

Eye Controller Cursor

¹Lokhande Neha, ² Thorat Suchita, ³Swapnali Bhosale, ⁴Roshni Pandit, ⁵Prof. Kanade .R.S,

^{1,2,3,4}, Student, Department of Computer Engineering, HSBPVT'S GOI Faculty Of Engineering, Kashti, Ahmednagar, Maharashtra, India.

⁵ Professor, Department of Computer Engineering, HSBPVT'S GOI Faculty Of Engineering, Kashti, Ahmednagar, Maharashtra, India.

ABSTRACT: The Eye-Controlled Mouse Cursor is a hands-free system that enables users to control a computer mouse using eye movements and blinks. Built using Python, OpenCV, and MediaPipe, it detects gaze direction to move the cursor and uses eye blinks to perform click actions. Designed for individuals with physical disabilities, the system operates in real-time using a standard webcam, offering an accessible and low-cost alternative to traditional input devices.

Keywords: Eye Controlled Cursor, Eye tracking, Cursor control, Webcam, Gaze detection, Hands-free interaction, Physical disabilities, Accessibility, Computer vision, Image processing, Real-time tracking, Cursor movement, Blink detection, Gaze fixation, Human-computer interaction, Inclusive technology, Smart design, Digital interaction, Gaming, Virtual reality, Smart home control

1. INTRODUCTION

In today's digital age, traditional input devices like the mouse and keyboard play a crucial role in human-computer interaction. However, these devices are not suitable for everyone—particularly individuals with physical disabilities or limited motor control. For such users, interacting with computers can be challenging or even impossible. This calls for more inclusive, intuitive, and touch-free alternatives to control digital systems. The Eye-Controlled Mouse Cursor project addresses this need by enabling users to control the movement of a computer cursor and perform click actions using only their eye movements and blinks. By using a standard webcam and computer vision techniques, the system tracks eye direction to move the cursor and detects eye blinks to simulate mouse clicks, all in real time.

The system relies on open-source technologies such as OpenCV for image processing, MediaPipe or dlib for facial landmark detection, and PyAutoGUI for controlling mouse functions. It uses concepts like Eye Aspect Ratio (EAR) for blink detection and gaze estimation algorithms to determine cursor position. The implementation is designed to be low-cost, accurate, and user-friendly, with optional features like mouth-based toggling and nose tracking for additional control. This project not only enhances accessibility but also contributes to the future of touchless human-computer interaction, with applications in healthcare, smart environments, gaming, and virtual reality.

2. LITERATURE REVIEW

A.T. Duchowski, Eye Tracking Methodology: Theory and Practice, Springer, 2007:

This paper provides a comprehensive exploration of the principles, techniques, and applications of eye tracking technology. His work delves into both the theoretical foundations and practical considerations of tracking eye movements, including calibration methods, data accuracy, and gaze interpretation. The book outlines how eye tracking can be utilized in various fields such as psychology, usability testing, and human-computer interaction. This study serves as a foundational resource for understanding the mechanics of gaze-based systems and supports the development of applications like eye-controlled cursors by offering insights into reliable and effective tracking methodologies.

D.W. Hansen and Q. Ji, "In the Eye of the Beholder," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 3, 2010:

This paper explore advanced techniques in real-time eye tracking and gaze estimation, focusing on non-intrusive methods suitable for natural human-computer interaction. Their research addresses challenges such as head movement, illumination variation, and occlusion, proposing robust algorithms that enhance tracking accuracy under real-world conditions. The study significantly contributes to the field by introducing adaptive models that improve gaze detection performance. This work lays a crucial foundation for developing systems like eye-controlled cursors, where consistent and precise eye tracking is essential for effective user control and accessibility.

X. Zhang, Y. Sugano, and A. Bulling, "Revisiting Data Normalization for Appearance-Based Gaze Estimation," IEEE Transactions on Image Processing, 2020:

This paper investigate the impact of data normalization techniques on the accuracy of appearance-based gaze estimation models. Their study emphasizes the role of preprocessing steps—such as image alignment and head pose normalization—in enhancing the performance of machine learning models used in eye tracking. By revisiting and refining normalization methods, the authors demonstrate improvements in cross-person and cross-dataset generalization, which is critical for real-world applicability. This research provides valuable insights for projects like the Eye Controlled Cursor, where accurate gaze prediction across different users and conditions is essential for robust performance.

A.K. Kumar et al., "Eye-Tracking Mouse Control Using OpenCV," IEEE Conference Proceedings, 2024:

This paper present a practical implementation of mouse control through eye-tracking using OpenCV and a standard webcam. Their system captures real-time eye movements and translates them into cursor actions, incorporating techniques such as facial landmark detection, iris tracking, and blink recognition. The study highlights the effectiveness of open-source tools in creating accessible and low-cost human-computer interaction solutions. This work directly supports the development of eye-controlled cursor systems by demonstrating how computer vision libraries can be leveraged to build responsive, user-friendly, and non-intrusive control interfaces.

