A Project report on

DRIVER'S DROWSINESS DETECTION SYSTEM

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology
In
Computer Science and Engineering

Submitted by

P.Amogh	**(19H51A0561)**
G.Keerthi	**(19H51A0572)**
K.Sri Harsha	**(19H51A0573)**
K.Vamshi Krishna	**(19H51A0574)**

Under the esteemed guidance of

(Major Dr. V A Narayana)

(PROFESSOR OF CSE AND PRINCIPAL AT CMRCET)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC & JNTUH , Approved by AICTE, Permanently Affiliated to JNTUH, Accredited by NBA.)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Mini Project-1 report entitled **"DRIVER'S DROWSINESS DETECTION SYSTEM"** being submitted by **P.Amogh (19H51A0561),G.Keerthi (19H51A0572), K.Sri Harsha (19H51A0573), K.Vamshi Krishna** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Major Dr. V A Narayana
Professor and Principal
Dept. of CSE

Dr. K. Vijaya kumar
Professor and HOD
Dept. of CSE

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Road traffic accidents, due to driver fatigue, tend to inflict high mortality rates comparing with accidents involving rested drivers. Currently there is an emerging automotive industry trend towards equipping vehicles with various driver-assistance technologies. The main purpose of this is the development and implementation of a system capable to detecting and alert, in real-time, the driver's level of fatigue. A system like this is expected to make the driver aware of the assumed danger when his level of driving and taking decisions are reduced and is indicating a sleep break as the necessary approach. By monitoring the state of the human eyes, it is assumed that the signs of driver fatigue can be detected early enough to prevent a possible road accident, which could result in severe injuries or ultimately, in fatalities. Hence, in this work we are focus on the video monitoring of the driver face, especially on his eyes position in time, when open or closed, using a machine learning object detection algorithm.

The proposed system manages to alert if the eyes of the driver are being kept closed for more than a certain amount of time by triggering a set of warning lights and sounds. The large-scale implementation of this type of system will drop the number of road accidents caused by the driver’s fatigue, thus saving countless lives and bringing a reduction of the socio-economic costs associated with these tragic events.

1. INTRODUCTION

1.1 DESCRIPTION

Many of the road accidents will occur due to drowsiness of thedriver. Driver fatigue is a significant factor in a large number of vehicle accidents. Drowsiness can be detected by monitoring the driver through continuous video stream with a mobile or camera.The general objective is to create a model that will indicate whether a person is feeling drowsy or not. The model takes image for every second and checks for eye blinking and mouth opening measures which helps in calculating the time for eye closed by perclos algorithm and the mouth aspect ratio. If the blinking is high with eyes is closed for a certain amount of time and the mouth aspect ratio is higher then usual due to yawning then it will indicate the driver through a sound.

1.2 PROBLEM STATEMENT

On road driver's fatigue and drowsiness is contributing more than 30% of reported road accidents. Driver drowsiness can be estimated by monitoring biomedical signals ,visual assessment of driver's bio-behavior from face images , by Rmonitoring drivers performance or by combines all above techniques. Proposed algorithm is based on live monitoring of EAR(Eye aspect Ratio) by application of Image processing. HD live video is decomposed in continues frames and facial landmarks has been detected using pre trained Neural Network based on Dlib functions. Dlib functions are trained using Cascading algorithm.Source Image processing libraries (OPEN CV) is used as primary Image processing tool.Python Language is used as main codding language.

1.3 PROJECT PURPOSE

Real Time Drowsiness behaviors which are related to fatigue are in the form of eye closing,head nodding or the brain activity. Hence, we can either measure change in physiological signals, such as brain waves, heart rate and blinking to monitor drowsiness or consider physical changes such as sagging leaning of driver's head and open/closed state of eyes.The former technique,while more accurate, is not realistic since highly sensitive electrodes would have to be attached directly on the

driver' body and hence which can be annoying and distracting to the driver. In addition, long time working would result in perspiration on the sensors, diminishing their ability to monitor accurately. The second technique is to measure physical changes (i.e. open/closed eyes to detect fatigue) is well suited for real
world conditions since it is non-intrusive by using a video camera to detect changes. In addition, micro sleeps that are short period of sleeps lasting 2 to 3 minutes are good Indicators of fatigue. Thus, by continuously monitoring the eyes of the driver one can detect the sleepy state of driver and a timely warning is issued.
Driver in-alertness is an important resulting from sleep deprivation or sleep disorders and is an important factor in the increasing number of the accidents on today's roads. Drowsy driver warning system can form the basis of the system to possibly reduce the accidents related to driver's drowsiness.The computer vision basedtechniques from the second category are particularly effective, because the drowsiness can be detected by observing the facial features and visual bio-behavior such as head position, gaze, eye openness, eyelid movements, and mouth openness. Proposed algorithm is based on computer vision method. The main focus is on the detection of blinks by estimating the EAR(Eye aspect Ratio). This is achieved by monitoring the eyes of the driver throughout the entire video sequence.

