



VIT[®]

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

DRIVER DROWSINESS DETECTION SYSTEM

REVIEW REPORT

Submitted by

SUYASH AGARWAL
(18BCE0273)

Prepared For

**IMAGE PROCESSING (CSE4019) – PROJECT
COMPONENT**

Submitted To

PROF. SWATHI J.N.

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Acknowledgement

It gives me immense pleasure to offer my sincere thanks to my project guide and mentor Dr. Swathi J.N. It would not have been possible to accomplish the milestones of the project without her invaluable academic support, meticulous scrutiny, technical supervision and timely advice.

I take this opportunity to express my deep sense of gratitude to all those who helped me in completion of this project, directly or indirectly. In this regard, I heartily thank my parents for their relentless support and encouragement.

Suyash Agarwal

18BCE0273

Executive Summary

Truck drivers and other people who spend most of their time driving on highways are prone to feeling sleepy and drowsy during the journey. This can lead to dangerous accidents and a possible loss of life and property. To prevent this, I propose an alarm system which utilises a driving camera which detects with the help of image processing if the driver is feeling sleepy or is not fully focused on driving. If the camera detects an anomaly, the alarm goes off and the vehicle indicators turn on to alert other vehicles driving nearby. I plan to train a neural network model which detects drowsiness and deploy it in a camera feed for continuous detection.

Table of Contents

Title	Page No.
Acknowledgement	2
Executive Summary	3
Table of Contents	4
List of Figures	5
1. Introduction	6
1.1 Objective	
1.2 Motivation	
1.3 Background	
2. Project description and goals	8
3. Technical Specification	9
4. Design approach and details	11
5. Schedule tasks and milestones	13
6. Project Demonstration	13
7. Cost Analysis/Result and Discussion	15

List of Figures

Figure	Page No.
Project Block Diagram	8
OpenCV logo	9
SciPy logo	9
Haar Feature examples	11
Haar features for various parts of face	11
Eye Aspect Ratio Calculation	12
Code Screenshot 1	13
Code Screenshot 2	14
Output Screenshot	14

1. Introduction

1.1 Objective

The objective of this project is to create a device that can be installed inside the dashboard of heavy-duty vehicles such as trucks, so that they can monitor the drowsiness levels of the truck driver in order to ensure that the driver is properly concentrating on his/her driving and is not feeling drowsy or sleepy which may lead to accidents resulting in loss of life and property. This device takes live camera feed of the driver's face and uses image processing to gauge whether the driver is feeling lazy by monitoring their eye span. If the system finds eye span below the permissible threshold, it raises an alarm prompting the driver to take rest in order to ensure road safety

1.2 Motivation

Approximately **1.35 million** people around the world die every single year due to road traffic accidents. Of these 1.35 million, **30 percent** of them happen due to driver drowsiness or fatigue. Despite these large numbers, a proper solution has not been implemented on a wide scale to prevent this mishap. Hence, to counter to this problem, I propose this AI based alarm system that alerts the driver if it finds their eyes drooping due to fatigue and tiredness.

1.3 Background

There has been some prior research on the topic and many people have come up with interesting solutions to tackle this problem. Almost all the implementation requires image processing to gauge fatigue levels. Some use modern convolution neural networks while some use the quicker method of haar-cascades and contouring.

1.3.1 Literature Review:

a. ***T. Hong, H. Qin and Q. Sun, "An Improved Real Time Eye State Identification System in Driver Drowsiness Detection," 2007 IEEE International Conference on Control and Automation, Guangzhou, China, 2007, pp. 1449-1453, doi: 10.1109/ICCA.2007.4376601.*** – This paper proposes an real-time eye state detection system to identify driver's drowsy state. The system optimize several image processing techniques to get better performance to reach the criteria of the drowsiness detection methods. Firstly, face region is detected using the optimized Haar-like feature detection scheme; secondly, they apply horizontal projection of the detected face and geometrical position of the eye on the face to get the eye region; finally, a new complexity function with

dynamic threshold to identify the eye state. **This method is what the project is primarily based on**

