Detection of Eye Gaze Position with Computer Vision

Projet CSC51073 – 2025

Nathan Duboisset Roméo Nazaret

École Polytechnique

10 décembre 2025

Context and Applications

Why eye tracking ?

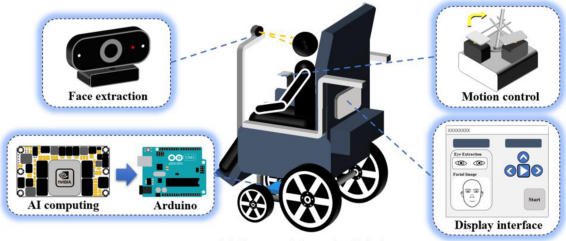


Figure – Pupil detection example

Context and Applications

Why eye tracking ?

- Anti-cheating systems in exams

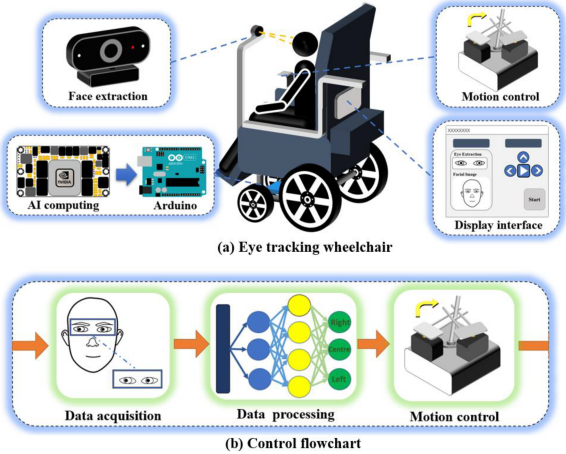


Figure – Pupil detection example

Context and Applications

Why eye tracking ?

- Anti-cheating systems in exams
- Accessibility for people with disabilities

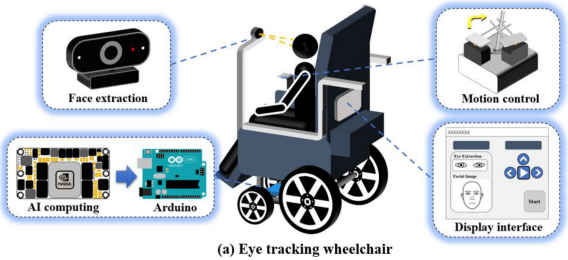


Figure – Pupil detection example

Context and Applications

Why eye tracking ?

- Anti-cheating systems in exams
- Accessibility for people with disabilities
- Human-computer interaction

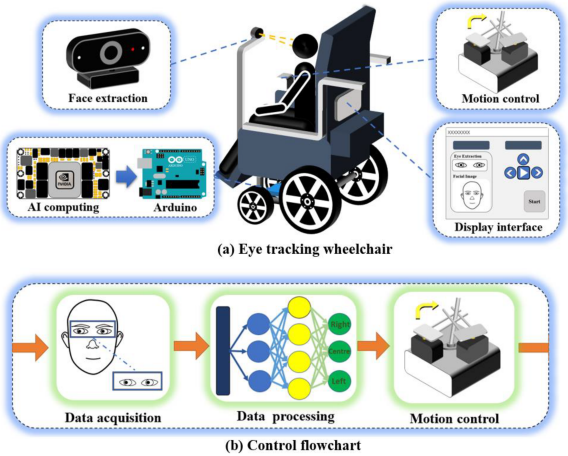


Figure – Pupil detection example

Context and Applications

Why eye tracking ?

- Anti-cheating systems in exams
- Accessibility for people with disabilities
- Human-computer interaction
- Real-time communication through eye movements

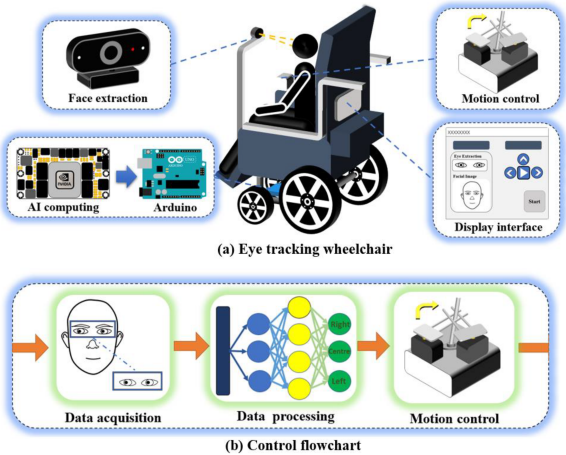


Figure – Pupil detection example

The Challenge

Technical Difficulties

- Eyes can be occluded (glasses, blinks, objects)

The Challenge

Technical Difficulties

- Eyes can be occluded (glasses, blinks, objects)
- Variable head positions and rotations

The Challenge

Technical Difficulties

- Eyes can be occluded (glasses, blinks, objects)
- Variable head positions and rotations
- Varying video quality and lighting conditions

The Challenge

Technical Difficulties

- Eyes can be occluded (glasses, blinks, objects)
- Variable head positions and rotations
- Varying video quality and lighting conditions
- Need for precision and low latency

The Challenge

Technical Difficulties

- Eyes can be occluded (glasses, blinks, objects)
- Variable head positions and rotations
- Varying video quality and lighting conditions
- Need for precision and low latency

