MASTER OF TECHNOLOGY

PATTERN RECOGNITION SYSTEMS (PRS)

Driver Alertness Detection System

Group 7

Name	Student ID
Hwang Sion	A0249263Y
Prerak Agarwal	A0116711R
Santi	A0249294R
Zhang Junfeng	A0249266U

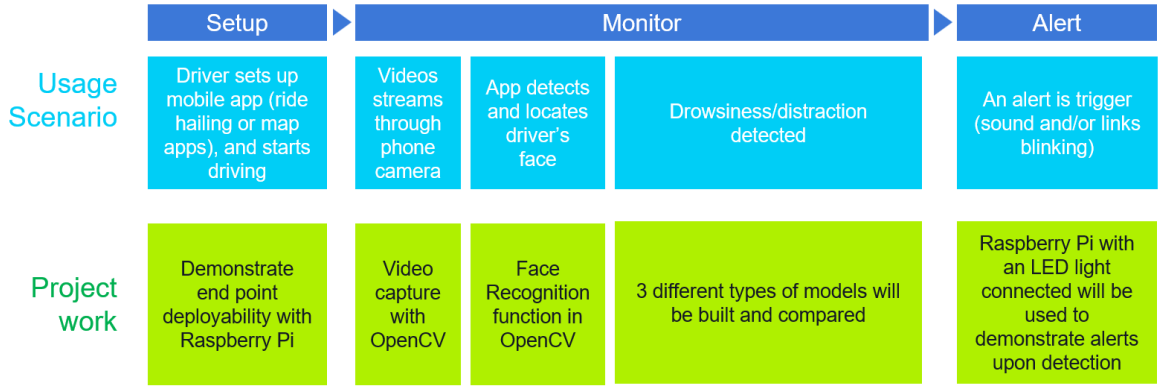
Contents

1 Objective

This document aims to provide an overview of “Driver Alertness Detection System”, highlighting system requirements, and steps for the installation. It is assumed that the user understands python programming.

2 System Overview

The system is design to fulfil the usage scenario which spans across setup, monitor and alert stages.



Technology stack differentiated into 2 categories, stack that being used for model training and stack that being used for deployment

	Training	Deployment
Software	**Python** • Tensorflow • Keras • Numpy • Sklearn • Matplotlibs • Pandas • Seaborn • PathLib • pygame • Face_Recognition • imutils • OpenCV • Tensorflow-metal to support GPU in macOS	**Python** • Tflite-runtime • Numpy • Facial-recognition • OpenCV • Tkinter • dlib
Hardware	**Google Colab Pro+ Subscription** • **RAM**: 90 GB • **GPU**: NVIDIA-SMI 460.32.03; Driver Version: 460.32.03; CUDA Version: 11.2; GPU Memory size: 40 GB **Apple Mac Studio- M1 Max** • **RAM**: 64 GB • **GPU**: 32 cores GPU with Metal GPUFamily Apple 7 **Google Colab Standard Subscription** • **RAM**: 13.6 GB • **GPU**: NVIDIA-SMI 460.32.03; Driver Version: 460.32.03;	**Raspberry Pi 4 Model B** • **Storage** (SD card): 32 GB • **RAM**: 8 GB **Extension to Raspberry Pi:** • GPIO Cable • Breadboard • Breadboard Adopter • LED Light with Resistor • USB Camera

	CUDA Version: 11.2; GPU Memory size: 14.75 GB **MSI Laptop** • **RAM**: 32 GB • **GPU**: NVIDIA GeForce RTX 3070 • **Processor**: 11th Gen Intel(R) Core(TM) i9-11900H @ 2.50GHz	

3 Installation

3.1 System Requirements

Description	Technical Specification
Hardware	**Personal PC** - CPU: 1.6 GHz or faster, 2-core Intel Core i3 or equivalent - GPU: *Optional - RAM: > 4 GB RAM - Hard disk: > 1 GB disk size **Raspberry Pi** - GPU: preferred to be set higher than default 128 - RAM: 8 GB - Hard disk: 32 GB - OS: Debian Bullseye
Software	OS: Windows 10 or Ubuntu 20.04 Software: Python 3 & above
Other Packages	All the required packages are included in requirement.txt. You can run 'pip install -r requirement.txt' to install all the required packages

3.2 Installation Steps

Step 1:

Install python in your computer.

- https://www.python.org/downloads/

Step 2:

Use 'git clone' command to download the project from the following URL:

- Project URL link: https://github.com/SionsML/PRS-PM-2022-07-02-ISY5002-GROUP_7.git

Step 3:

In the command line, change directory to the directory you have downloaded. Change directory to Group7 folder. In the same directory, run the following command:

- pip install -r requirement.txt

Step 4:

Change directory to ‘ui’ folder. Run the program with following command:

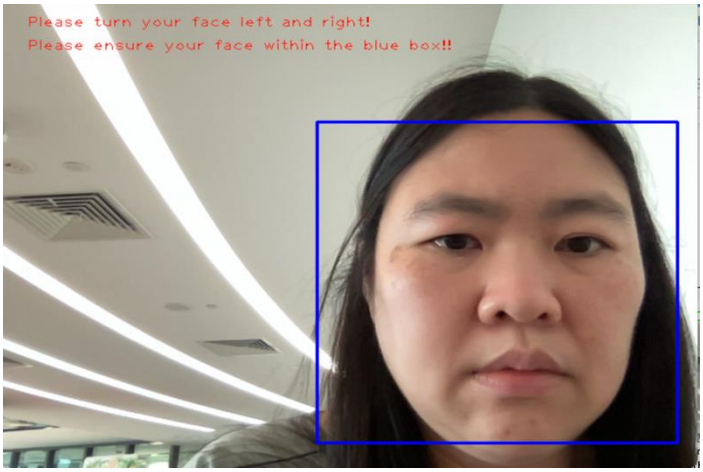
- python main.py

4 User Guide

Once the program runs, the UI screen below will pop up.

