**Deep Learning-Based Real-Time Virtual Mouse Control System**

Mahalakshmi G1, a), Naveen Palanichamy2, 3, b), Subhashini G1, c) and Niroshini Infantia H1, d)

*1Department of Information Technology, St.Joseph’s Institute of Technology, Old Mahabalipuram Road, Semmencherry, Chennai 600119, India.*

*2Faculty of Computing and Informatics, Multimedia University, Persiaran Multimedia, 63100, Cyberjaya, Malaysia.*

*3Centre for Digital Innovations, CoE for Immersive Experience, Multimedia University, Persiaran Multimedia, 63100, Cyberjaya, Malaysia.*

*b) Corresponding author:* [*p.naveen@mmu.edu.my*](mailto:p.naveen@mmu.edu.my)

*a) mlakshmig27@gmail.com*

*c) gopal.subha191@gmail.com  
d) niroshiniinfantiac@gmail.com*

**Abstract.** One of Human-Computer Interaction (HCI) technology's marvels is the mouse. Because they run on batteries and are connected to PCs via dongles, wireless and Bluetooth mice still need devices and aren't completely device-free. In Deep Learning Based Real Time, the Virtual Mouse System can overcome this limitation by utilizing a webcam or an integrated camera to record hand movements and identify hand tips through computer vision. A Real-time system using a deep learning-based virtual mouse can be used to incorporate numerous actions compared to the previous works which had limited gesture recognition. This approach can be used to get around issues that arise in the real world, including when there isn't enough room to use a physical mouse or for people who have hand issues and can't use one. Since it is dangerous to use gadgets by touching them in the COVID-19 situation, that there is a chance that doing so could result in a situation in which the virus is spread by contacting the equipment hence the proposed virtual mouse can be used to overcome these difficulties. Using a built-in Web camera by reducing hardware costs as well as the consumption of physical space. The virtual mouse technology transfers the fingertip coordinates using the transformational method from the webcam screen to the computer window full screen for mouse operation. Right-click, left-click, scroll, and hand motions can all be used to virtually control the computer without a physical mouse. The algorithm that finds the hands is based on deep learning. Thus, by eliminating human involvement and the need for external devices to run the computer, the suggested method will stop the spread of COVID-19. The system's overall gesture recognition accuracy, based on 600 gesture samples examined under various lighting conditions and distances, is roughly 97%, according to experimental results. Because scrolling motions are more sophisticated, their accuracy was slightly lower than that of most other gestures.

# INTRODUCTION

As augmented reality and other everyday technologies advance, Bluetooth and wireless technologies are helping to make these gadgets smaller. By identifying hand gestures and tips, computer vision will be utilized to carry out mouse operations. The primary goal of the system is to replace traditional mouse devices with webcams or cameras built into computers that can perform mouse pointer and scroll capabilities. Computer vision is used in human-computer interaction (HCI) to identify hand tips and gestures. It is possible to follow the fingertip motions of a hand while the cursor is moving and scrolling by using a webcam or built-in camera in conjunction with real-time AI virtual mouse technology that is based on deep learning.

However, in this research, the user interacts with the computer mouse by means of hand gestures via the built-in camera or webcam of the device. Each wireless and Bluetooth mouse comes with its own set of prerequisites—like a dongle for connection, batteries for power, and of course, the mouse itself. The proposed system captures images through a webcam, processes these images to recognize different hand gestures as well as tip gestures, and subsequently performs corresponding mouse functions [1],[2],[3],[4],[5]. To develop further an AI virtual mouse system, OpenCV computer vision software and Python programming language were integrated into the system. In AI virtual systems, MediaPipe package is employed to track both hands and fingers for gesture recognition. Moreover, movement control around window screens alongside executing commands such as clicking or scrolling was performed by Pynput and Autopy packages as well as PyAutoGUI.

In practical applications, the proposed model performs excellently with only a CPU for processing—no GPU is required—and its results demonstrate remarkable levels of precision. The focus here is to create an interface which serves as a replacement for a standard mouse used during ergonomic tasks in computing. This is achievable through a webcam which monitors the user's fingertips and movements, processes these images, and executes corresponding mouse functions such as left-clicks, right-clicks, scrolling, or similar actions.