b. *M. Omidyeganeh, A. Javadtalab and S. Shirmohammadi, "Intelligent driver drowsiness detection through fusion of yawning and eye closure," 2011 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems Proceedings, Ottawa, ON, Canada, 2011, pp. 1-6, doi: 10.1109/VECIMS.2011.6053857.* - In this approach, the driver's facial appearance is captured via a camera installed in the car. In the first step, the face region is detected and tracked in the captured video sequence utilizing computer vision techniques. Next, the eye and mouth areas are extracted from the face; and they are studied to find signs of driver fatigue. Finally, in a fusion phase the driver state is determined and a warning message is sent to the driver if the drowsiness is detected.

c. *L. Pauly and D. Sankar, "Detection of drowsiness based on HOG features and SVM classifiers," 2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, India, 2015, pp. 181-186, doi: 10.1109/ICRCICN.2015.7434232.* – In this paper they used a completely different approach and used Haar based cascade classifier for eye tracking and combination of Histogram of oriented gradient (HOG) features combined with Support Vector Machine (SVM) classifier for blink detection. This method had an accuracy of 91.6% according to their claim.

d. *M. Ngxande, J. Tapamo and M. Burke, "Driver drowsiness detection using behavioral measures and machine learning techniques: A review of state-of-art techniques," 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech), Bloemfontein, South Africa, 2017, pp. 156-161, doi: 10.1109/RoboMech.2017.8261140.* – This paper was basically a review of all the SOTA techniques that have been used throughout the years.

e. *Driver Drowsiness Detection System Based on Feature Representation Learning Using Various Deep Networks* by Fei Pan, Sunghun Kang, Sanghyuk Park – They used deep learning to recognise if the driver is drowsy or not.

2. Project description and goals

2.1 Block Diagram:

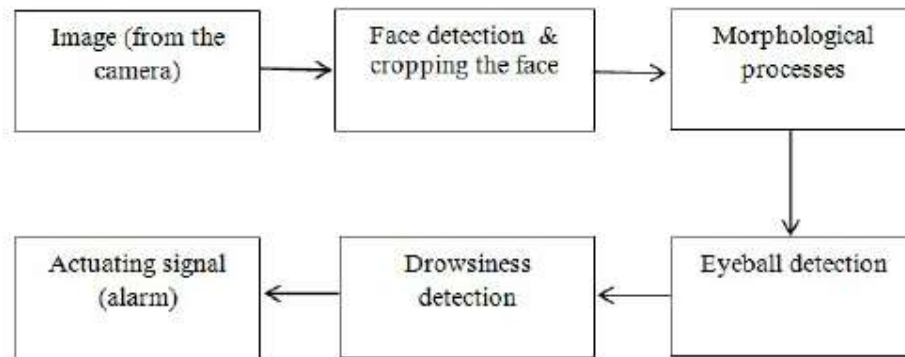


Figure 1: Project block diagram

As evident from the block diagram, a frame will be taken from the live video captured at regular intervals. The first step will be to crop the facial part from the image, after doing some morphological processes, the eyeballs will be detected. These eyeballs once detected will have some reference points marked on them which will help calculate span hence the drowsiness. If drowsiness is found then alarm is raised.

2.2 Goals:

By the end of this project, I expect to write a fully functioning code for carrying this out and make the project computationally feasible so that low level hardware e.g., IOT devices can run this with minimal lag.

3. Technical Specification

The project is implemented in python language completely. For this project many libraries have been used to implement the same as listed below:

a. *OpenCV*



Figure 2: OpenCV Logo

OpenCV (*Open-Source Computer Vision Library*) is a library of programming functions mainly aimed at real-time computer vision. The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations. This library has been primarily used in this project for capturing video and extracting frames as well as displaying the final output footage.

b. *SciPy*



Figure 3: SciPy logo

SciPy is a Python-based ecosystem of open-source software for mathematics, science, and engineering. It includes popular libraries like NumPy, Matplotlib, Pandas etc. Here in this project I have used its distance calculating functionality which can easily calculate Euclidean distances given two points.

c. *dlib*

Dlib is a toolkit for making real world machine learning and data analysis applications. In this project it has been used for frontal face detection and eyeball detection.

d. *imutils*

Imutils contains a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much more easier with OpenCV and both Python 2.7 and Python 3. In this project it has been used extract the facial landmarks predicted by dlib.

e. *playsound*

Playsound is a simple library to play mp3 files in one line of code and without any hassle. It has been used to play alarm in this project

4. Design approach and details

For the facial recognition part, I chose haar-cascades method over deep learning because of its speed and efficiency while having almost the same accuracy.

