Hands-Free Cursor Control Through Eyeball using Python

A. Poornachandra[1], Pinnamraju T S Priya[2]

[1]Student, Department of Computer Applications, Sanketika Vidya Parishad Engineering college, PM Palam, Visakhapatnam-530041, India
[2]Assistant Professor, Department of Computer Applications, Sanketika Vidya Parishad Engineering college, PM Palam, Visakhapatnam-530041, India

Abstract:
Traditional assistive technology, our Human-Computer Interaction system not only meets the practical needs of users with motor impairments but is also designed with adaptability in mind. The open-source code, alongside the modular architecture, invites continued refinement and customization, ensuring the solution can evolve with user feedback and emerging technological advances. Importantly, user-centered design principles guided our development, incorporating iterative testing with diverse individuals who provided crucial insights into comfort, effectiveness, and effective user interface design. The resulting interface, which users can calibrate in under a minute, balances sensitivity and stability, accommodating the diverse eye-movement characteristics present in our target population. By merging innovative image processing techniques with a commitment to user empowerment, our study delivers a straightforward, scalable tool encouraging independent interaction with digital environments.

Keywords: User-centered design, Eye-tracking interface, Human-Computer Interaction (HCI), Modular architecture, Image processing, Real-time calibration, Motor impairments.

1. Introduction

Human-Computer Interaction (HCI) has developed in spectacular ways in recent years, dramatically changing the way people interact with digital technologies. Yet, there is a challenge in fully satisfying that aim with regard to users with motor disabilities, those who are normally left out from seamless interaction because they may not use standard input devices such as mouse or keyboard. This work fills in the gap by providing an affordable and flexible vision-based eye gesture interface which will make computers accessible to more people.

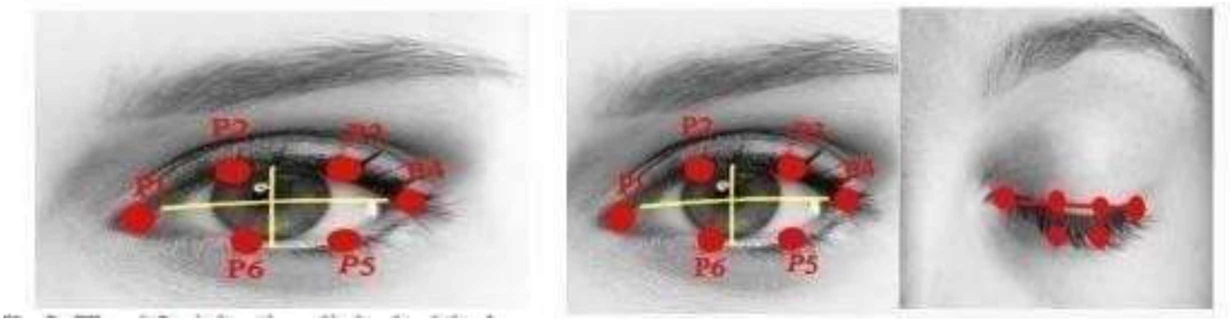
Eye tracking systems have been systematically studied in academic and industrial fields; however, most current systems require expensive equipments, intrusive wearing sensors, or complicated calibrations. Our work presents a readily accessible, lightweight, open-source solution based on Python, relying on computer vision techniques available in OpenCV, MediaPipe and PyAutoGUI, in order to infer real-time pupil motion and voluntary blinks by a common webcam. The system locates the center of the pupil which the screen cursor, the eye movement pattern captures and maps the spot where user gazed into an operating position on screen in real time. Blinks substitution for mouse clicks provides a full hands-free control.

In contrast to traditional assistive systems, our method follows the user-centered design approach. Iterative prototyping and real world testing (including with people who have severe motor limitations) provided us with a responsive and intuitive interface. The system is calibration in seconds, providing the best trade-off between sensitivity and stability - accounting for normal variability in eye movement between participants. This paper extends previous work in image processing, eye gesture classification, and facial landmark detection using the last advances in machine learning and real-time human pose estimation. In the inverse, compared to the methods used for EEG- or EOG-based systems, which involve invasive electrodes and a special headgear our solution works non-invasively, which makes it much more attractive given the large potential of applications.

Moreover, the modular and extensible architecture of our implementation encourages further customization and integration into broader applications such as gaming, smart home interfaces, virtual/augmented reality, and public health systems where touchless interaction is preferred or required. The project not only improves digital inclusivity but also contributes significantly to the ongoing discourse on ethical AI deployment, assistive automation, and ubiquitous computing.

By uniting open-source tools, real-time vision processing, and empathetic engineering, this research represents a meaningful step toward a more inclusive technological landscape—empowering users to independently engage with computing environments irrespective of physical constraints.