1.4 OBJECTIVES

To develop a system that is able to detect drowsiness of a driver.

To Improve the accuracy of existing systems by monitoring both eyes and mouth to calculate eye aspect ratio and distance between lips and warn the driver if the driver is drowsy or yawning.

1.5 MODULES DESCRIPTION

1.5.1 DATA COLLECTION

Data plays the most important role in this project. The data has to be genuine and trustworthy. Data has to be collected from the right sources and each record must have all the necessary fields to perform the analysis. It can either be procured from online sources or collected manually to attain more accuracy, but is a very tedious task. Once data is collected, it needs to be verified for relevance before finalizing as valid data for the project.

1.5.2 APPLYING ALGORITHMS

This module has two parts to it- identifying the algorithms and applying the algorithms.Identification plays a key role in this process and in the project as a whole. Identifying the right algorithm may be done through research and mostly trial and error. We pick the algorithms which yield the most accurate results and zero in on them. So based on testing we eliminated few algorithms and finally decided to rely on the cnn algorithm.

1.5.3 VERIFICATION/TESTING PHASE

In this module we just test lively after completion of project.Here we don’t need a data set since we are capturing the live video and doing the process on it. For verification we can do a false test and positive test and check whether is true for different positions of the user.

2.BACKGROUND WORK

2.1 EXISTING SOLUTION

2.1.1. IoT-Based Smart Alert System for Drowsy Driver Detection:

This model has a camera monitoring the driver’s eye blinking, eye closure, face detection, head posture, etc. with face landmark algorithm and Euclidean distance in the behavioral-based approach. in addition to monitoring the intensity of the collisions impacts, it is also keeps the records of the location for taking supportive action.

Figure *2.1.1: IoT-Based Smart Alert System for Drowsy Driver Detection*

2.1.2. Karolinska Sleepiness Scale Detection based on Sensors:

This model evaluates the level of drowsiness based on the driver’s personal estimation and many tools that have been used to translate this rating to a measure of driver drowsiness. the KSS ratings of drivers for every 5 min and used it as a reference to the EoG signal collected. It has a nine-point scale that has verbal anchors for each step.

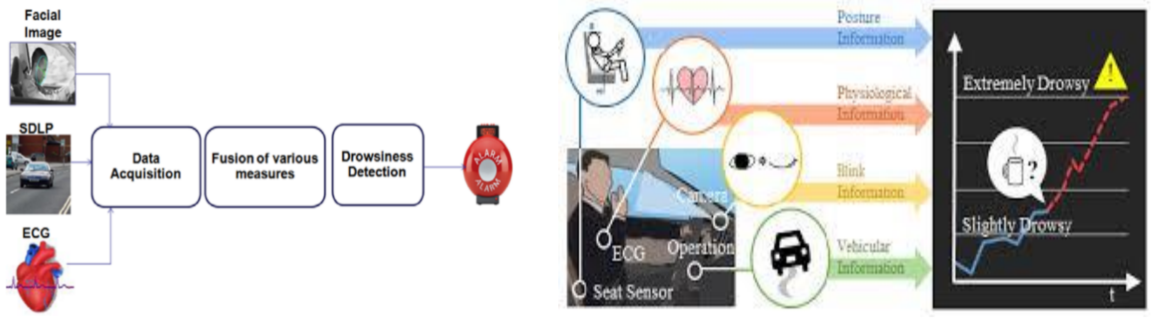


Figure 2.1.2:Karolinska Sleepiness Scale Detection based on Sensors

2.1.3. Dynamic Bayesian Network Model:

This model is done in a probabilistic way using various physiological and contextual information. Electroencephalogram (EEG), Electromyogram (EMG), and respiration signals were simultaneously recorded by wearable sensors and sent to computer by Bluetooth during the real driving. From these physiological information, fatigue likelihood can be achieved using kernel distribution estimate at different time sections.

