iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

MULTIMODAL HUMAN-COMPUTER INTERACTION USING HAND, EYE AND VOICE GESTURES

Badanapalli Aparna[1] (MCA student),
Nagaraju Vassey[2] (Assistant Professor),
Manne Naga VJ Manikanth[3] (Guest Faculty)
Department of Information Technology and Computer Applications
Andhra University College of Engineering, Visakhapatnam, AP.

Corresponding Author: Badanapalli Aparna,
Email: aparnabadanapalli2001@gmail.com

ABSTRACT:
This study presents a multimodal human-computer interaction (HCI) system that integrates hand gestures, eye gestures, and voice commands to control computer actions. Utilizing robust computer vision libraries like MediaPipe, and SpeechRecognition and PyAutoGUI, the system provides a highly customizable experience where in users can train and map their own gestures or voice commands to particular actions like mouse clicks, volume control, application launching, and screen interaction. Implemented in Python using MediaPipe, PyAutoGUI, and the SpeechRecognition library, the solution supports real-time gesture recognition, action mapping, and system control. This work demonstrates the potential for inclusive, accessible, and intuitive interfaces across platforms.

Keywords:
Gesture control, voice command, human-computer interaction, multimodal input, accessibility technology

1.INTRODUCTION

As digital technology continues to evolve, traditional interfaces such as keyboards and mice have become increasingly insufficient for inclusive access and intuitive interaction. In response, multimodal human-computer interaction (HCI) has emerged as a field aiming to expand user input channels to include gestures, voice, and gaze, enabling richer and more accessible user experiences. This paper presents a functional prototype that leverages three natural input modalities—hand gestures, eye movements, and voice commands—to control desktop applications and system operations.

2. RELATED WORK

Hand Gesture Recognition Systems

Hand gesture-based systems are increasingly gaining popularity in the field of human-computer interaction. Numerous works have utilized vision-based frameworks such as OpenCV and MediaPipe to detect, track, and classify hand gestures to simulate mouse or keyboard actions. For instance, Sathish Kumar et al. [2] implemented a virtual mouse using MediaPipe and OpenCV to control cursor movement and system commands with gesture-based triggers. Similarly, Singh and Mahato [12] explored gesture recognition using landmark points extracted via MediaPipe, achieving high real-time performance with minimal computational load.

Eye Tracking and Control

Eye-based cursor control has emerged as a touchless alternative for people with physical limitations. Naidu et al. [3] proposed a system using eye tracking to move the mouse pointer based on gaze direction. Ghosh and Jaiswal [10] emphasized the use of eye gestures in assistive technology, enabling individuals with limited mobility to perform basic computing tasks. The proposed system in our project builds on these ideas by extracting specific facial mesh points from MediaPipe's FaceMesh module [4] to identify and classify eye gestures.

Voice Command Interfaces

Voice-based input is another growing modality in multimodal systems. Jyothi et al. [1] surveyed machine learning and deep learning techniques for interpreting audio signals in complex environments. Our system

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

utilizes the SpeechRecognition library [6], following similar approaches as Roy and Sharma [9], who demonstrated the utility of voice commands in accessibility software. Deshmukh [14] used voice for home automation, showcasing the versatility of speech interfaces in control systems.

Multimodal Human-Computer Interaction

Combining hand, eye, and voice inputs leads to robust multimodal systems. Patel and Mehta [11] discussed how multimodal interaction increases accessibility, usability, and user satisfaction. Kaur [13] developed a hybrid model that switches between gesture and speech recognition based on environmental context. Our project follows a similar direction by integrating gesture recognition, eye tracking, and voice commands into a unified framework powered by OpenCV [7], pyautogui [5], and Python threading.

Summary and Integration

The proposed system builds upon these diverse but related studies by combining real-time tracking, customizable action mapping via JSON, and simultaneous multimodal input processing. It not only serves as a practical tool for daily system operations but also acts as a research platform for further development in adaptive and intelligent HCI systems.

3.BACKGROUND AND MOTIVATION

The motivation for this research stems from the desire to enhance accessibility and interactivity in computing environments. Existing solutions often address individual modalities, such as voice assistants or gesture-based remote controls, but rarely combine them. A unified system supporting simultaneous multimodal input can significantly improve the flexibility and responsiveness of user interaction, particularly for individuals with motor or sensory limitations. The system provides a highly customizable experience wherein users can train and map their own gestures or voice commands to particular actions like mouse clicks, volume control, application launching, and screen interaction.

