Real-Time Eye Gaze Typing and Control of Activities for Individuals with Severe Motor Disabilities

Dr. Nandhini[1], Logeshwaran M[2], Mounish S[3], Mano Vignesh[4], Irfan[5]
[1,2,3,4,5]*Computer Science and Engineering, Sri Krishna College of Engineering and Technology Coimbatore, India*
doi.org/10.64643/IJIRTV13I2-205960-459

***Abstract*—Eye movements are vital in human-computer interaction (HCI) as they reveal attention and mental state. Video-based gaze tracking has gained importance in recent years due to its non-intrusive and accurate nature. This paper proposes a cost-effective gaze-tracking system using iris movement with the MediaPipe face mesh model. A five-point calibration and regression approach are applied to predict gaze points, while z-index tracking supports real-time re-calibration and compensates for small head shifts. The system costs under $25, depending on the camera, and was tested with thirteen participants. It performs reliably under different lighting and distances, with moderate impact from glasses. The average frame processing time is 0.047 seconds, achieving 1.12° accuracy with head movement and 1.3° without, showing its potential as a practical and efficient HCI tool.**

***Index Terms*—Gaze tracking, Iris movement tracking, Media Pipe face mesh, 5-point calibration, Multiple regression, Z-index tracking.**

I. INTRODUCTION

Eye tracking has become very popular in many research areas over the past few years. This technology has many different uses, which is why researchers from different fields are interested in it. One area where eye tracking works really well is in human-computer interaction (HCI). People with serious physical disabilities can communicate better with others by using their eyes to control computers [1]. Having eye-based systems is very important because they help people with special needs, especially those who are disabled. These systems help them live better lives and interact with their surroundings more easily [2]. Researchers have come up with several ways to build eye tracking systems that work well.

Using eye tracking to understand how people use computers has grown a lot lately. But the early methods were not very user-friendly. They needed things like special contact lenses with coils [3] or electrodes attached to the skin [4], which made users uncomfortable and were hard to use. These old methods also had problems working in real-world situations. Some systems used special headgear [5] to track where people looked. While this was better than the earlier methods, it was still not practical for everyday use because people had to wear special equipment that got in the way of normal activities and made them feel awkward. Compared to these older ways, video-based eye tracking is much better. It's practical, works well, costs less, and doesn't bother users. People can track their eye movements accurately without needing uncomfortable equipment [6].

Many video-based eye tracking systems have been created recently. However, they cost too much money, which stops most people from using them. Some examples include Tobii Rex, ITU Gaze Tracker [7], and The Eye Tribe [8]. These systems have several problems: they're expensive, need special equipment you wear, are hard to set up, require users to sit at exact distances from the screen, and don't always work correctly. For example, Tobii Dynavox eye trackers cost between $5,000 and $10,000 depending on which model you buy [9], [10]. The ITU Gaze Tracker stops working properly when you move your head closer or farther from the camera. The Eye Tribe system becomes less accurate when you turn your head. Because of these issues, current eye tracking systems can't balance being fast, accurate, and reliable at the same time [8].

Most eye tracking research focuses on building communication systems, creating better algorithms, using computer vision methods, and studying facial

features to make systems work better while keeping costs low. The main challenge with cheap systems is that they must be simple and affordable, so they use less powerful hardware than expensive systems. Also, to figure out where someone is looking, the system needs to know how their head is positioned. Even though researchers have suggested many ways to do this using eye features, they're still looking for methods that let people hold their heads naturally while using the system. There are also two other big problems. First, many current methods don't work well when people hold their heads in natural positions. Second, it's hard to test these systems in unusual situations when you want them to work for everyone.