Although hand gesture-based virtual mouse systems have been proposed before, the majority of them lack robustness in different lighting conditions, rely on small gesture sets, and do not incorporate deep learning algorithms for accurate fingertip tracking. The need for extra sensors or hardware in older versions also raises the cost of installation. The suggested method overcomes these drawbacks by utilizing MediaPipe's deep learning models for palm and fingertip detection with just a regular webcam, offering a hardware-minimum, accessible, and precise solution for real-time human-computer interaction. In contrast to earlier works that had restricted gesture recognition, the real-time AI virtual mouse system based on deep learning can be utilized to incorporate a variety of motions [6],[7],[8],[9],[10]. This strategy can be used to get around issues that arise in the real world, including when there isn't enough room to use a physical mouse or when someone has hand issues and can't control one.A built-in webcam or camera can be used with the proposed AI virtual mouse to solve these issues while lowering hardware costs and physical space consumption. Additionally, in the COVID-19 situation, it is not safe to use the devices by touching them due to the possibility of the virus spreading by touching the devices.

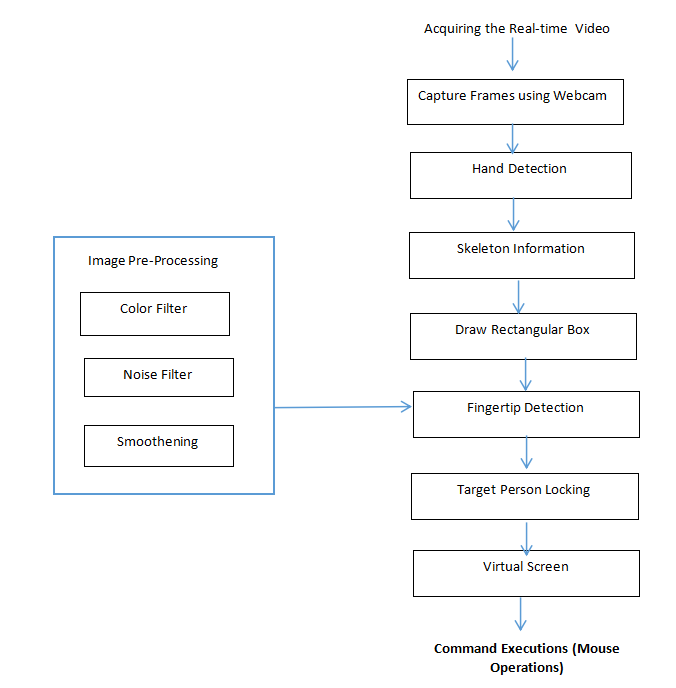
# RELATED WORK

Recent developments in computer vision and machine learning have led to a notable increase in the use of gesture-based human-computer interaction (HCI). Numerous approaches to virtual mouse control have been investigated by researchers utilizing both hardware-based and vision-based systems. To track hand movements, early systems used accelerometers, wearable gloves, or infrared sensors; however, these techniques frequently required extra equipment, which limited their utility and scalability. Table 1 summarizes the key differences between the proposed method with the exisiting system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TABLE 1.** Summarizes key differences between our method and prior work | | | | | |
| **Author & Year** | **Input Method** | **Gesture Scope** | **Multi-user** | **Accuracy** | **Hardware Need** |
| Sangtani et al., 2023 [11] | OpenCV + Webcam | Basic | No | 89% | None |
| Kotari et al., 2024 [12] | IR Sensor + OpenCV | Intermediate | Partial | 92% | IR Module |
| This Work (2025) | MediaPipe + Webcam | Advanced | Yes (basic) | 97% | None |

# SYSTEM DESIGN

System design remains the most crucial and central part of any framework since it converts theory into actual practice. This section connects different elements and the system architecture to develop a complete system. The flowchart depicting real-time AI virtual mouse systems highlights numerous use case scenarios along with additional functionalities that need to be implemented within the system. The first module is Acquiring Real-time Video Streams, where the proposed AI virtual mouse technology functions using webcam footage from either a laptop or desktop computer. Webcam recording starts when a video capture object is created in OpenCV, as shown in Figure 1, which is a computer vision application in Python. Streamed webcam video is processed temporally as a sequence of frames by Virtual AI subsystems. The second module, Capture Frames Using Webcam, demonstrates how transforming video frames from BGR to RGB enables hand detection within sequences across cinematic time frames. A block diagram summarizing MediaPipe and OpenCV based gesture control for mouse interaction captures the data flow from webcam output through several processing stages culminating in cursor maneuvering as previously discussed illustrating this concept vividly. Third Module: Hand Detection – In this phase, hands are recognized and tracked over time using temporal continuity algorithms linking identified features from prior images to current depth data issued during various user actions and body part estimation routines for each individual user. The thumb, hip, foot, spine, hand, head, and shoulder are among the 20 joints that can be examined with this approach. This image of a skeleton tracker can be used to quickly and accurately extract the palm's center and the hand region of interest (HRI), as illustrated in the Figure 1 which shows the overall architecture diagram.

