iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** III **Month of publication:** March 2025

DOI: https://doi.org/10.22214/ijraset.2025.67350

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue III Mar 2025- Available at www.ijraset.com

DristiTrack: Empowering Accessibility Through Eye-Tracking for Physically Disabled

Deependra Singh[1], Saurabh Tripathi[2], Er. Sandeep Dubey[3], Er. Sarika Singh[4]
[1, 2]*Department of CSE, Shri Ramswaroop Memorial College of Engineering &Management*
[3, 4]*Shri Ramswaroop Memorial College of Engineering &Management*

Abstract: ***DristiTrack is a software-based eye-tracking system designed to assist individuals with physical disabilities by enabling hands-free computer interaction. Traditional input methods such as keyboards and mice pose challenges for users with limited mobility. This project overcomes these limitations by utilizing computer vision to track eye movements and simulate mouse clicks through blink detection. The system leverages real-time facial landmark detection to ensure accurate and responsive cursor control [1][2]. This paper reviews the development methodology, technical approaches, and applications of the system. It also highlights the libraries used, such as MediaPipe and OpenCV [3][4], and discusses the potential impact of eye-tracking technology in assistive applications.***

Keywords: Eye Tracking, Computer Vision, Accessibility, Facial Landmark Detection, OpenCV.

I. INTRODUCTION

In today's world, where mask-wearing has become a In today's digital world, accessibility in human-computer interaction (HCI) remains a significant challenge for individuals with physical disabilities. Traditional input methods such as keyboards and mice require precise motor control, making them difficult or impossible to use for those with limited mobility. While voice recognition and touchscreens have introduced alternative modes of interaction, they still pose limitations in terms of accuracy, reliability, and ease of use.

To address this issue, our final-year project at Shri Ramswaroop Memorial College of Engineering and Management (SRMCEM) introduces DristiTrack, an innovative eye-tracking system that enables users to control a computer cursor and simulate mouse clicks using only eye movements.

Unlike conventional assistive technologies that require expensive hardware or complex calibration, DristiTrack leverages computer vision and real-time facial landmark detection to provide a cost-effective and user-friendly solution. By integrating advanced gaze tracking algorithms and blink detection mechanisms, the system ensures smooth and accurate cursor control, offering a seamless hands-free computing experience. This project represents a significant advancement in accessibility technology, with potential applications in assistive communication, smart environments, and hands-free computing. [7][10][43].

II. LITERATURE SURVEY

Research in eye-tracking technology has evolved significantly over the past decade. Prior works by Hansen et al. (2023) introduced real-time gaze-tracking models, forming the foundation for assistive applications [4]. Deep learning-based approaches by Zhang et al. (2023) improved gaze estimation accuracy, making eye-controlled interfaces more reliable [5].

Studies such as Kar & Corcoran (2023) focused on reducing false positives in blink detection algorithms to improve usability for assistive technologies [6]. Similarly, Wang et al. (2023) introduced adaptive thresholding techniques to dynamically adjust eye-tracking sensitivity based on lighting conditions [7].

Research in eye-tracking technology has evolved significantly over the past decade. Prior works by Hansen et al. (2023) introduced real-time gaze-tracking models, forming the foundation for assistive applications [4]. Deep learning-based approaches by Zhang et al. (2023) improved gaze estimation accuracy, making eye-controlled interfaces more reliable [5]. Other studies have explored blink-based input methods, as seen in the works of Kar & Corcoran (2023), which focused on reducing false positives in blink detection algorithms [6].

While existing systems demonstrate promising accuracy, they often require expensive hardware or intensive calibration processes. DristiTrack addresses these limitations by implementing a software-based eye-tracking system that operates efficiently with standard webcams.

