GESTURE-CONTROLLED VIRTUAL MOUSE FOR HANDS-FREE NAVIGATION

J. Amutha[1]
Assistant Professor
Computer Science and Engineering
P.S.R Engineering College, Sivakasi
Virudhunagar, India
jothi.amu93@gmail.com

S. Priyadarsini[2]
Professor
Computer Science and Engineering
P.S.R Engineering College, Sivakasi
Virudhunagar, India
priyadarsini@psr.edu.in

M. Rabin[3]
UG Student
Computer Science and Engineering
P.S.R Engineering College, Sivakasi
Virudhunagar, India
Rabinma11022004@gmail.com

M. Raja Muniyandi[4]
UG *Student*
Computer Science and Engineering
P.S.R Engineering College, Sivakasi
Virudhunagar, India
Rajamuniya43@gmail.com

B. Raja Pandi[5]
UG Student
Computer Science and Engineering
P.S.R Engineering College, Sivakasi
Virudhunagar, India
rpthehulk@gmail.com

Abstract

The Virtual Mouse Hand Gesture Control System introduces an innovative solution for human-computer interaction by replacing traditional hardware-based mouse devices with gesture recognition technology. This system leverages advancements in machine learning and computer vision, utilizing frameworks such as Mediapipe and OpenCV, to interpret hand gestures captured via a standard camera. This effectively frees up hardware for the control of cursor devices, as input is achieved through combinations of varying muscle movements. Key functionalities, including cursor movement, clicking, and dragging, are mapped to specific hand gestures, enabling seamless and touch-free interaction. Performance evaluations demonstrate significant improvements in accuracy and responsiveness compared to existing gesture-based systems, achieving a hand recognition accuracy of 95%. The system's adaptability across various lighting conditions ensures a reliable user experience.

Keywords: Gesture recognition, Virtual mouse, Gesture control, Human-computer interaction, Real-Time processing, OpenCV, Touchless interaction.

1 Introduction

Artificial Intelligence (AI) has made its way into Human-Computer Interaction (HCI), changing the existing paradigms such as keyboard and mouse interface. Gesture-based controls offer an intuitive, touchless alternative that enhances accessibility and efficiency. This project explores an AI-driven virtual mouse system that enables users to control a mouse pointer through hand gestures, utilizing computer vision technologies. Implemented using Python, Mediapipe, and OpenCV, the system achieves real-time hand tracking and gesture recognition. It eliminates reliance on physical devices, making it ideal for individuals with mobility challenges or scenarios where traditional input devices are impractical. Additionally, it provides a more natural, ergonomic interaction model, reducing the risk of repetitive strain injuries associated with conventional input devices. The system also supports customization, allowing users to adapt gesture sensitivity and functionality to their needs. With potential applications in healthcare, gaming, and assistive technologies, this approach represents a significant step toward more inclusive and user-centric computing experiences. This project aims to contribute to the growing field of AI-driven HCI solutions by enhancing the flexibility and ease of interacting with digital environments.

S. No	Name of the journals	Methods are Used	Accuracy
1.	Virtual Mouse using Hand Gesture	Human Computer Interaction, Python, Color detection, Web camera, Hand Gestures.	94%
2.	Virtually Controlling Computer using Hand-Gestures and Voice.	Survey of Convolutional Neural Networks in Python with Pybind11 and OpenCV	95%
3.	ESP32-CAM & OpenCV let you Control Virtual Mouse with Gestures.	ESP32 CAM & OpenCV.	94.5%

Table 1: Virtual Mouse Methods Comparison

2 Related Work:

The concept of gesture-controlled interfaces has been widely explored in recent years. It offers an innovative alternative to traditional input devices such as keyboards and mice. These systems use humans' natural ability to gesture to interact with computers. Improve user experience in a touch-free and easy-to-use way. There is some promising research involving a gesture-controlled virtual mouse, but it is still in development. It enables the user to operate the computer through hand movements. Conventional virtual mouse systems are primarily based on hand posture [7]. And is paired with a defined action; a closed fist, for example, may represent a left click. While an open hand might signify a right click. This is limited to a small number of gestures. And encountering problems of accuracy and confusion in recognizing multiple overlapping gestures, there is a need for more complex systems.

