Ministry of education and science of Ukraine
National University of Kyiv-Mohyla academy

Development of an application to assist people with motor impairments in interacting with digital interfaces

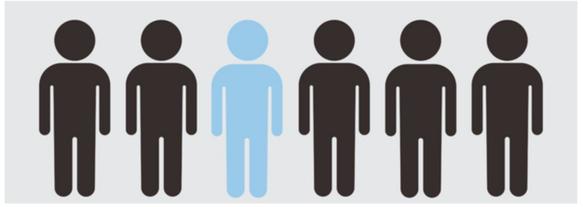
Volodymyr Materynskyi
Thesis supervisor: Dmytro Kuzmenko

Research objective

- Conduct review of existing interaction solutions.
- Select an appropriate technical approach
- Identify limitations and gaps in current implementations
- Design and develop a custom solution

Why inclusive human-computer interaction matters

- HCI is essential for full social and professional participation.
- Traditional input devices pose challenges for users with motor impairments.
- Demand for assistive technologies is rising due to conflicts, disasters, and global health crises.
- High cost and complexity make many current solutions inaccessible.



Scope of Experiments

Initial Idea:

- Use Python-based eye-tracking (OpenCV, GazeTracking) via API.

Challenges:

- High computational load
- Complex integration

Final Approach:

- Frontend wrappers around WebGazer.js & GazeCloudAPI

Application Wrapper Architecture

Chosen Library: WebGazer.js

Demo Setup:

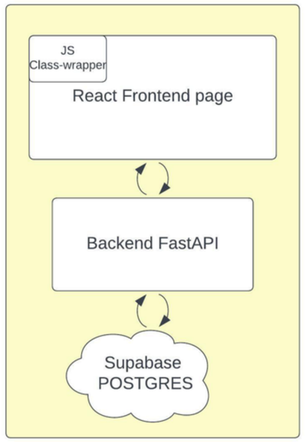
- React-based demo site
- Gaze-controlled interaction layer

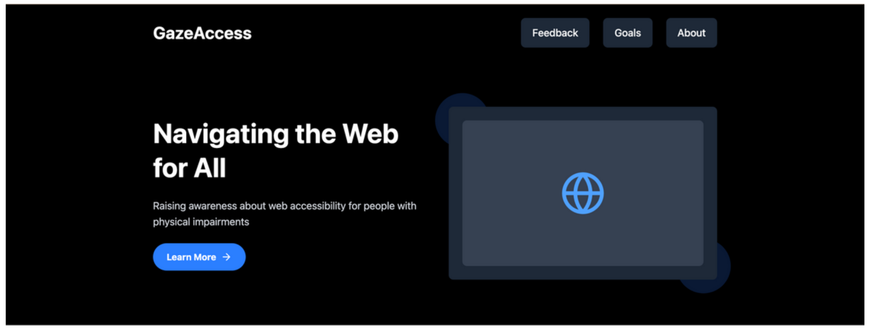
Initial Issue:

- Tightly coupled with demo site

Refactor Solution:

- Extracted logic into reusable JS class
- Achieved framework-agnostic design





Interaction Logic

Cursor Movement:

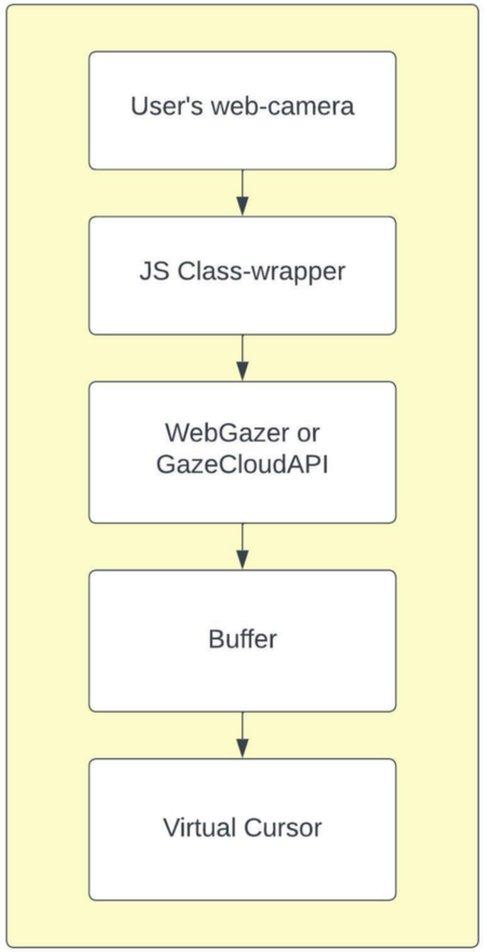
- Gaze prediction → virtual cursor update
- Cursor rendered in-browser (not system-level)

Click Simulation:

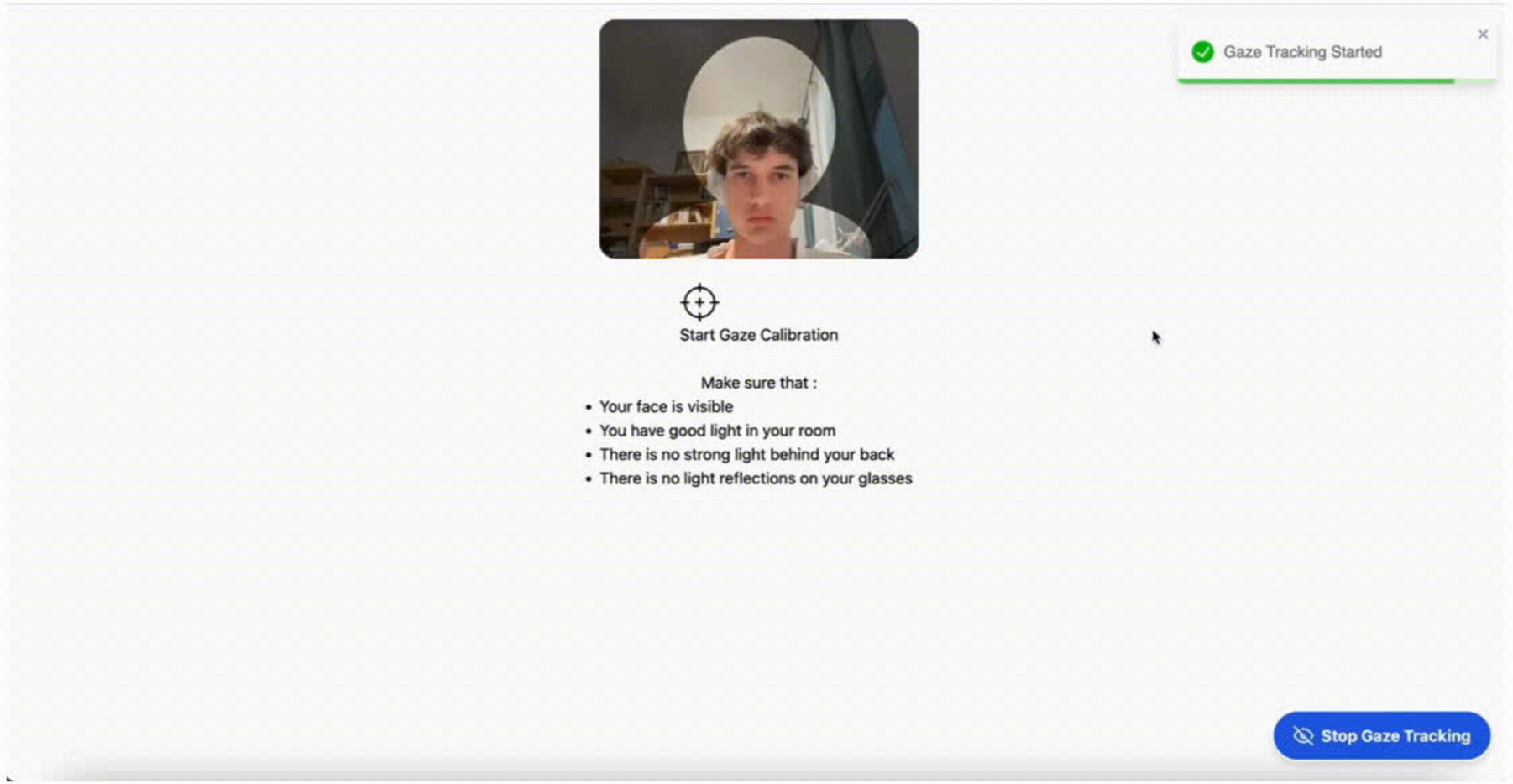
1. Cursor enters interactive area
2. Timer activates
3. If it stays: trigger synthetic click

Scroll Simulation:

- Entering top or down 10% of screen(configurable)



Demo


Gaze Tracking Started
Start Gaze Calibration
Make sure that :
• Your face is visible
• You have good light in your room
• There is no strong light behind your back
• There is no light reflections on your glasses
Stop Gaze Tracking

Challenges

1. Customization
    a. Limited control over tuning & accuracy
    b. Explore open-source models
2. Erratic Cursor Movement
    a. Cursor jitter due to inconsistent gaze predictions
    b. Buffer of last 5 points
3. Prediction Accuracy
    a. Improved but still not precise enough for small elements
    b. Directional vector-based control

Conclusion

Key achievements

- Lightweight, framework-agnostic JavaScript plugin
- Enables gaze-based web navigation
- Easy to configure, no external dependencies
- Compatible with any JS frontend
- Minimal resource usage

Current Limitations

- Accuracy depends on lighting & camera quality
- Inaccurate for small UI elements
- Limited to CPU; no GPU acceleration
- No support for text field interaction
- Based on outdated models with no training flexibility

Future Prospects

- Integrate modern or custom-trained gaze models
- Add virtual keyboard support for text input
- Refactor navigation logic for better responsiveness
- (“point & interact” over “point & focus”)
- Improve calibration and low-light performance

Thank you for your attention!