A

Project Report

On

"DRIVER DROWSINESS DETECTION USING MACHINE LEARNING"

Submitted in partial fulfillment of the requirements of
the degree of

Bachelor of Technology

In

Computer Engineering

Affiliated to

Dr.Babasaheb Ambedkar Technological University, Lonere

Submitted by

1.	**SUTAR JAGADISH NARENDRA**	**51631320181124510001**
2	**GHATAGE AMIT MANSING**	**1963131245003**
3	**BARDOL SHABANA MOULA**	**1963131245019**
4	**PATIL VEDIKA ADHIKRAO**	**1963131245023**
5	**KOCHARE MAYURI DATTATRAY**	**1963131245025**

Under the Guidance of

Prof. Mr. Kadam S. R.

Department Of Computer Engineering

Jaywant College of Engineering and Polytechnic,

KM.Gad

Shetkari Shikshan Prasarak Mandal's,

Jaywant College of Engineering and Polytechnic, KM.Gad

Academic Year 2021-22

CERTIFICATE

This is to certify that the Dissertation entitled **"DRIVER DROWSINESS DETECTION USING MACHINE LEARNING"** submitted by,

1	**SUTAR JAGADISH NARENDRA**	**51631320181124510001**
2	**GHATAGE AMIT MANSING**	**1963131245003**
3	**BARDOL SHABANA MOULA**	**1963131245019**
4	**PATIL VEDIKA ADHIKRAO**	**1963131245023**
5	**KOCHARE MAYURI DATTATRAY**	**1963131245025**

is a bonafide record of the work carried out by him towards the partial fulfillment of the requirements of **Dr.Babasaheb Ambedkar Technological University, Lonere**. for the award of the degree of **Bachelor of Technology (B.Tech)**, in Academic Year 2021-22.

Guide Name **Prof. Mr. Kadam S. R**	**Prof. Mr. Kadam S. R** (HoD) **Department of CSE**	**Dr. U. S. Sutar** (Principal) **JCEP, KM**

APPROVAL SHEET

The Dissertation here to entitled **"Driver Drowsiness Detection Using Machine Learning"** submitted by –

1.	**SUTAR JAGADISH NARENDRA**	**51631320181124510001**
2	**GHATAGE AMIT MANSING**	**1963131245003**
3	**BARDOL SHABANA MOULA**	**1963131245019**
4	**PATIL VEDIKA ADHIKRAO**	**1963131245023**
5	**KOCHARE MAYURI DATTATRAY**	**1963131245025**

in partial fulfillment of the requirements for the degree of **Bachelor of Technology (B.Tech)**, Computer Science and Engineering has been examined and is recommended for acceptance and approval.

Internal Examiner

External Examiner

Date:

Place: JCEP, KM Gad

DECLARATION

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources.

I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission.

I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1.	**SUTAR JAGADISH NARENDRA**	**51631320181124510001**
2.	**GHATAGE AMIT MANSING**	**1963131245003**
3.	**BARDOL SHABANA MOULA**	**1963131245019**
4.	**PATIL VEDIKA ADHIKRAO**	**1963131245023**
5.	**KOCHARE MAYURI DATTATRAY**	**1963131245025**

ACKNOWLEDGEMENT

It gives us a great pleasure to remain deeply indebted to our project guide **Prof. Mr. Kadam S. R** whom we have privilege to work. The faith & confidence shown by his in us boost our moral and motivate us to perform better in this project.

We convey our sincere thanks to **Prof. Mr. Kadam S. R** (H.O.D., Computer Science and Engineering Department) for his timely co-operation. Finally, thanks to all **Staff Members** and all friends and colleagues who support us in the development of project.

1.	**SUTAR JAGADISH NARENDRA**	**51631320181124510001**
2.	**GHATAGE AMIT MANSING**	**1963131245003**
3.	**BARDOL SHABANA MOULA**	**1963131245019**
4.	**PATIL VEDIKA ADHIKRAO**	**1963131245023**
5.	**KOCHARE MAYURI DATTATRAY**	**1963131245025**

ABSTRACT

Drowsiness of the drivers is the principal cause of injuries in the world. Because of loss of sleep and tiredness, drowsiness can occur even as riding. The first-rate manner to keep driving force and warn him before fall into sleep away from accidents because of drivers' drowsiness is to come across drowsiness of the driver.

To discover drowsiness many techniques like eye retina detection, facial function recognition has been used. We suggest a way of detecting motive force drowsiness, the usage of eye retina detection and pulse charge detection of the driving force. On this record, we endorse an extra accurate drowsiness detection approach which is a hybrid technique of eye retina detection and pulse sample detection.

Drivers often feel very drowsy while continuously driving for long distances without taking breaks. Drowsiness is a major factor in increasing the chances for a vehicle to meet accidents. According to various studies, Number of accidents caused by drowsiness is much higher than the number of accidents caused by drunk driving. The number of accidents caused by drowsiness can be reduced by having a proper system that can detect drowsiness, alert the driver and prevent major injuries. It needs a proper system that will alert drivers to prevent major injuries. The project starts an alarm when it detects the driver is drowsy and notify drivers to take a rest along with a nearby rest services area. It detects the driver's last sleeping time and suggests to take a rest.

