DRIVER DROWSINESS DETECTION USING DEEP LEARNING

A SEMINAR REPORT

Submitted by

ADITHYA R. - RA2011003020093

ADITYA J. P. - RA2011003020090

SUBASH ARAVINDH B. - RA2011003020095

under the guidance of

Ms. Kiruthika S. S.

Mr. Vadivelu A.

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

RAMAPURAM CAMPUS, CHENNAI-600089

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that the Seminar-II report titled "**DRIVER DROWSINESS DETECTION USING DEEP LEARNING**" is the bonafide work of **ADITHYA R[RA2011003020093], ADITYA JP[RA2011003020090], SUBASH ARAVINDH B[RA2011003020095]**" submitted for the course 18CSP106L Seminar – II. This report is a record of successful completion of the specified course evaluated based on literature reviews and the supervisor. No part of the Seminar Report has been submitted for any degree, diploma, title, or recognition before.

SUPERVISOR SIGNATURE	**HOD SIGNATURE**
Ms. S. S. Kiruthika	Dr. K. Raja
Assistant Professor	Head of Department
Dept. of Computer Science & Engineering	Dept. of Computer Science & Engineering
SRM Ramapuram	SRM Ramapuram

Submitted for the Viva Voce Examination held on ………………..... at SRM Institute of Science and Technology, Ramapuram Campus, Chennai-600089.

EXAMINER 1 **EXAMINER 2**

ABSTRACT

Nowadays, accidents are becoming a part of our life. It happens due to people's careless behaviours such as rash driving ,hearing songs or chatting with people with the help of earphones and driving which will lead to decrease in his concentration and is also the reason for death rates, injuries etc. Another factor which is to be taken is late night driving, especially midnight drives in lonely areas. When people tend to travel to long distances in night, automatically they tend to sleep as they are tired and when they drive with that drowsiness, they intend to hit the car in the tree or some other vehicle which are the reason for accidents or deaths. These things are known to us through social media, newspapers etc. To avoid this we have developed an drowsiness detection system which will detect whether the driver is drowsy especially during nights, it will alert the user with a signal so that driver will wake up. This system will be placed in car's gear. To implement this project, We use the concept of deep learning which will directly extract the features and classify the data. The algorithm implemented in this project, is Haar Cascades algorithm which will detect the face of the driver and check whether he is active or not.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	-	Artificial Intelligence
Open CV-ED	-	Open CV Eye Detector

CHAPTER 1

INTRODUCTION

Driver drowsiness detection systems can use cameras, eye tracking sensors and other hardware to monitor visual cues, where drowsiness can be detected through yawning frequency, eye-blinking frequency, eye-gaze movement, head movement and facial expressions. The systems can also monitor driving input behaviour to notice when there are erratic steering movements, pedal use and lane deviations. Different car models have different systems, but in most cases the driver is alerted to their potential drift of attention with some sort of noise, or vibration of the steering wheel or seat. The system may also remind the driver to take a break, especially if they have been on the road for an extended period. There are other independent devices that the driver can place in their vehicle themselves, which help to reduce the risks associated with loss of concentration while driving. Anti pilot tablet is also implemented using 26 different parameters, with the device resting on your dashboard. Wearable spectacles are coming in the form of caps, vests, wristbands and eyewear. Aimed at truck drivers, they monitor a variety of signals from the driver that can be used to detect if the driver is nodding off, with a headset that rests just beneath the driver's eyeline to analyse their head movements.

CHAPTER 2

PROBLEM STATEMENT

This project will define the problem of drowsiness which is common to the drivers who drive during night-time especially midnight .it also detects the face of the driver and tells whether he is active or drowsy. Implementation of this project is done by Haar Cascades algorithm. It uses computer vision and image classification techniques to classify a man's face and tell accordingly whether he is drowsy.it will look at the eyes of the driver and if the eyes are down, it will give an alert signal indicating that driver is sleeping. the sound will be very high if his eyes are fully closed. With this application, we can avoid many accidents and deaths which are taking place due to drowsiness

CHAPTER 3

SCOPE AND OBJECTIVE

3.1 Scope

This system will also feature of IOT devices which will be installed in the car by this system. This will also consider the features like weather conditions, sleeping hours of driver, state of the vehicle. Using angle sensor mounted on the steering column, driver steering behaviour is being measured**.**

3.2 Objective

The objective of this paper is to prepare a drowsiness system using deep learning. This method provides a technique for detection techniques like different ways of driver alertness and facilitates early detection of decline in alertness during travel. If any issues occur, he is warned with a different signal. If he does not respond to the signal, it will slow down the vehicle or stops the vehicle. This is done by the construction of automatic sensors in the vehicle.

CHAPTER 4

EXISTING SYSTEM

By using a non-intrusive machine vision-based concepts, drowsy/weariness of the driver detected system is developed. Many existing systems require a camera which is installed in front of driver. It points straight towards the face of the driver and monitors the driver's eyes to identify the drowsiness. For large vehicle such as heavy trucks and buses this arrangement is not pertinent. Bus has a large front glass window to have a broad view for safe driving. If we place a camera on the window of front glass, the camera blocks the frontal view of driver, so it is not practical. If the camera is placed on the frame which is just about the window, then the camera is unable to detain the anterior view of the face of the driver correctly. The open CV detector detects only 40% of face of driver in normal driving position in video recording of 10 minutes. In the oblique view, the Open CV eye detector (CV-ED) frequently or continuously fails to trace the pair of eyes. If the eyes are closed for five successive frames the system or machine concludes that the driver is declining slumbering and issues a warning signal. Hence existing system is not applicable for large vehicles. To conquer the problem of existing system, new detection system is developed in this project work.

CHAPTER 5

LITERATURE SURVEY

S. No.	Journal Name	Year	Title Name	Author	Description
1	IEEE transactions on structures, man, and cybernetics-component a: structures and humans	2008	Least Square regression for drowsiness detection	su.h and Zheng	This paper proposes a substitute generation of modelling driving force drowsiness supported statistics fusion approach with multiple eyelid motion characteristics - partial minimal squares regression (plsr), with which there is a sturdy connection between the eyelid movement features and therefore the tendency to drowsiness, to influence it threat hassle. The precarious accuracy and sturdiness of the version accordingly established has been validated, suggesting that it offers an alternative way of concurrently multi-fusing to extend our capacity to hit upon and are expecting drowsiness.
2	IEEE transaction on automobiles	2010	Digital camera-primarily based drowsiness reference for driving force kingdom classification underneath actual driving conditions	Friedrichs's and Zang	This paper proposes a substitute generation of modelling driving force drowsiness supported statistics fusion approach with multiple eyelid motion characteristics - partial minimal squares regression (plsr), with which there is a sturdy connection between the eyelid movement features and therefore the tendency to drowsiness, to influence it threat hassle.

					The precarious accuracy and sturdiness of the version accordingly established has been validated, suggesting that it offers an alternative way of concurrently multi-fusing to extend our capacity to hit upon and are expecting drowsiness.
3	IET intelligent delivery systems	2011	Driver drowsiness detection gadget below infrared illumination for a wise vehicle	Flores, m. J., Armengol, j. M., & de l. A. Escalera	In this paper to lower the range of fatalities, a module for a complicated driving force assistance device that automatically detects motive force drowsiness and additionally facilitates driving force distraction. Synthetic intelligence algorithms are used to system visual statistics to identify, song and examine each the driver's facial features and eyes to calculate the drowsiness index. This actual-time device operates at night-time because of the close to-infrared lighting machine. Ultimately, examples of different motive force pictures taken in a real automobile overnight are proven to verify the proposed set of rules.
4	IEEE transactions on intelligent transportation systems.	2013	Drowsiness Detection by visual analysis of eye and head state	Mbouna, R. O., Kong, S. G., & Chun M. G	In this paper, the eye position and head posture (HP) of a driver are continuously monitored for automobile alertness. The previous approaches of driver alertness monitoring used visual features to determine the driver alertness the visual features include eye

					closure, shaking of the head, etc. These were used to measure the level of drowsiness or the level of distraction. But this paper proposed a scheme that uses visual features along with eye indicator (EI), pupil interest (PA) and HP to acquire essential data on driver alertness. SVM or support vector machine classifies the video segments into caution or non-warning driving events. Experimental results showed that this system offers higher accuracy with very few errors and fake alarms for people of many different races and genders in actual street-like driving conditions.
5	IEEE Transaction	2014	Driver drowsiness detection through HMM based dynamic modelling	Tadesse, E., Sheng, W., Liu.M	In this paper they proposed a unique method of analysing the driver's facial expressions to detect drowsiness through dynamic modelling based on the Hidden Markov Model (HMM). This paper implemented the HMM algorithm by using a driving simulator setup to mimic driving conditions. But the experimental outcomes confirmed the effectiveness of this proposed approach.

Table 5.1 Literature Survey

CHAPTER 6

PROPOSED SYSTEM

Driver drowsiness detection using deep learning is the proposed system that aims to detect when a driver is feeling drowsy or sleepy while driving. The system can be implemented in different ways, but typically it involves using sensors to monitor the driver's behaviour and physiological indicators, such as eye movements, head position and mouth position. In the first module, this system detects whether the driver is drowsy or not by considering eyes and facial landmarks. The first step to it is to perform face recognition to detect blink of an eye, where the system recognizes the driver's eyes and then calculates the speed of eye blink and provides an appropriate output. HAAR cascade method uses OpenCV to detect the human faces. Human eye is dark in colour and the nose is bright in colour. Hence Haar cascade technique is used for extracting the face information using OpenCV. This method recognizes the position of the eye on the human dimension. In the system the face is monitored continuously till he/she stops the functioning of it. The mean shift method is used for finding the centre of image for image distribution, where the data is contributed on a plane. To achieve the dense part of the image, we must obtain the position and radius of the image. Eye detection, Dlib is the open-source library used for detecting the blink and calculating the eye aspect. ratio was the eye aspect. ratio has the threshold of 0.3 if the threshold is constant than system consider as the eyes are open, if the threshold value is decreased from 0.3 than system will consider as the drowsy state. The second module is facing recognition system. Here we're using object detection techniques in which features of faces will be captured. Positive and negative images are taken, and we feed the data to the system. Haar cascade is an object detection set of rules which is utilized in gadget learning proposed via Paul viola and Michael jones the paper "fast object detection the usage of a boosted cascade of easy functions" in 2002. On this set of rules, the device is skilled by means of taking a couple of photos of the users. Right here we will be taking a fixed of high-quality photographs and a set of terrible photos which we are able to be importing the information in the device. It's far then used to detect different objects present in the other pictures. Here, OpenCV offers pre-trained Haar cascade algorithms, that are organized into classes relying at the pictures given by dividing as test and train Datas.

6.1 Architecture

The device camera takes the live video of the driver as an input, the eye aspect ratio of the driver is taken into consideration to determine drowsiness. If both the left and right eye are detected to be closed for a considerable time an alert is set off classifying the driver to be drowsiness.

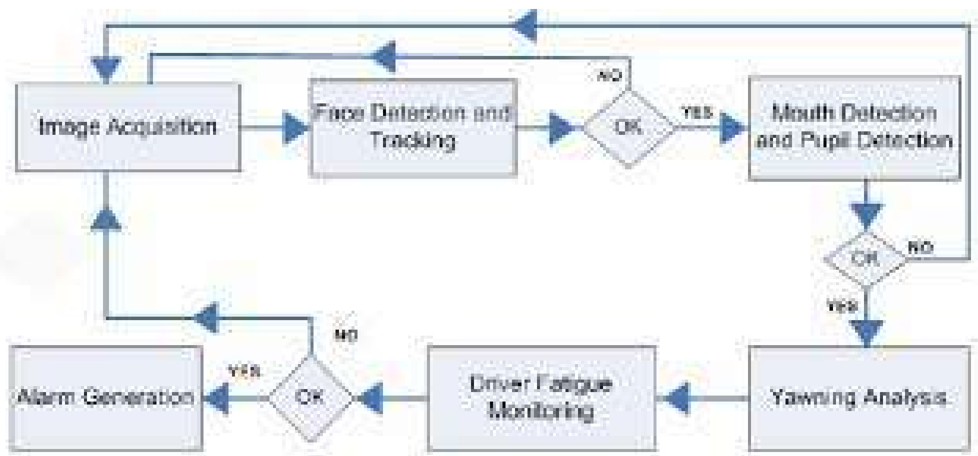


Fig.6.1.1 System Architecture

CHAPTER 7

MODULE DESCRIPTION

List of modules used are:

1. Data pre-processing
2. Feature Extraction
3. Training the system
4. Testing the system

7.1 Data Pre-processing:

The images are tested with the proposed algorithm, pre-processing is accomplished which is the primary stage of any face recognition gadget. A brand-new approach of pre-processing has been proposed for face recognition packages beneath uncontrolled and difficult lighting fixtures situations.

7.2 Feature Extraction:

As in brief alluded to in advance, primarily based at the facial landmarks that we extracted from the frames of the videos, we ventured into growing appropriate capabilities for our class version. At the same time as we hypothesized and examined several functions, the four core features that we concluded on for our very last models had been eye element ratio, mouth thing ratio, student circularity, and sooner or later, mouth element ratio over eye thing ratio.

7.3 Training the system:

To do face recognition, face recognizer must learn. The use of the pre-categorized dataset, we have designed a label “dataset” for our face recognition system, which is now used to use that dataset, which first trains the face identifier to use the dataset in OpenCV Python. Python Trainer in the same folder.p "File, and then create a folder in the same directory" Trainer. ", This is the binder where we will save our identifier after coaching. For our training and test data for drowsiness detection, we used the (shape_predictor_68_face_landmarks.dat).

7.4 Testing the system:

In this system we're adding up two different models, that is drowsiness detection and face recognition for safety and security purposes of the driver. First, we will be popping up with the GUI window which has two buttons, one is drowsiness detection and other is face recognition system. If we click the button drowsiness detection it will redirect you to the alert system. Here GUI consists of the live feed which is continuously fed to the camera with the bounding boxes detects the eyes and the face of the operator and if the operator closes his eyes for 3 - 4 seconds an alarm system will be activated. If you will be selecting face recognition you will be redirected to the face recognition model. Once camera captures your image it will verify with the data provided and it recognizes the data and displays the driver's name.

CODING:

Module 1: Face recognition System

```
import cv2
import os
import numpy as np
import PIL.Image

def face_train():

  face_id = None
  userLabel = Label(frame2, text = "Enter the user name: ",bg="gray47", font = f2)
  userLabel.grid(row=1, column= 0, padx=5, pady = 5)
  userEntry = Entry(frame2, width=10)
  userEntry.grid(row=1, column=1, columnspan = 2, padx=2)
  log.configure(text = "Type name and press ENTER")
  root.update()
  def setFaceId(event):
    username = userEntry.get().lower()
    log.configure(text = "[INFO] Initializing face capture. Look at the camera and wait
...")
    root.update()
    train(username)

  userEntry.bind('<Return>', setFaceId)

  def train(username):
    def getKey(val):
      for key, value in names.items():
        if val == value:
          return key

    if username not in names.values():
      last = len(names)
      names[last] = username
      l = Label(frame3, text = str(last)+". "+ username.title(), bg="gray47")
      l.grid(row = last+1, column = 0, sticky = W)
      root.update()
```

```
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Initialize individual sampling face count
count = 0
log.configure(text = "Collecting data...")
root.update()
while(True):

 ret, img = cam.read()
   #img = cv2.flip(img, -1) # flip video image vertically
   gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
   faces = face_detector.detectMultiScale(gray, 1.3, 5)

   for (x,y,w,h) in faces:

      cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
      count += 1

      # Save the captured image into the datasets folder
      cv2.imwrite("./dataset/" + username + '.' + str(last) + '.' + str(count) + ".jpg",
gray[y:y+h,x:x+w])

      cv2.imshow('image', img)
      #print("dataset/User." + str(face_id) + '.' + str(count) + ".jpg")

   k = cv2.waitKey(100) & 0xff # Press 'ESC' for exiting video

   if k == 27:
      break
   elif count >= 30: # Take 30 face sample and stop video
       break

# Do a bit of cleanup
log.configure(text = "[INFO] Finished.")
root.update()

cam.release()
cv2.destroyAllWindows()
```

```
# Initialize and start realtime video capture
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video widht
cam.set(4, 480) # set video height

# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)

while True:

   ret, img =cam.read()
   #img = cv2.flip(img, -1) # Flip vertically

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

   faces = faceCascade.detectMultiScale(
      gray,
      scaleFactor = 1.2,
      minNeighbors = 5,
      minSize = (int(minW), int(minH)),
     )

   for(x,y,w,h) in faces:

      cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

      id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

      # Check if confidence is less them 100 ==> "0" is perfect match
      if (confidence < 100):
         id = names[id]
         confidence = " {0}%".format(round(100 - confidence))
      else:
         id = "unknown"
         confidence = " {0}%".format(round(100 - confidence))

      cv2.putText(img, str(id).title(), (x+5,y-5), font, 1, (255,255,255), 2)
```

```
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");


# function to get the images and label data
def getImagesAndLabels(path):

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []

    for imagePath in imagePaths:
        PIL_img = PIL.Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')
        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)

        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)

    return faceSamples,ids

print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
log.configure(text = "[INFO] Training faces. It will take a few seconds. Wait ...")
root.update()
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))

# Save the model into trainer/trainer.yml
recognizer.write('trainer.yml') # recognizer.save() worked on Mac, but not on Pi

# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained".format(len(np.unique(ids))))
log.configure(text = "[INFO] {0} faces trained.".format(len(np.unique(ids))) + " Press
Esc to quit")
root.update()

        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

        id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

        # Check if confidence is less them 100 ==> "0" is perfect match
        if (confidence < 100):
            id = names[id]
            confidence = "  {0}%".format(round(100 - confidence))
        else:
            id = "unknown"
            confidence = "  {0}%".format(round(100 - confidence))

        cv2.putText(img, str(id).title(), (x+5,y-5), font, 1, (255,255,255), 2)
        cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)

    cv2.imshow('Detecting...',img)

    k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break

# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
```

MODULE 2: Drowsiness Detection system

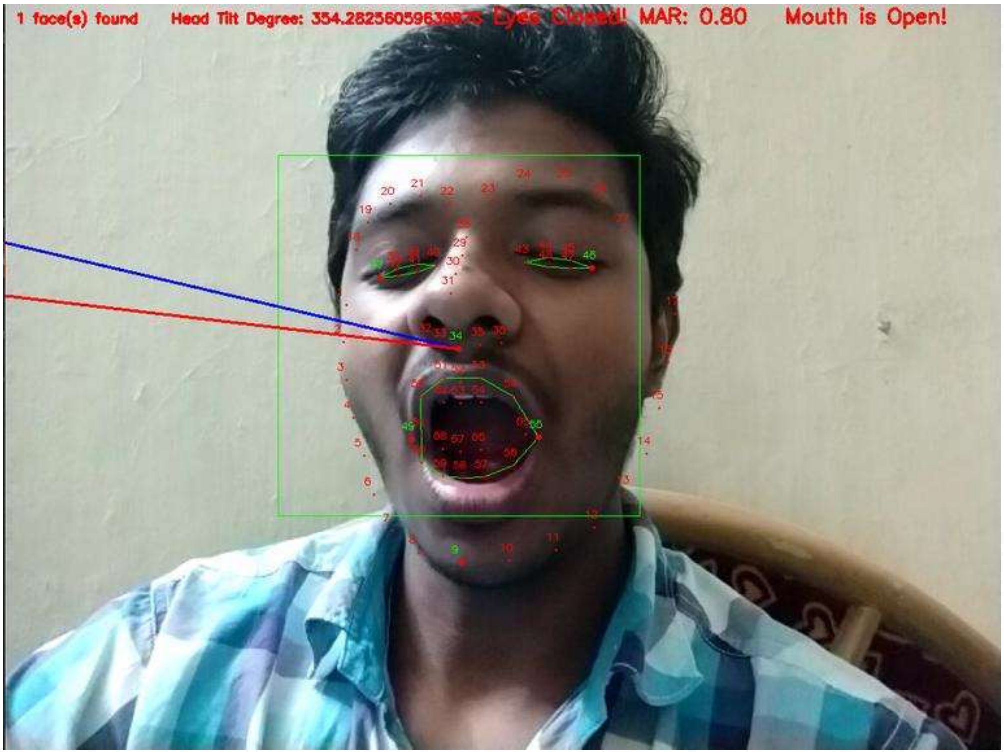
```
from scipy.spatial import distance
from imutils import face_utils
import imutils
import dlib
import cv2
import pygame

def drowsiness():
    log.configure(text = "[INFO] App starting. Press Esc to quit")
    root.update()

    def eye_aspect_ratio(eye):
        A = distance.euclidean(eye[1], eye[5])
        B = distance.euclidean(eye[2], eye[4])
        C = distance.euclidean(eye[0], eye[3])
        ear = (A + B) / (2.0 * C)
        return ear

    thresh = 0.25
    frame_check = 20
    detect = dlib.get_frontal_face_detector()
    predict = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")


    (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
    (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]


    #Initialize Pygame and load music
    import pygame
    pygame.mixer.init()
    pygame.mixer.music.load('beep-04.mp3')
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        subjects = detect(gray, 0)
        for subject in subjects:
            shape = predict(gray, subject)
 shape = face_utils.shape_to_np(shape)#converting to NumPy Array
            leftEye = shape[lStart:lEnd]
            rightEye = shape[rStart:rEnd]
            leftEAR = eye_aspect_ratio(leftEye)
            rightEAR = eye_aspect_ratio(rightEye)
 ear = (leftEAR + rightEAR) / 2.0
            leftEyeHull = cv2.convexHull(leftEye)
            rightEyeHull = cv2.convexHull(rightEye)
            cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
            cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
            if ear < thresh:
                flag += 1
                #print (flag)
                if flag >= frame_check:
                    pygame.mixer.music.play(-1)
        cv2.putText(frame, "****************ALERT!****************", (10, 30),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                    cv2.putText(frame,   "**************Driver   is   drowsy**************",
(10,325),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

            else:
                pygame.mixer.music.stop()
                flag = 0

        cv2.imshow("Drowsiness Detector", frame)
        key = cv2.waitKey(10) & 0xFF
        if key == 27:
            pygame.mixer.music.stop()
            cv2.destroyAllWindows()
            cap.release()
            break
```

```
        l.grid(row = 1, column = 0, sticky = W)

    for i in range(len(names)):
        l = Label(frame3, text = str(i)+". "+names[i].title(),bg="gray47")
        l.grid(row = i+1, column = 0, sticky = W)
    frame3.place(relx = 0.7, rely = 0.4, anchor = NW)

  return

fButton = Button(frame1, text="Face Recognition",bg="gray63", font = f3, command =
face_recog)
fButton.grid(row=1, column = 0, padx = 10, pady = 10)

dButton = Button(frame1, text="Drowsiness Detection",bg="gray63", font = f3, command
= drowsiness)
dButton.grid(row=1, column = 1, padx = 10, pady = 10)
frame1.place(relx = 0.5, rely = 0.2, anchor = CENTER)

frame4 = Frame(root, bg = "gray27", bd = 5, height = 100, width = 300)
frame4.place(relx=0, rely=1, anchor = SW)
frame4.pack_propagate(0)

log = Label(root, text = "", bg = "gray27", font = f4)
log.place(relx = 0, rely = 1, anchor = SW)

root.mainloop()
```

MODULE 3: GUI

```
from tkinter import *

# names related to ids: example ==> Marcelo: id=1, etc
names = {}

root = Tk()
root.title("Face Recognition and Drowsiness Detection")
root.geometry("500x350")
root.configure(bg="gray27")

# Fonts
f1 = ("Comic Sans MS", 15)
f2 = ("Comic Sans MS", 10)
f3 = ("Arial Bold", 10)
f4 = ("Nimbus Mono L", 9)

frame1 = Frame(root,bg="gray47", bd = 5, relief = GROOVE)
menu = Label(frame1, text="Choose an option from below: ",bg="gray47", font = f1)
menu.grid(row=0, column = 0, columnspan=2, padx = 10, pady=10)

path = "dataset"
imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
frame2 = Frame(root,bg="gray47", bd = 5, relief = GROOVE)
frame3 = Frame(root,bg="gray47", bd = 5, relief = GROOVE)

def face_recog():
    for imagePath in imagePaths:
        id = int(os.path.split(imagePath)[-1].split(".")[1])
name = str(os.path.split(imagePath)[-1].split(".")[0])
        if id not in names.keys():
            names[id] = name

    lbl1 = Label(frame2, text="Train/Predict?",bg="gray47", font = f2)
    lbl1.grid(row=0, column=0, pady=10)
    train = Button(frame2, text = "Train",bg="gray63", font = f3, command = face_train)
```

OUTPUT:



CHAPTER 8

DISADVANTAGES

Few of the models don't support night driving conditions or glasses, etc. Only few systems support real-time face recognition. No existing systems including both drowsiness detection system and vehicle security system using face recognition.

CHAPTER 9

FUTURE SCOPE

In the future we can enhance these models by adding zoom in features, where the system can predict at a high-rate accuracy, and it will be more efficient. With the update of technology and AI techniques we can completely upgrade our system such that cameras will be only able to capture the eyes of the driver instead of complete facial expression. Apart from this we can add up more features which are related to driver's safety such as seat belt verification, security related purposes and vehicle maintenance system can be added to it. In future we can take many parameters and make our system very more accurate and provide safety measures to our drivers. We can also detect the mouth region of the driver and use it to calculate the Mouth Aspect Ratio (MAR) and use it to detect frequent yawning and alert the driver based on a threshold value of the mouth aspect ratio. This would add a new dimension to the project and help us detect drowsiness with a higher accuracy. We also can upgrade the cameras and use night vision cameras to detect accurately in low light conditions.

CHAPTER 10

CONCLUSION

We have measured the driver's safety parameters. Firstly, in drowsiness detection model we made an alert system which can alert the driver whenever he feels drowsy for more than 3-4 seconds he'll be alarmed and can stay awake or take a break. Whereas in the second model, face recognition provides security to our vehicle by detecting the driver's face and providing access. The drowsiness detection system can be implemented in every vehicle such that we can prevent road accidents and decrease the death ratio which are caused due to drowsiness and the face recognition system is very helpful to maintain the security of the vehicle preventing vehicle thefts.

As AI techniques are growing vastly, we can make systems more intelligent to understand the requirements of the hour. We can introduce various models and use different types of algorithms to get the best results. Road accidents are common in countries like India. Due to small negligence there's a huge loss to the lives of the human. By adapting such systems, we can try to control the road accidents and the security of the vehicle can be maintain by taking the alert and security systems into consideration.

CHAPTER 11

REFERENCES

[1] su.h and Zheng(2008)”A partial method of least squares regression-based totally fusion model for predicting the trend in drowsiness” su, h., & zheng, g. IEEE transactions on structures, man, and cybernetics-component a: structures and humans, 38(5), 1085- 1092.

[2]Friedrichs's and Zang(2010)”Digital camera-primarily based drowsiness reference for driving force kingdom classification underneath actual driving conditions” friedrichs, f., & yang, b. ieee wise automobiles”.

[3] Flores, m. J., Armengol, j. M., & de l. A. Escalera(2011)”Driver drowsiness detection gadget below infrared illumination for a wise vehicle". Flores, m. J., Armengol, j. M., & de l. A. Escalera, a. iet intelligent delivery systems, 5(four), 241-251".

[4] Mbouna, R. O., Kong, S. G., & Chun M.G.(2013) “Visual analysis of eye state and head pose for driver alertness monitoring” Mouna, R. O., Kong, S. G., & Chun, M. G. IEEE transactions on intelligent transportation systems, 14(3), 1462-1469.”.

[5] Tadesse, E., Sheng, W.,Liu.M(2014) “Driver drowsiness detection through HMM based dynamic modelling” Tadesse, E., Sheng, W., & Liu.M. In IEEE International conference on robotics and automation (ICRA) (pp. 4003-4008). IEEE”.