Implementation of Finger Gesture-Based Medicine Delivery Robot Control System with MediaPipe

1*st Ahmad Zarkasi
Department of Computer Science
Sriwijaya University
Palembang, South Sumatera
ahmadzarkasi@unsri.ac.id

2nd Gede Pradnyananda
Department of Computer System
Sriwijaya University
Palembang, South Sumatera
gedenanda4444@gmail.com

3rd Sarmayanta Sembiring
Department of Computer Engineering
Sriwijaya University
Palembang, South Sumatera
yanta@unsri.ac.id

4th Aditya P. P. Prasetyo
Department of Computer Engineering
Sriwijaya University
Palembang, South Sumatera
aditrecca@gmail.com

5th Ricy Firnando
Department of Computer Engineering
Sriwijaya University
Palembang, South Sumatera
ricyfirnando@unsri.ac.id

6th Abdurahman
Department of Computer System
Sriwijaya University
Palembang, South Sumatera
abdurahman@unsri.ac.id

*Corresponding author: ahmadzarkasi@unsri.ac.id
Received: 2025-08-16; **Accepted:** 2026-04-29

***Abstract*—This research introduces a prototype drug delivery robot that is controlled entirely through finger gestures, without the need for direct touch or additional physical control devices. The system utilizes a Pi Camera connected to a Raspberry Pi to capture the image of the user's hand, which is then processed by MediaPipe Hands to perform detection and extraction of 21 point *landmarks* in *real-time*. The position and relationship between *landmarks* are converted into a five-bit binary vector that represents which finger is raised. This binary data is then sent via serial communication to the STM32 Nucleo microcontroller, which is tasked with translating the binary pattern into motion commands (forward, backward, turn right/left, stop) to drive the DC motor. The results of testing the gesture recognition system, which was performed 30 times for each command, showed a high success rate. The gesture Stop achieved perfect success (30/30), followed by 'Forward' and 'Right' with 10 successes, and 'Backward' and 'Left' with 9 successes. This test shows that the system is able to respond accurately to gesture commands at a distance of 20 to 250 cm. The robot was also able to execute all motion commands responsively and accurately according to the recognized gestures. This prototype proves that MediaPipe can be an efficient and reliable method to implement gesture-based robot control on resource-constrained *embedded* platforms, as well as potentially applied in healthcare environments to minimize physical contact.**

Keywords—mediapipe, STM32, RaspberryPi, Robot, Hand Gesture

I. INTRODUCTION

The use of hand gestures as a nonverbal communication medium has become a primary focus in the development of human-computer interaction systems. Hand gestures are body movements that involve specific positions and movements of fingers or hands to convey intentions or commands. In the context of technology, hand gestures are widely applied to enhance interactive experiences in areas such as virtual reality, interactive gaming, assistive device control, and alternative communication systems such as sign language [1], [2], [3]. The advantages of hand gestures are that they are natural, intuitive, and do not require additional physical devices such as a mouse or remote, making it easier for users to interact with the system.

One of the most recent approaches widely used in gesture recognition is the utilization of RGB camera-based computer vision, as implemented in MediaPipe Hands. MediaPipe is an open-source framework from Google that enables real-time hand gesture detection and tracking with only RGB camera input without the need for additional sensors [4]. The workflow of MediaPipe consists of two main stages, namely palm detection and finger landmark tracking, which is capable of generating the coordinates of 21 key points of the hand in 2.5D form. This capability allows the system to accurately recognize the shape and position of the finger and can be implemented on both mobile and desktop devices [5].

Along with the increasing needs in automation and healthcare, drug delivery robots are being adopted to improve efficiency and reduce the risk of direct contact between patients and medical personnel. Drug delivery robots play an important role in medical logistics distribution, especially in pandemic conditions or inpatient isolation rooms, because they can reduce disease transmission through direct human interaction [5]. Various approaches in robot navigation have been developed, ranging from those based on line sensors, environmental mapping to artificial intelligence-based control. However, hand gesture-based control systems provide advantages in terms of ease and naturalness of interaction, especially in the context of human-robot interaction (HRI) [6].

