DRIVER DROWSINESS DETECTOR

Project report submitted in fulfillment of the requirement for the degree of

Bachelor Of Technology

In

Computer Science and Engineering

By

Mudit Arya (151305)
Samarth Dixit(151296)

Under the supervision of

Mr. Ravindara Bhatt

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT, SOLAN, H.P., INDIA

CERTIFICATE

Candidates' Declaration

I hereby declare that the work represented in this report entitled **"Driver Drowsiness Detector"** in fulfillment of the requirements for the award of the degree of **Bachelor in Technology in computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2018 to May 2019 under the supervision of **Mr. Ravindara Bhatt**(Assistant Professor-CSE)

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Mudit Arya (151305) *Samarth Dixit (151296)*

This is to certify that the statement presented by the candidate is correct to the best of my knowledge.

(Supervisor Signature)

Dr Ravindara Bhatt

ACKNOWLEDGEMENT

The satisfaction that accompanies the completion of the project would be incomplete without the mention of the people who made it possible.

I might want to accept the open door to thank and express our profound feeling of appreciation to our staff guide Mr. Ravindara Bhatt for giving their significant direction at all the phases of the examination, valuable proposals and consistent consolation, without which it would have not been conceivable to finish the venture. The assistance and direction given by him every once in a while will convey us far in the adventure of our profession which we are going to set out.

I am grateful for their cooperation during the period of the project. I hope that I can build upon the experience and knowledge that i have gained and make a remarkable mark in the future.

TABLE OF CONTENTS

ABSTRACT

A computer vision based thoughts have been used for the creation of a Drowsy Driver Detection System. The little camera has been utilized by framework that concentrates straight towards the essence of driver and checks the driver's eyes with a particular ultimate objective to perceive weakness. A notice sign is issued to alert the driver, in such circumstance when exhaustion is perceived. The framework oversees using information picked up for the picture to find the facial tourist spots, which gets the area where the eyes of an individual may exist. On the off chance that the eyes of driver are discovered close for a specific measure of casings, the proposed framework accept that the driver is falling asleep and an alarm of caution has been issued. The structure can work just when the eyes are found, and works in encompassing lighting conditions too.

List of Figures

9	Facial landmarks set which is detected via dlib
10	Plot of eye aspect ratio over time

List of Tables

CHAPTER 1: INTRODUCTION

1.1 Introduction

Driver exhaustion is a noteworthy factor in countless mishaps. Late measurements gauge that yearly 1,200 passings and 76,000 wounds can be credited to fatigue related crashes

Driver drowsiness and fatigue is a major factor which results into numerous vehicle accidents. Developing and maintaining technologies which can efficiently detect or prevent drowsiness at the wheel and alert the driver before am mishap is a major challenge in the field of accident prevention systems. Because of the dangerous that drowsiness can cause on the roads some methods need to be developed for preventing counteracting its effects.

With the advent of modern technology and real time scanning systems using cameras we can prevent major mishaps on the road by alerting car driver who is feeling drowsy through a drowsiness detection system

The point of this undertaking is to build up a prototype drowsiness detection system. The spotlight will be put on planning a framework that will precisely monitor the open or shut condition of the driver's eyes continuously.

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.

1.2 Problem Statement

Designing a prototype Drowsiness Detection System which will focus on continuously and accurately monitoring the state of the driver's eyes in real time to check whether they are open or closed for more than a given period of time

1.3 Objectives

Driver drowsiness detection is a car safety technology which spares the life of the driver by avoiding mishaps when the driver is getting languid.

- The primary goal is to initially plan a framework to distinguish driver's sluggishness by persistently checking retina of the eye.
- The framework works disregarding driver wearing displays and in different lighting conditions.
- To caution the driver on the identification of laziness by utilizing ringer or alert.
- Speed of the vehicle can be reduced.
- Traffic management can be maintained by reducing the accidents.

1.4 Methodology

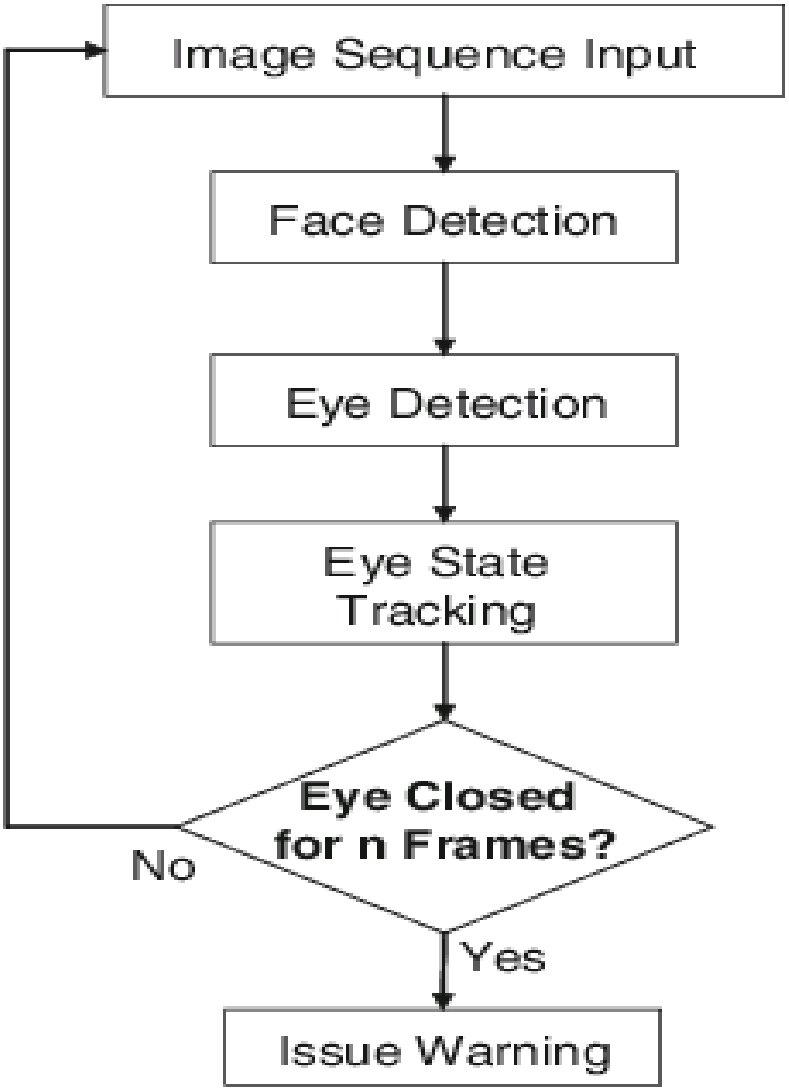


Fig1. Methodology

System Architecture of Driver Drowsiness Detection System

SYSTEM REQUIREMENTS

- **Hardware Requirements**:-

  Laptop, Camera, Monitor, Alarm, Mouse, Keyboard.

- **Software Requirements**:-

  OpenCV, dlib, and Python

The flowchart of the proposed system has been shown in the above figure. The camera captures the image and sends to the processor of the laptop which consists of 32 bit memory card installed with Open CV which helps in image processing.

If the signal crosses threshold of a set of continuous frames with EAR less than threshold value , it will automatically makes the alarm beep and the speed of the vehicle gets reduced. Otherwise that signal is rejected and next signal is processed.

WORKING :

Drivers face is monitored throughout using a video or web camera. In order to detect the drowsiness the first step is to detect the face using the set of frames taken by the camera. Then the location of the eyes is detected and retina of the eye is continuously monitored. The captured image is sent to the processor for image processing. It converts the received image to digital signal using Open CV.

The digital signal is transmitted from transmitter to the receiver. Both the transmitter and
the receiver are paired up. The signal is then passed to the LPC2148, the microcontroller. If the signal crosses the threshold value of EAR for a given number of frames, then the alarm beeps and the speed of the vehicle is automatically reduced.

Fig. Real time model

1.5 Organization of the report

The report is divide into five chapters. **Chapter 1** is a brief introduction about the project. It tells about the Objectives and Methodology of the project. **Chapter 2** is a in depth analysis of all the Research Papers and Documents that were used in making of this project. **Chapter 3** is an experimental analysis of how the project will be working when put under various test case scenarios. **Chapter 4** devises all the algorithms that would fuel up the project and finally **Chapter 5** is a Test plan on how the setup will be made and what data needs to extracted to improve our system.

