Thumb Gesture Controlled Mouse Movement Recognition System

Dr. S. N. Shah[1], Sakshi Binawade[2], Tanvi Mahajan[3], Sakshi Bhosale[4]
[1]*Prof. of Department of Computer Engineering, Sharadchandra Pawar College of Engineering and Technology, Someshwarnagar, Baramati, Maharashtra, Savitribai Phule Pune University, Pune*
[2,3,4]*BE Students, Department of Computer Engineering, Sharadchandra Pawar College of Engineering and Technology, Someshwarnagar, Baramati, Maharashtra, Savitribai Phule Pune University, Pune*

***Abstract*—This paper details the complete development of a vision-based human-computer interaction (HCI) framework that leverages thumb gestures for real-time mouse event simulation. Traditional input peripherals often present ergonomic challenges and require physical contact, which is undesirable in various professional environments. Our system utilizes a standard monocular RGB camera and a Python-based processing pipeline (OpenCV, NumPy, PyAutoGUI) to track the user's thumb. By employing YCbCr color segmentation and contour analysis, we achieved a calibration-free interface capable of executing cursor movement, left/right clicks, and scrolling. The final implementation includes optimized noise filtering and an adaptive gesture recognition algorithm that maintains high accuracy across diverse lighting conditions. Experimental results confirm a recognition accuracy of over 92% with a processing latency of less than 30ms, making it a viable tool for hands-free computing.**

***Index Terms*—Gesture Recognition, Computer Vision, Human-Computer Interaction (HCI), Python, Virtual Mouse, Image Processing, OpenCV, Motion Tracking.**

I. INTRODUCTION

Human-Computer Interaction (HCI) has traditionally been dominated by physical interfaces like the keyboard and mouse. However, as computing moves into the realms of Augmented Reality (AR), Virtual Reality (VR), and touchless medical interfaces, the need for Natural User Interfaces (NUI) has become critical. The fundamental goal of a NUI is to allow the user to interact with technology using movements that are inherent to human communication, such as speech and gestures.

Gesture-based interaction offers a degree of freedom that physical devices cannot match. While several systems use full-hand skeletal tracking, they often suffer from "gorilla arm" syndrome—user fatigue caused by holding the arm extended for long periods. Our project addresses this by focusing specifically on thumb gestures. The thumb offers a high range of motion with minimal physical effort, allowing for micro-gestures that are less fatiguing for the user.

The scope of this project involves the creation of a software-based bridge that captures video input, processes frames to identify the thumb, tracks its spatial coordinates, and translates these into OS-level mouse events. By avoiding expensive hardware like depth sensors (Kinect) or wearable armbands (Myo), we ensure that the system is accessible to any user with a standard laptop camera.

II. LITERATURE SURVEY AND RELATED WORK

The development of gesture recognition systems has followed several distinct paths over the last two decades.

A. Marker-Based vs. Markerless Systems

Early research focused heavily on marker-based systems. Rachit Puri (2010) demonstrated a system in MATLAB that utilized colored caps on fingers to facilitate easier segmentation. While effective, the requirement for external markers limits the spontaneity and portability of the system. In contrast, modern frameworks like the one proposed by Manav Ranawat (2021) utilize markerless tracking through Python and OpenCV, leveraging skin-tone detection to achieve a more natural user experience.

B. Sensor-Based Interaction

A parallel branch of research involves wearable sensors. Studies by Wenjun Chen and Zhen Zhang (2019) explored surface electromyography (sEMG) signals to detect muscle contractions in the forearm, mapping them to gestures. While these systems are highly accurate and work regardless of visual occlusion, the cost and intrusive nature of the hardware make them less suitable for general-purpose office or educational environments.

