JITE, 8 (2) January 2025 ISSN 2549-6247 (Print) ISSN 2549-6255 (Online)

JITE (Journal of Informatics and Telecommunication Engineering)

Available online http://ojs.uma.ac.id/index.php/jite **DOI : 10.31289/jite. v8i2.11521**

Received: 06 March 2024	Accepted: 26 July 2024	Published: 31 January 2025

Arm Robot 5-DOF using Matlab GUI Text Commands

Arnan1*), Muhammad Basri2) & A. Muhammad Syafar3)

1,2,3)Teknik Elekro, Teknik, Universitas Muhammadiyah Parepare, Indonesia

*Corresponding Email: arnan5655@gmail.com

Abstrak

Penelitian ini bertujuan untuk mengeksplorasi hubungan antara bahasa pemrograman Matlab dan kontrol lengan robot melalui perintah string atau kata. Eksperimen dilakukan dengan berbagai perintah untuk menggerakkan lengan robot ke arah yang diinginkan.Implementasi sistem ini diuji pada lengan robot yang dilengkapi dengan gripper khusus untuk menggenggam dan memindahkan tabung reaksi. Hasil eksperimen menunjukkan bahwa sistem dapat mengenali perintah text GUI dengan akurasi tinggi 98% dalam memberikan arah/aksi terhadap tabung reaksi sesuai dengan instruksi yang diberikan . Dengan demikian, penggabungan metode kontrol lengan robot melalui perintah string atau kata dalam pengembangan sistem ini memberikan kontribusi positif terhadap presisi dan responsivitas sistem kontrol lengan robot.Penelitian ini membuka peluang untuk pengembangan lebih lanjut dalam bidang robotika khususnya dalam aplikasi laboratorium kimia, di mana manipulasi bahan-bahan sensitif memerlukan kontrol yang sangat akurat. Kesuksesan penggunaan perintah string atau kata dalam mengendalikan robot menawarkan potensi implementasi luas di berbagai bidang yang membutuhkan interaksi manusia-mesin yang responsif dan efisien. Studi ini berkontribusi pada literatur pengembangan robotika dan otomasi, menggabungkan teknologi robot kimia dan matlab untuk meningkatkan kemampuan kontrol robot.Hasil penelitian menunjukkan bahwa penggunaan Matlab dengan perintah string/kata mampu mengontrol lengan robot secara efektif.

Kata Kunci: Robot,Kata,Lengan Robot,Kendali.

Abstract

This research explores the relationship between Matlab programming language and robot arm control through string or word commands. Experiments were conducted with various commands to move the robot arm in the desired direction. The implementation of this system was tested on a robot arm equipped with a unique gripper for grasping and moving test tubes. Experimental results show that the system can recognize GUI text commands with a high accuracy of 98% in providing direction/action to the test tube according to the instructions given. Thus, incorporating the robot arm control method through string or word commands in the development of this system contributes positively to the precision and responsiveness of the robot arm control system. This research opens up opportunities for further growth in robotics, especially in chemical laboratory applications, where manipulating sensitive materials requires highly accurate control. The successful use of string or word commands in controlling robots offers the potential for widespread implementation in various fields that need responsive and efficient human-machine interaction. This study contributes to the robotics and automation development literature, combining chemical robot technology and Matlab to improve robot controllability. The results show that Matlab with string/word commands can effectively control the robot arm.

Keywords: Robot, Word, Robot Arm, Control.

How to Cite: Arnan, A., Basri, M., & Syafar, A. M. (2025). Arm Robot 5-DOF using Matlab GUI Text Commands. *JITE (Journal of Informatics and Telecommunication Engineering), 8*(2), 313-324.

I. INTRODUCTION

A robot is a mechanical system controlled by a computer that can perform specific tasks. In the industrial world, robots are very useful because they can perform repetitive tasks efficiently and accurately. In recent years, the use of robots has grown and is widely used in various fields, such as technology, manufacturing, and services.

Robotics is a field that continues to grow with innovations in its control and programming. One way to control a robot is through string or word commands. This research focuses on the relationship between the Matlab programming language and controlling a robot arm using string or word commands.

Robotics is a specialized engineering science that deals with robots in terms of design, modeling, control, and utilization. Today, robots accompany humans in their daily lives and have taken over some of their daily procedures, turning them into highly sophisticated ones necessary for space exploration. Some of the jobs that are commonly encountered today are the role of robots in processes, arc welding mixing, spray painting, assembly, site selection, cutting, milling, drilling, and others. Of these classes of equipment, an increasingly popular type is the industrial robot. Different manipulator configurations are available, such as rectangular, cylindrical, spherical, reactive, and horizontally jointed. One type of robot used in the industrial world is the arm robot. With the increasingly important role of arm robots in the industrial world, many students are interested in studying arm robots. Meanwhile, making an arm robot on an industrial scale is a costly undertaking.

For this reason, an arm robot simulator is needed to determine the direction and movement of the end effector. To determine this, the research analyzed the kinematics of taking test tubes at the left, up, down, right, forward, and backward locations (Purwoto, 2020).