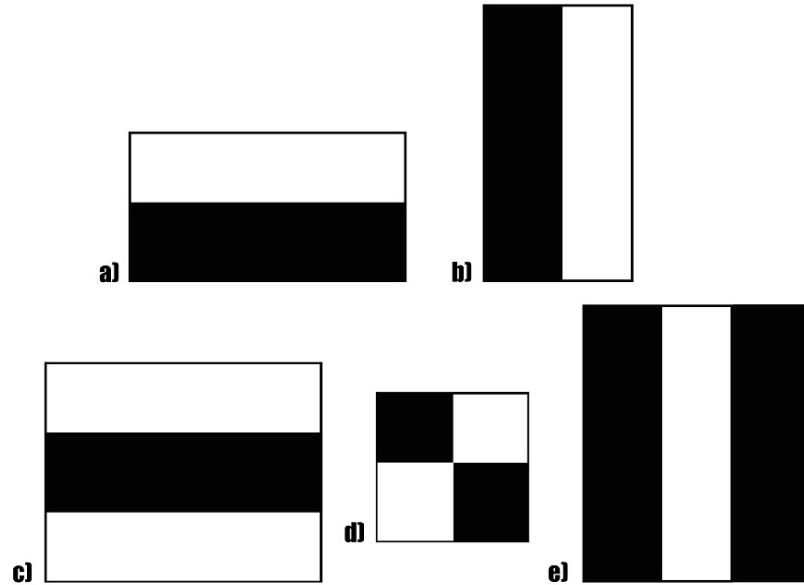


Figure 4: Haar feature examples

The objective here is to find out the sum of all the image pixels lying in the darker area of the haar feature and the sum of all the image pixels lying in the lighter area of the haar feature. And then find out their difference. Now if the image has an edge separating dark pixels on the right and light pixels on the left, then the haar value will be closer to 1. That means, we say that there is an edge detected if the haar value is closer to 1. Once the face is recognized and a region of interest is extracted, next step is to find the eyeballs.

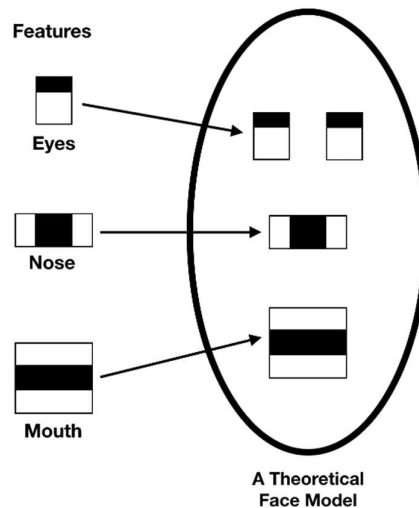


Figure 5: Haar features for various parts of face

The eyeballs are also detected using haar cascade features and extracted accordingly. Once we have the eyeball region of interest, the next step is to mark 6 reference points on the extraction of span calculations.

