Wireless Human-Robot Interaction: Replicating Hand Movements Using OpenCV and Embedded Systems

G. Manoj[1], Y. Tanmai Sabari[1], V. Brunda[1], P. Kalyan Babu[1], Y. Madhu Babu[1], and M. Vamshi Krishna[1]

[1] Department of ECE, Dhanekula Institute of Engineering and Technology, Vijayawada, India

gudisevabrothers@gmail.com, profvamshi@gmail.com

Abstract. The gesture-controlled robotic hand is designed for natural Human-Robot Interaction (HRI) using real-time hand gesture recognition. This aims to provide an intuitive, touchless interface using finger tracking. The proposed system integrates computer vision and embedded systems. Real-time hand gestures are captured using a webcam or camera feed and processed via OpenCV and MediaPipe. They are transmitted through an ESP8266 microcontroller. The robotic hand, built with five MG995 servo motors, mimics finger movements. Two operational modes, live and replay, are implemented. Experiments showed an average accuracy of 95%, with the thumb slightly lower (91%). Response times were typically 120–180 ms, with slight delays under poor lighting or weak signal strength. Replay mode reliably reproduced stored gestures with 95.2% accuracy. This developed system effectively shows contactless robotic control using computer vision and embedded systems, which offers a flexible platform for assistive technologies and teleoperation.

Keywords: Human-Robot Interaction, Real-time Gesture Replication, OpenCV, 3D-Printed Robotic Hand, Python, ESP8266, Embedded Systems, Wireless Communication.

1 Introduction

Human-robot interaction (HRI) has obtained importance in recent years, particularly in the fields of assistive technology, automation, and teleoperation [1]. The natural way to achieve robot control without any physical contact is provided by hand gesture recognition i.e. finger tracking and control [2].

In many existing methods, hand gesture recognition, mainly through finger tracking, provides a touchless and natural method for controlling robotic systems. Traditional robotic control techniques are often based on hardware-based controllers and predefined command sets, which limit flexibility and user engagement [3–5]. However, advancements in embedded systems and computer vision make development of gesture-based virtual control systems easier, improving both convenience and operational efficiency [6].

Despite these advancements, existing gesture recognition systems continue to face challenges related to accuracy, latency, and real-time responsiveness [7–10]. To overcome these limitations, OpenCV and MediaPipe are incorporated for real-time hand landmark detection and joint angle calculation [11]. The joint angles which are calculated, will then be mapped into servo motor positions, making way for a 3D-printed robotic hand to replicate human gestures with high accuracy [12].

The architecture includes an ESP8266 microcontroller that works as a wireless communication mediator and transmits commands from the gesture recognition module to the robotic hand [13]. The system supports two operational modes: Live Mode, where gestures are recognized and mimicked in real time and Replay Mode, where pre-recorded gestures are retrieved from a CSV file and executed sequentially [14]. To ensure smooth servo motor transitions, a filtering mechanism is involved, improving gesture execution properly [15].

Experimental results show high recognition accuracy under various lighting conditions [15], low latency during live mode and valid gesture reproduction in replay mode. This contributes to the development of gesture-driven robotic systems, with promising applications in prosthetics, remote robotic control, and human assistive technologies.

M. Sharma et al. (eds.), *Proceedings of the Third International Conference on Recent Advances in Computing Sciences (RACS 2025)*, Advances in Intelligent Systems Research 212,
https://doi.org/10.2991/978-94-6239-768-2_22

2 Literature Review

Wireless human-robot interaction, mainly finger detection, has seen great advancements in recent years because of computer vision and embedded systems. Many implementations have been introduced to recognize hand gestures and convert them into instructions for controlling devices.

Kavitha et al. [1] created a virtual mouse controlled by hand gestures using AI techniques, working in vision-based human-computer interaction. Yoon et al. [2] concentrated on using location, angle, and velocity for improving gesture recognition, while Mitra and Acharya [3] provided information on gesture recognition methods, focusing on vision-based and sensor-based methods. Depth sensors and real-time pose evaluation have been surveyed by Keskin et al. [4], giving high accuracy in hand tracking. Erol et al. [5] evaluated multiple vision-based hand movement estimation techniques, indicating the challenges in blocking and finger segmentation. Wu and Huang [6] earlier discussed the growth of gesture-based communication using traditional vision techniques.