C. Depth Sensing and 3D Modeling

The introduction of RGB-D cameras (like Microsoft Kinect) allowed researchers like Kai Li (2018) to use depth data for hand segmentation. This method effectively solves the problem of background interference but requires hardware that is not standard on most consumer laptops. Recent work by Dinghua He (2024) has looked into improving tracking speed using Particle Swarm Optimization (PSO) and Kernel Correlation Filtering (KCF), which provides high-speed tracking even with monocular RGB input. Our project builds upon these concepts by combining high-speed tracking with low-latency mouse simulation.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

The proposed system is built on a modular architecture designed for maximum throughput and accuracy.

A. Video Capture and Image Pre-processing

The system initializes the webcam and captures frames at a resolution of 640x480. To reduce computational load, frames are converted from the standard BGR color space to YCbCr. This color space is superior for skin detection because it separates the luminance (Y) from the chrominance (Cb and Cr).

1. Skin Masking: A binary mask is created where pixels within the skin-tone range (typically $80 < Cb < 120$ and $133 < Cr < 173$) are set to white.
2. Noise Reduction: We apply a series of morphological operations—specifically Erosion and Dilation—to remove small white noise (salt-and-pepper noise) and fill holes within the detected hand region.

B. Feature Extraction and Thumb Identification

Once the hand is segmented, the system identifies the contours.

1. Contour Detection: The largest contour is selected to represent the user's hand.
2. Convex Hull and Defects: The system calculates the convex hull of the hand contour. Convexity defects (the gaps between the fingers) are analyzed to find the highest point of the hand, which is identified as the thumb tip.
3. Centroid Calculation: The center of the hand is calculated to serve as a reference point for gesture distance measurements.

C. Tracking and Event Simulation

The (x, y) coordinates of the thumb tip are mapped to the screen's coordinate system.

- Resolution Mapping: Because the camera resolution (640x480) is different from the screen resolution (e.g., 1920x1080), a scaling factor is applied.
- Smoothing Algorithm: To prevent the cursor from vibrating due to hand tremors, we implement a Weighted Moving Average filter. The current cursor position is calculated as:

$$P_{new} = \alpha \cdot P_{current} + (1 - \alpha) \cdot P_{previous}$$

where α is the smoothing factor (typically 0.6).

D. Gesture Definitions

- Movement: Thumb position within the "tracking zone."
- Left Click: Triggered when the distance between the thumb and the index finger falls below a specific pixel threshold for 3 consecutive frames.
- Double Click: Two rapid "Left Click" gestures within 500ms.
- Right Click: Holding the thumb stationary in a "trigger corner" for 1 second.
- Scrolling: Moving the thumb vertically along the edge of the camera frame.

IV. HARDWARE AND SOFTWARE REQUIREMENTS

A. Hardware Requirements

- Processor: Intel Core i3 or higher (to maintain 30+ FPS).
- RAM: Minimum 4GB.
- Camera: Standard 720p built-in webcam or USB external camera.

B. Software Requirements

- Operating System: Windows 10/11 or Linux.
- Programming Language: Python 3.8+.
- Core Libraries: * *OpenCV:* For image processing and computer vision tasks.
  - *NumPy:* For high-speed array manipulations and mathematical operations.
  - *PyAutoGUI:* For cross-platform mouse and keyboard control.
  - *MediaPipe (Optional):* Used for skeleton-based refinements.

V. IMPLEMENTATION DETAILS

The implementation phase focused on optimizing the Python environment to ensure that image processing did not bottleneck the mouse simulation. We utilized multi-threading to separate the video capture thread from the gesture processing thread. This prevents the mouse cursor from "freezing" if the image processing takes longer than one frame cycle.

The "Thumb-Only" logic was refined by calculating the angle of the thumb relative to the wrist. If the hand is held in a "neutral" position, the thumb's distinct angle allows for easier identification compared to other fingers, which often overlap when viewed from a front-facing camera.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

Testing was conducted in three distinct scenarios: Bright Daylight, Artificial Office Light, and Low Light.

A. Performance Metrics

1. Detection Rate:

In bright and artificial light, the thumb detection rate was 96.5%. In low light, this dropped to 78% due to increased sensor noise in the webcam.