CHAPTER 2: LITERARY SURVEY

This part presents the literatrary survey of drowsiness detection approaches. According to the Survey on Driver Fatigue-Drowsiness Detection System, the detection system includes the processes of face image extraction, yawning tendency, blink of eyes detection, eye area extraction etc.

There are many experiments done with OpenCv for android also which is available for cheap smartphones as well. Other experiments conducted have resulted in utmost accuracy when camera were placed at different locations.

OpenCv is predominantly a technique for real time image processing which has free of cost implementations on latest computer vision algorithms. It has all required computer vision algorithms.

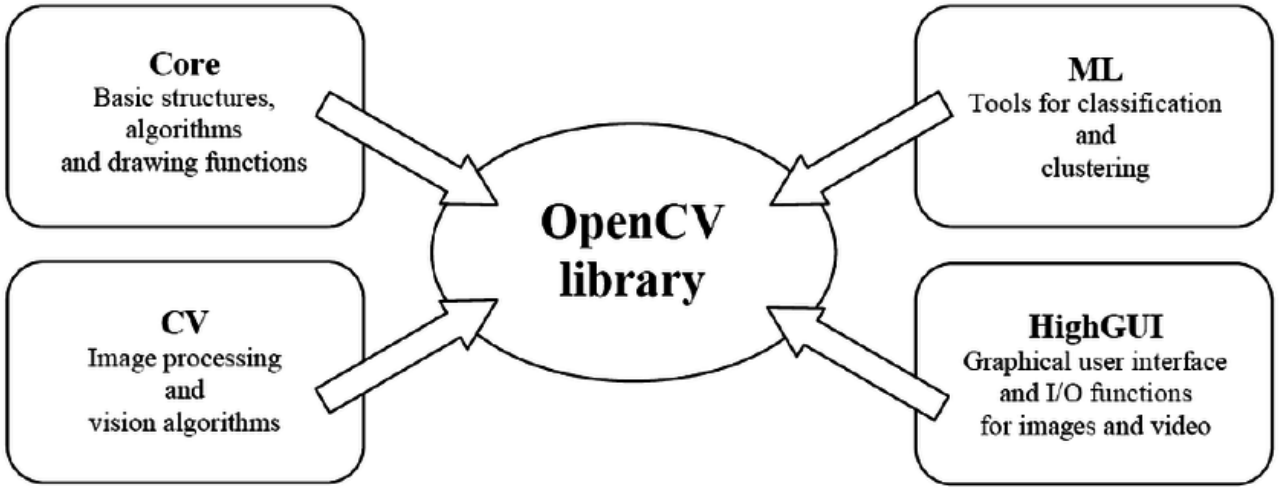


Fig2. OpenCv library

Through analysis bestowed during this paper, we have a tendency to developed associate nonintrusive image laptop vision framework for timespan checking of a driver's cautiousness. To begin with, the obligatory equipment and imaging calculations square measure created to at the same time remove various viewable signs that for the most part describe an individual's dimension of weariness. At that point, a probabilistic structure is developed to demonstrate weakness, that reliably joins totally extraordinary obvious signals and in this manner the applicable talk data to supply a tough and steady weariness file. These obvious signals portray palpebra movement,gaze,head movement,and facial highlights. The most pieces of the framework incorporate an equipment framework for the timeframe obtaining of video photos of the main impetus and differed PC vision calculations and their PC code.

Validation of this study has 2 parts in it. The first one involved the accuracy (of the computer vision algorithms / techniques) validation. Then the second involved the validation of the fatigue parameters that were computed in defining the extent of the fatigue.

The outcomes of the study showed that the prototype is efficient, dependable and accurate as well in detecting the drowsiness of the driver. All this was done in real-time and represents non-intrusive fatigue monitoring.

In another study which they tried developing similar fatigue detection system, the algorithm they proposed was designed to get the facial parts that include eyes and lips. For reducing the search areas in the provided input images, the algorithm extract the skin pixels as well which tells a lot about the possibilities that what all an algorithm can include in order to compute the drowsiness of the driver.

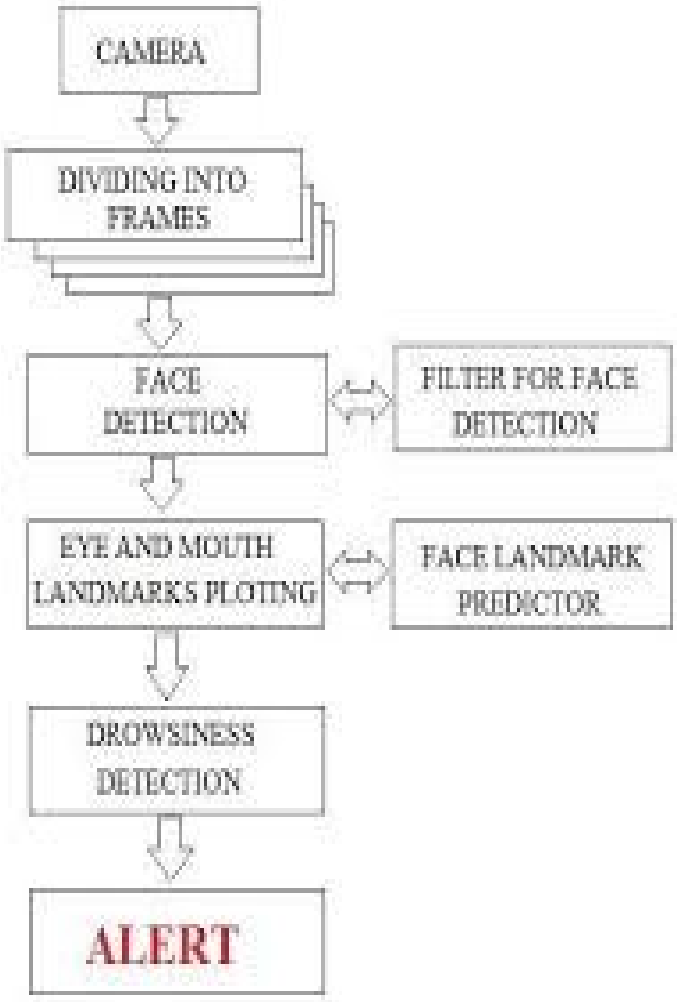
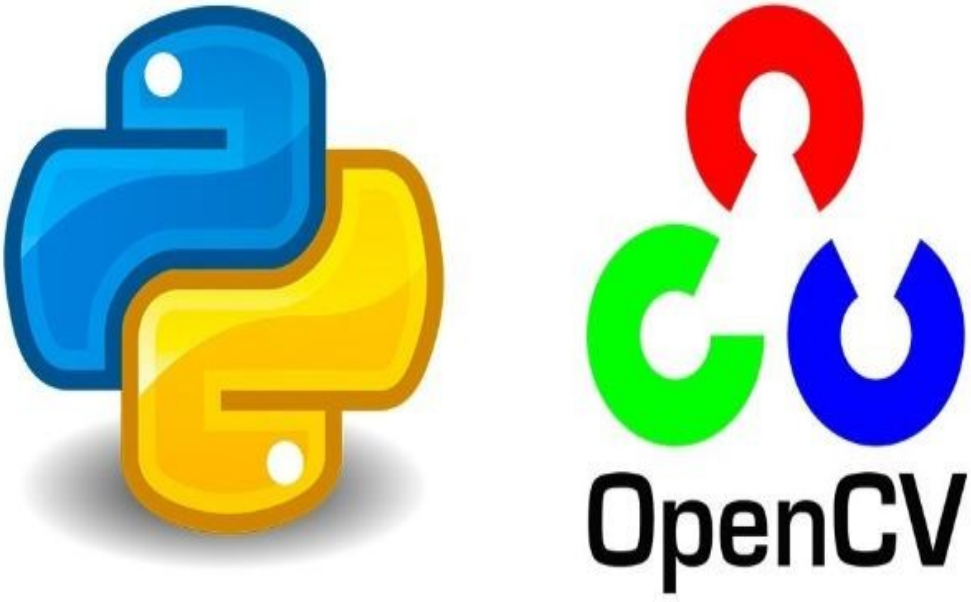


Fig3.overview of survey

OpenCV

OpenCV is an open source computer vision library accessible in python coding language to code for visionary capabilities of our smart pc.

OpenCV was expected for computational capability and having a high focus on ongoing picture location and distinguishing proof. OpenCV is coded with streamlined C and can take work with multicore processors. If we need progressively programmed improvement utilizing Intel models [Intel], you can purchase Intel's Integrated Performance Primitives (IPP) libraries [IPP]. These comprise of low-level schedules in different algorithmic regions which are streamlined. OpenCV consequently utilizes the IPP library, at runtime if that library is introduced.