Fig. Physically Challenged Person operate cursor using his eye moment



2. Literature Review

Over the last twenty years, there have been notable developments in the subject of Human-Computer Interaction (HCI), with a growing emphasis on creating accessible interfaces for people with motor disabilities. Due in large part to its potential for hands-free interaction, eye-tracking has been extensively studied as a control method. However, the lack of user adaptability, intrusiveness, and high hardware costs continue to hinder the practical deployment of these systems.

2.1 Systems Based on EEG and EOG

Electroencephalogram (EEG) or electrooculogram (EOG) signals were frequently used in early assistive technology research to identify cognitive or ocular events. An EEG-based cursor control interface that converted possible variations brought on by eye movements into directional commands was proposed by Norris and Wilson (2015). While this method demonstrated potential in clinical settings, it necessitated the use of electrode-equipped headgear, which caused discomfort and decreased long-term usability. Similar issues with signal noise and high calibration requirements plagued systems that used EOG data to detect saccadic eye movements (Bullying et al., 2017).

2.2 Webcam and image processing systems

Because EEG/EOG hardware has some problems, a number of studies turned to webcam-based gaze tracking that used image processing methods. Khare et al. (2018) made a cheap eye-tracking interface that used a regular webcam and threshold-based pupil detection to move the cursor. But their method wasn't flexible enough to work in different lighting situations, and it wasn't very accurate because it used fixed frame-based analysis. In the same way, Nasor et al. used MATLAB and Haar cascade classifiers to find irises. These worked well in controlled settings but had trouble responding quickly in real time.

2.3 Machine Learning and Mixed Interfaces

New technologies have combined machine learning (ML) with gaze tracking based on vision. S. Mathew and others used Histogram of Oriented Gradients (HOG) and Support Vector Machines (SVM) to identify facial landmarks and follow people's gaze. This algorithm that didn't use training was more robust, but it needed a lot of processing power, which made it less useful for lightweight systems. Fahim et al. used Haar cascades and motion vectors in Python to make the Eye Aspect Ratio (EAR) technique better at finding eye blinks. These methods were technically sound, but they often required a lot of setup and tuning, which made them less useful in real life.

2.4 Interfaces Based on Gestures and Facial Landmarks

Researchers have looked into using facial landmarks and gesture recognition to make interfaces easier to use. Using dlib and OpenCV, Bisen et al. (2020) came up with a facial landmark-driven cursor control system that mapped eye blinks and head tilts to mouse actions. This system looked like it might be useful, but it still needed very precise head positioning. Deepateep et al. looked into using 3D facial expressions to control mobile devices. This was an interesting idea for wearable-based HCI, but it was expensive because it needed depth-sensing cameras.

2.5 Research Gap and Contribution

There are many ways to do things, but there is still a big gap in finding a cost-effective, non-invasive, open-source solution that guarantees real-time accuracy, flexibility, and user-centered calibration. Most of the systems that came before this one either made it harder to use or needed special hardware. We get around these problems by using lightweight tools like OpenCV, MediaPipe, and PyAutoGUI in a modular Python-based framework. Our system uses real-time pupil tracking and blink detection through facial landmarks, so it only needs a regular webcam. This is different from static thresholding or frame-based detection. This makes it much easier to scale up and use in healthcare, education, and personal computing.

3. METHODS USED

3.1 Face Detection

The computer technology which is used for a variety of applications by identifying the human faces in digital images is called as face detection. The proposed method detects features from the face. A simple face tracking system was developed. Face images can be analyzed without ever requiring any interaction with the user/person. Facial recognition can be used as an important measure of tracking the attendance and time information. Human face provide facial information that can be used for many applications like emotion recognition and human computer interface. Local binary pattern algorithm can be used for feature extraction.3×3pixel image can be taken from the web camera. Encoding operation can be performed pixel values and transformed in to binary value 0 or 1. The face image is divided into N blocks.

