REAL – TIME DRIVER DROWSINESS DETECTION SYSTEM

A Project Report Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Technology

in

Computer Science and Engineering

by

PRAKHAR CHAURASIA	(1903480100076)
RASHI PANDEY	(1903480100081)
TANISHKA GUPTA	(1903480100113)
YASH CHATURVEDI	(1903480100128)

Under the Supervision of

Ms. Prachi Verma
(Assistant Professor)

PSIT COLLEGE OF ENGINEERING, KANPUR

to the

Faculty of Computer Science and Engineering

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly Uttar Pradesh Technical University)
May, 2023

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief. It contains no matter previously published or written by any other person nor material which to substantial extent has been accepted to the award of any other degree or diploma of the university or other institute of higher learning except where due acknowledge has been made in the text.

Signature:
Name : PRAKHAR CHAURASIA
Roll No : 1903480100076
Date :

Signature:
Name : RASHI PANDEY
Roll No : 1903480100081
Date :

Signature:
Name : TANISHKA GUPTA
Roll No : 1903480100113
Date :

Signature:
Name : YASH CHATURVEDI
Roll No : 1903480100128
Date :

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of B. Tech. Project **"Real-time Driver Drowsiness Detection System"** undertaken during B. Tech. Final year. We owe special debt of gratitude to our project guide **Ms. Prachi Verma (Assistant Professor, CSE)**, **PSIT College of Engineering**, **Kanpur** for her constant support and guide throughout course our work. Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only her cognizant efforts that our endeavors have seen light of the day.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Signature:
Name : PRAKHAR CHAURASIA
Roll No : 1903480100076
Date :

Signature:
Name : RASHI PANDEY
Roll No : 1903480100081
Date :

Signature:
Name : TANISHKA GUPTA
Roll No : 1903480100113
Date :

Signature:
Name : YASH CHATURVEDI
Roll No : 1903480100128
Date :

CERTIFICATE

This is to certify that the project titled **"Real-time Driver Drowsiness Detection System"** which is submitted by

- Prakhar Chaurasia (1903480100076)
- Rashi Pandey (1903480100081)
- Tanishka Gupta (1903480100113)
- Yash Chaturvedi (1903480100128)

in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering to PSIT College of Engineering, Kanpur, affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow during the academic year 2022-23, is the record of candidate's own work carried out by him/her under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

Mr. Abhay Kumar Tripathi
(Head of Dept., CSE)

Ms. Prachi Verma
(Assistant Professor, Dept. of CSE)

Real-time Driver Drowsiness Detection System

Ms. Prachi Verma (Assistant Professor, Dept. of CSE)

Prakhar Chaurasia Rashi Pandey Tanishka Gupta Yash Chaturvedi

ABSTRACT

Driver drowsiness is a common cause of accidents It is currently one of the primary causes of road accidents. Many accidents were caused by drowsy driving, according to the most current statistics. Each year, thousands of people are killed in car accidents caused by tired drivers. Drowsiness causes more than 30% of accidents. To avoid this and preserve lives, a system that detects drowsiness and notifies the driver is required.

In this project, we present a method for detecting driver weariness based on an eye aspect ratio (EAR). A smartphone's front-facing camera is used to capture the driver's face and extract facial landmarks using a face detection model and a landmark model, while a webcam is utilized to continually observe the driver. This model employs image processing technologies that highlight the driver's face and eyes in particular When a person with large eyes is drowsy, the size of his or her eyes will be different from the size of the eyes of a person with small eyes since the human eye has a different size in a normal physical state. As a result, under the system of this project, each person's ocular reference was built especially for them.

The model extracts the driver's face and predicts eye blinking based on the location of the eyes. The EAR value from the startup procedure for each driver is used as a baseline for determining whether a driver's eyes are closed or open based on the EAR threshold. If the EAR value (from the driving procedure) goes below the threshold several times in a row (with a reference of 1.5 seconds), the system will detect drowsiness. If the blinking rate becomes excessively rapid, the system alerts the driver with a sound. The project is designed to be lightweight, fast, and accurate, and to function on smartphones without the need for additional hardware or an internet connection.

Keywords — Drowsiness, Distraction, Eye detection, Eye Tracking, Face Detection, Perclos.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
ECG	Electrocardiography
EEG	Electroencephalography
FYP	Final Year Project
MIROS	Malaysia Institute of Road Safety
PERCLOS	Percentage of Eye Closure

CHAPTER 1

INTRODUCTION

One of the main causes of the majority of traffic accidents is driver intoxication. The dangers of being drowsy on the road can sometimes result in fatalities and significant financial losses as well as serious injuries. Drowsiness while operating a vehicle is associated with feeling lethargic, losing focus, and having fatigued eyes. India has a high accident rate because of distracted driving on the part of the drivers. Due to sleepiness, the driver's performance gradually declines. We created a system that can recognize the driver's drowsiness and alert him right away in order to prevent this anomaly. Through the use of a camera, this system records images as a video stream, locates the eyes, and recognizes faces. The Perclos algorithm is then used to analyze the eyes for drowsiness detection. The driver receives a drowsiness alarm via an alarm system based on the outcome.

To avoid these situations, we devised a system that detects the driver's tiredness and alerts him instantly. Many people drive on highways all day and night. This includes bus drivers, truck drivers, taxi drivers, and others who travel great distances and are sleep deprived. Driving while fatigued is extremely dangerous due to sleep deprivation.

Driver Drowsiness and sleep deprivation are important factors of many traffic accidents. Drowsy drivers endanger road safety and can cause significant injuries, culminating in the victim's death as well as economic loss. Drowsiness is a state in which a person feels lethargic, has difficulty concentrating, and has weariness in his eyes when driving a car.
The majority of accidents in India are caused by the driver's lack of concentration. Drowsiness impairs the driver's ability to drive over time.

Fatigue cannot be assessed, unlike alcohol and narcotics, which can be clearly detected and tested for readily. One of the best ways to combat sleepiness and fatigue-related accidents is to promote awareness and encourage drivers to disclose weariness when necessary. The first is difficult and far more expensive to acquire, while the latter is impossible without the former because driving for long hours pays well. When the lucrativeness of a sector's jobs increases, so does the wage link with it. This is also true when driving freight trucks at night.

Money also influences drivers to make poor decisions, such as driving all night despite exhaustion. This is because the drivers are unaware of the significant risks connected with driving while fatigued. Some governments have also put restrictions on the number of hours a driver can drive in a single day, but this is still insufficient to alleviate the problem because implementation is complex and expensive.

1.1PURPOSE

1.1.1 HUMAN PSYCHOLOGY WITH CURRENT TECHNOLOGY

Humans have always built machines and devised tactics to make their lives easier and safer, whether for boring tasks like getting to work or for more exciting ones like airline travel. Modes of transportation advanced with technological innovation, and human reliance on them grew enormously.

It has had a significant impact on our lives as we know them. We may now go to destinations at a rate that even our grandparents could not have imagined. Almost everyone in the modern world uses some form of transportation every day.

Some people are wealthy enough to possess cars, while others rely on public transportation. However, regardless of social status, there are some rules and codes of conduct for those who drive. One of them is staying alert and active while driving.

Neglecting our responsibilities to promote safer travel has resulted in hundreds of thousands of tragedies being related with this magnificent innovation each year.

Following traffic rules and regulations may seem insignificant to most people, yet it is critical.

While on the road, an automobile has the most power, and in the hands of irresponsible people, it can be damaging, and in some cases, that negligence can endanger the lives of those on the road. One example of carelessness is failing to admit when we are too fatigued to drive.

Many scholars have written research papers on driver sleepiness detection systems in order to monitor and prevent the negative consequences of such neglect.

However, some of the system's points and observations are not always correct.
As a result, this project was completed in order to provide data and another viewpoint on the problem at hand, in order to improve their implementations and further optimize the solution.

1.1.2 FACTS & STATISTICS

According to current statistics, 148,707 individuals died in car accidents in India alone in 2015. At least 21% of these were caused by weariness, which caused drivers to make mistakes.

This may be a lesser number even, as the role of exhaustion as a cause in an accident is often drastically underestimated among the many factors that might lead to an accident. Fatigue paired with poor infrastructure in developing countries such as India spells doom.

Fatigue, in general, is difficult to measure or observe, in contrast to alcohol and drugs, which have clear key symptoms and tests that are readily available. The greatest remedies to this problem are probably raising awareness about fatigue-related accidents and encouraging drivers to confess drowsiness when necessary.

The former is difficult and far more expensive to attain, and the latter is impossible without the former because driving for lengthy periods of time is extremely profitable.
When there is a greater demand for a job, the salaries connected with it rise, causing an increasing number of individuals to take it.

This is especially true when driving a vehicle at night. Money pushes drivers to make risky decisions, such as driving all night despite exhaustion. This is primarily because drivers are unaware of the significant risks connected with driving while fatigued.

Some countries have imposed restrictions on the number of hours a driver can drive in a row, but this is still insufficient to solve the problem because implementation is difficult and expensive.