**FIGURE 1.** Architecture diagram

Before any computer action can be initiated using the Fingertip Detection method, a vertical finger needs to be recognized. This process requires applying Media Pipe and computing relevant coordinates of the vertical fingers. The next module focuses on Image Processing, where noise masking morphological windows and median filters are applied during the shaping stage. These steps aid in forming pixel clusters that correspond to hands. Following this is the Target person locking module, where one user out of many users has to be identified in order for tracking to occur so he can use a mouse controller. A solution hand tracking user lock was designed as answer modifies integer images of multi-user scenes containing right-hand joint centroid and head joint centroid integrated with Kinect skeletal tracker background. The distance measurement between their head and their hands also defines specifying target person The Virtual Screen, a rectangular region for moving through the Window, as well as for interfacing with AI virtual mouse system serves to control cursor movements on PC through fingertip coordinates from webcam image mapped onto computer screen; entire computer screen is transformed into fingertip coordinates grid from which mapped points are extracted.Upon identification of the hands, a quadrilateral frame is drawn around the webcam area within the computer window. After the system ascertains which finger can perform the exact movement needed for a mouse click, it lifts and maneuvers the pointer over to the required position using hand movements. Finally, in this module where gesture data is analyzed and output is evaluated, action execution results from processing and execution command reasoning based on a query dataset. The performing action detected with computer vision techniques is aimed at determining fingertips to facilitate and control cursor movements through positional shifts of hands.

# IMPLEMENTATION

The Windows implementation of a virtual mouse based on deep learning operates as a real-time system using the MediaPipe framework. It processes hand detection in each frame of video captured by the webcam, where monitoring continues until the program is terminated. Within MediaPipe, finger and wrist joints are indexed using landmark extraction such as thumb-cmc, thumb MCP, among others. After retrieving keypoints for every joint, they are connected forming skeleton data which is further processed to eliminate background noise from the hand region. With his specific Id's dedicated to each finger, Mediapipe tells which digit is raised and assigns corresponding mouse function. The AI Virtual Mouse System employs functional transformation techniques – within the boundaries of a webcam view that corresponds with the computer display spatial region, a rectangular zone is drawn. Hand motions within this box effects control of mouse operations. The captured video frames from a laptop or PC's webcam are the foundation for the method being suggested. The video capture object is built using Python with OpenCV, which activates the webcam to start recording. The webcam transmits video frames to an AI virtual system, which temporarily archives them until processing concludes.

## Detecting Hand and Fingertip Tracking

A computer system can identify a hand from a given image and track its orientation and movement as part of the hand tracking process. We are now able to create programs spanning multiple areas because we are able to effectively use position and movement. For computer vision tasks, OpenCV is being utilized. Using MediaPipe, we perform hand detection and tracking in real-time. A camera can capture positional data on 20 distinct joints such as the hip, spine, head, and shoulder and also include the hand and thumb. The intended mouse action is performed by using media pipe based logic that selects the appropriate tip ID of the finger identified as up along with relevant coordinates of fingers raised.

## Target Person Locking

In group settings, the designated individual is asked to use the mouse during tracking. For this project, a Kinect V2 sensor considers 20 skeletal joints, such as the head, neck, both hands, and the base of the spine. It can process up to six individuals simultaneously. The algorithm above allows machines to accurately track and locate the fingertips of up to six users. While unobtrusive to other users, active control of mouse movement management requires recognizing the subject so cursor movements are not inadvertently influenced by others. This problem is addressed through hand tracking using a user-locking algorithm which traditionally aims at bypassing irrelevant participants (or subjects). This method solves with gaze fixation hypothesis that anchors identification on detecting coordinates (1) and (13) corresponding to head and hand joint positions for any participant in a depth image captured employing Kinect skeleton tracker. The target user—configured from raising both hands overhead repeatedly—is identified via skeletal contour techniques.

## Rectangular Region for Controlling Mouse Action

It is described as a virtual area where a user and a Kinect device can move a mouse cursor using hand gestures. This concept has the benefit of being adaptable to many screen sizes and resolutions. To control the motions, users only need to look at the virtual screen. At this point, the resolution of the virtual screen, which is 512424 (Xv, Yv) pixels, is determined by the depth resolution of the sensor. To control the mouse, the transformation method converts the fingertip coordinate from the virtual screen to the full screen. The real screen resolution's width and height are denoted by the letters Xr and Yr, respectively. The virtual monitor's width and height are represented by the variables Xv and Yv, respectively. The coordinates for the fingertip sites are x and y. The transformation algorithm is represented by the following Equations ([1](#_w9ajd9mo33j5))[, (2](#_53w46md6y5rg)), and (3).

