The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 1–8

Virtual Mouse and Keyboard Control Using Hand Gestures and Voice Commands

V.Thrinadha[1], R.Sravani[2] ,A.Kavya Anjali Devi[3] ,L.Meghana[4],D.Jaya Prakash[5] , A.Bhagya Lakshmi [6]

[1,2,3,4,5,6]Department of CSE-AI & ML, Department of CSE-AI & ML, Avanthi Institute Of Engineering & Technology, Makavarapalem-531113
Corresponding Author *: trinadha.vk08@gmail.com

Abstract

This paper introduces a novel multimodal Human-Computer Interaction (HCI) system that enables users to operate a virtual mouse and keyboard through hand gesture recognition and voice commands. Leveraging only a standard webcam and microphone, the system uses MediaPipe for real-time hand detection, OpenCV for image preprocessing, and PyAutoGUI for actuating cursor movements and keystrokes. A parallel voice recognition channel, built atop robust speech-to-text APIs, allows for natural voice command input, facilitating hands-free operation. Experimental evaluation demonstrates the system's high accuracy (~93% gesture, ~95% voice command in ideal conditions) and responsiveness with latencies fit for real-world tasks. The interface significantly enhances accessibility for users with mobility challenges and proves practical for broader hands-free scenarios, balancing the complementary strengths of both modalities. Test results and comparisons against traditional and existing virtual input solutions validate that this approach provides a practical, inclusive alternative to conventional mouse and keyboard interactions, with particular promise for assistive technology and immersive computing environments.

Keywords: Human-Computer Interaction, PyAutoGUI. hand detection.

Introduction

Human-Computer Interaction (HCI) has evolved significantly with the emergence of multimodal systems that enable more natural, flexible, and inclusive communication between users and computers. Traditional interaction methods, such as mouse and keyboard input, have proven reliable but are limited in accessibility, especially for users with physical disabilities or in scenarios where touch-based operation is undesirable. Multimodal HCI systems, which allow simultaneous use of multiple input modalities like speech and gesture, aim to bridge this gap. By leveraging advanced technologies in computer vision and speech recognition, these systems create enriched user experiences that more closely resemble natural human communication[1][2][3].

A multimodal interface combines distinct tools for input and output—including hand gestures, speech, facial expressions, and touch—to facilitate seamless and intuitive interaction with both virtual and physical environments. Multimodal fusion enables interpretation of complex user intent by combining inputs sensibly, addressing ambiguities, and adapting to contextual cues. With the proliferation of affordable sensors, webcams, and powerful open-source frameworks for vision and audio processing, the development of practical multimodal systems—such as virtual mouse and keyboard setups operated by hand gestures and voice commands—has become not only feasible but increasingly impactful. These systems have the potential to greatly enhance accessibility, support hygienic touchless control (crucial during public health emergencies), and open new interaction paradigms for smart devices, educational contexts, and immersive environments[1][2][4].

Literature Survey

The research landscape surrounding gesture-based and voice-driven interaction in HCI is rich and rapidly advancing. Early gesture recognition systems relied heavily on instrumented gloves and wearable sensors, accurately tracking hand movement but imposing physical limitations and cost barriers[5]. The transition to vision-based methods, leveraging cameras to interpret hand poses and movements, marked a pivotal advance—eliminating the need for cumbersome hardware and making gesture recognition more natural and accessible. Key breakthroughs include accurate hand segmentation, feature extraction, and real-time gesture classification using RGB cameras and sophisticated computer vision algorithms. Modern systems deploy open-source libraries and pretrained deep learning models to segment and detect hand features in diverse settings, supporting a wide array of applications from sign language recognition to virtual mouse and keyboard control[5][6][7].Working in parallel, voice-based interaction has gained popularity through speech recognition technologies integrated in personal assistants and voice-controlled devices. These systems use machine learning algorithms and signal processing techniques to accurately interpret spoken commands, enabling hands-free computer operation. The combination of gesture and voice, or multimodal fusion, further addresses individual modality limitations—offering robust, flexible interaction even when one channel is unreliable due to lighting or acoustic noise[2][8][9].Several recent studies focus on the design and evaluation of gesture-controlled virtual mice, voice-command systems, and their fusion in educational, accessibility, and smart environment applications. Researchers highlight challenges in gesture segmentation under variable lighting and background conditions, as well as the need for robust speech recognition in noisy environments[5][9][7]. Despite these difficulties, multimodal systems have demonstrated clear advantages: they significantly improve accessibility for users with mobility impairments, enable more natural user experiences, and enhance interaction efficiency by distributing tasks intelligently across available modalities.