The Computer’s Vision

PC's vision is the change of information from a still, or camcorder into either a depiction or another choice. Each and every such changes are performed to achieve a particular target. A Computer gains a cross section of numbers from a camera or from the circle, and it's just as simple as that. For the most part, there is no worked in example acknowledgment or programmed control of center and gap, no cross-relationship with long periods of experience. Generally, vision frameworks are still reasonably gullible.

The Origin of OpenCV

OpenCV left an Intel Research action proposed to drive CPU-raised applications.Toward this end, Intel moved various endeavors that included constant beam following and moreover 3D show dividers. One of the product engineers working for Intel at the time was visiting schools. He saw that several top school social affairs, like the MIT Media Lab, used to have well-made similarly as inside open PC vision frameworks—code which was supplied starting with one understudy then onto the next and which gave each resulting understudy an important establishment while building up his own vision application. Rather than rehashing the fundamental capacities from starting, another understudy may begin by adding to that which preceded .

OpenCV Structure and Content

OpenCV left an Intel Research movement planned to drive CPU-raised applications.Toward this end, Intel pushed various endeavors that included continuous beam following and moreover 3D show dividers. One of the product engineers working for Intel at the time was visiting schools. He saw that two or three top school social events, like the MIT Media Lab, used to have well-made similarly as inside open PC vision foundations

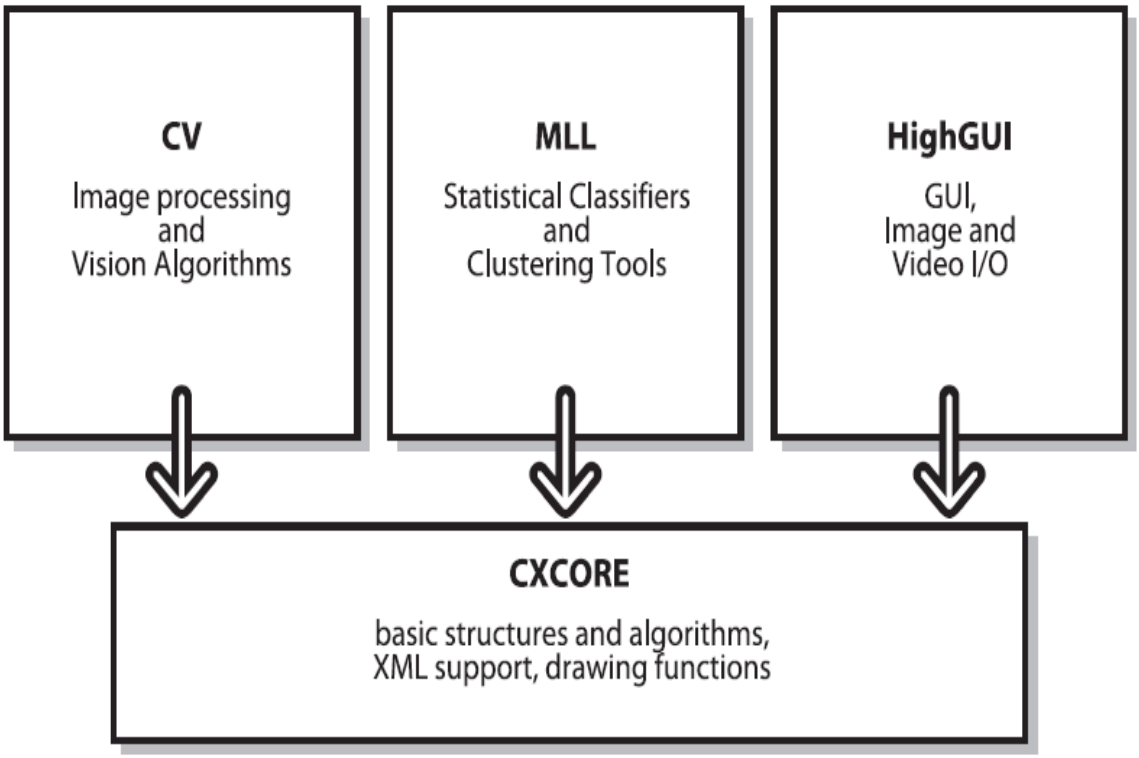


Fig: Parts of OpenCV

Why Open CV?

- **Specific**

OpenCV was planned for picture handling. Each capacity and information structure has been arranged in perspective on an Image Processing application. Then, Matlab, is very conventional.
You can get almost everything on the planet by methods for tool compartments. It may be money related tool stash or then again concentrated DNA tool compartments

- **Speedy**

Matlab is just excessively moderate. Matlab itself depended on Java. Similarly Java depended on C. So when we run a Matlab program, our PC gets caught up with attempting to translate and assemble all that convoluted Matlab code. At that point it is transformed into Java, lastly executes the code.

In case we use C/C++, we don't waste such time. We direct give machine language code to the PC, and it gets executed. So in the end we get more picture taking care of, and not additionally interpreting.
Ensuing to doing some constant picture handling with both Matlab and OpenCV, we typically got low speeds, a point of confinement of around 4-5 outlines arranged each second with Matlab. With OpenCV in any case, we get genuine persistent dealing with at around 30 outlines being handled every second.

Beyond any confusion we give the prize for speediness – a progressively enigmatic language to handle, yet it's unquestionably of true worth . We can complete a lot more work, as calculate some extremely perplexing arithmetic on pictures utilizing C and still pull off adequatespeeds for your application.

- **Efficient**

Matlab utilizes just an excessive amount of system assets. With OpenCV, we can pull off as pitiful as 10mb RAM for a constant application. Notwithstanding the way that with the present PCs, the RAM factor is surely not a noteworthy thing to be worried over. In any case, our tiredness identification framework is to be used inside a vehicle in a way that is non-meddlesome and little; so a low handling necessity is vital.

Subsequently we shall perceive as to how OpenCV is superior for a real-time drowsiness detection system

Machine Learning

The objective of AI is to transform information into data. Subsequent to having gained from a social affair of information, we need a machine that can address any question about the information:

- What are different information that are like given information? Is there a face in the picture?

- What sort of advertisement will impact the client?

- There is generally a cost parameter, hence the question arises :
    - Of the numerous items that we can profit from, which one will probably be purchased by the client if a promotion is appeared for it?

AI changes over information into data by identifying standards or examples from that information.

OpenCV's Machine Learning Algorithms

The ML calculations that are incorporated into OpenCV are given as pursues. Every one of the calculations are available in the ML library separated from Mahalanobis and K-implies, which are available in CVCORE, and the calculation of face recognition, which is available in CV. [4]

Mahalanobis:

It is a measure of distance that is responsible for the stretchiness of the data. We can divide out the covariance of the given data to find this out. In case of the covariance being the identity matrix (i.e. identical variance), this measure will be identical to the Euclidean distance. [4]

K-means Algorithm:

It is an unsupervised clustering algorithm which signifies a distribution of data w.r.t. K centers, K being chosen by the coder. The difference between K-means and expectation maximization is that in K-means the centers aren't Gaussian. Also the clusters formed look somewhat like soap bubbles, as centers compete to occupy the closest data points. All these cluster areas are usually used as a form of sparse histogram bin for representing the data. [4]

Normal or Naïve Bayes classifier algorithm:

It is a generative classifier where features are often assumed to be of Gaussian distribution and also statistically independent from one another. This assumption is usually false. That's why it's usually known as a ―naïve Bayes‖ classifier. That said, this method usually works surprisingly well. [4]

Decision trees algorithm:

It is a partially discriminative classifier. The tree we talk about just finds a singular data feature and determines a threshold value of the current node which best divides the data into different classes. The data is broken into parts and the procedure is recursively repeated through the left as well as the right branches of the decision tree. Even if it is not the top performer, it's usually the first thing we try as it is fast and has a very high functionality. [4]

Boosting:

It is a discriminative group of classifiers. In boosting, the finalclassification decision is made by taking into account the combined weighted classification decisions of the group of classifiers. We learn in training the group of classifiers one after the other. Each classifier present in the group is called a weak classifier. These weak classifiers are usually composed of single-variable decision trees known as ―stumps. Learning its classification decisions from the given data and also learning a weight for its vote based on its accuracy on the data are things the decision tree learns during training. While each classifier is trained one after the other, the data points are re-weighted to make more attention be paid to the data points in which errors were made. This continues until the net error over the entire data set, obtained from the combined weighted vote of all the decision trees present, falls below a certain threshold. This algorithm is usually effective when a very large quantity of training data is available. [4]