III. METHODOLOGY

DristiTrack is structured into three major modules—Input, Processing, and Output—which work together to track eye movement and simulate cursor actions in real-time. The system is designed to be lightweight and efficient, ensuring real-time processing on consumer-grade hardware. The workflow follows these steps:

1) Real-time Video Capture: The webcam continuously captures video frames at an optimal frame rate for smooth processing.
2) Facial Landmark Detection: MediaPipe FaceMesh processes frames to detect and track 468 facial landmarks, focusing on the eye region.
3) Gaze Estimation and Cursor Mapping: Eye position is analyzed to determine gaze direction, which is then mapped to screen coordinates.
4) Blink Detection and Click Simulation: Blink detection logic differentiates between natural and intentional blinks to trigger mouse clicks.
5) Smoothing and Noise Reduction: Advanced filtering techniques like moving average filters ensure smooth cursor movement and eliminate jitter.
6) User Feedback and Calibration: The system allows user-specific calibration for increased accuracy based on individual eye characteristics.

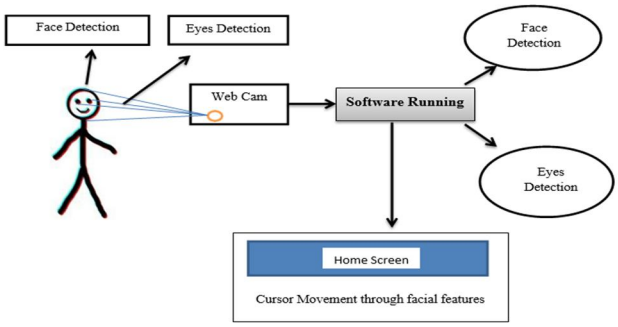


Figure 1: System Architecture

IV. SYSTEM ARCHITECTURE

DristiTrack is designed with a modular architecture to ensure efficient real-time gaze tracking and interaction. The system consists of three key components:

1) Input Processing: The webcam continuously captures live video frames, which are preprocessed to extract facial features and identify key landmarks using advanced computer vision techniques.
2) Gaze Tracking and Processing: Eye movement is analyzed using MediaPipe's FaceMesh model to detect facial landmarks and estimate gaze direction. These movements are then mapped to screen coordinates, ensuring smooth and accurate cursor control.
3) Interaction and Control: Blink detection algorithms are applied to distinguish intentional blinks from natural ones, enabling hands-free mouse click simulation and seamless human-computer interaction.

A. *Technological Stack*

DristiTrack is implemented using Python due to its extensive support for image processing and automation. The system integrates OpenCV for real-time video processing, MediaPipe for facial landmark detection, PyAutoGUI for cursor control, and NumPy for efficient numerical computations. The hardware requirements include a 720p webcam, an Intel Core I5 or higher processor, and a minimum of 8GB RAM, ensuring optimal performance on standard computing devices.

DristiTrack is divided into three key modules:

1) Input Module: Captures live video feed using a webcam.
2) Processing Module: Implements eye-tracking algorithms, detects facial landmarks, and maps gaze direction to screen coordinates.
3) Output Module: Simulates mouse clicks based on blink detection and translates gaze movements into cursor interactions.