1.2 DIFFERENT APPROACHES

Different methods can be used to determine whether a driver is drowsy. These three major categories can be used to group them:

1.2.1 BEHAVIOURAL PARAMETERS-BASED TECHNIQUES

This category includes assessing driver fatigue without the use of non-invasive instruments using the driver's eye closure ratio, blink rate, amount of yawning, head position, and facial expressions to analyze their behavior. The driver's eye-closure ratio is the current parameter used in this system.

1.2.2 VEHICULAR PARAMETERS-BASED TECHNIQUES

This category includes tracking a driver's driving habits to determine their level of fatigue. A few of these parameters are vehicle speed variability, steering wheel grip force, steering wheel angle, and lane change patterns.

1.2.3 PHYSIOLOGICAL PARAMETERS-BASED TECHNIQUES

This category includes methods that gauge driver drowsiness based on the driver's physical state. These parameters could include things like heart rate, breathing rate, body temperature, and many others.

These physiological parameters, as opposed to other various approaches, give the most accurate results because they are based on the biological makeup of the driver.

Each of the aforementioned strategies has pros and cons of its own. Any method can be used, depending on the accuracy of the desired result. The driver's body will be covered by the equipment as part of the physiological approach. This equipment has electrodes that can measure the driver's pulse rate, which could be uncomfortable for the driver while driving. Additionally, there is no guarantee that the driver will always wear this gear, which could lead to ineffective outcomes. Consequently, there is a drawback to the physiological approach. The effectiveness of the driver and his health are always the foundation of the vehicular-based approach.

There are additional limitations, such as the type of vehicle and the state of the road, both of which can change frequently. Therefore, it is best to use a camera to assess the driver visually in accordance with a behavioral-based approach. The driver cannot have any equipment on him or her. Therefore, this method is always the best course of action and can be used in any vehicle without requiring any modifications.

1.2.4 DIGITAL IMAGE PROCESSING

Digital computer processing of a two-dimensional image is generally referred to as "digital image processing." It refers to the digital processing of any two-dimensional data in a more general sense. An array of real numbers represented by a finite number of bits makes up a digital image. The main benefits of digital image processing techniques are their adaptability, repeatability, and ability to maintain the accuracy of original data.

A. PIXEL

The smallest component of an image is a pixel. Every pixel can represent any single value. The value of a pixel in an 8-bit grayscale image falls between 0 and 255. The values of a pixel at any given location reflect the power of the light photons striking there. A value proportional to the local light intensity is stored in each pixel.

B. DIGITAL IMAGE

Data numbers indicating different shades of red, green, and blue at a specific location on a grid of pixels are all that make up a digital image.

C. GRAY LEVEL

The intensity of the image at any given point is indicated by the pixel value, which is also referred to as the gray level. The formula previously used to convert an image to grayscale is:

$$Grayscale = \frac{Red + Green + Blue}{3}$$

However, since red has a longer wavelength, we use the formula:

Grayscale = ((0.3 * R) + (0.59 * G) + (0.11 * B)) **-- Eq: (1)**

1.3 MOTIVATION FOR THE WORK

In today's technologically advanced world, private transportation is rapidly expanding. When traveling a long distance, driving will be tiresome and boring. Long distance travel without adequate rest and sleep is one of the main factors contributing to the driver's lack of alertness. Driving while drowsy is possible for tired drivers. Drowsiness can cause dangerous and potentially fatal accidents in just a few fractions of a second. Continuous monitoring of the driver's alertness is necessary to prevent incidents of this nature, and the driver should be warned if drowsiness is detected. By doing this, we can significantly lower the number of accidents and save lives.

1.4 PROBLEM STATEMENT

The cause of a lot of traffic accidents is drowsy driving. Drowsiness can be identified by continuously streaming video of the driver using a camera or mobile device. The overall goal is to develop a model that can tell whether or not someone is feeling sleepy. The model takes an image every second, looks for blinking eyes, and uses the Perclos algorithm to determine when an eye closes. When the blink rate is high and the eye is closed for a predetermined period of time, the driver will be alerted by a sound.

1.5 SCOPE OF STUDY

In this project, we will focus on these following procedures:

A. Basic concept of drowsiness detection system

B. Familiarize with the signs of drowsiness

C. Determine the drowsiness from these parameters

- **a.** Eye blink
- **b.** Area of the pupils detected at eyes
- **c.** Yawning

D. Data collection and measurement.

E. Integration of the methods chosen.

F. Coding development and testing.

G. Complete testing and improvement.

1.6 ORGANIZATION OF THE THESIS

The organization of this thesis is as follows:

Chapter-1 is about introduction which gives an idea about our project domain i.e., Image Processing and title is explained namely, Automated driver drowsiness detection for non 2 wheelers. What is drowsiness and how it will be detected, which types are present are explained.

Chapter 2 is about a literature survey concerning this project. Here a brief look into all previous methods and existing models are examined.

Chapter 3 contains methodology. In this project, we used HAAR cascade for detecting face and eyes, and Perclos algorithm for calculating the blinking frequency of the eye. Even the architecture of the system is explained thoroughly.

Chapter 4 consists of experimental analysis and results in this sample code, testing results, system configurations such as software and hardware requirements, input and output images are displayed.

Chapter-5 explains the conclusion and future work about our project. Various ways of extending this project are explained here.

CHAPTER 2

LITERATURE SURVEY

Image processing in computer science refers to the manipulation of images using computer algorithms. Image processing has many benefits over analogue image processing as a subset or branch of digital signal processing. It permits the application of a much wider variety of algorithms to the input data and can prevent issues like the accumulation of noise and signal distortion during processing. Digital image processing can be modeled as a multidimensional system since images can exist in two dimensions.

2.1 DETECTION THROUGH ROI

A driver's face can be found using a region of interest (ROI). The area of interest is indicated by the blue rectangle. Getting the green rectangle area from the HAAR Cascade Classifier in the first frame, which includes height and width, is the first step in creating a ROI area. The rectangle is then magnified to create the region of interest.

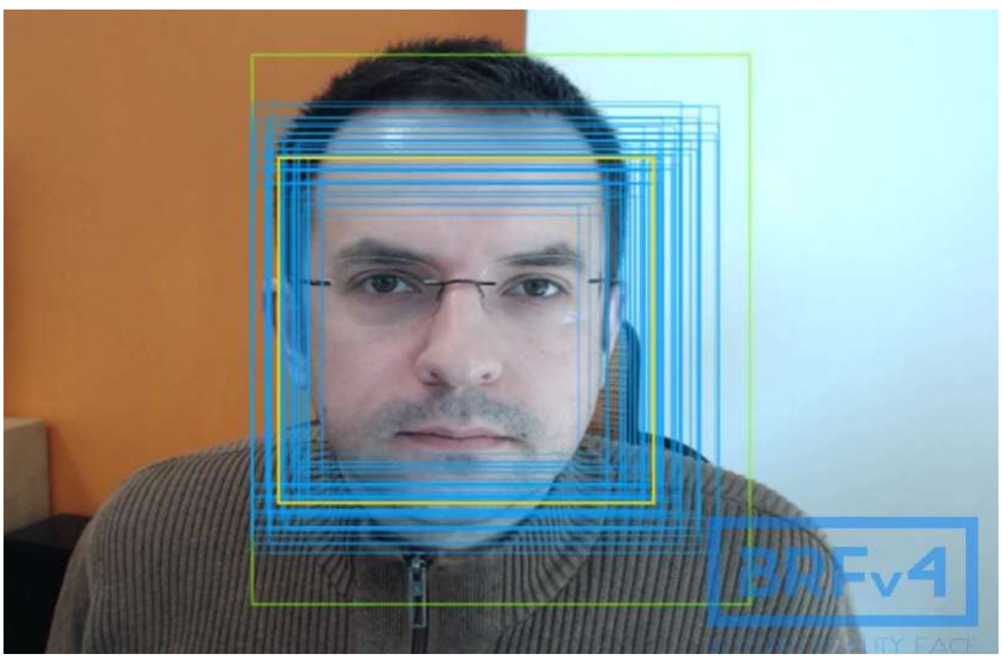

Fig 2.1 Region of Interest

DISADVANTAGES OF REGION OF INTEREST

1. It uses extra frames or squares to detect face detection.
2. It cannot be found in low light.
3. Why use the region of interest while the HAAR cascade classifier can do the same process?
4. It cannot be detected while using glasses while driving.

2.2 DETECTION THROUGH LBPH

This algorithm uses local binary pattern histograms to detect faces (LBPH). The creation of an intermediate image that encodes the original image in binary form is the first computational step in LBPH. The image is transformed into a matrix, and we must select its central value to serve as the threshold value. In order to define neighboring values, this value must be set to either 0 or 1.

The values that are 1 in the matrix form should be considered, and the other values should be ignored. Each pixel is represented by a value. This allows for the detection of the facial region.

DISADVANTAGES OF LBHP

1. It produces less urate results
2. The computational time is high.
3. This will work only if the data samples are less.

