IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Gesture-Based Interaction Using Computer Vision And Python

[1]Yash Vardhan, [2]Parth Sarthi, [3] Dr Aditya Kishor Saxena
[1]B.Tech in Computer Science and Eng., B.Tech in Computer Science and Eng., [3]Associate Professor
[1]Galgotias University, Greater Noida, [2]Galgotias University, Greater Noida
[3]Galgotias University, Greater Noida

Abstract: Computers are a huge part of our lives, and a lot of our daily work depends on them. Making them easier and more efficient to use is something we are always trying to improve. One of the most important tools for interacting with a computer is the mouse. While wireless mice, like Bluetooth ones, help us cut cords, they still need a USB connection, so they're not entirely device-free. This system solves that by offering a way to control the computer's cursor without any physical mouse, using just a camera. It uses MediaPipe and OpenCV to detect hand gestures through machine learning, allowing users to move the cursor, click, and scroll—completely hands-free. This makes interacting with computers easier and more convenient, without the need for any extra devices.

Index Terms **- Computer vision, hand gesture recognition, Media-pipe, virtual mouse.**

1.INTRODUCTION

Advancements in artificial intelligence (AI) and human-computer interaction (HCI) have significantly transformed the way we interact with digital devices. One of the most exciting developments in this space is the use of hand gesture recognition to control virtual interfaces. This technology offers a more intuitive and natural way to communicate with computers, especially for users who find conventional input methods like the mouse or keyboard challenging. Gesture recognition, a subfield of computer vision, uses AI algorithms to interpret hand movements, allowing users to interact with devices through simple gestures, without physical touch.

This paper introduces a hand gesture-controlled virtual mouse system designed to enhance user interaction by eliminating the need for a physical mouse. The system leverages machine learning and computer vision techniques, such as MediaPipe and OpenCV, to recognize hand gestures captured through a camera and convert them into mouse commands. By recognizing and processing various hand movements, the system can perform tasks like cursor navigation, clicks, and scrolling, creating a hands-free computing experience.
his innovative approach not only simplifies human-computer interaction but also opens up possibilities for more inclusive and accessible interfaces. By removing the reliance on traditional hardware, this system can adapt to diverse environments and user needs, making it a powerful tool for enhancing productivity and user engagement in the digital world.

I. PROBLEM DESCRIPTION AND OVERVIEW

Computers have become an essential tool in our daily lives, but they are not always accessible to everyone, especially the elderly and people with disabilities. For many, controlling a computer with a mouse or keyboard can be physically and cognitively demanding. This creates a barrier to technology for those who may need extra assistance, making it difficult for them to fully benefit from modern devices.

To address this, we propose an AI-powered virtual mouse system that allows users to control their computers using simple hand gestures. This system eliminates the need for physical contact with a traditional mouse, reducing the effort required to perform tasks like clicking, scrolling, and navigating. By recognizing both static and dynamic hand gestures, this AI system makes interacting with computers much easier and more intuitive, especially for users who face challenges with conventional input methods.

II. OBJECTIVE

The main goal of the Gesture Controlled Virtual Mouse is to create an alternative way for users to interact with their computers using only hand gestures. This system aims to replace the physical mouse with a touch-free interface, allowing users to control the cursor, perform left and right clicks, double-click, scroll, and even manage volume and brightness. We also aim to expand the system's functionality by adding new features, such as AirCanvas, which would allow users to draw or write using gestures in a virtual space. Ultimately, this project will improve accessibility, make technology more inclusive, and provide a more seamless user experience for a wide range of people.

2. LITERATURE REVIEW

There are multiple techniques for managing cursor movement using hand gestures; however, traditionally, DataGlove had to be worn for extended periods. This approach reduces the effectiveness of both the user and system performance. One of the major challenges of this method is its system complexity. In a study published by Dung-Hua Liou
and Chen-Chiung Hsieh, adaptive skin color models along with a motion history image based technique for detecting hand movement direction were implemented. The project achieved an average accuracy of 94.1%, with processing taking 3.81 milliseconds per frame. However, a key limitation was the difficulty in recognizing more intricate hand gestures in real-world environments. [2] Chang-Yi Kao and Chin-Shyurng Fahn primarily focused on the use of visual hand gesture identification within a human-computer interaction (HCI) interface for control purposes. Experimental results showed a face tracking rate of over 97% under normal conditions and over 94% when the face was
temporarily obscured. The system's performance is excellent, leading to plans for commercializing the robot. However, high-end computers are required to achieve precise results. [3] The main aim of the research conducted by Angel et al. was to design a real time hand gesture detection system using a skin color model. Hand gestures can effectively convey ideas and actions, and the various hand forms identified by the system can be processed to trigger related events, providing a more natural interface for computer vision systems. However, the system could not perform well in complex environments and was only functional in proper lighting conditions. [4]