Hand Gesture Recognition (HGR) is a highly active research area, with dozens of articles published annually focusing on the development of more natural and intuitive human-computer interaction (HCI) systems [7]. The goal is often to replace or complement traditional control devices such as the mouse and keyboard, as demonstrated in applications to control the mouse cursor [8] and PowerPoint presentations [9]. In many modern implementations, researchers utilize the Python programming language with

advanced *libraries* such as OpenCV for video processing and MediaPipe for *real-time* hand skeleton detection [8], [9]. MediaPipe, in particular, is popular for its ability to detect 21 three-dimensional (3D) hand landmarks, which is the foundation for gesture classification [8]. However, the application of HGR in the real world faces various challenges. One of them is detection accuracy in uncontrolled environments, where factors such as changing lighting conditions and complex backgrounds can affect system performance [10], [11]. In addition, although MediaPipe is highly reliable, its predictions sometimes generate biomechanically inaccurate 3D landmarks, which require further corrections to ensure the resulting gestures can be performed by humans [11]. Hardware considerations are also an important factor; implementation on an embedded platform such as the Raspberry Pi allows portability, but often requires longer processing times compared to a laptop or high-performance computer, which can affect overall system responsiveness.

The utilization of robots for delivery in a hospital environment has been proven to improve the efficiency of the drug use process and the timeliness of drug departure from the pharmacy, as well as significantly reduce drug waiting time [12]. To achieve this, modern delivery robots integrate various advanced technologies. Robot navigation systems often combine line follower technology with LiDAR scanning to effectively avoid both static and dynamic obstacles, ensuring safety in crowded environments such as hospital corridors [13], [14]. In addition, these robots are designed with advanced motion systems such as a *holonomic drive* that enables agile maneuvering in confined spaces, with a gyroscope and *rotary encoder* used for position control and error correction during navigation [15]. Frameworks such as the *Robot Operating System* (ROS) are then used to translate the recognized gestures into commands that can be executed by the robot, such as follow, stop, or return to a target location, which enables more fluid and natural human-robot interaction.

Various other approaches have also been developed for robot navigation and control systems. One of the artificial intelligence methods used is the Fuzzy Logic Controller (FLC), which allows robots to process distance and position information from sensors to avoid various moving and static obstacles autonomously [16]. For more structured environments such as hospitals, simpler yet effective navigation methods are also applied, for example, robots that follow lines (*line follower*) and use *Radio-frequency identification* (RFID) tags to identify specific locations such as patient rooms [17]. In terms of control, gesture-based interaction has evolved from early techniques that relied on skin color detection in perceptual color spaces such as HSV [18], to today's more sophisticated skeleton tracking methods. In addition to receiving commands, safe and efficient interaction in public spaces also requires the robot to be able to communicate its intentions to the humans around it. To this end, research on *External Human-Machine Interface* (eHMI) shows that the use of visual displays is preferred by pedestrians to understand the robot's navigation intentions, such as when it is about to turn or stop [19].

Hand gesture recognition generally follows two approaches: sensor-equipped gloves or computer vision with cameras [19], [20]. While gloves provide accurate data, they are costly, uncomfortable for long use, and restrict natural interaction. Vision-based methods are preferred for their natural, contactless interaction, though they face challenges like lighting, background complexity, and occlusion [20]. This research addresses these issues by developing a MediaPipe-based finger gesture control system for a drug delivery robot. The system enables real-time, touchless robot commands, aiming to improve hospital drug delivery efficiency and reduce disease transmission risks.

II. METHODOLOGY

This section outlines the methodology for developing the gesture-controlled drug delivery robot across three main phases: overall system architecture, hardware construction, and software design using MediaPipe. The complete workflow is also presented to illustrate how visual inputs are processed and translated into real-time motor commands.

A. System Design

This methodology is the initial stage in the research that will describe the outline of the system to be created. This description is illustrated in a flowchart that resembles a software design, but without in-depth technical details, and is more focused on explaining how the system will work as a whole. Fig. 1 illustrates the design of the system to be built.