2.3 BEHAVIOURAL BASED TECHNIQUES

Various methods for behaviorally based parameters include:

2.3.1 EYE TRACKING AND DYNAMIC TEMPLATE MATCHING

A real-time driver fatigue detection system based on vision is suggested to prevent traffic accidents. Using the HSI color model, the system first recognizes the driver's face in the input images. Second, the positions of the eyes are determined using the Sobel edge operator, which also obtains images of the eyes to use as a dynamic template for tracking the eyes.

Then, the captured images are transformed into the HSI color model to determine whether the driver's eyes are closed or open in order to assess their level of intoxication. For the purposes of face detection and eye tracking experiments, four test videos are used. The suggested system is contrasted with the labeled data that has been annotated by professionals. The proposed system's accuracy is 88.90% and its average correct rate is 99.01%.

2.3.2 MOUTH AND YAWNING ANALYSIS

The primary factor in traffic accidents is fatigue. Sarada Devi and Bajaj suggested a system for detecting driver fatigue based on mouth and yawning analysis to get around the problem.

First, the system detects the driver's mouth from the input images and tracks it using a cascade of classifier training. Then, SVM is used to train the mouth and yawning image datasets. SVM is ultimately used to categorize the regions of the mouth in order to detect yawning and signal fatigue.

The authors gather some videos for the experiment and choose 20 yawning images and more than 100 regular videos as the dataset. The outcomes demonstrate that the proposed system outperforms the system based on geometric features.

The suggested system recognizes yawning, alerts driver fatigue earlier, and helps ensure driver safety.

2.3.3 FACIAL EXPRESSIONS METHOD

Researchers use a complex system called Finite Element Analysis in a laboratory setting that uses a database of facial expressions as a template to detect drowsiness based on test results. The hardware-based Driver Drowsiness Detection system by Assari and Rahmati is based on facial expressions.

The hardware system makes use of infrared light because it offers many advantages, including simplicity of use and independence from ambient lighting conditions. The system first determines the face region from the input images using the background subtraction technique. Then facial expressions are obtained using horizontal projection and template matching. The tracking phase then investigates the incidence of sleepiness by determining facial states from changes in the facial components after elements discovered earlier are followed up using template matching.

Eye brow raising, yawning, and closed eyes for a predetermined amount of time are the three primary indicators that something is wrong, and the system generates an alert when any of these three things change. The experiment is carried out in a real-world driving environment. Images are captured by the webcam during testing under various lighting scenarios and from various subjects. The findings look into whether the system responds appropriately when the driver has a mustache, beard, or glasses on.

2.3.4 YAWNING EXTRACTION METHOD

Road accidents are primarily caused by fatigue or drowsiness. Alioua suggested the effective Yawning extraction technique for monitoring driver weariness to resolve the problem. First, the face region is extracted from the photos using the Support Vector Machine (SVM) technique in order to lower the cost of the edges that are needed. With the help of the suggested method, the mouth can be localized. First, facial edges are detected using a detection technique. Next, a vertical projection of the lower half of the face is computed to identify the right and left region boundaries. Finally, a horizontal projection of the resulting region is computed to identify the upper and lower limits of the mouth.

Finally, the Circular Hough Transform (CHT) is applied to the mouth region images to detect the wide-open mouth. The system generates an alert if it detects a significant number of consecutive frames in which the mouth is open widely. The outcomes are contrasted with those of other edge detectors, such as Sobel, Prewitt, Roberts, and Canny. The experiment makes use of six movies that depict actual driving situations, and the results are displayed as a confusion matrix. The suggested method beats all previous edge detection methods and achieves a 98% accuracy rate.

2.3.5 EYE CLOSURE AND HEAD POSTURES METHOD

The Drowsy Driver Detection Using Eye Closure and Head Postures was proposed by Teyeb. Webcam footage is first recorded, then the subsequent actions are carried out for each frame of the video. Utilizing the viola-Jones approach, the ROI (face and eyes) is found. The face is divided into three regions, and the HAAR classifier scans the top area, which displays the eye area.

The images are then trained using a wavelet network based neural network to detect the eye state, and the class to which each image belongs is determined by comparing the coefficients of the learning images to the coefficients of the testing images. When a closed eye is recognized in the frames, the duration of the closure is calculated, and drowsiness is detected if the value exceeds the pre-defined time.

The established system then calculates the head movements, which include rotation to the left or right, forward or backward inclination, and left or right. The recorded video is divided into frames, and the head images are extracted together with their positions. The tilted condition of the head is then determined by comparing the photos, and the same is done for additional head postures.

In order to evaluate drowsiness, the method combines estimation of head posture and the length of eye closure. Ten participants participate in a series of experiments to test the system. Results also indicate that the systems are 80% accurate.

2.3.6 REAL TIME ANALYSIS USING EYE AND YAWNING

Kumar proposed using behavioral indicators and gestures like eye blinking, head nodding, and yawning to identify the drivers' state in real-time driver fatigue detection.

The fundamental goal of the suggested method is to concurrently detect a closed eye and an open mouth and to sound an alarm upon successful detection. The camera situated in front of the driver is used by the system to first capture real-time footage. Then, using the training set of faces and eyes provided by OpenCV, the viola-jones method is applied to the collected video frames in order to detect faces and eyes.

A small rectangle is drawn around the eye's Centre, and a matrix is made to illustrate the Region of Interest (ROI) for eyes, which will be used in the following phase. Since both eyes blink simultaneously, only the right eye is checked to determine whether the eyes are closed. A closed eye is one that has been closed for a predetermined period of time.

First, the eyeball color is collected by sampling the RGB components on the Centre of the eye pixel in order to ascertain the eye condition. Then, to distinguish between an open and closed eye, absolute thresholding is performed on the eye ROI based on the color of the eyeball and an intensity map that displays the distribution of pixels on the y-axis, or height of the eyeball. The threshold value is 4 in this case. After that, the eye blinking rate is calculated after 100 frames if the eye blink is detected in each frame, in which case it will be treated as 1 and saved in the buffer.

The size of the mouth is then measured using a contour finding technique to identify the mouth's yawning motion. if the height exceeds a specific threshold. It indicates that the person is yawning. The performance of the suggested system has been measured for 20 days at various times while being used by people wearing glasses, without glasses, and with and without mustaches. When the drivers are without their glasses, the system operates at its best.

2.3.7 EYE BLINK DETECTION METHOD

The Driver Drowsiness System was proposed by Ahmad and Borolie and is based on non-intrusive machine-based notions. A web camera that is mounted in front of the driver makes up the system. For the sake of simulating events, both saved and online videos are considered.

First, the driver's head motions and facial expressions are captured by the camera. The video is then divided into frames, and each frame is individually processed. The Viola-Jones algorithm is used to identify faces in frames. Then, a cascade classifier is used to extract from the face the necessary features, such as the eyes, lips, and head. Rectangles on the face represent the area of interest.
The primary indicator of drowsiness in this case is decreased eye blink frequency, which typically ranges from 12 to 19 times per minute and suggests drowsiness. Average sleepiness is calculated rather than eye blinking.

Non-zero readings are displayed as partially or fully open eyes. The detected eye is identical to zero (closed eye). The average is calculated using equation (2).

$$\%d = \frac{No.\ of\ Closed\ Eyes\ Found}{No.of\ Frames} \qquad \text{........Eq: (2)}$$

The system generates an alarm to notify the driver if the value exceeds the predetermined threshold value. Additionally, yawning is also thought to cause the alarm. Videos from both online and offline sources are used in experiments run on two distinct systems. The outcomes demonstrate that the system is up to 90% efficient.

2.3.8 EYE CLOSENESS DETECTION METHOD

Khunpisuth develops an experiment that uses a Raspberry Pi camera and Raspberry Pi 3 model B to determine the degree of driver drowsiness. Video is first captured using the Pi camera, and then the HAAR cascade classifier from the Viola-Jones method is used to identify facial regions in the images.

Diverse photos are trained in various lighting conditions. The case study with 10 volunteers resulted in the percentage of 83.09%. The Region of Interest (ROI) is displayed as a blue rectangle. On the most recent frame received, the HAAR cascade classifier is once more applied, reducing the size of the ROI.

Drowsiness level is determined using eye blink rate after face detection. The authors utilize three templates to assess the eye blink an eye area while utilizing template matching to identify the eye region on the face. Given that it produces better results than other template matching techniques, OpenCV's CV TM CCOEFF NORMED is taken into consideration.

The combination of face and eye detection enables the measurement of eye blink and proximity rates. When the eyes are closed, the value of the closed eye is greater than the value of the open eye, and the opposite is true when the eyes are open. Authors assumed that a front-facing face would enable the HAAR cascade classifier to function. That is why authors suggested the technique to turn the tipped face back toward the front. checks whether or not the head is tilted before calculating the degrees of rotation (angle).

After the driver's face and eyes are accurately detected, their level of tiredness is calculated. The device flags tiredness if drivers' eyes blink excessively. To warn the driver, a loud sound will be produced when the level hits 100.

The results demonstrate that the proposed method achieves an accuracy of 99.59% when compared to the HAAR cascade. It is capable of detecting a face wearing glasses and operates in all lighting conditions.

