International Journal of Engineering Research and Sustainable Technologies
Volume 2, No.3, Sep 2024, P 23 - 29
ISSN: 2584-1394 (Online version)

ENHANCING AI-TOUCH FREE HAND GESTURE BASED HUMAN COMPUTER INTERACTION

Santhosh R, Shaik Abdul Hafiz
Department of CSE, Dr.M.G.R.Educational and Research Institute, Chennai, India
*** Corresponding author email address**: santhoshs3306@gmail.com

DOI: https://doi.org/10.63458/ijerst.v2i3.89 | ARK: https://n2t.net/ark:/61909/IJERST.v2i3.89

Abstract

In the era of touch-based Human-Computer Interaction (HCI) systems, there is a growing need to develop touch-free methodologies that improve user interaction. This research leverages rapid advancements in Artificial Intelligence (AI) technology, with the primary goal of enhancing touch-free HCI through the integration of AI-based hand gestures. The system aims to facilitate control without the need for traditional peripherals, such as a keyboard or mouse. Our system marks a significant advancement in reducing reliance on conventional input devices, utilizing a webcam as the primary input device. The approach combines several Python packages, including OpenCV, MediaPy, AutoGUI, PyTorch, and TensorFlow, to track hand movements and execute system actions. This paper introduces a camera-based vision control system that uses a sophisticated hand gesture algorithm, powered by machine learning models, to interpret finger gestures for managing various system functions. The system enables users to control the system cursor using natural hand and finger gestures, allowing precise control over cursor movements, clicks, scrolling, application launches, and the execution of keyboard shortcuts. This technological advancement significantly enhances human-computer interaction, providing a more interactive and user-friendly computing experience. By seamlessly integrating OpenCV, MediaPy, AutoGUI, PyTorch, TensorFlow, and a robust machine learning model, our approach demonstrates innovation in touch-free HCI. It also establishes a comprehensive framework for the development of future interactive systems. The utilization of these Python tools enhances the efficiency and scalability of the proposed system, making it a promising step toward the evolution of intuitive and accessible computing interfaces.

.

Keywords*:* *Touch-Free Human-Computer Interaction (HCI),Artificial Intelligence(AI), Hand Gestures, Webcam Vision Control System, OpenCV, TensorFlow, Machine Learning Model, System Control, Interactive Computing, Gesture Recognition, Computer Vision, Accessibility, Python*

1. Introduction

In the ever-evolving landscape of computer technology, the importance and demand for human-computer interaction (HCI) have surged. While touchscreen technology has gained massive popularity in mobile devices, its integration into computer systems presents various challenges. As an opportunity, computer vision technology employs webcams to set up virtual human-computer interaction devices, such as mice and keyboards. This study focuses on designing and implementing a finger-tracking-based virtual mouse application using a standard webcam. The goal is to create a computer-interacting object tracking utility, fostering enhanced digital human-computer interaction. With advancements in Artificial Intelligence (AI), virtual mouse devices have become increasingly valuable in our daily lives. Mhetar et al. (2014) introduced an AI-based virtual mouse system that uses finger recognition and hand gestures for computer-human interaction through computer vision. Similarly, Tsai et al. (2015) proposed an AI-based virtual mouse system capable of recognizing finger movements to control mouse cursor operations. These virtual mouse systems aim to replace traditional physical mouse devices by utilizing hand gestures and fingertip recognition for HCI. Varun K.S et al. (2019) emphasized the use of external webcams or integrated cameras for finger recognition and hand motion gestures to control mouse activities. The proposed system tracks hand and fingertip actions to perform tasks such as clicking, double-clicking, scrolling, and volume adjustment. The development of AI-based virtual mice involves the use of Python programming language, the PyAutoGUI library, and the OpenCV package for computer vision. The MediaPipe library is employed for hand and fingertip tracking, while additional libraries, including Autopy and PyAutoGUI, facilitate cursor movement and system actions.

The proposed AI-based virtual mouse addresses real-time challenges, such as limited space for physical mouse usage and safety concerns during the COVID-19 pandemic. Farooq et al. (2014) highlighted the importance of hand gestures and finger motion recognition using webcams to reduce reliance on external mouse devices. The primary goal of the proposed AI-based virtual mouse is to provide an alternative input device that reduces

dependency on physical mice for system control and activities. This is achieved through an integrated camera or webcam that captures hand and finger movements, enabling actions like clicking, scrolling, volume adjustment, and zooming.