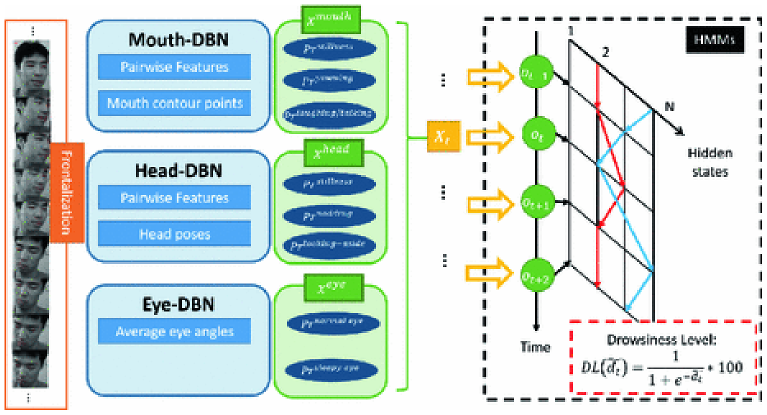


Figure 2.1.3. Dynamic Bayesian Network Model:

2.2 Disadvantages of the existing models:

- Lower accuracy due to features dependency chain.
- Difficult to see significant differences between tired and alert drivers.
- Road geometry influences the results.
- Very high sensitivity to changes in illumination conditions.
- Templates becoming mismatched when global minimum fit cannot be found for the model.

2.3 TECHNOLOGIES

Various technologies can be used to try to detect driver drowsiness.

Steering pattern monitoring

Primarily uses steering input from electric power steering system. Monitoring a driver this way only works as long as a driver actually steers a vehicle actively instead of using an automatic lane-keeping system.

Vehicle position in lane monitoring

Uses lane monitoring camera. Monitoring a driver this way only works as long as a driver actually steers a vehicle actively instead of using an automatic lane-keeping system.

Driver eye/face monitoring

Uses computer vision to observe the driver's face, either using a built-in camera[5] or on mobile devices.

Physiological measurement

Requires body sensors to measure parameters like brain activity, heart rate, skin conductance, muscle activity.

3.PROPOSED SYSTEM

3.1 Face and Eye Detection by OpenCV Algorithms

In this paper a novel approach to critical parts of face detection problems is given, based on analogic cellular neural network (OpenCV) algorithms. The proposed OpenCValgorithms find and help to normalize human faces is, effectively while cause for most accident related to the vehicles crashes. Driver fatigue their time requirement is a fraction of the previously used methods.The algorithm starts with the detection of heads on colour pictures using deviations in colour and structure of the human face and that of the background.By normalizing the distance and position of the reference points, all faces should be transformed into the same size and position.For normalization, eyes and mouth serve as point reference. Other OpenCV algorithm finds the eyes and mouth on any grayscale image by searching characteristic is features of the eyes,mouth and eye,mouth sockets.Tests made on a standard database show that the algorithm works very fast and it is reliable.In proposed method, first the image is acquired by the webcam for processing. The images of the driver are captured from the camera which is installed in front of the driver on the car dashboard. It will be passed to preprocessing which prepares the image for further processing by the system.Then we search and detect the faces in each individual frame. If no face is detected then another frame is acquired. If a face is detected, then a region of interest in marked within the face. This region of interest contains the eyes.After that the eyes are detected from the region of interest. If an eye is detected then there is no blink and the blink counter is set to "20‟. If the eyes are closed in a particular frame, then the blink counter is decremented and a blink is detected. When the eyes are closed for more than 20 frames then it is deducible that the driver is feeling drowsy.Hence drowsiness is detected and an alarm sounded.Next as an alternative we check the mouth distance ratio it is determined if the frequency is greater than 20 we raise alarm otherwise whole process is repeated.After that the whole process is repeated as long as the driver is driving the car. The overall flowchart for drowsiness detection system is shown.