Register User

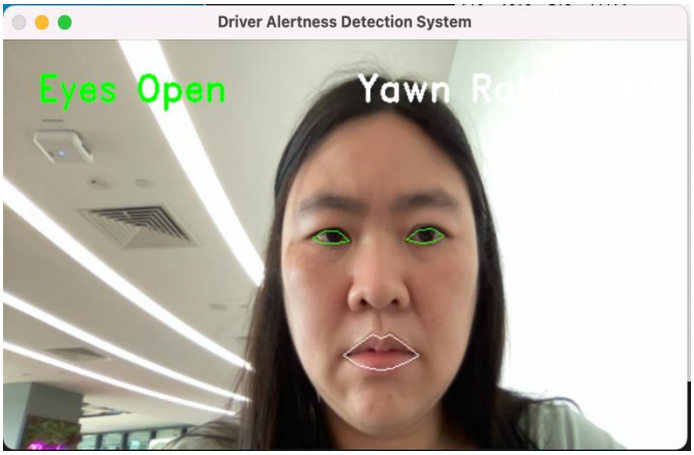
Users can click on “Register Me!” button to register the driver’s face. Only registered faces will be recognized by the program. This is to avoid recognizing passengers who are on board and only trace the faces of the driver.



- Register Me! – Upon selected Register Me! Camera will opne and system will recognize a face based on object detection, capture the images and encode this to pickle file to be used for other menus.

1st option “Front Camera Sleepiness Monitoring” is to monitor frontal face of registered driver to identify whether driver ‘s eye is closed/open, and driver is yawning or not based on mouth open ratio.

This model will throw alert if eyes are closed continuously for more than specified time or mouth is open above threshold for prolonged period.



Front Camera Sleepiness Monitoring – Upon selected this options will monitor whether the driver is registered user and will use the recognize face to detect eyes and mouth to determine whether driver is sleepy.

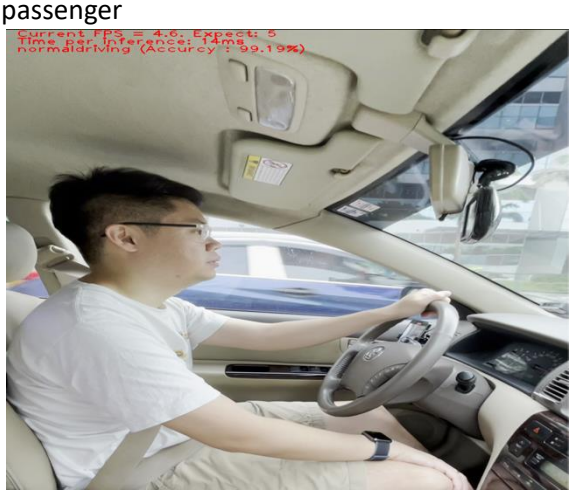
-

2nd option "Side Camera Alertness Monitoring" will detect full face and body posture to identify 2 things:

1. The labels for each video in this dataset are provided per video frame level and divided into two classes – 0 for 'Stillness / Normal Driving' and 1 for 'Drowsy'.
2. Driver's focus towards driving and the outside environment. There are many instances driver is losing focus due to some activities.

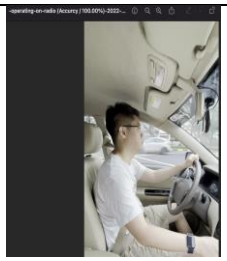
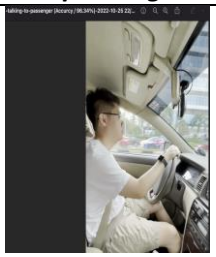
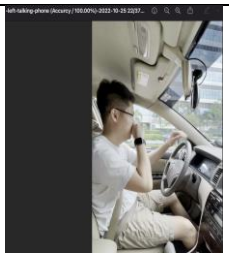
   There are 10 different classes which will be displayed in the screen:

   1. Normal Driving
   2. Texting with right hand
   3. Talking to phone with right hand
   4. Texting with left hand
   5. Talking to phone with left hand
   6. Turn on / actively working on radio
   7. Drinking
   8. Reaching Behind
   9. Playing with hair / make up
   10. Talking to passenger



- Side Camera Alertness Monitoring - Upon clicking camera will open and system will get frame images and pass to the model to get classification. If more x classification does not consider as normal driving system will inform driver to focus.

Example shown below with real time data:

Normal Driving	***Operating on Radio***	***Talking to passenger***	***Others***
 99.91% confident	 100% confident	 96.34% confident	 100% confident Use left hand for phone

3rd option "Front Camera Alertness Monitoring" will detect multiple frames sequentially to give contextual information of the frames taken:

The outcome of this is whether driver is drowsy or alert. In the sequence of frames, if the model classified as drowsy, the system would send a signal to alert.

Front Camera Alertness Monitoring - Upon clicking camera will open and system will get x frame of images and pass to the model to get classification.