GestoMouse: A Real-Time Gesture-Controlled Virtual Mouse System

Mohammed Shahil[1], Gishal S[2], K MGopika[3], Nandana A K[4], Ms Sruthi A[5]

[1]*Mohammed Shahil K M, Bachelor of Technology in CSE, NCERC*
[2]*Gishal S, Bachelor of Technology in CSE, NCER*
[3]*K M Gopika, Bachelor of Technology in CSE, NCERC*
[43]*Nandana A K, Bachelor of Technology in CSE, NCERC*
[5] *Professor, Department in CSE, NCERC*

---***---

1.ABSTRACT

The use of hand gesture recognition in controlling virtual devices has become popular due to the advancement of artificial intelligence technology. A hand gesture-controlled virtual mouse system that utilizes AI algorithms to recognize hand gestures and translate them into mouse movements is proposed in this paper. The system is designed to provide an alternative interface for people who have difficulty using a traditional mouse or keyboard. The proposed system uses a camera to capture images of the user's hand, which are processed by an AI algorithm to recognize the gestures being made. The system is trained using a dataset of hand gestures to recognize different gestures. Once the gesture is recognized, it is translated into a corresponding mouse movement, which is then executed on the virtual screen. The system is designed to be scalable and adaptable to different types of environments and devices. All the input operations can be virtually controlled by using dynamic/static hand gestures along with a voice assistant. In our work we make use of ML and Computer Vision algorithms to recognize hand gestures and voice commands, which works without any additional hardware requirements.

Key Words: optics, photonics, light, lasers, templates, journals

1.INTRODUCTION

As technology evolves, so does the way we interact with computers. Traditional input devices like keyboards and mice, while effective, have limitations—especially when it comes to flexibility, accessibility, and user experience. This has led to a growing demand for more natural, intuitive interaction methods, such as gesture-based interfaces. Gesto Mouse is a gesture-controlled virtual mouse system designed to replace the conventional mouse with hand movements. Using a standard webcam, Gesto Mouse tracks and interprets hand gestures in real time, translating them into mouse cursor movements and click events. This touch-free system is particularly valuable in settings where hygiene is critical or for users with physical impairments who find traditional devices difficult to use. The system works by leveraging computer vision and machine learning algorithms. Through image processing, Gesto Mouse detects and tracks the user's hand, identifying gestures such as moving, clicking, or scrolling. It then translates these gestures into corresponding actions on the screen, allowing users to control their computer seamlessly without needing to physically touch any device. Gesto Mouse not only enhances user accessibility but also represents a step toward more hygienic and efficient computing experiences. Its potential extends beyond personal use to industries such as healthcare, where it can be used in sterile environments, or in education and entertainment, where it can create more interactive, engaging experiences. In essence, Gesto Mouse offers a glimpse into the future of human-computer interaction, paving the way for more fluid, intuitive, and inclusive computing experiences. With its gesture-based controls, it is a major step forward in the evolution of how we interact with technology.

2.LITERATURE SURVEY

2.1. Glove-Based Gesture Recognition Systems

Early gesture recognition systems utilized wearable technologies like the DataGlove, which contained sensors to track hand and finger movements. These systems offered high accuracy but had several limitations such as high cost, physical discomfort, and the need for users to wear specialized equipment. This made them impractical for everyday or long-term use. Human-Computer Interaction (HCI) has advanced significantly over the past decades, shifting from traditional input devices like keyboards and mice to more natural, intuitive methods such as voice and gesture recognition. Among these, gesture-based interaction has gained considerable attention due to its ability to provide a touchless and user friendly interface. Early work in gesture recognition focused primarily on glove-based systems, such as the Data Glove, which captured hand and finger movements with the help of embedded sensors. Although these systems provided high precision, they were often expensive, required physical contact, and were uncomfortable for prolonged use, limiting their application in daily computing environments.