On the hardware side, Madgwicket al. [7] introduced an aim estimation method using IMU data, which has been key for embedded gesture systems. Sensor fusion techniques were further refined by Lee and Choi [8] for robust gesture recognition. For robotic control, Billard et al. [9] reviewed learning from demonstration techniques, enabling robots to mimic human actions. Kim et al. [10] showed effective gesture-to-robot command mappings using depth data. Furthermore, wearable interfaces like FlyJacket [12] and body–machine interfaces [11] advanced immersive control of aerial systems. Modern control and learning techniques were explored in works like Kim et al. [13] and Khansari-Zadeh and Billard [14], focusing on predictive and adaptive systems. Medina and Billard [15] introduced a task learning model for robotic behavior. These studies collectively form a strong foundation for developing wireless human-robot interaction systems that replicate human hand gestures using computer vision and embedded systems.

3 Methodology

The Methodology section provides an overview of the work, explains how the dataset was built, and discusses the technologies used.

3.1 Overview of General Work

This system enables human-robot interaction using real-time hand gestures. To enable real-time gestures, the proposed methodology combines computer vision and embedded systems [4].

The system captures real-time hand gestures using a webcam, then processes the movements using OpenCV and MediaPipe, and transfers servo control commands to an ESP8266 microcontroller via Wi-Fi. When the ESP8266 acts as a web server, it receives HTTP requests containing servo motor angles and executes the corresponding hand movements. The robotic hand consists of five MG995 servo motors, each controlling an individual finger. The system works in two operational modes: Live and Replay. The overall architecture of the proposed wireless robotic hand system is illustrated in **Fig. 1**

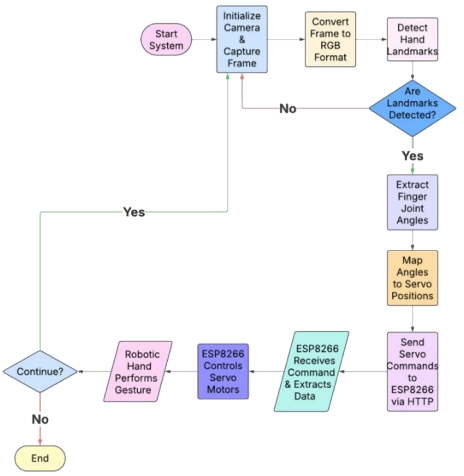

Fig. 1. General overview of work.

3.2 Building the Dataset

The system depends on MediaPipe's hand tracking framework which detects 21 key landmarks on the human hand. From these landmarks, joint angles are calculated using vector-based calculations [12]. The dataset is dynamically generated by recording these joint angles and then mapped to servo motor angles. For accurate gesture recognition, the system uses an angle calculation method and storage formats [11]. In the angle calculation method, the cosine rule is applied to compute angles between finger joints. In the storage format, the recorded gestures are stored in CSV files with each row representing a hand posture as a set of servo angles. The structure of the recorded gesture dataset stored in CSV format is presented in **Table 1**.

Table 1. CSV File Structure.

Index	Ring	Middle	Thumb	Pinky
0	90	0	1.118707	90
0.5777	0	0	8.009329	0.901654
1.177326	0	0	11.84308	1.112028
1.056946	0	0	4.996501	0.562679
1.084382	0	0	6.444495	0
1.901924	0	0	10.7976	1.221425

These stored values allow the replay mode to read data from CSV files and send the corresponding servo angles to the ESP8266.

3.3 Computer Vision Gesture Recognition

The system uses OpenCV and MediaPipe for real-time finger tracking. The webcam records video frames which are processed in RGB format for hand tracking [6]. MediaPipe detects 21 hand landmarks including the thumb, index, middle, ring, and pinky fingers. The 21 hand landmarks detected using MediaPipe are shown in **Fig. 2**.