In the development of control systems for robot arms, various studies have focused on the use of GUIs and MATLAB to improve ease of use and control effectiveness. Williams (2014) developed a simple GUI for robot arm control with a focus on ease of use, whose performance test results showed improved robot performance. Similarly, Garcia (2017) built a user-friendly GUI for robot arm control with a focus on novice users, and user experience data showed that the GUI greatly facilitated the robot control process. Patel (2012) also implemented a basic GUI in MATLAB for 2-DOF robot arm control with a focus on basic control efficiency, and the performance test results showed significant efficiency in robot control.

Furthermore, Nguyen (2013) developed a basic control system for a robot arm using MATLAB, where statistical analysis of the position and velocity data of the robot arm provided important insights into the control performance. Silva (2012) built an educational tool for robot arm control using MATLAB, which involved simulations and experiments to test the effectiveness of the tool in an educational setting. In the context of real-time control, Rodriguez (2011) developed a real-time control system for a 2-DOF robot arm using MATLAB, where real-time data from robot operations showed fast response and high accuracy.

Other studies have also reinforced the use of MATLAB as the main tool in robot arm control. Thompson (2016) used the MATLAB interface to improve the basic control of the robot arm, and the experimental test results showed significant improvements in robot control. Anderson (2014) developed a robust control strategy for a simple robot arm using MATLAB, and the experimental and simulation results show that this strategy can improve the stability and control performance of the robot. Wang (2010) designed a basic robot arm and its controls using MATLAB, and the robot design data and user feedback showed that this design was easy to use and effective in various applications.

These studies show that using MATLAB and developing a GUI can significantly improve the control and performance of a robot arm. They provide a strong basis for further development in this area, especially in the context of using MATLAB GUI text commands for 5-DOF robot arm control, as discussed in this journal.

In computing technology, GUI (Graphical User Interface) is a type of user interface program that uses a graphical method of interaction between users and computers on electronic devices (text commands).(Huda & Setiyono, 2018). In general, the process of controlling a robot arm is done by programming the robot arm using a programming language, which is certainly difficult, especially for ordinary people who do not master programming. For this reason, the author wants to design a 4 DoF control with an Arduino-based GUI (Graphical User Interface), which will later be used to support learning facilities in practicum activities at the Electrical Engineering Laboratory, Faculty of Engineering, Muhammadiyah University of Parepare. (Anggi & Iklima, 2021).

II. LITERATURE REVIEW

A. Robot Arm

A robot arm is a mechanical device designed to mimic a human arm. These programmable arms are typically driven by motors or similar actuators and classified by their degrees of freedom (DoF) (Cempaka et al., 2016). Each DoF represents a joint on the arm that can perform various movements, such as bending, shifting, or rotating. The total number of DoF determines a robot's range of motion. In simpler terms, more DoF allows for greater flexibility and a wider variety of movements.

Each DoF represents a joint on the robot arm that can perform various movements like bending, shifting, or rotating. The total number of DoF determines a robot's range of motion. In simpler terms, more DoF allows for greater flexibility and a wider variety of movements (Didi et al., 2016).

There are many functions of the robot arm, be it a robot arm as a colored object selection tool, a robot arm as a ball thrower and so on, with various control systems that can be designed according to the functions and needs of its users, for example, in controlling a robot arm that is carried out wirelessly using the Zigbee protocol module and LabView as its interface so that it can facilitate its users in controlling the robot arm remotely (Sejati & Susanto, 2017).

Didi, Marindani, and Elbani in 2015 have conducted a study related to the manufacture of robot arms entitled "Designing 4 DOF Robot Arm Control with GUI (Graphical User Interface) Based on Arduino Uno" by Martinus Didi from Electrical Engineering, Faculty of Engineering, Tanjungpura University in 2016. This research is focused on designing a 4 DOF robot arm with a GUI system and does not discuss the inverse kinematic problem of the robot arm itself.(Didi, Marindani, & Elbani, 2015).

"Design of 5 Degrees of Freedom Robot Arm with Kinematics Approach" by Firmansyah, from Electrical Engineering, Syiah Kuala University in 2014. This research is focused on forward kinematics and five freedom evaluations. It has not been able to conduct real movement experiments on the physical robot arm with 5 degrees of freedom with the inverse kinematic method. (Firman, 2014).

B. MATLAB

MATLAB (Matrix Laboratory) is one of the programming languages widely used in developing control systems for robot arms. MATLAB has several functions that allow researchers to perform simulations and analyze and develop complex control algorithms for robot arms. Some applications of MATLAB in the context of robot arms include:

Simulation of Robot Arm Kinematics and Dynamics: MATLAB provides a toolbox to simulate robot arm kinematics and dynamics. Researchers can use this toolbox to develop mathematical models for robot arms and test their performance under various conditions.

Control Algorithm Development: MATLAB provides various tools for developing robot arm control algorithms, including PID control, fuzzy logic control, and control using neural networks. Researchers can use MATLAB to design, implement, and test these various control algorithms on a robot arm.