2.2. Markerless Real-Time Recognition with OpenCV and MediaPipe

Markerless real-time gesture recognition has significantly advanced the way humans interact with computers, making it more natural and accessible. Instead of relying on physical devices like gloves or markers, modern systems utilize computer vision to detect and track hand movements using only a standard webcam. Two widely used tools that enable this technology are OpenCV and MediaPipe. OpenCV is a powerful open-source library that provides various image processing techniques such as skin detection, contour analysis, and motion tracking, which are essential for identifying hand gestures. On the other hand, MediaPipe, developed by Google, offers a high-performance framework specifically optimized for real-time hand tracking. It can detect 21 hand landmarks, allowing the system to understand finger positions and movements with high precision. The combination of OpenCV and MediaPipe enables accurate, real-time gesture recognition without the need for additional hardware or wearables. This markerless approach is not only cost-effective but also enhances user experience by offering a more intuitive and hygienic interface.

2.3. Classification of Gestures: Static vs. Dynamic

According to the work of Mitra and Acharya (2007), gestures can be classified into: Static gestures: hand poses that do not change over time. Dynamic gestures: gestures involving movement, such as swiping or dragging. Understanding both types is crucial in designing systems like Gestomouse that need to detect cursor movement as well as click actions. With the advancement of computer vision libraries like OpenCV and MediaPipe, real-time, markerless gesture recognition has become more accurate and accessible. These tools leverage techniques like skin detection, contour analysis, and landmark tracking to detect hand gestures using only a webcam. MediaPipe, for example, can detect 21 hand landmarks, enabling smooth interaction without any wearable devices. Classification of Gestures: Static vs. Dynamic:-Gestures used in human-computer interaction can generally be classified into two main categories: static gestures and dynamic gestures. This classification, as discussed by researchers like Mitra and Acharya, helps in designing systems that can accurately interpret user intentions. Static gestures refer to hand poses or shapes that are held steady for a brief period, such as showing a thumbs-up or an open palm. These gestures do not involve any movement over time and are recognized based on the shape or configuration of the hand at a specific moment. They are relatively easier to detect and process, making them suitable for applications like sign language interpretation or simple command inputs. In contrast, dynamic gestures involve motion and are recognized based on the movement of the hand over time. Examples include waving, pointing, dragging, or making a circle with the finger to simulate scrolling. Recognizing dynamic gestures is more complex, as it requires tracking the trajectory and speed of hand motion, often involving temporal data processing and pattern recognition. Dynamic gesture recognition systems must analyze continuous sequences of frames and often rely on motion tracking or temporal machine learning models like Hidden Markov Models (HMMs) or Recurrent Neural Networks (RNNs). For a system like Gestomouse, combining both static and dynamic gesture recognition is crucial to enable functionalities like click simulation (static) and cursor movement (dynamic), providing a seamless and intuitive user experience. 4. Machine Learning and Deep Learning Approaches:-Recent systems incorporate machine learning algorithms, especially Convolutional Neural Networks (CNNs), to classify complex gesture patterns with higher accuracy. These models improve adaptability to different users and environments but require significant training data and computational resources, which can be a NEHRUCOLLEGEOFENGIN

4. MODULE DESCRIPTION

<u>Input Capture Module:</u>
Captures real-time video feed from a webcam or motion sensor. Continuously streams hand movement data for processing

<u>Preprocessing Module:</u>
Enhances raw input by adjusting lighting, contrast, and filtering noise. Prepares clean image frames for accurate gesture recognition.

<u>Gesture Recognition Module:</u>
Detects hand shapes,positions and movements. Uses image processing or machine learning to recognize predefined gestures.

<u>Action Mapping Module:</u>
Maps recognized gestures to specific mouse actions (e.g., move, click, scroll). Translate gesture input into control commands.

<u>Mouse Control Interface Module:</u>
Sends commands to the operating system to simulate mouse functions. Execute actions like cursor movement, left/right clicks, dragging, etc.

<u>User Interface Module:</u>
Allows users to interact with the system settings and calibrate gestures. Displays feedback such as gesture detection success or errors.

<u>System Logger and Error Handler Module:</u>
Logs system activity and detected errors. Handles exceptions and provides alerts for failed gesture detection or hardware issues.