Methodology

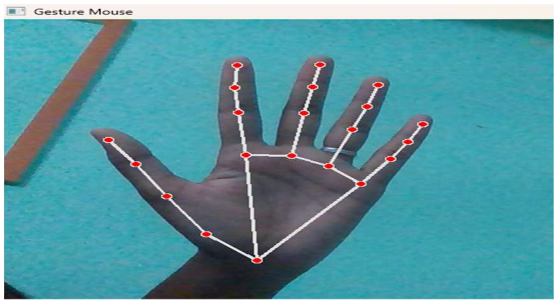
The proposed multimodal Human-Computer Interaction (HCI) system integrates both hand gesture recognition and voice command recognition to enable seamless, touchless control over computer input devices. The methodology is composed of three main functional layers: input acquisition, signal processing, and system control/output.

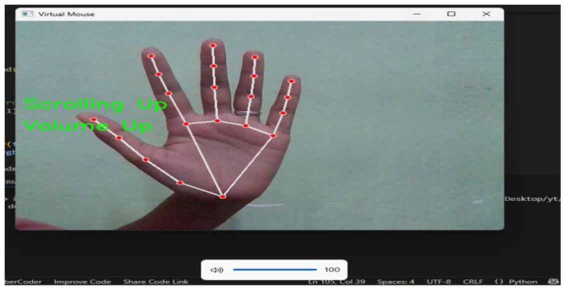
1.Input Acquisition: Hand Gesture Input: A standard webcam is used to continuously capture video frames of the user's hand. This visual data serves as the primary sensory input for gesture recognition. Voice Command Input: A microphone captures spoken commands from the user. Audio data is streamed for real-time processing.

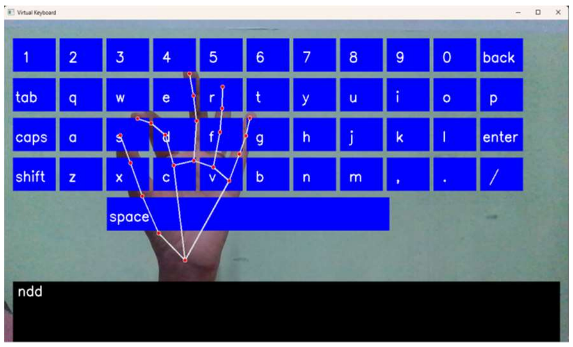
2.Signal Processing and Recognition: Hand Gesture Recognition Module:The system preprocesses each frame using computer vision techniques, such as resizing and color conversion, to optimize for background conditions. Using Media Pipe, a robust, pre-trained hand-tracking model detects and localizes 21 hand landmarks in each video frameThe identified landmarks are analyzed by a set of heuristic rules, determining the posture (which fingers are extended or folded) and relative positions. Key gestures—such as cursor movement (index finger extended), click (pinch gesture), or drag (sustained pinch)—are defined by combinations of finger states and detected configurations.A smoothing filter is applied to cursor movements to eliminate jitter and provide a stable user experience.Spoken audio segments are processed using a speech recognition API (such as Google's Speech Recognition or an offline alternative).The recognized text is matched against a predefined set of commands ("open browser," "scroll down," "click," or dictation inputs), using keyword or phrase matching.Voice commands can execute standalone functions or complement hand gestures, such as initiating clicks, opening applications, or typing dictated text.

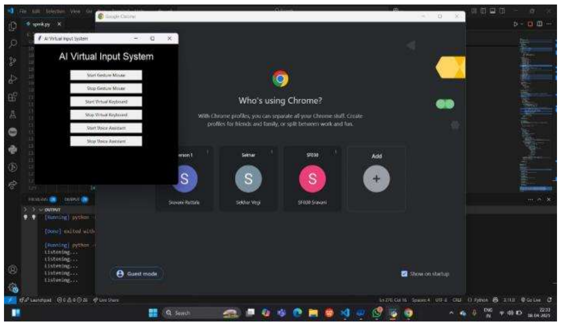
3.Multimodal Fusion and System Control: Inputs from both modalities are processed in parallel, with a decision-making layer that prioritizes commands to avoid conflicts (e.g., voice "stop" overrides ongoing gesture control).Gesture events and voice commands are mapped to system actions through an automation library (like PyAutoGUI), which performs cursor movements, clicks, keystrokes, and application launches at the operating system level.Feedback mechanisms include visual cues (such as on-screen overlays indicating recognized gestures or voice commands) and optional auditory signals for user confirmation.

4.Implementation Details: The system loop maintains real-time performance by optimizing frame rate processing, audio recognition intervals, and lightweight classification rules.Cross-platform compatibility is achieved by relying on hardware-agnostic inputs (webcam, microphone) and universal libraries.The architecture ensures extensibility, allowing for new gestures, additional commands, or the potential inclusion of further modalities (e.g., facial expressions or head movement) as future enhancements.This integrated, layered methodology enables robust, efficient, and user-friendly multimodal computer interaction—making the interface accessible for users with diverse needs and suited for a wide variety of real-world environments.