INDEX

Sr.No	Title	Page No
1.	Introduction	11
2.	Review of Literature	13
3.	Proposed Methodology	15
	3.1 The different types of methodologies have been developed to find out drowsiness.	15
	3.2 The various technology that can be used are discussed	15
	3.3 System Description	16
4.	Modeling Diagrams	24
5.	Results and Discussion	28
6.	Conclusion	36
7.	References	38

List of Figures

Fig.01	Flow Chart And Algorithm.
Fig.02	Five Haar Like Feature.
Fig.03	Example Of Represented Haar.
Fig.04.	Virtualizing the 68 landmarks co-ordinates.
Fig.05	Detection Of Both Eyes.
Fig.06	Open and closed eyes with landmarks p(i) automatically detected.
Fig.07	Plotted for several frames of a video sequence.
Fig.08	Drowsiness State.
Fig.09	Console Information.
Fig.10	Flow Chart Of system.
Fig.11	DFD for Level 0.
Fig.12	DFD for Level 1.
Fig.13	DFD for Level 2.
Fig.14	Sequence Diagram.
Fig.15	Output 1
Fig.16	Output 2
Fig.17	Output 3

Chapter 1

INTRODUCTION

INTRODUCTION

The increasing number of traffic accidents due to a driver's diminished vigilance level has become a serious problem for society. Some of these accidents are the result of the driver's medical condition. However, a majority of these accidents are related to driver fatigue, drowsiness of drivers. Car accidents associated with driver fatigue are more likely to be serious, leading to serious injuries and deaths. Fletcher et al. in has mentioned that 30% of all traffic accidents have been caused by drowsiness. It was demonstrated that driving performance deteriorates with increased drowsiness with resulting crashes constituting more than 20% of all vehicle accidents Traditionally transportation systems are no longer sufficient. One can use a number of different techniques for analyzing driver's drowsiness.

These techniques are Image Processing based techniques, convolutional neural network based techniques. And image processing based techniques can be divided in three categories. These categories are template matching technique, eye blinking technique, yawning based technique. These techniques are based on computer vision using image processing. In the computer vision technique, facial expressions of the driver like eyes blinking and head movements are generally used by the researchers to detect driver drowsiness.

Chapter 2

REVIEW OF LITERATURE

REVIEW OF LITERATURE

In this section, we have discussed various methodologies that have been proposed by researchers for drowsiness detection and blink detection during the recent years. Manu B.N in 2016, has proposed a method that detect the face using Haar feature-based cascade classifiers. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier that will detect the object. So along with the Haar feature-based classifiers, cascaded Ada boost classifier is exploited to recognize the face region then the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image.

Due to the difference of facial feature, Haar-like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of pixel values within rectangle area and during the process the Ada boost algorithm will allow all the face samples and it will discard the non-face samples of images.

Amna Rahman in 2015, has proposed a method to detect the drowsiness by using Eye state detection with Eye blinking strategy. In this method first, the image is converted to gray scale and the corners are detected using Harris corner detection algorithm which will detect the corner at both side and at down curve of eye lid. After tracing the points then it will make a straight line between the upper two points and locates the mid-point by calculation of the line, and it connects the mid-point with the lower point.

Now for each image it will perform the same procedure and it calculates the distance ‘d’ from the mid-point to the lower point to determine the eye state. Finally, the decision forthe eye state is made based on distance ’d’ calculated. If the distance is zero or is close to zero, the eye state is classified as “closed” otherwise the eye state is identified as “open”. They have also invoked intervals or time to know that the person is feeling drowsy or not. This is done by the average blink duration of a person is 100-400milliseconds (i.e. 0.1-0.4 of a second).

Chapter 3

PROPOSED METHODOLOGY

PROPOSED METHODOLOGY

- **3.1. The different types of methodologies have been developed to find out drowsiness.**

3.1.1. Physiological level approach:

This technique is an intrusive method where in electrodes are used to obtain pulse rate, heart rate and brain activity information. ECG is used to calculate the variations in heart rate and detect different conditions for drowsiness. The correlation between different signals such as ECG (electrocardiogram), EEG(electroencephalogram), and EMG (electromyogram) are made and then the output is generated whether the person is drowsy or not.

3.1.2. Behavioral based approach:

In this technique eye blinking frequency, head pose, etc. of a person is monitored through a camera and the person is alerted if any of these drowsiness symptoms are detected.

- **3.2. The various technology that can be used are discussed as:**

3.2.1. TensorFlow:

IT is an open-source software library f or dataflow programming across arrange of tasks. It is a symbolic math library, and is also used for machine learning ap plications such as neural networks. It is used for both research and production. TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays. These arrays are referred to as "tensors".

3.2.2. Machine learning:

Machine learning is the kind of programming which gives computers the capability to automatically learn from data without being explicitly programmed. This means in other words that these programs change their behavior by learning from data. Python is clearly one of the best languages for machine learning. Python does contain special libraries for machine learning namely scipy, pandas 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.

3.2.3. OpenCV:

OpenCV stands for Open Source Computer Vision. It's an Open Source BSD licensed library that includes hundreds of advanced Computer Vision algorithms that are optimized to use hardware acceleration. OpenCV is commonly used for machine learning, image processing, image manipulation, and much more. OpenCV has a modular structure. There are shared and static libraries and a CV Namespace. In short, OpenCV is used in our application to easily load bitmap files that contain landscaping pictures and perform a blend operation between two pictures so that one picture can be seen in the background of another picture. This image manipulation is easily performed in a few lines of code using OpenCV versus other methods. OpenCV.org is a must if you want to explore and dive deeper into image processing and machine learning in general.

➢ 3.3. System Description:

3.3.1. Flow Chart And Algorithm:

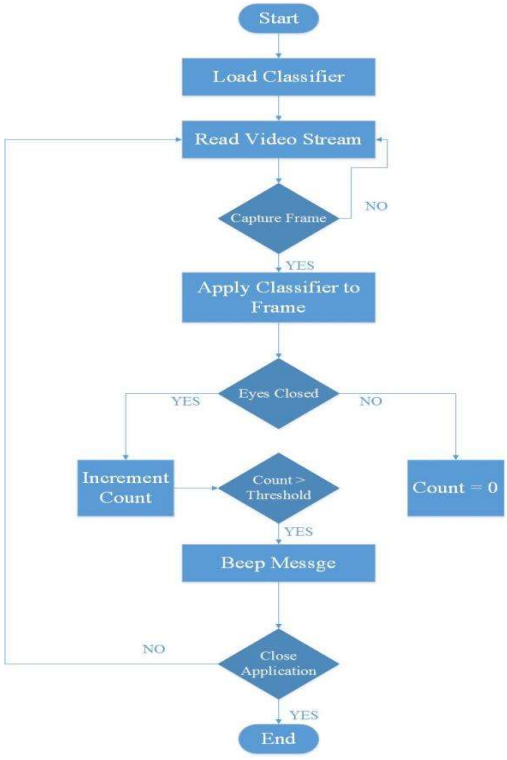

Fig 3.1: The various detection stages are discussed

3.3.2. Face Detection:

For the face Detection it uses Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images(images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle. Fig. 3.2 represents five haar like features & example is shown in Fig.3.3