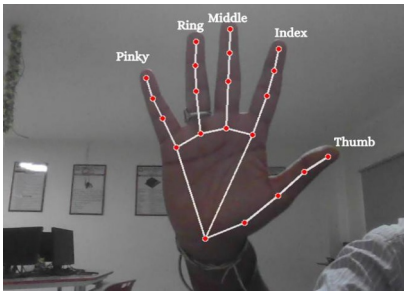

Fig. 2. Hand Landmark Detection with labels.

Vector formation for joint angles is calculated using three landmarks for each finger joint angle: Joint 1 (base), Joint 2 (middle), and Joint 3 (tip of the finger). These are calculated by: Vector 1 = Base to Middle; Vector 2 = Middle to Tip. The landmarks are then mapped to a 3D coordinate space (x, y, z).

For finger joint angle calculation, each finger's bending angle is calculated using vector mathematics:

$$\cos\theta = (V_1 \cdot V_2) / (|V_1||V_2|)$$

For mapping angles to servo positions, the detected finger angles are remapped from a technical range (5°–140°) to servo-compatible values (0°–180°). The gesture-to-servo mapping is:

```
Index = remap(joint_angles['index'], 5, 140, 0, 90)
ring = remap(joint_angles['ring'], 5, 130, 0, 90)
```

3.4 Wireless Communication & Data Handling

To transmit servo motor commands from the gesture recognition module to the robotic hand, the system uses Wi-Fi based communication. When the microcontroller starts acting as a web server, it receives servo position data through HTTP GET requests [8]. When hand gestures are detected, their finger angles are calculated, mapped to servo positions, and converted into commands [13]. These commands are sent over Wi-Fi to the microcontroller specifying the servo index and angle. The robotic hand performs the actions by detecting gestures. This system also provides a feedback mechanism to confirm successful execution or detect communication failures [15]. This wireless control mechanism eliminates the need for direct wiring, enhancing flexibility.

Experiments

This section provides a detailed experimental setup, experimental procedure, and testing conditions. The experimental setup consists of hardware components and software frameworks.

Hardware Components

The hardware components used are: (1) ESP8266 NodeMCU Microcontroller for handling servo commands over Wi-Fi; (2) MG995 Servo Motors (×5) for controlling individual fingers of the robotic hand; (3) 3D-Printed Robotic Hand for performing actions according to detected gestures; (4) Webcam or device camera for capturing real-time hand movements; and (5) Power Supply for Servos, providing stable voltage for motor operation.

Software Frameworks

OpenCV and MediaPipe are used to detect hand landmarks and extract joint angles. Python handles gesture processing and Wi-Fi communication, transmitting servo angles to the ESP8266. The Arduino IDE (ESP8266 Programming) receives HTTP requests and controls servos. CSV files store recorded gestures for playback mode.

Experimental Procedure

The system was tested in two operational modes. The user interface for selecting the operational mode (Live/Replay) is shown in **Fig. 3**.

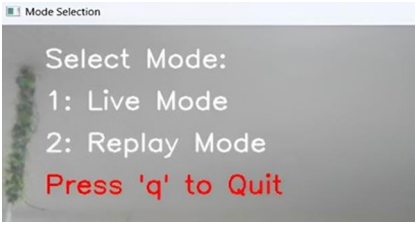

Fig. 3. Selecting Mode.

Real-Time Gesture Control (Live Mode). When the user places a hand in front of the camera, OpenCV captures live frames and MediaPipe detects 21 hand landmarks and extracts finger joint angles. These angles are then mapped to servo positions using a remap function, and sent to ESP8266 over Wi-Fi through HTTP requests. The robotic hand moves instantly according to the detected gestures [13]. The complete experimental hardware setup is presented in ***Fig. 4***.

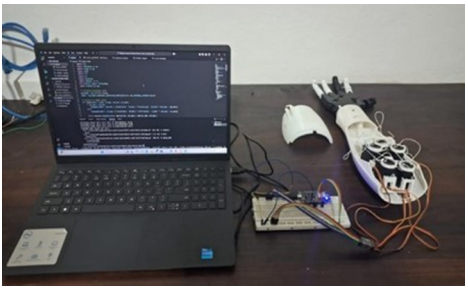

Fig. 4. Whole setup showing integration of hardware components.

