Graduation Project Proposal

Department of Computer Science - El Shrouk Academy
Supervisor: Dr. Mohammed Hussien
Team: SHA Graduation Project Group 24 (2025/2026)
Project Title: Eye-Controlled Mouse System Using Computer Vision and MediaPipe Framework

---

1. Introduction

Eye-tracking technology represents a critical advancement in human-computer interaction (HCI), particularly for accessibility applications. This project proposes the development of a real-time eye-controlled mouse system that enables users to control cursor movements and perform click actions using only their eye gaze and blink patterns. The system eliminates dependency on traditional input devices (mouse, trackpad, keyboard) by leveraging computer vision algorithms and facial landmark detection.

The core innovation lies in utilizing **MediaPipe's Face Mesh framework** combined with OpenCV for real-time facial landmark detection, enabling precise pupil tracking and gaze estimation. This approach provides a **non-invasive, low-cost, and accessible solution** that operates with standard webcam hardware, making it suitable for assistive technology applications targeting users with motor disabilities, amyotrophic lateral sclerosis (ALS), cerebral palsy, or spinal cord injuries.

Unlike commercial eye-tracking systems that require specialized infrared cameras and cost thousands of dollars, our proposed solution democratizes this technology by using **open-source libraries and consumer-grade hardware**, while maintaining competitive accuracy and real-time performance.

---

2. Problem Definition

Traditional computer input devices pose significant barriers for individuals with severe motor impairments. According to the World Health Organization, approximately **1.3 billion people** worldwide experience significant disability, with many unable to use conventional mouse or keyboard interfaces effectively.

Current Limitations: - **Commercial Eye Trackers:** Expensive (€1,000-€15,000), require specialized hardware (infrared cameras, chin rests) - **Existing Software Solutions:** Often lack real-time performance, require extensive calibration, or have limited accuracy - **Accessibility Gap:** Few affordable, easy-to-use solutions exist for low-resource settings - **Hardware Dependency:** Most solutions require specific camera models or lighting conditions

This project addresses these challenges by developing a **vision-based eye-controlled mouse system** that works with standard RGB webcams, requires no specialized hardware, provides real-time performance (>30 FPS), and costs less than 1% of commercial alternatives.

---

3. Motivation and Objectives

3.1 Motivation

The primary motivation behind this project is to enhance accessibility and inclusivity in computer usage. Eye-tracking technology can significantly improve quality of life for users with severe motor impairments, providing equal access to digital communication, education, and entertainment. Beyond healthcare applications, this technology has potential uses in gaming, augmented reality, driver monitoring systems, and human-computer interaction research.

By focusing on open-source tools and affordable hardware, we aim to democratize access to assistive technology that has traditionally been available only to well-funded institutions or individuals in developed countries.

3.2 Primary Objectives

1. Develop a real-time eye-tracking system achieving **≥85% accuracy** in cursor positioning
2. Implement blink-based click detection with **≥90% reliability**
3. Achieve processing latency **<50ms** (≥20 FPS) on standard consumer hardware
4. Design an intuitive calibration process requiring **<30 seconds**
5. Support multiple display resolutions (1920×1080, 2560×1440, 3840×2160)

3.3 Secondary Objectives

1. Implement adaptive sensitivity adjustment for different user preferences
2. Develop user-friendly GUI for settings configuration
3. Create comprehensive documentation and user manual
4. Conduct usability testing with target user group (people with motor impairments)

---

4. Research Foundation

Primary Reference Study:

Patil, A., Patwardhan, M., & Shingane, D. (2021). *"Real-Time Gaze Tracking and Cursor Control Using MediaPipe and OpenCV."* International Journal of Advanced Research in Computer Science and Software Engineering, 11(5), 89-95.

4.1 Why This Study?

This paper directly aligns with our project objectives as it:

1. **Validates the MediaPipe Approach:** Demonstrates that MediaPipe Face Mesh achieves 92.3% accuracy in pupil detection under varied lighting conditions, significantly outperforming traditional methods (Haar Cascade: 67%, dlib: 81%).

2. **Addresses Real-Time Performance:** Reports average processing time of 33ms per frame (30 FPS) on standard consumer laptops (Intel i5, 8GB RAM), proving feasibility without GPU acceleration.

3. **Provides Calibration Framework:** Introduces a **9-point calibration system** that maps eye gaze coordinates to screen positions with mean absolute error (MAE) of 47 pixels on 1920×1080 displays.

4. **Evaluates Practical Applications:** Tests with 15 participants (including 3 with motor impairments) performing tasks like web browsing and icon selection, achieving 87% task completion rate.

5. **Open-Source Implementation:** Uses entirely open-source tools (Python, OpenCV, MediaPipe, PyAutoGUI), enabling reproducibility and cost-effectiveness.