Random trees algorithm:

It is a discriminative woods of a great deal of choice trees, every one of which is worked down to a maximal part profundity. At the season of

adapting, each hub of each tree is permitted a decision of part factors, however just from an arbitrarily produced subset of the considerable number of information highlights. This makes sure that all the trees become statistically independent and a decision maker. In the run mode, all the trees get an unweighted vote. Random trees are usually quite effective. They can also perform regression by taking the average of the output numbers from every tree. [4]

Face detector algorithm:

It is an object detection application. It is based on a smart use of boosting. A trained frontal face detector is available with the OpenCV distribution. This works remarkably well. We can train the algorithm for other objects by using the software provided. This works wonderfully for rigid objects with characteristic views. [4]

Expectation maximization (EM) algorithm:

It is used for clustering. It is a generative unsupervised algorithm It fits N multidimensional Gaussians to the data, N being chosen by the user. It can act as an efficient way for representing a more complex distribution using only a few parameters (i.e. means and variances). Usually used in segmentation, it can be compared with K-means. [4]

K-nearest neighbors:

It is one of the simplest discriminative classifier. The training data is simply stored using labels. Then, a test data point is classified in accordance to the majority vote of the K nearest data points. K-nearest neighbours is probably

th simplest algorithm we can use.It is usually effective but it can be slow. It also requires a lot of memory. [4]

Neural networks / Multilayer perceptron (MLP) algorithm:

It is a discriminative algorithm which almost always contains hidden units in between the output and the input nodes for better representation of the input signal. It is slow to train, however it is quite fast to run. It remains the best performer for applications like letter recognition. [4]

Support vector machine (SVM) algorithm:

It is a discriminative classifier that is likewise equipped for doing relapse. Here, a separation work in the middle of two information focuses is characterized in a higher-dimensional space. (Anticipating information onto higher measurements helps in making the information more probable for direct division.) Support vector machine (SVM) gets the hang of isolating hyperplanes which maximally separate every one of the classes in the higher measurement. This will in general be the best when there is restricted information. Be that as it may, when huge informational indexes are accessible, boosting or arbitrary trees are liked. [4]

Getting to use Machine Learning for Computer Vision

Typically, most calculations take an information vector having numerous highlights as info. Here, the quantity of highlights may number in the thousands. On the off chance that our undertaking is perceiving a particular sort of article—take for instance, an individual's face. The main issue that we experience is getting and marking the preparation information which falls into positive (for example there is a face in the window) and negative (for example no face) cases. We before long understand that countenances can show up at different scales: for example their picture may comprise of just a couple of pixels, or we may take a gander at an ear which is filling the entire screen. More terrible still, faces are typically impeded. We need to characterize what we really mean when we state that a face is in the window. [4]

Subsequent to having marked the information that was acquired from different sources, we ought to choose which highlights we have to remove from these items. Likewise, we should recognize what objects we are after. On the off chance that the countenances dependably seem upstanding, there is no purpose behind utilizing revolution invariant highlights and furthermore no explanation behind endeavoring to turn the articles before handling.

All in all, we should attempt to discover highlights that express a little invariance in the articles. These can be scale-tolerant histograms of angles or hues or even the prominent SIFT highlights.

When we have imperative foundation window data, we would first be able to evacuate it so as to enable different articles to emerge. At that point we play out

our picture handling. This may comprise of normalizing the picture and after that registering the different highlights. The subsequent information vectors are altogether given the name that is related with the article, activity, or window. [4]

When the information is gotten and changed over into highlight vectors, we separate the information into preparing sets, approval sets and test sets. It is fitting to do our learning, approval, and testing utilizing a cross-approval structure. Here, the information is part into K subsets and we run different preparing (possibly approval) just as test sessions. Every session comprises of different arrangements of information that assume the jobs of preparing (approval) and test. The test outcomes acquired from these different sessions are utilized for averaging to get the last execution result. An increasingly precise picture of how the classifier performs when sent in task can be given by cross-approval.

Since our information is prepared, we should pick a classifier. Generally the decision of the classifier is controlled by computational, information, and memory necessities. For specific applications, as online client inclination displaying, we have to prepare the classifier rapidly. In such a case, closest neighbors, ordinary Bayes, or choice trees ought to be a decent decision. At the point when memory is the essential thought, choice trees or neural systems are utilized for their space productivity.

When we have sufficient energy to prepare our classifier yet it needs to run rapidly, neural systems can be a decent decision, similarly as with ordinary Bayes classifiers and bolster vector machines. When we have room schedule-wise to prepare yet require high exactness, at that point boosting and irregular trees are great decisions. When we simply need a simple and reasonable check climate our highlights are picked well or not, at that point choice trees or closest neighbors ought to be utilized. For a decent out of the container characterization execution, boosting or irregular trees are attempted. [4]

CHAPTER-3: SYSTEM DEVELOPMENT

3.1 Computational Analysis

Matching

Dashboard mounted camera is used to monitor the eyes of the driver in real time to detect drowsiness

3.1.1 Drowsiness Detection Design

A camera is setup that looks for faces in the input video stream and monitors frames of faces. In the event that a face is identified, facial milestone identification is connected and the eye district is removed from the edges of the video stream.

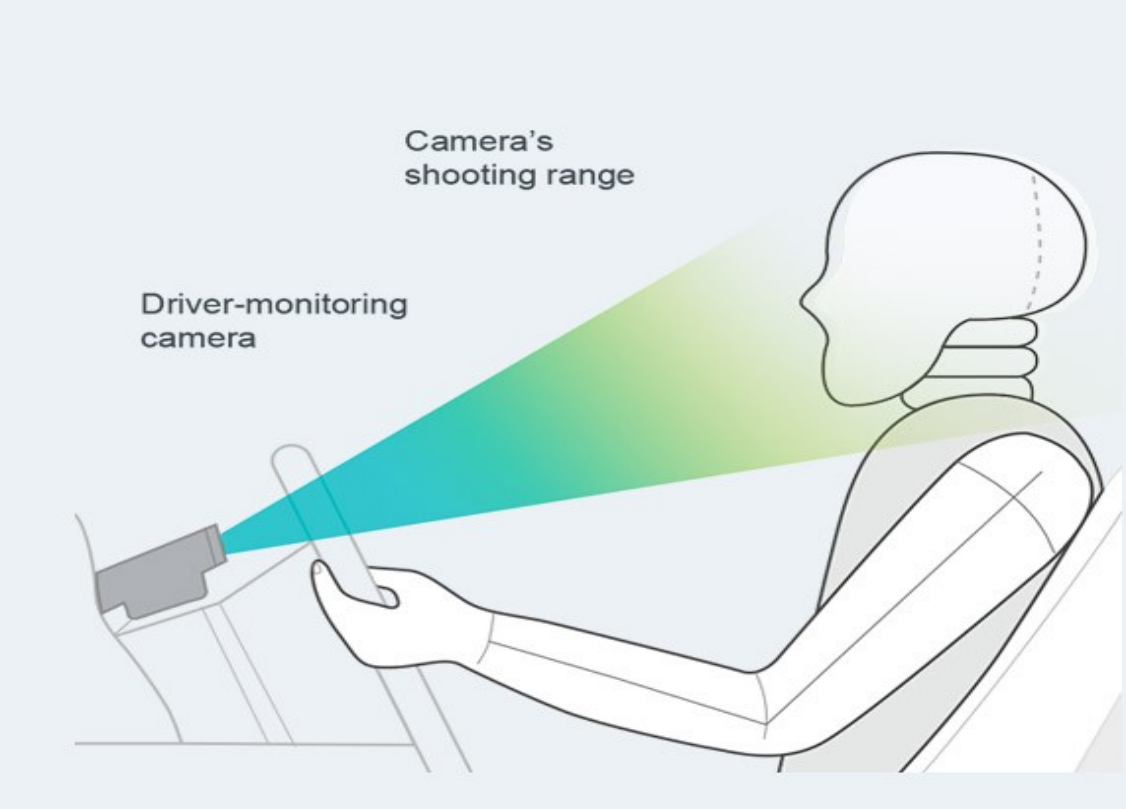


Fig4. Practical design of the system

Based on the work by Soukupová and Čech in their 2016 paper, *Real-Time Eye Blink Detection using Facial Landmarks*, we can then derive an equation that reflects this relation called the *eye aspect ratio* (EAR):

