**Drowsiness Detection with OpenCV**

We are going to extend this method and use it to determine how long a given person’s eyes have been closed for. If there eyes have been closed for a certain amount of time, we’ll assume that they are starting to doze off and play an alarm to wake them up and grab their attention.

To accomplish this task, I’ve broken down today’s tutorial into three parts.

In the first part, I’ll show you how I setup my camera in my car so I could easily detect my face and apply facial landmark localization to monitor my eyes.

I’ll then demonstrate how we can implement our own drowsiness detector using OpenCV, dlib, and Python.

Finally, I’ll hop in my car and go for a drive (and pretend to be falling asleep as I do).

As we’ll see, the drowsiness detector works well and reliably alerts me each time I start to “snooze”.

The camera I used for this project is a USB camera. Camera as it:

* Is relatively affordable.
* Can shoot in full 1080p.
* Is plug-and-play compatible with nearly *every* device I’ve tried it with the Raspberry Pi.

**Drowsiness Algorithm**

The general flow of our drowsiness detection algorithm is fairly straightforward.

* First, we’ll setup a camera that monitors a stream for faces:
* If a face is found, we apply facial landmark detection and extract the eye regions:
* Now that we have the eye regions, we can compute the eye aspect ratio (detailed [here](https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/)) to determine if the eyes are closed:
* If the eye aspect ratio indicates that the eyes have been closed for a sufficiently long enough amount of time, we’ll sound an alarm to wake up the driver:
* Implement the drowsiness detection algorithm detailed above using OpenCV, dlib, and Python.