Driver drowsiness detection system

A Project Stage II Report Submitted By

M.SNEHA-22011102055

A.KAPIL-22011102040

KISHORE C-22011102042

HARSHITHA JAMPANI-22011102032

In partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER ENGINEERING

Under the guidance of

Prof. Dr. Rohini B. Jadhav

DEPARTMENT OF COMPUTER ENGINEERING BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY), COLLEGE OF ENGINEERING, PUNE- 411043

2020-21

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY), COLLEGE OF ENGINEERING, PUNE-43

CERTIFICATE

This is to certify that the Project Stage1 report titled **DRIVER DROWSINESS DETECTION SYSTEM**, has been carried out by the following students in partial fulfilment of the degree of **BACHELOR OF TECHNOLOGY** in Computer Engineering of Bharati Vidyapeeth (Deemed to be University) Pune, during the academic year 2019-2020.

Team:

1. Radhika Agarwal
2. Niharika Singh
3. Nidhi Jain

Prof. Dr. Rohini B. Jadhav (Project Guide)	External Examiner	Prof. Dr. S.B. Vanjale HOD of Computer Engineering

Place: Pune
Date: 04/08/2021

ACKNOWLEDGEMENTS

Before we commence forward with our Project, we would collectively like to express our gratitude to the College and to Computer Department of Bharati Vidyapeeth University, College of Engineering for providing the students with their constant support and guiding us forever on the path towards knowledge.

We would also like to take this special opportunity to thank, first and foremost, Dr. S.B. Vanjale, our head of department for being a constant source of encouragement and mentoring us in authoring our Project.

We would also like to thank Prof. Rohini Jadhav, project guide for helping us in every step along the way. Your insightful remarks really challenged us to think and helped us bring out the best in us by constantly reviewing our reports and providing invaluable feedback.

We would also like to express our gratitude to the Principal of our college, Prof. Dr. Anand Bhalerao for giving us this wonderful opportunity of creating this dynamic Project by giving us access to the resources and enlightening us with his kind words.

Lastly, we would like to thank our friends and family for constantly supporting us in our endeavours and being with us, gracing us with their blessings and affection. The completion of this project would never have been possible without the help of any of the above-mentioned people.

ABSTRACT

- Driver fatigue is one of the major causes of accidents in the world.
- Detecting the drowsiness of the driver is one of the surest ways of measuring driver fatigue. In this project we aim to develop a prototype drowsiness detection system.
- This system works by monitoring the eyes of the driver and sounding an alarm when he/she is drowsy.
- The system so designed is a non-intrusive real-time monitoring system.
- The priority is on improving the safety of the driver without being obtrusive. In this project the eye blink of the driver is detected. If the drivers eyes remain closed for more than a certain period of time, the driver is said to be drowsy and an alarm is sounded.
- The programming for this is done in OpenCV using the Haarcascade library for the detection of facial features.

CONTENTS

LIST OF FIGURES

Chapter 1: Introduction

Driver fatigue is a significant factor in a large number of vehicle accidents.

The development of technologies for detecting or preventing drowsiness at the wheel is a major challenge in the field of accident-avoidance systems. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects.

The aim of this project is to develop a prototype drowsiness detection system. The focus will be placed on designing a system that will accurately monitor the open or closed state of the driver's eyes in real-time.

By monitoring the eyes, it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident.

Detection of fatigue involves the observation of eye movements and blink patterns in a sequence of images of a face.

OpenCV came in. OpenCV is an open source computer vision library. It is designed for computational efficiency and with a strong focus on real time applications. It helps to build sophisticated vision applications quickly and easily. OpenCV satisfied the low processing power and high speed requirements of our application.

We have used the Haartraining applications in OpenCV to detect the face and eyes. This creates a classifier given a set of positive and negative samples.

The steps were as follows:-

- Gather a data set of face and eye. These should be stored in one or more directories indexed by a text file. A lot of high quality data is required for the classifier to work well.

- The utility application createsamples() is used to build a vector output file. Using this file we can repeat the training procedure. It extracts the positive samples from images before normalizing and resizing to specified width and height.

- The Viola Jones cascade decides whether or not the object in an image is similar to the training set. Any image that doesn't contain the object of interest can be turned into negative sample. So in order to learn any object it is required to take a sample of negative background image. All these negative images are put in one file and then it's indexed.

- Training of the image is done using boosting. In training we learn the group of classifiers one at a time. Each classifier in the group is a weak classifier. These weak classifiers are typically composed of a single variable decision tree called stumps.

In training the decision stump learns its classification decisions from its data and also learns a weight for its vote from its accuracy on the data. Between training each classifier one by one, the data points are reweighted so that more attention is paid to the data points where errors were made. This process continues until the total error over the dataset arising from the combined weighted vote of the decision trees falls below a certain threshold

The face.xml and haarcascade-eye.xml files are created. These xml files are directly used for object detection. It detects a sequence of objects (in our case face and eyes). Haarcascade-eye.xml is designed only for open eyes. So when eyes are closed the system doesn't detect anything. This is a blink. When a blink lasts for more than 15 frames, the driver is judged to be drowsy and an alarm is sounded.

Chapter 2: Literature Survey

2.1 Literature Review

There are many previous researches regarding driver drowsiness detection system that can be used as a reference to develop a real-time system on detecting drowsiness for drivers. There is also several method which use different approaches to detect the drowsiness signs.

Drowsiness and Fatigue

Antoine Picot et al, stated that drowsiness is where a person is in the middle of awake and sleepy state. This situation leads the driver to not giving full attention to their driving. Therefore, the vehicle can no longer be controlled due to the driver being in a semi-conscious state. According to Gianluca Borghini et al, mental fatigue is a factor of drowsiness and it caused the person who experiences to not be able to perform because it decreases the efficiency of the brain to respond towards sudden events.

2.2 Various Model

Electroencephalography (EEG) for Drowsiness Detection

• Electroencephalography (EEG) is a method that measures the brain electrical activity. As shown in Figure 3, it can be used to measure the heartbeat, eye blink and even major physical movement such as head movement.

• It can be used on human or animal as subjects to get the brain activity. It uses a special hardware that place sensors around the top of the head area to sense any electrical brain activity.

• The EEG method is best to be applied for drowsiness and fatigue detection.

• In the method, EEG have four types of frequency components that can be analyzed, i.e. alpha (α), beta (β), theta (θ) and delta (δ). When the power is increased in alpha (α) and delta (δ) frequency bands, it shows that the driver is facing fatigue and drowsiness

• The disadvantages of this method are, it is very sensitive to noise around the sensors. For example, when the person is doing the EEG experiment, the surrounding area must be completely silent. The noise will interfere with the sensors that detect the brain activity.

• Another disadvantage of this method is that even if the result might be accurate, it is not suitable to use for real driving application [10]. Imagine when a person is driving and he is wearing something on his head with full of wires and when the driver moves their head, the wire

may strip off from their place

Fig 2.1 for EEG

Yawning Detection Method

- According to, drowsiness of a person can be observed by looking at their face and behavior. The author propose a method where drowsiness can be detected by mouth positioning and the images were process by using cascade of classifier that has been proposed by Viola-Jones for faces.

- The images were compared with the set of images data for mouth and yawning. Some people will close their mouth by their hand while yawning.

- It is an obstacle to get good images if a person is closing their mouth while yawning but yawning is definitely a sign of a person having drowsiness and fatigue

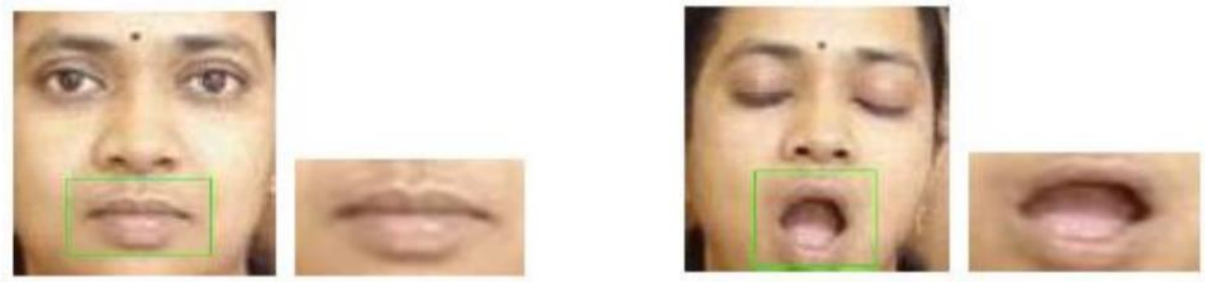


Fig 3.1 for the yawning detection method

:

Chapter 3: Problem Definition

Problem definition:

- Current drowsiness detection systems monitoring the driver's condition requires complex computation and expensive equipment, not comfortable to wear during driving and is not suitable for driving conditions; for example, Electroencephalography (EEG) and Electrocardiography (ECG), i. e. deecting the brain frequency and measuring the rhythm of heart, respectively.

- A drowsiness detection system which use a camera placed in front of the driver is more suitable to be use but the physical signs that will indicate drowsiness need to be located first in order to come up with a drowsiness detection algorithm that is reliable and accurate.

- Lighting intensity and while the driver tilt their face left or right are the problems occur during detection of eyes and mouth region.

- Therefore, this project aims to analyze all the previous research and method, hence propose a method to detect drowsiness by using video or webcam.

- It analyzes the video images that have been recorded and come up with a system that can analyze each frame of the video.

3.1 Objectives

The project focuses on these objectives, which are-

- To suggest ways to detect fatigue and drowsiness while driving.

- To study on eyes from the video images of participants in the experiment of driving simulation that can be used as an indicator of fatigue and drowsiness.

- To develop a system that use eyes closure and yawning as a way to detect fatigue and drowsiness

3.2 Proposed Models

- Drowsiness of a person can be measured by the extended period of time for which his/her eyes are in closed state.

- In our system, primary attention is given to the faster detection and processing of data.The number of frames for which eyes are closed is monitored. If the number of frames exceeds a certain value, then a warning message is generated on the display showing that

the driver is feeling drowsy.

- In our algorithm, first the image is acquired by the webcam for processing.
- Then we use the Haarcascade file face.xml to search and detect the faces in each individual frame.
- If no face is detected then another frame is acquired.
- If a face is detected, then a region of interest in marked within the face. This region of interest contains the eyes. Defining a region of interest significantly reduces the computational requirements of the system.
- After that the eyes are detected from the region of interest by using Haarcascade_eye.xml. If an eye is detected then there is no blink and the blink counter K is set to '0'.
- If the eyes are closed in a particular frame, then the blink counter is incremented and a blink is detected.
- When the eyes are closed for more than 4 frames then it is deducible that the driver is feeling drowsy. Hence drowsiness is detected and an alarm sounded. After that the whole process is repeated as long as the driver is driving the car.

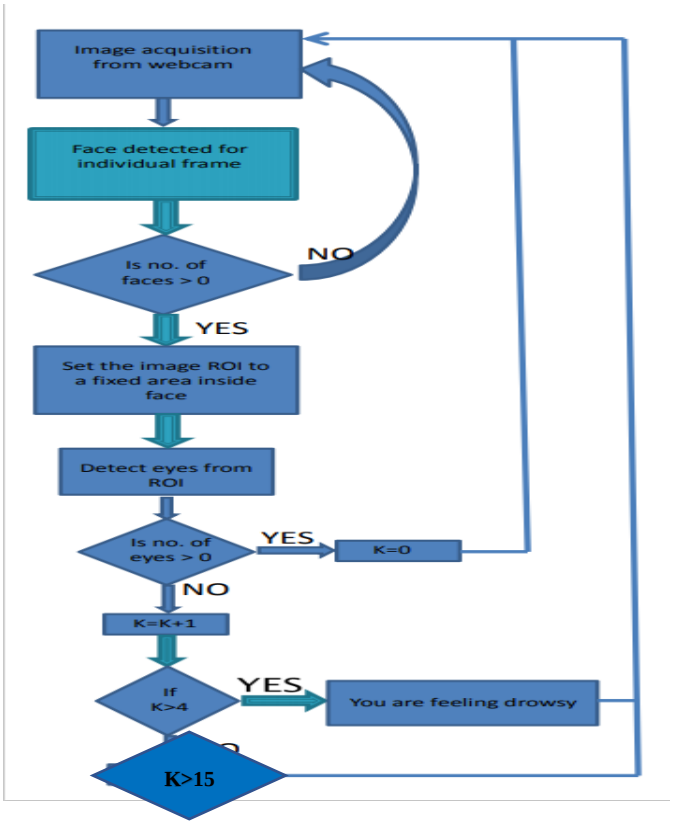


Fig 4.1 flowchart for drowsiness detection system

Chapter 4: Requirement Analysis

Requirement Analysis is the first and important phase of the software developing activity in developing any kind of project effectively. We started to list out all the functionalities that our application should provide. There are possibilities for minor changes with respect to the functionalities over the course of development. following are the requirements that have been considered: -

4.1 Requirement Specification

4.1.1 Functional Requirements

- Tensorflow.js- TensorFlow.js is an open source WebGL-accelerated JavaScript library for machine intelligence. It brings highly performant machine learning building blocks to your fingertips, allowing you to train neural networks in a browser or run pre-trained models in inference mode.

- Keras- It is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon user error. This makes it easy to learn and easy to use. As a Keras user, you are more productive, allowing you to try more ideas than your competition, faster -- which in turn helps you win machine learning competitions.

- Python3 - Python language is one of the most flexible languages and can be used for various purposes. Python has gained huge popularity base of this. Python does contain special libraries for machine learning namely scipy and numpy which great for linear algebra and getting to know kernel methods of machine learning. The language is great to use when working with machine learning algorithms and has easy syntax relatively. For beginners, this is the best language to use and to start with.

- Webcam - A webcam is a video camera that feeds or streams its image in real time to or through a computer to a computer network. When "captured" by the computer, the video stream

may be saved, viewed or sent on to other networks travelling through systems such as the internet, and e-mailed as an attachment. When sent to a remote location, the video stream may be saved, viewed or on sent there.

- PyGame- pygame is a Python wrapper for the SDL library, which stands for Simple DirectMedia Layer. SDL provides cross-platform access to your system's underlying multimedia hardware components, such as sound, video, mouse, keyboard, and joystick. So in the project we will use it for the alarm purpose

4.1.2 Non-Functional Requirements

Non-functional requirements are not directly related to the functional behaviour of the system.

- Application must be user friendly, simple and interactive.

- The user interface is designed in such way that users with little knowledge, should be able to access this application.

4.1.3 Software Specifications

- Operating System: Windows 7 or higher, Linux

- IDE: Python IDE

- Language used: Python

4.1.4 Hardware Specifications

- Microphone, Webcam

- LCD display
- Processor: Pentium dual core
- Processor speed: 1.6GHz (minimum)
- RAM: 1 GB
- Disk Space: 250 MB or higher

Chapter 5: System Design

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Overall product architecture, the subsystems that compose the product, and the way subsystems are allocated to processors are depicted using the System Design. UML is used to model system designs. Unified Modelling Language is a standard object-oriented analysis and design language. Use Case diagram and Sequence diagram, which are types of UML diagrams, of the application are shown below.

5.1 Class Diagram

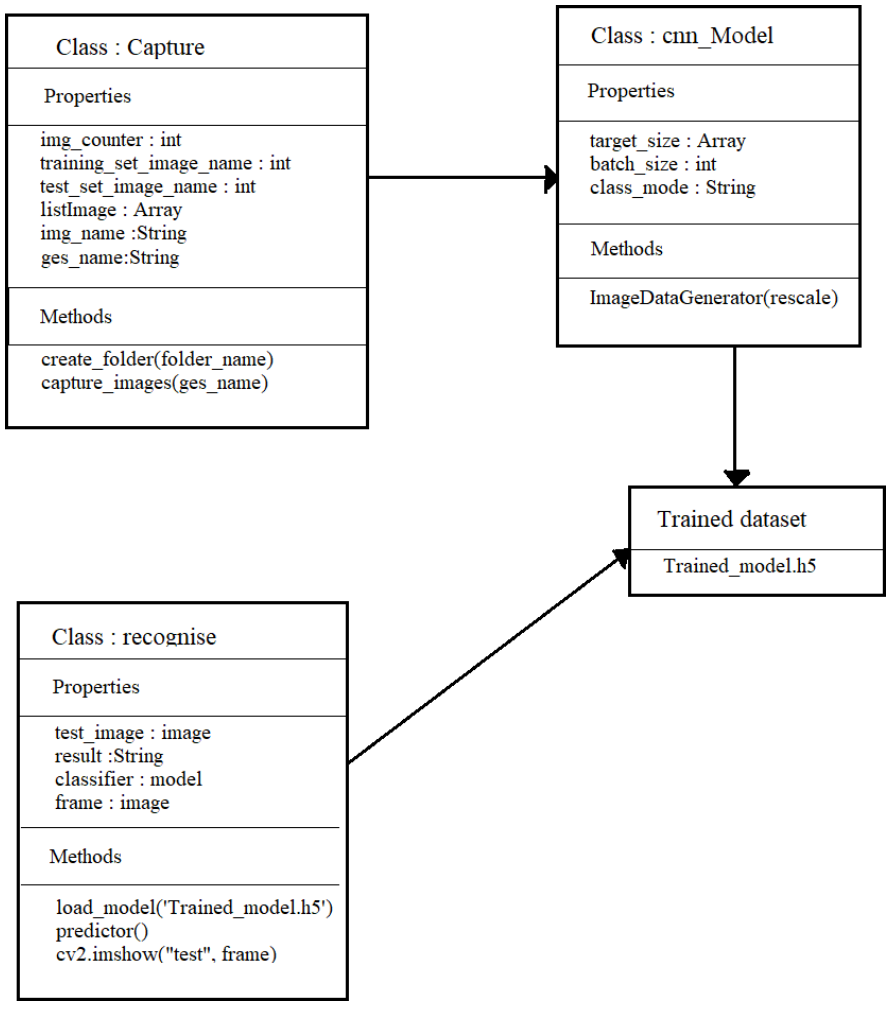


Fig No 5.1-Class Diagram

5.1 Use Case Diagram

A Use Case Diagram consists of set of elements and the relationships between them. It depicts all the scenarios, regarding how our application interacts with users and other external systems to achieve the goals of application. The main components of a use case diagram include actors, use cases and their relationships. The use case is an external view of the system that represents some actions that the user performs to get a job done. Actors are the users who interact with the application.

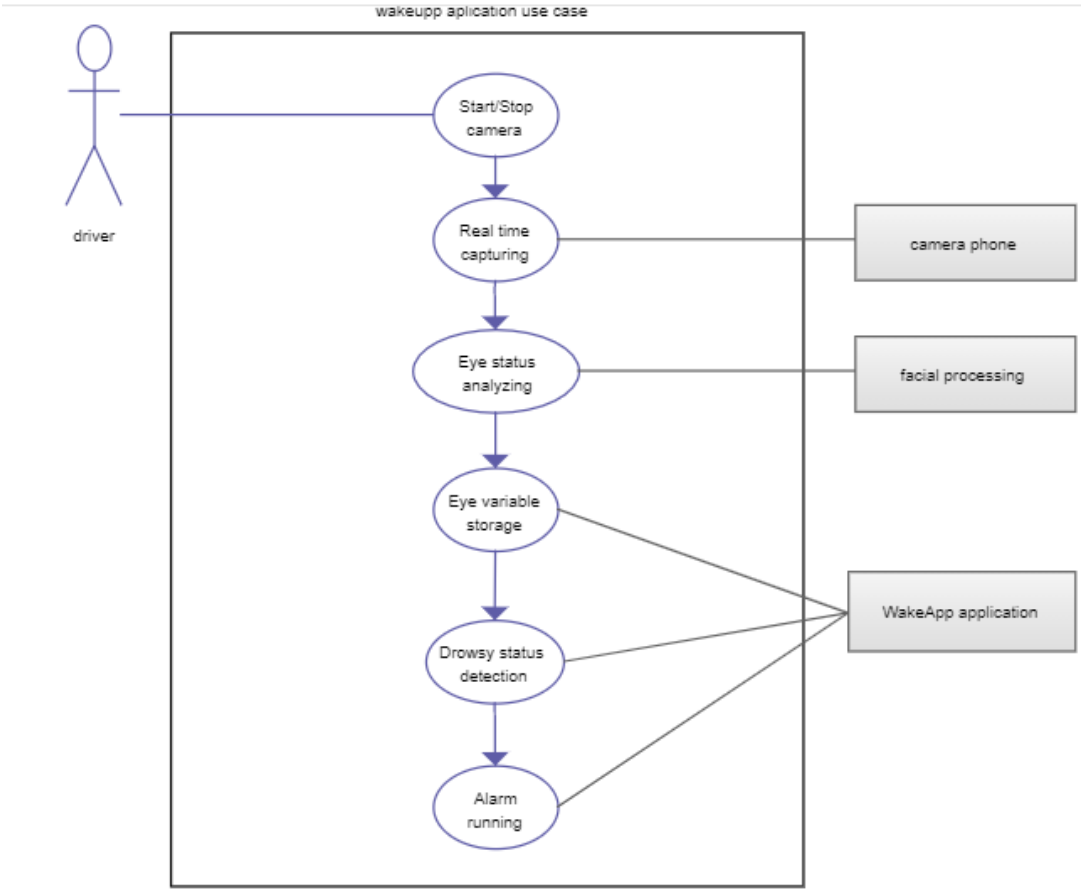


Fig No 5.2-Use Case Diagram

5.2 Sequence Diagram

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes.

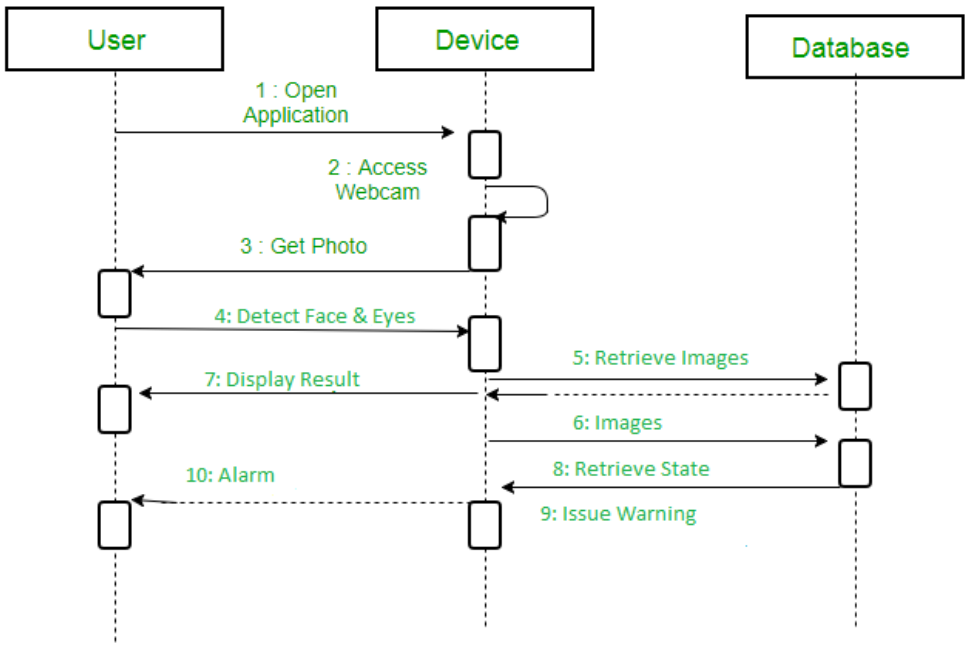


Fig No 5.3-Sequence Diagram

. Chapter 6: System Testing

In this we have used use case testing to test our system.

Use Case Testing is a functional black box testing technique that helps testers to identify test scenarios that exercise the whole system on each transaction basis from start to finish.

6.1. Test Cases & Test Results

Test ID	Score	Test Condition	System Behavior	Expected Result
T01	0	Pass	No Alarm	No Alarm
T02	3	Pass	No Alarm	No Alarm
T03	8	Pass	No Alarm	No Alarm
T04	13	Pass	No Alarm	No Alarm
T05	15	Pass	No Alarm	No Alarm
T06	16	Pass	Alarm	Alarm
T07	17	Pass	Alarm	Alarm
T08	24	Pass	Alarm	Alarm

Chapter 7: Implementation

In this project we used different libraries of python.

All are listed below:

- OpenCv: OpenCV is a library using which we have developed real-time computer vision application which focuses on image processing, video capture and analysis including features like face detection.
- OS: OS module provides functions for interacting with the operating system. The os and os.path modules include many functions to interact with the file system.
- Keras: We used keras for developing and evaluating our model. It wraps the efficient numerical computation library TensorFlow and allows us to define and train neural network models in just a few lines of code. H5 is a file format to store structured data. Keras saves models in this format as it can easily store the weights and model configuration in a single file.
- NumPy: NumPy is a general-purpose array processing package which provides tools for handling the n-dimensional arrays. It provides various computing tools such as comprehensive mathematical functions, linear algebra routines. NumPy provides both the flexibility of Python and the speed of well-optimized compiled C code.
- Pygame: We used sound libraries for alarming sound.
  - Mixer module helps control the music used in pygame programs.
- Time module: The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of our code

: Code:

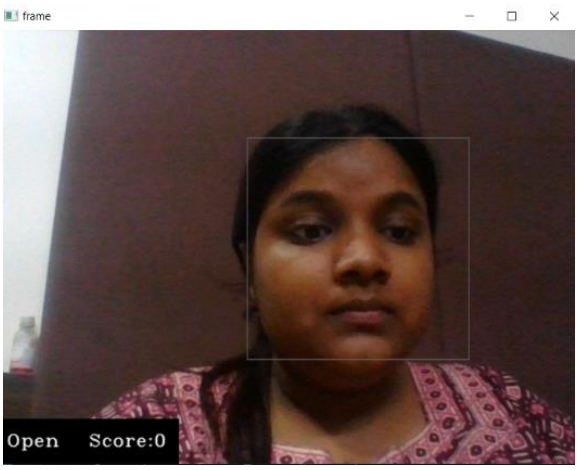
```
import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time


mixer.init()
sound = mixer.Sound('alarm.wav')

face = cv2.CascadeClassifier('haar cascade files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haar cascade files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haar cascade files\haarcascade_righteye_2splits.xml')



lbl=['Close','Open']

model = load_model('models/cnncat2.h5')
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count=0
score=0
```

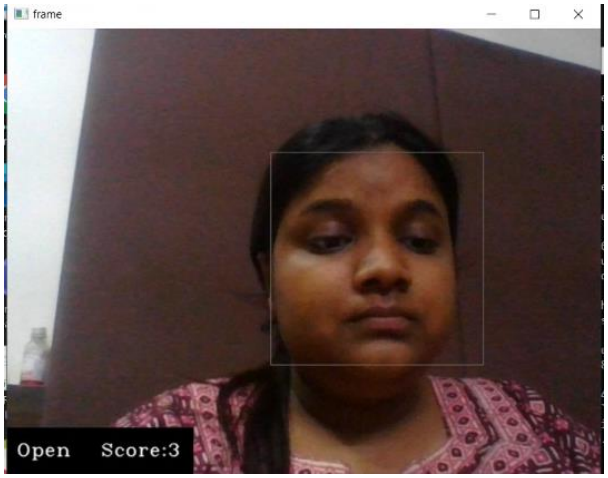
```
thicc=2
rpred=[99]
lpred=[99]

while(True):
    ret, frame = cap.read()
    height,width = frame.shape[:2]

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))
    left_eye = leye.detectMultiScale(gray)
    right_eye =  reye.detectMultiScale(gray)

    cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) , thickness=cv2.FILLED )

    for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )

    for (x,y,w,h) in right_eye:
        r_eye=frame[y:y+h,x:x+w]
        count=count+1
        r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
        r_eye = cv2.resize(r_eye,(24,24))
r_eye= r_eye/255
        r_eye=  r_eye.reshape(24,24,-1)
        r_eye = np.expand_dims(r_eye,axis=0)
        rpred = model.predict_classes(r_eye)
        if(rpred[0]==1):
            lbl='Open'
```

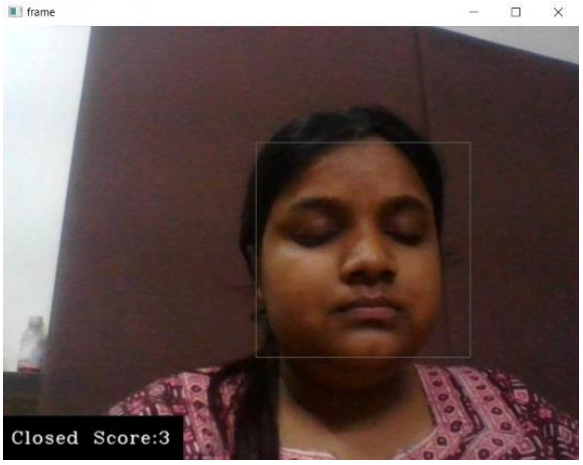
```
    if(rpred[0]==0):
        lbl='Closed'
    break

for (x,y,w,h) in left_eye:
    l_eye=frame[y:y+h,x:x+w]
    count=count+1
    l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
    l_eye = cv2.resize(l_eye,(24,24))
    l_eye= l_eye/255
    l_eye=l_eye.reshape(24,24,-1)
    l_eye = np.expand_dims(l_eye,axis=0)
    lpred = model.predict_classes(l_eye)
    if(lpred[0]==1):
        lbl='Open'
    if(lpred[0]==0):
        lbl='Closed'
    break

if(rpred[0]==0 and lpred[0]==0):
   score=score+1
```

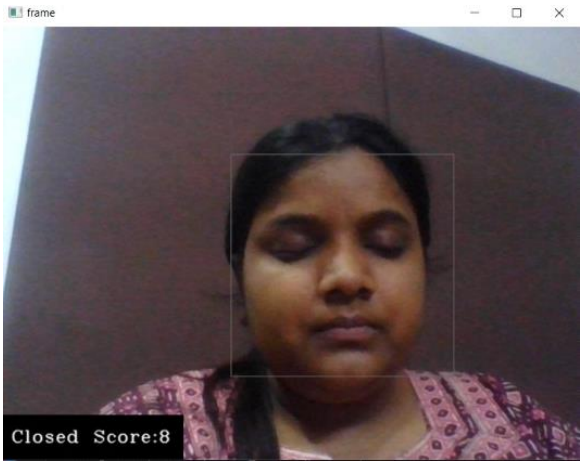
```
cv2.putText(frame,"Closed",(10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)
    # if(rpred[0]==1 or lpred[0]==1):
    else:
        score=score-1
        cv2.putText(frame,"Open",(10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)



    if(score<0):
        score=0
    cv2.putText(frame,'Score:'+str(score),(100,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)
    if(score>15):
        #person is feeling sleepy so we beep the alarm
        cv2.imwrite(os.path.join(path,'image.jpg'),frame)
        try:
            sound.play()

        except:  # isplaying = False
            pass
        if(thicc<16):
            thicc= thicc+2
        else:
            thicc=thicc-2
            if(thicc<2):
                thicc=2
cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWind
```

