Driver Monitoring System using AI

Technologies for Autonomous Vehicles - First Assignment

Francesco Renis s314803

April 23, 2024

Introduction

This assignment aims to develop a basic Driver Monitoring System capable of detecting driver drowsiness or distraction by analyzing their head and eye gaze directions using an AI model. Based on this information, the program prints an alarm message when critical thresholds are exceeded, indicating that the driver is drowsy or distracted.

1 Google MediaPipe FaceMesh

This system relies on MediaPipe FaceMesh, an AI Model developed by Google that provides 3D coordinates of 468 facial landmarks positions given an RGB camera image. In addition to that, the model can provide iris tracking thanks to 10 additional 2D points, that also enable the estimation of the subject distance by means of geometrical computations.
Each of the landmarks thus generated is therefore identified by a position index within the resulting structure (basically an array with 468 3D and 10 2D points). Additional details on model features and how it was trained can be found at [1, 2, 3]

2 Development details

This Monitoring System is implemented via a Python script, whose base code structure was provided by Researcher Jacopo Sini from Politecnico di Torino. Necessary libraries are OpenCV, NumPy and, as said, MediaPipe.
After creating an instance of the MediaPipe FaceMesh model with specified parameters, the script basically drives an infinite loop. At each iteration a new frame is taken from the camera source and processed by the model, which returns the produced face mesh. All the useful points are filtered by their index, rescaled to the size of the input image (since the mesh is normalized in [0,1] interval) and saved to variables. In the meanwhile, all the camera images are shown to the user.

2.1 Environment

All the additions to the base version of the script have been written using Visual Studio Code IDE by Microsoft on a Ubuntu 22.04 environment. The camera used is a medium-low quality external 1080p webcam by TECKNET.

3 Drowsiness detection

Two fundamental metrics have been introduced in order to detect if driver is drowzy:

- **Eye Aspect Ratio (EAR)** which is a measure of the eye aperture, defined as:

$$EAR = \frac{|Y_2 - Y_6| + |Y_3 - Y_5|}{2|X_1 - X_4|}$$

  where the points 1-6 are as shown in Figure 1 and in this code correspond, respectively, to the points at indices $33, 160, 158, 133, 153, 144$ for the right eye and $362, 385, 387, 263, 373, 380$ for the left eye. In terms of code implementation, the average of the EARs of the two eyes is used in all the calculations.

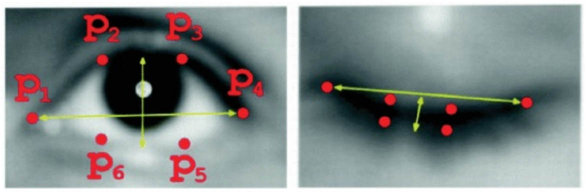


Figure 1: Eye Aspect Ratio (EAR)

- **Percentage of eye Closure (PERCLOS)** which is related to EAR computation and, as the name suggests, measures the percentage of eye closure with respect to a reference maximum EAR value. This number has been statically set as $EAR_{open} = 0.28$, which has generally shown to be a suitable value. The PERCLOS is in general calculated with a formula involving different contributes of time intervals relative to different EAR percentages. In this specific context, however, a simple but good approximation of this measure has been defined as:

$$PERCLOS = t_1 + t_2$$

  where t_1 is the interval of time when $EAR > 80\% EAR_{open}$ and t_2 is the interval of time when $EAR < 70\% EAR_{open}$

In order to detect driver drowsiness, thresholds to those measures have been set. In particular, biomedical analyses highlight that a person driving in somnolence state tends to reduce his or her blinking frequency to try to stay awake, resulting in a longer time spent with eyes wide open. Therefore the Monitoring System developed prints a warning message when the PERCLOS exceeds a threshold of $10s$. Practically this means that a warning is triggered when the driver is detected to be trying to stay awake or to be slowly closing its eyes.
In terms of coding details, after the EARs and the corresponding average EAR are computed, the functionality is implemented simply by means of a time variable that is incremented if the average EAR goes over or below the specified PERCLOS thresholds. Then, at the end of each iteration of the infinite while loop, there is a check on the PERCLOS $10s$ threshold and eventually the warning message is printed. As a potentially useful information, the computed avg EAR is printed to the user at each video frame.