Another study explored a multimedia-based hand gesture recognition system using simple computer vision techniques. In a paper published by Ashwini M. Patil and colleagues, a significant limitation was found before gesture comparison algorithms could be utilized— hand segmentation and skin pixel identification from stored frames needed to be completed first. [5] In a project that used color detection methods to capture hand motions via a camera, the key component of this technique was the webcam. The study by Abhik Banerjee and Abhirup Ghosh focused on constructing a cost-effective virtual human
computer interface device. However, their system had limitations such as requiring a light operating background and avoiding brightly colored objects. High-configuration computers were necessary for optimal functionality. [6] Yimin Zhou et al. reported on a study that introduced a high-level hand feature extraction technique for real-time gesture detection. The system exhibited high accuracy in recognizing both flexed and extended fingers. However, the system's use was limited to high-configuration computers. [7] In an experiment conducted by Pooja Kumari and colleagues, several color bands were used to perform

various functions. Mouse actions were controlled by the number of colors rather than different gestures. This reliance on color limits gesture-based interaction. [8] In a study by Aashni Haria and her team, a background extraction and contour detection system was developed. They performed two rounds of evaluations to test the method's accuracy. The first round involved simple backgrounds, while the second used more complex backdrops.
Each gesture was repeated ten times per setting, and the average accuracy was 85% for simple backgrounds and 80% for complex ones. However, the system's performance was very slow. [9]

Another project focused on controlling a cursor using hand gestures detected through a webcam's color detection technique. Published by Abhilash SS and others, the system was limited to a few mouse actions and did not work with a static background. [10] Kollipara Sai Varun and colleagues provided a detailed explanation of the algorithms and methods for virtual mouse control using color detection. OpenCV (Open Source Computer Vision Library) was used for video capture, and the highlighted color chosen by the user controlled mouse movement. [11] This project has applications in presentations and can reduce workspace requirements and the need for additional hardware. Fingertip tracking for a virtual mouse is a popular approach to interacting with computers without using a traditional mouse. In this paper, Kabid Hasan Shibly and colleagues proposed a virtual mouse system that uses fingertip detection and RGB-D images. The system captures frames using a webcam and processes them to track gestures, which are then translated into mouse functions. This approach eliminates the dependency on hardware mice and can aid in advancing HCI technology. The system was implemented in Python using the OpenCV library and could potentially replace conventional mice and remote controllers. [12] The goal of an AI-simulated mouse device is to replace the need for a hardware mouse by controlling the cursor with hand gestures. In a study published by B. Nagaraj et al., this project achieved 99% accuracy, surpassing previous proposals. The system used hand gesture recognition to create a virtual mouse, tracking hand movements to control the cursor. [13] Ahmed, Muhammad, et al., presented various object detection techniques, such as object detection (OD), salient object detection (SOD), and category-specific detection (COD). They analyzed several deep learning methods used in complex environments and compared their performance in terms of output, time efficiency, and traditional techniques. Their work also provided an overview of the available public datasets and discussed evaluation metrics while highlighting the limitations and future research directions. [14] V. Tiwari et al. used the VGC16 pre-trained model for image classification. They compared its results with other models, such as the baseline CNN and a three-block VGG model, and analyzed the impact of data augmentation on accuracy. The VGC16 model achieved an accuracy of 98.97%, which was significantly higher than the baseline CNN and the VGC3 models. [15]

A model with server-side and client-side components was proposed by Li Wensheng et al. On the server side, they used adaptive online training for mouse movement and finger detection through a BP neural network, as well asfinger tracking via a mean shift algorithm,
generating appropriate messages to be sent to the client. On the client side, the system processed the received messages and used the server's API function to get the coordinates of multiple fingertips to control the application. However, inconsistent results were observed due to varying skin tones during adaptive online training. [16]