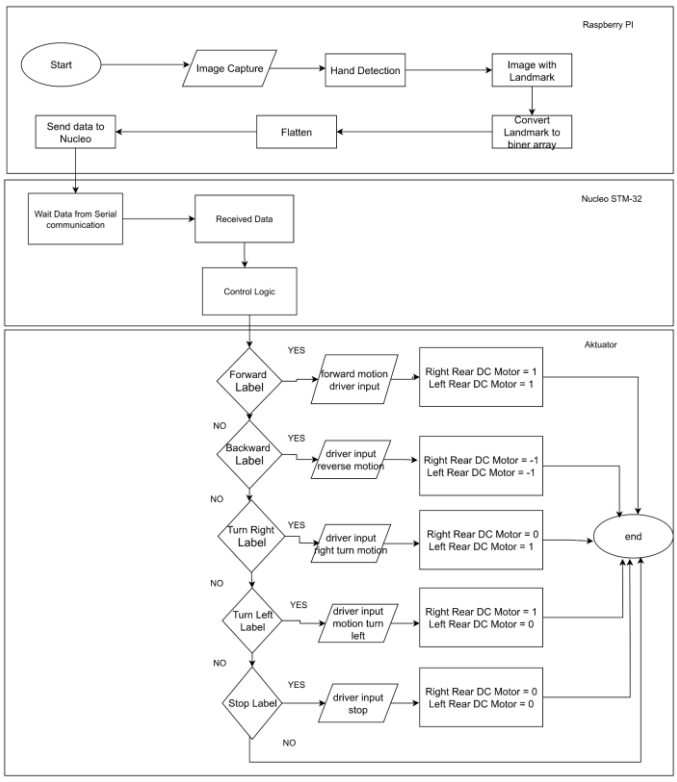


Fig. 1. System Design

The system utilizes the camera on the main processing unit in the form of a Raspberry Pi to capture images. Once captured, the image undergoes a preprocessing stage, where the RGB image format will be converted to Grayscale to simplify the visual information to be processed. The next stage is hand detection, where the system uses MediaPipe Hands, a machine learning-based framework from Google

that can real-time detect and track 21 landmark points on the hand with only RGB camera input.

The result of this detection is the coordinates of the finger landmarks that represent the shape and position of the user's hand. From these coordinates, the system will extract the finger gesture features needed for classification. Classification is done by matching the position of landmarks with predefined gesture patterns. Each recognized gesture will be converted into digital commands. This command data is then sent via serial communication to the microcontroller, namely Arduino Mega, which is in charge of executing commands according to the recognized gesture, such as activating the motor or moving the drug delivery robot in a certain direction. With this approach, user finger gestures can be translated into robot actions in real-time, without direct physical contact, making the system more natural and safe to use in healthcare scenarios.

B. Hardware Design

This stage details the construction of the drug delivery robot. The design starts with creating a robot chassis using Blender software. The design is made by considering the size, mounting points of electronic components, and stability when moving. After the design is complete, the robot frame is printed using a 3D printer with the help of Ultimaker Cura software as slicing software. The printing process produces parts of the frame, which are then assembled into a complete structure of the robot. The design drawing and the final result of the assembled robot are shown in Fig. 2.

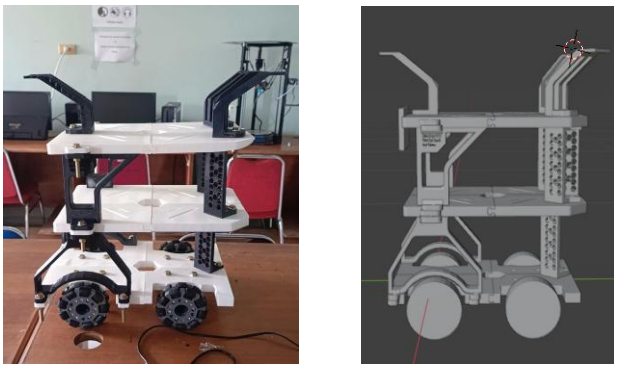


Fig. 2. Robot Frame

Once completed, various components are installed onto the frame to support system operations. These components can be seen in Table 1.

The Fig. above shows the overall system diagram consisting of several main components that are integrated with each other to support the functions of gesture recognition and robot movement control. Raspberry Pi acts as the main processing unit for the visual image, receiving input from the Pi camera and running the finger gesture detection and classification process using MediaPipe. After the gesture is recognized and converted into a digital command, the command is sent via serial communication to the STM32 microcontroller (Nucleo Board), which functions as an actuator controller.

