Mouse management using OpenCV

Presentation and project made by Marcos Hidalgo Baños

Motivation of the project.

- **Why is it interesting for development?**

  By implementing an alternative that optimice daily work and modernizes the use of the conventional mouse our productivity will be increased when working on the computer.

  Also, it could provide you a cool look!

- **Then, why is it not used in our devices?**

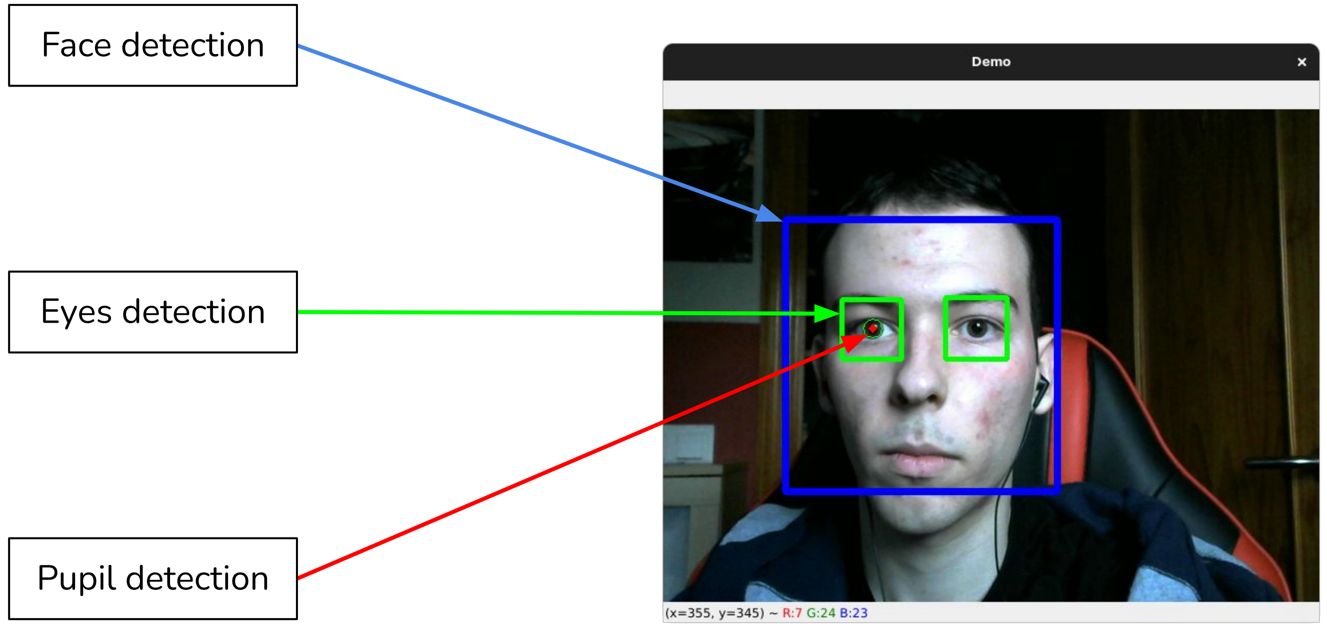
  It's so inaccurate that is more of a problem than an advantage. Nevertheless, I personally wanted to try my best and face all the difficulties the best way as possible.

Solution proposed.

01	Capture each frame and turn it into grayscale	• Since our program gets video inputs, each frame needs to be treated individually and sequentially. • The detectors only work properly on gray images.
02	Draw the rectangle area around the detected face	• Import pre-trained model for face detection. • **Cascades** return the set of detected faces. • Using cv2 we may draw the rectangle area.
03	Detect the eyes area and draw its corresponding rectangle	• Filter model output in search of false positives. • We will select only one eye for tracking.
04	Within the eyes, detect the iris and locate its center	• Detecting circles within the eyes area eliminates additional figures not being the iris outside the face. • Draw the pupil big circle and its coordinates center. • This can be possible with cv2.**HoughCircles()**.
05	Calculate the difference of centers between the previous position and the actual one	• Knowing the center and previous one we can calculate how must move the mouse. • Once the mouse has already move, check for click events through **pynput.mouse** library.

Going from larger to sorter frame size

Obtained results.



Conclusion and project improvements.

Although the project seems solid, it faces some difficulties that make it not entirely useful:

➔ The hardest aspect is definitely **pupil tracking**. Since our eyes are constantly moving and its displacement are very small the HoughCircles function is not precise enough.

- <u>Possible solution:</u> Create an array of pupils centers (length 4 or 5) and average. If this does not solve it, we may consider changing the method.

➔ We may also consider **mouse interactions**. The used library does not allow clicks to happen once the window is minimized, so it is not practical. In addition to that, due to the imprecisions described the mouse displacement is quite abrupt.

- <u>Possible solution:</u> Try another handler for mouse events (interactive response)