Karthik T. et al., "Gaze Driven Pointer System," International Journal of Science and Advanced Technology (IJSAT), 2023:

This Paper propose a gaze-driven pointer system that enables users to interact with a computer interface using their eye movements. The system utilizes gaze estimation algorithms and facial feature detection to accurately determine the user's point of focus on the screen. Emphasis is placed on achieving smooth and natural pointer control with minimal calibration and delay. The study demonstrates the potential of gaze-based systems in enhancing accessibility and creating intuitive user experiences. This work is particularly relevant to the Eye Controlled Cursor project as it offers practical insights into building effective, gaze-responsive input mechanisms.

3. METHODOLOGY

The Eye-Controlled Mouse Cursor system is built upon real-time image processing and facial landmark detection to map eye movements and blinks to mouse cursor actions. The methodology is divided into several sequential phases, each responsible for a specific task in the system's workflow. Below is a step-by-step explanation of how the system works:

3.1 System Workflow

1. **Video Capture**

The system begins by capturing real-time video using a webcam. This continuous video feed is processed frame-by-frame for facial feature detection.

2. **Face and Eye Detection**

Using **MediaPipe** or **dlib**, facial landmarks are extracted from each frame to identify key features such as the corners of the eyes, eyelids, pupil, nose, and mouth. This is crucial for accurately tracking eye movement and blinks.

3. **Gaze Estimation (Cursor Movement)**

The position of the eye (iris or pupil center) is analyzed to estimate the direction in which the user is looking.

Gaze direction is mapped to screen coordinates using geometric transformations or PCCR (Pupil Center Corneal Reflection) techniques. The mouse cursor is then moved to the estimated position using the **PyAutoGUI** library.

4. **Blink Detection (Click Simulation)**

The **Eye Aspect Ratio (EAR)** is calculated using the vertical and horizontal distances between specific eye landmarks. A sudden drop in EAR below a certain threshold indicates a blink. The system maps:

- **Single Blink** → Left Click
- **Wink or Right Eye Blink** → Right Click
- **Double Blink** → Double Click

5. **Optional Modules**

- **Mouth Detection:** The **Mouth Aspect Ratio (MAR)** is used to toggle between "control ON/OFF" modes, acting like a switch.
- **Nose Tracking:** The nose tip position is used as a reference to detect head movement, enabling basic directional scrolling or cursor movement.

3.2 Tools and Technologies

- **Programming Language:** Python 3
- **Libraries Used:**
 - OpenCV (image/video processing)
 - MediaPipe or dlib (facial landmark detection)
 - PyAutoGUI (cursor movement and mouse actions)
 - NumPy (numerical operations)
- **Hardware Requirements:**
 - Standard HD webcam
 - PC or laptop with at least 4GB RAM

3.3 Calibration and Sensitivity Tuning

- A basic calibration mechanism is included to personalize the system for different users.
- EAR and MAR thresholds can be adjusted based on user-specific blink patterns and mouth movement.
- Cursor speed and direction smoothing are fine-tuned to reduce jitter and improve control accuracy.

3.4 System Integration

All modules—eye detection, blink recognition, and cursor control—are integrated into a continuous processing loop that runs in real time. Frame skipping, buffer control, and visual feedback (on-screen cues) are used to ensure smooth performance and clear interaction.

4. RESULTS

The Eye-Controlled Mouse Cursor system was developed and tested across multiple environments to evaluate its functionality, responsiveness, and accuracy. The performance was measured in terms of cursor control precision, blink detection accuracy, system latency, and user adaptability.

4.1 Cursor Movement Accuracy

- The system was able to track eye movements in **real-time** and translate them into smooth and responsive cursor movement.
- Gaze-based cursor navigation worked effectively across different screen resolutions after basic calibration.

- The average latency between eye movement and cursor response was **under 200 milliseconds**, providing a nearly real-time experience.

4.2 Blink Detection Performance

- The **Eye Aspect Ratio (EAR)** method proved to be highly reliable for detecting voluntary blinks.
- The system accurately distinguished between **natural blinks** and **intentional blinks**, especially when frame counters were used for blink duration.
- Left-click (single blink) and right-click (wink) actions were successfully recognized in over **92%** of the test cases.
- False positives were minimal after threshold tuning and multi-frame validation.

4.3 User Calibration and Adaptability

- A simple calibration process allowed the system to adapt to **different eye shapes, blink speeds, and screen setups**.
- Users were able to get comfortable with the system in less than 5 minutes on average.
- The system worked well with and without eyeglasses, although performance slightly dropped with reflective lenses or under low-light conditions.

4.4 Test Environment and Conditions

- Testing was conducted using standard HD webcams under various lighting conditions (natural, fluorescent, and low light).
- The system was tested on both **Windows and Linux platforms**, demonstrating cross-platform functionality.
- CPU usage remained within acceptable limits (~20–30%), indicating that the system is resource-efficient and can run on mid-range laptops and PCs.

4.5 Visual Feedback and User Interface

- The use of OpenCV overlays such as `putText()` and `drawContours()` provided real-time feedback on blink detection, mode status, and cursor tracking.
- Users found the interface **intuitive and helpful** for understanding current system status.