3. ALGORITHMS AND TOOLS USED

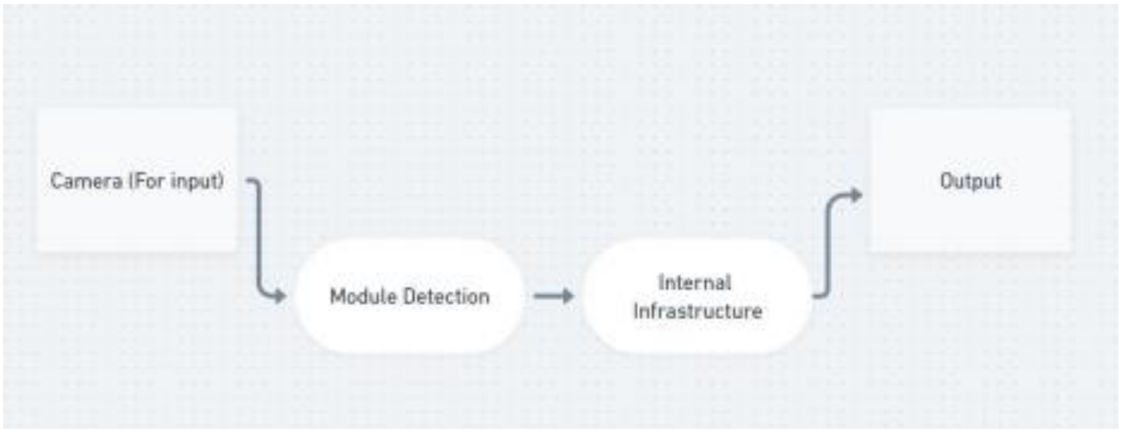

Figure 1 : Block Diagram of System

3.1. MEDIAPIPE

MediaPipe, developed by Google, is an open-source framework that provides a wide range of machine learning solutions tailored for tasks like hand tracking, object detection, and pose estimation. This framework efficiently processes visual data using pre-built models and pipelines, making it highly suitable for gesture recognition applications. Some key features of MediaPipe include:
Video and Audio Analysis: Real-time processing of video and audio streams, including decoding, filtering, and segmentation.

- Facial Landmark Tracking: Detection and continuous tracking of facial features, enabling applications such as facial recognition and emotion detection.
- Hand Motion Tracking: Accurate tracking of hand movements for gesture-based interaction with virtual interfaces.
- Real-time Object Recognition: Enables the detection and tracking of objects in real-time, supporting applications like augmented reality and robotics.

3.2. OpenCV (Open Source Computer Vision Library)

OpenCV is a highly popular open-source library for computer vision and machine learning, offering various tools for image processing, object detection, and gesture recognition. OpenCV plays a fundamental role in many hand gesture-controlled systems, allowing the processing of images and videos, face detection, and more. It simplifies the development of AI-driven computer vision applications and integrates seamlessly with machine learning techniques.

3.3. Deep Learning Frameworks (TensorFlow, PyTorch)

Frameworks such as TensorFlow and PyTorch are widely used to build and train neural networks, providing necessary tools for creating, optimizing, and deploying deep learning models. These frameworks are crucial in gesture recognition systems, where complex models are needed to interpret hand gestures accurately.

3.4.Image Processing Techniques

Techniques such as thresholding, contour detection, edge detection, and feature extraction are applied to preprocess images before feeding them into gesture recognition models. These methods enhance input data quality, ensuring more precise gesture detection.

3.5. Python Programming Language

Python is an ideal choice for building gesture-controlled systems due to its simplicity and a vast array of machine learning and computer vision libraries. With its ease of use, Python allows developers to efficiently integrate models and frameworks like OpenCV and MediaPipe for creating robust AI systems.

3.6.**Hardware Requirements (Webcam)**

The system relies on a webcam (either built-in or external) to capture hand movements. The quality and responsiveness of the gesture control are largely dependent on the clarity and resolution of the webcam. High-quality webcams provide sharper image capture, leading to more accurate gesture recognition and smoother user experience.

4. METHODOLOGY

4.1. Capturing Video and Processing

The system begins by capturing real-time video using a webcam. The captured frames are then processed to detect hand positions. The video feed is converted from BGR (Blue-Green-Red) to RGB (Red-Green-Blue) using OpenCV to make it compatible with gesture recognition algorithms.

```
def findHands(self, img, draw=True):
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)
```

3.2. Rectangular Area for Gesture-Based Mouse Control

Within the display window, a rectangular area is assigned where the system tracks hand gestures to perform mouse actions. Once hands are detected within this area, the system identifies which fingers are extended, allowing the user to control the cursor and perform designated actions.