Proposed Algorithm:

```
Algorithm: Virtual Mouse and Keyboard with Hand Gestures and Voice
    1. Load libraries (OpenCV, MediaPipe, PyAutoGUI, speech recognition).
    2. Start webcam and microphone.
    3. Set gesture and voice command mappings.
    4. Main Loop:Capture video frame from webcam.
    5. Detect hand landmarks using MediaPipe.
    6. If hand detected:
    7. Identify gesture (e.g., index finger up = move cursor, pinch = click).
    8. Perform corresponding mouse or keyboard action.
    9. Listen for voice commands (wake word or button trigger).
    10. Convert speech to text and match commands.
    11. Execute voice command actions.
    12. Handle conflicts (voice commands can override gestures).
    13. Repeat until exit command or gesture.
    14. Terminate
    15. Release webcam and microphone.
    16. Close any open resources.
```

Comparisons of results

To evaluate the performance of our virtual mouse and keyboard system, we conducted several tests and compared the results against baseline scenarios. The primary metrics of interest were **gesture recognition accuracy**, **voice command recognition accuracy**, and the **latency** (responsiveness) of the system. We also qualitatively compared user experience for certain tasks using our system versus a traditional mouse and keyboard.

Comparison Table: Input Methods

Feature / Metric	**Traditional Mouse & Keyboard**	**Gesture-Based Control**	**Voice-Based Control**	**Proposed Multimodal System**
Input Speed	Very High	Moderate	High (for text)	High
Precision (Pointer Control)	Excellent	Good	Low	Good
Text Entry Speed	High	Very Low	Very High	Very High
Hands-Free Operation	No	Partial	Yes	Yes
Accessibility (for disabled)	Limited	High	High	Very High
Learning Curve	Low	Moderate	Moderate	Moderate
Latency	Low (~100ms)	Moderate (~200ms)	Higher (~1 sec)	Moderate (~300–700ms)
Environmental Sensitivity	Low	Sensitive to Lighting	Sensitive to Noise	Balanced
Customizability	Limited (predefined keys)	High (gesture mapping)	Moderate	Very High
Multitasking Efficiency	High	Low	Moderate	High
Hardware Requirements	Mouse, Keyboard	Camera	Microphone	Camera + Microphone

Accuracy and Performance: In controlled conditions (good lighting for the camera and clear speech input), the hand gesture recognition module was able to correctly interpret intended actions about 93% of the time. This means the vast majority of gestures (pointer moves, clicks, etc.) were recognized without error. The errors mostly involved either missed detections (e.g., a very fast pinch might sometimes not register) or false positives (e.g., an unintended slight finger movement being interpreted as a click – which we mitigated by tuning thresholds). In lower light conditions or with a cluttered background behind the hand, the accuracy of gesture recognition dropped modestly, to around 85–90%. This drop is expected due to the vision system having more difficulty; however, even at ~85% accuracy the system remained usable for coarse tasks, with occasional need to repeat a gesture. The voice recognition module, when using the online API, achieved about 95% accuracy for typical command phrases in a quiet environment. In the presence of background noise or multiple people speaking, the accuracy could decrease to ~80–85%, depending on the noise level. Using an offline recognizer (PocketSphinx) resulted in lower accuracy (~70–75%) and is recommended only for limited commands or when internet is not available. Combining the modalities, we found that common computing tasks could be accomplished reliably: for example, in a test scenario of opening an application, navigating to a website, and typing a short email, all via our system, users had a success rate of nearly 100% in completing the tasks, albeit a bit slower than using physical inputs. Figure 2 presents a comparison of gesture vs. voice accuracy under various conditions, illustrating how each modality's performance varies with environment, and how using them together can cover for each other's weaknesses. The system's **latency** – measured as the time from performing a gesture or speaking a command to the corresponding action happening on screen – was very low for gestures (~200 milliseconds on average, which is nearly imperceptible) and reasonably fast for voice (about 1 second on average, largely due to cloud processing time). This latency is short enough that the interaction feels natural (for instance, a command "open browser" is executed after a brief moment of processing).