Video-based eye tracking has gotten a lot of attention recently because it can help people with disabilities live better lives. This paper describes a cheap, easy-to-use, and reliable system that works by tracking iris movement. The system finds faces in real-time video using the Media Pipe model to focus on the face area. It uses a regular webcam to calibrate where the user looks at specific points on the screen, then uses multiple regression to predict where they're looking based on this training data. Our work also includes real-time re-calibration that automatically adjusts when the user changes their body position or posture. Here are the main things our paper contributes:

1. A new, low-cost HCI system that tracks the iris, is easy to set up, uses personalized 5-point calibration, and applies regression methods.
2. Noise Reduction: We ignore the first 0.5 seconds of user gaze at each calibration point to reduce errors. Recording data when the iris isn't focused yet leads to wrong measurements.
3. Multiple Regression: Instead of using one algorithm for both X and Y coordinates, we calibrate the X and Y directions separately. This way, if one coordinate has problems, the other one can still be predicted correctly.
4. Head Movement: During calibration, we record other points on the face to measure any head movement from the starting position. We make proper adjustments (up to 15%) for any head movement that goes beyond what was recorded during calibration. This method also helps us guess the user's position even when they move outside the originally recorded area.
5. Z-index Tracking (real-time re-calibration): When the distance between the camera and user changes, it can cause problems with iris tracking. So we created a real-time re-calibration system that figures out how much the user has moved forward or backward.
6. The head movement adjustments and Z-index re-calibration give users more freedom to move around while still being able to control the screen with their eyes.

The structure of this paper is as follows: Section II discusses related studies and summarizes prior work in this area. Section III describes the proposed approach in detail. Section IV presents the experimental results, and Section V concludes the paper while outlining possible directions for future research.

II. RELATED WORKS

Eye tracking technology determines where a person is looking by monitoring iris and pupil movements. Several approaches have been developed including electrical signals (EOG), infrared light (IROG), special contact lenses with coils, and video cameras (VOG) [11]. Video-based methods are classified into three types: model-based, appearance-based, and feature-based approaches [12].

Feature-based methods extract key eye landmarks like pupil centre, iris edges, and eye corners to determine gaze direction. Aunsri and Rattarom [2] developed a low-cost HCI system using a webcam to track multiple eye features with neural networks, eliminating calibration needs. Torricelli et al. [15] used iris and corner detection with a general regression neural network (GRNN), achieving 2.40 degrees accuracy. Valenti et al. [16] combined head pose with eye features, reducing errors to achieve 2-5 degrees accuracy. Karatay et al. [17] created a real-time interface using this model for disability assistance, achieving 23.33 characters per minute typing speed with 95.12% accuracy.

Model-based methods use mathematical models of eye anatomy and movement dynamics. One study [18] proposed a calibration-free 3D head pose estimation system tracking pupil center with machine learning, achieving 3.81 degrees accuracy. Another research [19] introduced a self-calibrating model for reading applications with coarse-to-fine filtering and pattern-matching techniques converting 2D matching to 1D calculations. Researchers [20] developed a binocular tracking device for 3D displays, enabling accurate 3D

point-of-gaze estimation without additional hardware. Appearance-based methods use full eye images without explicit feature extraction to estimate gaze direction [13], [14]. Recent advances focus on machine learning and deep learning approaches that adapt to different users and handle challenging conditions like varying lighting and head movements. However, many advanced systems require expensive hardware, creating a gap between high-performance and affordable solutions that drives research toward cost-effective systems maintaining good accuracy.

III. PROPOSED METHODOLOGY

The process begins with the webcam capturing a live video stream, from which the system detects the user's face, eyes, and blinking activity. After isolating the eye region, a blink ratio is calculated by comparing its vertical and horizontal dimensions. When this ratio falls below 0.3, it is classified as a blink, whereas higher values are treated as open-eye states and not counted.

When a blink is detected, the system converts the gaze coordinates (landmark.x and landmark.y) into a position on the on-screen interface. This lets the user select a letter, word, or phrase with a blink. The chosen item is processed by the word prediction module and then sent either as a command to an Arduino board for device control or to a text-to-speech engine for voice output.