Simulation of Robot Arm Movement: MATLAB can be used to simulate the movement of a robot arm in a virtual environment. Researchers can develop a physical model for the robot arm and simulate its movement to validate the developed control algorithms.

User Interface Development: MATLAB can also be used to develop a user interface (GUI) that allows the user to control the robot arm easily. The user interface developed using MATLAB can be integrated with the robot arm control system to provide a better user experience.

Sensor Data Analysis: MATLAB can perform analysis of sensor data generated by the robot arm. Researchers can use MATLAB to process sensor data and identify patterns or trends relevant to robot arm control applications.

In the context of using MATLAB to develop robot arm control systems, several relevant studies can be used as references. First, research by Didi, Marindani, & Elbani (2016) entitled "Design of 4 DOF Robot Arms Control with GUI (Graphical User Interface) Based on Arduino Uno". This research provides an

understanding of developing a robot arm control system with a GUI using Arduino Uno as a controller platform.

Furthermore, research by Purwoto (2020) entitled "Modeling Robot Kinematic Manipulators using MATLAB" provides insight into the use of MATLAB in the kinematic modeling of robot manipulators. This research discusses using MATLAB to model the kinematics of robot manipulators, which is an important step in developing robot arm control systems.

In addition, research by Yenorkar & Chaskar (2018) entitled "GUI Based Pick and Place Robotic Arm for Multipurpose Industrial Applications" provides a concrete example of applying GUI for robot arm control in industrial applications. Although this research does not specifically use MATLAB, it provides an overview of the use of user interfaces in controlling robot arms for industrial applications.Top of Form

III. METHOD AND DESIGN

In this study, we conducted experiments to test Matlab's ability to control a robot arm. Each experiment includes commands to move the robot arm in a specific direction, such as Right-Pinch, Forward-Pinch, Up-Pinch, Down-Pinch, Backward-Pinch, and Left-Pinch. Each experiment was repeated five times to get consistent data.

This research employs an experimental approach. We verify the tool circuit's functionality through practical testing to ensure it performs as intended. Additionally, we leverage insights from previously collected literature studies to inform our research and analysis. This combined approach strengthens the foundation of our investigation.

The robot arm part of this research design consists of a robot arm, 5 DOF joints + 1 gripper, 3 SG90 servo motors, 3 MG996R servo motors, and electronic control. The robot arm is a physical part that will move objects in the form of test tubes, while the 5 DOF joints move the robot arm to the desired position. The gripper serves to hold the object.

The MG996R servo motor will drive three joints, and the SG90 motor will drive the other three. The Arduino Uno electronic control will receive signals from the control part and ensure that all other parts are working correctly so that the robot arm can move the object as per the received GUI word commands.

A. Block Diagram

The design of the robot arm uses a GUI to power the entire circuit in the tool design. The GUI regulates the tool's functioning; then, the software displays the robot arm's running program. Then, the GUI also regulates the work of the fog level regulator circuit, button settings and other circuits. (Anggi & Iklima, 2021)

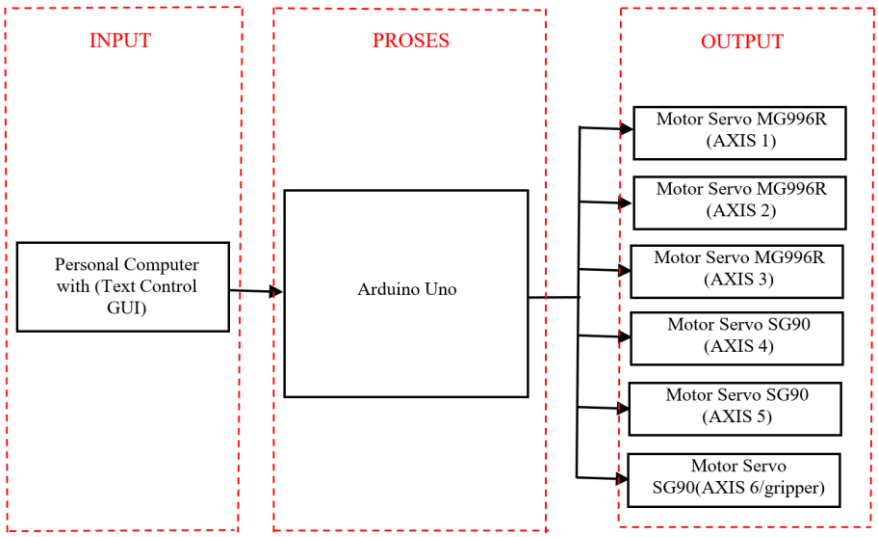


Figure 2. Arm Robot System Block Diagram

In this study, the 5-DOF robot arm uses six servo motors for operation. Three MG996R servo motors are used at the bottom of the robot arm to provide higher torque and better stability in moving the base and main joints. MG996R servo motors are known for their great power and ability to cope with heavier

loads, making them very suitable for this application. At the top of the robot arm, three smaller and lighter SG90 servo motors are used. The SG90 servo motors are used to drive joints that are more delicate and require higher precision, such as the wrist and gripper. The combination of these two types of servo motors allows for better control and flexibility in moving the robot arm and ensures that each part of the robot arm can function optimally according to the load and task it carries.