$$\mathrm{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

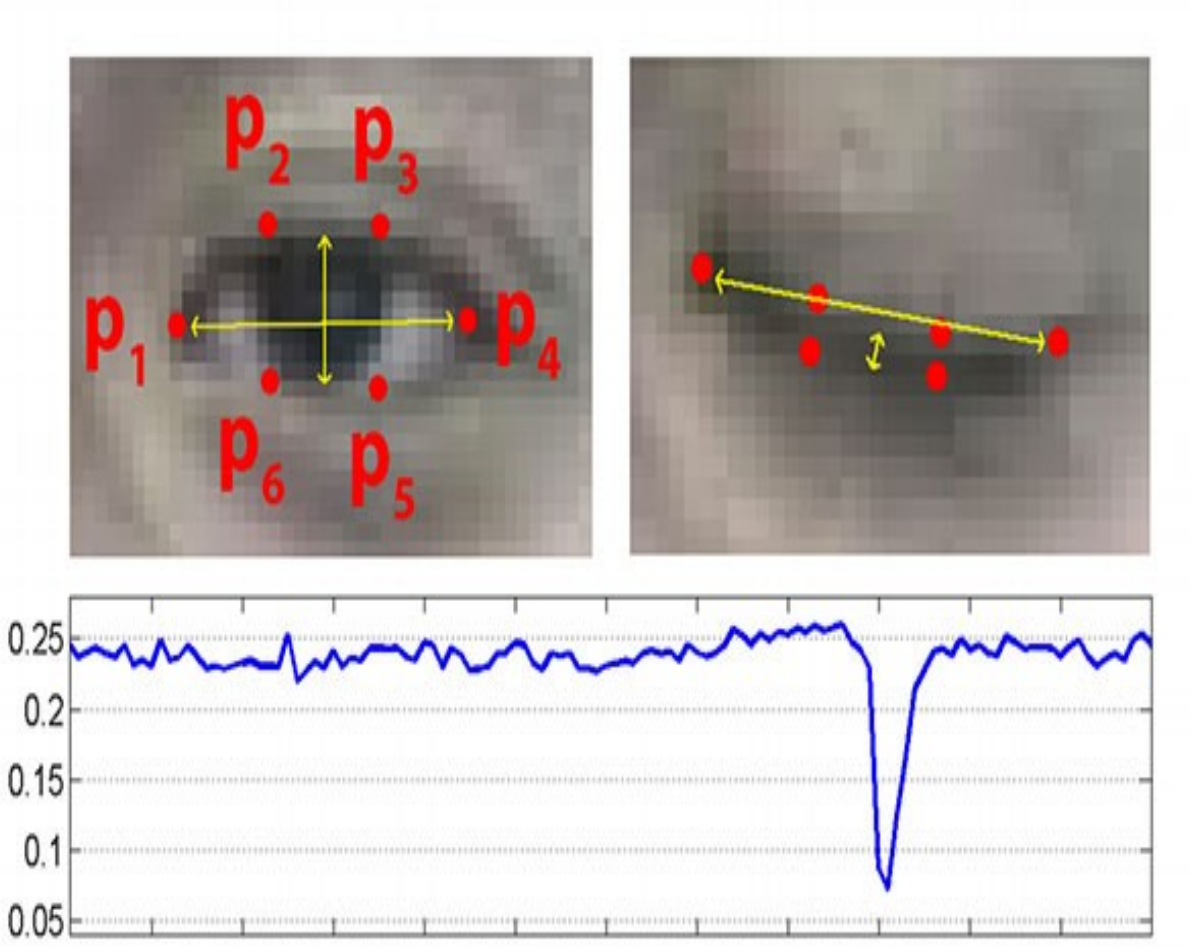
Figure 4: The eye aspect ratio equation.

Where *p1, ..., p6* are 2D facial landmark locations.

The numerator of this equation computes the distance between the vertical eye landmarks while the denominator computes the distance between horizontal eye landmarks, weighting the denominator appropriately since there is only *one* set of horizontal points but *two* sets of vertical points.

Fig5.Eye Aspect Ratio Calculation Formula

If the aspect ratio of the eye indicates that they have been closed for greater than a dedicated fixed time, we will sound an alarm system so that the driver wakes up.



Picture 1 : Eye marks when the eye is open

Picture 2 : Eye marks when the eye is closed

Picture 3: Eye Aspect Ratio plotted over time, the downfall in the aspect ratio graph shows a blink of the driver

fig6. Eye aspect ratio

We observe the aspect ratio of the eye remains constant for a period of time indicating that the eye was open, then it falls rapidly to zero and then increases again which indicates the person blinked

We will be observing this eye aspect ratio in our drowsiness detector case to see if the value remains constant or falls to zero but not increases again implying that the driver has closed his eyes for extended period

3.1.2 Developing Image Processing solutions using OpenCV & dlib

OpenCV was developed keeping image processing in mind. Every function and data struct of OpenCV concerns itself with an Image Processing library. Comparatively, Matlab is hugely of generic use & slow.

Any usefulness can be accomplished by methods for tool kits in OpenCV, it might be money related tool compartments or explicit DNA tool stash

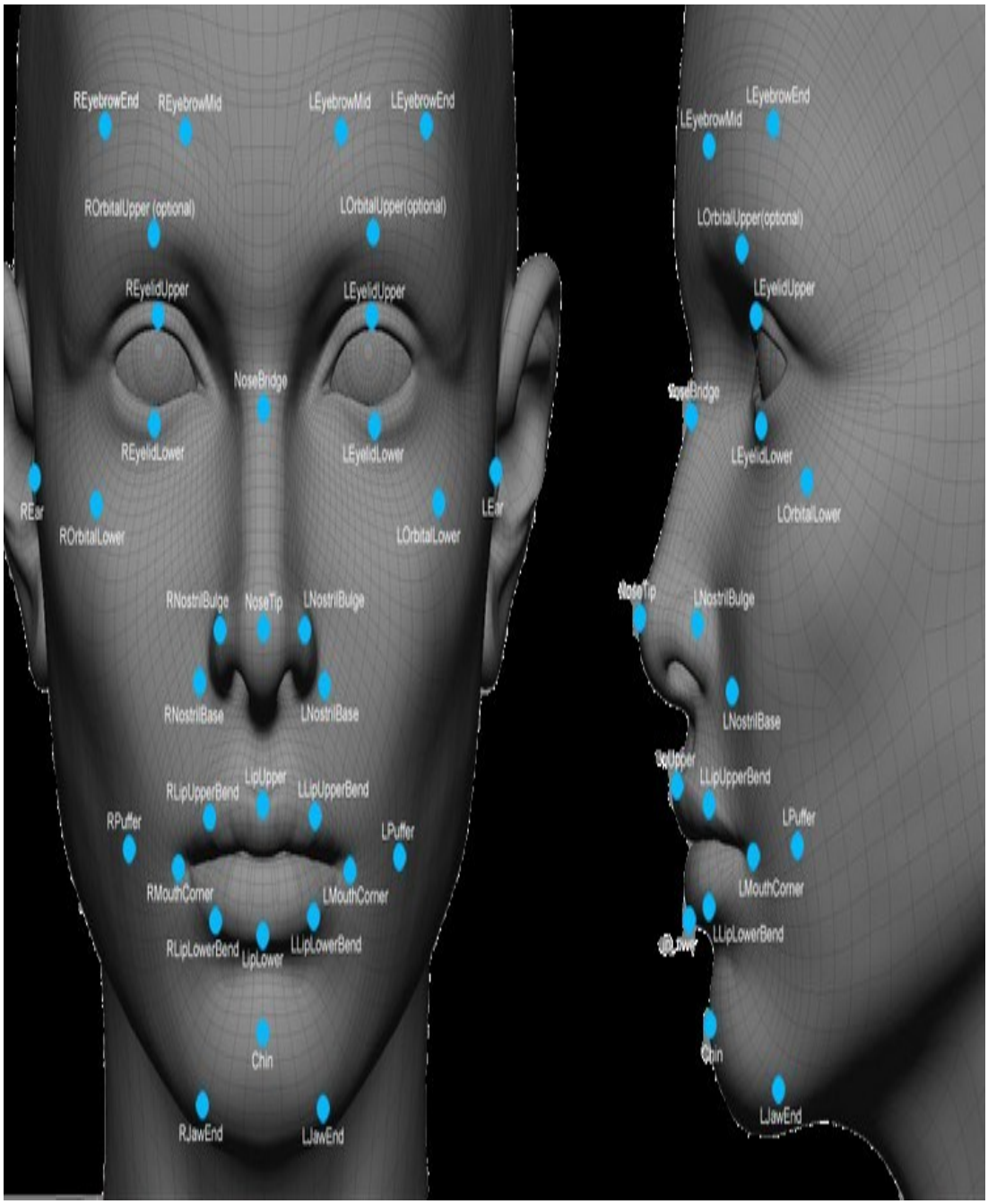


Fig7. Facial landmarks by opencv

Also the dlib library comes with a oriented gradients based face detector histogram a facial landmark predictor comes bundled in the library.