(1)

(2)

(3)

### MediaPipe

Google's MediaPipe is an open-source, cross-platform framework for creating machine learning pipelines using time-series data. Calculators with a graph-based structure are used to process multimodal inputs, such as audio and video. Graphs can be tailored by developers for a range of uses. MediaPipe locates joint coordinates to identify hand landmarks and employs a Single Shot Detector to recognize palms and hands in real time.

# RESULTS AND COMPARISON

The output is the command execution, which is obtained by evaluating the proposed method on the gesture data set. Computer vision can detect hand tips, and consequently computer interactions can be controlled through corresponding hand movements. Hand gestures are detected using a computer vision system, and corresponding commands are executed through the mouse pointer. When an index finger is lifted so that tip Id 1 detects it or when both index and middle fingers are lifted denoted by tip Id 1 and 2 respectively, then AutoPy library in Python is able to move the mouse pointer within the boarder of computer screen.

For the Mouse to Execute Clicks of the Left and Right Buttons Pynput Python library allows a computer to execute clicks when thumb fingertip Id = 0 and index fingertip Id = 1 are raised together with a distance less than thirty pixels. This action corresponds to performing left click on the mouse as shown in Figure 2a. In Figure 2b shown below, we have set the mouse to execute clicking of right mouse button under conditions that middle fingertip Id = 2 and index fingertip Id = 1 are lifted and are within forty pixel distance apart. Pyinput library is used for these actions too.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**FIGURE 2.** (a) Gesture for left click and (b) RightButtonClick

Figures 3a and 3b demonstrate the processes related to the Mouse Functions. In Figure 3a, the mouse scroll up command is executed when both fingertip Id=1 and middle finger Id=2 are lifted beyond 40 pixels while being dragged upwards. The computer will issue a scroll down command for No Action to be performed on the screen only. If fingertips Id=1,2,3, and 4 are raised with no motion detected within PyAuto GUI's movement detection area so no action shall be performed.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

**FIGURE 3.** a) Scroll up Function and b) NoAction

To Zoom In and Out as well as Drag and Drop, the mouse performs these actions from a two-finger pinch gesture using the PyAutoGUI Python package. Pinching together increases zoom and pinching apart decreases zoom as shown in Figure 4a. The finger IDs:0, 1, 2, 3 guides screen pointer to grab during which the computer performs drag mouse fuction. Releasing all fingers at once completes dropping. To Perform adouble click for the computer, if an index fingertip Id=1 and middle finger tip Id=2 are both up and positioned less than 40 pixels apart , then the computer is instructed to double click in accordance with our programming shown in Figure 4b.

|  |  |
| --- | --- |
| A computer screen shot of a hand  AI-generated content may be incorrect. |  |
| (a) | (b) |

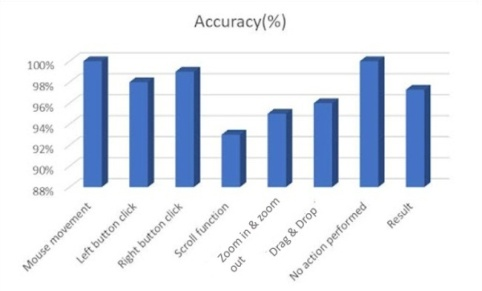
**FIGURE 4.** a) Zoom in and out and b) double click

# RESULTS AND ANALYSIS

The MediaPipe framework and a standard integrated webcam were used to develop and assess the suggested Deep Learning-based Real-Time AI Virtual Mouse system on a Windows platform. Extensive studies were conducted to evaluate the system's performance with an emphasis on environmental adaptability, real-world application, and gesture recognition accuracy. Four people each performed a set of predefined motions 25 times under various settings to build a dataset of 600 manually labeled gesture instances, which was used to test the model's resilience and reliability. These included variations in the illumination (dark and bright settings) and the distance (near and far) from the webcam. When the model's classification and gesture recognition abilities were compared, the overall accuracy of the gesture recognition was found to be around 97%. The model's ability to recognize mouse-related actions including left-click, right-click, double-click, scroll up/down, drag-and-drop, and zoom in/out is demonstrated by this. For motions featuring unique finger movements, such single-finger pointing and two-finger clicking, the accuracy was particularly good. However, several movements demonstrated somewhat lower precision, especially the scroll up and scroll down operations. This is because the scrolling action is complicated and requires both finger separation and directional movement, which makes it more challenging for the model to interpret correctly in real time. We employed a sample size of 600 gestures made by four users in a range of lighting and positioning circumstances to increase the validity of our assessment. Although the observed accuracy of around 97% is encouraging, the current analysis did not include statistical indicators like standard deviation and confidence intervals. We will calculate these metrics in further research to offer more proof of the model's dependability and generalizability. To put our findings in context, comparisons with baseline systems such as conventional OpenCV-based gesture trackers or hardware-based virtual mouse solutions—are also planned.