- Finger Recognition: The system begins moving the cursor as soon as it detects which fingers are raised.
- Dynamic Hand Gestures: Depending on the configuration, different gestures trigger specific actions such as left-click, right-click, or scrolling.

By continuously tracking the hand's movement and position in the camera's frame, the system can perform seamless mouse operations without the need for any physical mouse

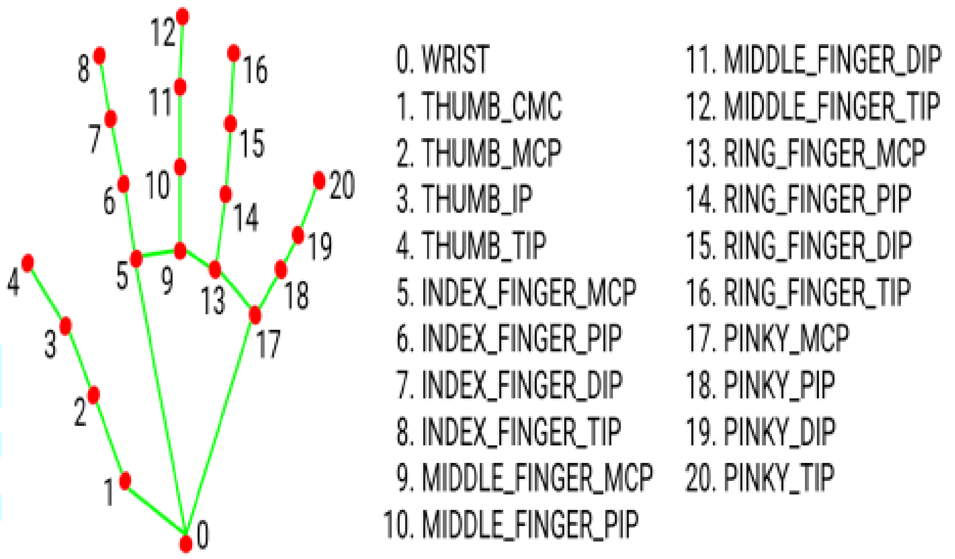

Figure 2 : Mediapipe Coordinates

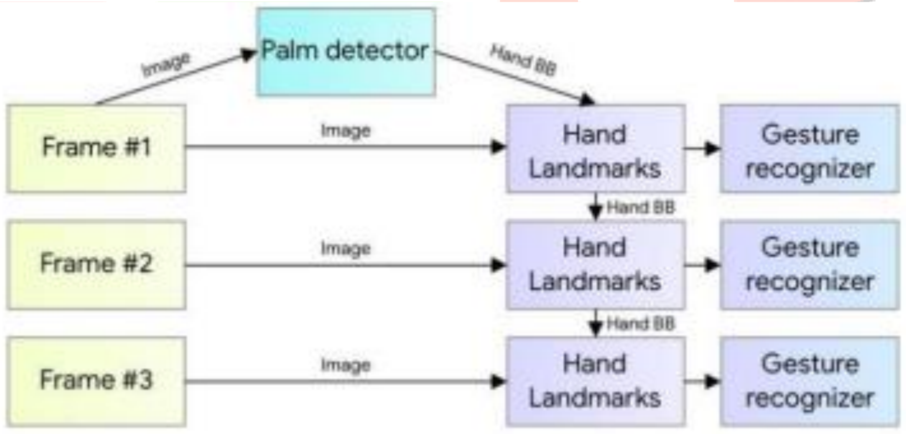

Figure 3 : System OverView

5.IMPLEMENTATION

In the proposed system, MediaPipe and OpenCV are used to enable a touch-free, gesture controlled virtual mouse. By recognizing hand movements through the camera, users can perform common mouse functions without any physical contact. Below are the main gestures and how they are implemented:

5.1. Neutral Gesture (Stopping Gesture)

- Purpose: When no action is intended, the neutral gesture keeps the system idle. This could be an open hand without any significant movement.
- Implementation: When the system doesn't detect any gesture or detects a specific neutral position (like an open hand), it doesn't perform any mouse operation.

5.2.**Move Cursor**

- Purpose: Move the cursor by simply moving your hand.
- Implementation: The system tracks the hand's position in real-time and moves the cursor accordingly. The speed of the cursor is proportional to the hand's movement speed.