Facial landmarks generated by dlib is an indexable list as described in below image

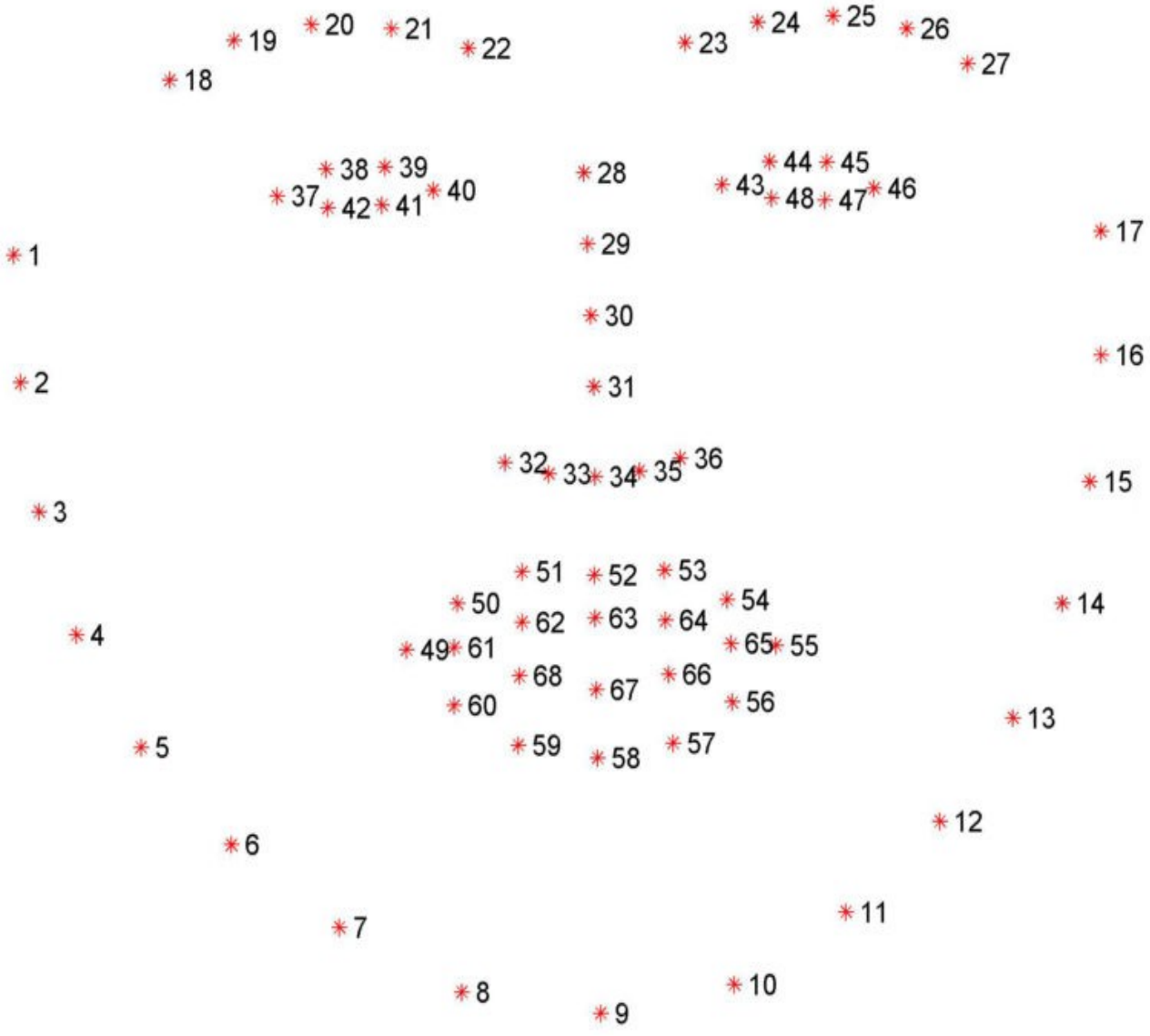
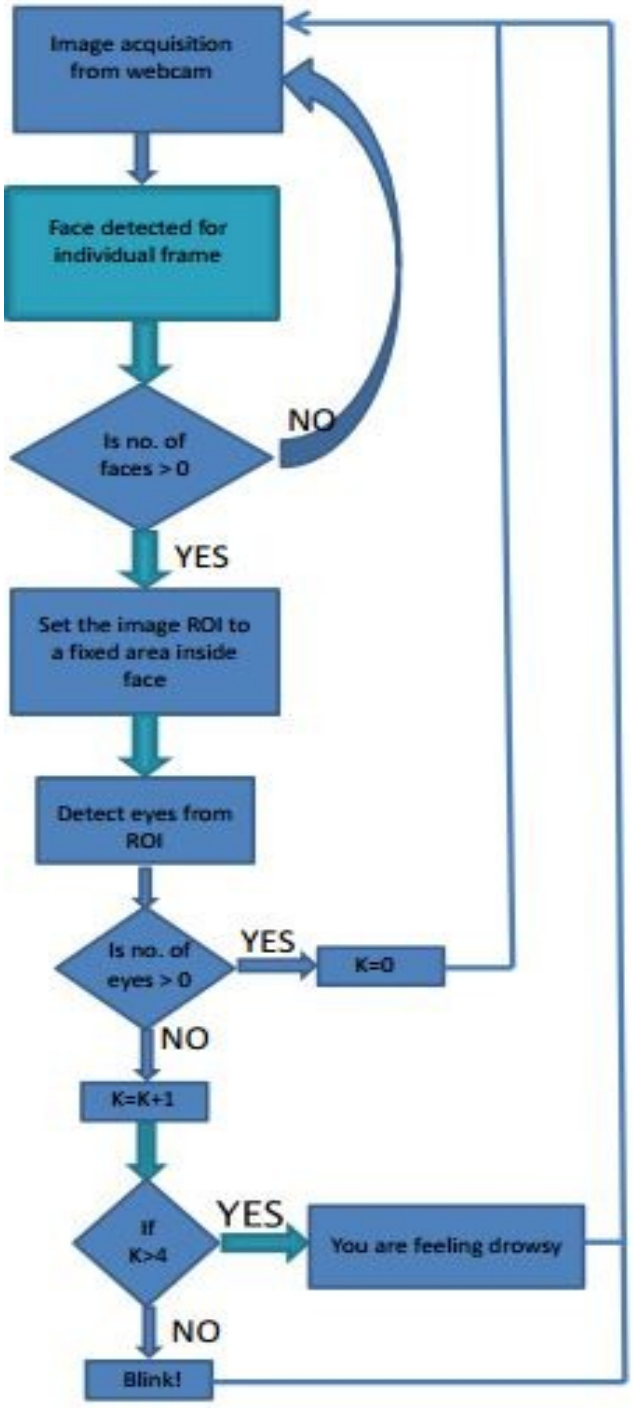


Fig8.Facial landmarks set which is detected via dlib

CHAPTER-4: ALGORITHM & IMPLEMENTATION



Algorithmic steps:-

The overall algorithm is pretty straightforward one. First we have used a camera which is setup at desirable position in a car that looks for faces stream.

If face gets detected, the facial landmark detection task is applied and region of eyes is extracted.

Once we get the eye region, we calculate the Eye Aspect Ratio to find out if the eye-lids are down for a substantial amount of time.

On the off chance that the Eye Aspect Ratio demonstrates that the eyes are shut for a considerably long measure of time, the alert will sound noisy to wake the driver up.For the functionalities of the system and to make it work efficiently we have used OpenCv, dlib and Python.

The implementation of the drowsiness detector system includes machine learning algorithms which are in turn included in OpenCv ML algorithms. There are numerous ML algorithms but for our purpose we required only the **face detector algorithm**.

It is fundamentally an item discovery ground-breaking application. Additionally, prepared frontal face identifier is accessible with the OpenCv circulation.
It works efficiently well overall. It can also be used to detect various different types of objects with the required software.