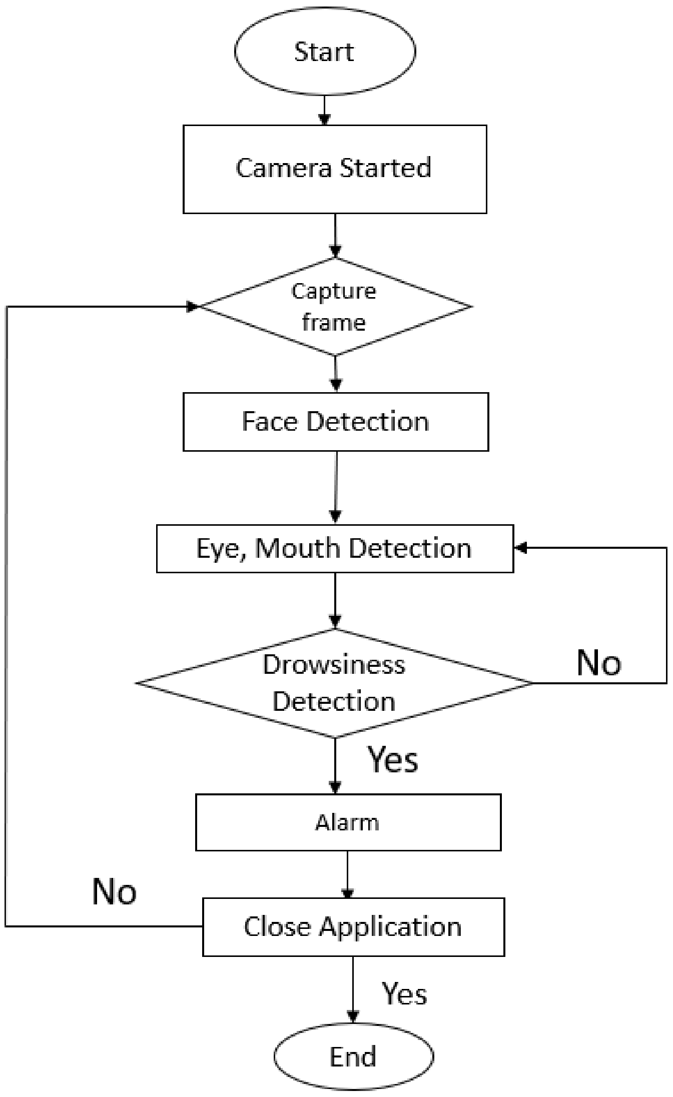


Figure 3.1. Flowchart of Driver Drowsiness Detection System

3.2 Face Detection

Face detection is accomplished by the OpenCV algorithm proposed by Paul Viola and Michael Jones in 2001. Due to the complex background, it is not a good choice to locate or detect both the eyes in the original image, for this we will take much more time on searching the whole window with poor results. So firstly, we will locate the face, and reduce the range in which we will detect both the eyes. After doing this we can improve the tracking speed and correct rate, reduce the affect of the complex background. Besides,we propose a very simple but powerful method to reduce the computing complexity.

i)OpenCV :The simple features used are suggestive approaches of OpenCV basis functions which have been used by Papageorgiouetal.A OpenCV-like feature considers affixed rectangular regions at a specific part in a detection window; each OpenCV like feature expressed by two or three jointed black and white. The value of a OpenCV like feature is the difference between the sums of the pixel values within the black and white rectangular regions. These sums are used to find the difference between regions. Then the differences can be used to classify the sub region of an image.

ii) *Cascaded classifier*:The cascade classifier consists of number of stages, where each stage is a collection of weak learners. The weak learners are simple classifiers known as decision stumps. Boosting is used to train the classifiers. It provides the ability to train a highly accurate classifier by taking a weighted average of the decisions made by the weak learners.Each stage of the classifier shows the region defined by the current location of the sliding window as either positive or negative. Positive indicates an object was found and negative indicates no object. If the label is negative, the classification of this region is complete, and the detector shifts the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive. It is used to eliminate less likely regions quickly so that no more processing is needed. Hence, the speed of overall algorithm is increased.

3.3 System Description

3.3.1 Eye,Mouth Detection

The facial landmark detector implemented inside dlib produces 68 *(x, y)*-coordinates that map to *specific facial structures*. These 68 point mappings were obtained by training a shape predictor on the labeled iBUG 300-W dataset.

Below we can visualize what each of these 68 coordinates map to:

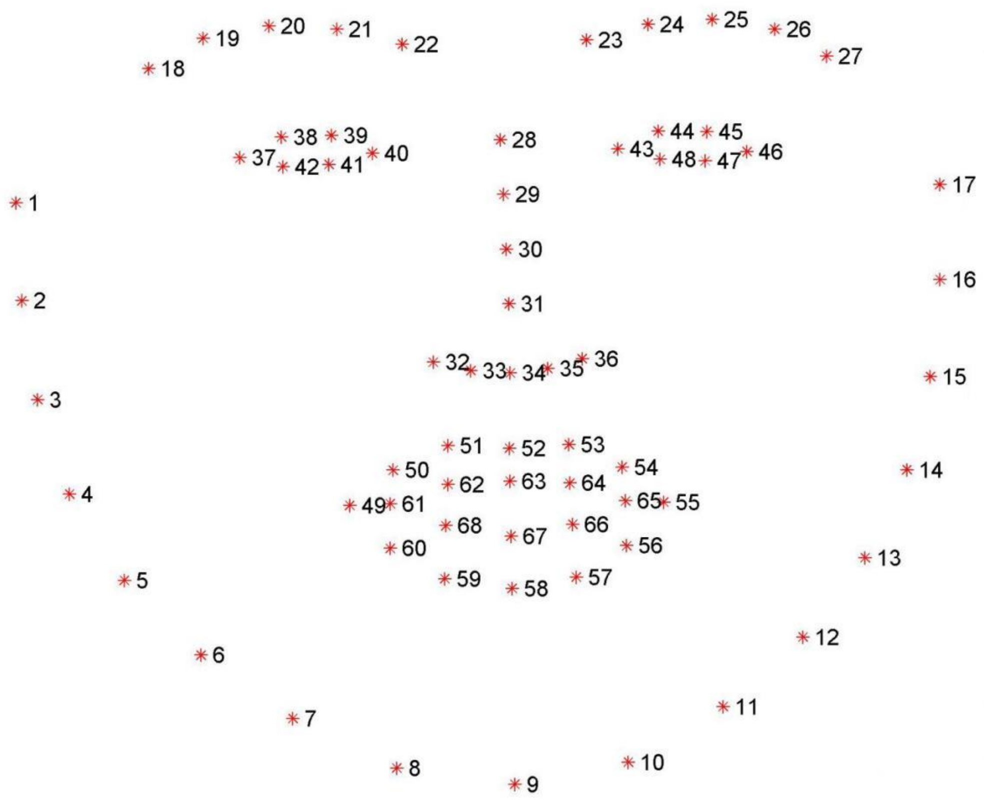


Figure 3.3.1: Facial landmarks

Examining the image, we can see that facial regions can be accessed via simple Python indexing (assuming zero-indexing with Python since the image above is one-indexed):

- The *mouth* can be accessed through points *[48, 68]*.
- The *right eyebrow* through points *[17, 22]*.
- The *left eyebrow* through points *[22, 27]*.
- The *right eye* using *[36, 42]*.
- The *left eye* with *[42, 48]*.

- The *nose* using *[27, 35]*.
- And the jaw via *[0, 17]*.

These mappings are encoded inside the FACIAL_LANDMARKS_IDXS dictionary inside face_utils of the imutils library:

3.3.2 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 the eyes 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 is 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 nonblinking patterns.

3.3.3 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 computed.

EAR = ||p2 − p6|| + ||p3 − p5|| (1)

2||p1 − p4||

where p1, . . ., p6 are the 2D landmark locations, depicted in Fig. 1. 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.

3.3.4 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”. 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 the eyes 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.

3.3.5 Drowsiness Detection:

The last step of the algorithm is to determine the person's condition based on a pre-set 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 a 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.

3.4 Technology Used

3.4.1 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.

3.4.2 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,4 image processing, image proc. 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 meths.

3.5 TOOLS AND IMAGE PROCESSING LIBRARIES

Following optimized tools and image processing libraries are used by author for implementation of presented algorithm. Open CV: OpenCV (Open-source Computer Vision) is the Swiss Army knife of computer vision.

It has a wide range of modules that can help us with a lot of computer vision problems. But perhaps the most useful part of OpenCV is its architecture and memory management. It provides you with a framework in which you can work with images and video in any way you want, using OpenCV's algorithms or your own, without worrying about allocating and reallocating memory for your images. Open CV libraries and functions are highly optimized and can be used for real time image and video processing. OPENCV's highly optimized image processing function are used by author for real time image processing of live video feed from camera.

DLib: Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open source licensing allows you to use it in any application, free of charge. Open Source Dib library is used by author for implementation of CNN(Neural Networks). Highly optimized Pre-learned facial shape predictor and detectors functions are used by author for detectionof facial landmarks. Facial landmarks were further used for extracting eye coordinates.

Python: Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping inthepres fcomplex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc. Python is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability. Python language is used by author due to his cross platform compatibility as main coding language for algorithm. Open CV and Dlib libraries are integrated in python interpreter for using readymade optimized functions.

3.6 Hardware Requirements

1.Processor Intel or high level

2.RAM 2 GB

3.Space on disk Min 2 GB

4.Device Any device that has internet access

5.Execution space Min 100mb

3.7 Sofrware Requirements

1.Python3,pycharm

2.Open cv

3.Libraries like imutils,Scipy,face recognition,numpy,argparse,thread,dlib,wincom

4.DESIGNING

This projected is designed to achieve certain goals which are:

λ Driver drowsiness detection is a car safety technology which helps to save the life of the driver by preventing accidents when the driver is getting drowsy.

λ The main objective is to first design a system to detect driver’s drowsiness by continuously monitoring retina of the eye.

λ The system works in spite of driver wearing spectacles and in various lighting conditions.

λ To alert the driver on the detection of drowsiness by using buzzer or alarm.

λ Speed of the vehicle can be reduced.

λ Traffic management can be maintained by reducing the accidents.

The drowsiness detection and correction system developed is capable of detecting drowsiness in a rapid manner. The system which can differentiate normal eye blink and drowsiness which can prevent the driver from entering the state of sleepiness while driving. The system works well even in case of drivers wearing spectacles and under low light conditions also. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for about two seconds, the alarm beeps to alert the driver and the speed of the vehicle is reduced. By doing this many accidents will reduced and provides safe life to the driver and vehicle safety. A system for driver safety and car security is presented only in the luxurious costly cars. Using drowsiness detection system, driver safety can be implemented in normal cars also.Once the eye aspect ratio calculated, algorithm can threshold it to determine if a person is blinking the eye aspect ratio will remain approximately constant when the eyes are open and then will rapidly approach zero during a blink, then increase again as the eye opens. The duration of blink further provide estimation of microsleep. The proposed algorithm has been tested on personal car driver for testing purposes. For authentic results, the camera position was focused on the driver’s face. Further, the algorithm has been tested in day time driving and Night time using IR camera. The results are discussed in Result section and found satisfactory. The proposed algorithm focused solely on using the eye aspect ratio as a quantitative metric to determine if a person has blinked in a video.