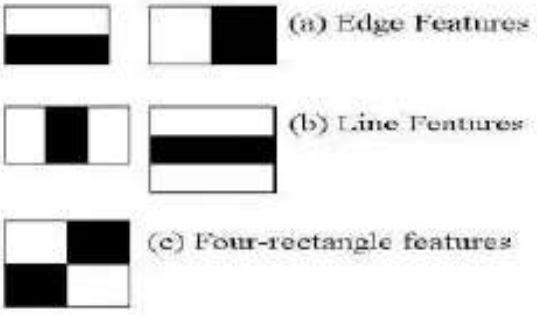

Fig 3.2: Represents Five Haar Features

Fig 3.3: Example

A cascaded Ada boost classifier with the Haar-like features is exploited to find out the face region. First, the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image. Due to the difference of facial feature, Haar- like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of pixel values within rectangle areas. The features can be represented by the different composition of the black region and white region.

A cascaded Ada boost classifier is a strong classifier which is a combination of several weak classifiers. Each weak classifier is trained by Ada boost algorithm. If a candidate sample passes through the cascaded Ada boost classifier, the face region can be found. Almost all of face samples can pass through and non face samples can be rejected.

3.3.3. Eye Detection:

In the system we have used facial landmark prediction for eye detection Facial landmarks are used to localize and represent salient regions of the face, such as:

- Eyes
- Eyebrows
- Nose
- Mouth

Jawline Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. In the context of facial landmarks, our goal is detecting important facial structures on the face using shape prediction methods. Detecting facial landmarks is therefore a one step process:

- Localize the face in the image.

Detect the key facial structures on the face ROI. Localize the face in the image: The face image is localized by Haar feature-based cascade classifiers which was discussed in the first step of our algorithm i.e. face detection .Detect the key facial structures on the face ROI: There are a variety of facial landmark detectors, but all methods essentially try to localize and label the following facial regions:

- Mouth
- Right eyebrow
- Left eyebrow
- Right eye
- Left eye
- Nose

The facial landmark detector included in the dlib library is an implementation of the One Millisecond Face Alignment with an Ensemble of Regression Trees paper by Kazemi and Sullivan (2014).This method starts by using:

1. A training set of labeled facial landmarks on an image. These images are manually labeled, specifying specific (x, y)-coordinates of regions surrounding each facialstructure.2.
2. Priors, of more specifically, the probability on distance between pairs of input pixels. The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face. The indexes of the 68 coordinates can be visualized on the image below:

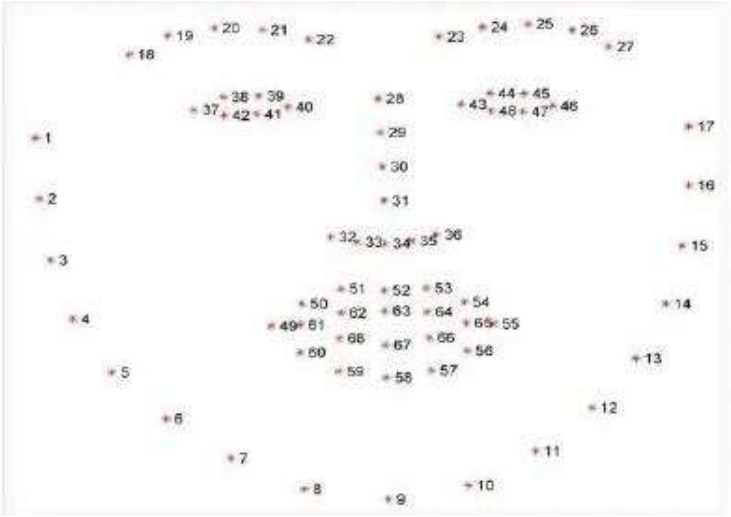

Fig 3.4: Virtualizing the 68 landmarks co-ordinates

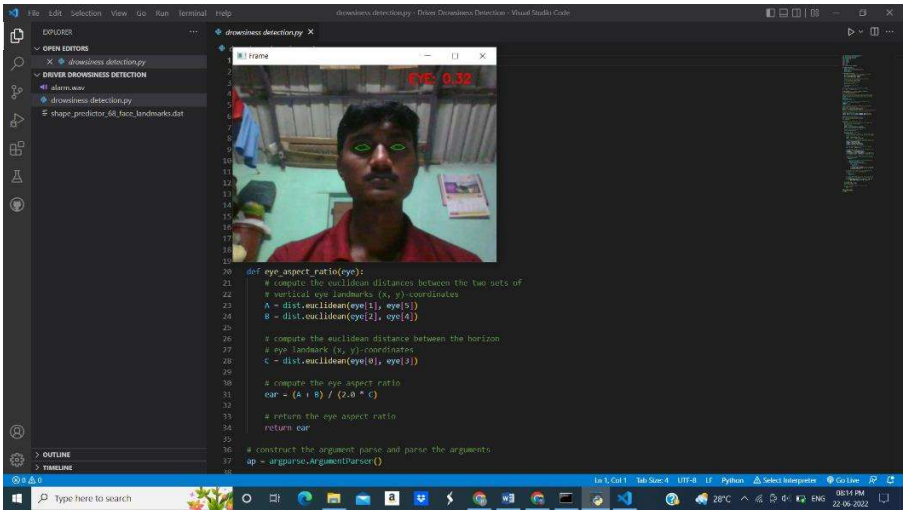

Fig 3.5: Detection Of Both Eyes

We can detect and access both the eye region by the following facial landmark index show below

- The right eye using [36, 42].
- The left eye with [42, 48].