4.1 Starting to build the detector system with OpenCv

First, we create a new file drowsy_detect.py and write the following script in it : -

Drowsiness detection with OpenCV Python

```python
# import the necessary packages
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 playsound
import argparse
import imutils
import time
import dlib
import cv2
```

Now for calculating the eye-aspect ratio we need to compute the Euclidean distance between the facial landmarks points which in turn needs SciPy package in python. (It not a strict requirement but SciPy is needed if work related to computer vision or image processing is intended).

Also the package named imutils is needed for image processing and computer vision functions to assist the working with OpenCv.

The thread class is imported so that we can beep the alarm in a different thread from main thread so that it is ensured that our script doesn't stop/pause executing while the alarm beeps. In order to play a file of the wav or mp3 format, we need playsound library.

4.2 Facial landmarks and eye aspect ratio calculation

For detecting and localising facial landmarks we will require the dlib library hence we import it.

Eye_aspect_ratio function is defined to calculate the distance between the eye landmarks taken vertically and distances between the eye landmarks taken horizontally.

Drowsiness detection with OpenCV — Python

```
18 def eye_aspect_ratio(eye):
19     # compute the euclidean distances between the two sets of
20     # vertical eye landmarks (x, y)-coordinates
21     A = dist.euclidean(eye[1], eye[5])
22     B = dist.euclidean(eye[2], eye[4])
23 
24     # compute the euclidean distance between the horizontal
25     # eye landmark (x, y)-coordinates
26     C = dist.euclidean(eye[0], eye[3])
27 
28     # compute the eye aspect ratio
29     ear = (A + B) / (2.0 * C)
30 
31     # return the eye aspect ratio
32     return ear
```

So, when the eye is open, the value returned for the eye aspect ratio will be a constant approximately. Then the value will rapidly decrease reaching zero in case of an eye blink.

When the eye is closed, eye aspect ratio again approaches to an approximate constant value which is very smaller compared to that when the eye is open.

Therefore, the dip in the aspect ratio indicates blink of the eyes.

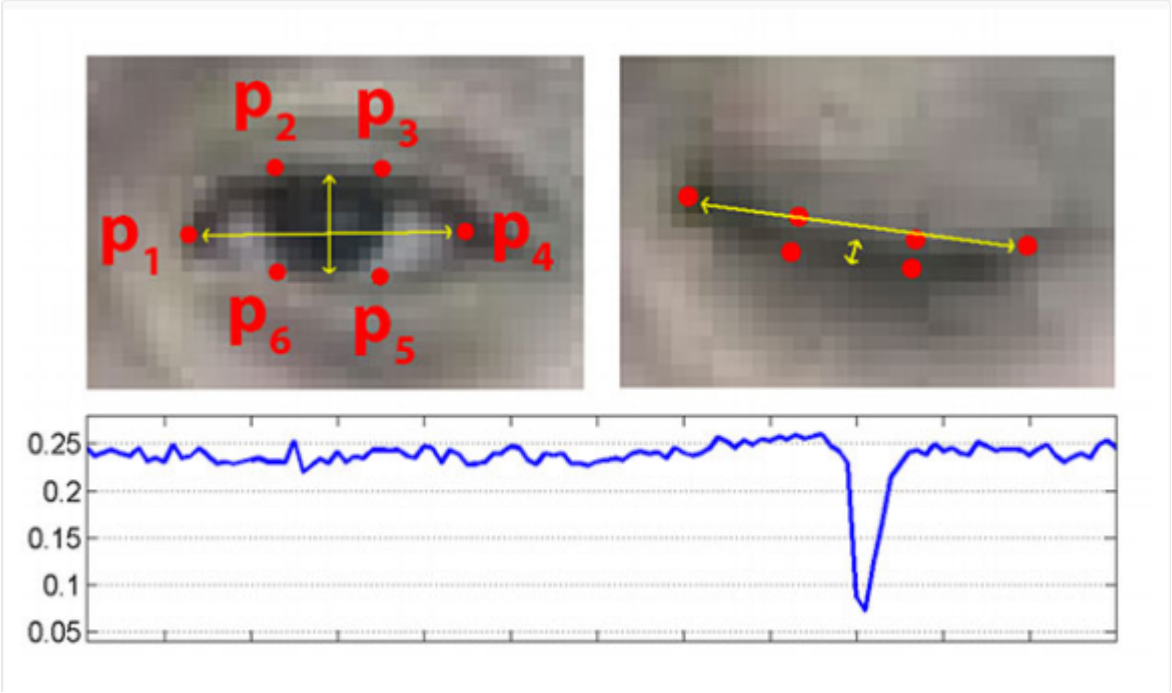


Fig9.Plot of eye aspect ratio over time

4.3 Important variables in the script

EYE_AR_THRESH: it's a threshold value for the eye aspect ratio. If ratio becomes lower than this value, counter starts for the number of frames the eyes remain closed.

EYE_AR_CONSEC_FRAMES: If the value for the number of frames for which the eyes remain closed exceeds this variable's value , the alarm is activated.

```
Drowsiness detection with OpenCV                                    Python
# 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 = 48

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

4.4 dlib library for face detection

the dlib library serves us with a facial landmark detector as well as facial landmark predictor.Below is the facial landmarks that are produced by the library.

Now from these landmarks, we just churn out the eye regions successfully.