5.TECHNOLOGIES USED

OpenCV(Open Source Computer Vision Library): Used for image processing tasks such as video capture, frame conversion, contour detection, and background filtering.

MediaPipe: Developed by Google, this framework is used for real-time hand tracking and detection of 21 hand landmarks to accurately recognize gestures.

Python Programming Language: Acts as the main language for implementation due to its simplicity, rich library support, and ease of integration with vision-based tools.

NumPy: Used for efficient array operations and mathematical computations involved in gesture processing.

Webcam(Standard Camera): Serves as the only required hardware to capture live video input for gesture recognition

SOFTWARE REQUIREMENTS

Operating System: Windows 10/11, Linux, or macOS

Programming Language: Python (preferred for development) Libraries/Frameworks:OpenCV (for image processing) MediaPipe or TensorFlow (for gesture recognition) PyAutoGUI (for mouse control)

IDE: Visual Studio Code, PyCharm, or any Python-compatible IDE.

HARDWARE REQUIREMENTS

Processor: Minimum Intel i3 (Recommended: Intel i5 or higher).

RAM: Minimum 4GB(Recommended: 8 GB or higher).

Storage: At least 500 MB of free disk space.

Monitor: Minimum resolution of 1024x768

6.SYSTEM DESIGN

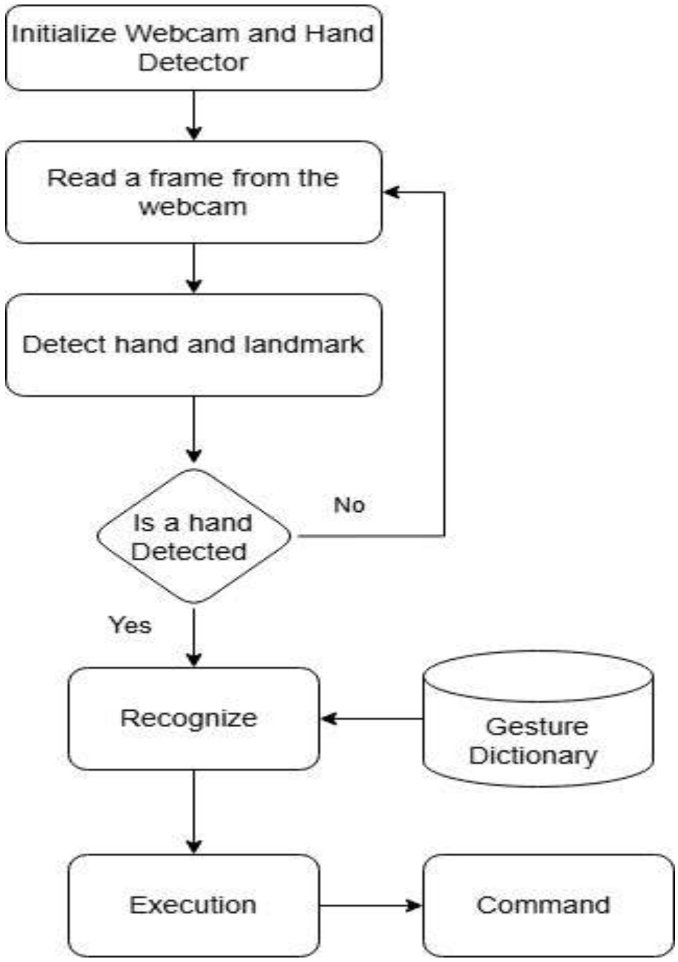

Figure6.1 flowchart

7. RESULTS

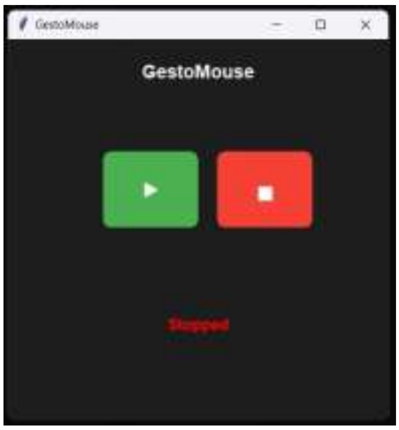

Figure 7.1 UI

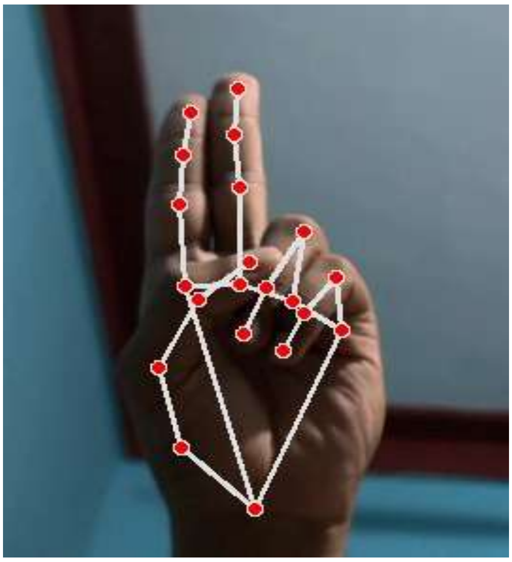

Figure **7.**2 gesture for double click

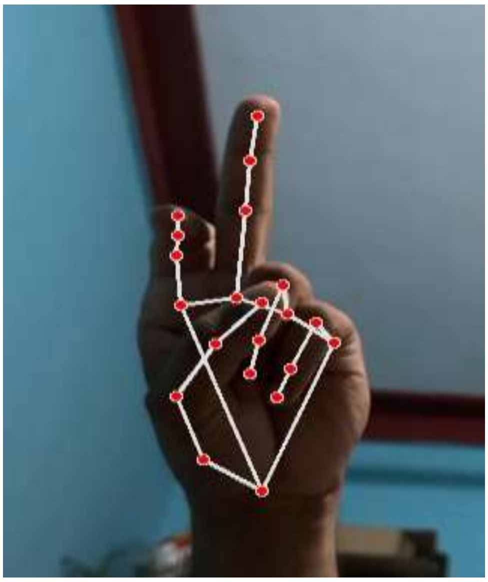

Figure **7.**3 gesture for left click

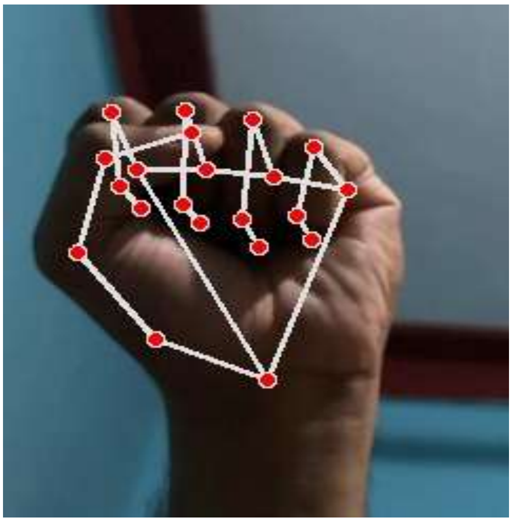

Figure **7.**3 gesture for drag and drop

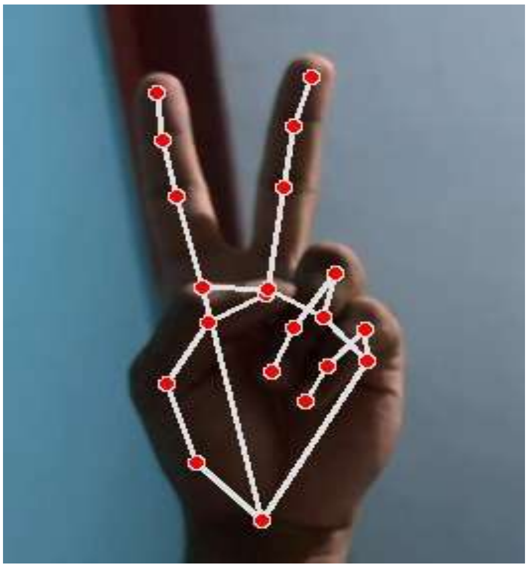

Figure 7.4 gesture for cursor movements

8.DISCUSSION AND ANALYSIS

8.1. ADVANTAGES

The GestoMouse system introduces a user-friendly and cost-effective approach to human-computer interaction by replacing traditional input devices with real-time hand gestures. By leveraging only a standard webcam, it eliminates the need for expensive or intrusive hardware such as gloves or depth sensors. This not only reduces overall system cost but also