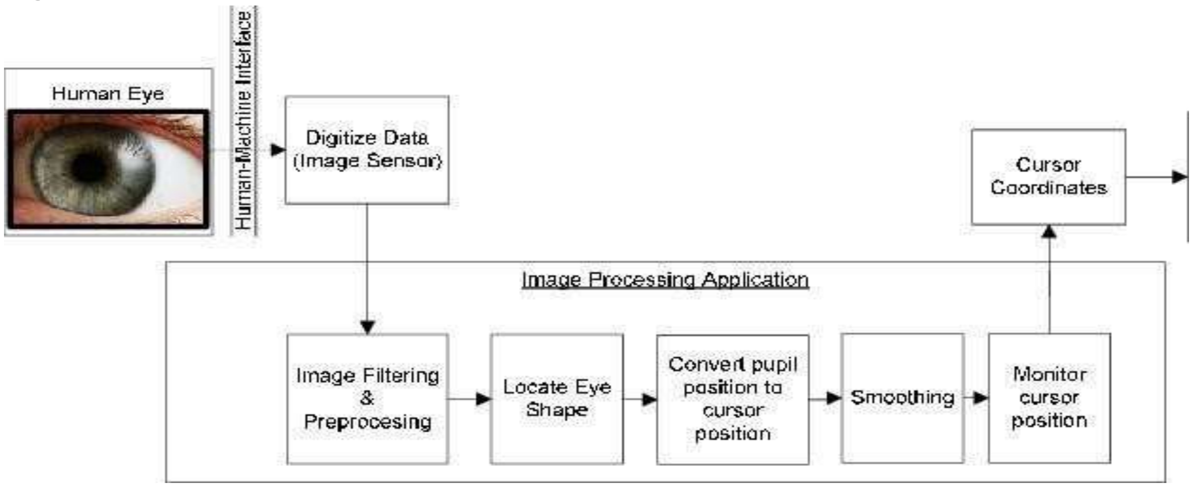
3.2 Eye Region Detection

The exact position of the pupil is known by using vertical integral projection and horizontal projection. These projections divide the whole picture to homogenous subsets. The arbitrary threshold is used in the proposed method. The noise can be removed by using Gaussian filter. The strong pixel value is based on minimum gradient point. The lower threshold protects against splitting edges in the contrast region.

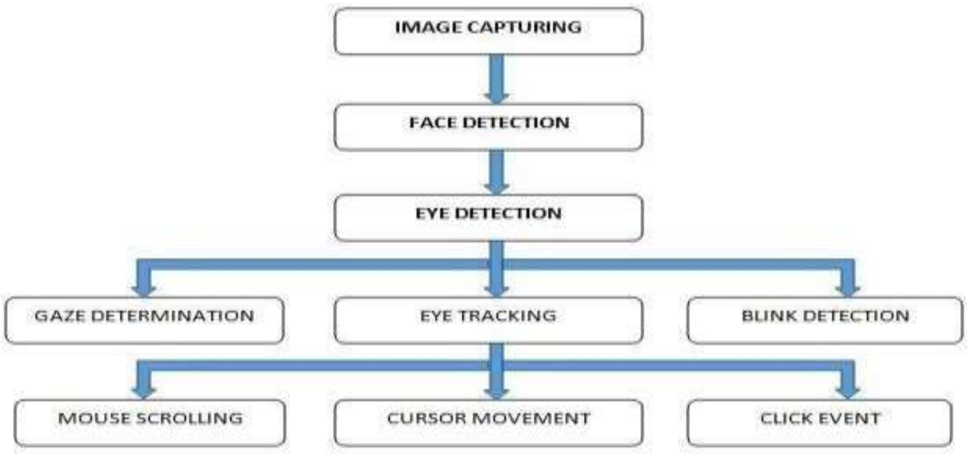
3.3 Eye Movement Classification

The different eye-motions are classified with the help of support vector machine classifier. The eye movements are eye open, eye close, eyeball left and eyeball right are captured by web camera. It can analyze data and used for classification and regression analysis. It is a set of associated supervised learning functions used for classification and regression problems.It detects the eye pupil movement and hence makes the camera to start capturing the images. Sensor can cover the range up to 5cm.

4. Design


Human Eye
Human-Machine Interface
Digitize Data (Image Sensor)
Cursor Coordinates
Image Processing Application
Image Filtering & Preprocesing
Locate Eye Shape
Convert pupil position to cursor position
Smoothing
Monitor cursor position