Pre-Recorded Gesture Playback (Replay Mode). When the user selects a pre-recorded gesture file containing stored servo positions, the file is read line by line, where each row contains servo angles for one frame. These angles are sent to ESP8266 over Wi-Fi, and the robotic hand recreates the movement. A delay between frames ensures smooth execution. The sequence of replay-mode execution is illustrated in **Fig. 5**. The experimental testing conditions used for system evaluation are summarized in ***Table 2***.

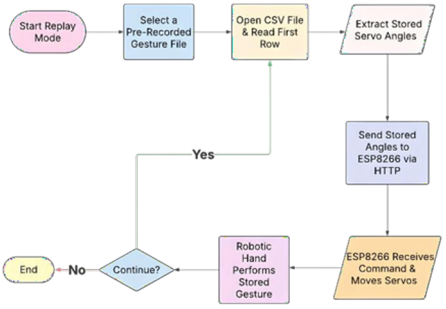

Fig. 5. Replay mode operation.

Table 2. Testing Conditions.

Test Parameter	Test Conditions	Observations
Lighting	Bright, Dim, Dark	Low accuracy in dark condition
Gesture speed	Slow, Medium, Fast	Fast gestures reduced accuracy

Test Parameter	Test Conditions	Observations
Camera distance	30 cm, 50 cm, 100 cm	Greater distance caused tracking delays
Wi-Fi latency	Strong vs. Weak signal	Weak signal increased response time

Table 3. Accuracy under lighting conditions.

Lighting Condition	Approx. Lux Value	Accuracy (%)
Bright Light	~500 lux	96.2
Dim Light	100–200 lux	93.5
Low Light	<100 lux	89.7

The test conditions above are based on lighting conditions, gesture speed, camera distance, and Wi-Fi latency parameters to check the total accuracy performance of the system. In low light and during fast gestures, the accuracy has dropped. Greater camera distance causes more delays, and latency in Wi-Fi signals causes weak response time [11]. The effect of different lighting conditions on recognition accuracy is presented in **Table 3**

Results and Discussion

The results of the gesture-controlled robotic hand focused on recognition accuracy, response time, and stability of execution. The system was tested in different lighting environments, gesture speeds, and distances from the camera. The finger-wise gesture recognition accuracy is illustrated in **Fig. 6**.

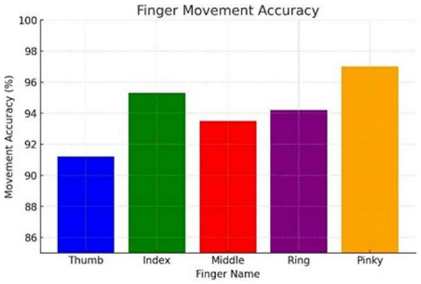

Fig. 6. Finger movement accuracy calculation graph.

In this graph, the thumb finger showed slightly lower accuracy (91.3%) because of its unique rotation, which may not be captured by MediaPipe's standard linear finger movement detection, compared to other fingers showing an average 95% accuracy. The response time of the proposed system under different operating conditions is shown in **Fig. 7**.

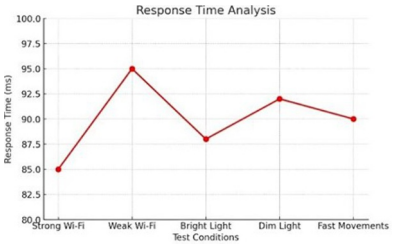

Fig. 7. Response time analysis graph.

In this response time analysis graph, the system had a fast response of 85 ms in strong Wi-Fi conditions and a delay of 95 ms during weak Wi-Fi and dim light conditions. The replay mode performance results are presented in **Fig. 8**.

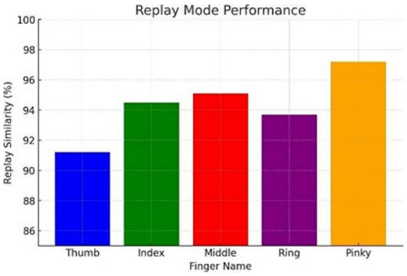

Fig. 8. Replay mode performance graph.