improves user comfort and mobility. The contactless nature of the interface makes it ideal for use in sterile or high-hygiene environments such as hospitals, labs, or public kiosks. Additionally, the system is highly accessible to individuals with physical disabilities who may find conventional mouse or keyboard usage challenging. Its modular architecture and use of open-source frameworks further allow easy customization, platform compatibility, and scalability across various computing environments.

8.2. FUTURE WORKS

While the current implementation of GestoMouse performs well under standard conditions, several areas for enhancement remain. One key improvement involves expanding gesture recognition to adapt to varying lighting conditions, background clutter, and different skin tones to ensure consistent performance. Future versions may incorporate deep learning models with higher precision and temporal awareness, such as LSTMs or Transformers, to better detect complex and dynamic gestures. Integration with voice assistants and smart devices could further enhance usability, particularly for smart home or IoT applications. Moreover, extending the system to recognize multi-hand gestures or adding support for customizable gesture commands could broaden its application scope, making it more versatile in domains such as gaming, education, and virtual reality.

9.CONCLUSION

The Gestomouse system represents a significant step forward in user interaction technologies, offering an intuitive and touch-free way for users to control their computers or digital devices. By utilizing gesture recognition, Gestomouse provides an alternative to traditional input devices like mice and trackpads, enhancing user experience in various fields such as accessibility, gaming, virtual reality (VR), and professional applications. Its core features-cursor movement, clicking, scrolling, and gesture customization-create an interactive system that allows for more natural and ergonomic control .The future of Gestomouse holds exciting potential for further advancements that could enhance its functionality and usability. One key area for improvement is the accuracy of gesture recognition. As machine learning and artificial intelligence continue to evolve, integrating more sophisticated algorithms will help the system better interpret complex hand movements and adapt to different environments, such as low lighting or noisy backgrounds.