The gaze detection part of the software is done in Python, using the OpenCV library for machine vision. Generally, the process of facial recognition is hierarchical, meaning we find the bigger components first, then narrow it down to specifics.

Face components

* This part was fairly simple to implement, we use a library called dlib, which allows us to map our face to certain facial landmarks (as seen by the dots in the image)
* From this we can crop out the eyes, or other facial components and analyze them in detail.

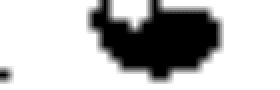


Eyes and iris direction

* Using the dlib landmarks from before, we can apply a mask around the eye region and isolate the eye to obtain an image with only the pupil, and possibly some edge objects.
* With a threshold function, we can obtain a black and white picture of the pupil like below. This thresholding value will be one of the key variables to adjust to optimize performance. The challenge here is that if it’s too low, it may not work for blue, or any non-dark eye colour, but if it’s too high, it will capture noise.

Iris analysis

* Now that we have an isolated image of the pupil, we can apply techniques to determine the gaze direction. There are many ways to do this, including blob detection, or gradient directions, etc.
* For our purpose, since we only need to know the general direction of the gaze, we can simply split the image horizontally and vertically and count the pixels in each side. Subtracting one side with the other (right – left or top – bottom) will give us the value shown in the screen above.
* One case the algorithm struggles with is looking downwards. Since an eye is more wide than tall, there is little difference between looking up or down (especially down). To solve this, we use the fact that people tend to close their eyes a bit when looking downwards. We can factor in the ratio of height to width when doing downward detection.
* When blinking, this screen goes almost white, so we can utilize a ratio of pixels and total image size.



Gaze direction

* Now that we have a value for gaze direction, we just need to set a threshold that indicates whether the person is looking in whichever direction. This threshold can be set in real time with calibration.
* In addition, we apply an average over time on the gaze values to smooth out any abrupt changes due to unexpected circumstances.