

Hand Gesture-Based Mouse Control System Using OpenCV and Media Pipe for Real-Time Interaction

VEENA BHAT¹, PANCHAMI B S²

¹Assistant Professor, AI & DS, Department, Shri Dharmasthala Manjunatheshwara Institute of Technology, Karnataka

²BE Scholar, CSE, Department, Shri Dharmasthala Manjunatheshwara Institute of Technology, Karnataka

Abstract- *The study presents a gesture-based interaction system designed for real-time mouse control using OpenCV and Mediapipe. This system uses hand gestures to provide an intuitive and contactless way to interact with computers, significantly enhancing user experience and accessibility for persons who struggle with conventional input devices like a mouse or keyboard. Utilizing a single webcam, the system continuously captures and monitors hand movements. These movements are handled through pattern recognition algorithms to accurately recognize specific gestures, with each recognized gesture corresponding to various computer actions, including mouse movement, clicks, and scrolling. The system is engineered for user-friendliness and efficiency, allowing users to navigate their computer screens effortlessly without physical contact. Results from the study highlight the practicality and effectiveness of employing hand gestures for basic computer control tasks, presenting a promising approach for hands-free interaction in both everyday and specialized computing scenarios.*

Indexed Terms- *Gesture Recognition, OpenCV, Mediapipe, Mouse Control, Human-Computer Interaction.*

I. INTRODUCTION

Human-computer interaction (HCI) is a field that studies the methods complete which humans engage with computers and seeks to improve the interface between users and machines. Traditionally, this interaction has depended significantly on devices like the keyboard and mouse. Though these tools have been the standard for decades, technological advancements have covered for more automatic and natural interaction methods. One such approach is the

implementation of natural user interfaces (NUIs), which aim to make interactions with technology more seamless and aligned with human behaviour.

Gesture recognition, a key component of NUIs, allows users to control devices through physical movements rather than traditional input methods. This type of interaction is not only more natural and also enhances accuracy for users with physical limitations who may find conventional input devices difficult to use. By interpreting hand gestures, computers can understand and respond to commands without the need for physical contact laying the groundwork for a more seamless and natural user experience.

The system explores the development of a gesture-based interaction system that enables real-time mouse control. The system is built using OpenCV, a powerful computer vision library, and Mediapipe, a framework designed for building multimodal machine learning applications. The integration of these tools enables the system to effectively capture and process hand gestures using only a standard webcam. To improve the accessibility of computing systems by allowing users to perform the tasks such as controlling the cursor, left click, right click, double click and other operations through simple hand gestures.

This paper explores the technical aspects of the system's implementation, focusing on the gesture recognition process, the mapping of gestures to specific computer actions, and the system's overall effectiveness in real-world applications. Our results prove that gesture recognition can serve as an effective solution for fundamental computer control tasks, providing enhancements in both convenience and accessibility for users.

II. LITERATURE REVIEW

Gesture recognition has evolved significantly over the past decade, driven by advancements in computer vision, machine learning, and sensor technologies. Early systems like Microsoft's Kinect and Leap Motion controllers paved the way for 3D gesture capture, enabling applications in gaming and virtual reality. Though, these systems often needed special hardware and had issues with accuracy and reliability. Recent developments in computer vision frameworks, such as Mediapipe, have democratized gesture recognition by using deep learning models to interpret 2D hand movements from standard webcam feeds. Mediapipe's hand tracking solution offers real-time performance and high accuracy, making it suitable for a general range of applications from HCI to interactive art installations.