The system uses Media Pipe for precise facial landmark detection and is designed to be intuitive for users with physical disabilities. Blink detection functions as the primary control method. A simple virtual interface provides menus for typing and quick phrase selection. Once an option is chosen, the output can be directed to:

Text-to-speech for voice communication,

Word completion for faster typing, and

Home automation/IoT control for operating devices.

In this way, the system gives paralyzed users the ability to communicate and interact with their environment using only their eye movements and blinks.

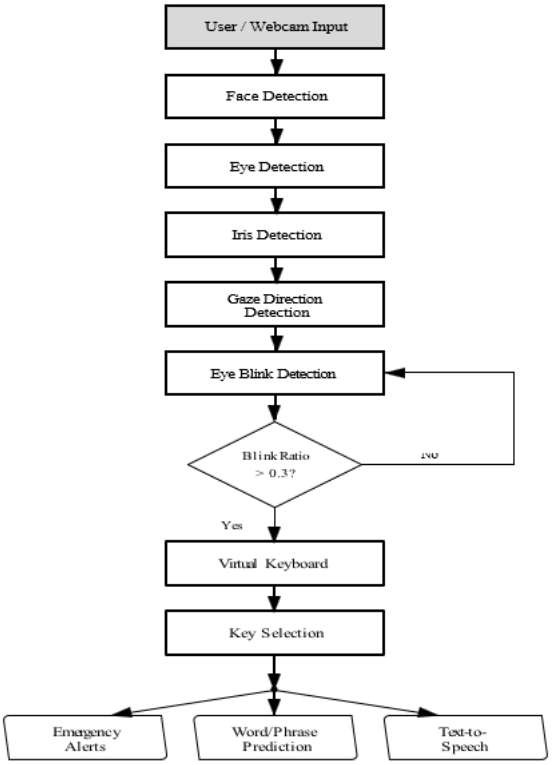

Fig.1. System flow

The system in Figure 1 is designed to support people with paralysis by allowing them to operate a computer without physical effort. It relies on eye movements for control, giving users the ability to type, speak, and manage smart devices in their surroundings. A simple blink can be used to select options or send commands to IoT appliances such as lights or fans.

The system is built from three main components:

1. Detection module – tracks the user's face, eyes, and blinks in real time.
2. Virtual interface – shows an on-screen keyboard and menus for selecting phrases or commands.
3. Output module – delivers the selected option to text-to-speech for voice communication, to word prediction for faster typing, or to smart home controls for operating devices.

A. Detecting Face Using Dense Landmarks

In contrast to the 68-point landmark model in DLib, this system adopts Media Pipe Face Mesh, which generates 468 facial landmarks in 3D space. The higher resolution increases robustness against partial occlusions and varying illumination.

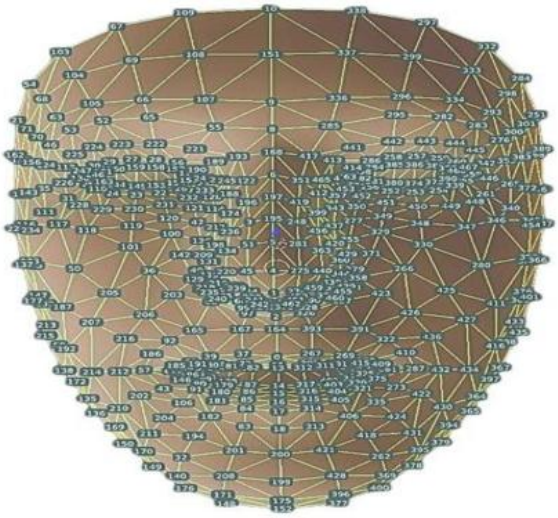

Fig. 2. Facial Landmark.