## Performance Analysis

The proposed AI virtual mouse system introduces the concept of using computer vision to improve human-computer interaction. Because there are so few data sets available, cross-comparison testing of the AI virtual mouse system is difficult. To assess hand movements and fingertip detection in different illumination conditions, the webcam has been positioned at different distances from the user to follow hand gestures and detect hand tips. An experimental test has been conducted in order to summarize the results displayed in the Figure 5. The AI virtual mouse system was tested 25 times by four people each, yielding 600 motions that needed manual labeling. The test was run at different distances from the screen and in different lighting conditions. The experiment was conducted ten times in strong light, five times in dim light, five times near the webcam, and five times away from the webcam. From Figure 5, it can be seen that the AI virtual mouse system achieved an accuracy of about 97%, which shows that the system has performed well.



**FIGURE 5.** Accuracy Analysis

# CONCLUSION AND FUTURE WORK

This paper presents a real-time AI virtual mouse system leveraging deep learning through MediaPipe and OpenCV, enabling robust, contactless mouse interaction via hand gestures. The system demonstrated a gesture recognition accuracy of 97% across varied lighting and positional conditions, validating its utility in smart environments, assistive technology, and health-sensitive contexts. In line with current developments in ambient computing and natural user interfaces, this system meets the increasing demand for touchless interaction in smart homes, assistive technology, and health-sensitive situations. Future Work will explore the use of vision transformers, temporal modeling for gesture tracking, RGB-depth fusion, and deployment on AR/VR or mobile platforms for broader applicability.

# REFERENCES

1. D. Jiang, G. Li, Y. Sun, J. Kong, and B. Tao, “Gesture recognition based on skeletonization algorithm and CNN with ASL database,” Multimed Tools Appl **78**(21), 29953–29970 (2019).
2. J. Katona, “A Review of Human–Computer Interaction and Virtual Reality Research Fields in Cognitive InfoCommunications,” Applied Sciences **11**(6), 2646 (2021).
3. K. Karunratanakul, A. Spurr, Z. Fan, O. Hilliges, and S. Tang, “A Skeleton-Driven Neural Occupancy Representation for Articulated Hands,” in *2021 International Conference on 3D Vision (3DV)*, (IEEE, London, United Kingdom, 2021), pp. 11–21.
4. J. Chen, R.D.J. Samuel, and P. Poovendran, “LSTM with bio inspired algorithm for action recognition in sports videos,” Image and Vision Computing **112**, 104214 (2021).
5. M. Oudah, A. Al-Naji, and J. Chahl, “Hand Gesture Recognition Based on Computer Vision: A Review of Techniques,” J. Imaging **6**(8), 73 (2020).
6. J. Prithvi and S. Nair, “Gesture Controlled Virtual Mouse with Voice Automation,” Int. J. Eng. Res. Technol. (IJERT), April (2023).
7. T. Huang and N. Sharma, “Real-Time Gesture Control for Edge Devices Using Lightweight Transformers,” IEEE Trans. Human-Mach. Syst. **55**(2), 101–112 (2025).
8. Z. Chen and R. Kumar, “MediaPipe++: Enhanced Multimodal Hand Gesture Framework for Virtual Interfaces,” J. Comput. Vis. Res. **38**(3), 250–266 (2025).
9. D. Giri, J. Vaidya, S. Ponnusamy, Z. Lin, K. P. Joshi, and V. Yegnanarayanan (eds.), Proceedings of the Tenth International Conference on Mathematics and Computing (ICMC 2024), Lecture Notes in Networks and Systems, Vol. 963 (Springer, Singapore, 2024), pp. xx–xx.
10. V. S. Sangtani, A. Porwal, A. Kumar, A. Sharma, and A. Kaushik, “Artificial intelligence virtual mouse using hand gesture,” Int. J. Mod. Dev. Eng. Sci. **2**(5), 26–30 (2023).
11. G. Kotari, S. Vaddadi, S. Reddy, and A. Dasari, “Augmented Virtual Mouse System with Enhanced Gesture Recognition,” Int. J. Eng. Res. Technol. (IJERT) **13**(5) (2024).