The paper "Control Mouse using Hand Gesture and Voice" presents a system that integrates hand gesture recognition with voice commands for controlling the computer mouse. It utilizes real-time camera input and apply machine learning algorithms, mainly deep learning methods such as Convolution Neural Networks (CNN), implemented through MediaPipe and other tools. The system aims to enhance user comfort and reduce computational time, with potential applications in reducing virus transmission risk, controlling robots, and assisting individuals with hand disabilities. The authors highlight the use of RGB-D images combined with fingertip detection, employing methods such as colored caps and hand gesture detection. Key algorithms mentioned include OpenCV, Autopy, and MediaPipe [1]. "Implementation of Virtual Mouse Control System Using Hand Gestures for Web Service Discovery" designed for discovering web services like SOAP, UDDI, and WSDL. The system uses computer vision to allow gesture-based interaction, with hand movements recorded by a camera and processed using color segmentation and detection techniques. The authors review various gesture recognition methods, including approaches by Varun Sharma, Aashni Hariaa, and Tomasz Grzejszczak. The system design incorporates MediaPipe, OpenCV, and Autopy for data processing and gesture interpretation [2].

"Virtual Mouse Using Hand Gesture" describes two

approaches: one using color caps and another using hand gesture detection. The methodology involves finger detection, hand gesture tracking, and implementing tracking results on an on-screen cursor. The system uses hand contour detection and convex hull creation for gesture tracking. The paper also discusses the integration of PyAutoGUI for additional functionalities like left and right-click actions [3].

"Hands-Free Computing Gesture-Based Virtual Mouse and Keyboard" explores the use of computer vision to produce a simulated optical mouse and keyboard using hand gestures. The system captures different hand gestures through a webcam, mapping them to cursor actions and keyboard functions [4].

The paper "Virtual Mouse and Keyboard" uses computer vision techniques like convex hull defect detection to recognize different hand motions and map them to mouse and keyboard functions [5]. "Gesture Controlled Virtual Mouse with Voice Assistant" presents a system that integrates hand gesture control with voice commands. It utilizes OpenCV and MediaPipe for gesture recognition and includes AI and natural language processing for voice recognition [6].

"Deep Learning Based Real Time AI Virtual Mouse System Using Computer Vision" proposes a computer-generated mouse system that utilizes natural hand motions and deep learning algorithms. The paper discusses the computer vision techniques, image processing, and machine learning for accurate hand gesture recognition [7].

Gesture Recognition Techniques

Recent literature discusses various techniques for gesture recognition, including:

- Depth-Based Approaches: Using depth sensors like Kinect to capture 3D hand movements.
- Vision-Based Approaches: Utilizing cameras and Computer vision techniques for 2D hand tracking.
- Automated Learning in Gesture Recognition: Applying neural networks and deep learning models to improve accuracy and robustness.

Applications of Gesture Recognition

Gesture recognition technology is utilized in a wide

range of industries:

Healthcare: Universal design technologies.

Education: Engaging educational settings and online classrooms.

Entertainment: Gaming, virtual reality, and augmented reality experiences.

III. METHODOLOGY

The system utilizes a webcam for capturing video input. OpenCV handles real-time image processing, while Mediapipe is employed for hand gesture detection. Detected hand gestures are converted into actions such as mouse movement, clicking, scrolling, and toggling application visibility.

Hand Gesture Detection

Mediapipe's hand solutions detect and track hand landmarks in real-time, with high confidence and precise movement tracking for a single hand.

Gesture Mapping

Hand gestures are mapped to specific actions:

- **Mouse Movement:** The position of the index finger tip controls cursor position on the screen.
- **Clicking:** Bending different combinations of fingers triggers left or right mouse clicks. A fist gesture can initiate double clicks.
- **Scrolling:** The vertical position of index finger tip is used to scroll up or down.
- **Application Visibility:** A specific gesture involving all fingers bent toggles application visibility using Alt + Tab.

Mapping Hand Gestures to Mouse Actions

Actions like mouse pointer movements, clicks, and gestures are mapped based on the location and bending of hand key points are detected using Mediapipe's hand-tracking module. These joints are detected in real-time through a webcam and are processed to calculate angles between joints or distances between fingers, helping in determining specific gestures.

For example, the angle between joints of the index finger is calculated using the function ``get_angle``, and if angle is below a certain threshold (for example 75 degrees), the finger is considered bent. The mouse movement is controlled by tracking the index finger

tip's x and y coordinates scaled to match the screen resolution, and smoothing is applied for a smooth cursor movement.