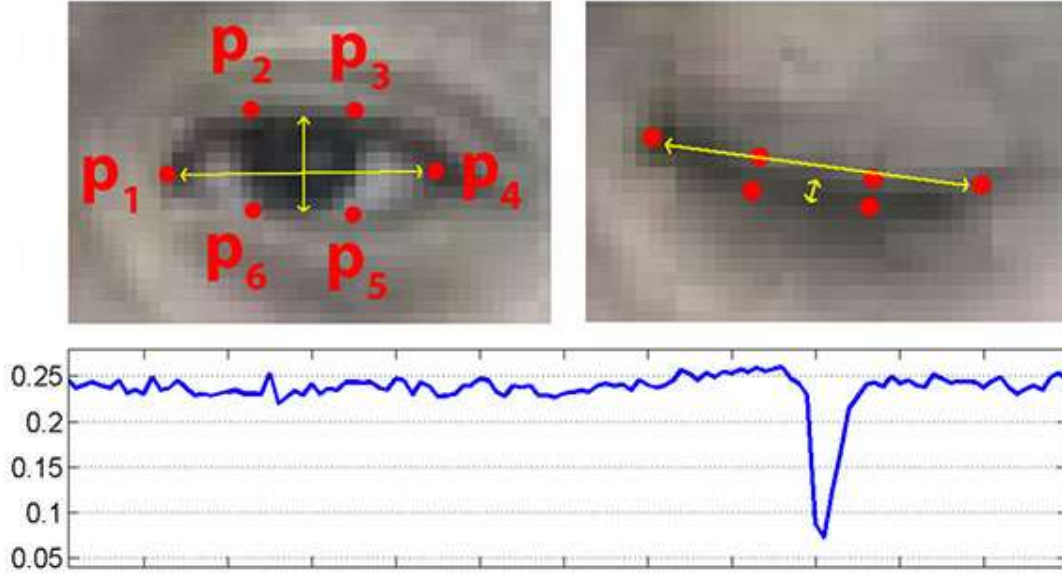


Figure 6: Eye Aspect Ratio calculation

As we can see above, the 6 reference points are marked above. Using this information, we put the coordinates of the 6 points to this equation:

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

If this Eye Aspect Ratio value falls below a threshold, in this case 0.25, the an alarm goes off alerting the driver to adequate steps like taking appropriate amount of rest before continuing the journey to avoid any mishaps.

Constraints

One constraint I faced was that during the integration of alarm to the code, audio was freezing the video capturing functionality hence every thing was lagging. To tackle this problem, I used the concept of multithreading to run both processes parallelly to avoid deadlocks.

Trade-offs

One significant tradeoff that I came across was not using raspberry pi for the demo of this project. The reason was that its camera has a very low capture rate (9fps) which is not at all feasible to use in a practical scenario.