TABLE 1. HARDWARE REQUIREMENTS

Component Name	Quantity
Raspberry Pi 3	1
Pi Camera	1
Nucleo STM32	1
DC motor	4
Jumper Cable	As necessary
BTS-7960 Driver Motor	4
Screws and bolts	As necessary

After the components are prepared, they will be assembled so that they can support the function from processing to movement. The description of the circuit in this study can be seen in the Fig. below

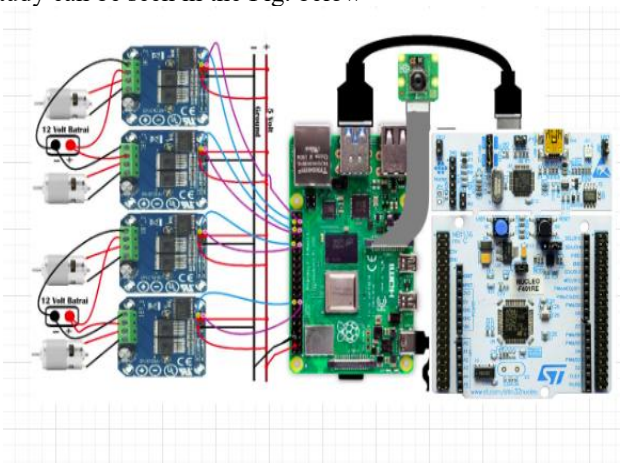


Fig. 3. Electronic Circuits

From the microcontroller, the signal is sent to a 4-channel relay module that functions as an electronic switch to control the current to each 12V DC motor. Each relay channel is configured to control the direction or activation of a specific motor attached to the robot wheel. Power flow to the motor is selectively controlled based on the command decoded from the gesture. With this configuration, the system allows the robot to move forward, backward, turn right, turn left, or stop according to the recognized finger gesture instructions in real-time.

C. Software Design

This stage involves designing the software responsible for processing visual input from finger gestures to produce movement instructions for the drug delivery robot. This software runs on a Raspberry Pi as a visual processing center and was developed using the Python programming language with the integration of the MediaPipe library for hand recognition. The processing flow can be seen in Fig. 4.

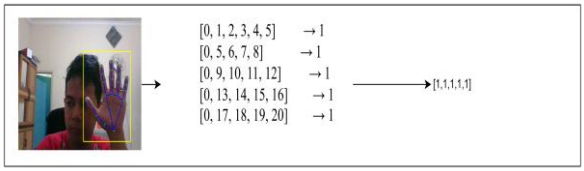


Fig. 4. Processing Scheme

Fig. 4 shows how the system's workflow in recognizing finger gestures starts with the input image from the camera. MediaPipe Hands performs hand detection and marks 21 landmark points on the hand visible in the image. These landmark points represent each finger segment and joint point from the thumb to the pinky. This process is done in real-time, so the gesture shown by the user can be processed by the system immediately. For example, the thumb is represented by index [0, 1, 2, 3, 4], the index finger by [0, 5, 6, 7, 8], and so on. The system then checks the status of each finger, whether it is open (1) or closed (0) by analyzing the angle or relative position between landmark points.

As a result of this process, the system will form a binary vector containing five values [x1, x2, x3, x4, x5], where each value represents the status of the thumb to the pinky. Fig. 4 also demonstrates an example of the gesture recognition output, illustrating the landmark points interconnected by blue lines to visually form a complete hand structure. Once the gesture is recognized, the system matches it with an instruction table to determine the robot's movement command. This table is statically defined and includes various gesture combinations that represent commands such as forward, backward, left turn, right turn, and stop. An example of a command table can be seen in Table 2.

On the Raspberry Pi side, the system was built using the Python programming language. The MediaPipe Hands framework is used to detect 21 finger landmarks from the camera image in real-time. After that, processing is performed to determine which finger is raised by comparing the vertical position (y-coordinate) of each finger segment against the previous segment. The result is a five-bit binary representation, each indicating the status of the finger (1 = raised, 0 = not raised) from thumb to pinky. There are 5 instructions in this study, described in the following Table 2.

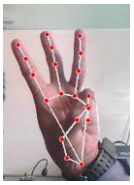
After the gesture is translated into commands, the data is sent via serial communication to the Arduino Mega. The Arduino then controls the L298N motor driver to set the direction and speed of the motor according to the received command. With this approach, the system is able to translate user finger gestures into robot control instructions intuitively, responsively, and without direct physical touch.