Different gestures like scrolling and clicking are mapped by identifying the combinations of bent or extended fingers. For example, when all fingers are bent, a fist is detected, and actions such as scrolling are triggered based on the vertical position of the hand. PyAutoGUI and Pynput libraries are used to simulate these actions and allowing users to control various functions through hand gestures.

Implementation

The implementation consists of multiple components:
Hand Detection & Tracking: Utilizing Mediapipe's hand model to identify hand landmarks in video frames.

Gesture Recognition: Recognizing specific gestures (such as a fist or bent fingers) based on the positions of the hand key points.

Action Mapping: Mapping the known gestures to corresponding actions (e.g., mouse movement, clicks, scrolling).

Algorithm 1: Hand Detection and Tracking

Step 1: Initialize the screen size and hand detection model.

Step 2: Define a function to control the mouse cursor movement based on the smoothed position of the index finger tip.

Step 3: Define a function to check if a specified finger is bent using key point positions.

Step 4: Define a function to detect gestures and perform actions such as mouse cursor movements or clicks.

Step 5: Define the main function to continuously capture and process video frames, detect gestures, and execute corresponding actions.

Algorithm 2: Gesture Recognition and Action Mapping

Step 1: Define the function `get_angle (p1, p2, p3)`:

Compute the angle between three points using the `atan2` function.

Step 2: Define the function `distance (p1, p2)`:

Compute the Euclidean distance between two points.

Experimental Setup

Experiments were conducted to evaluate the system's performance in different environments and lighting conditions. The setup included a standard webcam connected to a laptop running the gesture recognition software.

Environment

Experiments were conducted in both indoor and outdoor settings to assess system robustness under different lighting conditions.

Metrics

The evaluation of the system's performance relied on the following metrics:

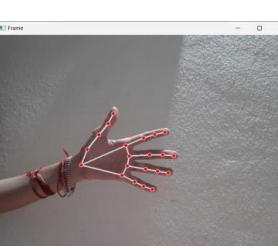
- Accuracy: The rate of accurately identified gestures.
- Latency: The time taken to process and respond to a gesture.
- Usability: User feedback on the ease of use and perception of the system.

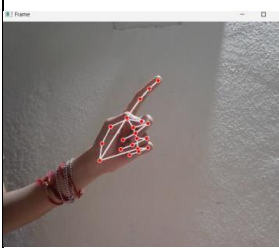
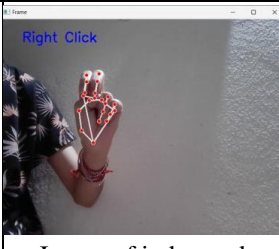
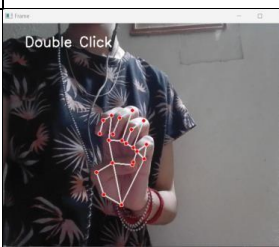

IV. RESULTS AND DISCUSSION

The system executed real-time hand gesture recognition and executed corresponding actions with an intuitive and contactless method for basic computer control tasks.

In Table 1 different components of the system are tested and the output is compared with expected behavior.

Table 1: Unit test

Test case No	Input	Expected behavior/Output	Status P=Pass F=Fail
1	 Image of hand	Detects the hand and identifies finger positions.	P

2	 Image of index finger tip	The cursor moves with the position of the index finger tip.	P
3	 Image of index and middle fingers bent	Right click is performed.	P
4	 Image of index, middle, and ring fingers bent	Double click is performed.	P
5	 Image of index finger bent	Left click is performed.	P

Accuracy

The system accurately recognized hand gestures about 90% accuracy rate in well-lit environments. Accuracy was slightly reduced in low-light environments but remained above 80%. Accuracy was evaluated in real-time during live user interactions. The system's gestures were continuously compared against expected outcomes to determine recognition accuracy.

Latency

The average latency for recognizing gestures and executing actions was 100 milliseconds, providing a

smooth and real-time experience. Latency was measured live during user interactions, recording the time from gesture detection to action execution to ensure timely responses.