2. Literature Review

The AI-based virtual mouse utilizes hand gestures and finger recognition to offer cursor control without the need for additional hardware. Shibly et al. (2015) introduced a system that enables users to navigate the system cursor using hand gestures and colored finger caps. By recognizing various hand gestures and finger movements, the system facilitates essential mouse operations such as left-clicking, right-clicking, scrolling, and dragging. Joshi et al. (2015) focused on enhancing system interaction through hand gestures, allowing users to perform various activities. By leveraging a laptop's built-in camera, hand movements are recognized to control the mouse cursor, enabling basic operations like cursor movement, selection, and re-selection. Matlani et al. (2021) proposed a cost-effective hand and finger recognition system for notebooks, which supports cursor pointing, clicking, and file transfer actions. Implemented with simple algorithms, this system enhances interaction between connected devices using hand gestures. The virtual mouse system, scripted in Python, ensures interactivity, responsiveness, and ease of implementation, due to Python's simplicity and platform independence. Defining hand movement actions enhances the system's extensibility, as suggested by Haria et al. (2017), although the range of hand gestures may be limited. Existing systems primarily rely on either wired or wireless mice for cursor control, whereas Tran et al. (2021) introduced virtual mouse control via hand gestures and skin color recognition. This approach allows basic mouse operations without the need for colored finger caps, thereby enhancing flexibility. Mhetar et al. (2014) highlighted the complexity of existing systems that utilize static hand recognition, making them challenging to understand and use. These systems often depend on factors like fingertip identification and hand shape for defined actions, which can complicate usability. In summary, AI-based virtual mouse systems provide innovative alternatives to traditional mouse control by utilizing hand gestures and finger recognition for improved interaction and usability. Through advancements in technology and algorithm design, these systems aim to simplify human-computer interaction while minimizing hardware requirements and reducing user complexity.

3. Problem Formulation

The virtual mouse system operates using hand recognition tools and color caps on fingers, eliminating the need for additional hardware while enabling cursor control through simple hand gestures. Sharma et al. (2015) introduced a video-based hand gesture and finger recognition system using webcam input, where the system identifies hand color to determine the cursor's position. However, executing this algorithm in real-world environments presents challenges due to several factors:

- Environmental Noise
- Ambient Light Conditions
- Variations in Skin Textures
- Background Objects that Share a Similar Skin Tone

The accuracy of the color determination algorithm is critical for ensuring functionality across diverse skin tones and lighting conditions. Additionally, for the clicking functionality, users must maintain a 15-degree angle between their fingers to register a click. Robertson et al. (2004) proposed a system that could replace traditional mice, using algorithms based on colored ribbons for mouse control. These research papers contribute significantly to the field and inspire further exploration in related areas.

The proposed project offers flexibility in development and seamless integration into existing systems. The algorithm development proceeds through the following steps:

Step 1: Image Capture : The camera captures an image of the user's hand.

Step 2: Hand Extraction :The system extracts and recognizes the human hands from the captured image.

Step 3: Coordinate System :The positions of the hands are mapped to a conventional coordinate system.

Step 4: Frame Detection: Upon detecting the second frame, the system proceeds to the next stage.

Step 5: Position Comparison: The system captures the positions of the hands in the second frame, compares them with the first frame, and moves the cursor accordingly.

Step 6: Click Detection: The system measures the angle between the user's fingers. If the angle is less than 15 degrees, it triggers a left-click, allowing for mouse operations using just the hands.

Step 7: Volume Control; If the angle between the forefinger and thumb exceeds 90 degrees, the volume is increased. If it falls below 45 degrees, the volume is decreased.

By following these steps, users can perform common mouse functions seamlessly using hand gestures, enhancing interaction efficiency and eliminating the need for physical mouse device .

4. System Methodology

4.1 Flowchart

A flowchart serves as a visual representation of the sequence of processes or data flow necessary to complete a task. For the virtual mouse system, the first step involves scanning the user's hand, followed by tracking finger movements and performing corresponding actions based on these movements. These actions include cursor movement, clicking, scrolling, and executing keyboard shortcuts, as shown in Figure 1. The flowchart outlines the sequential steps involved in using the virtual mouse system, starting from hand scanning and progressing through the recognition of finger gestures to trigger various system functions. The flow of actions ensures that the user can seamlessly control the system with intuitive hand gestures.