B. Eye and Iris Localization

Within F, Media Pipe Iris detects the iris centre for each eye along with four boundary points. Let the left and right iris centres be (Lx, Ly), (Rx, Ry). The midpoint of both irises is computed as:

M(x) = (Lx + Rx) / 2, M(y) = (Ly + Ry) / 2

The iris diameter D is estimated from horizontal iris landmarks using Euclidean distance:

D = sqrt ((x1 - x2) ^2 + (y1 - y2) ^2)

This serves both for depth estimation (Z-axis correction) and re-calibration.

C. Eye blink detection

Eye blink events are detected using the Eye Aspect Ratio (EAR) computed from eyelid landmarks: EAR = (||p2 - p6|| + ||p3 - p5||) / (2 * ||p1 - p4||) where p1...p6 are eye contour points. If EAR < θ (with θ ≈ 0.2), a blink is registered. The duration of closure is also considered to distinguish involuntary blinks from intentional selections.

Fig. 3. Blinking Detection

If a user blinks for about 0.3 to 0.4 seconds, which is the normal blink duration for humans, the blink ratio is taken as 0.3. If the eyes stay closed for longer than this, it is treated as closed eyelids. When the eyes are open, the vertical and horizontal lines look almost the same. But during an eye closure, the vertical line becomes shorter or disappears. The ratio is then calculated between the vertical and horizontal lines. If this ratio goes above the set threshold, the eye is considered closed; otherwise, it is marked as open.

D. Calibration and Gaze Mapping

To personalize the system, a five-point calibration is employed. Users fixate on predefined screen points S = {(sx, sy)}. Corresponding iris centers (Mx, My) are recorded, and a linear regression model maps iris coordinates to screen coordinates:

sx = αx * Mx + βx, sy = αy * My + βy Real-time re-calibration adjusts mapping when iris diameter changes, using a scaling factor: Sf = D_ref / D_obs where D_ref is the reference iris diameter and D_obs is the observed diameter.

E. Virtual Keyboard Input

The calibrated gaze is mapped to the virtual keyboard. A blink confirms the selection of the highlighted character. The typed sequence W = {w1, w2, ... wn} is updated after each selection.

F. Advanced Word and Phrase Prediction

Unlike the base model which used n-gram statistics, this work integrates transformer-based language models (LLMs) such as Mistral, Ollama, or Gemini for next-word prediction. The probability distribution over the vocabulary is defined as: P(w_{t+1} | w1, w2, ... wt) = softmax (fθ(w1, ... wt)) where fθ represents the LLM's contextual embedding function. This allows prediction of not only the next word but also context-relevant phrases, greatly reducing typing effort. device.

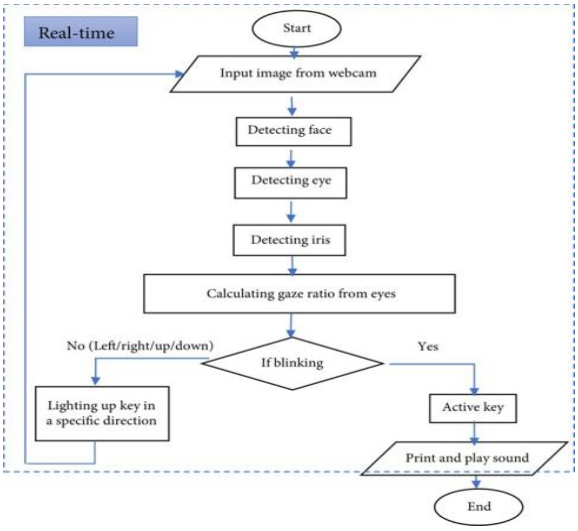

Fig. 4. Working of Virtual Keyboard

G. User Interface

The system provides three types of user interfaces. The first is the eye-typing interface, which uses a custom keyboard (Fig. 5) created with NumPy. Each key is mapped by setting its screen boundary and assigned value, allowing the user to type words and sentences through eye movements.