2.4 VEHICULAR PARAMETER-BASED TECHNIQUES

2.4.1 REAL TIME LANE DETECTION SYSTEM

In the modern era, traffic accidents are all too common, endangering both the lives of those on the road and their property. Road accidents can occur for a variety of causes, including: rash driving, inexperience, disobeying signs, jumping signals, etc.

The Drivers' Drowsiness Detection system was suggested by Katyal et al. as a solution to the problems. The system operates in two stages: first, it uses the Hough transform to identify lanes.

Secondly, it looks for signs of tiredness in the drivers' eyes. The viola jones approach is used to detect faces first, followed by picture segmentation, Otsu thresholding, and clever edge detection for eye detection.

The data are used with the circle detection through transform method to find eyeballs and gauge level of exhaustion. Low lightning conditions won't affect its effectiveness. The results indicate that the suggested strategy is beneficial for drivers who take long journeys, drive at night, or drive after drinking.

2.4.2 TIME SERIES ANALYSIS OF STEERING WHEEL ANGULAR VELOCITY

Zhenhai invented the Driver Drowsiness Detection method, which uses time series analysis of steering wheel angular velocity to prevent traffic accidents.

The technique first examines the steering behavior below tiredness, and then employs a temporal detection window as a detection characteristic to ascertain the steering wheel's angular velocity over time. If the detection feature in the temporal window complies with the variability restrictions and extent requirements, the

Drowsiness is consequently picked up on. The experiment is run using actual testers, and the findings reveal that the new method performs better than the old ones and is applicable to real-world situations.

2.4.3 STEERING WHEEL ANGLE FOR REAL DRIVING CONDITIONS FOR DDT

Li suggested utilizing a drowsiness detection system to monitor driver fatigue levels in real-world situations while using steering wheel angles in order to prevent traffic accidents (SWA). The sensors mounted to the steering lever are where the SWA data is gathered.

The system extracts Approximate Entropy (ApEn) features first from fixed sliding windows on time series of real-time steering wheel angles, and then linearizes the ApEn features utilizing the deviation of adaptive linear piecewise fitting approach. The technique then determines the warping distance between a series of sample data's linear characteristics.

Finally, the system uses warping distance in accordance with the created decision classifier to determine the drivers' level of tiredness. The empirical analysis employs data gathered over 14.68 hours of actual driving time, evaluated on two levels of fatigue: drowsy and awake.

Results indicate that the suggested system is useful for preventing traffic accidents brought on by fatigued drivers and capable of working online with an accuracy of 78.01%.

2.4.4 AUTOMATIC DETECTION OF DRIVER FATIGUE

To combat driver fatigue, it is suggested to use information on steering wheel angles (SWA) and yaw angles (YA) in actual driving situations to detect driver fatigue online.

The system first examines the operation features of SWA and YA in various states of fatigue, calculates the ApEn features on time series of shot sliding window, and then, using dynamic time series of non-linear feature construction theory and inputting fatigue features, designs a 2-6-6-3 multi-level Back Propagation (BP) neural network classifier to detect fatigue.

An experiment lasting 15 hours is conducted on a genuine road setting for empirical analysis. The specialists analyzed the data and divided it into three categories: drowsy, very drowsy, and awake. And the experiment obtains an average level of accuracy in fatigue detection of 88.02%, which is significant for engineering applications.

2.5 DROWSINESS DETECTION THROUGH PHYSIOLOGICAL APPROACH

The majority of the time, physiological measurements have been used to identify drowsiness since they can provide a quick and accurate measurement. EEG, eyelid closure, eye movements, heart rate, pupil size, skin conductivity, and cortical formation are all conceivable measurements.

The systems that are most precise among these methods are those that take into account a person's bodily experience. There are two methods for carrying out this process. measurement of physical changes, such as a slouched posture, the driver's head leaning, and the open/closed state of the eyes, as well as physiological changes, such as changes in heart rate, brain waves, and eye blinking.

2.5.1 EEG - BASED DRIVER FATIGUE DETECTION

The electrical activity produced by human brain nerve cells, or more specifically, the cortical movement, can be measured using an EEG approach. The EEG-action is always available, and recordings demonstrate both arbitrary and periodic behavior.

Although the thalamus and other subcritical regions of the human brain can also initiate some neural activity, the cerebral cortex is the EEG's primary origin. The EEG reflects the balance of postsynaptic inhibitory and excitatory potentials in nerve cells. Synchronous nerve cell activation is what causes the musical movement.

To prevent the road accidents that are typically brought on by drivers' fatigue, a technique for detecting it using electroencephalogram (EEG) signals has been developed.

The suggested approach first determines the index for various levels of tiredness. The technology uses inexpensive single electrode neural signal acquisition equipment to calculate the EEG signal as input.

Data sets for simulated car drivers operating under various levels of tiredness are being gathered in order to assess the suggested strategy. The results demonstrate that the suggested system is capable of detecting fatigue in all subjects.

DISADVANTAGES:

1. High cost
2. Sensors Required
3. Cannot see behind objects
4. Takes longer time

2.5.2 WAVELET ANALYSIS OF HEART RATE VARIABILITY & SVM CLASSIFIER

Li and Chung proposed the Heart Rate Variability (HRV) wavelet analysis and Support Vector Machine (SVM) classifier for the driver drowsiness detection. The primary goal is to classify awake and sleepy drivers using wavelet transforms of short-term HRV signals.

The system divides the input Photoplethysmography (PPG) signal into 1-minute intervals before employing the average percentage of eyelid closure over pupil over time (PERCLOS) measurement over the interval to verify two driving occurrences. Second, the system uses wavelets and Fast Fourier Transform (FFT) to extract features from the HRV time series.

For feature extraction and classification, respectively, Receiver Operating Curve (ROC) and SVM classifier are employed. The wavelet-based method outperforms the FFT-based method, according to the ROC analysis.

Finally, the SVM classifier extracted from the HRV signals is trained using the real-time requirements for drowsiness detection, FFT, and wavelet features. The accuracy, sensitivity, and specificity of the classification performed using wavelet-based features are 95%, 95%, and 95% respectively.

The accuracy of the FFT-based results is 68.85 percent. The outcomes demonstrate that wavelet-based methods outperform FFT-based methods in terms of performance.

2.5.3 PULSE SENSOR METHOD

Prior research has mostly focused on a driver's physical state to identify tiredness. Rahim uses infrared heart-rate sensors or pulse sensors to identify the sleepy drivers. The drivers' finger or hand is used by the pulse sensor to monitor heart rate.

The sensor, which is fastened to the finger or hand, measures how much blood is passing through the finger. The infrared light then reflects off the finger and travels to the transmitter as the amount of oxygen in the blood is revealed. The sensor detects changes in oxygen levels that are attached to an Arduino microcontroller.

The software processing HRV frequency domain then visualizes the heart rate. According to experimental findings, the ratio of low to high frequencies (LF/HF) decreases as drivers become drowsy and many accidents can be prevented if an alert is sent promptly.

2.5.4 WEARABLE DRIVER DROWSINESS DETECTION SYSTEM

Applications for mobile devices have been created to identify driver drowsiness. However, using a phone while driving might divert attention and result in an accident. Leng suggested a wearable-style sleepiness detection technology as a solution to the problem.

The technology makes use of a custom-made wristband with a galvanic skin response sensor and PPG signal. The mobile device, which serves as the primary assessing unit, receives the sensor data. The mobile devices' built-in motion sensors are used to analyze the collected data.

Heart rate, respiratory rate, stress level, pulse rate variability, and adjustment counter are the next five features to be extracted from the data. Additionally, the features are employed by the SVM classifier as computation parameters to identify the condition of somnolence.

According to the experimental findings, the proposed system's accuracy can reach up to 98.02%. To warn the motorist, a mobile phone generates a graphical and vibrating alarm.

2.5.5 WIRELESS WEARABLES METHOD

Warwick suggested employing a wearable biosensor dubbed a "Bio-harness" to detect tiredness in order to prevent fatal car accidents.

Two phases make up the system. Using a bio-harness, the physiological data of the driver is gathered in the first phase. The data is then analyzed to identify the crucial indicators of drowsiness, such as the ECG, heart rate, posture, and others.

The development of a mobile app and creation of a drowsiness detection algorithm will take place in the second phase.

2.5.6 DRIVER FATIGUE DETECTION SYSTEM

The Driver Fatigue Detection System is presented by Chellappa. The fundamental function of the technology is to identify sleepiness while the car is moving. The system consists of three parts: a data processing module, an alert unit, and external hardware (sensors and a camera).

The USB port is used by the hardware unit to communicate with the rest of the system. Somatic sensors are used to continuously monitor physiological and physical parameters like pulse rate, yawning, closed eyes, blink duration, and others. To determine drowsiness, the processing module combines a variety of factors. Ultimately, the alarm device notifies the driver at several stages based on the severity of the symptoms.

2.5.7 HYBRID APPROACH UTILIZING PHYSIOLOGICAL FEATURES

Awais presented a hybrid technique that combines ECG and EEG characteristics to enhance the performance of detection. The procedure extracts time and frequency domain features initially, such as time domain statistical data, spectral measures, complexity measures, and descriptors from EEG. Afterward, features such heart rate, HRV, low frequency, high frequency, and LF/HF ratio utilizing ECG.

