AI-DRIVEN CURSOR CONTROL VIA FACIAL EXPRESSIONS AND VOICE INPUT

Gundra Ishwarya
B. Tech Student, Department of AIML
Aditya University Surampalem, India
21a91a6144@aec.edu.in

Nagadevara Navya Sri
B. Tech Student, Department of AIML
Aditya University Surampalem, India
21a91a6155@aec.edu.in

Buridi Amrutha Varsha
B. Tech Student, Department of AIML
Aditya University Surampalem, India
22a95a6104@aec.edu.in

Mommileti Pravallika
Assistant professor, Department of AIML
Aditya University Surampalem, India
chinnimommileti@gmail.com

Abstract- This project is a hands-free control system for a computer based on facial recognition and voice control, intended mainly for those with motor disabilities or mobility issues. The program enables users to control the mouse pointer based on actual facial movement detected by a webcam. It also supports voice command recognition for carrying out basic mouse operations like left click, right click, double click, typing, and pressing the Enter key. The interface is divided into three main sections: Cursor Settings, Voice Commands, and a middle camera feed. Users can adjust cursor speed, circle size, and visual colors through RGB values in the Cursor Settings panel. The Voice Commands panel provides configuration of custom voice triggers for every action. A live video stream from the webcam is used to help monitor the direction in which the user's face is pointing, supporting accurate and dynamic cursor control. The system is also flexible, easily visually customizable, and easy to use, characteristics that make the system a valuable tool in enhancing accessibility. While presently functioning, the program would be improved by enhanced visual feedback along with more advanced user input. Overall, this project demonstrates a new approach to human-computer interaction by combining face tracking with speech recognition in place of traditional mouse input.

Keywords - face recognition, voice commands, hands-free control, human-computer interaction, accessibility technology, cursor control, assistive devices, facial tracking, speech recognition, real-time interaction, Python, OpenCV, PyAutoGUI, MongoDB, and Tkinter GUI.

I. INTRODUCTION

This is a new application that combines computer vision, speech recognition, and user interface design to provide a computer interaction system that is accessible and hands-free. The main goal of this project is to design a system that enables users to manipulate a computer cursor with facial movements and perform mouse operations with voice commands. This solution is especially focused on people with physical disabilities, allowing them to use digital devices more independently and effectively.

The project utilizes various technologies in an effort to achieve its purpose. OpenCV and Dlib are utilized in order to detect faces and track them so that the application can detect the user face in real-time and map head movement into cursor movement. PyAutoGUI is utilized in manipulating mouse movement according to the position of the face and voice commands.

Speech recognition allows users to provide voice commands like "left click", "double click", or "type" and the system will read them and execute according to command. MongoDB is employed to log every user action with timestamps and details and thus offers a backend to track system usage and performance. Furthermore, an

interface based on GUI and Tkinter has been implemented to enable administrators or users to see these logs based on

filters such as the type of action or the date to make the system transparent and user-friendly.

The face-controlled mouse interface is optimized in terms of usability and accessibility. The screen is divided into major areas: cursor settings, voice command settings, and a middle camera feed. Customization of cursor speed, graphical display items such as circle and line color, and text display is provided in cursor settings, giving users a choice to personalize the system to suit their needs and perception. Voice command options enable users to create their own command words for various operations so that the system can be made user-adjustable based on personal speech patterns. The center camera feed displays the face of the user and follows movement but can be enhanced further with the inclusion of real-time visual feedback markers for cursor position and face detection.

This project has a high contribution to the discipline of assistive technology and human-computer interaction. By combining facial recognition and speech input, it provides a multimodal control interface that is independent of the use of standard hardware such as a keyboard or mouse. It presents opportunities for mobility-impaired users to interact with digital systems without physical contact, thus enhancing digital accessibility. Additionally, the logging system supported by MongoDB provides a layer of responsibility and usability tracking, which can be employed to monitor how the system is used, diagnose problems, or even inform future enhancement.