The second interface is for home automation, where images of household devices are shown (Fig. 6). The user can select an appliance to control it. The third is quick access, which displays common phrases and emergency messages as images on the screen (Fig. 7). The user can choose one of these options with a simple eye blink.

Fig.5. Typing by Virtual Keyboard.

H. Home Automation Integration

For IoT device control, icons representing appliances are displayed. A confirmed gaze selection triggers commands transmitted via microcontroller (Arduino/Raspberry Pi).

IV. EXPERIMENTAL RESULTS

To detect eye blinks and use them for typing through a virtual keyboard, the system is implemented in Python with the help of OpenCV, NumPy, and Media Pipe libraries. The application runs in real time and continuously records the user's actions and results.

A. Implementation

- Facial and Blink Detection:

The process starts with a webcam capturing the user's face. The video stream is analyzed using Media Pipe's Face Mesh and Iris modules to precisely locate the eyes. Since the primary objective is to enable typing through blinks, the system must distinguish between natural and intentional eye closures. As illustrated in Fig. 4, the algorithm calculates a blink ratio and compares it with a predefined threshold. If the ratio falls below the threshold, the system recognizes it as a deliberate blink and selects the corresponding key from the on-screen keyboard.

- Key Selection and Typing using Virtual Keyboard:

The user directs their gaze toward the desired key on the virtual keyboard. When they maintain a blink briefly while the cursor highlights that key, the system interprets it as a confirmation, similar to pressing the "Enter" key on a physical keyboard. The chosen character is then displayed on the screen. For example, Fig. 5 demonstrates how the system was able to type the phrase "How are you" using this method.

- Word Prediction:

Once characters are entered, the word prediction module suggests the most common words that begin with those letters in the same sequence. Manual testing showed that the module produced correct and frequently used words about 80% of the time, making typing faster and easier for the user.

B. Testing Results

The proposed system underwent thorough testing to ensure it satisfied the required performance standards. To achieve this, several unit and functional tests were carried out using Python's pytest framework. The system was considered reliable and of acceptable quality only after all tests were successfully completed.

In addition to software testing, the system's usability was evaluated with a trial involving a randomly selected group of participants. Their interaction with the system helped assess its practicality for real-world use. A summary of the experimental findings is presented in Table I.

Table I. Result Table

Sl. No	Name	Results
1	Total gaze	1598
2	Total eye blink	2178
3	Missed false positive	82
4	Missed blinks	210
5	Accuracy	95