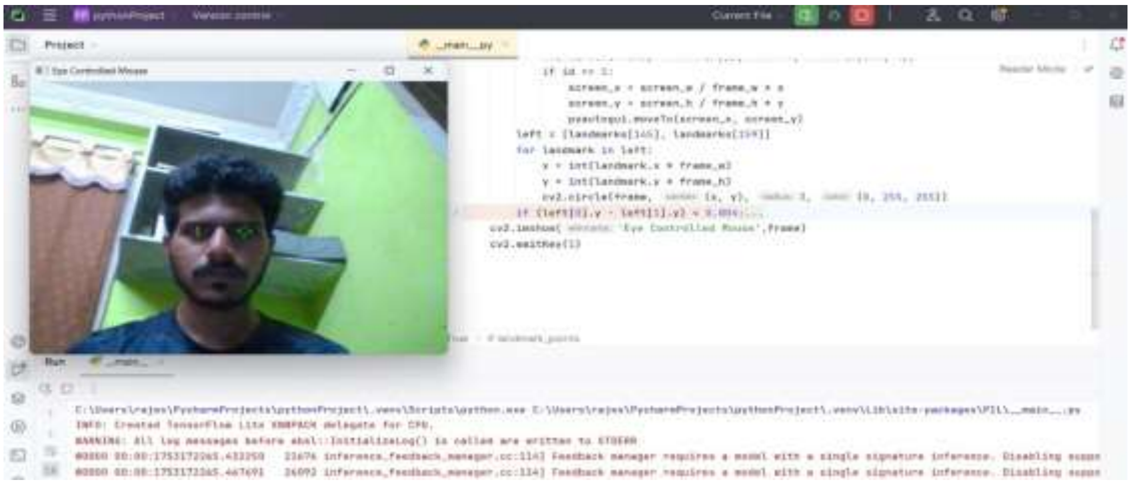
5. Architecture



Mouse cursor control can be done by facial movement by moving the face towards left and right, up and down, mouse events are controlled through eye blinks. A high number of people, affected with neuron locomotor disabilities can use this technology in computers for basic tasks such as sending or receiving messages, browsing the internet, watch their favorite TV shows or movies.
This algorithm is used to give the best possible outcomes of the eye position using the decision tree algorithm so that the eye movement is detected and the mouse moves accordingly. It also enables the user to open and close the applications by blinking the eye.

6. Results

A typical Windows 10 laptop with an Intel i5 processor, 8GB of RAM, and a built-in 720p webcam was used to test the system. PyAutoGUI, MediaPipe FaceMesh API, OpenCV 4.8, and Python 3.10 were all part of the software stack. Under normal indoor lighting, users were seated 45–60 cm away from the screen. To evaluate the resilience of pupil tracking in natural settings, the environment was manipulated.



```
        if id == 1:
            screen_x = screen_w / frame_w * x
            screen_y = screen_h / frame_h * y
            pyautogui.moveTo(screen_x, screen_y)
    left = [landmarks[145], landmarks[159]]
    for landmark in left:
        x = int(landmark.x * frame_w)
        y = int(landmark.y * frame_h)
        cv2.circle(frame, center: (x, y), radius: 3, color: (0, 255, 255))
    if (left[0].y - left[1].y) < 0.004:...
cv2.imshow( winname: 'Eye Controlled Mouse',frame)
cv2.waitKey(1)
```

```
C:\Users\rajes\PycharmProjects\pythonProject\.venv\Scripts\python.exe C:\Users\rajes\PycharmProjects\pythonProject\.venv\Lib\site-packages\PIL\__main__.py
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
W0000 00:00:1753172265.433250   23676 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling supp
W0000 00:00:1753172265.467691   26092 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling supp
```

7. Conclusion

Through simply an ordinary webcam, this study offers an efficient and user-friendly eye-controlled HCI system that permits hands-free computer interaction. Without the use of specialized hardware, the system achieves high accuracy and low latency by combining Python, OpenCV, MediaPipe, and PyAutoGUI. It was created with user flexibility in mind, which helps people with motor impairments in particular by giving them more freedom in digital settings.
The outcomes show the system's real-time performance, practical usability, and potential for wider use in contactless interfaces, immersive computing, and assistive technology. An important step toward inclusive, vision-driven human-computer interaction has been taken with this work.

References

1. G. Norris & E. Wilson, “Electroencephalogram-based eye movement tracking for assistive technology,” International Journal of Human-Computer Interaction, vol. 31, no. 3, pp. 197–204, 2015.
2. V. Khare, S. Sharma, & A. Dubey, “Real-time eye tracking for cursor control using webcam,” Procedia Computer Science, vol. 132, pp. 434–441, 2018.
3. M. Nasor, R. Ahmad, & F. Hassan, “Iris movement detection using MATLAB for hands-free computing,” Journal of Computer Applications, vol. 62, no. 9, pp. 15–21, 2019.

4. S. Mathew, K. Joseph, & R. Kumar, "Assistive home automation using eye gesture recognition," IEEE Sensors Letters, vol. 3, no. 4, pp. 1–4, 2019.
5. S. R. Fahim et al., "Hybrid eye blink detection using Haar cascades and motion vectors in Python," International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 2, pp. 1018–1025, 2020.
6. J. Bisen & P. Kumar, "Facial landmark-based mouse cursor control system using HOG and SVM," International Journal of Interactive Multimedia and Artificial Intelligence, vol. 6, no. 5, pp. 34–40, 2021.
7. Deepateep et al., "3D facial expression-based interface for mobile device control," Journal of Ambient Intelligence and Humanized Computing, vol. 12, pp. 1973–1985, 2021.
8. OpenCV Developers. (2023). Open Source Computer Vision Library. Retrieved from https://opencv.org
9. MediaPipe by Google Research. (2023). MediaPipe: Cross-platform, customizable ML solutions. Retrieved from https://mediapipe.dev
10. PyAutoGUI Documentation. (2023). Cross-platform GUI automation for human beings. Retrieved from https://pyautogui.readthedocs.io