The association between perceived sleepiness and drowsiness is then studied. T-tests, which can distinguish between the awake and drowsy, are employed to pick only statistically relevant features. The features that were taken from the ECG and EEG are combined to examine how SVM performance has improved.
The channel reduction and its effect on the effectiveness of detection are the other major contribution.

The method uses physiological information gathered during the driving simulation study to compare the differences between the alert and drowsy states. A monotonous driving environment is employed to make the participants feel sleepy.

The proposed method showed that, instead of using them separately, integrating ECG and EEG increases the system's effectiveness in discriminating between alert and drowsy states. By simply merging the ECG and EEG, the accuracy level can be increased to 80%, according to the analysis of channel reduction.

The system's performance shows that the proposed approach can be used as a real-world sleepiness detection system.

2.6 OTHER METHODOLIGIES:

Numerous fatalities are caused each year by fatigue-related motor vehicle accidents. A study found that tiredness causes an average of 90 fatal accidents per day and 20% of all accidents on a yearly basis. Continuously driving drivers run the risk of becoming fatigued.

Therefore, drowsy driving detection and its indication can greatly reduce the number of accidents. Some image processing approaches, including as viola jones, Adaboost, HAAR cascade, gobar features, and facial land mark identification, have been shown to reduce these types of accidents. The methods for identifying sleepiness include the following.

M.A. Assari and M. Rahmati proposed a method in which the driver's tiredness is identified by identifying the face through horizontal projection on the image and tracking the facial components—eyebrows, mouth, and eyes—via template matching technique. The suggested MATLAB's simulation platform has been used to implement the method (Simulink). The use of IR lighting as a source of light improved this system's ability to recognize faces.

A face-detection system based on a cascade of classifiers trained using the Adaboost algorithm was presented by Tianyi Hong. The integral picture of the original image is applied during optimization in this system to create a clever filter for cascade processing and boost performance. Better and quicker computational results have been achieved by using integrated performance primitives (IPP). This system has been verified on the embedded GENE-8310 platform.

A method based on a physiological approach was proposed by B. Warwick in which the driver wears a wireless biosensor called a Bio Harness, a wearable device that can collect physiological data and communicate it to a smartphone. The Fast Fourier Transform (FFT) and Power Spectral Density (PSD) are then used to analyze this data, producing the appropriate vectored inputs that may be fed into a Neural Network. The researchers' sleepiness detecting mobile app is used to run this system.

K. Dwivedi created a system that uses representational learning to detect driver drowsiness. A SoftMax layer classifier is trained to determine if a driver is or is not drowsy using the characteristics extracted from the photos by a 2-layer convolutional neural network using a HAAR-like face detector. This technology was successful in detecting tiredness and notifying the driver with a respectable result of 78% accuracy.

A system created by J.J. Yan converts recorded pictures into grayscale using the Sobel operator for edge detection. Template matching is used to determine the eyes' location. Binarization and rapid sort approaches, which also validate the distribution of the black pixels in the grayscale image, are employed to ascertain the states of the eyes.

In this study, P80 is used as the primary physical condition indicator for the driver. It is assumed that the driver is drowsy if the percentage of black pixels falls below this threshold.

2.7 EXISTING SYSTEM

The current drowsiness detection systems make use of equipment that measures things like blood pressure, heart rate, and respiration rate. The use of these gadgets may make the motorist uncomfortable while driving. It cannot be guaranteed that drivers wear these gadgets at all times when operating a vehicle. may malfunction or become lost, which could cause a poor degree of accuracy in the outcome. Low light circumstances do not lend itself to the current technology producing good outcomes. Lower accuracy is the result of the inability to recognize the driver's face and eyes in low or dark lighting situations.

CHAPTER 3

METHODOLOGY

3.1 PROPOSED SYSTEM:

In our drowsiness detector case, we will be monitoring the eye aspect ratio to see if the value falls but does not increase again, thus implying that the person has closed their eyes then the driver is alerted for the drowsiness through an alert from the app.

3.1.1 ARCHITECTURE:

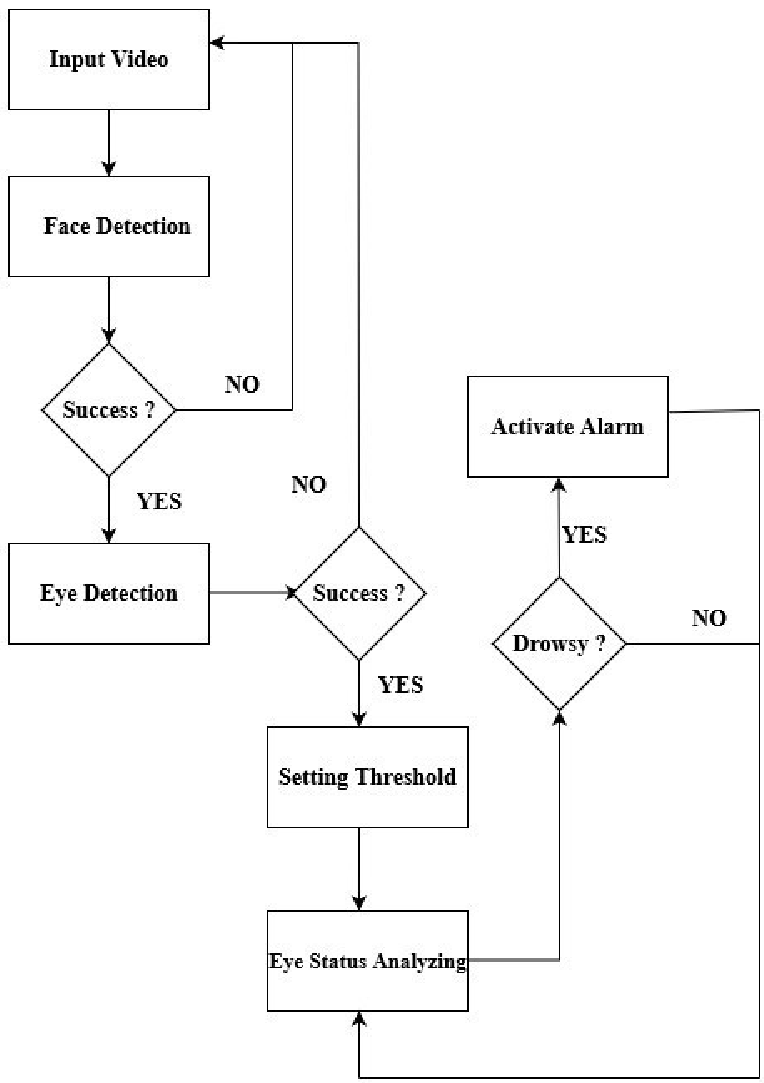

Fig 3.1 Architecture of the Real-time Driver Drowsiness Detection System

This is the architecture for detecting the drowsiness of the driver.

A. **Real-time Input Video***:* In this stage, the application requests camera permissions; if granted, it uses the user's Android smartphone's camera and starts the processing.

B. **Face Detection***:* The face is detected using the dlib library's pre-trained facial landmark detector **(shape_predictor_68_face_landmarks.dat)**, which estimates the location of 68 (x, y)-coordinates that map to facial structures on the face.

C. Facial landmarks represent salient regions of the face, such as:

   a) Eyes
   b) Eyebrows
   c) Nose
   d) Mouth
   e) Jawline

D. **Eyes Detection***:* To extract the eye regions from a set of facial landmarks, we simply need to know the correct array slice indexes:

   a) The right eye using [36, 42]
   b) The left eye with [42, 48]

E. **Setting up of Threshold:** The human eye has a different size in a normal physical state, so when a person with large eyes is drowsy, the large opening of his eyes will be different from the size of the eyes of someone with small eye size.

F. As a result, in this project, the system was designed with a unique eye reference for each person. When a driver starts driving a vehicle, preliminary data retrieval is performed assuming the driver's initial condition is not drowsy.

G. This initial data will be used as a reference for threshold decision making in the process of determining drowsiness. In this project using the EAR value of each driver from the initialization process, and the EAR value is also used as a baseline for stating a driver's eyes are closed or open based on the EAR threshold.

H. **Eye Status Analyzing**: This phase determines that the eyes are closed or not using the EAR value. If the EAR value is below the threshold for several consecutive times (with a reference of 1.5 seconds) the driver will be detected drowsy by the system.

I. **Alarm Beeping**: This is the final phase of the architecture workflow. If the eyes found closed (if detected drowsiness) then it immediately alerts driver by beeping the alarm.

3.2 MODULAR DIVISION:

The entire architecture is divided into five modules.

1. Face Detection
2. Eye Detection
3. Face Tracking
4. Eye Tracking
5. Drowsiness Detection
6. Setting Up Of Threshold

1. FACE DETECTION

This module takes input from the camera and tries to detect a face in the video input. The detection of the face is achieved through the HAAR classifiers mainly, the Frontal face cascade classifier. The face is detected in a rectangle format and converted to grayscale image and stored in the memory which can be used for training the model.