Recent advances in computer vision and machine learning have made dynamic gesture recognition more feasible. This allows the system to continuously interpret hand movements and perform complex actions based on those movements. Developed by Google, Mediapipe has become a popular framework for real-time hand tracking. Uses deep learning algorithms to detect and track important hand gestures. This provides a high level of accuracy and adaptability for real-time applications. OpenCV is a widely used tool in computer vision. It plays an important role in processing video frames to efficiently track hand gestures. Such technologies form the foundation to establish better, faster and more flexible gesture-based systems. Unlike a static recognition system. The modern gesture-controlled virtual mouse systems combine hand gestures with new input modalities, like voice, providing flexibility and control through the combination of voice and user recognition. Hand gestures…Perform advanced actions without contacting a physical device. It, like with any machine, makes for a much smoother and more natural human-computer interaction (HCI). This allows users to perform actions such as mouse clicking, dragging, scrolling, and even typing with minimal direct contact with the computer; they can do it. The adoption of ESP32-CAM modules and embedded systems will play an important role in improving these systems with ESP32-CAM, a cost-effective and efficient Wi-Fi camera module. Allows for wireless gesture tracking It allows users to control their devices remotely. By combining the capabilities of the ESP32-CAM with Python applications using MediaPipe and OpenCV, developers can create powerful creations. Gestures without additional hardware. Can create a virtual mouse system that can be used the proposed AI-based virtual mouse system aims to improve existing systems by offering a comprehensive alternative to traditional input devices. This system processes hand gestures captured by the camera, such as left-clicking, right-clicking, scrolling, or dragging. It will be converted into action.

By integrating voice commands, the system provides hands-free control of your computer. Allows users to perform various tasks Get a lot with minimal physical interaction. This method not only improves the interaction between the user and the computer. But it also introduces the possibility of hands-free access and control in a variety of environments, with medical and industrial applications for I/O functions combining dynamic gesture recognition and voice commands that Powered by AI. The new solution is presented, which greatly reduces reliance on traditional input devices.

3 Proposed Methodology

The here mentioned system is proposed to create a virtual mouse for the user to coordinate into a computer using hand gestures and do not require any hardware peripherals. The system uses computer vision and machine learning algorithms to observe a user's hand and interpret specific finger movements and gestures, mapping these different users input actions to standard mouse actions like moving the mouse pointer, clicking, dragging, and scrolling. In this context, the main idea behind creating this system is to develop a contactless, intuitive, and effective user input that can supply HCI (Human-Computer Interaction) experience in lieu of using the input devices in general. OpenCV, an open-source computer vision library which supports the basic acquisition and processing of images in real-time, is used as the foundation of the system.

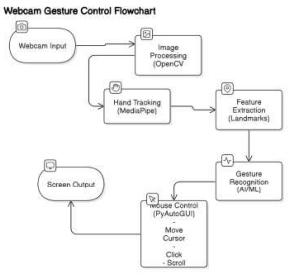


Figure 1. Gesture Control Flowchart

This system uses OpenCV to capture real-time video streams from the webcam, pre-process frames through grayscale conversion, thresholding, and Gaussian blurring in order to assist in hand detection performance. Frames now pass through the Mediapipe Hands module which is a deep-learning-based solution developed by Google for hand tracking and landmark detection in real-time. Mediapipe is a very powerful solution that enables the identification of 21 key points on each hand making it easy for the system to get all finger positions, orientation and relative movement patterns.

4 Implementations

After obtaining hand landmarks, the system employs gesture recognition algorithms to identify certain finger placements and associate these gestures with relevant commands similar to mouse actions. For example, moving your cursor requires just one index finger extended, and pinching your index finger and thumb together simulates a left-click. Swiping up or down with two fingers can also scroll, and dragging can be used by pinching two fingers. For accurate mouse movement, a number of techniques are used, including Kalman filtering to avoid jitter in hand motion, and noise reduction (especially important in real-life environments). In order to connect the recognized gestures to actual computer actions, the system uses the PyAutoGUI library that allows you to programmatically control the mouse and keyboard. PyAutoGUI has a lot of functions you need; you can move the cursor, click buttons, scroll pages, and even press keyboard keys, all programmatically. The detected hand gestures are mapped to these functions, enabling the virtual mouse to interact smoothly with applications, web browsers, and operating system interfaces. Some advantages of this system include contactless operation, which is advantageous in public spaces and health-sensitive sites. Moreover, it improves accessibility for disabled people, when they cannot use their hands properly, a simple touch of the finger with a g-ypad is an alternative input solution which is not dependent on the heuristics of moving the hand in an ordinate way to the screen like a mouse. By utilizing adaptive thresholds for gesture detection and dynamically adjusting the lighting and accuracy through machine-learning techniques, the system proves to be robust under differing conditions, including different illuminations and hand orientations. It improves gesture detection as well as enhances overall usability, thus heralding a new era of gesture-based computing as a scalable, low-overhead, and efficient input modality for next-generation human-computer interaction.

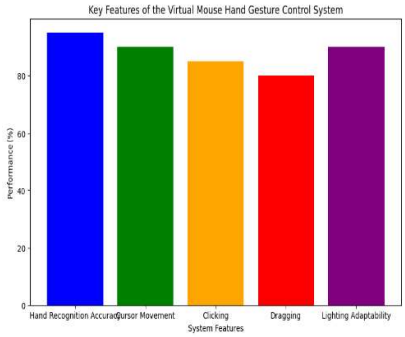


Figure 2: Key Features of the virtual mouse hand gesture control system

with the synergy of OpenCV and Mediapipe with PyAutoGUI, this system can not only create a user interface that induces a feeling of improvement but also promote further developments in the direction of touchless technology and AI-based interaction interfaces.