Finally, the voice-controlled mouse and face-controlled computer with the log viewer is a practical proof of concept for how the new technologies can be merged together to increase usability and accessibility. Not only does it offer a new way to control a computer, but it also makes the data about usage clearly transparent and reviewable. The system is a worthwhile prototype in the expanding field of smart assistive tools and hands-free computing solutions.

II. LITERATURE SURVEY

Over the last few years, human-computer interaction (HCI) has developed from traditional input devices such as keyboards and mice to more accessible and intuitive interfaces. One such development is based on controlling computer systems through facial recognition and voice commands, allowing hands-free use and enhancing digital accessibility.

Facial recognition technologies have been in increasing demand with their effectiveness in monitoring and tracing user activity. Viola and Jones presented an object detection model that was widely adopted as the basis of face detection real-time algorithms [1]. OpenCV, which is one of the most used computer vision libraries, has reliable functionality to perform face tracking on real-time systems [2]. Dlib adds this ability with its provision of facial landmark detection by employing ensemble regression trees [3].

Voice recognition has also been instrumental in helping with accessibility. Libraries such as Python's speech_recognition [4] and Google's Speech API [5] have made it easier to incorporate speech input into programs. These technologies are vital for individuals who have mobility impairments, allowing them to control mouse clicks and typing by voice commands [6].

The work of previous tasks such as Camera Mouse [7] and GazePointer [8] promoted webcam-based pointing, establishing precedent for face-dependent pointer control. These solutions partially addressed issues without integrated voice-controlled interaction or adjustability. Investigations also uncover how camera-based point systems have tremendous potential in physical strain minimization and long-term usability [9].

MongoDB, a NoSQL database, is increasingly employed to store unstructured logs and event data because it has a document-oriented structure with flexibility [10]. This comes in handy where real-time logging of user behavior is important for debugging, usage analytics, or monitoring [11].
Current projects have integrated more than one modality like facial expressions, eye blinks, and voice commands to offer more comprehensive control systems [12]. Most of them encounter difficulties in calibration, sensitivity to the environment, and visual feedback [13]. Merging visual customization such as cursor speed and color schemes, such as in the current project, offers a better user experience due to adapting to varied needs.
Voice command integration continues to be a research interest area, particularly on language processing and command disambiguation. Research indicates that voice interfaces through commands can improve task completion time over manual interaction for some users [14].

Despite advancement, few systems present an integrated strategy with face control, voice command, and logging tracking under a single interface. The intended project fills this lacuna by facilitating hands-free operation, personalized voice command mapping, and live logging through MongoDB. This convergence not only promotes accessibility but also provides for highly configurable interaction monitoring, which proves to be valuable in diagnostics and behavior analytics [15].

III. System and Design

This project addresses the problem that people with physical disabilities are unable to employ normal input devices like a mouse or a keyboard. The most significant problem is the lack of readily available inexpensive and simple systems for hands-free computer interaction. In order to overcome this, the project offers a face-controlled mouse system involving the use of a webcam to capture facial movements and a microphone for speech command recognition. It also features an integrated log viewer that retains all user activity for transparency, analysis, or debugging.

The primary goal is to create an environment where one can carry out cursor movement and mouse actions using anything but the hands. Face gestures are scanned via a webcam and interpreted as computer vision data, and voice commands initiate such operations as click, typing, or Enter key press. The entire activity is monitored and stored in real-time with MongoDB such that user input can be assessed through a visualization interface.