4 Distraction detection

Driver distraction detection relies on the calculation of head and eyes gaze angles. If the combination of those angles results as not focused for more than 3 seconds, a warning is triggered and a message is printed to the user interface. More details on the detection of head and eye gaze focus follows.

4.1 Head gaze direction

Head gaze direction recognition is based on geometrical perspective computations relying mainly on the 3D mesh points returned by MediaPipe FaceMesh model. The basic idea of this implementation is to use a Least Squares approach to find the pose of a plane that includes a set of significant points of the face, and comparing it with the plane in the 2D reference system of the camera. The computed plane at each instant will be rotated with respect to its projection of given yaw and pitch angles. At last, the roll angle is simply

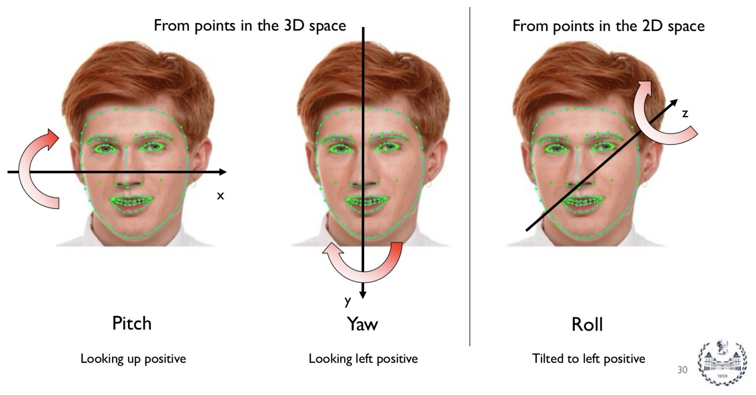


Figure 2: Head roll, pitch, yaw angles

the rotation angle between the facial plane and its projection, with the first resorted on the 2d plane of the camera. Specifically, the facial significant points are the ones of the mesh at indices $33, 263, 1, 61, 291, 199$. The rotational vector in 3D space with respect to the 2D perspective is then obtained by means of $SolvePnP()$ function provided by OpenCV library, using the defined camera matrix parameters. An important note on this is that we do not require the focal length to be determined exactly, since no actual transformation to the real reference system is exploited and all the computations are relative to one another. Moreover we can assume that the image is not subject to significant distortion, since the camera used for this project has no particularly excessive focal length. After that, the corresponding rotational matrix is then obtained thanks to $Rodrigues()$ function from OpenCV, that achieves the transformation using Rodrigues' rotation formula. Using the $RQDecomp3x3()$ function the rotational matrix is then converted into angles, that are saved, converted in degrees and printed to the user interface. If any of three angles is higher than 30°, the head gaze is considered as not focused.
In order to better visualize what the System is doing, a line from the nose is drawn on the frame projected towards the computed head gaze direction.

4.2 Eyes gaze direction

The estimation of eyes gaze direction relies mainly on the 2D points provided by the Iris Tracking functionality of the MediaPipe model. The application of the same algorithm described previously for the head gaze direction would result as highly imprecise and not reliable, mainly due to the eye size being very small with respect to the whole image, therefore a different approach is taken. The strategy simply consists in defining a rectangle centered in the eye orbit, with its perimeter describing a threshold that, if surpassed, make the system consider the eye gaze as not focused. The basic intuition comes from the fact that the surface of the iris of a human eye describes a 45° angle with respect to the total area of the eye. The projection of the iris surface becomes a rectangle for simplicity, and it is reshaped to tighten the angle.
Practically speaking, the eye center is intended as the eye orbit center, and it is calculated as intersection of the line connecting the point at the top of the eye orbit below the eyebrow (point 27 for the right eye and 257 for the left eye) and the point at the bottom of the eye (point 23 for the right eye and 253 for the left eye) with the line connecting the left and right palpebral angles (respectively points 33 and 133 for the right eye, and 362 and 263 for the left eye). The 8 2D points describing the left and right iris are used to compute the axes of the iris circle (an ellipse in projection), which are then rescaled to tighten the angles considered. For each image frame captured by the camera, if the pupils' centers (the 2 remaining 2D points provided) fall outside of the defined thresholds, the eye gaze would result as not focused. Due to the strategy adopted, it is not possible to predict specific eye pitch and yaw angles (no roll) but only their sign just to distinguish the direction, even if we are not specifically interested in that. Again, everything is shown to the user interface.