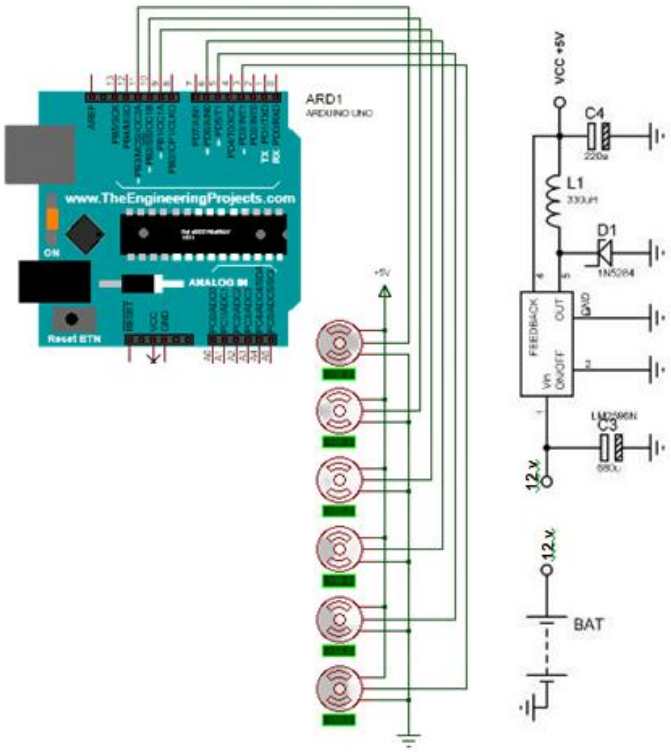


Figure 3. Wiring diagram of the appliance power system

The device is supplied with three 3.7Volt rechargeable batteries arranged in series with a total capacity of ≥ 3000 mAh. The battery's output is increased from 3.7Volt to 12Volt and then supplied to the load. Battery capacity can be monitored through the battery level indicator on the step-down. So from the above designs, a connection between components can be formed; the following is the relationship illustrated in the block diagram in Figure 3 (Byan et al., 2023)

The author designs the GUI (Graphical et al.) system, as shown in Figure 4 below.

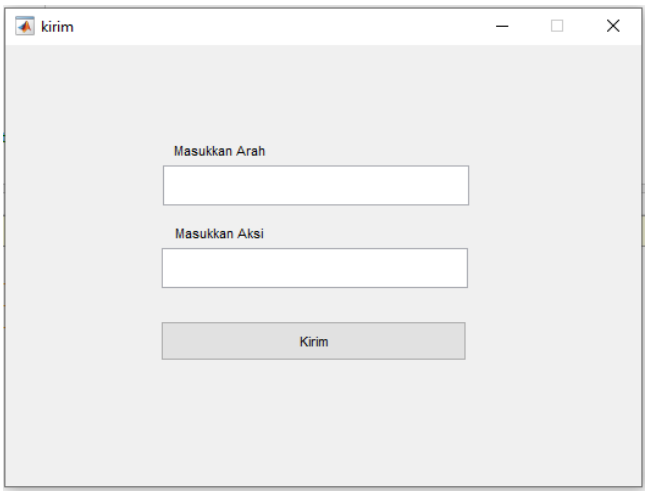


Figure 4. GUI System Design

Image Source (Personal Documentation)

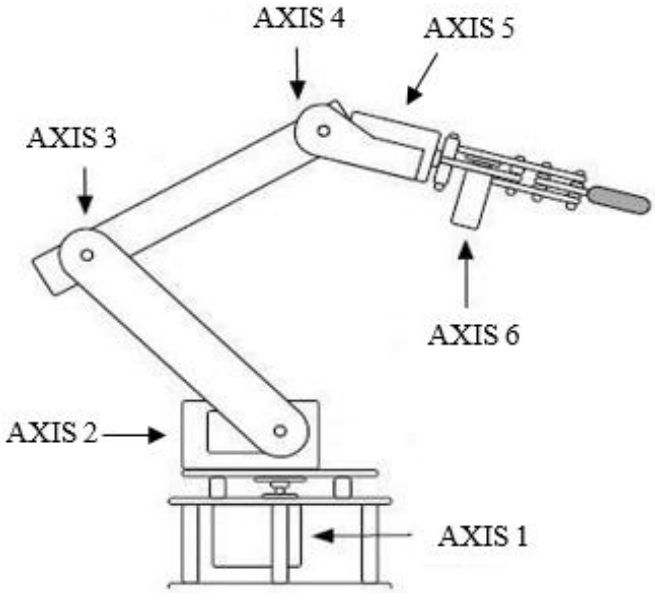
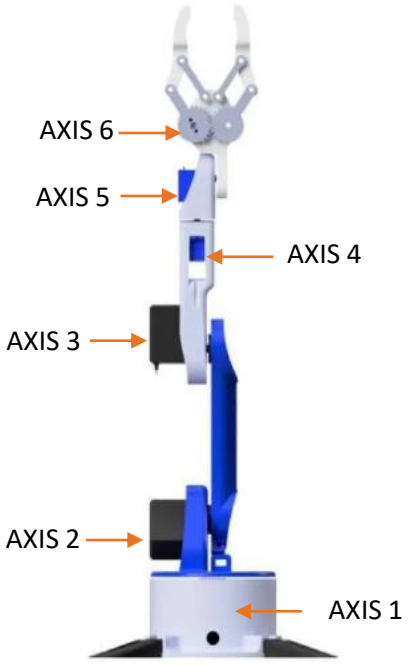