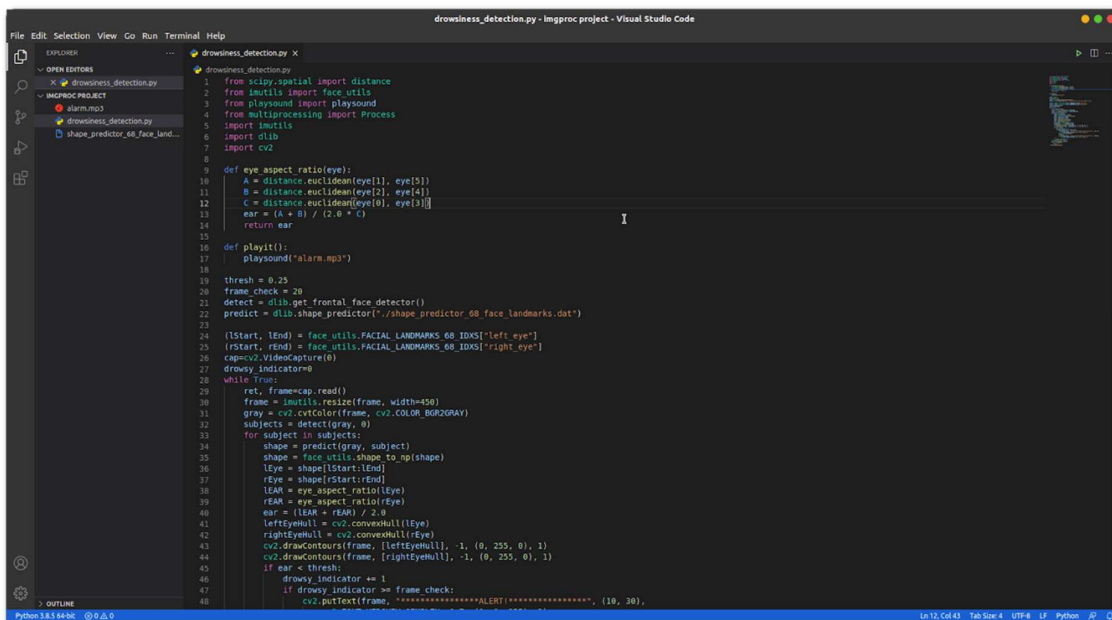
5. Schedule, Tasks and Milestones

As this is a solo project, the timeline of making this project was as follows:

1. Learning about Haar-Cascades, Eye aspect ratio and how they work
2. Learning opencv and implementing live video capture and frame retrieval
3. Implementing fully functioning code in Python
4. Adding Alarm to alert the driver

6. Project Demonstration

6.1 Code:



```
1 from scipy.spatial import distance
2 from imutils import face_utils
3 from playsound import playsound
4 from multiprocessing import Process
5 import imutils
6 import dlib
7 import cv2
8
9 def eye_aspect_ratio(eye):
10     A = distance.euclidean(eye[1], eye[5])
11     B = distance.euclidean(eye[2], eye[4])
12     C = distance.euclidean(eye[0], eye[3])
13     ear = (A + B) / (2.0 * C)
14     return ear
15
16 def playIt():
17     playsound("alarm.mp3")
18
19 thresh = 0.25
20 frame_check = 20
21 detect = dlib.get_frontal_face_detector()
22 predict = dlib.shape_predictor("../shape_predictor_68_face_landmarks.dat")
23
24 (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
25 (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
26 cap=cv2.VideoCapture(0)
27 drowsy_indicator=0
28 while True:
29     ret, frame=cap.read()
30     frame = imutils.resize(frame, width=450)
31     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
32     subjects = detect(gray, 0)
33     for subject in subjects:
34         shape = predict(gray, subject)
35         shape = face_utils.shape_to_np(shape)
36         lEye = shape[lStart:lEnd]
37         rEye = shape[rStart:rEnd]
38         lEAR = eye_aspect_ratio(lEye)
39         rEAR = eye_aspect_ratio(rEye)
40         ear = (lEAR + rEAR) / 2.0
41         leftEyeHull = cv2.convexHull(lEye)
42         rightEyeHull = cv2.convexHull(rEye)
43         cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
44         cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
45         if ear < thresh:
46             drowsy_indicator += 1
47             if drowsy_indicator >= frame_check:
48                 cv2.putText(frame, "*****ALERT*****", (10, 30),
```

