DETECTION OF EYE LOOK POISITION WITH COMPUTER VISION

Projet CSC51073 – 2025

20 novembre 2025

Nathan Duboisset - Roméo Nazaret

TABLE DES MATIÈRES

1 Introduction **3**

1.1 Context and motivation . . . 3

1.2 Problem . . . 3

1.3 Project goals . . . 4

2 State of the art **5**

2.1 AFIG 2007 . . . 5

2.2 MediaPipe Face Mesh algorithms . . . 5

2.3 Other approaches . . . 5

3 Implemented method **6**

3.1 General architecture . . . 6

3.2 Gathering of positions data . . . 6

3.3 Detection of the position of the screen . . . 6

3.4 Estimation of the eye gaze . . . 7

3.5 Choice of language and libraries . . . 7

4 Results **8**

4.1 Experimental protocol . . . 8

4.2 Quantitative performances . . . 8

4.3 Qualitative analysis . . . 8

5 Conclusion **9**

1 INTRODUCTION

1.1 Context and motivation

Computer vision and image processing are now widely used in many applications. Eye tracking and pupil detection have become important tools with practical uses.

Pupil detection systems have several useful applications. They can be used as anti-cheating systems in exams by monitoring where students look. They also help people with severe disabilities communicate through eye movements when they cannot move other parts of their body. These applications show why we need accurate and real-time pupil detection methods.

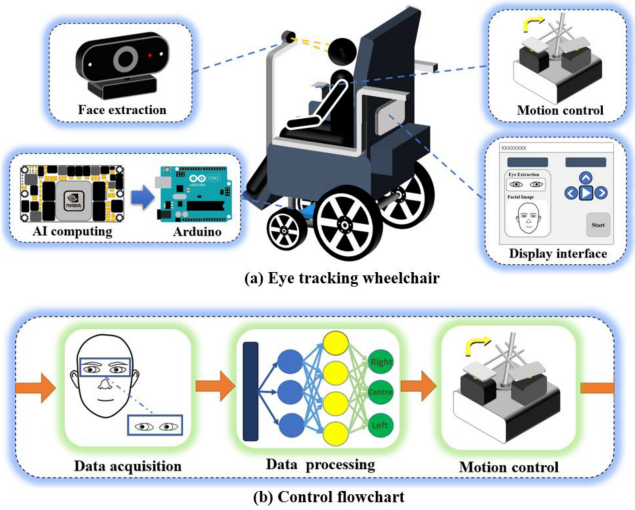

Figure 1 – Example of pupil detection application

1.2 Problem

Detecting the position of the eyes is a difficult task because the eyes can be occluded by other objects, the eyes can be closed, they can be looking in different directions, etc. Also there is a lot of technical problem : how to have precise description of the eyes position regardless of head position, quality of the video, number of images per second, etc.

We must be able to adapt to all these different situations while keeping a good precision and a low latency, thus using fast enough solutions.

1.3 Project goals

Our goal is to detect the position of the eyes in a video with a good precision and a low latency. We will use a computer vision algorithm to detect the eyes position. We will also try to use a machine learning algorithm to detect the eyes position. As we will use the webcam of our computer, we will have a difficult enough problem, as the quality of the video is not always good.

2

STATE OF THE ART

2.1 AFIG 2007

2.2 MediaPipe Face Mesh algorithms

2.3 Other approaches

3

IMPLEMENTED METHOD

3.1 General architecture

We chose to focus first on the detection of the eyes gaze position rather than detecting the position of the objects themselves as these tools are already widely used and have a lot of research, and good and fast implementations. Thus we used mediapipe and took the computing from there.

3.2 Gathering of positions data

We use mediapipe to gather the positions of the eyes and the face. We use the face mesh algorithm to get the positions of the eyes and the face. We use the iris landmarks to get the positions of the pupils. We also use mediapipe to gather the position of the pupils on the face.(TODO : pupil position detection) Using this, we compute where should be the position of the middle of the eyeballs, and thus obtain two vectors supposed to be the direction of the eye gaze.

3.3 Detection of the position of the screen

Once we had the vectors modelizing the position of the eyes, we can intersect them with the expected position of the screen, and knowing the dimensions of the screen, its resolution etc. we can compute the position of the gaze on the screen. But to have that computed, we need to know the distance between the head seen by the camera and the screen, even when the head is moving. We also need to know the orientation of the screen, because we can only access the images of the camera. In the beginning, we can make the assumption that the screen is right in front of the camera, "parallel to the face of the person".

To have this, we can use many methods. We tried to implement two of them(TODO) :

The first on is to use the openCV calibration function, which works well but takes a printed pattern to calibrate, thus the system is not really autonomous. Using this method, we can calibrate the system with a printed pattern, and then use the calibration to compute the position of the gaze on the screen. OpenCV once calibrated gives us the position of the items we "show" it, thus we can have a good 3d position approximation. One of the advantages of this method is also that it has a better

understanding of the distortion of the camera, i.e. the fact that the image is not a perfect projection of the 3d world, and that the projection of the plannar image of the camera is not a plannar image in the 3d world.

The second idea is to bypass the need to know exactly where the plane of the screen is by using our own calibration method. The idea is to use a standard calibration for this type of work which is asking the user to look at something we know on the screen, then move. Here, the idea is to ask the user to look at the center of the screen, and assume that the distance face / screen is proportional to a function of the size of the face.

To compute this, we ask the user to fix a point, then move forward and backward. we compute the vectors of the eye gaze, and consider they meet (or closely meet) at a point on the screen. This is not perfect as the gaze is not always perpendicular to the screen, but it can give a good approximation. By computing using smoothed data on this, we can calibrate this function on our own, and thus later use it to determine really quickly the position of the gaze on the screen.

3.4 Estimation of the eye gaze

TODO : intersect vectors with screen, take average TODO : take sum of vectors, intersect with screen TODO : machine learning from this

3.5 Choice of language and libraries

TODO : tried to use c++ ? python a bit slow

4 RESULTS

4.1 Experimental protocol

TODO : data used, metrics

4.2 Quantitative performances

TODO : precision, latency, detection rate

4.3 Qualitative analysis

TODO : success and failure cases, visualizations

5

CONCLUSION

TODO : summarize the project, the results and the perspectives

REFERENCES

RÉFÉRENCES

[1] B. Raynal. *Reconnaissance de la pupille par morphologie mathématique.* Actes de l'AFIG, 2007.