III. Result and Discussion

A. Gesture Recognition Testing

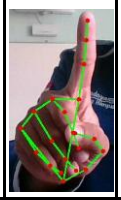
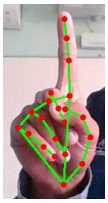
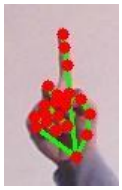
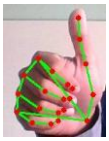
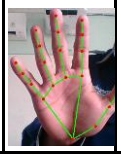
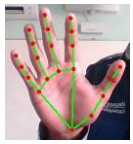
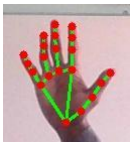
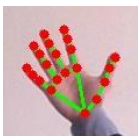
Tests were conducted 30 times for each gesture variation at a certain distance to evaluate the success rate of the system in accurately recognizing finger gestures. The recognition result table contains information about the type of gesture tested, the distance between the hand and the camera, and the commands generated by the system. In addition, Table 3 also records the number of successes and failures from 30 tests per gesture.

TABLE 2. Motion Command Instruction

Image	Binary Gesture	Command
	11111	STOP
	10000	FORWARD
	11110	Backward
	11100	Right
	11000	Left

The recognition results show that at the optimal distance (about 20-250 cm from the camera), the system is able to recognize gestures with high accuracy. The system performs robust recognition and adapts well to various distances. Therefore, distance variations have a minimal impact on recognition accuracy. The accuracy graph can be seen in Fig. 5. Based on the results of testing five types of gestures, an evaluation graph is shown in Fig. 5 to visualize the accuracy of the system's recognition of each gesture command. This graph represents the number of successes from 30 trials performed for each gesture, namely *Right*, *Left*, *Forward*, *Backward*, and *Stop*. The vertical axis shows the number of successful recognitions, while the horizontal axis shows the type of gesture command tested.

TABLE 3. GESTURE RECOGNITION TESTING

Gesture type	Distance (cm)						Success 30 trials)
	20	50	100	150	200	250	
Right							24
Left							23
Forward							23
Backward							26
Stop							28

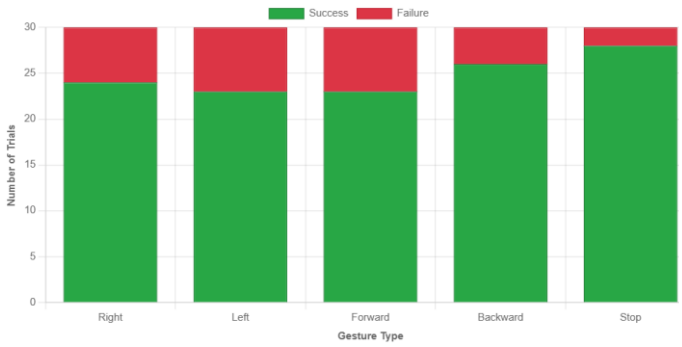


Fig. 5. Accuracy Chart

From the graph, it can be observed that the Stop gesture has the highest accuracy rate with a 100% success rate (30/30), indicating that this gesture is the easiest for the system to recognize, possibly due to its distinctive shape and lack of ambiguity. Gestures *Right*, *Left*, and *Forward* show fairly stable results with success between 23 and 26 times out of 30 trials, which still indicates good performance from the recognition system. Meanwhile, the *Backward* gesture showed slightly lower results, which could be due to the shape of the gesture that tends to be similar to other gestures or a lack of visual contrast in the camera's viewpoint.

Overall, this graph is the main indicator that the MediaPipe-based gesture recognition system developed is quite accurate and consistent in recognizing the main gestures used to control the robot. Assuming the ideal lighting conditions and distance during the test are maintained, the system can be said to be stable, responsive, and reliable for the implementation of robot control through real-time finger gestures. This evaluation also provides insight for further development in order to improve the accuracy of gestures that tend to be less consistently recognized.

B. Robot Movement

After the finger gesture is successfully recognized by the system through the detection and classification process using MediaPipe, the gesture is then converted into a binary vector that represents the status of each finger. This vector is then mapped against certain commands that have been defined in the movement instruction table. After the system recognizes the gesture and translates it into a digital command, the command is sent from the Raspberry Pi to the Arduino Mega via the UART serial communication line. The Arduino is then tasked with translating the command into a motor control signal, which is sent to the L298N motor driver to set

the direction and speed of movement of the DC motor on the robot.