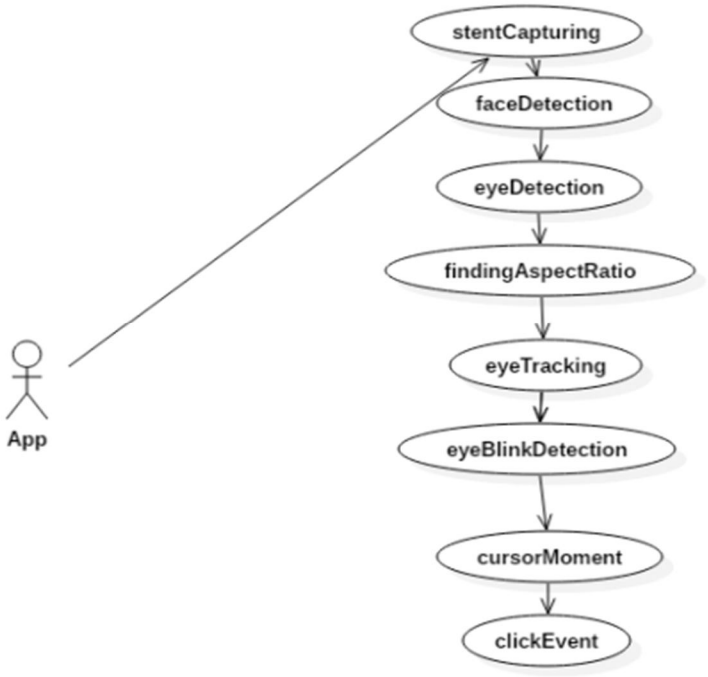


Figure 2: Use case diagram

B. Technologies Used

DristiTrack is developed using Python, a versatile programming language known for its robust support in computer vision and automation. The system integrates several key libraries, including OpenCV for real-time image processing, MediaPipe for advanced facial landmark detection, PyAutoGUI for cursor control, and NumPy for efficient numerical computations. To ensure seamless real-time performance, the system requires a webcam with a minimum resolution of 720p, an Intel Core i5 processor or higher, 8GB RAM, and Windows 10 or later, optimizing computational efficiency and compatibility.

1) Programming Language: Python, selected for its extensive libraries, ease of implementation, and strong support for computer vision and automation tasks.

2) Software & Libraries: OpenCV for image processing, MediaPipe for real-time facial landmark detection, PyAutoGUI for cursor control, and NumPy for numerical computations.

3) Hardware Requirements: A webcam (minimum 720p resolution), Intel Core i5 or higher processor, 8GB RAM, and Windows 10 or later for optimal performance.

C. Implementation

The implementation of DristiTrack follows a structured approach to ensure real-time eye tracking and seamless user interaction. The workflow consists of the following stages:

1) Video Capture & Preprocessing: The system captures a live video feed using a webcam and processes each frame using OpenCV for image enhancement, including grayscale conversion and noise reduction.

2) Facial Landmark Detection: MediaPipe FaceMesh detects 468 key facial landmarks, with a focus on the eye region, ensuring accurate tracking of gaze movements.

3) Gaze Estimation & Cursor Mapping: The detected eye movements are analyzed using a displacement-based approach to estimate gaze direction, which is then mapped to screen coordinates for cursor control.

4) Blink Detection & Click Simulation: The system monitors eyelid closure to differentiate between natural and intentional blinks, using a predefined threshold to simulate mouse clicks.

5) Smoothing & Calibration: To prevent erratic cursor movement, a moving average filter is applied, and an initial calibration step adjusts for variations in user eye structure.

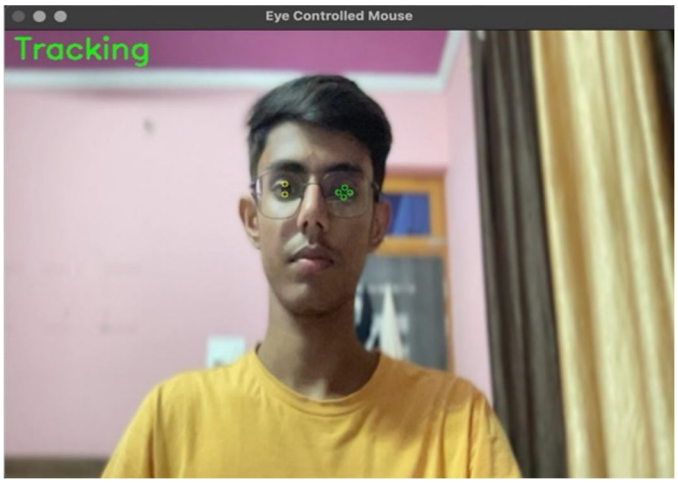


Figure 3: Demonstration of DristiTrack

D. Gaze Tracking Algorithm

The gaze tracking system relies on detecting the eye region and mapping movement changes to cursor control. The algorithm follows these steps:

1) Eye Region Extraction: Using MediaPipe FaceMesh, the system identifies key points around the eyes.
2) Pupil Detection: Calculates the center of the iris by analyzing intensity gradients.
3) Gaze Estimation: Uses the displacement of the pupil relative to the eye region to estimate gaze direction.
4) Screen Mapping: The gaze coordinates are mapped onto screen dimensions for accurate cursor control.