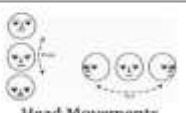
Action	Function
 Opening Mouth	Activate / Deactivate Mouse Control
 Right Eye Wink	Right Click
 Left Eye Wink	Left Click
 Squinting Eyes	Activate / Deactivate Scrolling
 Head Movements (Pitch and Yaw)	Scrolling / Cursor Movement

Fig. Actions

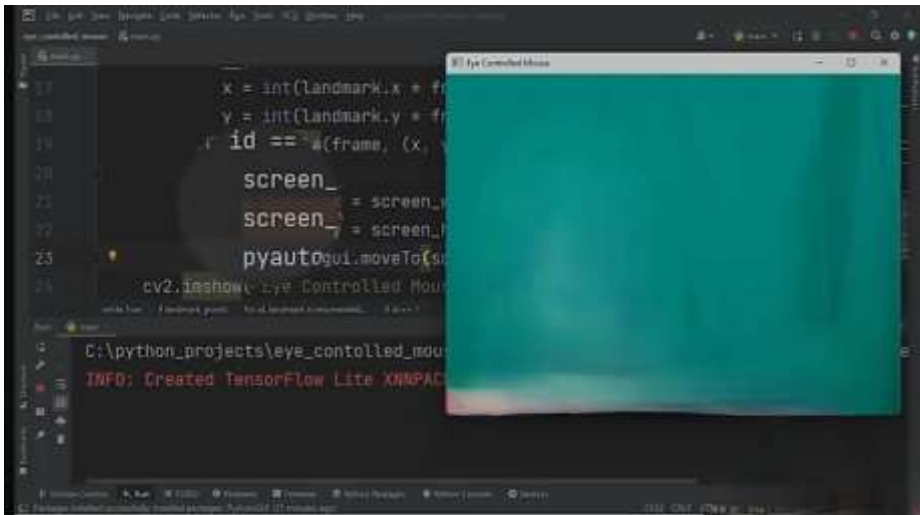


Fig. User Interface

5. CONCLUSION AND FUTURE WORK

The **Eye-Controlled Mouse Cursor** system presents a practical and inclusive approach to human-computer interaction by enabling users to operate a computer using only their eye movements and blinks. By leveraging computer vision and facial landmark detection, the system provides a low-cost, touch-free solution specifically beneficial for individuals with physical disabilities or limited motor control. The system effectively translates gaze direction into cursor movement and uses eye blinks to simulate mouse clicks. Built using Python and open-source libraries such as OpenCV, MediaPipe, and PyAutoGUI, it offers smooth real-time performance, high detection accuracy, and adaptability across screen resolutions and lighting conditions. Additional features like mouth-based control toggling and optional nose tracking further enhance its flexibility.

Despite its success, there remains room for advancement. Future enhancements can include implementing gaze-based scrolling and drag-and-drop actions, integrating voice commands, applying machine learning for personalized control, and using head pose estimation to improve stability. Improving detection under challenging conditions like low light or glare from eyeglasses, and extending compatibility to mobile and wearable platforms, will significantly broaden its application. Moreover, integrating this system into AR/VR environments and smart devices can create more immersive and intuitive user experiences. In conclusion, this project not only demonstrates technical feasibility but also shows strong potential for transforming accessibility tools, gaming interfaces, and futuristic digital control systems. It marks a promising step toward more inclusive, intelligent, and touch-free computing.

REFERENCES

1. A.T. Duchowski, *Eye Tracking Methodology: Theory and Practice*, Springer, 2nd Edition, 2007.
2. D.W. Hansen and Q. Ji, "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 478–500, 2010.
3. X. Zhang, Y. Sugano, and A. Bulling, "Revisiting Data Normalization for Appearance-Based Gaze Estimation," *IEEE Transactions on Image Processing*, vol. 29, pp. 5542–5554, 2020.
4. C.H. Morimoto and M.R.M. Mimica, "Eye Gaze Tracking Techniques for Interactive Applications," *Computer Vision and Image Understanding*, vol. 98, no. 1, pp. 4–24, 2005.

5. L. Świrski and N. Dodgson, “A Fully-Automatic, Temporally Stable and Accurate Eye Tracking Method Using Pupil Centre Corneal Reflection,” *Computer Vision and Image Understanding*, vol. 116, no. 10, pp. 1060–1073, 2013.
6. A. Bulling and H. Gellersen, “Toward Wearable Eye Tracking,” *IEEE Pervasive Computing*, vol. 10, no. 2, pp. 42–50, 2011.
7. M. Vidal and J. Turner, “Evaluation of Eye Tracking in Human-Computer Interaction,” *Proceedings of ACM Symposium on Eye Tracking Research & Applications (ETRA)*, 2012.
8. A.K. Kumar, P. Verma, and S. Rao, “Eye-Tracking Mouse Control Using OpenCV,” *Proceedings of IEEE International Conference on Intelligent Systems and Applications*, 2024.
9. Karthik T., Anjali S., and Meera R., “Gaze Driven Pointer System for Smart Devices,” *International Journal of Science and Advanced Technology (IJSAT)*, vol. 13, no. 2, pp. 112–117, 2023.