TABLE 4. ROBOT MOVEMENT RESULT

Motion Command	Starting Position	End Position	Execution Time
Forward			Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762 Move: Forward, Execution Time : 2.0762
Backward			Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672 Move: Backward, Execution Time : 1.5672
Left			Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094 Move: Left, Execution Time : 1.0094
Right			Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788 Move: Right, Execution Time : 1.88788
Stop			Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097 Move: Stop, Execution Time : 2.0097

To evaluate the overall performance of the system, a series of tests was conducted on the response and accuracy of robot movements based on several types of finger gestures that represent certain instructions. The instructions tested in this study include five main types of commands, namely: forward, backward, turn right, turn left, and stop. The test is conducted by placing the robot at a certain starting point, then the user gives a finger gesture in front of the camera, and the system will process the gesture until the robot executes the movement according to the command.

The results of the test are presented in a structured manner in the form of an evaluation table. This table includes several important parameters, viz: the type of gesture instruction given, visual documentation in the form of the initial position of the robot (displayed through a photo before the command is executed), the final position of the robot after the command has been executed, and the execution time, which is the duration from when the gesture is recognized until the robot completes the movement and stops at the final position. This test is designed to measure the extent to which the system is able to produce movements in accordance with gesture instructions, as well as to determine the speed of system

response to the given visual input. The complete data of the robot movement test results are presented in Table 4. In general, the results show that the robot is able to respond well to commands and perform movements according to instructions in a relatively consistent and fast execution time. Visual documentation in the form of photos of the initial position and final position provides clear evidence that the system is able to translate finger gestures into precise robot actions.

V. CONCLUSION

The implementation of a MediaPipe-based control system for a drug-delivery robot has proven effective in recognizing hand gestures in real-time and translating them into navigational commands. By extracting 21 hand landmarks into binary vectors, the system identifies finger gestures with an accuracy exceeding 90% at an ideal operational range of 30-50 cm. Experimental results confirm the performance stability of the system under specific lighting and distance conditions, establishing a reliable decision-making framework for robotic control. Furthermore, the integration of Raspberry Pi and Arduino Mega via serial communication enables responsive control with consistent execution times across all navigation commands. This system successfully fulfills the research objectives by developing a gesture-based prototype that minimizes physical contact, offering significant potential for implementation in healthcare environments. Future work may focus on integrating additional navigation sensors and enhancing gesture classification accuracy through more sophisticated machine learning algorithms.

ACKNOWLEDGEMENTS

The first author would like to thank the Faculty of Computer Science, Strategic Pervasive Computing and Intelligent Embedded Systems Research Group (SPCIES-RG), and COMNETS RG, as well as Universitas Sriwijaya's Embedded Systems, Control System, and Robotic Laboratory.

REFERENCES

[1] J. Qi et al., “Computer vision-based hand gesture recognition for human-robot interaction: a review,” *Complex & Intelligent Systems*, vol. 10, pp. 1581–1606, 2024, doi: 10.1007/s40747-023-01173-6.

[2] A. Sharma et al., “Hand Gesture Recognition using Image Processing and Feature Extraction Techniques,” *Procedia Computer Science*, vol. 173, pp. 181–190, 2020, doi: 10.1016/j.procs.2020.06.022.

[3] A. O. Hashi et al., “A Systematic Review of Hand Gesture Recognition: An Update From 2018 to 2024,” *IEEE Access*, vol. 12, pp. 143599–143623, 2024 doi: https: 10.1109/ACCESS.2024.3421992.

[4] F. Zhang et al., “MediaPipe Hands: On-device Real-time Hand Tracking,” *Google Research*, 2020,doi : 10.48550/arXiv.2006.10214.

[5] S. Kavirayani et al., “Robot for Delivery of Medicines to Patients Using Artificial Intelligence in Health Care,” in *2021 IEEE Conference Publications,* doi: doi.org/10.1109/B-HTC50970.2020.9297948.

[6] V. Moysiadis et al., “An Integrated Real-Time Hand Gesture Recognition Framework for Human–Robot Interaction in Agriculture,” *Applied Sciences*, vol. 12, no. 8160, 2022,doi: 10.3390/app12168160.