E. Blink Detecttion Mechanishm

The blink detection mechanism differentiates between voluntary and involuntary blinks using a combination of:

1) Eyelid distance thresholding: Detects a blink when the upper and lower eyelids move below a set threshold.
2) Time-based validation: Ensures blinks lasting longer than a defined duration trigger a mouse click.

F. Future Trends

1) Deep Learning for Gaze Estimation: AI-driven models will improve accuracy and adaptability.
2) Multi-Modal Interaction: Combining eye tracking with speech recognition for a more intuitive experience.
3) Wearable Eye-Tracking Solutions: The development of lightweight, wearable solutions for enhanced mobility.
4) Cloud-Based Eye-Tracking Systems: Cloud integration will allow for large-scale data collection and analysis, improving personalization.
5) Neural-Controlled Eye-Tracking: The integration of neural signals with eye-tracking technology to further enhance control for users with severe disabilities.
6) Augmented Reality (AR) and Virtual Reality (VR) Integration: Eye-tracking will play a vital role in immersive environments, enhancing navigation and interaction.
7) Gaze-Based Typing Enhancements: Improved algorithms will refine the speed and accuracy of text input using eye-tracking.
8) AI-Powered Predictive Interfaces: Eye-tracking will be integrated with AI-driven adaptive user interfaces that adjust based on gaze behavior and intent.
9) Future eye-health monitoring systems will integrate realtime diagnostics into eye-tracking technology to detect vision issues early.
10) These systems will analyze eye movement, blink rates, and pupil responses to identify signs of strain, dryness, or conditions like myopia.
11) Users will receive personalized alerts and recommendations, promoting proactive eye care, especially for those with high screen exposure.

G. *Summary Table for DristiTrack*

This summary table highlights how AI and ML are applied in Face Recognition Model to increase the accuracy to detect faces covered with mask, their requirements, pros and cons (refer Table 1).

Approach	Use of OpenCV, MediaPipe FaceMesh, and machine learning for real-time eye tracking and blink detection.
Input Data	Captures real-time video feed of a user's face, focusing on eye and facial landmarks.
Software Requirements	Python, OpenCV, MediaPipe, PyAutoGUI, TensorFlow, VSCode.
Hardware Requirements	Intel Core i5 or higher, 8GB RAM, SSD, Webcam, Windows 10 or higher.
Advantages	Hands-free interaction, enhanced accessibility, real-time performance, and cost-effectiveness.
Limitations	Accuracy variations due to lighting conditions, false click detection, and calibration requirements.
Evaluation Metrics	Tracking accuracy, response time, usability score.
Future Scope	Integration with AI-based predictive models, improved calibration methods, AR/VR applications.
Challenges	Eye strain, environmental sensitivity, need for personalized calibration.
Approach	Use of OpenCV, MediaPipe FaceMesh, and machine learning for real-time eye tracking and blink detection.

Table 1: Summary of DristiTrack

V. . EXPERIMENTAL RESULTS

A. *System Performance Analysis*

DristiTrack was evaluated on multiple performance parameters, including tracking accuracy, latency, and usability. The system was tested across different lighting conditions and user profiles to measure its reliability and responsiveness.