The system requires fundamental hardware like a webcam, a microphone, and a computer with regular specs (at least 4GB RAM and a dual core processor). On the software front, it employs Python along with libraries like OpenCV for camera operations, Dlib for facial landmark detection, PyAutoGUI for cursor manipulation, and SpeechRecognition for voice-based input. Tkinter is utilized to create the graphical user interface and MongoDB for saving action logs.
Functionally, the system accommodates cursor movement through facial direction, voice-controlled mouse operations, a live camera feed, and action logging. It also accommodates customization of cursor speed, visual aspects such as color, and voice command keywords. These functionalities are made available through an easy-to-use GUI. The application is started through a start button and terminated upon user request.
From a non-functional perspective, the system is optimized for high usability and accessibility, particularly for users with physical disabilities. It is real-time performance optimized and guaranteed to run reliably without too many interruptions. The interface is minimal and easy to set up, and the system is future-proofable to handle more sophisticated input modes like gesture or eye tracking.
From a design standpoint, the architecture is modular where each component i.e., face tracking, voice recognition, GUI, and logging is implemented as a separate module. The modules communicate with each other in real time and provide a seamless experience. The face tracking module takes video input, detects facial landmarks, and maps head movement to cursor movement. The voice command module identifies speech input and cross-references it against specified commands to perform mouse actions. The GUI module offers preference setting and status monitoring. The logging module stores all interactions with time stamps in the database for later analysis.

IV. METHODOLOGY

The approach followed in this project is the integration of face recognition and voice command systems to operate mouse without physical intervention. The project is developed following a modular development procedure, where each module—face tracking, voice recognition, GUI interface, and logging—is separately developed and tested prior to the integration. Requirements gathering is the procedure followed in the beginning of development, where the requirements of users with motor disabilities are researched. The first priority is to make the system interactive, accessible, and user-friendly.

The first process is webcam face tracking. This is achieved using Python libraries such as OpenCV and Dlib. OpenCV handles the webcam video stream, and Dlib identifies facial landmarks. The landmarks, especially nose and eye coordinates, are mapped to cursor movement. Head movements in different directions map to cursor movement on the screen. The cursor sensitivity can be adjusted using the interface provided.

The second component of the methodology is voice recognition. This is done through the use of the SpeechRecognition library in Python. Voice commands such as "left click", "right click", "double click", "type", and "enter" are predefined. Upon detection of these words from the microphone input, PyAutoGUI is utilized to perform the corresponding mouse or keyboard action. Users are able to modify the voice commands through the interface to their speech patterns or preference.

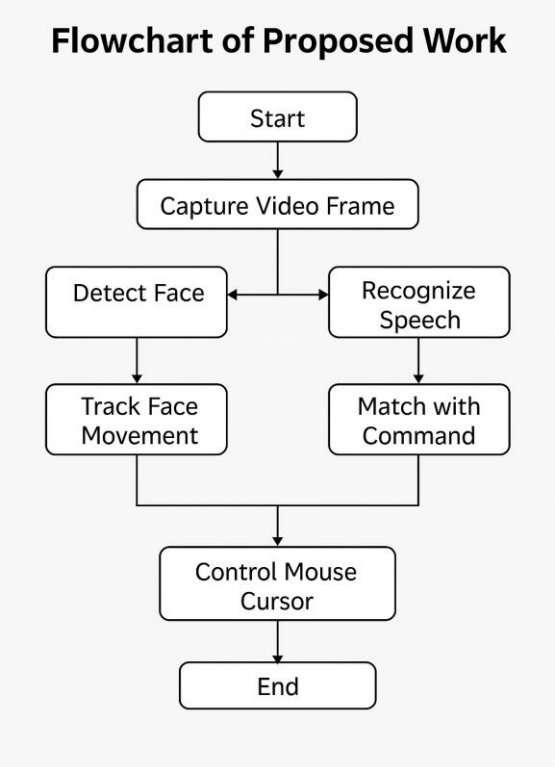

Figure 1 Flow chart of proposed work

The third aspect is the graphical user interface (GUI) implemented with Tkinter. The GUI contains two principal panels: one visual and cursor-setting panel and one voice command configuration panel. Through the GUI, users can input RGB values to be used in visual indicators, set cursor speed, and execute the program. A middle window region is used to present the real-time feed from the camera.

The last component is the action logging system. This is managed using MongoDB, where all user activity—a move of the cursor, click, or voice-activated functionality—is logged with a timestamp. There is also a Log Viewer application developed using Tkinter where users or developers can filter, search, and analyze the logs according to action type and date. This does not make the system interactive but also transparent and traceable.