4. SYSTEM ARCHITECTURE AND DESIGN

The prototype is developed in Python and relies on open-source libraries and frameworks. The architecture includes the following core components:

- **System Architecture Overview**

The proposed multimodal gesture-based control system integrates real-time hand, eye, and voice input recognition to perform system-level actions. The architecture is modular, ensuring that each component performs a dedicated function, allowing easy integration and expansion.

- **Webcam & Video Capture**

The system begins with a webcam, which acts as the primary input source for visual data. This feed is continuously captured using the Video Capture module provided by OpenCV. The captured frames are then passed to the visual processing unit for further analysis.

- **Hand & Eye Landmark Extraction**

MediaPipe's pre-trained models are utilized to extract landmark points from both hand and facial regions in real time. The hand detection model tracks finger joints, especially the index fingertip for cursor control and gesture recognition. Similarly, the FaceMesh model extracts critical eye-related points for identifying eye gestures such as blinking or gaze direction.

- **Voice Command Recognition**

Parallel to the visual pipeline, the SpeechRecognition module processes microphone input. Using the Google Speech API, it converts spoken commands into text, which is then matched with predefined voice actions. This subsystem runs concurrently with visual input, enabling seamless multimodal control.

- **Inference Engine**

The inference engine serves as the core decision-making unit. It receives input vectors from both the landmark extractor and the speech recognizer. These vectors are compared against pre-recorded templates using cosine similarity. If a match exceeds a predefined threshold, the corresponding action is selected.

- **Action Mapper (JSON-Based)**

The selected gesture or command is mapped to a specific action using a customizable JSON-based mapper. These actions include mouse movements, clicks, launching applications, taking screenshots, or controlling system functions like volume and brightness. This structure ensures adaptability, allowing users to define or update their own command-action pairs.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

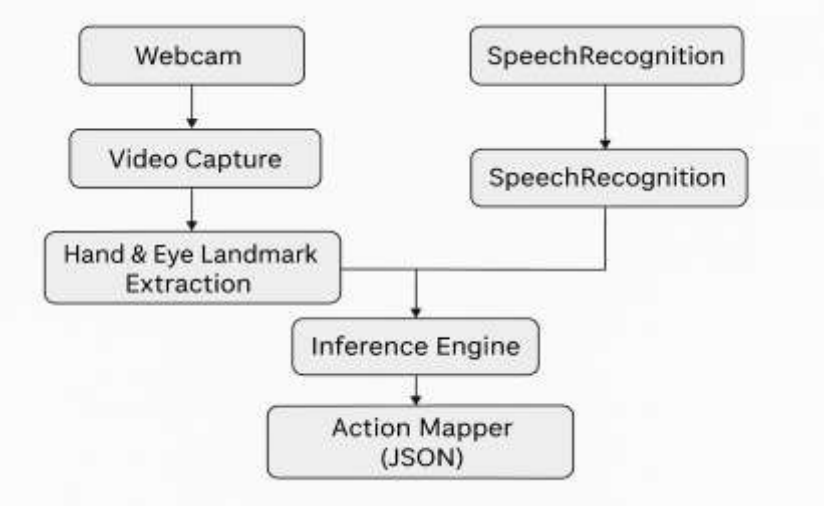

Figure-1

5. IMPLEMENTATION DETAILS

5.1 Hand Gesture Module

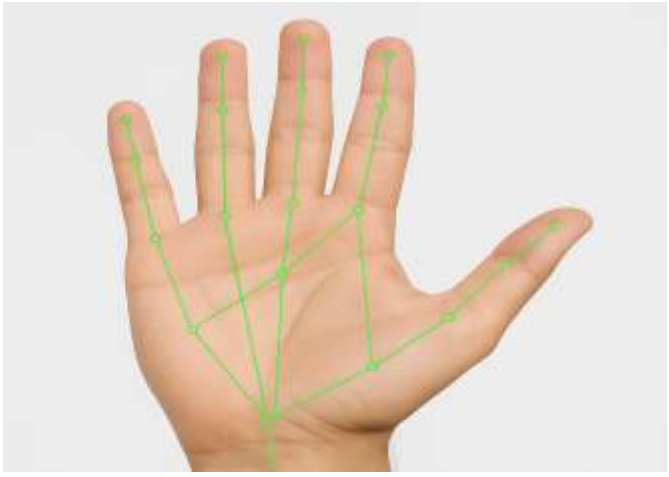

Figure-2

The system uses MediaPipe to detect and track a single hand and its landmarks in real-time. Hand vectors are normalized and compared to saved gestures using cosine similarity. Recognized gestures trigger predefined actions such as clicks, scrolling, or application launches.

