
```

```
  }
}
if (servo01SP[i] < servo01SP[i + 1]) {
  for (int j = servo01SP[i]; j <= servo01SP[i + 1]; j++) {
    servo01.write(j);
    delay(speedDelay);
  }
}
// Servo 2
if (servo02SP[i] == servo02SP[i + 1]) {
}
if (servo02SP[i] > servo02SP[i + 1]) {
  for (int j = servo02SP[i]; j >= servo02SP[i + 1]; j--) {
    servo02.write(j);
    delay(speedDelay);
  }
}
if (servo02SP[i] < servo02SP[i + 1]) {
  for (int j = servo02SP[i]; j <= servo02SP[i + 1]; j++) {
    servo02.write(j);
    delay(speedDelay);
  }
}
// Servo 3
if (servo03SP[i] == servo03SP[i + 1]) {
```

```
  }
  if (servo03SP[i] > servo03SP[i + 1]) {
    for (int j = servo03SP[i]; j >= servo03SP[i + 1]; j--) {
      servo03.write(j);
      delay(speedDelay);
    }
  }
  if (servo03SP[i] < servo03SP[i + 1]) {
    for (int j = servo03SP[i]; j <= servo03SP[i + 1]; j++) {
      servo03.write(j);
      delay(speedDelay);
    }
  }
  // Servo 4
  if (servo04SP[i] == servo04SP[i + 1]) {
  }
  if (servo04SP[i] > servo04SP[i + 1]) {
    for (int j = servo04SP[i]; j >= servo04SP[i + 1]; j--) {
      servo04.write(j);
      delay(speedDelay);
    }
  }
  if (servo04SP[i] < servo04SP[i + 1]) {
    for (int j = servo04SP[i]; j <= servo04SP[i + 1]; j++) {
      servo04.write(j);
```

```
    delay(speedDelay);
  }
}
// Servo 5
if (servo05SP[i] == servo05SP[i + 1]) {
}
if (servo05SP[i] > servo05SP[i + 1]) {
  for (int j = servo05SP[i]; j >= servo05SP[i + 1]; j--) {
    servo05.write(j);
    delay(speedDelay);
  }
}
if (servo05SP[i] < servo05SP[i + 1]) {
  for (int j = servo05SP[i]; j <= servo05SP[i + 1]; j++) {
    servo05.write(j);
    delay(speedDelay);
  }
}
// Servo 6
if (servo06SP[i] == servo06SP[i + 1]) {
}
if (servo06SP[i] > servo06SP[i + 1]) {
  for (int j = servo06SP[i]; j >= servo06SP[i + 1]; j--) {
    servo06.write(j);
    delay(speedDelay);
```

```
      }
    }
    if (servo06SP[i] < servo06SP[i + 1]) {
      for (int j = servo06SP[i]; j <= servo06SP[i + 1]; j++) {
        servo06.write(j);
        delay(speedDelay);
      }
    }
  }
}
}
```