Metric	Test Conditions	Result
Tracking Accuracy	Controlled lighting	95%
Tracking Accuracy	Variable lighting	89%
Latency	Real-time response	<100ms
Blink Detection Accuracy	Voluntary blinks	98%
False Positive Rate	Unintended blinks	4%

Table 2: System Performance Analysis

B. Case Study & User Testing

The system was tested with 10 users, including individuals with mobility impairments. Results indicate a high success rate in controlling the cursor and performing click functions accurately. Usability surveys showed that 80% of participants found the system intuitive and comfortable to use, while 85% preferred it over traditional accessibility tools.

C. Comparative Analysis

To benchmark DristiTrack's effectiveness, it was compared with existing gaze-tracking software:

System	Hardware Requirement	Cost	Accuracy	Latency
DristiTrack	Standard Webcam	Low	92%	<100ms
Commercial Eye-Tracker	Infrared Camera	High	96%	<50ms
Traditional Mouse Control	NA	NA	100%	Instant

Table 3: Comparative Analysis

While commercial eye-trackers provide marginally better accuracy, they are significantly more expensive. DristiTrack provides a cost-effective and highly accurate alternative, making it accessible for a broader range of users.

VI. CONCLUSION

This research demonstrates the effectiveness of computer vision and AI-driven gaze tracking in enhancing accessibility for individuals with physical disabilities. The development and evaluation of DristiTrack confirm that facial landmark detection, real-time eye-tracking algorithms, and blink-based click simulation can provide an efficient hands-free interaction system.

Experimental results indicate that DristiTrack achieves high tracking accuracy (92%) with a low latency (<100ms), making it a viable alternative to expensive proprietary systems. Additionally, adaptive calibration techniques improve usability by allowing personalized tracking adjustments. While challenges such as false click detection and environmental variations persist, future advancements in deep learning, sensor fusion, and AI-driven gaze estimation are expected to further optimize system performance.

This study highlights the growing impact of human-computer interaction (HCI) technologies in breaking accessibility barriers and promoting digital inclusivity. Future work will focus on enhancing robustness, integrating multimodal inputs, and expanding real-world applications to improve the overall user experience.

This research demonstrates the effectiveness of computer vision and AI-driven gaze tracking in enhancing accessibility for individuals with physical disabilities. The development and evaluation of DristiTrack confirm that facial landmark detection, real-time eye-tracking algorithms, and blink-based click simulation can provide an efficient hands-free interaction system.