The Replay Mode Performance graph clearly shows that the system achieved high accuracy in detection of finger movements (95.2%), though the thumb finger shows lower accuracy (91.5%).

Conclusion

This system presents a gesture-controlled 3D-printed robotic hand that utilizes computer vision (OpenCV&MediaPipe), wireless communication (ESP8266), and servo motor control to achieve real-time and replay-based movement execution. The system achieved an accuracy of 95% under standard conditions. The response time is 120–180 ms for most gestures and 250 ms for rapid gestures. The replay mode successfully replicated stored gestures with 95.2% accuracy. Lighting conditions and signal strength affected the system's performance.

Future advancements include improvement of accuracy under low-light conditions by applying adaptive brightness correction. Reducing network latency in Wi-Fi communication can improve response time. Expansion of the system can enable recognition of more hand gestures. Integration of AI-based learning models can improve gesture classification. This system demonstrates the growth in gesture-based robotic control across various applications, proving to be a low-cost and vision-based robotic hand system capable of reliable and accurate performance for real-world human-robot interaction.

Acknowledgments.

The authors would like to thank Dhanekula Institute of Engineering and Technology for providing the resources and facilities required for this research.

Disclosure of Interests.

The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Kavitha R., Janasruthi S.U., Lokitha S., Tharani G.: Hand Gesture Controlled Virtual Mouse Using Artificial Intelligence. IEEE Xplore, Vol. 9, Issue 2 (2023).
2. Yoon, M.H., Lee, S.H., Kim, J.H.: Hand Gesture Recognition Using Combined Features of Location, Angle and Velocity. IEEE Transactions on Industrial Informatics, 15(2), 1249–1257 (2019).
3. Mitra, S., Acharya, T.: Gesture Recognition: A Survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 37(3), 311–324 (2007).
4. Keskin, C., Kırac, F., Kara, Y.E., Akarun, L.: Real Time Hand Pose Estimation Using Depth Sensors. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(8), 1614–1627 (2014).

5. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X.: Vision-Based Hand Pose Estimation: A Review. Computer Vision and Image Understanding, 108(1–2), 52–73 (2007).

6. Wu, Y., Huang, T.S.: Vision-Based Gesture Recognition: A Review. In: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, pp. 103–115 (1999).

7. Madgwick, S.O., Harrison, A.J., Vaidyanathan, R.: Estimation of IMU and MARG Orientation Using a Gradient Descent Algorithm. In: 2011 IEEE International Conference on Rehabilitation Robotics, pp. 1–7 (2011).

8. Lee, C.S., Choi, Y.S.: A Sensor Fusion Method for User Hand Gesture Recognition. IEEE Transactions on Consumer Electronics, 56(3), 1434–1442 (2010).

9. Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot Programming by Demonstration. In: Springer Handbook of Robotics, pp. 1371–1394. Springer (2008).

10. Kim, D., Nguyen, T.H., Pham, A.T., Tran, D.H.: Hand Gesture Recognition Using Depth Data for Robot Control. In: 21st International Conference on System Theory, Control and Computing (ICSTCC), pp. 679–684 (2017).

11. Miehlbradt, J., et al.: Data-Driven Body–Machine Interface for the Accurate Control of Drones. Proceedings of the National Academy of Sciences, 115(31), 7913–7918 (2018).

12. Rognon, C., et al.: FlyJacket: An Upper Body Soft Exoskeleton for Immersive Drone Control. IEEE Robotics and Automation Letters, 3(3), 2362–2369 (2018).

13. Kim, S., Shukla, A., Billard, A.: Catching Objects in Flight. IEEE Transactions on Robotics, 30(5), 1049–1065 (2014).

14. Khansari-Zadeh, S.M., Billard, A.: Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models. IEEE Transactions on Robotics, 27(5), 943–957 (2011).

15. Medina, J.R., Billard, A.: Learning Stable Task Sequences from Demonstration with Linear Parameter Varying Systems and Hidden Markov Models. In: Conference on Robot Learning, pp. 165–174 (2017).