4.1 Sequence Diagram

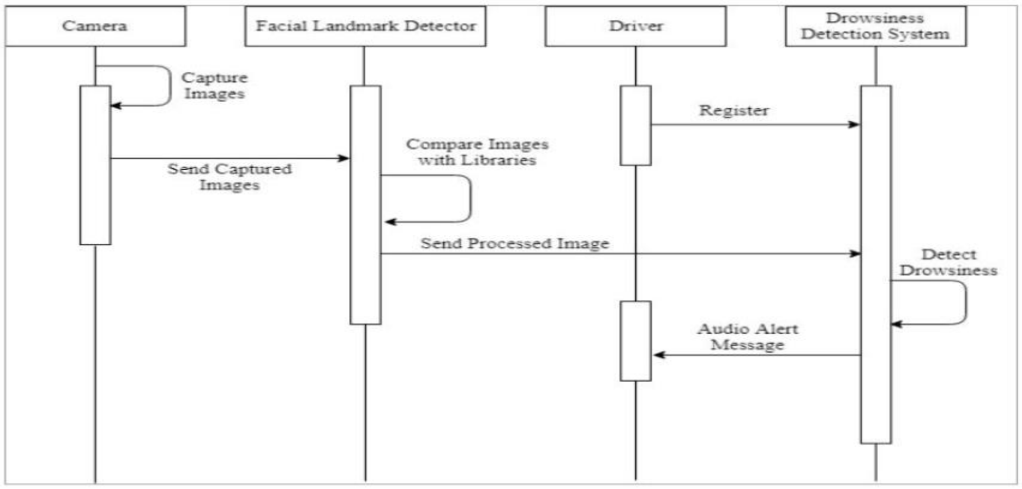


Figure 4.1:Sequence Diagram

1. Camera : Captures the video o f the driver when the vehical is in motion and sends the captured video to the storage unit.
2. Facial Landmark Detector : Detects the drivers facial changes and reports then.
3. Driver : The person who is driving the vehicle.
4. Drowsiness Detection Sysytem : The Image capturing mechanism where sequence of images are processed using various algorithms to detect whether the driver is fatigue or not.