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our Mentor Ms. SRUTHI A for her invaluable guidance, encouragement, and continuous support throughout the development of this project. Her expertise and insightful feedback played a crucial role in enhancing our understanding and improving the project's quality. We are also thankful to Nehru College of Engineering and Research Centre, Thiruvilwamala, for providing the necessary resources and a conducive learning environment, enabling us to explore and apply our technical knowledge effectively. Finally, we deeply appreciate the unwavering support and encouragement from our family and friends, whose motivation kept us inspired throughout this journey. This project has been a significant learning experience, and we are truly grateful to everyone who contributed to its successful completion.

REFERENCES

[1] Mitra, S., & Acharya, T. (2007). *Gesture recognition: A survey*. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 37(3), 311–324.

https://doi.org/10.1109/TSMCC.2007.89320

[2] Zhang, C., Tian, Y., & Liu, Y. (2020). *Hand gesture recognition using deep learning networks with enhanced channel attention*. IEEE Transactions on Industrial Electronics, 67(11), 9356–9366.

https://doi.org/10.1109/TIE.2019.2940954

[3] Zhang, X., Cui, Z., & Wang, Y. (2021). *A CNN-based hand gesture recognition system using real-time video input*. Procedia Computer Science, 183, 232–239.

https://doi.org/10.1016/j.procs.2021.02.031

[4] Hassan, S. M., & Hannan, M. A. (2021). *Hand Gesture Recognition: A Literature Review*. International Journal of Human-Computer Interaction, 37(4), 308–328.

https://doi.org/10.1080/10447318.2020.1860516

[5] Kim, Y., & Choi, J. (2018). *A Deep Learning Approach to Hand Gesture Recognition for Human–Computer Interaction*. Sensors, 18(11), 3766.

https://doi.org/10.3390/s18113766