```
Drowsiness detection with OpenCV                                    Python
# 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"]
```

.

CHAPTER-5: TEST PLAN

5.1 Test Cases to check the drowsiness

Following is the table representing four test cases that are to encountered while doing this project that concerns with the drowsiness of the driver.

Test cases	**Eyes Detected**	**Eye closure**	**Result**
Case1	NO	NO	No result
Case2	NO	NO	No result
Case3	YES	NO	No alarm
Case4	YES	YES	Alarm beeps

Table1 . Test Cases

At the point when the eyes are shut for more than certain measure of edges then we find that the driver is feeling tired. Henceforth these cases are distinguished is and a caution sounded.

CHAPTER 6. RESULT & PERFORMANCE ANALYSIS

6.1 Summary:

To get the outcome a large no of pictures were taken and their accuracy in deciding eye flickers and drowsiness was tried.

For this venture we utilized a 5 megapixel webcam associated with the PC. The webcam had inbuilt white LEDs connected to it to show it is working. In real time scenario, infrared LEDs ought to be utilized rather than white LEDs with the goal that the framework is non-meddling. Inbuilt speaker is utilized to deliver sound output so as to awaken the driver when drowsiness is detected.

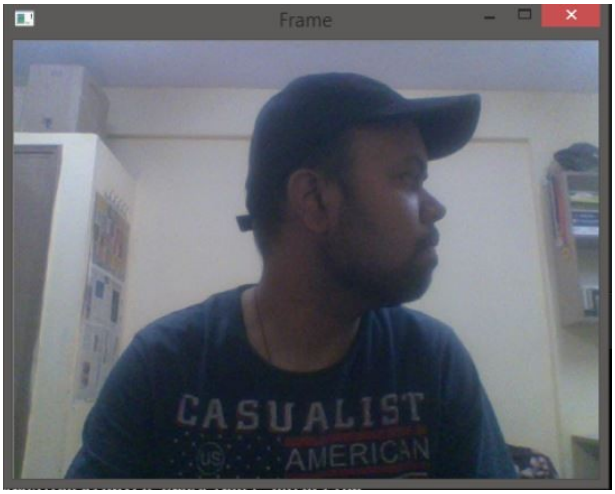
The framework was tried for various individuals in various surrounding lighting conditions (daytime and evening time). At the point when the webcam backdrop illumination was turned ON and the face is kept at an ideal distance, at that point the framework can identify blinks and drowsiness with over 95% accuracy.

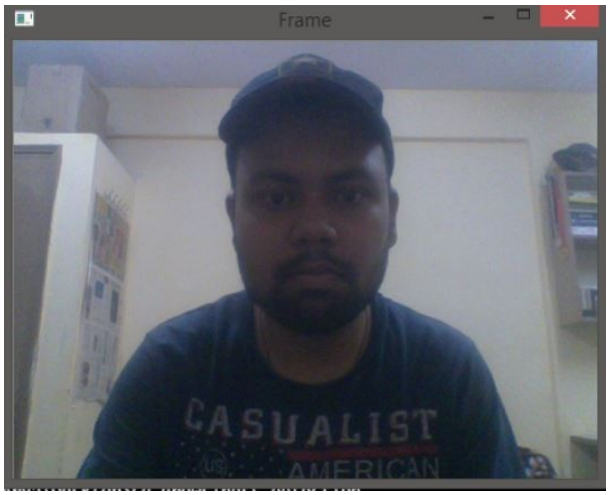
This is a decent outcome and can be executed by real time systems as well. Sample outputs for various conditions in different pictures is given beneath. Three pictures were taken; one in which just the eyes were identified and the other in which they were not and another where drowsiness is detected.

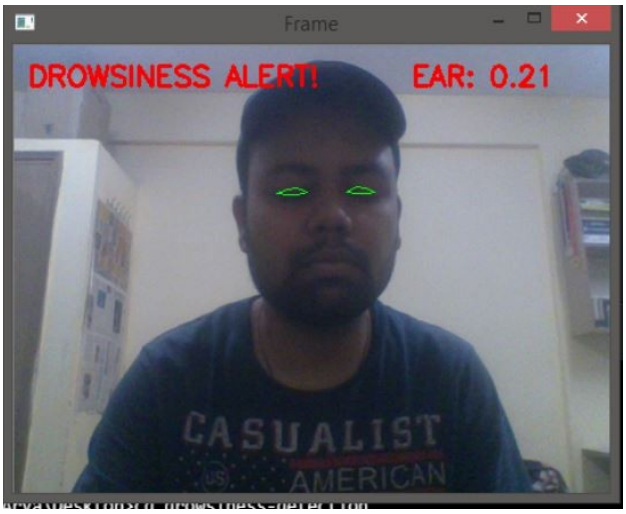
6.2 Section For Sa mple Images:

Sample set 1

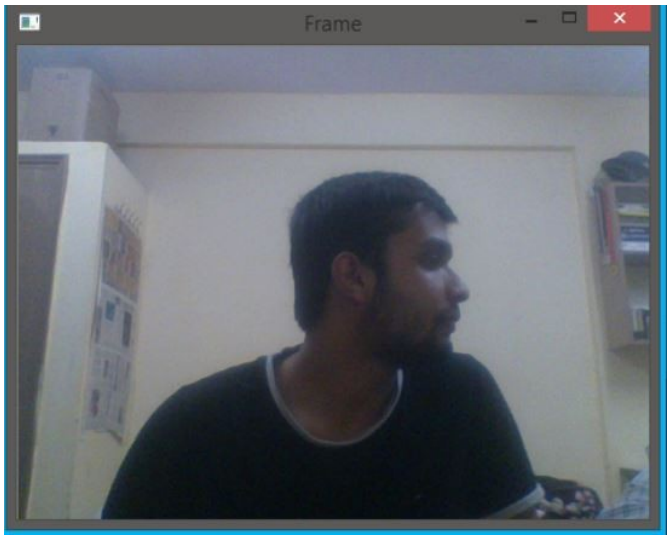
Sample set 2





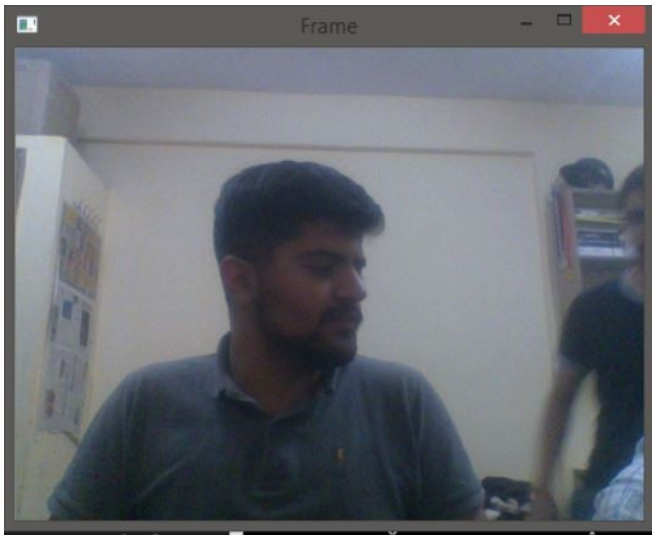


Sample set 3


Frame


Frame
EAR: 0.28

Sample set 4



Sample set 5


Frame

6.3 Table and Analysis: -

Numerous examples with shifting exactnesses were assembled and consequently a table plotted for them

I/p	Eyes Detection Accuracy	Drowsiness Accuracy
Sample 1	100%	87.5 %
Sample 2	95%	100%
Sample 3	80%	62.5%
Sample 4	100%	87.5%
Sample 5	100%	100%
TOTAL	95%	87.5%

Table2.Result table

Every individual who volunteered for the test will be approached to squint multiple times and act languid multiple times amid the test procedure. The eye squinting exactness was determined by beneath referenced recipe

Eye Detection Accuracy = total number of times eyes detected /
(total no. of eyes detected+ total no of times eyes not detected)

Drowsiness Detection Accuracy = total no. of times alarm sounds /
(total no. of times alarm sounds + total no of times alarm didn't sound)

Face or eyes sometimes might not be detected due to lack of ample ambient light. It will in general be seen from the above table that in case model 3 isn't mulled over, at that point the framework has an accuracy of about 100%. That said; the high proportion of disappointments in test 3 exhibits that the framework is slanted to botch and has certain obstacles. In test 3 we didn't utilize ample backdrop lights for the webcam. The subsequent poor lighting conditions gave a very error prone result.

6.4 Limitations: -

1. Dependency on proper ambient light:-

With poor lighting conditions once in a while the framework is unfit to perceive the eyes. So it gives a wrong result which must be managed. Continuously

circumstance infrared setting enlightenments should be used to repel from poor lighting conditions.

2. An optimum range is required: -

Exactly when the division among face and webcam isn't at perfect range then certain issues develop. Exactly when face is unreasonably close webcam (less than 25 cm), then the framework is unfit to perceive the face from the image. Right when face is a long way from the webcam (more than 80cm) by then the setting light is missing to edify the face fittingly. So eyes are not related to high precision which results in botch in recognizable proof of sluggishness. This issue isn't truly considered as progressively circumstance, the partition between driver's face and webcam is perfect so the issue never develops.

3. Orientation of face: -

At the point when the face is tilted to a specific degree it will in general be perceived, anyway past this the framework can't identify the face. So when the face isn't recognized, eyes are also not distinguished.

4. Problem with multiple faces: -

In case more than one face is recognized by the webcam, at that point framework gives an incorrect result. This issue isn't huge as we have to recognize the tiredness of a solitary driver

5. Poor detection of a person's eyes with spectacles: -

At the point when the driver wears glasses the system may not detect eyes which is the most noteworthy disadvantage of these systems. This issue has not yet been settled and is a test for practically all eye detection systems structured up until now.

CHAPTER 7. CONCLUSION

Thus we would have successfully designed and developed partial implementation of the Driver Drowsiness Detector using Python and OpenCv along with the a cam to detect the face.
The system to be developed is to be tested and limitations are identified. T he rest of the work will be done according to what is planned already.

Future Scope

This framework can be stretched out further to have abundant security highlights, for example, just a

certain no of individuals can have specialist get to or work the vehicle.

If there should be an occurrence of an endeavor to robbery, the vehicle's motor don't begin or an alarm sounds.

A picture of the burglar is taken in an attempted theft &sent to the owner of the vehicle who can register a case against the thief of the vehicle.

References

[1]. Ameratunga.S , Bailey.J, Connor.J, Civil.I, Dunn.R , Jackson.R , Norton.R, and Robinson.E, –Driver sleepiness and risk of serious injury to car occupants: Population control study.‖ *British Medical Journal*, vol. 324, 2002, pp. 1125–1129.

[2]. Bronte.S, Bergasa.L, Delgado.B, Garcia.I, Hernandez.N and Sevillano.M, –Vision-based drowsiness detector for a realistic driving simulator,‖ in *IEEE Intelligent Transportations Systems Conference (ITSC)*, 2010.

[3]. Distante.A, D'Orazio.T, Guaragnella.C and Leo.M, –A visual approach for driver inattention detection,‖ *Pattern Recogn.*, vol. 40, no. 8, 2007, pp. 2341–2355.

[4]. Bradski.G, Kaehler.A, -Learning OpenCV, *O'Reilly,* 2008

[5]. Igarashi.K ,Itou.K, Itakura.F, Miyajima.C, Ozawa.T, Takeda.K and Wakita.T, –Driver identification using driving behavior signals,‖ *IEICE - Trans. Inf. Syst.*, vol. E89- D, 2006.

[6]. Nakano.T, Suzuki.M, Yamamoto.N, Yamamoto.O and Yamamoto.S, –Measurement of driver's consciousness by image processing a method for presuming driver's drowsiness by eye- blinks coping with individual differences.‖ *Systems, Man and Cybernetics*, vol. 4, 2006