4.2Data Flow Diagram

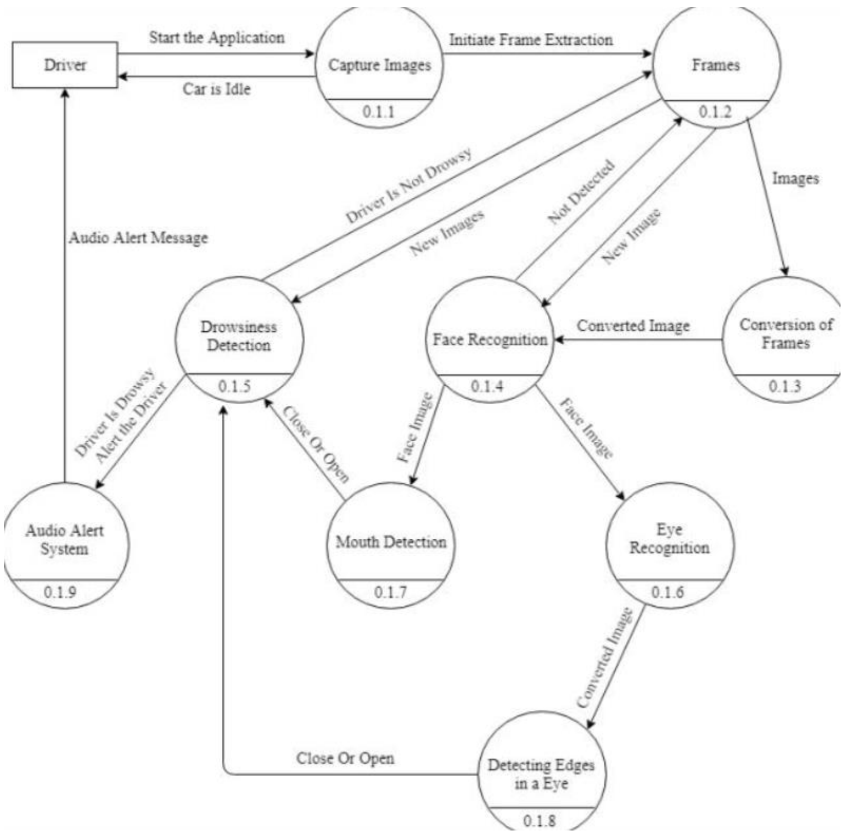


Figure 4.2: Data Flow Diagram

The first step in building a blink detector is to perform facial landmark detection to localize the eyes in a given frame from a video stream. The eye aspect ratio for each eye can be calculated using Euclidian distance functions of OPEN CV which is a singular value, relating the distances between the vertical eye landmark points to the distances between the horizontal landmark points. Once the eye aspect ratio calculated, algorithm can threshold it to determine if a person is blinking the eye aspect ratio will remain approximately constant when the eyes are open and then will rapidly approach zero during a blink, then increase again as the eye opens. The duration of blink further provide estimation of microsleep.

5.RESULT AND DISSCUSION

5.1 IMPLEMENTATION

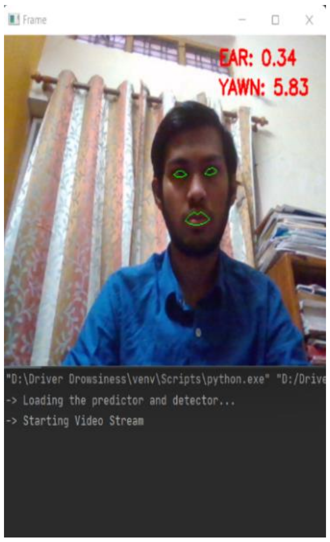
a) MAIN CODE:

```
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 argparse
import imutils
import time
import dlib
import cv2
import os


def alarm(msg):
    global alarm_status
    global alarm_status2
    global saying

    while alarm_status:
        print('call')
        s = 'espeak "' + msg + '"'
        os.system(s)

    if alarm_status2:
        print('call')
        saying = True
        s = 'espeak "' + msg + '"'
        os.system(s)
        saying = False


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


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

    leftEye = shape[lStart:lEnd]
    rightEye = shape[rStart:rEnd]

    leftEAR = eye_aspect_ratio(leftEye)
    rightEAR = eye_aspect_ratio(rightEye)

    ear = (leftEAR + rightEAR) / 2.0
    return (ear, leftEye, rightEye)


def lip_distance(shape):
    top_lip = shape[50:53]
    top_lip = np.concatenate((top_lip, shape[61:64]))

    low_lip = shape[56:59]
    low_lip = np.concatenate((low_lip, shape[65:68]))

    top_mean = np.mean(top_lip, axis=0)
    low_mean = np.mean(low_lip, axis=0)

    distance = abs(top_mean[1] - low_mean[1])
    return distance


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



EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 30
YAWN_THRESH = 20

alarm_status = False
alarm_status2 = False
saying = False

COUNTER = 0

print("-> Loading the predictor and detector...")
```

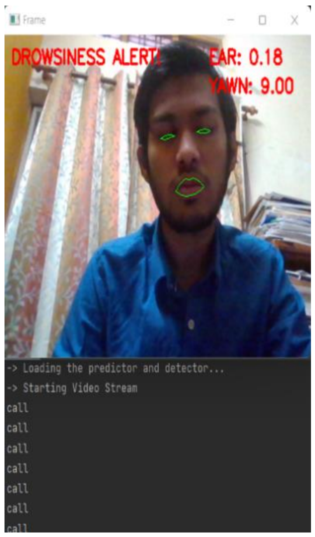
```
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")  #
    Faster but less accurate
predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

print("-> Starting Video Stream")
vs = cv2.VideoCapture(0)
vs.set(3,450)
vs.set(4,450)



while True:

   ret, frame = vs.read()
   frame = imutils.resize(frame, width=450)
   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

   rects = detector.detectMultiScale(gray, scaleFactor=1.1,
                          minNeighbors=5, minSize=(30, 30),
                          flags=cv2.CASCADE_SCALE_IMAGE)



   for (x, y, w, h) in rects:
      rect = dlib.rectangle(int(x), int(y), int(x + w), int(y + h))

      shape = predictor(gray, rect)
      shape = face_utils.shape_to_np(shape)





 eye = final_ear(shape)
      ear = eye[0]
```

```
leftEye = eye[1]
rightEye = eye[2]

distance = lip_distance(shape)

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)

lip = shape[48:60]
cv2.drawContours(frame, [lip], -1, (0, 255, 0), 1)

if ear < EYE_AR_THRESH:
    COUNTER += 1

    if COUNTER >= EYE_AR_CONSEC_FRAMES:
        if alarm_status == False:
            alarm_status = True
            t = Thread(target=alarm, args=('wake up sir',))
            t.deamon = True
            t.start()

        cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

else:
    COUNTER = 0
    alarm_status = False

if (distance > YAWN_THRESH):
    cv2.putText(frame, "Yawn Alert", (10, 30),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
              if alarm_status2 == False and saying == False:
                 alarm_status2 = True
                 t = Thread(target=alarm, args=('take some fresh air sir',))
                 t.deamon = True
                 t.start()




        else:
              alarm_status2 = False

            cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
                      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            cv2.putText(frame, "YAWN: {:.2f}".format(distance), (300, 60),
                      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

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

          if key == ord("q"):
             break

        cv2.destroyAllWindows()
           vs.stop()
```