The face is detected using the dlib library's pre-trained facial landmark detector **(shape_predictor_68_face_landmarks.dat)**, which estimates the location of 68 (x, y)-coordinates that map to facial structures on the face.

Facial landmarks represent salient regions of the face, such as:

i. Eyes
ii. Eyebrows
iii. Nose
iv. Mouth
v. Jawline

The indexes of the 68 coordinates can be visualized on the image below:

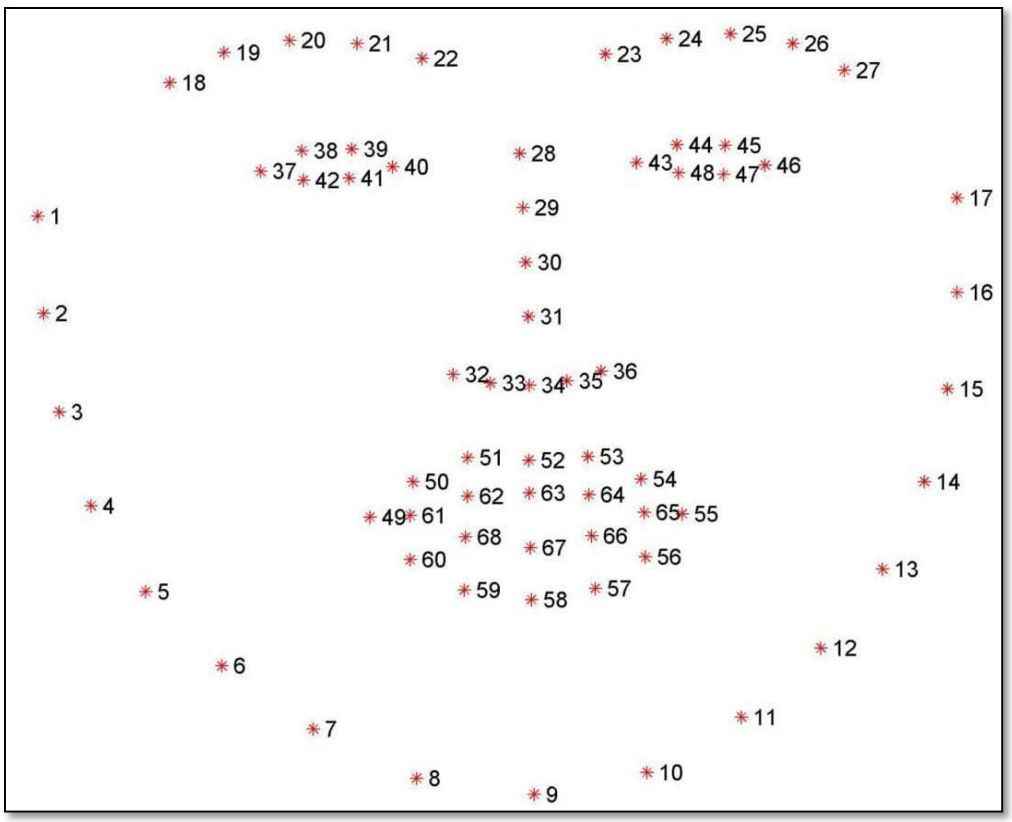

Fig 3.2.1 Dlib Facial Landmarks

2. EYE DETECTION

Since the model works on building a detection system for drowsiness, we need to focus on the eyes to detect drowsiness. The eyes are detected through the video input by implementing a HAAR classifier namely HAAR Cascade Eye Classifier. The eyes are detected in rectangular format.

To extract the eye regions from a set of facial landmarks, we simply need to know the correct array slice indexes:

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

3. FACE TRACKING

Due to the real-time nature of the project, we need to track the faces continuously for any form of distraction. Hence the faces are continuously detected during the entire time.

4. EYE TRACKING

The input to this module is taken from the previous module. The eyes state is determined through Perclos algorithm.

5. DROWSINESS DETECTION

In the previous module the frequency is calculated and if it remains 0 for a longer period then the driver is alerted for the drowsiness through an alert from the system.

In our drowsiness detector case, we'll be monitoring the eye aspect ratio to see if the value falls but does not increase again, thus implying that the person has closed their eyes then the driver is alerted for the drowsiness through an alert from the app.

6. SETTING UP OF THRESHOLD

The human eye has a different size in a normal physical state, so when a person with large eyes is drowsy, the large opening of his eyes will be different from the size of the eyes of someone with small eye size. As a result, in this project, the system was designed with a unique eye reference for each person.

When a driver starts driving a vehicle, preliminary data retrieval is performed assuming the driver's initial condition is not drowsy. This initial data will be used as a reference for threshold decision making in the process of determining drowsiness. In this project using the EAR value of each driver from the initialization process, and the EAR value is also used as a baseline for stating a driver’s eyes are closed or open based on the EAR threshold.

If the EAR value (from driving process) is below the threshold for several consecutive times (with a reference of 1.5 seconds) the driver will be detected drowsy by the system. If detected drowsiness, it will be given a response in the form of alert sound.

3.2.1 HAAR CASCADE

The foundation of HAAR Cascade is the idea of features that Paul Viola and Michael Jones put forth in their 2001 paper, "Rapid Object Detection using a Boosted Cascade of Simple Features." Using machine learning, a cascade function is trained using a large number of both positive and negative images. It can be utilized to find objects in a picture or a video.

This algorithm comprises of four stages:

i. HAAR Feature Selection
ii. Creating Integral Images
iii. Adaboost Training
iv. Cascading Classifiers

Despite being used to identify almost all objects, HAAR Cascade is particularly well-liked for identifying faces in pictures. Adaboost is a service that both chooses the best features and trains classifiers to use them. This algorithm creates a "strong" classifier by linearly combining weighted, basic "weak" classifiers.

An adjacent rectangular region in a detection window is taken into account by a HAAR feature, which then calculates the difference between the sums of the intensities of the pixels in each region.

During the detection stage, a window with the target size is moved over the input image, and HAAR features are computed for each section of the image. Then, this difference is contrasted with a learned threshold that distinguishes between objects and non-objects.

Since a large number of HAAR features are required to accurately describe an object and each HAAR feature is only a "weak classifier," or slightly better than random guessing, they are arranged into cascade classifiers to create a strong classifier. Digital image features that resemble HAARs are used for object recognition. They were employed in the first real-time face detector and got their name from how intuitively they resembled HAAR wavelets. In a detection window, a HAAR-like feature considers adjacent rectangular regions at a specific location, sums the pixel intensities in each region, and then determines the difference between these sums. Subdivisions of an image are then categorized using this distinction. For instance, it is a well-known fact that, on all faces, the area around the eyes is darker than the area around the cheeks.

Thus, based on Fig. 3.2, a typical HAAR feature for face detection is a pair of adjacent rectangles that are located above the cheek and eye regions. These rectangles are positioned in relation to a detection window, which serves as the target object's bounding box.

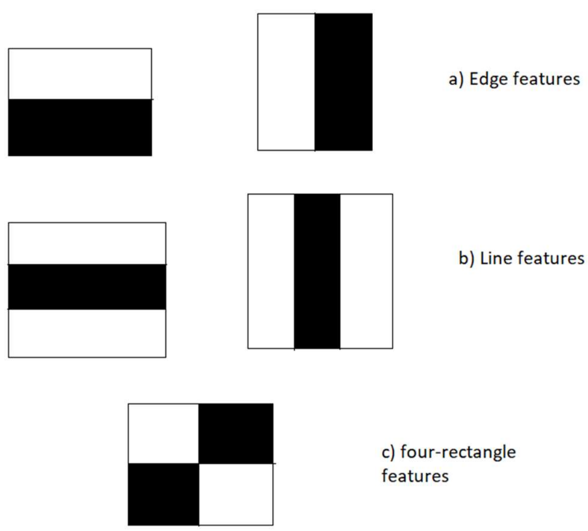

Fig 3.2.2 Different types in feature extraction

The target size window is moved over the input image during the detection stage of the viola jones object detection framework, and the HAAR-like feature is calculated for each area of the image.

Then, this difference is contrasted with a learned threshold that distinguishes between objects and non-objects. A large number of HAAR like features are required to accurately describe an object because one HAAR-like feature is only a weak learner or classifier.

The HAAR-like features are arranged in the Viola-Jones object detection framework in a structure known as a classifier cascade to create a powerful learner or classifier.

A HAAR-like feature's primary advantage over most other features is its calculation speed. A HAAR-like feature of any size can be calculated in constant time by using integral images. A data structure and algorithm called an integral image are used to create the sum of values in a rectangular subset of a grid. In order to obtain the summations of pixel intensities within a window, fewer computations must be performed.

The goal is to convert an input image into a summed-area table, where any point (x, y) in the table has a value equal to the sum of all the pixels above and to the left of (x, y), inclusive:

$$I(x,y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x', y')$$

Where I(x,y) is the value of the integral image pixel in the position (x,y), while i(x,y) is the corresponding intensity in the original image.

It is a recursive formula, hence, if we start from one corner of the input image, we will have the same result in the integral image.