These annotations are part of the 68 point iBUG 300-W dataset which the dlib facial landmark predictor was trained on. It's important to note that other flavors of facial landmark detectors exist, including the 194 point model that can be trained on the HELEN dataset. Regardless of which dataset is used, the same dlib framework can be leveraged to train a shape predictor on the input training data.

3.3.4. Recognition of Eye's State:

The eye area can be estimated from optical flow, by sparse tracking or by frame-to-frame intensity differencing and adaptive thresholding. And Finally, a decision is made whether theeyes are or are not covered by eyelids. A different approach is to infer the state of the eye opening from a single image, as e.g. by correlation matching with open and closed eye templates, a heuristic horizontal or vertical image intensity projection over the eye region, a parametric model fitting

to find the eyelids, or active shape models. A major drawback of the previous approaches is that they usually implicitly impose too strong requirements on the setup, in the sense of a relative face-camera pose (head orientation), image resolution, illumination, motion dynamics, etc. Especially the heuristic methods that use raw image intensity are likely to be very sensitive despite their real-time performance.

Therefore, we propose a simple but efficient algorithm to detect eye blinks by using a recent facial landmark detector. A single scalar quantity that reflects a level of the eye opening i s derived from the landmarks. Finally, having a per-frame sequence of the eye-opening estimates, the eye blinks are found by an SVM classifier that is trained on examples of blinking and non- blinking patterns.

3.3.5. Eye Aspect Ratio Calculation:

For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is

$$\text{Computed EYE} = \|p2 - p6\| + \|p3 - p5\| \ (1)2\|p1 - p4\|$$

where p1, . . ., p6 are the 2D landmark locations, depicted in Fig. 3.6. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye. It is partially person and head pose insensitive. Aspect ratio of the open eye has a small variance among individuals, and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged.

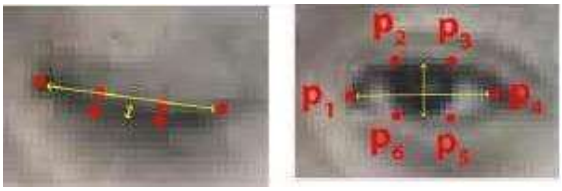

Fig 3.6: Open and closed eyes with landmarks

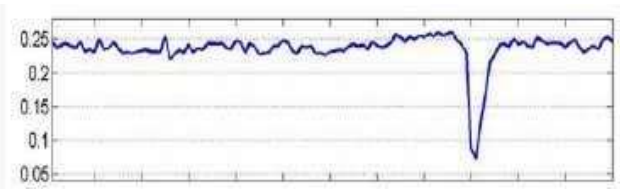

Fig 3.7: Plotted Graph For Closed Eyes

3.3.6. Eye State Determination:

Finally, the decision for the eye state is made based on EAR calculated in the previous step. If the distance is zero or is close to zero, the eye state is classified as “closed” otherwise the eye

state is identified as “open”.

3.3.7. Drowsiness Detection:

The last step of the algorithm is to determine the person’s condition based on a preset condition for drowsiness. The average blink duration of a person is 100-400 milliseconds (i.e. 0.1-0.4 of a second).

Hence if a person is drowsy his eye closure must be beyond this interval. We set at time frame of 5 seconds. If the eyes remain closed for five or more seconds, drowsiness is detected and alert pop regarding this is triggered.

