TASK 1: “FATIGUE AND SOMNOLENCE DETECTION”

V-Model

Requirement analysis or user requirements:

The main user requirement

UReq0: Designing a somnolence detection system for drivers

System design:

Req0: Collecting data from a rotative camera (In case the driver moves his head while driving), using image processing programs to detect the driver’s face and ayes, and decide if he is sleepy and tired.

Req1: After deciding if driver is sleepy, alert him with a sound signal, not sudden or loud so that he doesn’t lose control over the vehicle.

Req2: Develop the solution on an embedded platform so it could be easily integrated on the vehicle main system without disturbing the driver’s comfort.

Req3: Communication should be done using a CAN-Bus that transfers data to the dashboard. it is the most used communication protocol for the automotive industry and ensures an important number of nodes (ECU).

Architecture design:

AReq0: For this function we will need a camera, rotary support for the hardware. For the software, python and OpenCV library will be the tools used. For the first application, the image processing will be done on a still image, then on a live camera feed.

AReq1: For the sound signal, a small buzzer will be used to generate a light signal in case the driver is sleeping.

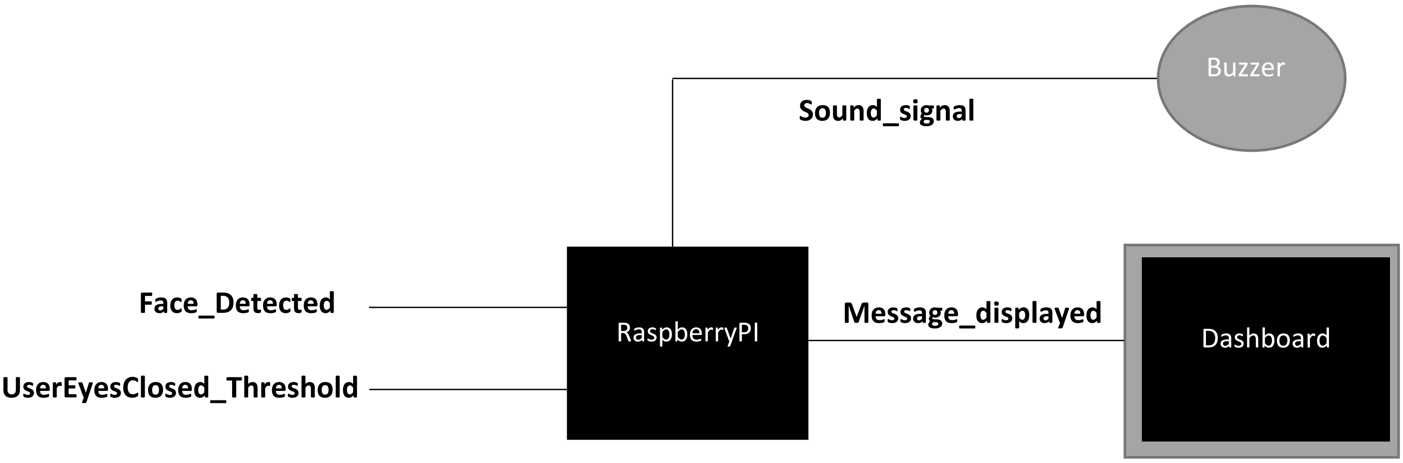
AReq2: To make this solution embedded, the system will be transferred on a Raspberry card. The data will be processed throughout this card and should be fully integrated in the vehicle.

AReq3: For this requirement, we will use a Can bus module for the communication between the different ECUs. For the dashboard, we use an interactive screen for information display.

Module Design:

The program will have inputs and outputs and should generate a signal in case it detects any sign of fatigue, using face expressions or eyes detection. We should set a time threshold, that will be used by the program to determine if the user is sleeping or just closing his eyes, for example a two second threshold. Therefore, the program should be implemented this way:

```
Threshold = 2
IF Face_Detected = TRUE
   AND UserEyesClosed_Threshold = TRUE
THEN Sound_signal = ON Message_diplayed = TRUE
ELSE Sound_signal = OFF Message_displayed = FALSE
```



Unit testing:

Ut0: Face is detected

Ut1: Eyes are detected

Ut2: Threshold is used

Ut3: Sound signal test (Intensity and frequency)

Ut4: Camera → Raspberry test

Ut5: Raspberry → CAN Module test

Ut6: Raspberry → Dashboard test

Integration testing:

It0: After linking the Req2, Req1 and Req0. We test if the sound signal is triggered in case the user is sleepy.

It1: After connecting the Req0 and Req2. We test if the driver's face and eyes are detected from a live camera feed.

It2: For the Req2 and Req3, we test if there are no errors in the transfer process in the Can-bus of the data collected, and if there are no errors displayed on the dashboard.

System testing:

	INPUTS	OUTPUTS
S0	There is no Face	No sound signal No indicator on the dashboard
S1	Face and eyes detected Driver is AWAKE	No sound signal No indicator on the dashboard
S2	Face and eyes detected Driver is SLEEPING	Sound signal Indicator/Message on the dashboard