Each subsystem of the system is individually tested for precision, latency, and responsiveness. Integration testing will make sure the modules are coherent in their interworkings. Real-time response and usability take priority to provide access for users who depend on assistive technology.

In general, this approach offers a systematic and adaptable method of developing a hands-free mouse control program. The system is designed to be adaptive, with ease of customization, future enhancement, and application in different environments where conventional input devices are not possible.

V. IMPLEMENTATION

The application of the face and voice-controlled mouse system consists of several interrelated modules that were written mainly with Python. The system employs a webcam for face tracking, a microphone for recording voice commands, and other Python libraries for processing these inputs and converting them into mouse or keyboard inputs. The main modules are face tracking, voice recognition, GUI development, and activity logging.

The face tracking module is done with the OpenCV and Dlib libraries. OpenCV records the video stream from the user's webcam, and Dlib detects facial landmarks. Nose or eye location is tracked in real time, and the offset from a calibrated central point is converted to cursor movement with the PyAutoGUI library. Users can control the mouse pointer by head movement.

For voice recognition, SpeechRecognition library is utilized. Standard commands like "left click", "right click", "double click", "type", and "enter" are translated to equivalent PyAutoGUI calls. The user can customize these commands via the GUI. As soon as a voice command is recognized and matched, the system executes the allocated action immediately, providing an efficient hands-free operation.

Graphical User Interface (GUI) is developed using Tkinter. GUI has two main parts—cursor settings and voice command. Users can set cursor speed, indicator circle size, and adjust on-screen element color by providing RGB values. Voice command part provides the facility to set custom mouse and keyboard commands through phrases. A "Start Program" button starts the face and speech tracking system.

The camera stream appears in the center of the GUI window, although actual tracking is done in the background. The user interface is dark with high-contrast text so that it remains easy to use even under low-light levels. All aspects are user-configurable, so the application is versatile for various users and applications.

In order to record user interactions for debugging and usage monitoring, there is a logging mechanism based on MongoDB. All the actions—movement of cursors, clicking, or command—is saved with time stamp into a MongoDB database. Another Log Viewer application is designed with Tkinter, and there users or programmers can observe, filter, and analyze those logs according to date or type of action.
In the implementation, modular programming principles are followed to enable reusability and maintenance of code with ease. The separate modules are tested individually prior to assembling them in complete form. The assembled system provides a real-time, efficient, and responsive means of computer control without the use of the traditional mouse or keyboard. This implementation is especially beneficial to motor-disabled users, enabling ease of use and autonom

VI. RESULTS AND CONCLUSION

The system was implemented and tested on different machines with typical webcams and microphones for its responsiveness, accuracy, and usability. The cursor motion controlled by the face worked properly in the presence of good lighting, as the Dlib-based face landmark detection tracked the nose position of the user accurately to translate the mouse pointer. The cursor velocity and direction were quite consistent with the motion of the user's head, and could be controlled through the GUI options. The incorporation of live customization options like cursor speed and color schemes made the user experience more pleasing.