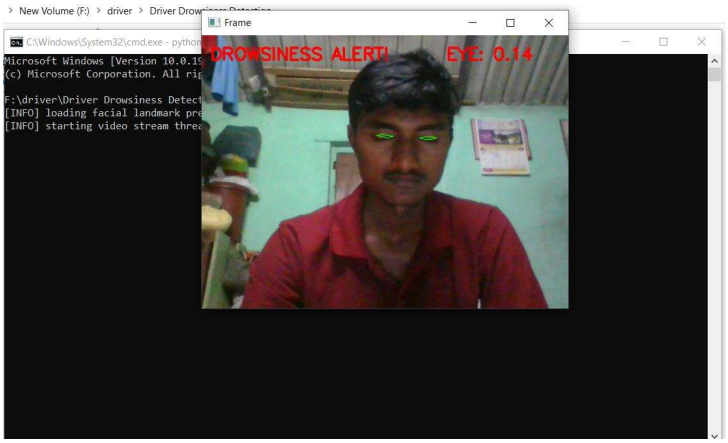

Fig 3.8 Drowsiness State

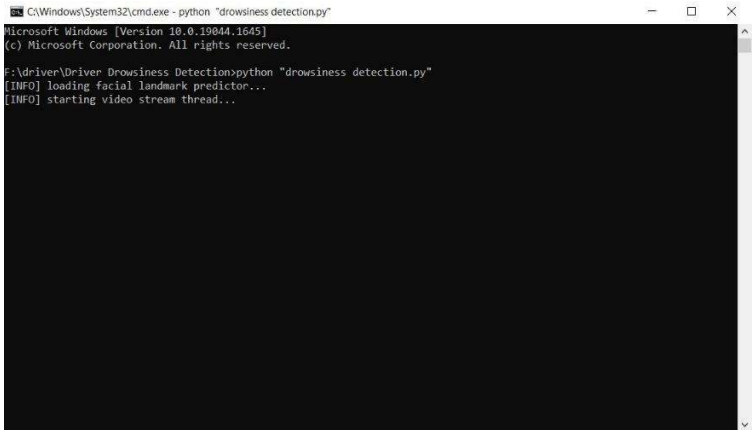

Fig 3.9 Console Information

Chapter 4

MODELLING DIAGRAM

MODELLING DIAGRAM

4.1. Project Work Flow:

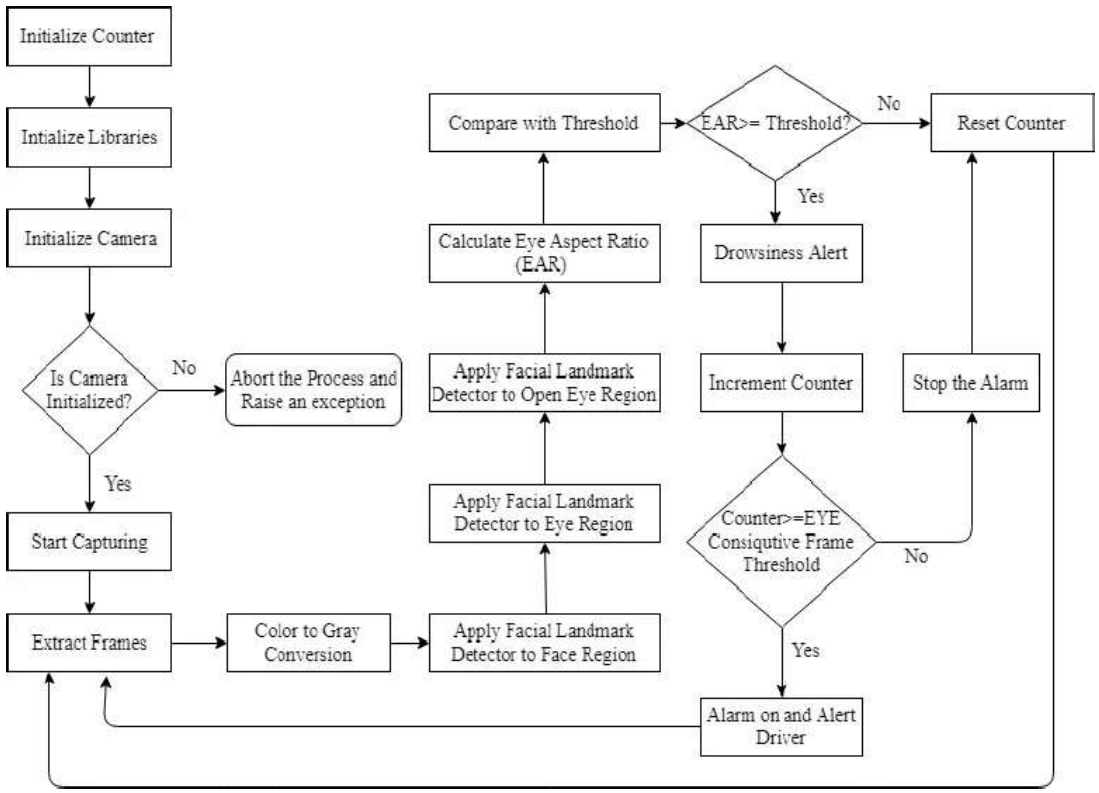

Fig 4.1: Flow Chart Of System

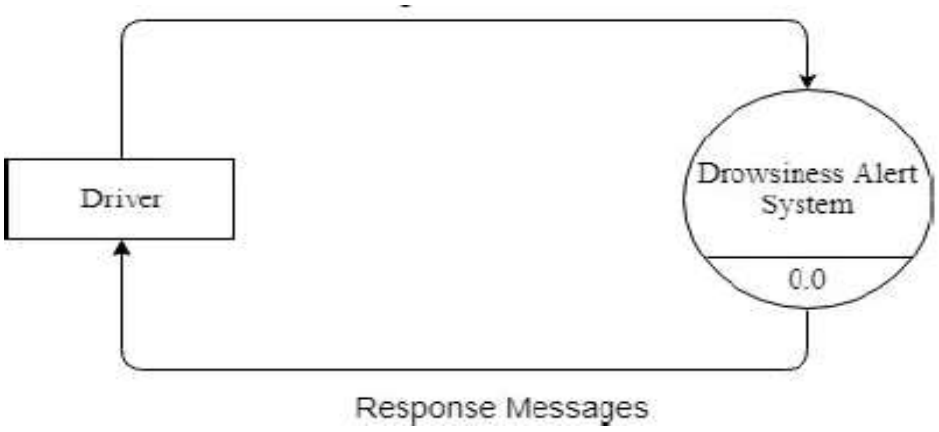

Fig 4.2 DFD Level 0

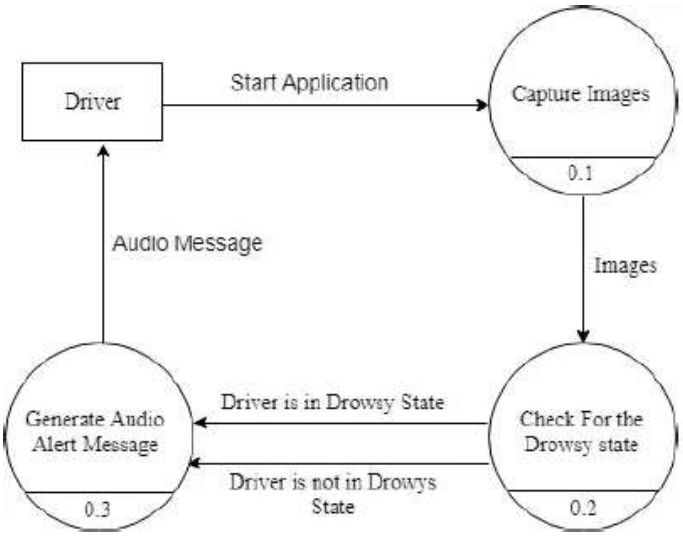

Fig 4.3: DFD Level 1

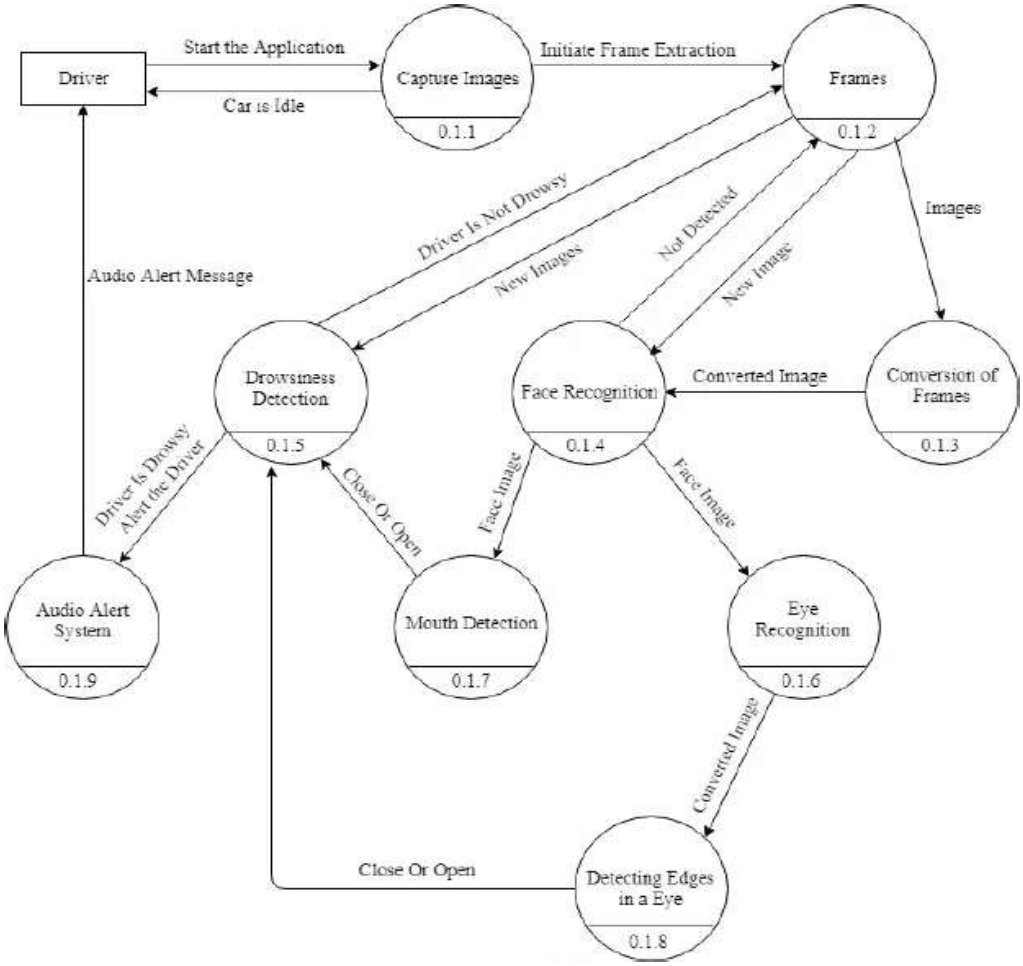

Fig 4.4: DFD Level 2

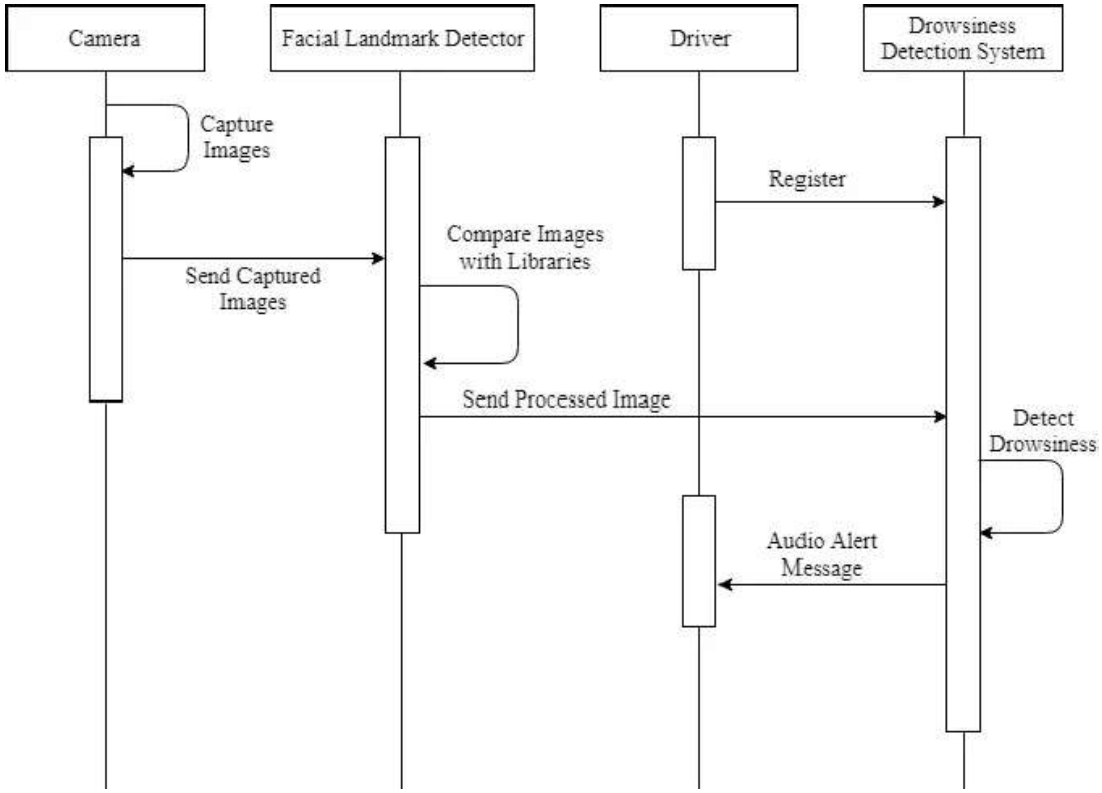

Fig 4.5: Sequence Diagram

Chapter 5

RESULTS AND DISCUSSIONS

RESULTS AND DISCUSSIONS

Implementation of drowsiness detection with Python and OpenCV was done which includes the following steps: Successful runtime capturing of video with camera. Captured video was divided into frames and each frame were analyzed. Successful detection of face followed by detection of eye.

If closure of eye for successive frames were detected, then it is classified as drowsy condition else it is regarded as normal blink and the loop of capturing image and analyzing the state of driver is carried out again and again. In this implementation during the drowsy state the eye is not surrounded by circle or it is not detected, and corresponding message is shown.

