Virtual Mouse and Keyboard Through Gesture Control

Prof. G.B Chavan[1], Omkar Bagade[2], Dada Choudhar[3], Ahad Zari[4]
Prof. *Chavan Ganesh Baban, SVPM COE MALEGAON(BK), BARAMATI.*
Bagade Omkar Ravindra, SVPM COE MALEGAON(BK), BARAMATI.
Choudhar Dada Ramdas, SVPM COE MALEGAON(BK), BARAMATI.
Zari Abdul Ahad Sameer, SVPM COE MALEGAON(BK), BARAMATI.

Abstract: This paper presents a gesture recognition-based input system combining virtual mouse and keyboard functionality using advanced hand and eye tracking. Designed for contactless interaction, this system uses real-time webcam feeds processed through computer vision algorithms (MediaPipe and Haar Cascade). It allows precise cursor control via eye gestures and efficient keyboard input via hand tracking. The integration of Kalman filtering, PyAutoGUI, and Pygame ensures smooth operation. This environmentally sustainable system is tailored for hygienic, accessible use cases like healthcare, education, and public terminals.

Index Terms - Gesture recognition, virtual mouse, virtual keyboard, human-computer interaction, MediaPipe, Haar cascade, Kalman filter.

I. INTRODUCTION

In recent years, advancements in Human-Computer Interaction (HCI) have significantly transformed the way users engage with digital systems. A notable trend in this evolution is the shift from traditional physical input devices, such as keyboards and mice, to more intuitive and natural gesture-based methods. These conventional input tools, while effective in many settings, present limitations in environments where physical contact is either impractical or undesirable. For instance, in sterile environments like hospitals, laboratories, or clean rooms, maintaining hygiene is a top priority, and frequent touching of surfaces can pose contamination risks. Similarly, in public or shared spaces, concerns over the spread of infectious diseases—especially in light of recent global pandemics—have accelerated the demand for contactless interaction methods.

In response to these challenges, this research explores the development of a virtual input system that leverages webcam-based eye and hand gesture recognition. By using standard camera hardware combined with advanced computer vision and machine learning algorithms, the system can accurately detect and interpret user gestures in real-time. This allows users to interact with digital interfaces without any physical contact, offering a seamless and hygienic alternative to traditional input methods.

Moreover, beyond hygiene-related benefits, this gesture-based system enhances accessibility for individuals with physical disabilities or mobility impairments, enabling them to interact with computers in a more inclusive and user-friendly manner. The implementation of such touchless technology introduces a new dimension of control that is not only innovative but also practical in a wide range of modern applications, including healthcare, assistive technology, smart homes, and more.

Overall, this research aims to contribute to the growing field of gesture-based HCI by demonstrating how webcam-driven virtual input systems can provide effective, real-time control while addressing current concerns around hygiene and accessibility.

II. LITERATURE SURVEY

Gesture-controlled interfaces have been a prominent area of research within Human-Computer Interaction (HCI), with numerous efforts aimed at creating more natural and intuitive ways for users to interact with digital systems.

One of the foundational contributions to this field is the *Sixth Sense Technology* developed by Sadhana Rao. This system introduced the concept of wearable gestural input, using a combination of a camera and small projector worn around the neck to track hand movements and display interactive content on physical surfaces. This innovation demonstrated the potential of merging the physical and digital worlds through

gesture-based interaction and set the stage for further exploration of touchless control technologies.

In addition to wearable solutions, researchers like Game and Mahajan have explored gesture-based systems that rely on vision-based tracking for virtual interaction. Their work focused on detecting hand movements and interpreting them as input commands, paving the way for more refined and application-specific gesture recognition systems. These studies emphasized the importance of real-time performance and accuracy, along with the need for a responsive interface that can adapt to various user behaviors and environments.

Building upon these foundational studies, our research introduces a more integrated and multimodal approach to gesture recognition. Unlike earlier works that focused primarily on hand gestures or required wearable devices, our system combines both eye movement and hand gestures to create a more comprehensive virtual input interface. This is achieved through the use of Python and MediaPipe, a framework by Google that offers robust and efficient real-time tracking of facial and hand landmarks.

By fusing multiple gesture modalities and implementing them at a system level, our work enhances the interaction experience and improves the flexibility of touchless control. This approach not only provides a more natural form of interaction but also addresses usability in accessibility-focused and hygiene-sensitive scenarios—areas where traditional systems fall short.

In summary, while past research has laid a strong foundation for gesture-controlled systems, our work advances the field by integrating multiple input sources and modern real-time processing tools to offer a more versatile, practical, and accessible solution.

III. PROPOSED SYSTEM

- Eye-controlled Virtual Mouse: Using MediaPipe FaceMesh and Haar cascade classifiers, it tracks the eye position to move the mouse pointer. Blinks are interpreted as clicks.
- Gesture-based Virtual Keyboard: Implemented using MediaPipe Hands and Pygame. Fingertip proximity (index and thumb) triggers virtual key presses. A three-finger gesture opens applications (e.g., Zerodha).
- Unified UI Launcher: Built in Tkinter for switching between modules.