2. Gesture Accuracy:
   - Movement: 98%
   - Left Click: 91%
   - Right Click: 88%

3. System Latency:

The end-to-end delay (from hand movement to screen update) was measured at approximately 28ms, which is imperceptible to the average user.

B. User Experience Analysis

A group of five students tested the system for 30 minutes of web browsing. 80% of users reported that the thumb-based navigation was more comfortable than full-hand gestures. The most common error recorded was "accidental clicks" caused by the user moving their thumb too close to their palm during navigation.

VII. CHALLENGES AND LIMITATIONS

Despite the high success rate, certain limitations remain:

- Background Interference:

If the user is wearing skin-colored clothing or has a complex background with skin-toned objects, the segmentation mask may become corrupted.

- Static Positioning:

The user must remain within the "viewing frustum" of the webcam; moving too far to the side causes loss of tracking.

- Lighting Sensitivity:

The YCbCr model, while robust, still struggles with extreme shadows which change the Cr value of the skin.

VIII. CONCLUSION AND FUTURE SCOPE

The "Thumb Gesture Controlled Mouse Movement Recognition System" has been successfully completed, meeting all the design criteria established at the start of the semester. We have transitioned from a 50% completion prototype to a fully functional, real-time HCI tool. The system proves that sophisticated computer vision tasks can be performed on consumer-grade hardware using open-source libraries.

Future Scope:

The next iteration of this project could involve the use of Deep Learning (CNNs) to recognize complex hand shapes regardless of lighting. Additionally, integrating voice commands alongside gestures would create a truly "Multimodal" HCI system, allowing for even more complex operations like drag-and-drop and window resizing.

REFERENCES

[1] H. Johnson and J. Saniie, "Distributed Gesture Controlled Systems for Human-Machine Interface," in *Proc. IEEE International Conference on Electro Information Technology (eIT)*, 2022.

[2] M. Ranawat, R. Shankarmani, and M. Rajadhyaksha, "Hand Gesture Recognition Based Virtual Mouse Events," in *Proc. 2nd International Conference for Emerging Technology (INCET)*, 2021.

[3] K. Li, J. Cheng, and Q. Zhang, "Hand Gesture Tracking and Recognition based Human-Computer Interaction System," in *Proc. IEEE Conference on Information and Automation*, 2018.

[4] He and Y. Yang, "Design of Human–Computer Interaction Gesture Tracking Model Based on Improved PSO and KCF Algorithms," *IEEE Access*, 2024.

[5] L. Yu and S. Fei, "Large-Screen Interactive Technology With 3D Sensor Based on Clustering Filtering Method," *IEEE Access*, 2020.

[6] A. Agrawal, R. Raj, and S. Porwal, "Vision-based Multimodal Human-Computer Interaction using Hand and Head Gestures," in *Proc. IEEE International Conference on Information and Communication Technologies (ICT)*, 2013.

[7] W. Chen and Z. Zhang, "Hand Gesture Recognition using sEMG Signals Based on Support Vector Machine," in *Proc. IEEE International Conference on Information Technology, Networking, Electronic and Automation Control Conference (ITAIC)*, 2019.

[8] J. Ryu, D. Im, and H.-J. Yoo, "AI SoCs for AR/VR User-Interaction," in *Proc. IEEE International Electron Devices Meeting (IEDM)*, 2021.

[9] R. Puri, "Gesture Recognition Based Mouse Events," Samsung Research India, 2010.

[10] P. Liu *et al*., "A Substation Virtual Environment Based on Motion Capture," Nanjing University of Science and Technology, 2017.

[11] X. Zhang and Y. Sun, "Evolutionary Game and Collaboration Mechanism of HCI for Intelligent Aircraft Cockpits," *IEEE Transactions on Human-Machine Systems*, 2022.

[12] Khan, "Computer Vision Based Mouse Control Using Object Detection and Marker Motion Tracking," *International Journal of Computer Science and Mobile Computing (IJCSMC)*, 2020.