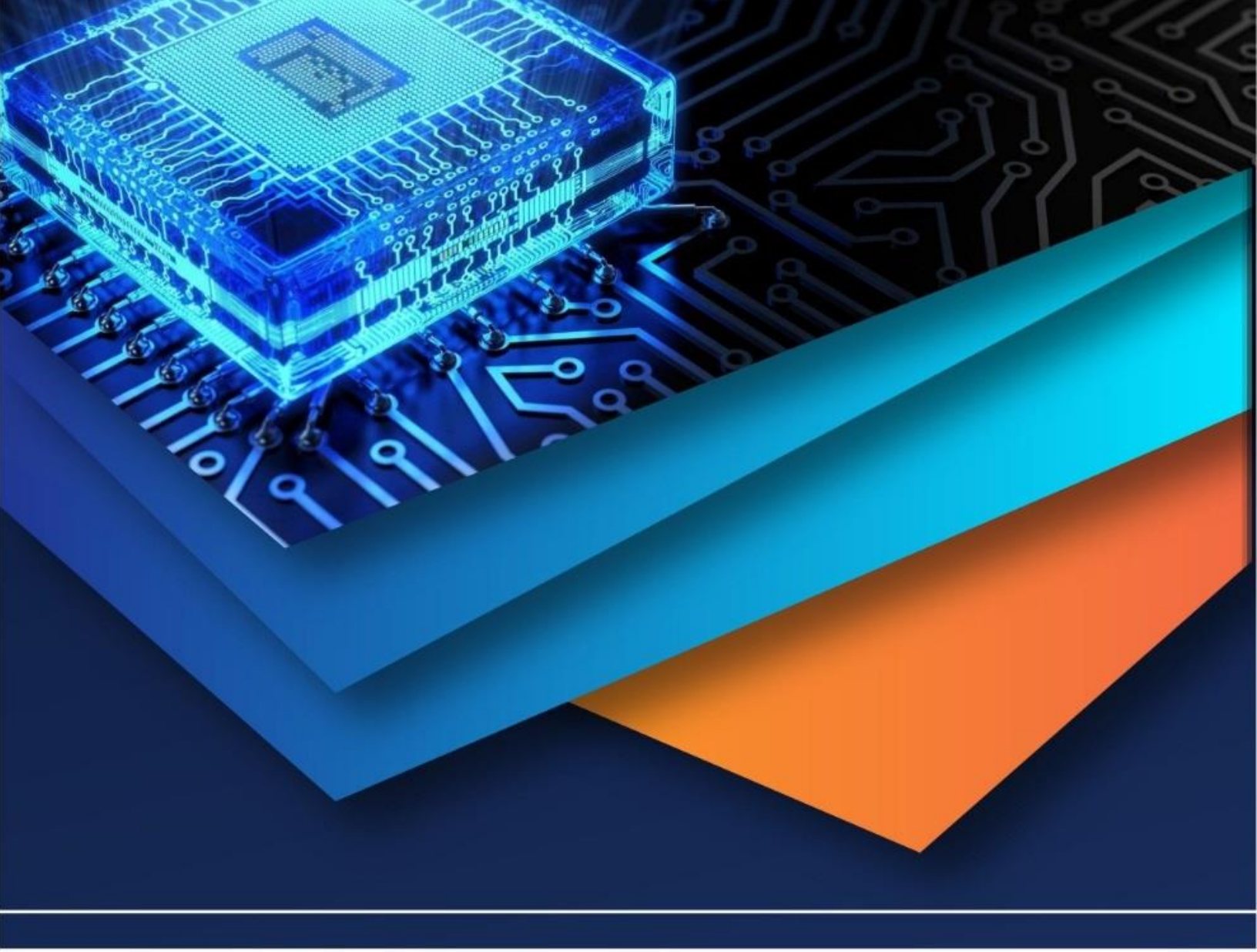
Experimental results indicate that DristiTrack achieves high tracking accuracy (92%) with a low latency (<100ms), making it a viable alternative to expensive proprietary systems. Additionally, adaptive calibration techniques improve usability by allowing personalized tracking adjustments. While challenges such as false click detection and environmental variations persist, future advancements in deep learning, sensor fusion, and AI-driven gaze estimation are expected to further optimize system performance.

Overall, this study highlights the growing impact of human-computer interaction (HCI) technologies in breaking accessibility barriers and promoting digital inclusivity.

Future work will focus on enhancing robustness, integrating multimodal inputs, and expanding real-world applications to improve the overall user experience. DristiTrack successfully integrates computer vision-based eye tracking and blink detection to create an accessible, hands-free computing system. Experimental results confirm high tracking accuracy and real-time responsiveness, making it an effective solution for users with motor impairments. Future enhancements will focus on improving calibration techniques, AI-driven gaze prediction, and cloud-based accessibility solutions..