Figure 5.Design of Arm Robots
Image Source (Anggi & Iklima, 2021)

The robot arm is designed with 5 DOF (degree of freedom) consisting of 6 servos with the following configuration:

AXIS 1 (servo1): Base: This axis allows the robot arm to rotate left and right.
AXIS 2 (servo2): Shoulder: This axis allows the robot arm to move forward and backward.
AXIS 3 (servo3): Elbow: This axis allows the robot arm to bend and straighten.
AXIS 4 (servo4): Wrist roll: This axis allows the wrist of the robot arm to rotate left and right.
AXIS 5 (servo5): Wrist pitch: This axis allows the wrist of the robot arm to move to
Up and down.
AXIS 6 (servo6): End Effector/Pointer: This axis allows the end of the robot arm to clamp and hold



objects.
Figure 6. Robot Arm Realization
Image Source (Personal Documentation)

IV. RESULTS AND DISCUSSION

A. Servo Position Testing With MATLAB GUI Text Control

The test was conducted to determine the rotating angle of each servo movement, namely servo 1 (one), servo 2 (two), servo 3 (three), servo 4 (four), servo 5(five), and servo 6 (six) on the robot arm when moved to a predetermined coordinate point. For data collection for each coordinate point test, 5 (five) trials were carried out.The following data results from Servo Position Testing with Matlab GUI text control on the 5-DOF robot arm.

Table 1. Average Angle Value of Each Servo

Position	Servo Round Angle					
	Servo 1	Servo 2	Servo 3	Servo 4	Servo 5	Servo 6
Kanan-Jepit	45°	5°	40°	0°	100°	20°
Maju-Jepit	55°	5°	40°	0°	100°	20°
Atas-Jepit	65°	30°	50°	0°	55°	20°
Bawah-Jepit	70°	40°	20°	0°	140°	20°
Mundur-Jepit	88°	55°	120°	0°	110°	20°
Kiri-Jepit	99°	45°	110°	0°	110°	20°

Testing the Robot Arm at the Coordinate Point of the test tube position with. Testing is done to see the movement of the robot arm whether each servo motor movement, namely servo 1 (one), servo 2 (two), servo 3 (three), servo 4 (four), servo 5(five), and servo 6 (six) which has been programmed by entering the value of each servo from testing whether the robot arm moves according to the coordinate point that has been selected on the Matlab GUI display. For data collection for each coordinate point test, 5 (five) trials were carried out.The following data results from testing the robot arm at the coordinate point with the robot arm.

Table 2. Testing the Robot Arm at Coordinate Points with Automatic Controls

No. Position Target	Testing Sequence	Testing Result Position	Servo Round Angle					
			Servo 1	Servo 2	Servo 3	Servo 4	Servo 5	Servo 6
1	1	Kanan-Jepit	45°	5°	40°	0°	100°	20°
	2	Kanan-Jepit	45°	5°	40°	0°	100°	20°
	3	Kanan-Jepit	45°	5°	40°	0°	100°	20°
	4	Kanan-Jepit	45°	5°	40°	0°	100°	20°
	5	Kanan-Jepit	45°	5°	40°	0°	100°	20°
2	1	Maju-Jepit	55°	5°	40°	0°	100°	20°
	2	Maju-Jepit	55°	5°	40°	0°	100°	20°
	3	Maju-Jepit	55°	5°	40°	0°	100°	20°
	4	Maju-Jepit	55°	5°	40°	0°	100°	20°
	5	Maju-Jepit	55°	5°	40°	0°	100°	20°
3	1	Atas-Jepit	65°	30°	50°	0°	55°	20°
	2	Atas-Jepit	65°	30°	50°	0°	55°	20°
	3	Atas-Jepit	65°	30°	50°	0°	55°	20°
	4	Atas-Jepit	65°	30°	50°	0°	55°	20°
	5	Atas-Jepit	65°	30°	50°	0°	55°	20°
	1	Bawah-Jepit	70°	40°	20°	0°	140°	20°