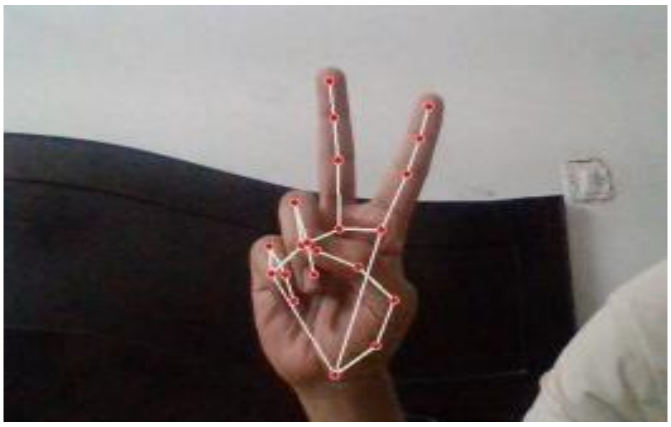

Figure 4 : Cursor Movement

5.3.**Left Click**

- Purpose: Perform a single left-click.
- Implementation: A specific gesture, such as pinching two fingers together, can be mapped to a left-click.

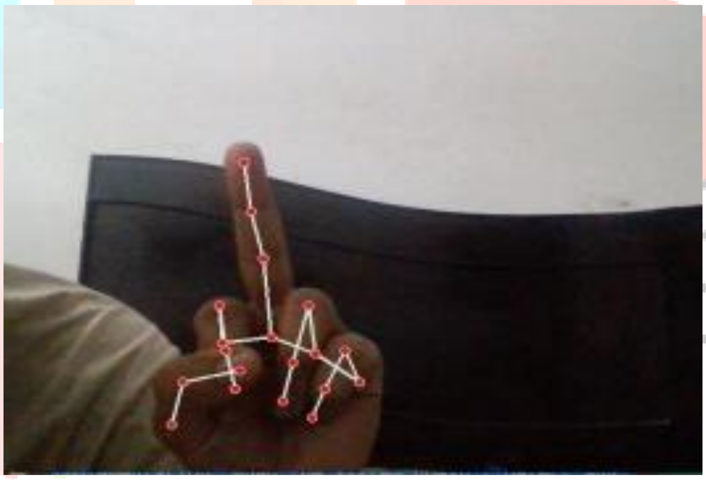

Figure 5 : Left Click

5.4.**Right Click**

- Purpose: Perform a right-click.
- Implementation: Another distinct gesture, like forming a circle with the index finger and thumb, can trigger a right-click.

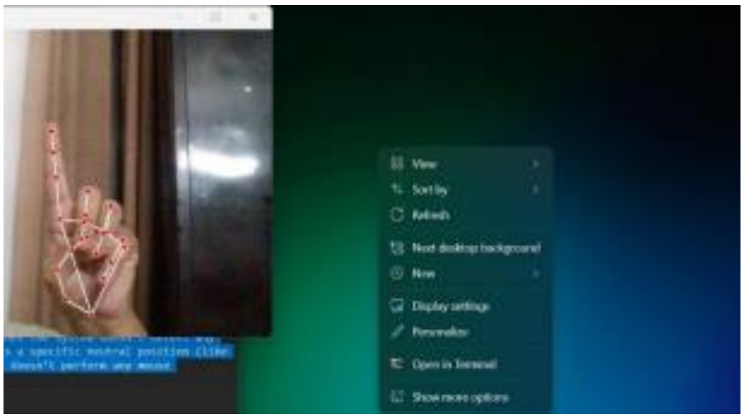

Figure 6 : Right Click

5.5. Scrolling

- Purpose: Scroll vertically or horizontally.
- Implementation: The pinch gesture can be used for scrolling, with vertical or horizontal hand movements determining the scroll direction.

5.6. Drag and Drop

- Purpose: Drag and drop files or icons.
- Implementation: Once the pinch gesture is activated (holding down), you can drag an item, and once the pinch is released, the item is dropped.

5.7. Multiple Item Selection

- Purpose: Select multiple items.
- Implementation: A specific gesture, such as holding two fingers together, can trigger the selection of multiple items.