5.1. Code :

Drowsiness detection.py

```
from scipy.spatial import distance as dist
from imutils.video import VideoStream
from imutils import face_utils
from threading import Thread
import numpy as np
import pyglet
import argparse
import imutils
import time
import dlib
import cv2
from playsound import playsound

def sound_alarm(path):
    # play an alarm sound
    music = pyglet.resource.media('alarm.wav')
    music.play()
```

```
    pyglet.app.run()

def eye_aspect_ratio(eye):
    # compute the euclidean distances between the two sets of
    # vertical eye landmarks (x, y)-coordinates
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])

    # compute the euclidean distance between the horizon
    # eye landmark (x, y)-coordinates
    C = dist.euclidean(eye[0], eye[3])

    # compute the eye aspect ratio
    ear = (A + B) / (2.0 * C)

 # return the eye aspect ratio
return ear

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()

ap.add_argument("-w", "--webcam", type=int, default=0,
    help="index of webcam on system")
args = vars(ap.parse_args())

# define two constants, one for the eye aspect ratio to indicate
# blink and then a second constant for the number of consecutive
# frames the eye must be below the threshold for to set off the
# alarm
EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 45
```

```
# initialize the frame counter as well as a boolean used to
# indicate if the alarm is going off
COUNTER = 0
ALARM_ON = False

# initialize dlib's face detector (HOG-based) and then create
# the facial landmark predictor
print("[INFO] loading facial landmark predictor...")
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

# grab the indexes of the facial landmarks for the left and
# right eye, respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

# start the video stream thread
print("[INFO] starting video stream thread...")
vs = VideoStream(src=args["webcam"]).start()
time.sleep(1.0)

# loop over frames from the video stream
while True:
    # grab the frame from the threaded video file stream, resize
    # it, and convert it to grayscale
    # channels)
    frame = vs.read()
    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
# detect faces in the grayscale frame
rects = detector(gray, 0)

# loop over the face detections
for rect in rects:
    # determine the facial landmarks for the face region, then
    # convert the facial landmark (x, y)-coordinates to a NumPy
    # array
    shape = predictor(gray, rect)
    shape = face_utils.shape_to_np(shape)

    # extract the left and right eye coordinates, then use the
    # coordinates to compute the eye aspect ratio for both eyes
    leftEye = shape[lStart:lEnd]
    rightEye = shape[rStart:rEnd]
    leftEAR = eye_aspect_ratio(leftEye)
    rightEAR = eye_aspect_ratio(rightEye)

    # average the eye aspect ratio together for both eyes
    ear = (leftEAR + rightEAR) / 2.0

    # compute the convex hull for the left and right eye, then
    # visualize each of the eyes
    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)

    # check to see if the eye aspect ratio is below the blink
    # threshold, and if so, increment the blink frame counter
    if ear < EYE_AR_THRESH:
```

```
        COUNTER += 1

        # if the eyes were closed for a sufficient number of
        # then sound the alarm
        if COUNTER >= EYE_AR_CONSEC_FRAMES:
            # if the alarm is not on, turn it on
            if not ALARM_ON:
                ALARM_ON = True
            # draw an alarm on the frame
            cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            playsound('alarm.wav')