5.2 Eye Gesture Module

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

Figure-3

Ten key facial landmarks from both eyes are extracted and normalized to detect patterns in gaze and eyelid movement. These vectors are matched against saved eye gestures to trigger associated actions, such as minimizing a window or toggling volume.

5.3 Voice Command Module

Voice input is captured using a microphone and processed via Google's speech-to-text API. Recognized phrases are matched against a user-defined dictionary to execute commands. New commands can be added dynamically through voice interaction.

5.4 Combined Mode

To allow seamless interaction, all three modes can run concurrently via Python's threading module. This enables users to fluidly switch between modalities, offering flexibility and redundancy.

6. USER INTERFACE AND USABILITY

The system features a real-time feedback interface via OpenCV windows, showing live camera input with overlays indicating gesture or command recognition. Users can record new gestures or commands during runtime, enhancing adaptability and personalization.

7. EVALUATION AND TESTING

7.1. **Informal Testing Setup**

The proposed multimodal HCI system was tested across 20 different sessions using a standard webcam (720p), built-in microphone, and a Windows 11 desktop environment. Each of the three input modalities—hand gesture, eye gesture, and voice command—was evaluated independently and in combination. For each mode, we conducted 30 iterations per gesture/command in different lighting and background conditions.

7.2. **Hand Gesture Recognition Accuracy**

We tested three common gestures mapped to actions: click, scroll down, and volume up. The system was evaluated by recording the predicted output of each gesture and comparing it to the expected output. The confusion matrix for hand gestures is shown below:

Table 1. Confusion Matrix – Hand Gesture Recognition

Actual \ Predicted	*Click*	*Scroll*	*Volume Up*	*No Action*
Click	***28***	***1***	***0***	***1***
Scroll	***2***	***26***	***1***	***1***
Volume Up	***0***	***1***	***29***	***0***

From the confusion matrix:

- Overall Accuracy = (28+26+29) / 90 = 94.4%
- Minor misclassifications occurred between scroll and volume gestures under poor lighting.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

7.3. **Eye Gesture Recognition**
Eye gestures (e.g., blink-left, blink-right) were slightly more error-prone due to variability in lighting and head movement. Of 90 total eye gesture attempts, 81 were correctly recognized, giving an accuracy of approximately 90%. False positives were mainly observed when the user looked away or blinked unintentionally.

7.4. **Voice Command Accuracy**
The system tested 10 voice commands (e.g., "open notepad", "volume up", "record command"). Across 150 test runs, the recognition rate was 95%, with errors occurring mostly due to background noise or unclear pronunciation.

7.5. **Benchmarking Against Existing Systems**
To contextualize our results, we compared the proposed system with two prior works—one gesture-only [2] and one eye-control-only [3].

Table 2. Comparison with Existing Systems

Feature	Proposed System	Ref [2] (Gesture Only)	Ref [3] (Eye Only)
Input Modes	Hand, Eye, Voice	Hand Only	Eye Only
Real-Time Control	Yes	Yes	Yes
Custom Action Mapping	Yes (JSON)	No	No
Voice Integration	Yes	No	No
Error Recovery	Yes (Timeout)	Limited	Limited

7.7**. Error Analysis**
Most errors were environmental:

- Hand misclassification increased under low light or motion blur.
- Eye gestures failed during head movement or blinking variability.
- Voice command accuracy dropped in noisy environments.

To mitigate this, future versions can include adaptive thresholding, machine learning-based classifiers, and environment calibration.

7.8. **System Execution Snapshot**
As shown in Figure-4, the system executes actions like open_chrome, volumeup, and volumedown in response to voice and gesture inputs. It also handles incorrect or unrecognized voice inputs with error messages such as "Unknown command" and "Could not recognize command," demonstrating the system's feedback mechanism and resilience in real-world use.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