Figure 7: Code Screenshot 1

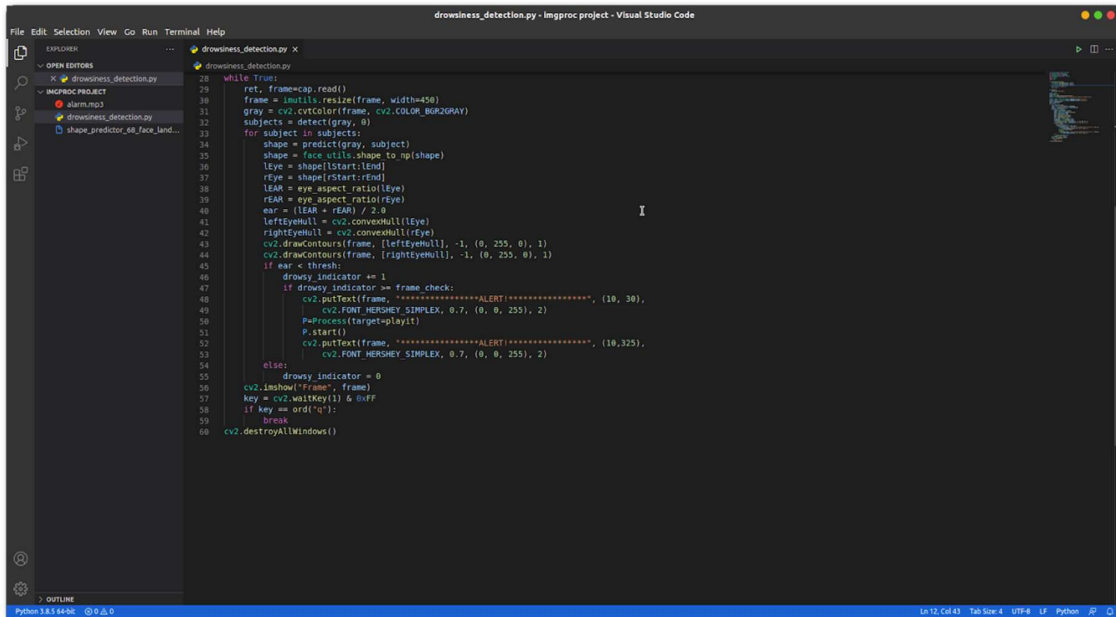


Figure 8: Code Screenshot 2

6.2 Output:

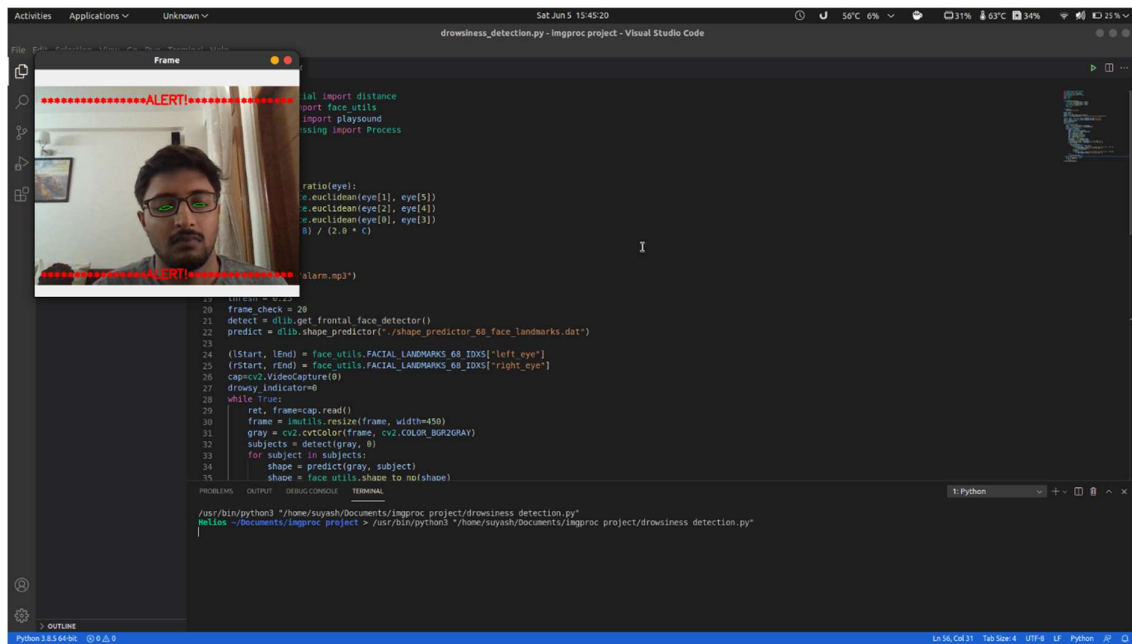


Figure 9: Output Screenshot

Conclusion

In this project, I learnt how face detection is carried out using haar cascades and also implemented it using python. I learnt how extract features from an image to detect various aspects and to use them in a creative way to fulfill the objective of my project.

As for future additions, the first thing that can be done is to create cheap and cost-effective hardware for the same so that it can be deployed on vehicles where it can be used in a large scale and actually help prevent accidents which costs so much damage to life and property. Another interesting feature that can be added is to add more prediction criteria and not only relying on eyes. For example, driver yawning could be a sign of fatigue. Another interesting metric could be the reaction time to outside stimuli, for example, time taken for the driver to stop their vehicle on red light or other any obstacle.