Our Goal

Detect eye gaze position using a standard **webcam** with good precision and real-time performance

State of the Art

AFIG 2007 (Raynal)

Morphological approach

- C_m : morphological quality
- C_c : colorimetric quality
- No training data needed
- Computationally efficient

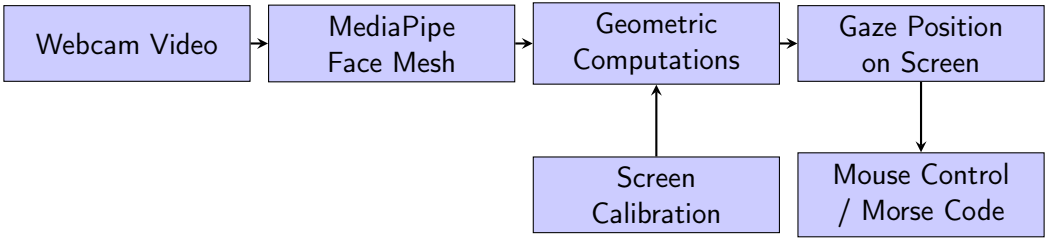
MediaPipe Face Mesh

Deep learning approach

- 468 facial landmarks
- CNN-based model
- Real-time on standard hardware
- 32 landmarks per eye

$\Rightarrow$ Our choice : MediaPipe + geometric computations

Our Pipeline Architecture



Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks

Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks
- Eye and iris positions

Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks
- Eye and iris positions
- Pupil positions relative to face

Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks
- Eye and iris positions
- Pupil positions relative to face
- Compute eyeball center positions

Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks
- Eye and iris positions
- Pupil positions relative to face
- Compute eyeball center positions
- $\Rightarrow$ Two vectors for gaze direction

Data Gathering with MediaPipe

What we extract :

- Face mesh with 468 landmarks
- Eye and iris positions
- Pupil positions relative to face
- Compute eyeball center positions
- $\Rightarrow$ Two vectors for gaze direction

Alternative Approaches We Tried

- CNN for direct pupil detection : less efficient
- Geometric disk fitting (circle detection) : unreliable
- **Final choice :** MediaPipe method (most reliable)

Screen Position Detection

Challenge : Find screen position and orientation from camera view

Method 1 : OpenCV

- Uses printed calibration pattern
- Accurate 3D positioning
- Handles camera distortion
- Not autonomous

Method 2 : Custom Calibration

- User looks at screen center
- Move forward/backward
- Compute gaze vector intersections
- Quick and autonomous

Distance estimation : *face size* $\propto$ *distance to screen*

Computing Gaze Position on Screen

Two approaches :

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average
2. Add the two vectors, then intersect with plane

Gaze Estimation & Interaction

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average
2. Add the two vectors, then intersect with plane

$\Rightarrow$ Method 1 chosen : more accurate and easier to debug

Gaze Estimation & Interaction

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average
2. Add the two vectors, then intersect with plane

⇒ Method 1 chosen : more accurate and easier to debug

Communication Methods

- **Mouse control :** Eye blinks = mouse clicks

Gaze Estimation & Interaction

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average
2. Add the two vectors, then intersect with plane

⇒ Method 1 chosen : more accurate and easier to debug

Communication Methods

- **Mouse control :** Eye blinks = mouse clicks
- **Morse code :** Single blink = dot, double blink = dash

Gaze Estimation & Interaction

Computing Gaze Position on Screen

Two approaches :

1. Intersect each eye vector with screen plane, then average
2. Add the two vectors, then intersect with plane

$\Rightarrow$ Method 1 chosen : more accurate and easier to debug

Communication Methods

- **Mouse control :** Eye blinks = mouse clicks
- **Morse code :** Single blink = dot, double blink = dash
- Challenge : distinguish intentional vs. natural blinking

Experimental Results

Test setup :

- User at 60–70 cm from laptop screen
- Multiple target positions (corners, center, intermediate)
- Measure error distance on physical screen

Quantitative Performance

- $\sim$5 cm average error
- **Real-time** performance
- High detection rate
- Instant recovery after blinks

Qualitative Analysis

Works well :

- Normal lighting
- Moderate head motion
- No occlusions

Failure cases :

- Strong head rotation
- Distance too close/far

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Limitations & Future Improvements

- Simplify calibration procedure (more automatic)

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Limitations & Future Improvements

- Simplify calibration procedure (more automatic)
- Improve robustness to head pose variations

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Limitations & Future Improvements

- Simplify calibration procedure (more automatic)
- Improve robustness to head pose variations
- Better handling of lighting changes

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Limitations & Future Improvements

- Simplify calibration procedure (more automatic)
- Improve robustness to head pose variations
- Better handling of lighting changes
- Combine geometric + learning-based approaches

Conclusion & Future Work

What We Achieved

- Complete gaze tracking pipeline with **standard webcam**
- Real-time performance with low latency
- Sufficient precision for large button interaction
- Two communication modes : mouse control and Morse code

Limitations & Future Improvements

- Simplify calibration procedure (more automatic)
- Improve robustness to head pose variations
- Better handling of lighting changes
- Combine geometric + learning-based approaches

Thank you for your attention !