[7] N. Mohamed, M. B. Mustafa, and N. Jomhari, "A Review of the Hand Gesture Recognition System: Current Progress and Future Directions," *IEEE Access*, vol. 9, pp. 157422–157437, 2021, doi: 10.1109/ACCESS.2021.3129650

[8] A. M. Chalik, B. A. Qowy, F. Hanafi, and A. Nuraminah, "Mouse Tracking Tangan dengan Klasifikasi Gestur Menggunakan OpenCV dan Mediapipe," *J. Ilm. Tek. Inform. dan Komun.*, vol. 1, no. 2, pp. 10–18, 2021

[9] D. Khairianto and R. Firdaus, "Penerapan Hand Gesture Recognition sebagai Media Kontrol Presentasi Aplikasi PowerPoint," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 2, pp. 1852–1860, 2024,doi: 10.36040/jati.v8i2.9167.

[10] H. Susilawati and F. Nuraeni, "Raspberry Based Hand Gesture Recognition Using Haar Cascade and Local Binary Pattern Histogram," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, vol. 8, no. 4, pp. 621–633, 2022, doi: 10.26555/jiteki.v8i4.24643

[11] Y. Zhang and G. Notni, "3D geometric features based real-time American sign language recognition using PointNet and MLP with MediaPipe hand skeleton detection," *Measurement: Sensors*, vol. 35, p. 101697, 2024, doi: 10.1016/j.measen.2024.101697

[12] M. R. Summerfield, F. J. Seagull, N. Vaidya, and Y. Xiao, "Use of pharmacy delivery robots in intensive care units," *Am J Health-Syst Pharm*, vol. 68, no. 1, pp. 77-83, 2011,doi:10.2146/ajhp100012.

[13] S. Senthilkumar, B. Dhanalakshmi, P. Shanmuga Prabha, A. Verma, E. Muniyandy, and S. Sridhar, "An IoT Intelligent Approach for Safety and Efficiency of Robotic Medicine Delivery in Hospitals," *J. Electrical Systems*, vol. 20, no. 2s, pp. 820-827, 2024,doi: 10.52783/jes.1677.

[14] H. Cao, X. Huang, J. Zhuang, J. Xu, and Z. Shao, "CIoT-Robot: Cloud and IoT Assisted Indoor Robot for Medicine Delivery," in *2018 Joint International Advanced Engineering and Technology Research Conference (JIAET 2018)*, Atlantis Press, 2018, pp. 85-89,doi: 10.2991/jiaet-18.2018.14.

[15] A. Bagade, A. Kulkarni, P. Nangare, P. Shinde, and S. Gudadhe, "Robotic Assistant for Medicine and Food Delivery in Healthcare," in *Cyber Security, Privacy and Networking*, D. P. Agrawal, N. Nedjah, B. B. Gupta, and G. M. Perez, Eds. Singapore: Springer, 2022, pp. 49-59,doi: 10.1007/978-981-16-8664-1_5.

[16] F. Umam, "Pengembangan Sistem Kendali Pergerakan Autonomous Mobile Robot untuk Mendapatkan Jalur Bebas Hambatan Menggunakan Fuzzy Logic Controller," *Jurnal Ilmiah Mikrotek*, vol. 1, no. 1, pp. 35-42, 2013

[17] A. Joy et al., "MedRobo: Medicine Delivering and Patient Parameter Monitoring Robot," in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, 2021, pp. 1808-1812,doi: 10.1109/ICACCS51430.2021.9441692.

[18] M. Manigandan and I. Manju Jackin, "Wireless Vision based Mobile Robot control using Hand Gesture Recognition through Perceptual Color Space," in *2010 International Conference on Advances in Computer Engineering*, Bangalore, India, 2010, pp. 95-99,doi: 10.1109/ACE.2010.69.

[19] S. S. Kannan, A. Lee, and B.-C. Min, "External Human-Machine Interface on Delivery Robots: Expression of Navigation Intent of the Robot," *arXiv preprint arXiv:2108.03045*, 2021,doi: https://doi.org/10.48550/arXiv.2108.03045

[20] M. Oudah, A. Al-Naji, and J. Chahl, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques," *J. Imaging*, vol. 6, no. 8, p. 73, 2020,doi: 10.3390/jimaging6080073.