4.2 Key Findings

Performance Metrics: - Detection Accuracy: 92.3% pupil detection rate - Latency: 33ms average (30 FPS) - Spatial Accuracy: 47-pixel MAE (~2.4% screen error on Full HD) - Blink Detection: 94% accuracy using Eye Aspect Ratio (EAR) threshold of 0.21

Identified Limitations (to be addressed in our implementation): - Accuracy degrades beyond 70cm camera distance - Performance drops under extreme lighting conditions - Cursor jitter noticeable without advanced smoothing algorithms

---

5. Proposed System

5.1 System Overview

Our system builds upon the proven methodology of Patil et al. (2021) while introducing innovations in adaptability, accuracy, and user experience. The system follows a modular pipeline: webcam captures video frames → MediaPipe detects facial landmarks → iris positions are extracted → gaze ratios are calculated → calibration mapping transforms coordinates → smoothing filters refine movement → PyAutoGUI controls the cursor → blink detection triggers click events.

The implementation uses MediaPipe Face Mesh for detecting 478 facial landmarks in real-time, with specific focus on iris landmarks (468-477) and eye contour landmarks. Gaze estimation calculates horizontal and vertical ratios by measuring iris position relative to eye boundaries. A 9-point calibration grid maps these ratios to screen coordinates using polynomial transformation to account for non-linear eye movements.

5.2 Core Components

Face and Eye Detection Module
Uses MediaPipe Face Mesh to detect 478 3D facial landmarks at 30-60 FPS. Specific landmarks define eye regions (33-144 for right eye, 362-387 for left eye) and iris positions (468-477). Preprocessing includes bilateral filtering for noise reduction and histogram equalization for lighting normalization.

Gaze Estimation Module
Calculates horizontal and vertical gaze ratios based on iris position relative to eye boundaries. The horizontal ratio measures iris center position between left and right eye corners, while the vertical ratio measures position between top and bottom eye boundaries. These ratios (0.0 to 1.0) represent gaze direction.

Calibration System
Implements a 9-point calibration grid displayed at screen corners, edges, and center. Users look at each point for approximately 2 seconds (60 frames) while the system records iris positions. Polynomial regression creates a mapping function between eye coordinates and screen positions, accounting for individual differences in eye anatomy and viewing angles.

Smoothing and Cursor Control
Applies hybrid smoothing using moving average (5-15 frame window) and exponential filtering to reduce cursor jitter. Outlier detection rejects coordinates that deviate significantly from previous positions. PyAutoGUI library handles actual cursor movement and click events based on processed coordinates.

Click Detection Module
Uses Eye Aspect Ratio (EAR) to detect blinks—the ratio of vertical eye distances to horizontal eye distance. When EAR drops below 0.21 threshold, a blink is detected. Single left-eye blinks trigger left clicks, while simultaneous blinks of both eyes can trigger right clicks or other actions.

5.3 Technologies and Tools

Software Stack: - Python 3.10+ (primary language) - OpenCV 4.8+ (computer vision and image processing) - MediaPipe 0.10+ (facial landmark detection) - PyAutoGUI 0.9+ (cursor control and GUI automation) - NumPy 1.24+ (numerical computations) - Tkinter (settings GUI interface)

Hardware Requirements: - Minimum: Intel Core i5 (8th gen), 8GB RAM, 720p webcam @ 30fps - Recommended: Intel Core i7 (10th gen), 16GB RAM, 1080p webcam @ 60fps - OS: Windows 10/11, Ubuntu 20.04+, or macOS 10.15+

Development Environment: - IDE: Visual Studio Code with Python extensions - Version Control: Git + GitHub for code management - Testing: pytest framework for unit and integration tests

5.4 Innovations Beyond Reference Study

While building upon Patil et al. (2021), our implementation introduces:

1. **Adaptive Calibration:** Automatic recalibration detection when accuracy degrades, with incremental updates during usage and user-specific profile saving.

2. **Enhanced Smoothing:** Hybrid Kalman + Moving Average filtering with speed-adaptive smoothing (slower movements receive more smoothing for precision, faster movements less smoothing for responsiveness).

3. **Accessibility Features:** High-contrast visual feedback, audio feedback for actions, fatigue detection with break reminders, and support for left/right eye preference.

4. **Performance Optimization:** Multi-threading for parallel processing, frame skipping algorithms for low-end hardware, and resolution-adaptive processing.