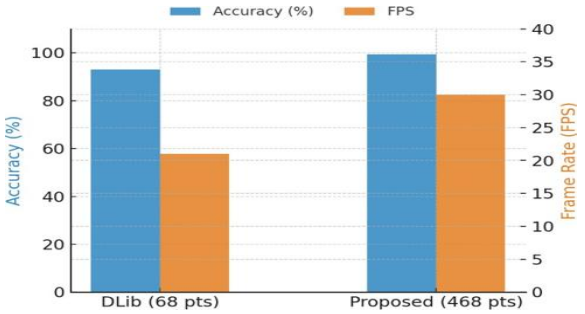

Fig 6. Face and Eye Detection Performance

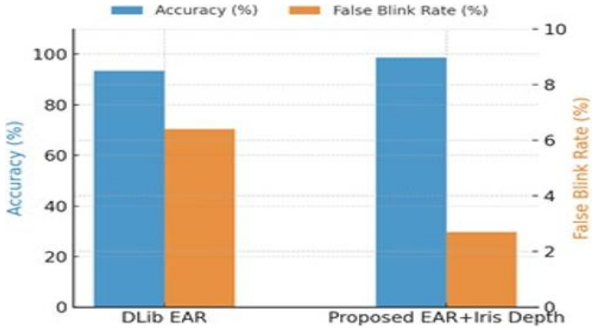

Fig 7. Blink Detection Comparison

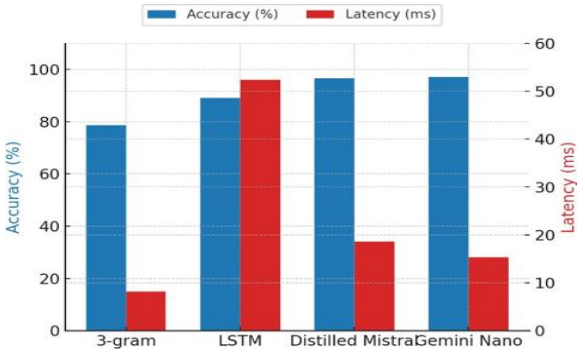

Fig 8. Transformer Models Comparison

V. CONCLUSION AND FUTURE WORK

This model introduced a real-time iris-based cursor control system for human–computer interaction (HCI) using only a standard webcam. By employing the Media Pipe Face Mesh model, a 5-point calibration method, and regression-based learning, the system achieves accurate and low-cost gaze tracking without requiring high-resolution eye data. The inclusion of real-time re-calibration ensures reliable performance under varying lighting conditions and natural head movements.

In future, we aim to increase accuracy with more calibration points and explore CNN-based feature training. Additional planned extensions include:

- Sending email or SMS alerts to caregivers in emergencies.
- Providing voice feedback in native languages.
- Using CNN models to detect wounds or infections for healthcare support.
- Adding buzzers or alerts for quick attention.

These improvements will further enhance the system's usability, safety, and adaptability, making gaze-based interaction more practical for diverse real-world scenarios.

REFERENCES

[1] G. R. Chhimpa *et al*., "Empowering Individuals with Disabilities: A Real-Time, Cost-Effective, Calibration-Free Assistive System Utilizing Eye Tracking," *Journal of Real-Time Image Processing*, 2024.

[2] N. Aunsri and S. Rattarom, "Novel Eye-Based Features for Head Pose-Free Gaze Estimation with Web Camera," *Ain Shams Engineering Journal*, 2022.

[3] Y.-M. Cheung and Q. Peng, "Eye Gaze Tracking with a Web Camera in a Desktop Environment," *IEEE Transactions on Human-Machine Systems*, 2015.

[4] X. Zhang *et al*., "Eye Tracking Based Control System for Natural HCI," *Computational Intelligence and Neuroscience*, 2017.

[5] A. Kar and P. Corcoran, "A Review and Analysis of Eye-Gaze Estimation Systems," *IEEE Access*, 2017.

[6] R. Karatay *et al*., "A Real-Time Eye Movement-Based Computer Interface for People with Disabilities," *Smart Health*, 2024.

[7] Z. Wan *et al*., "Self-Calibrating Gaze Estimation via Matching Spatio-Temporal Reading Patterns and Eye-Feature Patterns," *IEEE Transactions on Industrial Informatics*, 2024.

[8] J. Chen and Q. Ji, "A Probabilistic Approach to Online Eye Gaze Tracking Without Explicit Personal Calibration," *IEEE Transactions on Image Processing*, 2015.

[9] L. Sun *et al*., "Real-Time Gaze Estimation with Online Calibration," *IEEE Multimedia*, 2014.

[10] N. M. Arar *et al*., "A Regression-Based User Calibration Framework for Real-Time Gaze Estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[11] R. G. Bozomitu *et al*., "Development of an Eye Tracking-Based HCI for Real-Time Applications," *Sensors*, 2019.
[12] M. Tonsen *et al*., "InvisibleEye: Mobile Eye Tracking Using Multiple Low-Resolution Cameras and Learning-Based Gaze Estimation," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2017.
[13] M. F. Ansari *et al*., "Person-Specific Gaze Estimation from Low-Quality Webcam Images," *Sensors*, 2023.
[14] M. Adnan *et al*., "A Robust Framework for Real-Time Iris Landmarks Detection Using Deep Learning," *Applied Sciences*, 2022.