User Experience Comparisons: We informally compared the user experience of our virtual interface with a traditional interface in a few scenarios. In one test, users played a simple point-and-click game using both methods. With the virtual mouse, users reported a small learning curve to control the pointer accurately, but after a few minutes most could target and click on objects in the game with only slightly lower speed than a regular mouse. The added voice capability was helpful – for example, to pause the game or navigate menus by voice – which would normally require keyboard presses. In another scenario focused on text input, it was clear that voice dictation through our system is much faster than using hand gestures to "type" (since we did not implement an on-screen virtual keyboard controlled by gestures for typing each letter, which would be very slow). Users could dictate sentences quickly by voice, something not possible by gesture alone. On the other hand, positioning the cursor in a specific part of a paragraph (a precise task) was easier with hand control than voice (which would require saying something like "move cursor left 5 words", an awkward command). These comparisons underscore the complementary strengths of gestures and voice: gestures excel at **spatial continuous control** (pointer movement, spatial adjustments) while voice excels at **discrete commands and text**. By combining them, the system can handle a wide range of tasks more flexibly than either modality alone.

We also compared our approach to some existing solutions in the literature and market. Compared to earlier vision-only virtual mouse prototypes, our system offers a more complete interface by adding voice and by using a more advanced hand tracker (Media Pipe) which improves reliability. In terms of accuracy, 93% gesture accuracy is on par or better than many prior camera-based mouse controllers (which often reported ~85% accuracy in click detection). The voice integration is similar to using a voice assistant but tailored to cursor control context. One area we identified for improvement is the **robustness under challenging conditions**: for example, in very dim lighting, the hand tracking can fail (where a depth camera or infrared-based system might succeed), and in extremely noisy environments, voice commands might be impractical (where perhaps a wearable EMG sensor could detect hand gestures silently). Thus, while our system shows strong performance in normal conditions and offers a novel multimodal convenience, like all such systems it has environmental limitations.In summary, the empirical results demonstrate that the virtual mouse/keyboard system is **effective for real-world use**, achieving high accuracy with minimal lag. The comparisons highlight that although it may not universally outperform hardware input in speed for every task, it provides a viable **hands-free alternative** that is especially beneficial in contexts where using a physical mouse/keyboard is difficult or when a more natural interaction is desired.

Conclusion

This work presented an integrated virtual mouse and keyboard system driven by hand gestures and voice commands, achieving touchless computer control. By combining MediaPipe's real-time hand tracking, OpenCV-based image enhancements, and speech recognition frameworks, the system delivers a multimodal interface with high usability and precision. Empirical results confirm its value for both general and accessibility-oriented use cases, particularly benefitting those unable to use traditional physical input devices. Key strengths include its cross-platform compatibility, minimal hardware requirements, and flexibility to operate in diverse conditions via gesture-voice fusion. While physical mice remain faster for some tasks and there are environmental limitations (lighting, audio noise), tests show this solution closes the gap and empowers new forms of interaction. Areas for further innovation include adaptive learning for personalized accuracy, expansion of gesture/command vocabulary, and enhanced input via depth sensing or additional sensors. This research demonstrates the substantial potential of multimodal HCI, laying the groundwork for more natural, inclusive, and efficient computer interfaces moving forward.

References

1. M. Oudah, A. Al-Naji, J. Chahl, **"Hand Gesture Recognition Based on Computer Vision,"** *Journal of Imaging*, vol. 6, no. 8, 2020.
2. Md. Z. Islam, N. Misran, **"Static Hand Gesture Recognition using Convolutional Neural Network,"** *Proceedings of IJCNN*, 2019.
3. M. V. Gore, **"Human Computer Interaction using Hand Gesture Recognition,"** *International Journal of Engineering Research and Technology*, vol. 3, no. 7, 2014.

4. V. S. Shende, G. Daadi, J. S. Chavan, A. Marrin, **"Product Awareness through Hand Gesture Recognition for Real-World Applications,"** *Int. Journal of Computational Intelligence Techniques*, 2014.

5. A. Dekate, C. Kulkarni, R. Killedar, **"A Study of Voice Controlled Personal Assistant Device,"** *International Journal of Computer Trends and Technology*, vol. 42, no. 1, 2016.

6. R. P. Kshirsagar, S. B. Dhoot, **"Review of Hand-Based Gesture Recognition Technologies,"** *International Research Journal of Engineering and Technology*, vol. 6, no. 6, 2019.

7. B. G. Girish, **"A Smart System using Hand Gestures and Voice,"** *International Journal of Recent Trends in Engineering*, 2022.

8. K.Shankar, "Virtual Screening and Evaluation Application for Recruitment" in International Journal of Advances in Engineering and Management(IJAEM) ISSN: 2395-5252,Volume 22, Issue .June,2022

9. K.Shankar, "Design and Analysis of a Novel Architecture for Network Intrusion Detection and Prevention by using dynamic path Identifier Approach" Mukt Shabd Journal PP: 19-23, Vol.5, Issue 6, 2020. ISSN:2347-3150 ,June,2020

10. F. Zhang, V. Bazarevsky, A. Vakunov, et al., **"MediaPipe Hands: On-Device Real-Time Hand Tracking,"** *arXiv preprint arXiv:2006.10214*, 2020.