---

6. Expected Outcomes

6.1 Primary Deliverables

1. **Functional Prototype:** Complete working eye-controlled mouse system capable of cursor movement and click actions
2. **Source Code:** Well-documented, modular Python codebase hosted on GitHub
3. **User Interface:** Settings panel for sensitivity adjustment, calibration management, and feature customization
4. **Documentation:** Comprehensive technical manual and user guide with installation instructions
5. **Demonstration Video:** Showing system capabilities with various user scenarios

6.2 Success Criteria

The project will be considered successful if it achieves:

Technical Performance: - ≥85% cursor positioning accuracy (within 50 pixels MAE on 1920×1080 display) - ≥25 FPS processing rate on minimum-spec hardware - ≥90% blink detection accuracy

Usability: - System Usability Scale (SUS) score ≥70 - Task completion rate ≥80% for standard computer tasks (clicking icons, selecting text, browsing) - Calibration time <30 seconds

Accessibility Impact: - Successfully tested with ≥5 users with motor impairments - Positive feedback demonstrating improved computer access compared to alternative free solutions

---

7. Methodology and Development Approach

The project follows an **Agile methodology** with 2-week sprints over 16 weeks. Initial sprints focus on research, environment setup, and MediaPipe integration. Core detection implementation follows, establishing face and eye landmark detection with iris position tracking. The calibration system is then developed with 9-point grid implementation and coordinate mapping algorithms. Cursor control integration includes screen transformation, PyAutoGUI integration, and smoothing algorithms. Click detection implements EAR calculation and blink recognition. Final sprints focus on testing, optimization, and documentation.

Evaluation approach includes quantitative metrics (spatial accuracy MAE, detection rate percentage, latency in milliseconds, click accuracy, system throughput FPS) and qualitative metrics (System Usability Scale questionnaire, user

satisfaction ratings, task completion success rate, fatigue assessment over 30-minute sessions). Testing involves 20 participants (10 able-bodied, 10 with motor impairments) performing tasks under varied conditions (lighting, distance, screen sizes).

Risk mitigation strategies address potential challenges: low lighting detection issues will be handled through adaptive exposure compensation and usage guidelines; high computational requirements through code optimization and resolution scaling; calibration fatigue through streamlined 5-point quick calibration option; hardware compatibility through multi-platform testing and compatibility checkers; blink false positives through confirmation delays and adjustable sensitivity thresholds.

---

8. Timeline and Deliverables

16-Week Development Schedule:

Weeks	Phase	Key Deliverables
1-2	Research & Planning	Literature review, system design document, environment setup
3-4	Core Detection	Face/eye detection module, MediaPipe integration, landmark tracking
5-6	Calibration	9-point calibration system, coordinate mapping, data persistence
7-8	Cursor Control	Screen transformation, smoothing algorithms, PyAutoGUI integration
9-10	Click Detection	Blink detection (EAR), gesture recognition, click event handling
11-12	Testing	User testing protocol, accuracy measurement, performance optimization
13-14	Refinement	Bug fixes, UI improvements, accessibility features
15	Documentation	Technical manual, user guide, code documentation, demo video
16	Final Delivery	Project presentation, code submission, final report

Milestones: - Week 4: Working face and iris detection - Week 8: Functional calibration and cursor movement - Week 12: Complete system with click detection - Week 16: Tested, documented, production-ready system

---

9. Conclusion

This project builds upon the proven methodology of Patil et al. (2021) while introducing significant innovations in adaptability, accuracy, and user experience. By leveraging modern computer vision frameworks (MediaPipe) and focusing on accessibility, we aim to create a practical, affordable solution that can genuinely improve quality of life for users with motor disabilities.

The combination of rigorous research foundation, clear technical approach, and measurable objectives positions this project to make a meaningful contribution to assistive technology while demonstrating advanced computer vision and software engineering skills. The expected outcomes include not only a functional prototype but also validated improvements in accessibility and usability compared to existing free solutions.

References

Primary Source: 1. Patil, A., Patwardhan, M., & Shingane, D. (2021). Real-Time Gaze Tracking and Cursor Control Using MediaPipe and OpenCV. *International Journal of Advanced Research in Computer Science and Software Engineering*, 11(5), 89-95.

Supporting References: 2. Lugaresi, C., et al. (2019). MediaPipe: A Framework for Building Perception Pipelines. *arXiv preprint arXiv:1906.08172*. 3. Soukupová, T., & Čech, J. (2016). Real-Time Eye Blink Detection using Facial Landmarks. *21st Computer Vision Winter Workshop*. 4. World Health Organization. (2022). Disability and Health Fact Sheet. WHO Press. 5. Zhang, X., et al. (2020). It's Written All Over Your Face: Full-Face Appearance-Based Gaze Estimation. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

Team Information

Supervisor: Dr. Mohammed Hussien
Department: Computer Science, El Shrouk Academy
Academic Year: 2025/2026

This proposal represents our commitment to developing accessible technology that empowers individuals with disabilities while demonstrating technical excellence in computer vision and software engineering.