Algorithm:

BEGIN
 # Step 1: Train Gesture Classifier (One-time)
IF training_mode_enabled THEN
Collect landmark data and gesture labels
Train SVM classifier
Save trained model
END IF

 # Step 2: Load Gesture Classifier
Load the pre-trained SVM model from the saved file

 # Step 3: Initialize Hand Tracking and Cursor Smoothing
Initialize Mediapipe Hand Tracking
Initialize Kalman Filter for cursor smoothing

 # Step 4: Define Utility Functions

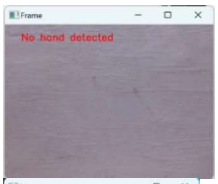
Convert hand landmarks to numerical feature array
RETURN processed landmarks
FUNCTION detect_lighting(frame):
Convert frame to grayscale
Calculate average brightness
IF brightness > threshold THEN
RETURN "bright"
ELSE
RETURN "dim"
END IF
FUNCTION adjust_thresholds(lighting_condition):
IF lighting_condition == "bright" THEN
Set gesture classifier threshold to high
ELSE
Set gesture classifier threshold to low
END IF
FUNCTION map_to_screen(cursor_pos, screen_width, screen_height):Convert hand position to screen coordinates
RETURN (mapped_x, mapped_y)

 # Step 5: Main Loop for Real-Time Tracking
Open webcam video stream
Get screen width and height
WHILE webcam is open:
Capture a frame
IF frame empty THEN
BREAK
Flip and process frame
Detect hand landmarks using Mediapipe
IF hand landmarks detected:
Detect lighting condition Adjust gesture classifier thresholds
FUNCTION preprocess_landmarks(landmarks):
FOR each detected hand:
Preprocess hand landmarks
Predict gesture using classifier
Get confidence score
Calibrate cursor using Kalman Filter for smoothing:
This maps the position of the hand to screen coordinates, and moves the mouse cursor to that location.
Gesture and the confidence score on the screen. So, the video feed will be shown with an overlay.
IF the 'ESC' key is pressed,
THEN BREAK. Release the webcam and close all windows.
END

5 Results and Discussion

For human-computer interaction, the suggested virtual mouse implementing hand gestures is a very efficient and intuitive method. It uses the Mediapipe Hand Tracking module and OpenCV to detect and track hand landmarks in real-time. Assocations were drawn between different hand gestures (wield, pinch, slide, finger position) and the fundamental mouse actions (cursor movement, left-click, right-click, double-click, drag-and-drop). The additional features of slide navigation and screenshot capture further enhanced the user experience. The performance of the system in real-time was further evaluated on the parameters of accuracy, responsiveness, and usability. High detection accuracy of the system well above 0.7 ensured reliable hand tracking. Smooth cursor movement was affected by a smoothing factor of 0.6, considerably eliminating abrupt jumps and increasing pointer stability. Different click gestures- left and right- could be duly identified through angle and distance computation between specific landmarks of the hands. The pinch gesture for drag and drop was one of the more operable gestures, letting one manipulate objects without the use of any physical input means seamlessly.

One of the system's greatest strengths is the fact that really exists solely the standard webcam at its disposal, permitting a low-threshold solution to touchless computing. Also, the movement smoothing improved the accuracy and solved the jittery hand-tracking problems that are recurrent in any gesture-based interface. During testing, though, a few problems were identified. Assessment of the landmarks for extraction was occasionally impaired due to environmental lighting and the background clutter.



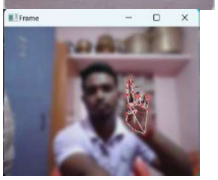
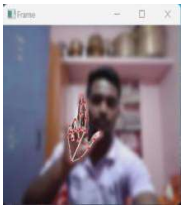


Fig.3 No Hand Detection Fig.4 Mouse Moving Action

The system starts with capturing the video frames through the webcam. If a hand is not detected within the frame, the system returns back to its idle state and nothing happens to the mouse. It keeps trying to find hand landmarks and if no hand is detected, the whole thing just sits there until the hand returns.

This system tracks the movement of the user's hand, specifically the real-time position of the raised index finger (or other specified finger). As the finger moves, the system maps finger coordinates to the screen, then moves the cursor, simulating mouse movement.



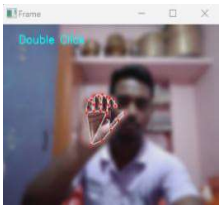


Fig.5 Mouse Movement Action Stopped

Fig.6 Double Click Action

It then monitors the position of the hand. If the hand stays still for a specified period of time or if a "rest" gesture (e.g. a fist or palm down) is detected, the system stops the mouse movement, thus stopping the cursor.