IV. SYSTEM ARCHITECTURE

The architecture of the system is composed of interconnected modules that handle video input, gesture detection, interpretation, and event execution. Below is a step-by-step breakdown:

1. Webcam Input: Captures real-time video stream.
2. Preprocessing Unit: Converts frame to grayscale or RGB depending on the module (Haar Cascade or MediaPipe).
3. Detection Unit:
   - Uses Haar Cascade to detect face and eyes.
   - Uses MediaPipe to detect and track hands.
4. Landmark Extraction: Specific facial or hand landmarks are extracted using MediaPipe's built-in models.
5. Gesture Interpretation Module:
   - Eye gestures like blinking are interpreted as mouse clicks.
   - Hand gestures like pinching are mapped to key presses.
6. Smoothing Filter (Kalman Filter): Applies smoothing to reduce jitter in cursor movement.
7. Action Executor:
   - Uses PyAutoGUI to simulate mouse and keyboard events.
8. Graphical Interface (Tkinter): Allows users to select between virtual mouse and keyboard modes.

Each module communicates through Python scripts and real-time data streams, forming a low-latency, modular, and extensible system for touchless interaction.

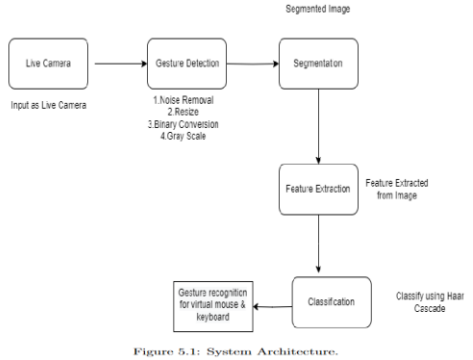


Figure 5.1: System Architecture.

V.ALGORITHM AND METHODOLOGY

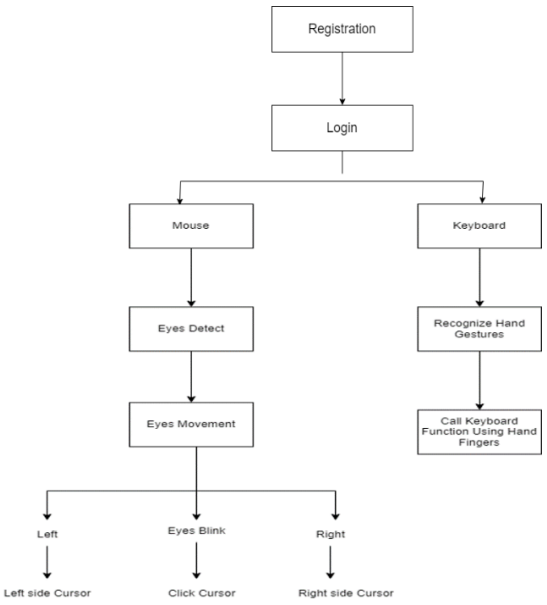
A. Haar Cascade Algorithm Used for face and eye detection. Based on rectangular Haar-like features and integral images. Classifies patterns by computing the difference of intensities between adjacent rectangular regions. It uses multiple trained stages for efficient real-time detection.

B. MediaPipe Framework MediaPipe Hands identifies 21 hand landmarks, including fingertips and joints. FaceMesh tracks 468 facial landmarks. The eye landmark at ID 474 is used for mouse movement, and the distance between IDs 145 and 159 detects blinks.

C. Kalman Filter Used to predict cursor motion and smooth noisy landmark data. Equations:
State: X = [x, y, dx, dy]
Measurement: Z = [x, y]
Prediction: X' = A*X*
Correction: X = X' + K(Z - H*X')

D. Mathematical Model For Haar Feature: F = ∑(Pixels in White Region) - ∑(Pixels in Black Region)
Integral Image (Iint) helps compute F efficiently: Iint(x, y) = I(x, y) + Iint(x-1, y) + Iint(x, y-1) - Iint(x-1, y-1)
Gesture Decision Function: If |Index_tip - Thumb_tip| < ε, then Key Press

VI. FLOW CHART



VII. HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE:

- CPU: Intel i3 and above
- Camera: HD webcam or depth camera
- RAM: 4 GB
- Storage: 256–512 GB
- Display: Standard HD display

SOFTWARE:

- OS: Windows 10
- IDEs: PyCharm, Spyder, **VS Code**
- Language: Python
- Libraries: MediaPipe, OpenCV, PyAutoGUI, Pygame, Tkinter

VIII. RESULT AND OBSERVATIONS

- Real-time tracking achieved ~90% gesture accuracy.
- Smooth pointer movement with Kalman filtering.
- Responsive typing on virtual keyboard.
- Successfully opened a web app using gesture trigger.
- User-friendly performance on mid-tier hardware.

IX. CONCLUSION

This research delivers an effective, contact-free human-computer interface using combined hand and eye gesture recognition. The modular, real-time system serves as a cost-efficient, hygienic alternative to traditional input. It holds potential for future expansion into augmented reality and remote operation platforms.

ACKNOWLEDGMENT

We sincerely thank Prof. G.B. Chavan for his guidance, encouragement, and constructive feedback during this project.

REFERENCE

[1] S. Sadhana Rao, "Sixth Sense Technology," ICCCI–2010, pp.336–339.
[2] Game P. M., Mahajan A.R., “A gestural user interface to Interact with computer system,” IJSAT Vol. II, Issue I, 2011.
[3] Christy, A., et al. “AI based automatic decelerating vehicle control system,” IJATCSE, Vol. 8, Issue. 6, 2019.
[4] G. M. Gandhi, Salvi, “AI Integrated Blockchain for Autonomous Cars,” ICONSTEM, 2019.
[5] Praveena M.D.A., et al., “Smart attendance monitoring using OpenCV,” JCTN, Vol. 16, 2019.
[6] M.S. Roobini, Dr. M. Lakshmi, “Classification of Diabetes using ML,” IJITEE, Vol. 8, 2019.