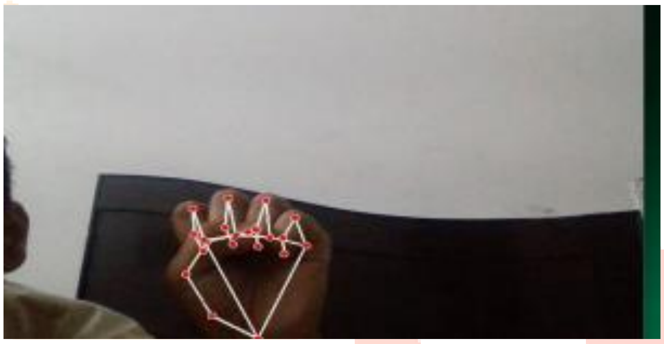

Figure 7 : Movement

5.8. Volume Control

- Purpose: Control system volume.
- Implementation: The pinch gesture can be used to increase or decrease volume, with the volume level changing based on how far the gesture moves.

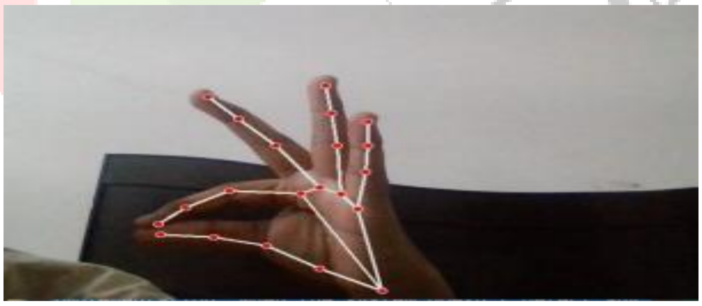

Figure 8 : Volume Control

5.9. Adding the AirCanvas Implementation

For the AirCanvas feature, you could implement a gesture that allows users to draw or write on a virtual canvas using their finger movements.

AirCanvas Gesture

- Purpose: Draw on a virtual canvas using hand gestures.
- Implementation: The system tracks the hand's movements, and by detecting a specific gesture (like holding a finger out), it allows the user to "draw" on the screen.

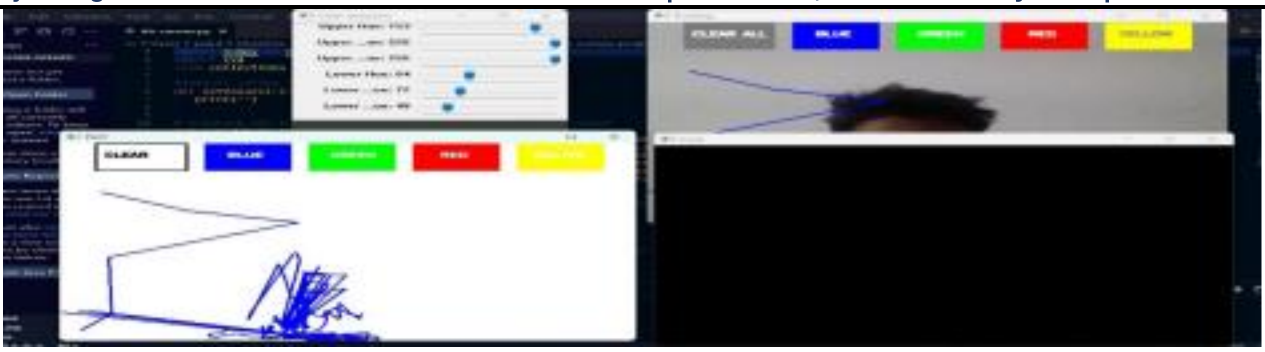

Figure 9 : Air Canvas

6. Conclusion

The goal of our proposed system is to replace traditional computer mouse functions with hand gestures using computer vision technology. This system is made possible by utilizing an external or inbuilt webcam to capture live video, which is processed to detect hand gestures and execute specific mouse actions. With an accuracy rate exceeding 90%, the system performs significantly better than many pre-existing models, making it suitable for a wide range of practical applications. Based on our experimental results, the AI-powered virtual mouse demonstrates strong performance and high precision, offering a reliable and efficient alternative to conventional input devices. While the system has shown remarkable results, there are still a few areas that need improvement, such as optimizing the right-click functionality and enhancing the drag-and-drop feature for smoother interactions. Moving forward, we plan to refine the system's algorithm to address these limitations and introduce additional features to further expand the system's capabilities. Overall, the virtual mouse system offers a more intuitive and seamless way to interact with computers, making navigation, scrolling, and other tasks more natural and efficient, providing users with a greatly enhanced experience.