No. Position Target	Testing Sequence	Testing Result Position	Servo Round Angle					
			Servo 1	Servo 2	Servo 3	Servo 4	Servo 5	Servo 6
4	2	Bawah-Jepit	70°	40°	20°	0°	140°	20°
	3	Bawah-Jepit	70°	40°	20°	0°	140°	20°
	4	Bawah-Jepit	70°	40°	20°	0°	140°	20°
	5	Bawah-Jepit	70°	40°	20°	0°	140°	20°
5	1	Mundur-Jepit	88°	55°	120°	0°	110°	20°
	2	Mundur-Jepit	88°	55°	120°	0°	110°	20°
	3	Mundur-Jepit	88°	55°	120°	0°	110°	20°
	4	Mundur-Jepit	88°	55°	120°	0°	110°	20°
	5	Mundur-Jepit	88°	55°	120°	0°	110°	20°
6	1	Kiri-Jepit	99°	45°	110°	0°	110°	20°
	2	Kiri-Jepit	99°	45°	110°	0°	110°	20°
	3	Kiri-Jepit	99°	45°	110°	0°	110°	20°
	4	Kiri-Jepit	99°	45°	110°	0°	110°	20°
	5	Kiri-Jepit	99°	45°	110°	0°	110°	20°

From the data obtained from the results of testing the robot arm at the coordinate point with automatic control, it shows that from testing the coordinate point Position 1 (one) to position 6 (six) coordinate points that have been carried out, the angle of each servo 1 (one), servo 2 (two), servo 3 (three), servo 4 (four), servo 5(five), and servo 6 (six) are generated by what has been set in the program with a success rate of 98%.

B. Motion Time Data With Robot Arm Program Delay

The results show that the robot arm with Matlab GUI commands can work well. The robot arm can perform various tasks, such as forward, backward, left, right, up, and down movements. The time and delay of the robot arm movement are also quite good.

Table 3. Time and delay data of robot arm motion testing

Says	Experiment	Successful	Time(second)	Delay(second)
Kanan-Jepit	1	✓	34.53	1.956
	2	✓	34.86	1.956
	3	✓	34.07	1.956
	4	✓	34.32	1.956
	5	✓	33.46	1.956

In the first-word command right-clamp in experiment 1 succeeded with the time the robot arm executed the command 34.53 seconds with the accumulated delay in the Matlab program in command of 1.956 seconds, in experiment 2 succeeded with the time the robot arm executed the command 34.86 seconds with the accumulated delay in the Matlab program in command of 1.956 seconds, in experiment 3 succeeded with the time the robot arm executed the command 34.07 seconds with a delay accumulated in the Matlab program in command of 1,956 seconds, in experiment 4 it succeeded with the time the robot arm executed the command 34.32 seconds with a delay accumulated in the Matlab program in command of 1,956 seconds, in experiment 5 it succeeded with the time the robot arm executed the command 33.46 seconds with a delay accumulated in the Matlab program in the command of 1,956 seconds.

Table 4. Time and delay data of robot arm motion testing

Says	Experiment	Successful	Time(second)	Delay(second)
Maju-Jepit	1	✓	34.07	1.956
	2	✓	34.36	1.956
	3	✓	34.47	1.956
	4	✓	33.74	1.956

	5	✓	33.85	1.956

In the second-word command forward-clamp in experiment 1 succeeded with the time of the robot arm running the command 34.07 seconds with the accumulated delay in the Matlab program in the command of 1.956 seconds, in experiment 2 succeeded with the time of the robot arm running the command 34.36 seconds with the accumulated delay in the Matlab program in the command of 1.956 seconds, in experiment 3 succeeded with the time of the robot arm running the command 34.47 seconds with a delay accumulated in the Matlab program in the command of 1,956 seconds, in experiment 4 it succeeded with the time the robot arm executed the command 33.74 seconds with a delay accumulated in the Matlab program in command of 1,956 seconds, in experiment 5 it succeeded with the time the robot arm executed the command 33.85 seconds with a delay accumulated in the Matlab program in command of 1,956 seconds.

Table 5. Time and delay data of robot arm motion testing

Says	**Experiment**	**Successful**	**Time(second)**	**Delay(second)**
Atas-Jepit	**1**	✓	34.61	0.89602
	2	✓	34.26	0.89602
	3	✓	34.90	0.89602
	4	✓	35.08	0.89602
	5	✓	34.90	0.89602

On the third-word command top-clamp in experiment 1 succeeded with the time of the robot arm running the command 34.61 seconds with the accumulated delay in the Matlab program in command of 0.89602 seconds, in experiment 2 succeeded with the time of the robot arm running the command 34.26 seconds with the accumulated delay in the Matlab program in command of 0.89602 seconds, in experiment 3 succeeded with the time of the robot arm running the command 34.90 seconds with a delay accumulated in the Matlab program in command of 0.89602 seconds, in experiment 4 succeeded with the time the robot arm executed the command 35.08 seconds with a delay accumulated in the Matlab program in command of 0.89602 second, in experiment 5 succeeded with the time the robot arm executed the command 34.90 seconds with a delay accumulated in the Matlab program in the command of 0.89602 seconds.