It is important to note that when gazing downwards the upper eyelid naturally moves alongside with the eye, leading to confusion in the AI prediction model. To compensate for this, there is an additional threshold

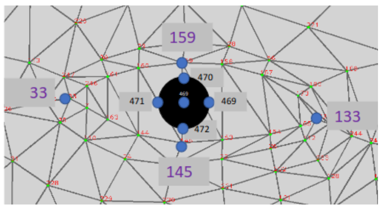


Figure 3: 3D and 2D right eye mesh points

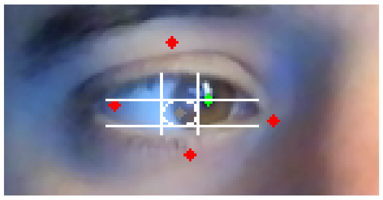
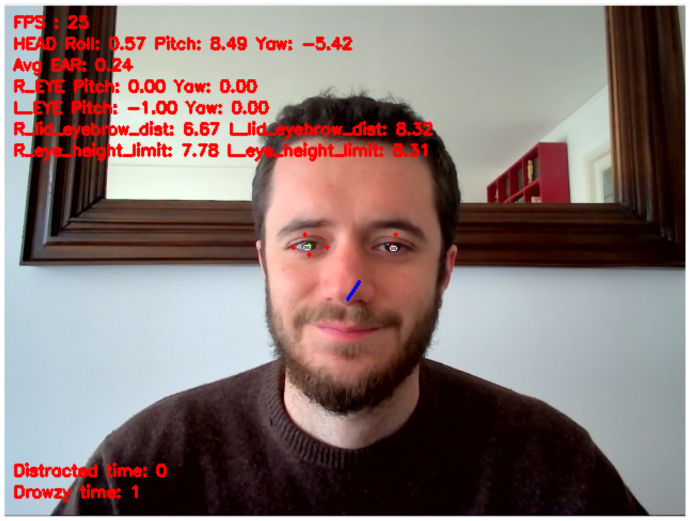


Figure 4: Right eye gaze not focused

set on the distance computed between the upper eyebrow and the upper eyelid: if that goes over a certain value (set to 40% of the eye height), the distraction time counter stars, and eventually the warning gets triggered.
Below a picture of the full user interface with the program running.



```
FPS : 25
HEAD Roll: 0.57 Pitch: 8.49 Yaw: -5.42
Avg EAR: 0.24
R_EYE Pitch: 0.00 Yaw: 0.00
L_EYE Pitch: -1.00 Yaw: 0.00
R_lid_eyebrow_dist: 6.67 L_lid_eyebrow_dist: 8.32
R_eye_height_limit: 7.78 L_eye_height_limit: 8.31

Distracted time: 0
Drowzy time: 1
```

Figure 5: User Interface

References

[1] *Google MediaPipe FaceMesh*. Face landmark detection guide. URL: `https://developers.google.com/mediapipe/solutions/vision/face_landmarker`.

[2] Ivan Grishchenko et al. *Attention Mesh: High-fidelity Face Mesh Prediction in Real-time*. June 19, 2020. arXiv: `2006.10962[cs]`. URL: `http://arxiv.org/abs/2006.10962` (visited on 04/19/2024).

[3] Yury Kartynnik et al. *Real-time Facial Surface Geometry from Monocular Video on Mobile GPUs*. July 15, 2019. arXiv: `1907.06724[cs]`. URL: `http://arxiv.org/abs/1907.06724` (visited on 04/19/2024).