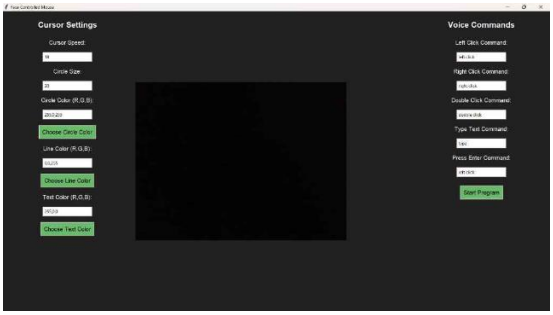

Figure 2 UI for Customization

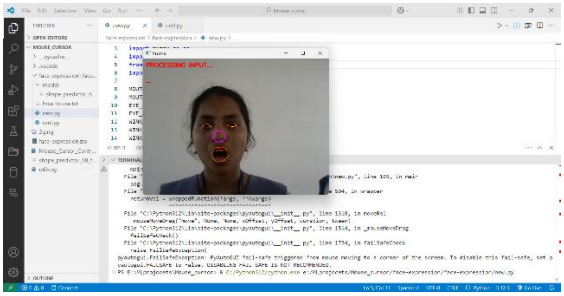

Figure 3 Activating Mouse

The voice recognition module proved to be highly accurate when commands were enunciated clearly in a clean environment. Commands such as "left click," "right click," and "type" were uniformly detected and carried out within an acceptable response time. Command detection errors were mainly noted in noisy environments or when non-native English pronunciation was used. Nevertheless, the capacity for user-defined command words enhanced system flexibility for various users. The logging module was able to record all activity into MongoDB with accurate timestamps, and the included Log Viewer application enabled users to filter and review logs at ease.

In summary, the system integrates face tracking and voice recognition well to offer a hands-free, computer-accessible control solution. It provides logging, real-time feedback, and customizability both for the users and the developers. The application is particularly beneficial for people with motor disabilities, enabling them to access digital devices on their own. Though voice detection could be optimized for diverse acoustic conditions, the project adequately proves the practicality and feasibility of a multimodal control interface with low-cost hardware and open-source software.

VII. FUTURE EXTENSION

The system in place provides a strong basis for hands-free computer interaction based on face tracking and voice recognition. Yet, a few possible augmentations can be made to enhance its usefulness, versatility, and scalability. One of the significant expansions would be the inclusion of machine learning-based gesture recognition, allowing users to execute other operations like scrolling, dragging, or zooming through certain head or facial movements. Adding such smart gesture classifiers can provide possibilities of interaction beyond mere mouse movement and clicks.

Another significant enhancement would be the addition of natural language processing (NLP) functionality to parse more advanced or chatty voice commands. This would enable users to execute compound operations, like "open browser and find weather," enhancing the system's flexibility. For increased accessibility, regional language and accent support could be included through tailor-trained speech models, making the system more accessible to users from varying linguistic backgrounds.

Extensions further can include cross-platform support, for example, making the tool accessible on mobile devices or web-based platforms using technologies like WebRTC and React Native. In user experience, introducing visual hints or augmented reality overlays to display tracking status, gesture recognition, or command execution feedback can significantly enhance usability and trust in real-time usage.

References

[1] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. CVPR.

[2] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

[3] King, D. E. (2009). Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research.

[4] Uberi. (2013). SpeechRecognition Library for Python. https://pypi.org/project/SpeechRecognition/

[5] Google Cloud. (n.d.). Speech-to-Text API. https://cloud.google.com/speech-to-text

[6] Wobbrock, J. O., et al. (2011). Ability-based design: Concept, principles, and examples. ACM Transactions on Accessible Computing.

[7] Betke, M., Gips, J., & Fleming, P. (2002). The Camera Mouse: Visual tracking of body features to provide computer access for people with severe disabilities. IEEE Transactions on Neural Systems and Rehabilitation Engineering.

[8] ITU Gaze Tracker. (2012). GazePointer. https://www.gazepointer.com/

[9] Gips, J., & Betke, M. (2003). Using face tracking for hands-free computer control. Human-Computer Interaction.

[10] Chodorow, K. (2013). MongoDB: The Definitive Guide. O'Reilly Media.

[11] Banker, K. (2011). MongoDB in Action. Manning Publications.

[12] Kim, J., & Grauman, K. (2010). Mouse-free navigation in rich image collections. CHI.

[13] Argyros, A. A., & Lourakis, M. I. A. (2004). Real-time tracking of multiple skin-colored objects with a possibly moving camera. ECCV.

[14] Oviatt, S. (2000). Taming recognition errors with a multimodal interface. Communications of the ACM.

[15] Mandal, B., et al. (2017). A review on facial expression recognition in the wild: databases, methods, and challenges. IEEE Transactions on Pattern Analysis and Machine Intelligence.