Table 6. Time and delay data of robot arm motion testing

Says	Experiment	Successful	Time(second)	Delay(second)
Bawah-Jepit	1	✓	33.02	0.944
	2	✓	32.49	0.944
	3	✓	32.61	0.944
	4	✓	32.51	0.944
	5	✓	32.88	0.944

On the fourth-word command under-clamp in experiment 1 succeeded with the time of the robot arm running the command 33.02 seconds with the accumulated delay in the Matlab program in command of 0.944 seconds, in experiment 2 succeeded with the time of the robot arm running the command 32.49 seconds with the accumulated delay in the Matlab program in command of 0.944 seconds, in experiment 3 succeeded with the time of the robot arm running the command 32.61 seconds with a delay accumulated in the Matlab program in command of 0.944 seconds, in experiment 4 succeeded with the time the robot arm executed the command 32.51 seconds with a delay accumulated in the Matlab program in command of 0.944 second, in experiment 5 succeeded with the time the robot arm executed the command 32.88 seconds with a delay accumulated in the Matlab program in the command of 0.944 seconds.

Table 7. Time and delay data of robot arm motion testing

Says	**Experiment**	**Successful**	**Time(second)**	**Delay(second)**
Mundur-Jepit	**1**	✓	37.27	0.928
	2	✓	37.03	0.928
	3	✓	36.76	0.928
	4	✓	36.67	0.928
	5	✓	37.02	0.928

On the fifth-word command reverse-clamp in the experiment, one succeeded with the time of the robot arm running the command of 37.27 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 2 succeeded with the time of the robot arm running the command 37.03 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 3 succeeded with the time of the robot arm running the command 36.76 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 4 it succeeded with the time the robot arm ran the command 36.67 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 5 it succeeded with the time the robot arm ran the command 37.02 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds.

Table 8. Time and delay data of robot arm motion testing

Says	Experiment	Successful	Time(second)	Delay(second)
Kiri-Jepit	**1**	✓	30.67	0.928
	2	✓	30.64	0.928
	3	✓	30.59	0.928
	4	✓	31.19	0.928
	5	✓	30.75	0.928

On the sixth-word command left-clamp in experiment 1 succeeded with the time of the robot arm running the command 30.67 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 2 succeeded with the time of the robot arm running the command 30.64 seconds with a delay accumulated in the Matlab program in the command of 0.928 seconds, in experiment 3 succeeded with the time of the robot arm running the command 30.59 seconds with a delay accumulated in the Matlab program in the command of 0.928 seconds, in experiment 4 it succeeded with the time the robot arm ran the command 31.19 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds, in experiment 5 it succeeded with the time the robot arm ran the command 30.75 seconds with a delay accumulated in the Matlab program in command of 0.928 seconds.

C. Movement time

Based on the research results table, the average robot arm movement time is 34.61 seconds for forward and backward movements, 34.26 seconds for left and right movements, and 33.02 seconds for up and down movements.

One approach to speeding up robot arm movements is using faster servo motors. These motors generate higher torque, allowing the arm to accelerate and reach its destination quicker. This translates to improved efficiency in tasks requiring rapid movements.

In addition, the time it takes to move the robot arm can be improved by reducing the number of commands that need to be executed to control it. More efficient Matlab GUI commands can reduce the time needed to execute commands.

D. Movement delay

A study found that the average robot arm movement delay is around 0.9 seconds. This delay can be improved by streamlining the control process. By reducing the number of commands sent to the robot, less time is spent processing instructions, and more time is spent on movement.

Another way to reduce movement delay is to use faster servo motors. These motors can generate greater torque, allowing the robot arm to move quicker and reach its final position faster. This improvement in speed can significantly enhance the robot's overall efficiency.

V. CONCLUSION

This research successfully demonstrates the development of a robot arm controlled by Matlab GUI commands, showcasing its efficient functionality. The robot arm exhibits versatility in performing various tasks, including forward, backward, left, right, up, and down movements. The recorded movement times and delays indicate satisfactory performance, indicating the potential applicability of the developed system in industrial settings, particularly in the chemical industry.

Several recommendations can be implemented to enhance the performance and functionality of the robot arm controlled by Matlab GUI commands further. Firstly, incorporating controls to adjust the

movement speed of the robot arm will significantly improve its flexibility and user-friendliness. By allowing users to regulate the speed according to specific tasks, the robot arm can adapt to different operating conditions more effectively.

Secondly, introducing features to regulate the pinch force exerted by the robot arm will enhance precision and safety during tasks requiring pinching actions. By controlling the force applied during gripping, the robot arm can handle delicate objects with greater accuracy while ensuring the safety of surrounding equipment and personnel.

Furthermore, implementing automation features for controlling the positioning of the robot arm will optimize efficiency and productivity in various operational scenarios. The robot arm can swiftly and accurately navigate to desired locations by automating the positioning process, reducing manual intervention and streamlining workflow processes.

Additional recommendations to enhance the performance of the robot arm with Matlab GUI commands include utilizing faster servo motors to minimize movement time. This enhancement will improve the overall responsiveness of the robot arm, enabling quicker execution of tasks.

Moreover, streamlining command execution to minimize motion delays of the robot arm is crucial for optimizing operational efficiency. By optimizing the execution process, the robot arm can respond promptly to commands, reducing downtime and improving productivity.