1	3
7	9

0	0	0
0	1	4
0	8	20

Fig 3.3 Calculation of integral image from an actual image

Since we need to go back one step in order to begin the recursive formula, we add one row and one column of zeros. The integral of this image will be w+1 pixels wide and h+1 pixels high if the actual image is w pixels wide and h pixels high.

Starting with the first pixel in the original image from fig. 3.3 with intensity 1, let's begin the computations. The integral image produces the same result because it is computing (1+0+0). Pixel 3 then changes to pixel 4, since it is 3+1+0+0.

The same method yields a "8" (7+1+0) and a "20" (9+3+1+7).

We now have a new picture. This image is helpful for a special integral image property. In fact, it turned out that if you need to calculate the summation within a window in the input image, that summation is equal to a linear combination of the corner of the corresponding window in the integral image, as shown in the following example:

$$\sum_{\substack{x0< x\leq x1 \\ y0<y\leq y1}} i(x,y) = I\ (D) + I\ (A) - I(B) - I(C)$$

Where A, B, C and D are the corners of the corresponding window in the integral image of fig. 3.4

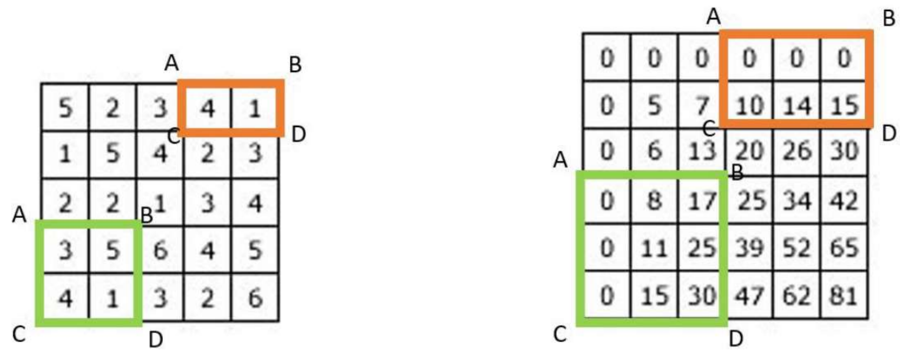

Fig 3.4 Converting a 5x5 image to a 6x6 integral image

As a result, fewer computations are required. Consider a 100x100 image with a 9x9 window to get an idea. It takes 8 operations to calculate the sum of the pixel intensities contained within that window. We get 800 operations if we do this process 100 times. Let's examine the integral image methodology now. The summed-area table must first be computed, which takes 56 operations.

The above formula, which is composed of 3 operations, can then be used to compute the sum of pixel intensity using the same 99 window. As a result, there were 56+3*100=356 operations in total. It's less than a half, as you can see. The algorithm known as the HAAR Cascade, which is heavily used in computer vision, is based on this process.

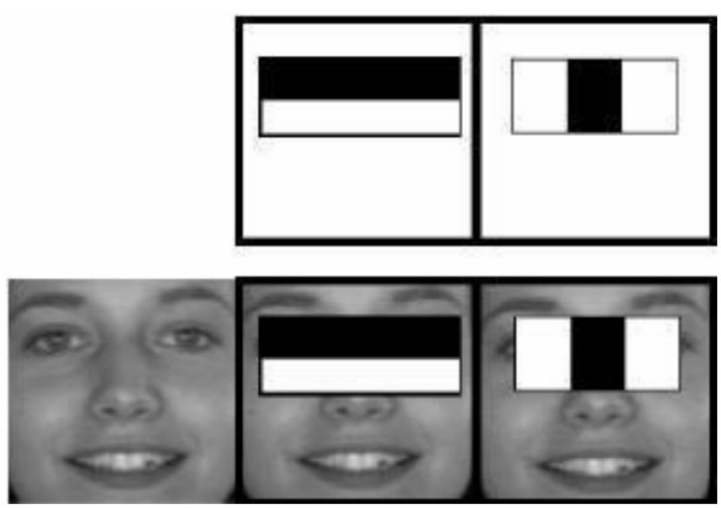

Fig 3.5 Extracting different features

The cascade classifiers for the eye's region are shown in fig. 3.5. In the second image, the area around the eyes is black, or both eyes are white. The system recognizes them as the eyes if it detects the same pixel intensity at both locations.

The detection is based on the pixel intensity surrounding a specific object. The area around the eye is black, while the area around the eye is white. If white pixels are present, the area is recognized as an eye.

The Adaboost technique, which chooses the best features and trains the classifiers that use them, is used to select the most pertinent features.

3.2.2 PERCLOS

Percentage of Eye Closure (PERCLOS) is defined as the proportion of time for which the eyelid remains closed more than 70-80% within a predefined time period. Level of drowsiness can be judged based on the PERCLOS threshold value.

Perclos is a drowsy detection tool that estimates the percentage of time that the eyelids are closed over the pupil. It can produce useful results and is used by many real-time drowsiness detection systems. To capture the eyelid closure movement and improve the accuracy of the system, developers employ a variety of hardware.

This project makes use of a dashboard-mounted camera that is configured so that the driver can be seen on the image. This aids in more accurate face detection and Perclos measure calculation of eyelid closure frequency. Each eye has six total points marked, and the Euclidean distance is computed for each eye.

The eye aspect ratio for each eye is then calculated for average eye-aspect ratio.

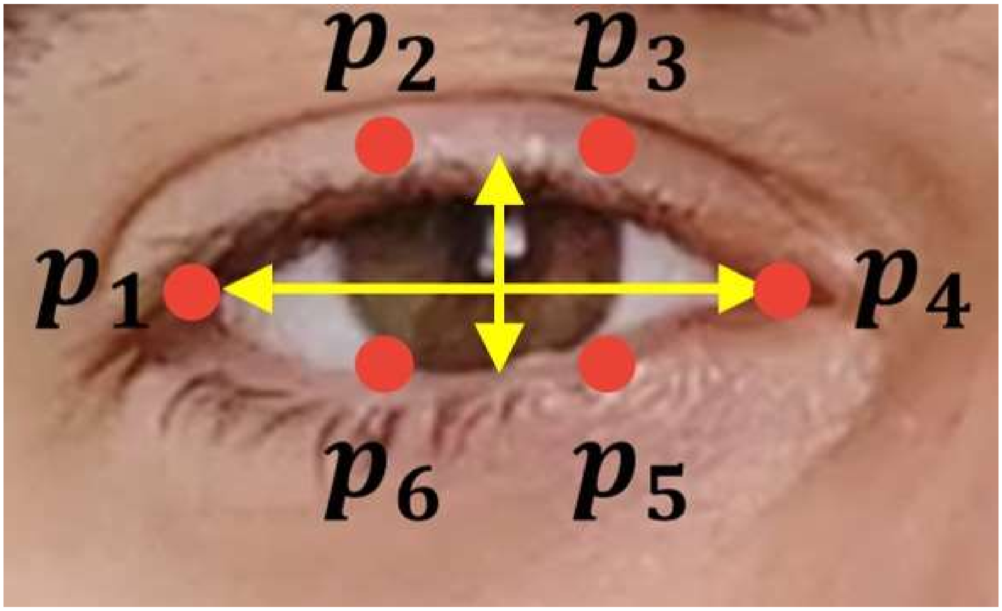

Fig 3.6 Perclos algorithm

If you notice, each eye is represented using 6 landmarks points. The EAR for a single eye is calculated using this formula:

$$\boldsymbol{EAR = \frac{(A+B)}{(2.0*C)}} \qquad \textbf{Eq …… (5)}$$

Where,

A is the distance between the 2-points (p2 and p6)

B is the distance between the 2-points (p3 and p5)

C is the distance between 2-points (p1 and p4)

For this system, there are 20 frames and a 0.25 threshold value.

This value determines whether the alarm will sound. The driver will be identified as being drowsy and alert through the sounding of an alarm if the eye aspect ratio is below the threshold value for the specified number of frames.

The more the EAR, the more widely eye is open. We would decide a minimum EAR value and used this to decide if the eye is closed or not.

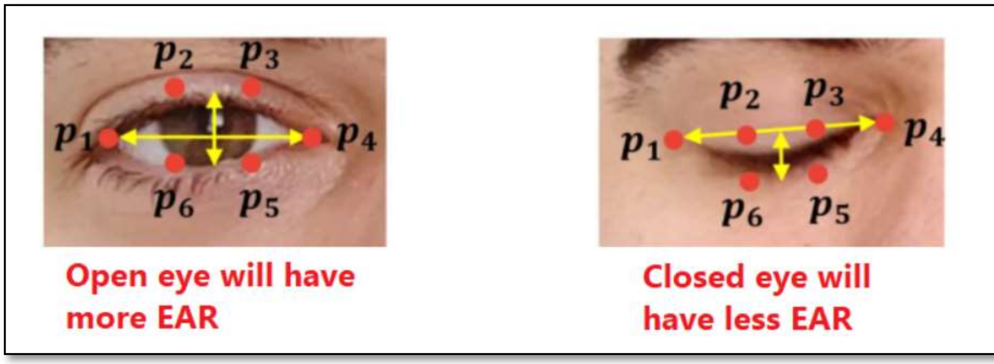

Fig 3.7 Closed and Open Eye EAR

The eye aspect ratio is constant (indicating the eye is open), then rapidly drops to zero, then increases again, indicating a blink has taken place.