The system is listening for a particular gesture, like a pinch, or the specific configuration of raised fingers. This gesture when recognized generates a double click action which led to the simulation of the action usually done using a mouse that is quickly pressing both left or both right button of mouse clicks twice in succession.

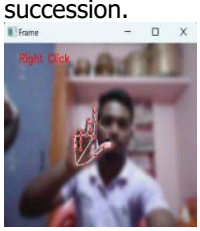


Fig.7 Right Click Action

Fig.8 Left Click Action

If the user makes a certain gesture (for example, raising the thumb and index finger together), the system interprets it as a right click. The right click emulates a physical click of the button that is integrated at the right side of a computer mouse and makes context menus open on the screen or grants added functionality.

A left click action is triggered when the user performs an action like no matter how the finger button is formed: An index finger being raised, or a pinch. This simulates a mouse click event, which is used both for item selection and button activation — that is, like a left click on a mouse.

5 Conclusion

The Virtual Mouse System is a step forward for HCI because it provides a hands-free solution to traditional input devices. The technology integrates real-time gesture recognition and computer vision technologies to provide intuitive and efficient user interface interaction. Although the system performs well across different conditions, future work may expand to include additional gestures, multi-user functionality, and greater specificity and adaptability.

6 Reference:

1. 2001 Rao, A.K., Gordon, A.M. The role of tactile information in pointing movements. Exp. BrainRes. 138,438–445. https://doi.org/10.1007/s002210100717.
2. Masurovsky, A., Chojecki, P., Runde, D., Lafci, M., Przewozny, D., Gaebler, M.,2020. Design aspects of a controller-free hand tracking interaction technique for grab-and-place virtual reality tasks. Multimodal Technol. Interact. 4,91. Retrived from https://doi.org/10.3390/mti4040091.
3. Lira, M., Egito, J.H., Dall'Agnol, P.A., Amodio, D.M., Gonçalves, Ó. F., Boggio, P.S., 2017. The role of skin colour in the rubber hand illusion: Is ownership influenced by skin colour? Sci. Rep. 7, 15745. https://doi.org/10.1038/s41598-017- 16137-3.
4. Danckert, J., Goodale, M. A., 2001. Enhanced performance for visually guided pointing in lower visual field. Exp. Brain Res. 137, 303–308. https://doi.org/10.1007/s002210000653.
5. Carlton, B., 2021. HaptX Teases True-Contact Haptic Gloves For VR And Robotics. VRScout. DOI https://vrscout.com/news/haptx-truecontact-haptic- gloves-vr/ (accessed 3.10.21).
6. J Amutha, S Priyadarsini, S Prasanth, MR Senkumar, M Ramnath, T Jasperline (2024), Oral Lesion Cell Segmentation and Classification using Convolution Neural Network Technique, 2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC), https://doi.org/10.1109/AIC61668.2024.10730889.
7. Rahman, M., Islam, M., & Hossain, M. (2020). Real-Time Hand Gesture Recognition System for Human-Computer Interaction. International Journal of Artificial Intelligence and Applications, 11(3), 123-132. https://doi.org/10.5121/ijai.2020.11309.
8. Patel, D., Sharma, K., & Kaur, G. (2019). Real-Time Hand Gesture Recognition Using OpenCV and Python for Virtual Mouse Applications. International Journal of Computational Vision and Robotics, 9(1), 27-34. https://doi.org/10.1504/IJCVR.2019.095741
9. Ali, S., & Khan, M. (2018). Hand Gesture Recognition for Human-Computer Interaction: A Review. Journal of Computer Science, 14(8), 1186-1195. https://doi.org/10.3844/jcssp.2018.1186.1195
10. Pradhan, P., & Patra, M. (2021). Machine Learning Techniques for Hand Gesture Recognition: A Review. Journal of King Saud University - Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2021.05.012
11. Raghav, S., & Kumar, V. (2020). Real-Time Hand Gesture Recognition Using OpenCV: A Comprehensive Approach. International Journal of Advanced Research in Computer Science, 11(5), 99-104. https://doi.org/10.26483/ijarcs.v11i5.7044
12. Rajesh, K., & Subramanian, S. (2021). Gesture Recognition and Its Applications in Human-Computer Interaction. Journal of Engineering and Technology, 9(2), 85-90. https://doi.org/10.1109/JET.2021.9076517
13. Gupta, S., & Bansal, R. (2022). Gesture-Based Human-Computer Interaction Using Mediapipe and PyAutoGUI. Proceedings of the 5th International Conference on Machine Learning and Intelligent Systems, 44-53. https://doi.org/10.1109/ICMLIS.2022.9406857