7. Future Scope

Despite the system's overall success, there are still a few limitations to address, such as reduced precision in the right-click operation and the slightly cumbersome drag-and-drop functionality. These are areas where we are actively working to improve accuracy and user experience. In the future, we aim to introduce voice assistant capabilities to make the system even more intuitive and further enhance the HumanComputer Interaction (HCI) experience. Additionally, we plan to incorporate features like AirCanvas, allowing users to interact with virtual drawing spaces using hand gestures. By adding such functionalities, we hope to broaden the scope of the system, making it useful in creative applications and expanding its appeal to a wider audience. This continuous development will not only improve current usability but also position the system as a forward thinking solution for gesture-based interaction in various fields.

References

[1] Banerjee, A., Ghosh, A., Bharadwaj, K., & Saikia, H. (2014). Mouse control using a web camera based on colour detection. arXiv preprint arXiv:1403.4722.
[2] Hsieh, C.-C., Liou, D.-H., & Lee, D. (2010). A real-time hand gesture recognition system using motion history image. In Proceedings of the IEEE International Conference on Signal Processing Systems (ICSPS) (Vol. 2, pp. 10.1109/ICSPS.2010.5555462).
[3] Kao, C.-Y., & Fahn, C.-S. (2011). A human-machine interaction technique: Hand gesture recognition based on hidden Markov models with trajectory of hand motion. Procedia Engineering, 15, 3739–3743.
[4] Neethu, P. S. (2013). Real-time static & dynamic hand gesture recognition. International Journal of Scientific & Engineering Research, 4(3).
[5] Patil, A. M., Dudhane, S. U., & Gandhi, M. B. (2013). Cursor control system using hand gesture recognition. International Journal of Advanced Research in Computer and Communication Engineering, 2(5).
[6] Banerjee, A., Ghosh, A., & Bharadwaj, K. (2014). Mouse control using a web camera based on color detection. International Journal of Computer Trends and Technology, 9.
[7] Zhou, Y., Jiang, G., & Lin, Y. (2016). A novel finger and hand pose estimation technique for real-time hand gesture recognition. Pattern Recognition, 49, 102–114.

[8] Kumari, P., Singh, S., & Pasi, V. K. (2016). Cursor control using hand gestures. In Recent Trends in Future Prospective in Engineering & Management Technology (pp. 2016–2017). International Journal of Computer Applications.

[9] Haria, A., Subramanian, A., Asokkumar, N., Poddar, S., & Nayak, J. S. (2017). Hand gesture recognition for human-computer interaction. In Proceedings of the 7th International Conference on Advances in Computing and Communications, ICACC- 2017 (pp. 367–374). Procedia Computer Science.

[10] Abhilash, S. S., Thomas, L., Wilson, N., & Chaithanya, C. (2018). Virtual mouse using hand gesture. International Research Journal of Engineering and Technology (IRJET), 5(4), 37.

[11] Varun, K. S., Puneeth, I., & Jacob, T. P. (2015). Virtual mouse implementation using OpenCV. In Proceedings of the Chinese Intelligent Systems Conference (Vol. 2, pp. 329–340).

[12] Shibly, K. H., Dey, S. K., Islam, M. A., & Sourav, S. I. (2021). Design and development of hand gesture-based virtual mouse. In D. L. Quam (Ed.), Gesture recognition with a data glove (Vol. 2, pp. 755–760).

[13] Nagaraj, B., Jaya, J., Shankar, S., & Ajay, P. (2021). Deep learning-based real-time AI virtual mouse system using computer vision to avoid COVID-19 spread. Journal of Applied Artificial Intelligence, 11(2).

[14] Ahmed, M., et al. (2021). Survey and performance analysis of deep learning-based object detection in challenging environments. Sensors (Basel, Switzerland), 21(15), 5116. https://doi.org/10.3390/s21155116

[15] Tiwari, V., Pandey, C., Dwivedi, A., & Yadav, V. (2020). Image classification using deep neural network. In Proceedings of the 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN) (pp. 730–733).

[16] Li, W., Deng, C., & Lv, Y. (2010). Implementation of virtual mouse based on machine vision. In Proceedings of the International Conference on Apperceiving Computing and Intelligence Analysis (pp. 367–371).