Usability

The system was found to be highly usable, with users expressing satisfaction regarding the intuitive nature and effectiveness of the gesture-based controls. Usability was assessed through live user trials, and feedback was gathered to quantify the ease of use and naturalness of the gesture-based interactions. **Lighting Conditions:** Variability in lighting conditions posed a challenge for accurate gesture recognition. Future work could focus on improving robustness under diverse lighting conditions.

Background Clutter: Complex backgrounds sometimes interfered with hand detection. Implementing background subtraction techniques could enhance the system's accuracy.

Improvements

Algorithm Optimization: Further optimization of the gesture recognition algorithms could reduce latency and improve real-time performance.

Extended Gesture Set: Expanding the set of recognized gestures could enable more complex interactions and functionalities.

CONCLUSION

The paper presents a practical and efficient approach to gesture-based interaction using OpenCV and Mediapipe. The system provides an affordable solution that improves user experience and accessibility using simple hand gestures. The results demonstrate that this gesture-based system can be effectively integrated into everyday computing tasks, providing users with an alternative and innovative method of interacting with digital devices. The implementation of the system demonstrates its potential to become a valuable tool in both personal and professional computing environments, offering an accessible and user-friendly interface for a wide choice of users.

Future Work

The current system provides a strong base for gesture recognition. However, several enhancements could further improve its functionality. Future improvements could include:

- **Multi-Hand Support:** Extending the system to recognize and respond to gestures from both hands simultaneously.
- **Enhanced Gesture Set:** Expanding the range of recognized gestures to include more complex commands.
- **User Customization:** Allowing users to define and customize gestures based on personal preferences.
- **Machine Learning Integration:** Incorporating machine learning algorithms to improve gesture recognition accuracy and adaptability.
- **Robustness to Environmental Changes:** Enhancing the system's performance under varying lighting conditions and backgrounds.

REFERENCES

- [1] J. Nandwalkar, M. Mandal, A. Khirari, and T. Bhalchim, "Control Mouse using Hand Gesture and Voice," Faculty of Computer Department, Datta Meghe College of Engineering, Airoli, Navi Mumbai, International Journal of Intelligent Systems and Applications in Engineering, ISSN: 2147- 6799.
- [2] G. V. Bhole, S. Deshmukh, M. D. Gayakwad, and P. R. Devale, "Implementation of Virtual Mouse Control System Using Hand Gestures for Web Service Discovery," International Research Journal of Modernization in Engineering Technology and Science, vol. 06, no. 01, January 2024.
- [3] D. D. Meshram, P. Dighore, A. Hedao, K. Dhoble, and K. Paunikar, "Virtual Mouse Using Hand Gesture," EasyChair preprints, May 13, 2024.
- [4] S. Mane, G. Jadhav, V. Sherkar, S. Jadhav, and P. Patil, "Hands-Free Computing Gesture-Based Virtual Mouse and Keyboard," Sinhgad Institute of Technology, Lonavala, Maharashtra, India, International Research Journal of Modernization in Engineering Technology and Science, vol. 06, no. 02, February 2024.

- [5] P. Muntode, V. Sonawane, S. Patil, and G. Bhandari, "Virtual Mouse And Keyboard," Jspm's Bhivrabai Sawant Institute Of Engineering, Pune, Maharashtra, India, International Research Journal of Modernization in Engineering Technology and Science, vol. 06, no. 02, February 2024.
- [6] I. Khan, V. Kanchan, S. Bharambe, A. Thada, and R. Patil, "Gesture Controlled Virtual Mouse with Voice Assistant," Terna Engineering College, Nerul, Navi Mumbai, India, International Journal of Multidisciplinary Engineering in Current Research - IJMEC, vol. 9, no. 01, Jan. 2024.
- [7] Abhijeet and M. D. Asif, "Deep Learning Based Real Time Ai Virtual Mouse System Using Computer Vision," J.B Institute of Engineering and Technology, 2024.