5.2Results



```
"D:\Driver Drowsiness\venv\Scripts\python.exe" "D:/Drive
-> Loading the predictor and detector...
-> Starting Video Stream
```



```
-> Loading the predictor and detector...
-> Starting Video Stream
call
call
call
call
call
call
call
```

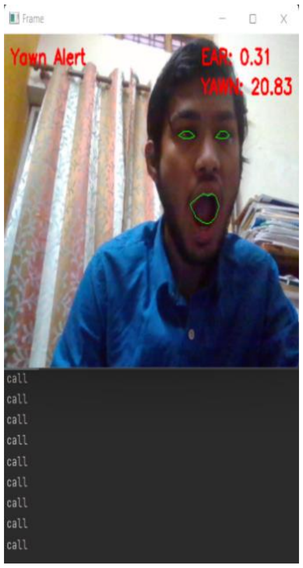
Fig5.2.1:A Testcase Where the driver is

Where the driver is Drowsy

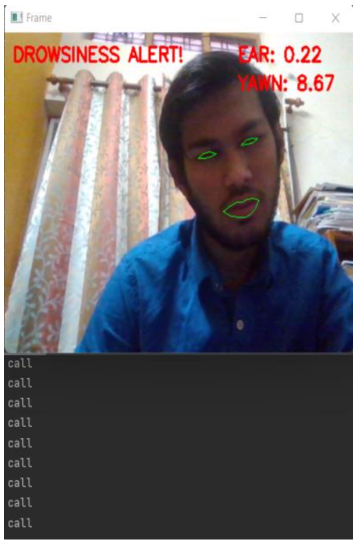
yawning

Fig 5.2.2:A Testcase

neither drowsy nor



```
call
call
call
call
call
call
call
call
call
```



```
call
call
call
call
call
call
call
call
call
```

Fig 5.2.3:A Testcase Where the

Driver is driver is yawning

Fig 5.2.4:A Testcase Where the

Drowsy but at an angle

6.CONCLUSION AND FUTUREWORK

This project tells how to build a drowsiness detector using OpenCV, Python, and Dlib opensource Libraries. The first step in building a blink detector is to perform facial landmark detection to localize the eyes in a given frame from a video stream. The eye aspect ratio for each eye can be calculated using Euclidian distance functions of OPEN CV , which is a singular value, relating the distances between the vertical eye landmark points to the distances between the horizontal landmark points. Once the eye aspect ratio calculated, algorithm can threshold it to determine if a person is blinking the eye aspect ratio will remain approximately constant when the eyes are open and then will rapidly approach zero during a blink, then increase again as the eye opens. The duration of blink further provide estimation of microsleep. The drowsiness detection system developed is capable of detecting drowsiness in a rapid manner. The system which can differentiate normal eye blink and drowsiness in addition to yawning which can prevent the driver from entering the state of sleepiness while driving. The system works well even in case of drivers wearing spectacles and under low light conditions also.During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for about four to six seconds or yawn, the alarm beeps to alert the driver. By doing this many accidents will reduced and provides safe life to the driver and vehicle safety.

7.REFERENCES

a. Detecting Driver Drowsiness Based on Sensors: A Review (nih.gov)

b. IoT-Based Smart Alert System for Drowsy Driver Detection (hindawi.com)

c. Drowsiness Detection with Machine Learning | by Grant Zhong | Towards Data Science

d. Association for Safe International Road Travel (ASIRT), Road Crash Statistics.http://asirt.org/initiatives/informingroadusers/road-safety-facts/road-crash-statistics, 2016

e. https://en.wikipedia.org/wiki/Microslee

f. Journal of VLSI Signal Processing 23, 497–511 (1999) c °1999 Kluer Acadmic Publishers.Manufactured in The Netherlands.

g. https://docs.opencv.org/trunk/d7/d8b/tutorial_py_face_de tection.html

h. Eye Detection Using Morphological and Color Image Procesing Tanmay Rajpathaka, Ratnesh Kumar and Eric Schwartzb

i. A Robust Algorithm for Eye Deteuction on Grey Intensity Face without Spectacles- JCS&T Vol. 5 No. 3