    # otherwise, the eye aspect ratio is not below the blink
    # threshold, so reset the counter and alarm
    else:
        COUNTER = 0
        ALARM_ON = False

    # draw the computed eye aspect ratio on the frame to help
    # with debugging and setting the correct eye aspect ratio
    # thresholds and frame counters
    cv2.putText(frame, "EYE: {:.2f}".format(ear), (300, 30),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# show the frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF

# if the `q` key was pressed, break from the loop
if key == ord("q"):
```

```
        break

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

5.1.2. Screenshots of the system shown below :

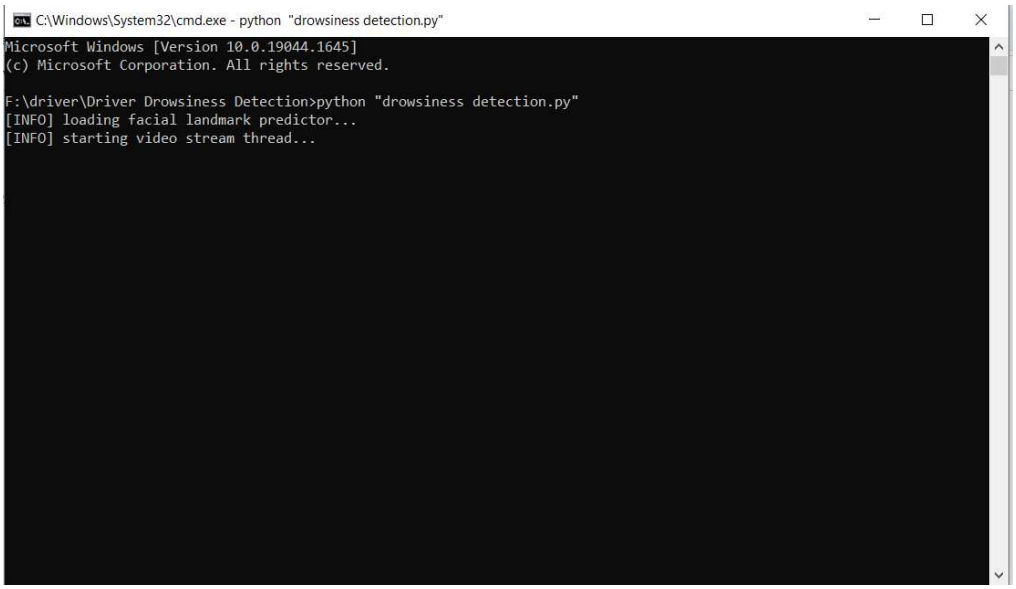

Fig 5.1: Output 1

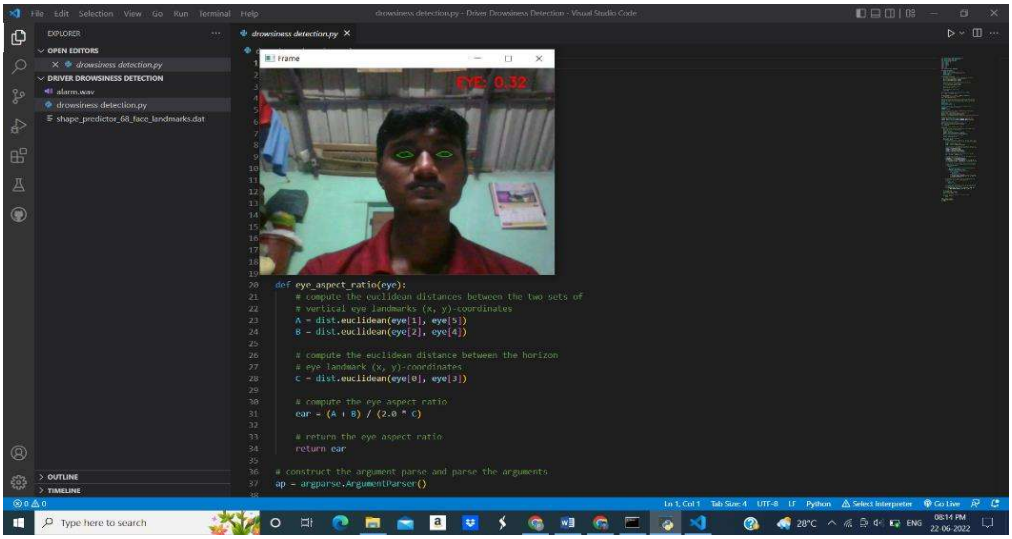

Fig 5.2: Output 2

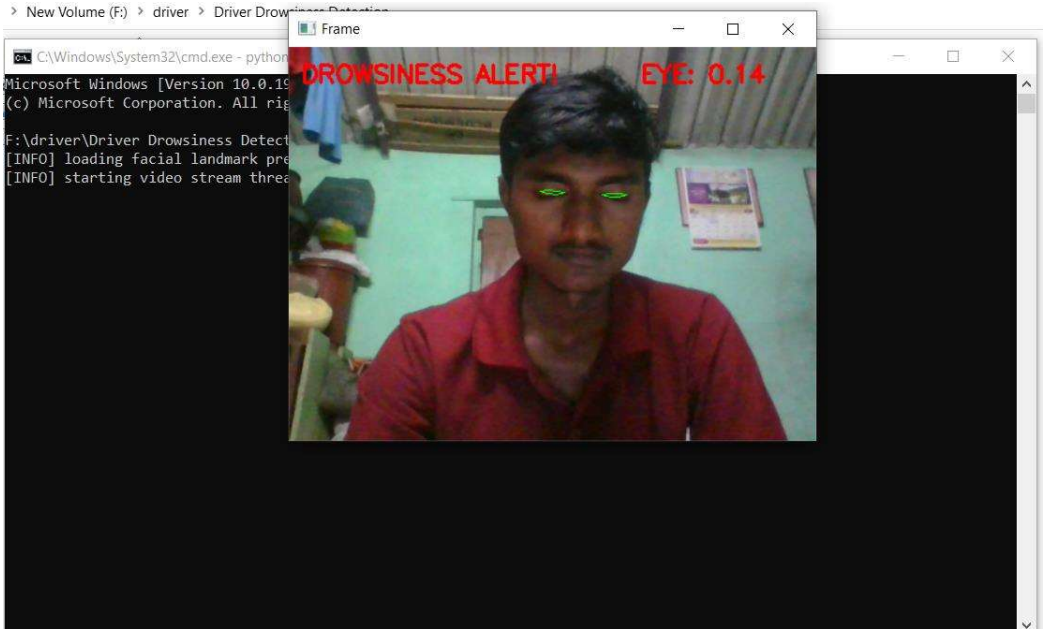

Fig 5.3: Output 3

5.1.3. Future Work:

Our model is designed for detection of drowsy state of eye and give and alert signal or warning in the form of audio alarm. But the response of driver after being warned may not be enough to stop causing the accident meaning that if the driver is slow in responding towards the warning signal then accident may occur. Hence to avoid this we can design and fit a motor driven system and synchronize it with the warning signal so that the vehicle will slow down after getting the warning signal automatically.

Chapert 6

CONCLUSION

CONCLUSION

A real-time eye blink detection algorithm was presented. We quantitatively demonstrated that Haar feature-based cascade classifiers and regression-based facial landmark detectors are precise enough to reliably estimate the positive images of face and a level of eye openness.

While they are robust to low image quality (low image resolution in a large extent) and in-the-wild.

It absolutely meets the goals and requirements of the device. The framework has accomplished an unfaltering kingdom in which all the bugs have been disposed of. The framework cognizant customers who are acquainted with the framework and realize it is Retracted a focal point and the reality that it looks after the difficulty of stressing out for individuals having fatigue-associated problems to inform them about the drowsiness level whilst riding.

Chapter 7

REFERENCES

REFERENCES

7.1. IEEE Standard Journal Paper-

[1] Facial Features Monitoring for Real Time Drowsiness Detection byManu B.N, 2016 12th International Conference on Innovations in InformationTechnology (IIT)[Pg. 7881]https://ieeexplore.ieee.org/document/7880030

[2] Real Time Drowsiness Detection using Eye Blink Monitoring by Amna RahmanDepartment of Software Engineering Fatima Jinnah Women University 2015 NationalSoftware Engineering Conference(NSEC2015)https://ieeexplore.ieee.org/document/7396336

[3]Implementation of the Driver Drowsiness Detection System by K. SrijayathiInternationaln Journal of Science, Engineering and Technology Research (IJSETR)Volume 2, Issue 9, September 2013.

7.2. Names of Websites referred-

1. https://www.codeproject.com/Articles/26897/TrackEye-Real-Time-Tracking-Of-Human-Eyes-Using-ahttps://realpython.com/face-recognition-withpython/
2. https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python dlib/
3. https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/
4. https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-dlib-opencv-python/
5. https://www.codeproject.com/Articles/26897/TrackEye-Real-Time-Tracking-Of-Human-Eyes
6. https://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_detection.html
7. https://www.learnopencv.com/training-better-haar-lbp-cascade-eye-detector-opencv/