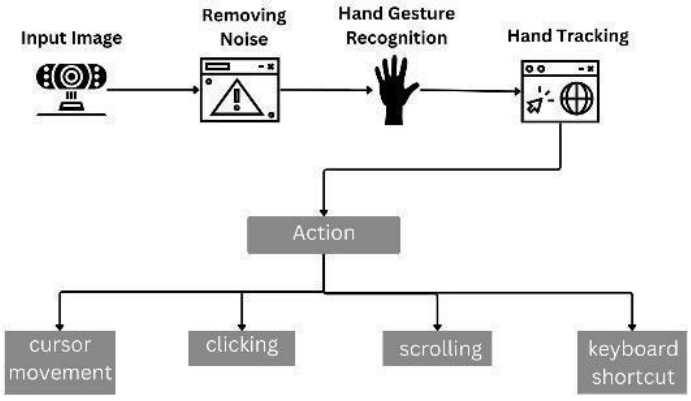

Fig.1 Activity diagram

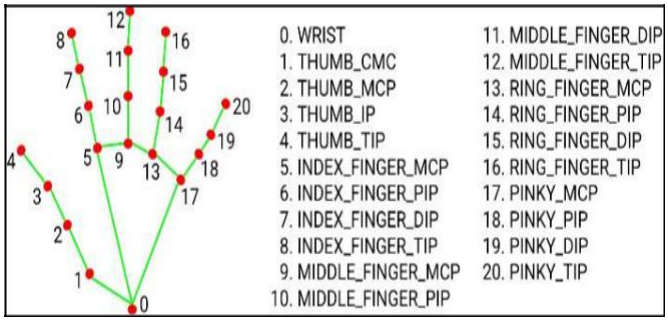

Fig.2 Represent hand's finger land marks

4.2 Activity Diagram

An Activity Diagram, as shown in Figure 6, outlines the sequence of actions and flow of activities required to accomplish a task. This diagram is particularly useful for illustrating the workflow of the virtual mouse system, where the focus is on hand detection using an external webcam or built-in camera. The system processes various

hand and finger gestures to perform different actions, such as cursor movement, volume adjustment, single and double clicks, and drag operations.

In Figure 2, each point on the hand's fingers is identified by a unique number ranging from 0 to 20. These numbers represent specific key points used in calculations within the system's algorithm. The following hand gestures and corresponding functionalities are supported by the virtual mouse system:

Single Click: If the thumb and forefinger are brought together (i.e., they touch), the system recognizes this gesture as a single click.

Volume Up: If the distance between the thumb and forefinger exceeds a threshold (e.g., more than 100 pixels), the system interprets this gesture as a command to increase the volume.

Volume Down: If the distance between the thumb and forefinger is between 0 and 40 pixels, the system interprets this as a command to decrease the volume.

Double Click: A quick gesture where the thumb and forefinger are used to simulate a double-click action.

Drag and Drop: For dragging an item (like a file or folder), the user must double-click the item using the thumb and forefinger, then hold the gesture to drag it to a new position. The item is dropped when the fingers are released.

The Activity Diagram visually maps out these operations, showing how the virtual mouse system recognizes and processes different gestures to control the system interactively. This approach enables users to perform various actions intuitively and efficiently without the need for physical input devices.

5. Implementation

5.1 Scanning the Fingers and Distance between the Fingers

Figure 3 illustrates the process of hand scanning using the MediaPipe library in Python. The objective is to trace the movement of the hand, scan, and mark specific points on the fingers. Each red dot represents a unique ID, which is crucial for implementing various functionalities.

5.2 Increase Brightness

Figure 4 depicts a scenario in which screen brightness is increased by combining the upper point of the right hand's thumb (denoted by ID4) with the index finger (ID8) and then moving the hand towards the right (left to right). This movement results in an increase in the screen brightness. By utilizing the MediaPipe and AutoGUI libraries to track hand movements, screen brightness can be effectively manipulated.