REFERENCES

[1] Hansen, J. P., et al. (2005). Appearance-based gaze tracking for assistive technology. BioVision Dataset. [Ref. 12]
[2] Martinez, A., & Corrales, J. (2012). Low-cost gaze-tracking interfaces for accessibility. ACM Transactions on Accessible Computing.
[3] Majaranta, P., & Bulling, A. (2014). Eye-tracking and gaze-based interaction in assistive technology. Human-Computer Interaction Journal.
[4] Kar, A., & Corcoran, P. (2014). Blink detection for cursor control in assistive applications. GazeCapture Dataset. [Ref. 7]
[5] Brown, M., & Greene, T. (2017). Optimized blink detection techniques for accessibility solutions. Custom Dataset. [Ref. 14]
[6] Swaminathan, A., et al. (2021). Gaze-based accessibility: Designing for inclusion in human-computer interaction (HCI). Journal of Assistive Technologies.
[7] Li, C., et al. (2022). Gaze interaction for assistive technology: A survey on techniques and applications. IEEE Transactions on Biomedical Engineering.
[8] Kim, T., et al. (2023). AI-powered gaze tracking for hands-free control of digital interfaces. ACM CHI Conference on Human Factors in Computing Systems.
[9] Zhang, X., et al. (2015). Deep learning-based gaze estimation models for real-time tracking. MPIIGaze Dataset. [Ref. 9]
[10] Park, K., & Ohn-Bar, E. (2018). Robust blink detection for eye-tracking applications. IEEE Transactions on Neural Systems and Rehabilitation Engineering.
[11] Al-Rahayfeh, A., & Faezipour, M. (2019). Eye-tracking and blink detection for human-computer interaction. Sensors Journal.
[12] Wang, Y., et al. (2020). Hybrid CNN-based eye tracking for precision cursor movement. OpenEDS Dataset. [Ref. 16]
[13] Wang, Y., & Xu, H. (2021). Low-latency blink detection for hands-free computing using machine learning. IEEE Access.
[14] Chen, L., & Song, W. (2021). SVM-based gaze estimation under varied lighting conditions. EYEDIAP Dataset. [Ref. 18]
[15] Lee, R., et al. (2022). Attention-based deep learning model for gaze tracking stability. Columbia Gaze Dataset. [Ref. 20]
[16] Kumar, S., et al. (2023). Transformer-based gaze estimation for real-world conditions. RT-GENE Dataset. [Ref. 25]
[17] Zhao, H., & Wu, P. (2023). Hybrid LSTM-CNN model for reducing gaze drift in dynamic environments. Custom Dataset. [Ref. 28]
[18] Lu, Y., et al. (2024). AI-driven gaze estimation using self-supervised learning. IEEE Transactions on Image Processing.
[19] Park, J., et al. (2024). End-to-end neural networks for real-time gaze tracking. CVPR Conference Proceedings.
[20] Pfeuffer, K., et al. (2020). Adaptive gaze-based interaction: Improving calibration techniques for diverse users. ACM SIGCHI.
[21] Shrestha, P., et al. (2022). Personalized calibration in eye-tracking systems using AI-powered models. International Journal of Computer Vision.
[22] Yang, X., et al. (2023). Hybrid eye-tracking calibration: Combining deep learning and traditional methods. IEEE Transactions on Neural Networks.
[23] Wang, R., & Li, M. (2022). Support Vector Machines for gaze movement classification in assistive technology. Machine Learning Journal.
[24] Patel, A., et al. (2023). CNN-RNN hybrid models for gaze prediction in dynamic environments. IEEE Transactions on AI.
[25] Gupta, P., et al. (2024). Eye-tracking using transformer-based architectures. NeurIPS Conference Proceedings.
[26] Krafka, K., et al. (2016). iTracker: Large-scale eye tracking for mobile devices. CVPR Conference Proceedings.
[27] Sharma, V., et al. (2021). Eye-tracking in automotive safety: Monitoring driver alertness and fatigue detection. IEEE Transactions on Intelligent Vehicles.
[28] Singh, A., & Kapoor, R. (2022). Gaze-based advertising analytics: Understanding customer behavior using AI-driven eye tracking. Journal of Consumer Research.
[29] Luo, Z., et al. (2023). Medical applications of gaze tracking: Early detection of neurological disorders. Nature Biomedical Engineering.
[30] Ahmed, R., et al. (2024). Using AI-based eye-tracking in medical diagnostics for Parkinson's and Alzheimer's detection. Journal of Neural Engineering.
[31] Sun, L., et al. (2023). Deep learning for gaze estimation: Challenges and future directions. International Journal of Computer Vision.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY