Driver Drowsiness Detection System

Design Project

By: YUVRAJ CHAURASIA

Dr Shakuntala Misra National Rehabilitation University, LKO

Submitted By

YUVRAJ CHAURASIA

B-TECH (CSE) 3rd YEAR

DSMNRU LUCKNOW

Submitted to

EIsystems , TECHNEX,IIT(BHU)

Administration Support by

SUMAN KR. MISHRA (Coordinator)

Dr Shakuntala Misra National Rehabilitation University, LKO

Acknowledgement

I would like to sincerely thank my coordinator

Mr. Suman kr. Mishra for giving his guidance , encouragement and readiness to answer to answer all the question go on show his willingness to help us in achieving my goals, I would also like to thank all the people who supported in completion of this project , for their valuable time and support during the project.

INDEX

DRIVER DROWSINESS DETECTION SYSTEM

Drowsy Driver Detection System has been developed using a non-intrusive machine vision based concepts. The system uses a small monochrome security camera that points directly towards the driver's face and monitors the driver's eyes in order to detect fatigue. In such a case when fatigue is detected, a warning signal is issued to alert the driver. This report describes how to find the eyes, and also how to determine if the eyes are open or closed. The algorithm developed is unique to any currently published papers, which was a primary objective of the project. The system deals with using information obtained for the binary version of the image to find the edges of the face, which narrows the area of where the eyes may exist. Once the face area is found, the eyes are found by computing the horizontal averages in the area. Taking into account the knowledge that eye regions in the present great intensity changes, the eyes are located by finding the significant intensity changes in the face.
Once the eyes are located, measuring the distances between the intensity changes in the eye area determine whether the eyes are open or closed. A large distance corresponds to eye closure. If the eyes are found closed for 5 consecutive frames,

the system draws the conclusion that the driver is falling asleep and issues a warning signal. The system is also able to detect when the eyes cannot be found, and works under reasonable lighting conditions.

INTRODUCTION:

Automotive population is increasing exponentially in our country. The Biggest problem regarding the increased use of vehicles is the rising number of road accidents. Road accidents are undoubtedly a global menace in our country. The frequency of road accidents in India is among the highest in the world. According to the reports of the National Crime Records Bureau (NCRB) about 135,000 road accidents-related deaths occur every year in India. The Global Status Report on Road Safety published by the World Health Organization (WHO) identified the major causes of road accidents are due errors and carelessness of the driver.

Driver sleepiness, alcoholism and carelessness are the key contributions in the accident scenario. The fatalities, associated expenses and related dangers have been recognized as serious threat to the country. All these factors led to the development of Intelligent Transportation Systems (ITS). ITS includes driver assistance systems like Adaptive Cruise Control, Park Assistance Systems, Pedestrian Detection Systems, Intelligent Headlights, Blind Spot Detection

Systems, etc. Taking into account of these factors, the driver's state is a major challenge for designing advanced driver assistance systems Driver errors and carelessness contribute most of the road accidents occurring nowadays. The major driver errors are caused by drowsiness, drunken and reckless behaviour of the driver. The resulted errors and mistakes contribute much loss to the humanity. In order to minimize the effects of driver abnormalities, a system for abnormality monitoring has to be inbuilt with the vehicle.

OBJECTIVE:

Drowsiness is a process where level of consciousness decrease due to lack of sleep or fatigue and can cause a person falls asleep. When driver is drowsy, the driver could lose control of the car so it was suddenly possible to deviate from the road and crashed into a barrier or a car.

Drowsiness detection techniques, in accordance with the parameters used for detection is divided into two sections i.e. intrusive method and a non-intrusive method. The main difference of these two methods is that the intrusive method.

An instrument connected to the driver and then the value of the instrument are recorded and checked. But intrusive approach has high accuracy, which is proportional to driver discomfort, so this method is rarely used.

PROPOSED SYSTEM:

There are several different algorithms and methods for eye tracking, and monitoring. Most of them in some way relate to features of the eye (typically reflections from the eye) within a video image of the driver. The original aim of this project was to use the retinal reflection as a means to finding the eyes on the face, and then using the absence of this reflection as a way of detecting when the eyes are closed. Applying this algorithm on consecutive video frames may aid in the calculation of eye closure period. Eye closure period for drowsy drivers are longer than normal blinking. It is also very little longer time could result in severe crash. So we will warn the driver as soon as closed eye is detected.

A. Sensing Phase

Eye Camera is used for sensing the eyes of the driver. Alcohol sensor is used for sensing the presence of alcohol content in the driver's breath. The accelerometer present on the vehicle suspension unit senses the downward acceleration of the vehicle toward the road humps and pits.

B. Detection Phase

The analysis of information from the sensors and camera are done to deduce the driver's current driving behaviour style. The open/ state of eyes is deduced by means of image processing techniques using computer vision. The image processing techniques are performed inside PC.

C. Correction Phase

This phase is responsible for doing the corrective actions required for that particular detected abnormal behaviour. The corrective actions include in-vehicle alarms, turning of the engine and GSM communication with the authorities. The corrective measures according to the behaviour detected. Corrections for drowsiness include in- vehicle alarms and its repetition turns the engine off. Drunken behaviour is rectified by in-vehicle alarms, if not GSM communication with the authorities are done. Reckless measures include in-vehicle alarms and repetition will turn off the engine Certain issues related to the low cost implementation of the proposed system with all its functionalities include the data fusion from different sensors and the image processing techniques. Also the addition of more sensors and algorithms to improve the accuracy and perfection of the system will be a challenge in front of this work.

D. System Process

(i) Eye detection function:

After inputting a facial image, pre-processing is first performed by binarizing the image. The top and sides of the face are detected to narrow down the area of where the eyes exist. Using the sides of the face, the centre of the face is found, which will be used as a reference when comparing the left and right eyes. Moving down from the top of the face, horizontal averages (average intensity value for each y coordinate) of the face area are calculated. Large changes in the averages are used to define the eye area. The following explains the eye detection procedure in the order of the processing operations. All images were generating in Mat lab using the image processing toolbox.

(ii) Binarization :

The first step to localize the eyes is binarizing the picture. Binarization is converting the image to a binary image. The background is uniformly black, and the face is primary white.

(iii) Removal of Noise:

The removal of noise in the binary image is very straightforward. The key to this is to stop at left and right edge of the face; otherwise the information of where the edges of the face are will be lost

The flowchart of the algorithm is represented in Figure 1

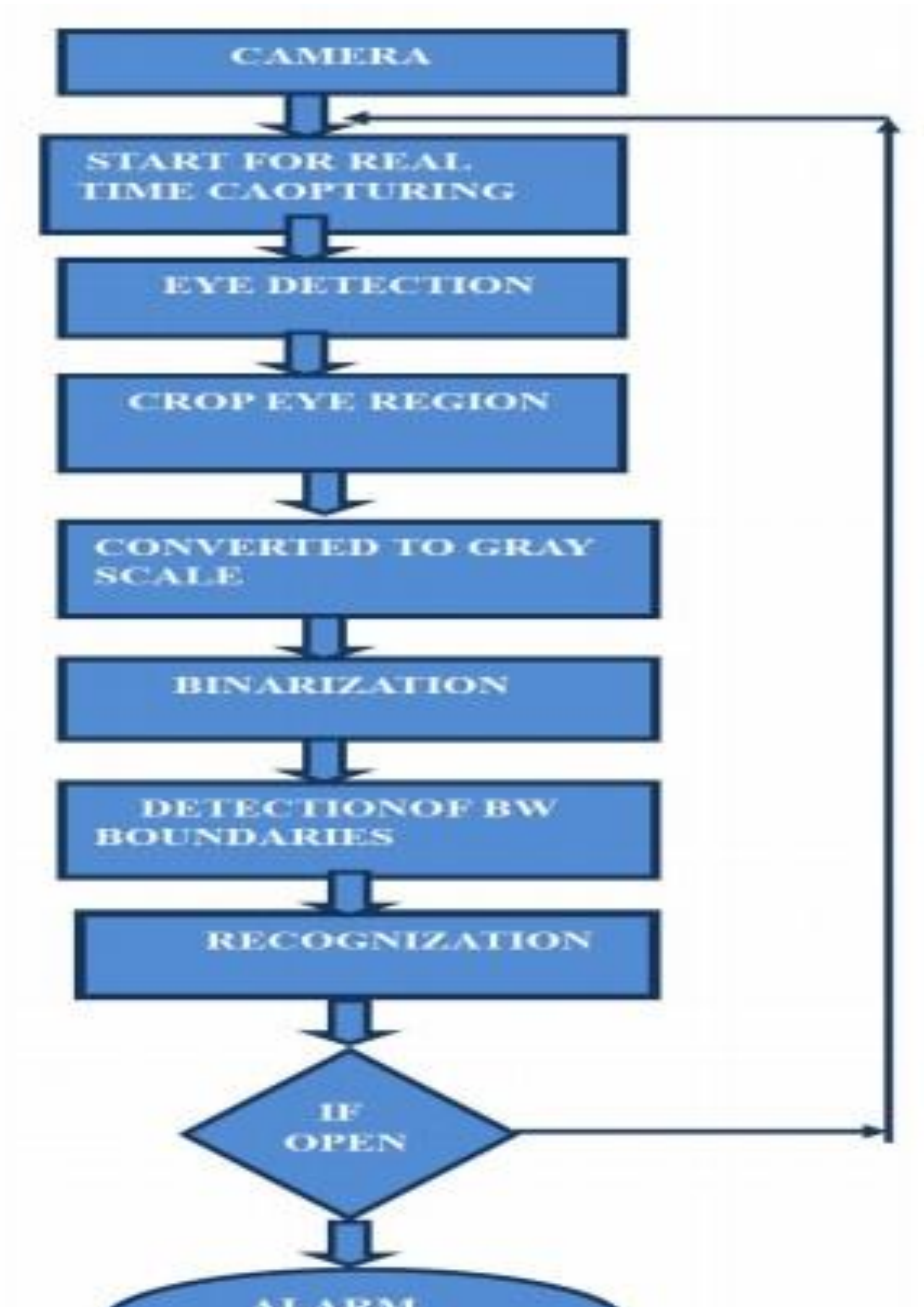


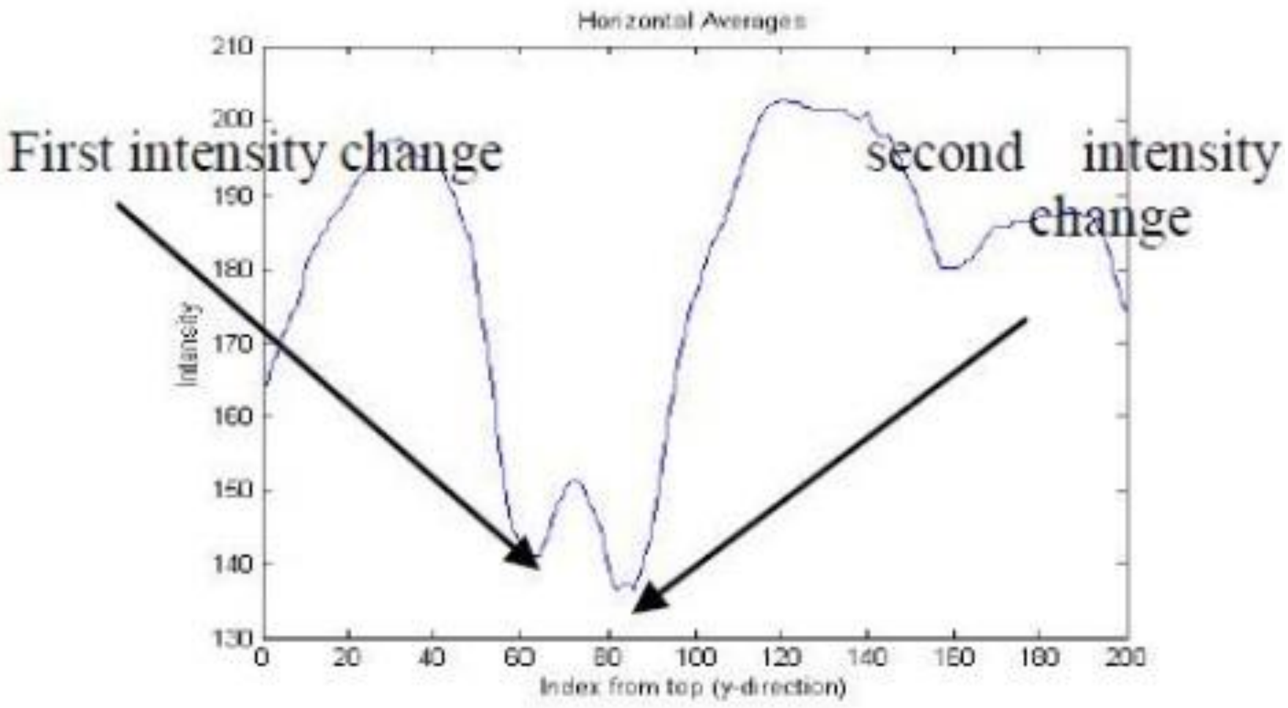
Fig .1.Diagram of the Proposed System

(iv) Finding Intensity Changes :

The next step in locating the eyes is finding the intensity changes on the face. This is done using the original image, not the binary image. The first step is to calculate the average intensity for each y – coordinate. This is called the horizontal average, since the averages are taken among the horizontal values. The valleys (dips) in the plot of the horizontal values indicate intensity changes. When the horizontal values were initially plotted, it was found that there were many small valleys, which do not represent intensity changes, but result from small differences in the averages. To correct this, a smoothing algorithm was implemented. The smoothing algorithm eliminated and small changes, resulting in a more smooth, clean graph. After obtaining the horizontal average data, the next step is to find the most significant valleys, which will indicate the eye area.

E. Detection of Vertical Eye Position

The first largest valley with the lowest y – coordinate is the eyebrow, and the second largest valley with the next lowest y-coordinate is the eye.



The areas of the left and right side are compared to check whether the eyes are found correctly. Calculating the left side means taking the averages from the left edge to the centre of the face, and similarly for the right side of the face. The reason for doing the two sides separately is because when the driver's head is tilted the horizontal averages are not accurate. For example if the head is tilted to the right, the horizontal average of the eyebrow area will be of the left eyebrow, and possibly the right hand side of the forehead.

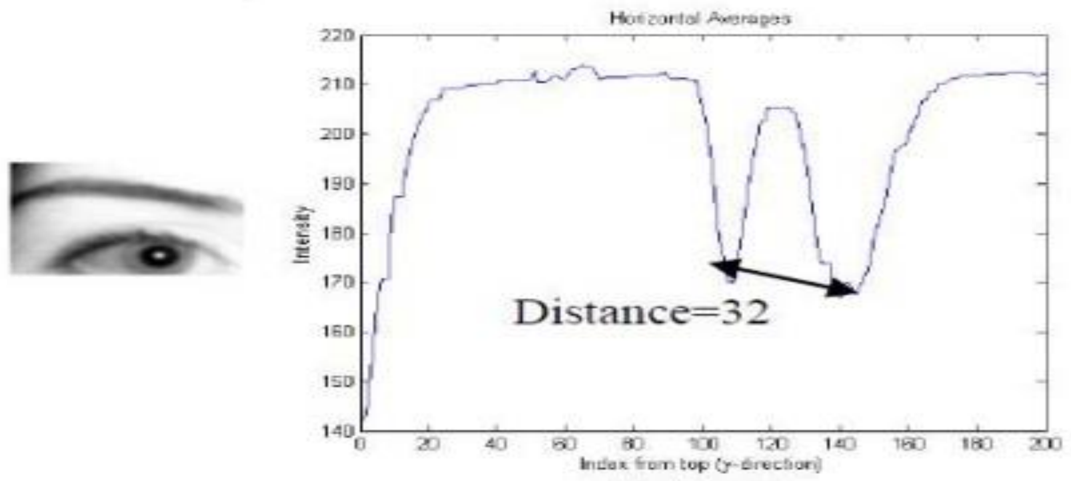
RESULT:

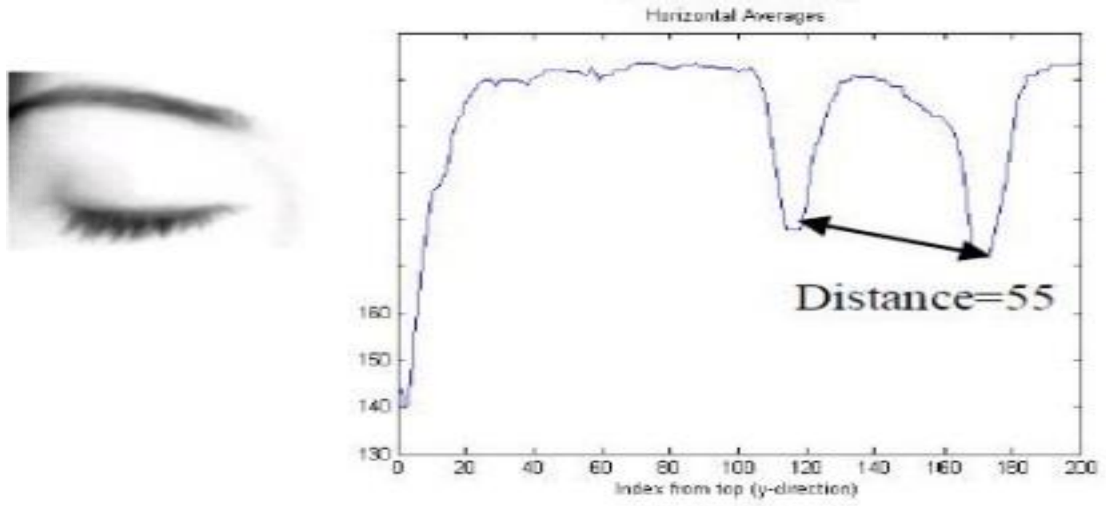
A. Drowsiness Detection Function

The state of the eyes (whether it is open or closed) is determined by distance between the first two intensity changes found in the above step. When the eyes are closed, the distance between the y – coordinates of the intensity changes is larger if compared to when the eyes are open. Fig. 3. Comparison between opened and closed eyes The limitation to this is if the driver moves their face closer to or further from the camera. If this occurs, the distances will vary, since the number of pixels the face takes up varies, as seen below. Because of this limitation, the system developed assumes that the driver’s face stays almost the same distance from the camera at all times

B. Judging Drowsiness

When there are 5 consecutive frames find the eye closed, then the alarm is activated, and a driver is alerted to wake up. Consecutive number of closed frames is needed to avoid including instances of eye closure due to blinking.





FUTURE WORK:

Fatigue is often ranked as a major factor in causing road crashes although its contribution to individual cases is hard to measure and is often not reported as a cause of crash. Driver fatigue is particularly dangerous because one of the symptoms is decreased ability to judge our own level of tiredness. Estimates suggest that fatigue is a factor in up to 30% of fatal crashes and 15%of serious injury crashes.

also contributes to approximately 25% of insurance losses in the heavy vehicle industry. The yawing and nodding of head can be implemented for detecting drowsiness for future work.

CONCLUSION:

The driver abnormality monitoring system developed is capable of detecting drowsiness, drunken and reckless behaviours of driver in a short time. The Drowsiness Detection System developed based on eye closure of the driver can differentiate normal eye blink and drowsiness and detect the drowsiness while driving. The proposed system can prevent the accidents due to the sleepiness while driving. The system works well even in case of drivers wearing spectacles and even under low light conditions if the camera delivers better output. Information about the head and eyes position is obtained through various self-developed image processing algorithms. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for too long, a warning is issued. processing judges the driver's alertness level on the basis continuous eye closures.

CODE INFORMATION

A. Software Required :

Python 3.6.8

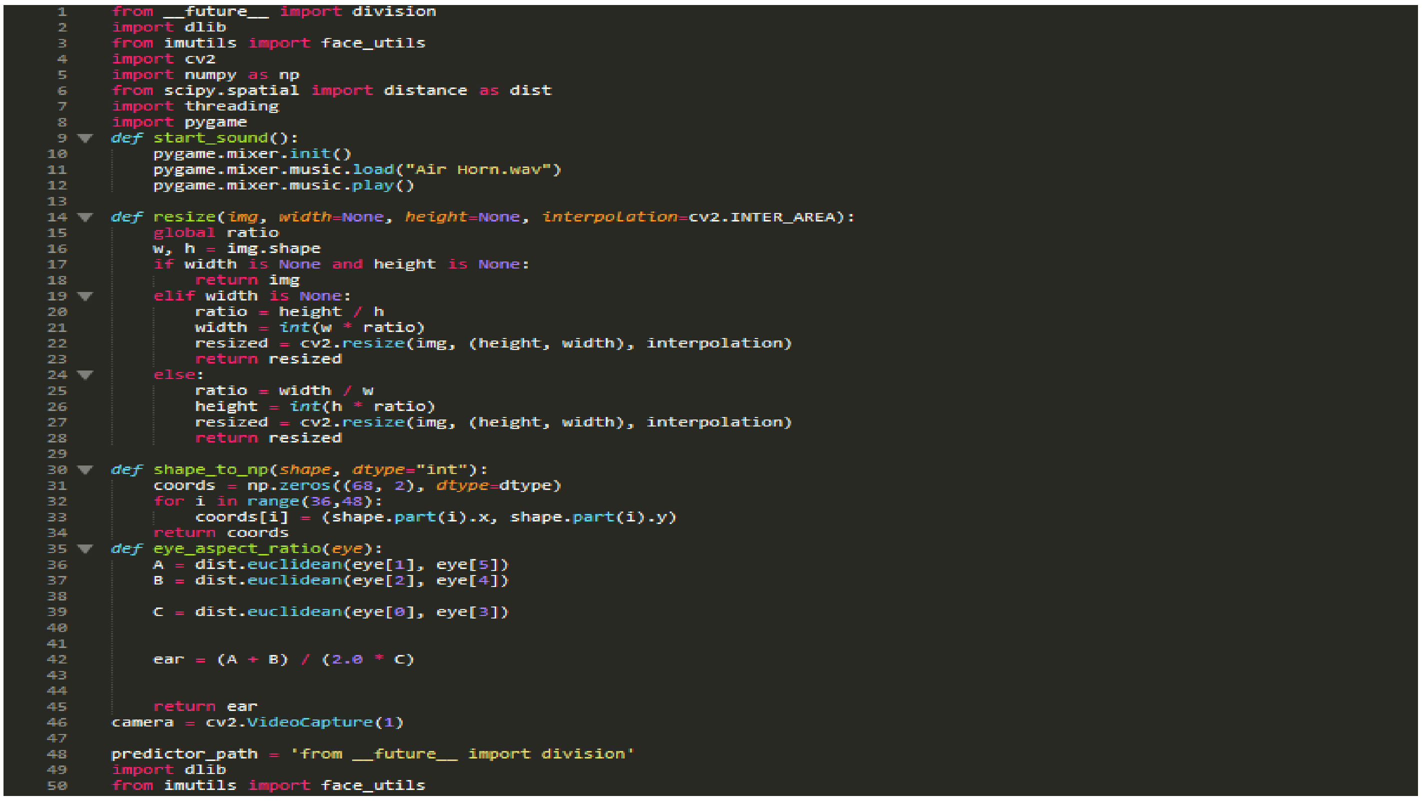
Anaconda (jupyter)

Files which is to be used in jupyter is downloaded from CMD

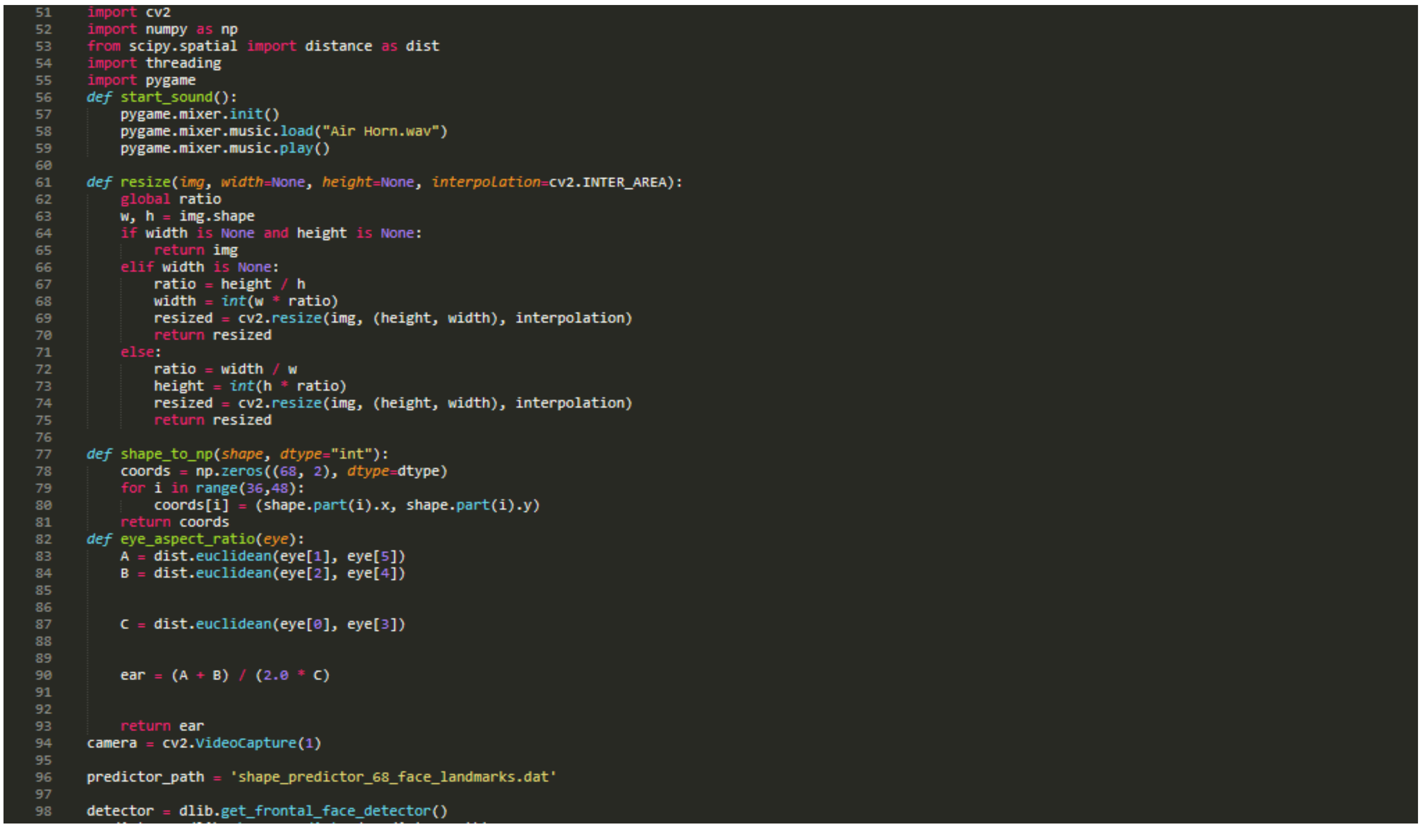
By “pip” command

Example : pip install pkg_name

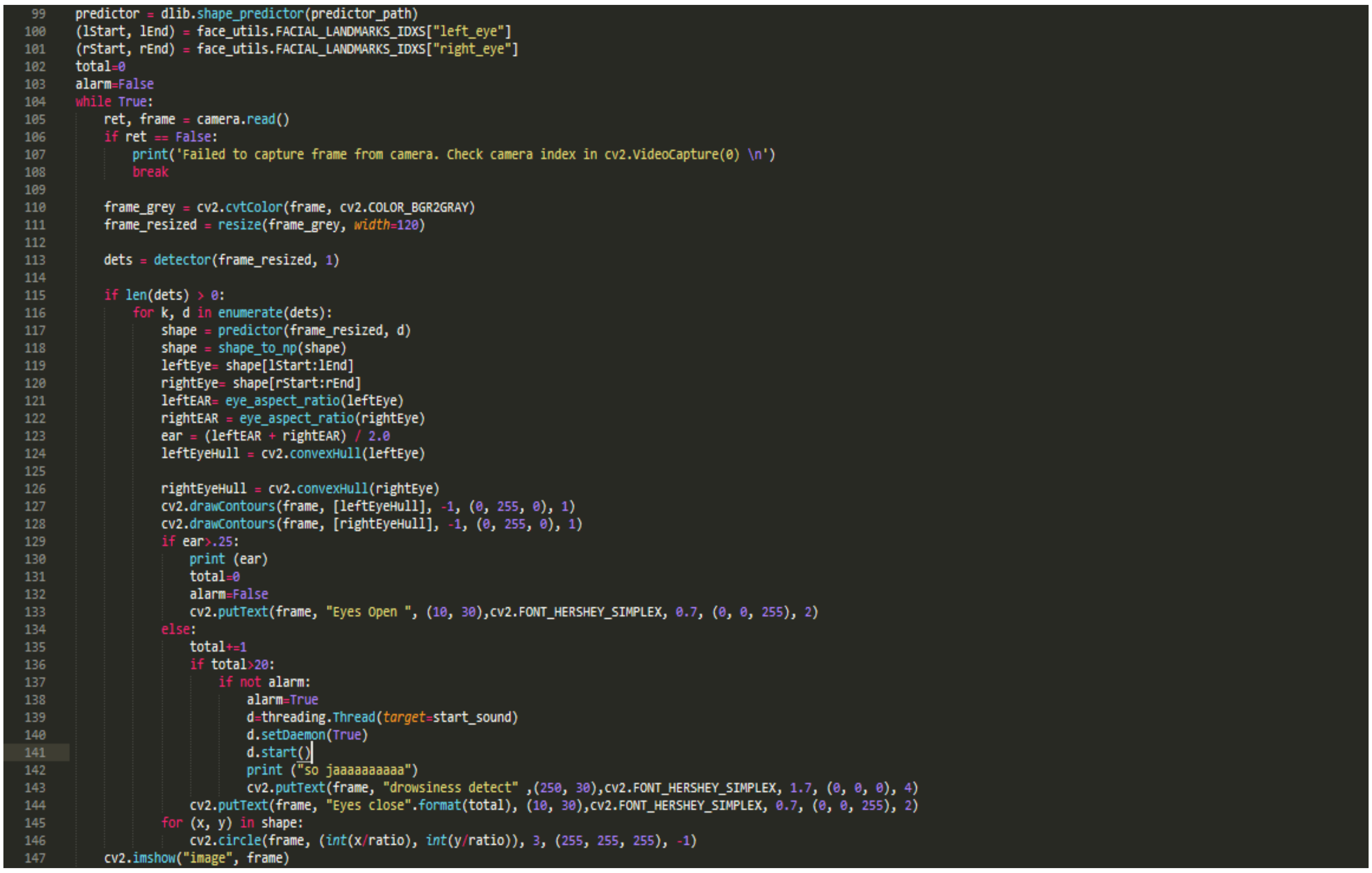
SOURCE CODE



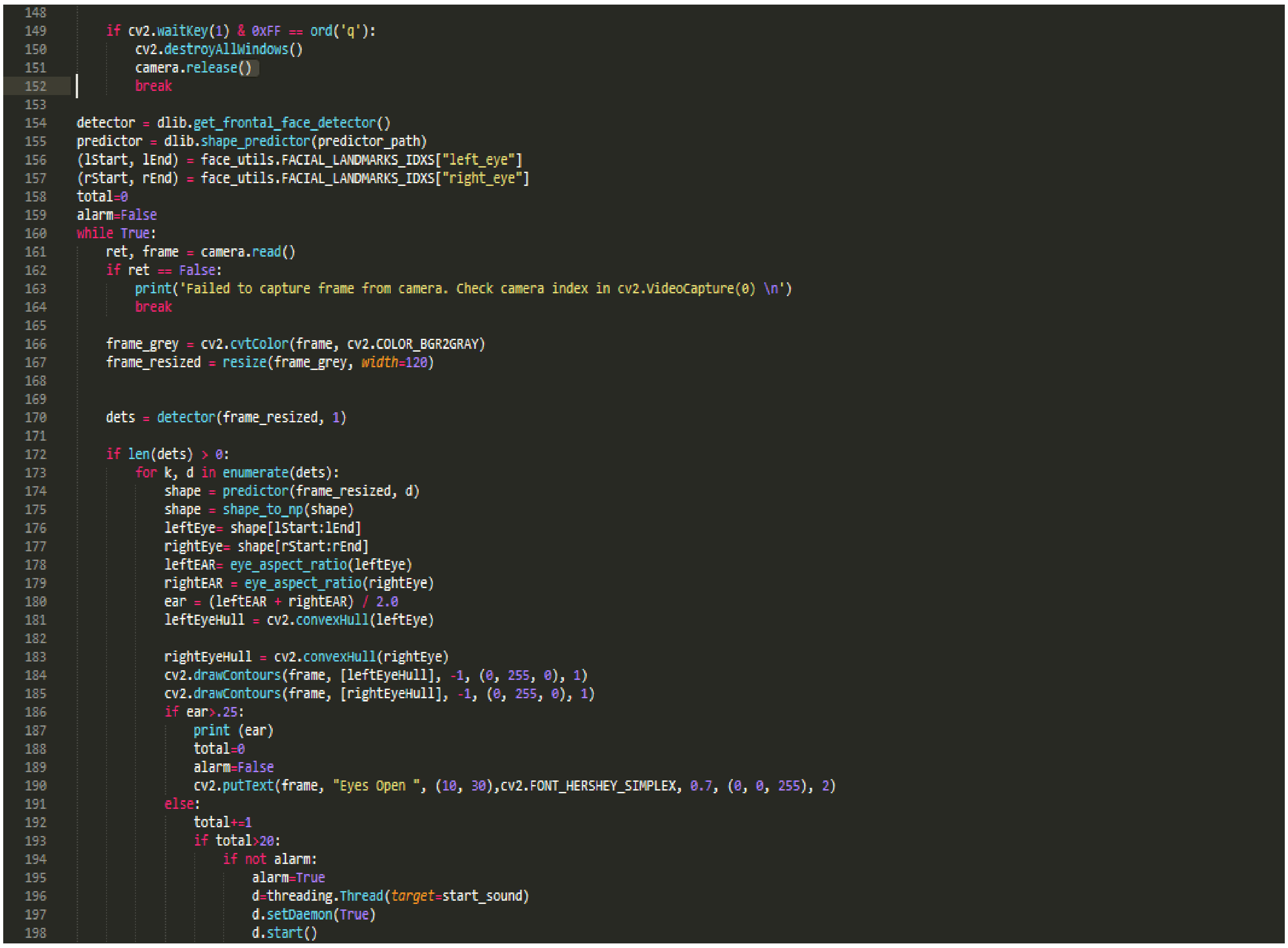
```python
from __future__ import division
import dlib
from imutils import face_utils
import cv2
import numpy as np
from scipy.spatial import distance as dist
import threading
import pygame
def start_sound():
    pygame.mixer.init()
    pygame.mixer.music.load("Air Horn.wav")
    pygame.mixer.music.play()

def resize(img, width=None, height=None, interpolation=cv2.INTER_AREA):
    global ratio
    w, h = img.shape
    if width is None and height is None:
        return img
    elif width is None:
        ratio = height / h
        width = int(w * ratio)
        resized = cv2.resize(img, (height, width), interpolation)
        return resized
    else:
        ratio = width / w
        height = int(h * ratio)
        resized = cv2.resize(img, (height, width), interpolation)
        return resized

def shape_to_np(shape, dtype="int"):
    coords = np.zeros((68, 2), dtype=dtype)
    for i in range(36,48):
        coords[i] = (shape.part(i).x, shape.part(i).y)
    return coords
def eye_aspect_ratio(eye):
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])

    C = dist.euclidean(eye[0], eye[3])


    ear = (A + B) / (2.0 * C)


    return ear
camera = cv2.VideoCapture(1)

predictor_path = 'from __future__ import division'
import dlib
from imutils import face_utils
```



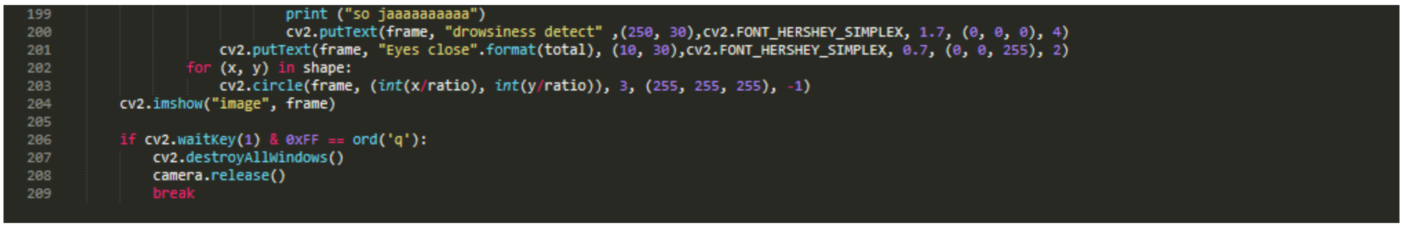
```python
import cv2
import numpy as np
from scipy.spatial import distance as dist
import threading
import pygame
def start_sound():
    pygame.mixer.init()
    pygame.mixer.music.load("Air Horn.wav")
    pygame.mixer.music.play()

def resize(img, width=None, height=None, interpolation=cv2.INTER_AREA):
    global ratio
    w, h = img.shape
    if width is None and height is None:
        return img
    elif width is None:
        ratio = height / h
        width = int(w * ratio)
        resized = cv2.resize(img, (height, width), interpolation)
        return resized
    else:
        ratio = width / w
        height = int(h * ratio)
        resized = cv2.resize(img, (height, width), interpolation)
        return resized

def shape_to_np(shape, dtype="int"):
    coords = np.zeros((68, 2), dtype=dtype)
    for i in range(36,48):
        coords[i] = (shape.part(i).x, shape.part(i).y)
    return coords
def eye_aspect_ratio(eye):
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])


    C = dist.euclidean(eye[0], eye[3])


    ear = (A + B) / (2.0 * C)


    return ear
camera = cv2.VideoCapture(1)

predictor_path = 'shape_predictor_68_face_landmarks.dat'

detector = dlib.get_frontal_face_detector()
```



```python
predictor = dlib.shape_predictor(predictor_path)
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
total=0
alarm=False
while True:
    ret, frame = camera.read()
    if ret == False:
        print('Failed to capture frame from camera. Check camera index in cv2.VideoCapture(0) \n')
        break

    frame_grey = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    frame_resized = resize(frame_grey, width=120)

    dets = detector(frame_resized, 1)

    if len(dets) > 0:
        for k, d in enumerate(dets):
            shape = predictor(frame_resized, d)
            shape = shape_to_np(shape)
            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>.25:
                print (ear)
                total=0
                alarm=False
                cv2.putText(frame, "Eyes Open ", (10, 30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            else:
                total+=1
                if total>20:
                    if not alarm:
                        alarm=True
                        d=threading.Thread(target=start_sound)
                        d.setDaemon(True)
                        d.start()
                        print ("so jaaaaaaaaa")
                        cv2.putText(frame, "drowsiness detect" ,(250, 30),cv2.FONT_HERSHEY_SIMPLEX, 1.7, (0, 0, 0), 4)
                cv2.putText(frame, "Eyes close".format(total), (10, 30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            for (x, y) in shape:
                cv2.circle(frame, (int(x/ratio), int(y/ratio)), 3, (255, 255, 255), -1)
    cv2.imshow("image", frame)
```



```python

    if cv2.waitKey(1) & 0xFF == ord('q'):
        cv2.destroyAllWindows()
        camera.release()
        break

detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(predictor_path)
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
total=0
alarm=False
while True:
    ret, frame = camera.read()
    if ret == False:
        print('Failed to capture frame from camera. Check camera index in cv2.VideoCapture(0) \n')
        break

    frame_grey = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    frame_resized = resize(frame_grey, width=120)


    dets = detector(frame_resized, 1)

    if len(dets) > 0:
        for k, d in enumerate(dets):
            shape = predictor(frame_resized, d)
            shape = shape_to_np(shape)
            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>.25:
                print (ear)
                total=0
                alarm=False
                cv2.putText(frame, "Eyes Open ", (10, 30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            else:
                total+=1
                if total>20:
                    if not alarm:
                        alarm=True
                        d=threading.Thread(target=start_sound)
                        d.setDaemon(True)
                        d.start()
```



```
199                         print ("so jaaaaaaaaaa")
200                         cv2.putText(frame, "drowsiness detect" ,(250, 30),cv2.FONT_HERSHEY_SIMPLEX, 1.7, (0, 0, 0), 4)
201                 cv2.putText(frame, "Eyes close".format(total), (10, 30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
202             for (x, y) in shape:
203                 cv2.circle(frame, (int(x/ratio), int(y/ratio)), 3, (255, 255, 255), -1)
204     cv2.imshow("image", frame)
205 
206     if cv2.waitKey(1) & 0xFF == ord('q'):
207         cv2.destroyAllWindows()
208         camera.release()
209         break
```

Link for dat file (shape_predictor_68_face_landmarks.dat) used in line no. 96 is given below download and paste it to your pwd Location.

https://raw.githubusercontent.com/davisking/dlib-models/master/shape_predictor_68_face_landmarks.dat.bz2

To download the source code visit link given below

https://drive.google.com/file/d/1oba9bfmglpga3V7WLGhaatqsU55URyln/view

References

https://www.seminarsonly.com/Engineering-Projects/Computer/driver-drowsiness-detection-system.php

https://www.codeproject.com/Articles/26897/TrackEy

https://github.com/raja434/driver-fatigue-detection-system