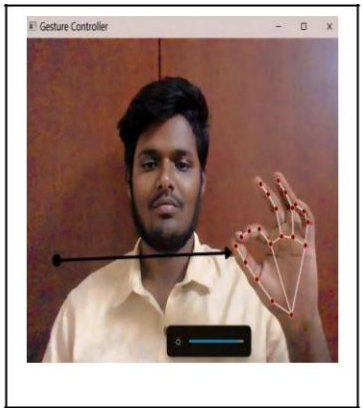

Fig 3. Scanning of hand Finger **Fig 4**. Increase of the brightness **Fig. 5** Decrease of Brightness

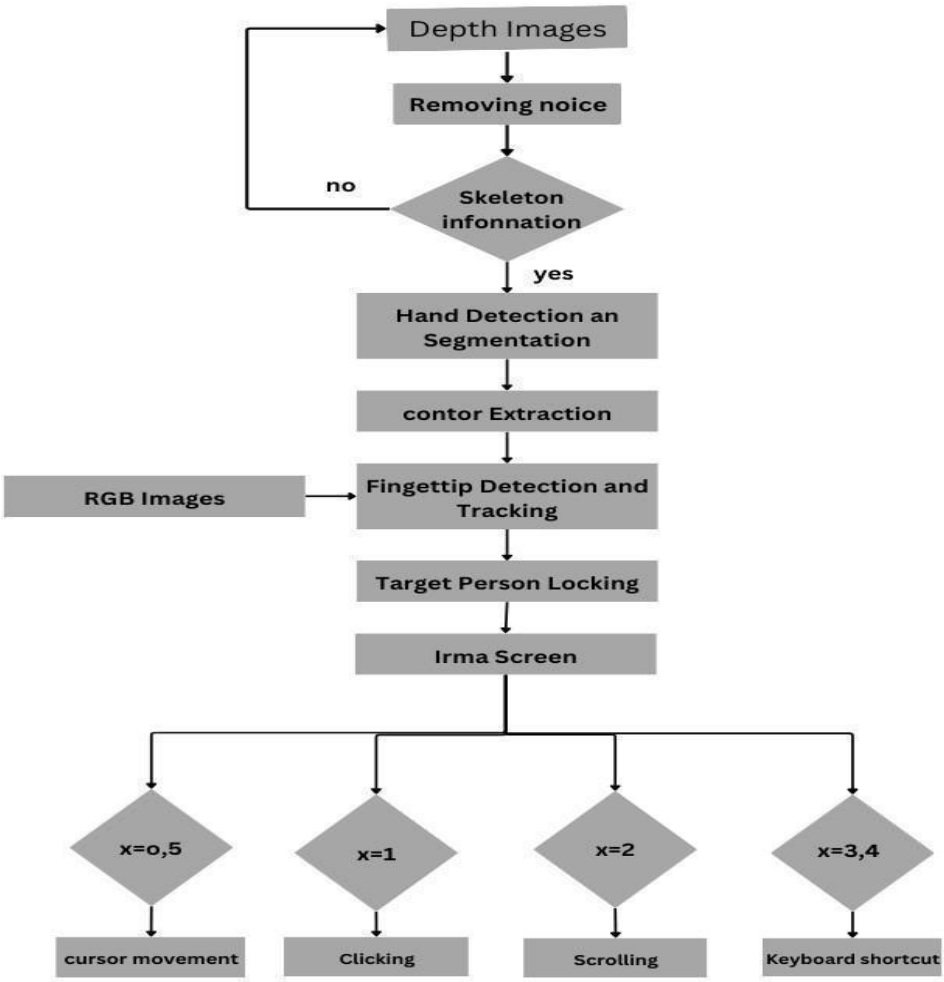

Fig 6. Activity Diagram for Virtual Mouse

5.2 Decrease Brightness

Figure 5 depicts a scenario where screen brightness is decreased by combining the upper point of the right hand's thumb (denoted by ID4) with the index finger (ID8) and then moving the fingers towards the left (right to left). This movement results in a decrease in screen brightness. Similarly, utilizing the MediaPipe and AutoGUI libraries to track finger movements enables the manipulation of screen brightness.

5.3 Scrolling down

Figure 7 depicts a scenario where scrolling down the screen is achieved by combining the upper point of the left hand's thumb (denoted by ID4) with the index finger (ID8) and then moving the hand in a downward direction (up to down). This movement causes the screen to scroll down. By utilizing the MediaPipe and AutoGUI libraries to track hand movements, the screen scroll can be manipulated effectively.

5.4 Scrolling UP:

Figure 8 depicts a scenario where scrolling up the screen is achieved by combining the upper point of the left hand's thumb (denoted by ID4) with the index finger (ID8) and then moving the hand in an upward direction (down to up). This movement causes the screen to scroll up. By utilizing the MediaPipe and AutoGUI libraries to track hand movements, screen scrolling can be effectively manipulated.