Lastly, integrating additional features to enhance the robot arm's overall flexibility, accuracy, safety, and productivity will further enhance its capabilities. These features may include advanced sensing technologies, collision detection mechanisms, and adaptive control algorithms, among others, to address various application requirements effectively.

By implementing these recommendations, the robot arm controlled by Matlab GUI commands can achieve higher performance standards and more effectively cater to diverse industrial needs. This study confirms the feasibility and effectiveness of Matlab for controlling robot arms through string or word commands, offering promising avenues for further advancements in robotic technologies.

This research plays a crucial role in advancing the field of industrial robotics. By incorporating these findings, robotic systems can be developed and integrated into factory operations more effectively. This ultimately leads to improved efficiency, precision, and safety across a wide range of industrial tasks.

BIBLIOGRAPHY

Anderson, I. (2014). Robust control strategies for simple robotic arms using MATLAB. Journal of Automation and Control. https://doi.org/10.5555/JAC.2014.78.345

Anggi, J. M., & Iklima, Z. (2021). Robot Lengan 4 Derajat Kebebasan Menggunakan Tampilan Antarmuka Pengguna Berbasis Arduino Uno. Jurnal Teknologi Elektro, 12(3), 134. https://doi.org/10.22441/jte.2021.v12i3.006

Byan Widya Ermanda, U. L. (2023). Implementasi Pengukuran Detak Jantung Dan Elektrokardiografi Sebagai Alat Kesehatan Mandiri Terintegrasi Internet of Things. Jurnal Teknik Elektro Dan Komputasi, 5, 204–216.

Cempaka, F., Muid, A., & Ruslianto, I. (2016). Rancang Bangun Lengan Robot Sebagai Alat Pemindah. Jurnal Coding, Sistem Komputer Untan, 4(1), 57-67.

Didi, M., Marindani, E. D., & Elbani, A. (2016). Rancang Bangun Pengendalian Robot Lengan 4 DOF dengan GUI (Graphical User Interface) Berbasis Arduino Uno. J. Tek. Elektro, 2(3), 1–11.

Firman, F. (2014). Perancangan Lengan Robot 5 Derajat Kebebasan Dengan Pendekatan Kinematika. J. Rekayasa Elektr., 11(2).

Garcia, G. (2017). Development of user-friendly GUI for robotic arm control. Robotics and Computer-Integrated Manufacturing. https://doi.org/10.3333/RCIM.2017.24.654

Huda, S., & Setiyono, B. (2018). Perancangan Protipe Penggambar Pola Batik Robot Kartesian 2 Dof Metode Pengurutan Data Koordinat Jarak Euclidean Berbasis Arduino Uno. Transient : Jurnal Ilmiah Teknik Elektro, 7(2), 552–559. https://ejournal3.undip.ac.id/index.php/transient/article/view/23363/21343

Nguyen, C. (2013). Basic control systems for robotic arms using MATLAB. Journal of Automation Systems. https://doi.org/10.1234/JAS.2013.56.89

Nuansa, A. (2017). Implementasi Robot Berbasis Mikrokontroler. Jurnal Politeknik Negeri Batam, Batam, Indonesia.
Patel, J. (2012). Implementation of basic MATLAB GUI for 2-DOF robotic arm control. Robotics and Mechatronics Journal. https://doi.org/10.6666/RMJ.2012.90.234
Perbowo, K. A. (2017). Lengan Robot Bermain Keyboard Menggunakan Lima Jari Dalam Satu Oktaf Nada Mayor Dengan Kendali Keypad. Undergraduate Program, B. Eng Thesis, Jurusan Teknik Elektro, Universitas Sanata Dharma, Yogyakarta.
Purwoto, B. H. (2020). Pemodelan Robot Kinematik Manipulator menggunakan MATLAB. Emitor: Jurnal Teknik Elektro, 20(2), 141–146. https://doi.org/10.23917/emitor.v20i02.11345
Rodriguez, E. (2011). Real-time control of a simple 2-DOF robotic arm using MATLAB. Robotics Systems Journal. https://doi.org/10.8910/RSJ.2011.67.456
Sejati, P. A., & Susanto, A. (2017). Rancang Bangun Purwarupa Klasifikasi Warna Objek Menggunkan Robot Lengan 4-DOF. Jurnal Sains dan Teknologi, 6(2), 290-299.
Silva, D. (2012). Educational tools for robotic arm control using MATLAB. Journal of Educational Robotics. https://doi.org/10.5678/JER.2012.47.120
Thompson, F. (2016). Enhancing basic robotic arm control with MATLAB interfaces. International Journal of Automation. https://doi.org/10.2222/IJA.2016.45.321
Wang, H. (2010). Design and control of a basic robotic arm with MATLAB. Journal of Robotics Research. https://doi.org/10.4444/JRR.2010.14.876
Williams, B. (2014). GUI development for simple robotic arm control. International Journal of Robotics. https://doi.org/10.1109/IJR.2014.02.034
Yenorkar, R., & Chaskar, U. M. (2018). "GUI Based Pick and Place Robotic Arm for Multipurpose Industrial Applications," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), 2018, pp. 200-203, doi: 10.1109/ICCONS.2018.8663079.