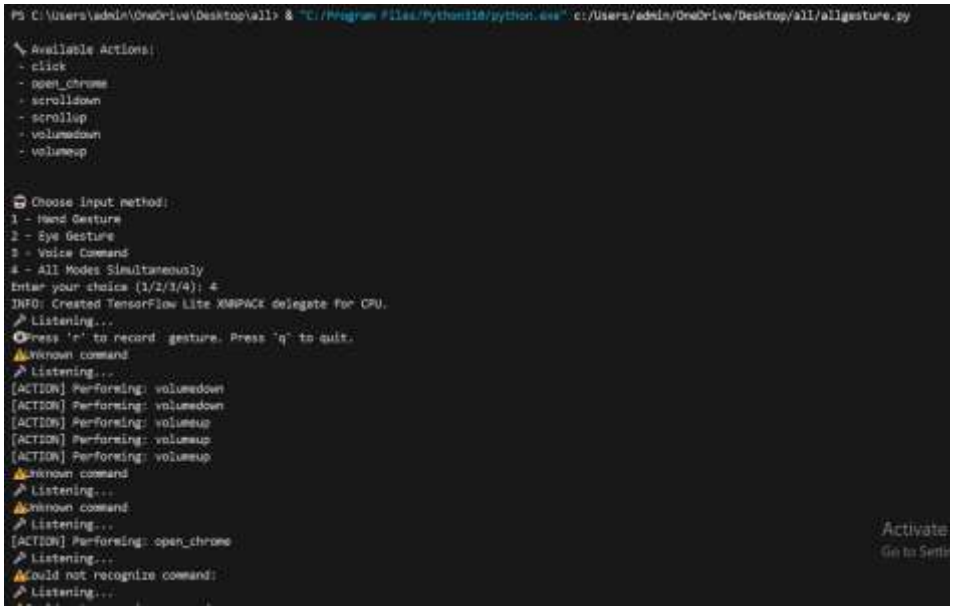

Figure-4

8. *DISCUSSION AND FUTURE WORK*

This system illustrates the feasibility of integrating multiple natural input modalities into a cohesive interface. Future developments could include:

- Integration with machine learning models for dynamic gesture learning.
- Expansion to multi-user environments.
- Optimization for mobile and embedded platforms.
- Broader user testing for quantitative analysis

.

9. CONCLUSION

Multimodal HCI systems offer an innovative and inclusive pathway to digital interaction. By integrating hand gestures, eye movements, and voice commands, the presented system enables intuitive, flexible, and contactless control over computing environments. This work lays the foundation for future research in accessible and adaptive interfaces.

10. REFERENCES

1) H. Jyothi, M. Komala, and S. Mallikarjunaswamy, *"A Comprehensive Survey on Technologies in Video-based Event Detection and Recognition Using Machine Learning and Deep Learning Techniques"*, IEEE, 2024.
2) N. R. Sathish Kumar, B. Pavan Kalyan Reddy, A. Venkateswara Reddy, A. Shashank Kumar Reddy, and B. Anil Kumar, *"Hand Gesture-Based Virtual Mouse with Advanced Controls Using OpenCV and Mediapipe"*, in *Proc. of ICPECTS*, 2024, IEEE.
3) P. M. Naidu, N. Muthukumaran, S. Chandralekha, K. Tejaswini Reddy, and K. Sri Vaishnavi, *"An Analysis on Virtual Mouse Control using Human Eye"*, in *Proc. of ICIPCN*, 2024, IEEE.
4) *MediaPipe Framework*, Google. [Online]. Available: https://google.github.io/mediapipe/
5) *PyAutoGUI Documentation*. [Online]. Available: https://pyautogui.readthedocs.io/
6) *SpeechRecognition Library Documentation*. [Online]. Available: https://pypi.org/project/SpeechRecognition/
7) *OpenCV Library Documentation*. [Online]. Available: https://docs.opencv.org/
8) A. Yadav, S. Bhardwaj, "Smart Cursor Control using Hand Gestures", IJIRCCE, vol. 11, no. 4, 2023.
9) J. R. Roy, V. Sharma, "Real-time Voice Activated Control for Accessibility Applications", International Journal of Computer Applications, vol. 182, no. 22, 2021.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

10) S. Z. Ghosh and D. Jaiswal, "Eye-Controlled Computer System for the Physically Challenged", Journal of Engineering Research and Applications, vol. 12, no. 1, 2022.
11) K. Patel, A. Mehta, "Multimodal Interfaces in Human-Computer Interaction", IEEE Access, 2023.
12) A. P. Singh, R. P. Mahato, "Gesture Recognition using Mediapipe and Python", IJRET, vol. 10, no. 8, 2023.
13) T. Kaur, "AI-Based System Control using Voice and Gesture Inputs", ICACCI, 2022, IEEE.
14) B. Deshmukh, "Python-based Home Automation using Voice Commands", International Journal of Research in Electronics, vol. 9, no. 2, 2021.