Fig.7 Screening is Scrolling down

Fig.8 Screening is Scrolling Up

6. Conclusion

The virtual mouse provides users with the ability to navigate the system cursor using finger and hand gestures, enabling actions such as left-clicking, right-clicking, and dragging. It also facilitates file transfers between systems connected via the internet. As the landscape of computer technology rapidly evolves, the demand for effective human-computer interaction has surged. While touchscreen technology is prevalent in mobile devices, its integration into desktop systems presents challenges. In this context, computer vision technology emerges as an alternative, using webcams to create virtual human-computer interaction devices like mice and keyboards.

This study focuses on implementing a finger-tracking-based virtual mouse application using a regular webcam. The goal is to develop an object-tracking application that enables computer interaction without the need for additional hardware. The AI-based virtual mouse operates through hand gesture and finger recognition, allowing users to control cursor movement with no physical mouse. Users can navigate the system cursor, perform left-clicking, right-clicking, scrolling, and dragging actions using hand gestures and colored finger caps. By utilizing a laptop's built-in camera, hand movements are recognized to control the mouse cursor on the screen.

The primary objective of this AI-based virtual mouse is to offer an alternative input device, reducing reliance on physical mouse systems for controlling the computer. This is achieved through the use of an inbuilt camera or webcam to capture hand gestures and finger movements, enabling various functions such as clicking, scrolling, volume adjustment, and zooming. By integrating computer vision technology and hand gesture recognition, the virtual mouse enhances user interaction with computer systems, offering an intuitive and efficient alternative to traditional input devices.

References

1. Mhetar,A.,Sriroop,B.K., Kavya,A.G.S., Nayak,R., Javali, R., Suma, K. V. Virtual mouse. In International Conference on Circuits, Communication, Control and Computing, pp. 69-72. IEEE. 2014.
2. Tsai,Tsung-Han, Chih-ChiHuang, Kung-Long Zhang. ‘Embedded virtual mouse system by using hand gesture recognition.’ 2015 IEEE International Conference on Consumer Electronics-Taiwan. IEEE, 2015.
3. Varun, K.S, Puneeth, Virtual mouse implementation using open CV. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 435-438). IEEE. 2019.
4. Farooq, Javeria, and Muhaddisa Barat Ali. ‘Real time hand gesture recognition for computer interaction.’ 2014 International Conference on Robotics and Emerging Allied Technologies in Engineering (iCREATE). IEEE, 2014.
5. Shibly, Kabid Hassan, Samrat Kumar Dey, Md Aminul Islam, and Shahriar Iftekhar Showrav. ‘Design and development of hand gesture based virtual mouse.’In 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology(ICASERT),pp.1-5.IEEE, 2019.
6. Joshi, Hritik, Ratnesh Litoriya, and Dharmendra Mangal.’Design of a Virtual Mouse Using Gesture Recognition and Machine Learning.’2022.
7. Haria, A., Subramanian, A., Asokkumar, N., Poddar, S., Nayak, J. S. ‘Hand gesture recognition for human computer interaction’. Procedia computer science, No.115, pp.367-374.
8. Sharma, Ram Pratap, Gyanendra K.Verma. ‘Human computer interaction using hand gesture.’Procedia Computer Science , No. 54 pp.721-727.2015.
9. Robertson, Paul, Robert Laddaga, MaxVan Kleek. "Virtual mouse vision-based interface.’ Proceedings of the 9th international conference on Intelligent user interfaces’. 2004.

10. Tran, Dinh-Son, Ngoc-Huynh Ho, Hyung-Jeong Yang, Eu-Tteum Baek, Soo-Hyung Kim, and Gueesang Lee."Real-time hand gestures potting and recognition using RGB-D camera and 3D convolutional neural network."AppliedSciences 10, no. 2 , pp.722. 2020.
11. Mhetar, Ashish, et al. "Virtual mouse." International Conference on Circuits, Communication, Control and Computing. IEEE, 2014.
12. Matlani, Roshnee, Roshan Dadlani, Sharv Dumbre, Shruti Mishra ,AbhaTewari."Virtual Mouse using Hand Gestures." In 2021 International Conference on Technological Advancements and Innovations(ICTAI),pp. 340-345.IEEE, 2021.