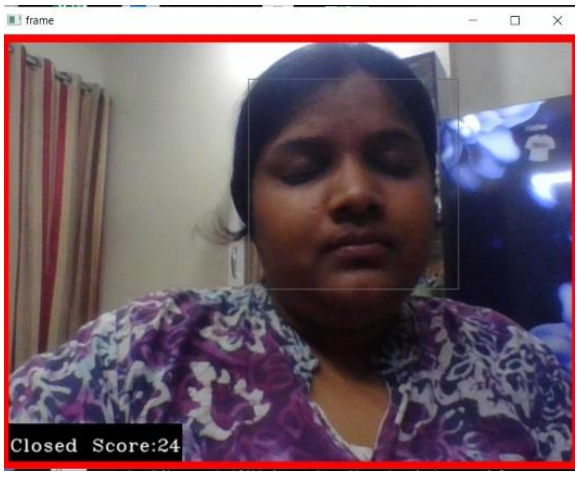
Chapter 6: Results

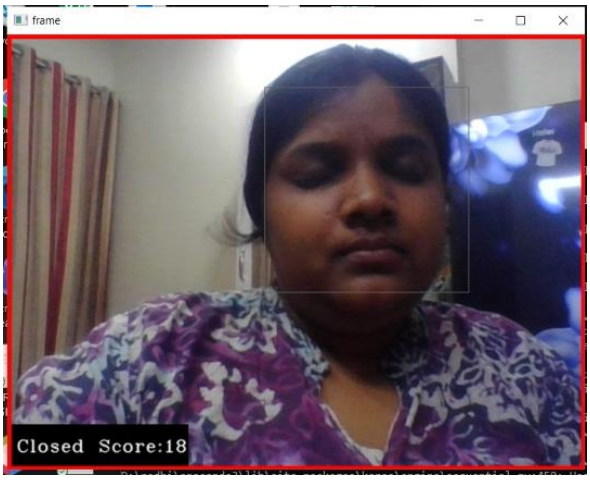












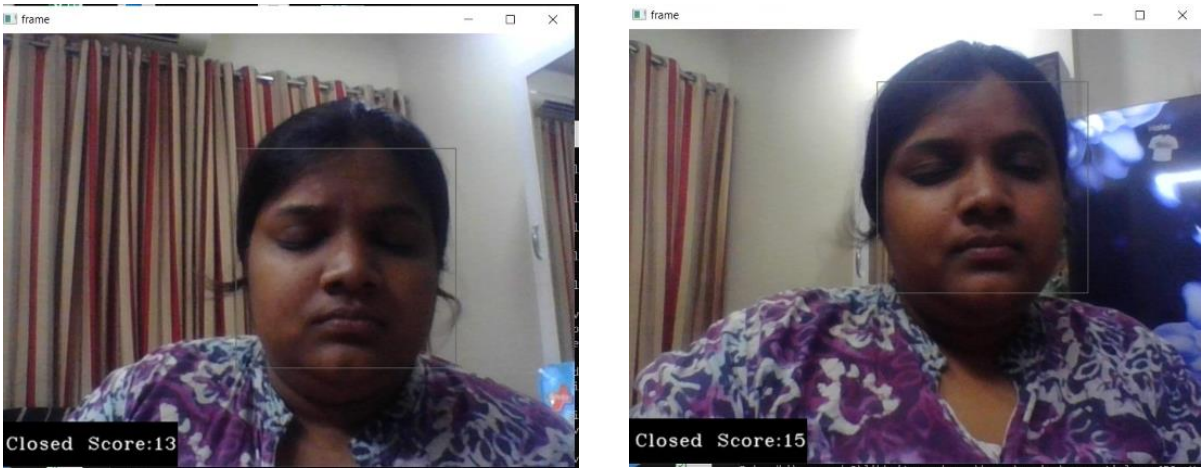


Fig no 6.1 of result

Chapter 7: Conclusion

The purpose of the drowsiness detection system is to aid in the prevention of accidents passenger and commercial vehicles. The system will detect the early symptoms of drowsiness before the driver has fully lost all attentiveness and warn the driver that they are no longer capable of operating the vehicle safely. This device will not, however, guarantee that the driver will be fully awakened and that an accident will be avoided. It is simply a tool for improving driver safety; focusing primarily on long-haul truck drivers, nighttime drivers, people driving long distances alone or people suffering from sleep deprivation.

Thus we have successfully designed a prototype drowsiness detection system using OpenCV software and Haar Classifiers. The system so developed was successfully tested, its limitations identified and a future plan of action developed

References

- https://data-flair.training/blogs/python-project-driver-drowsiness-detection-system/
- http://ethesis.nitrkl.ac.in/3373/1/thesis-108EI038-026.pdf
- http://utpedia.utp.edu.my/13469/1/badiuzaman.pdf
- https://www.bogotobogo.com/cplusplus/files/OReilly%20Learning%20OpenCV.pdf