CHAPTER 4

EXPERIMENTAL ANALYSIS AND RESULTS

4.1 SYSTEM CONFIGURATION

4.1.1 SOFTWARE REQUIREMENTS

These are the software requirements for running this project.

- Android version 5.1.1 (Lollipop) or above
- Pre-installed OpenCV Manager apk
- Download shape_predictor_68_face_landmark.dat and store in internal storage of the android device

4.1.2 HARDWARE REQUIREMENTS

These are the hardware requirements for running this project.

- Android Phone with
  - minimum SDK 21
  - 1GB RAM or above
  - 32GB Storage or above
  - Minimum 2 MP, f/2.2 Front Camera
- Adjustable Mobile Holder for Car Dashboard

4.2 SYSTEM DESIGN

4.2.1 USE CASE DIAGRAM

In this diagram, the main use case is "Detect Drowsiness." The system detects drowsiness by monitoring several types of data, such as facial recognition and eye-tracking. These use cases are represented by the "Detect Face" and "Detect Eye" use cases respectively

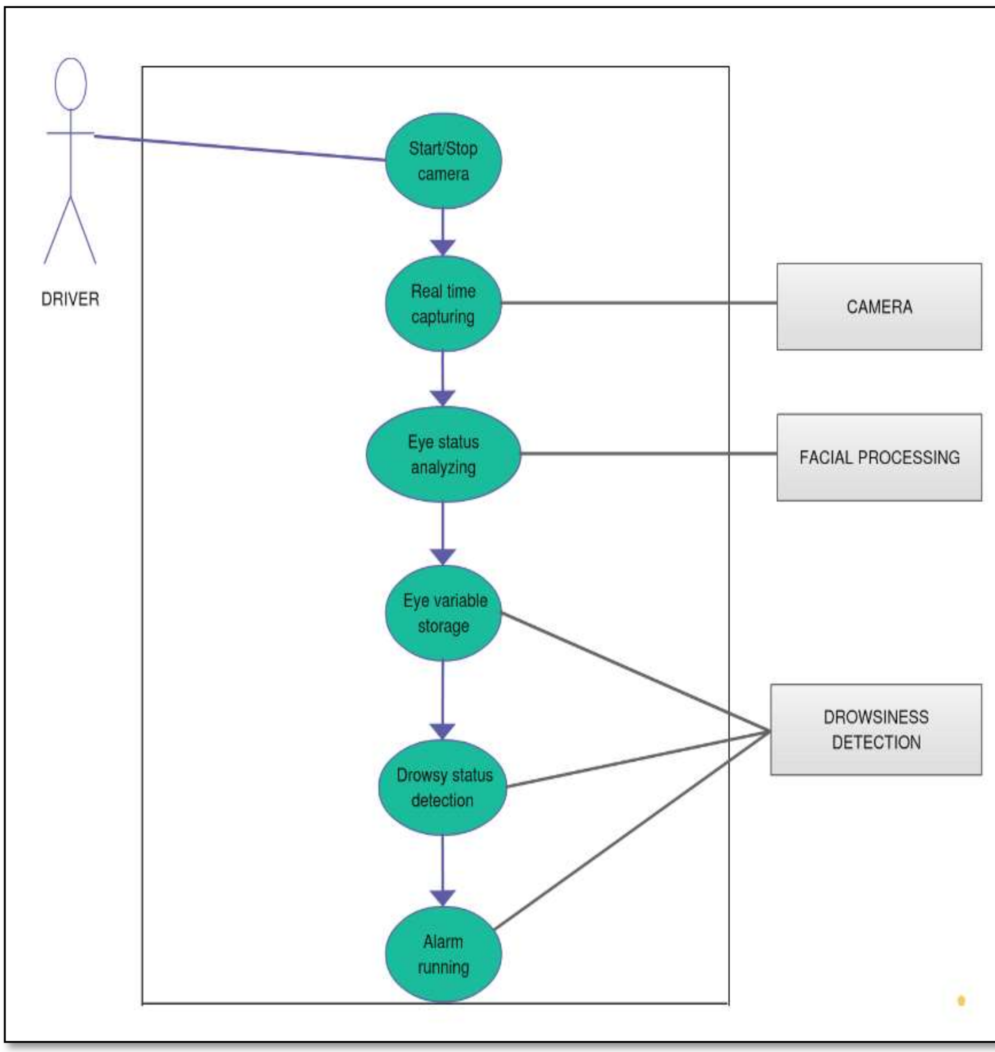

Fig 4.2.1 Use Case Diagram

4.2.2 ACTIVITY DIAGRAM

In this diagram, the system starts by gathering data from various sources, such as a camera to monitor the driver's behavior. Once the data is collected, the system processes it to extract meaningful information such as the driver's facial expressions and eye movements.

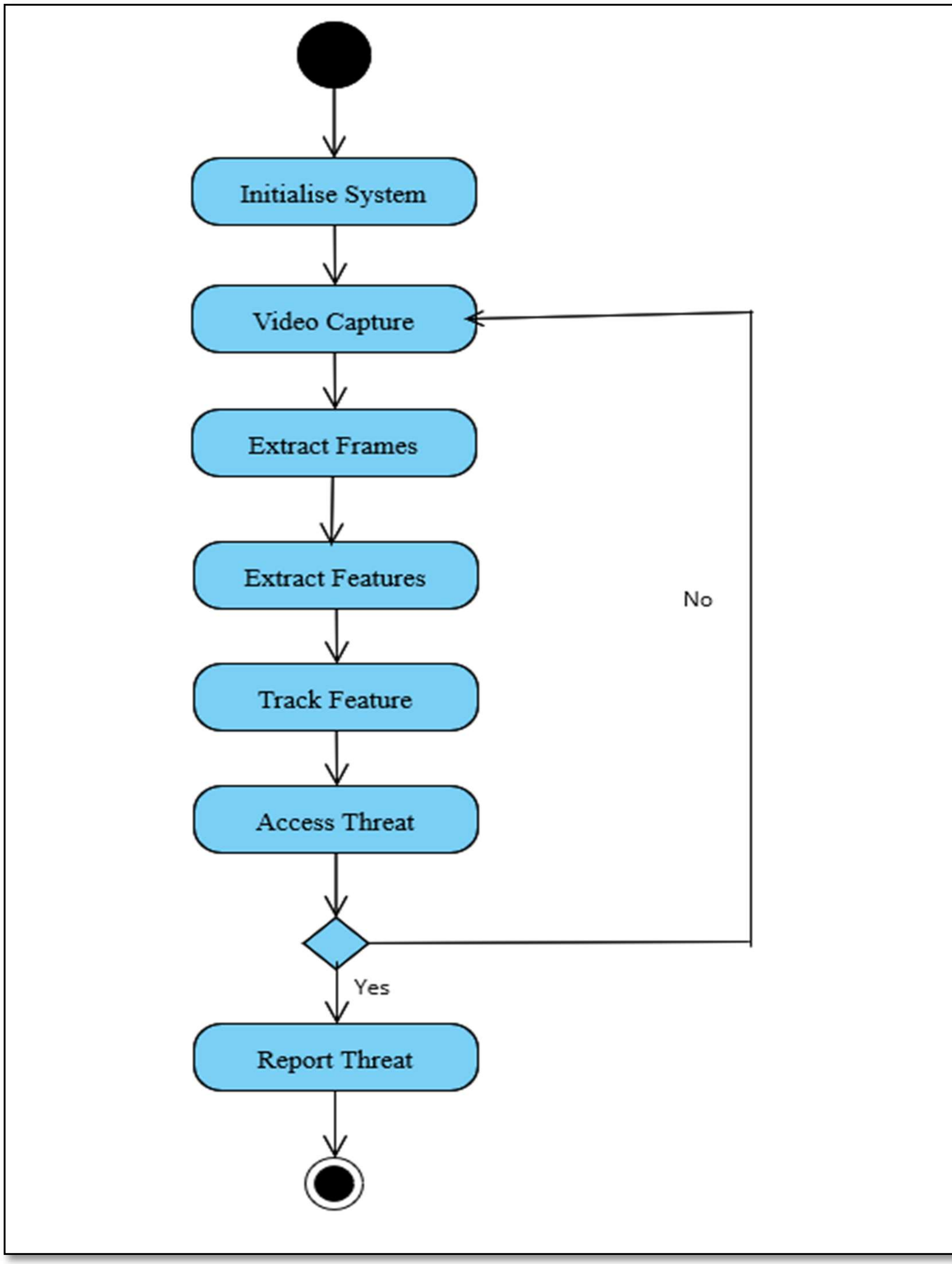

Fig 4.2.2 Activity Diagram

4.2.3 CLASS DIAGRAM

In this diagram, the "DRIVER" class represents the main component of the system. It contains two attributes: "DETECTION SYSTEM" and "ANALYSIS SYSTEM" that monitor the driver's behavior, and "drowsinessThreshold," which is a float that represents the threshold value for detecting drowsiness.

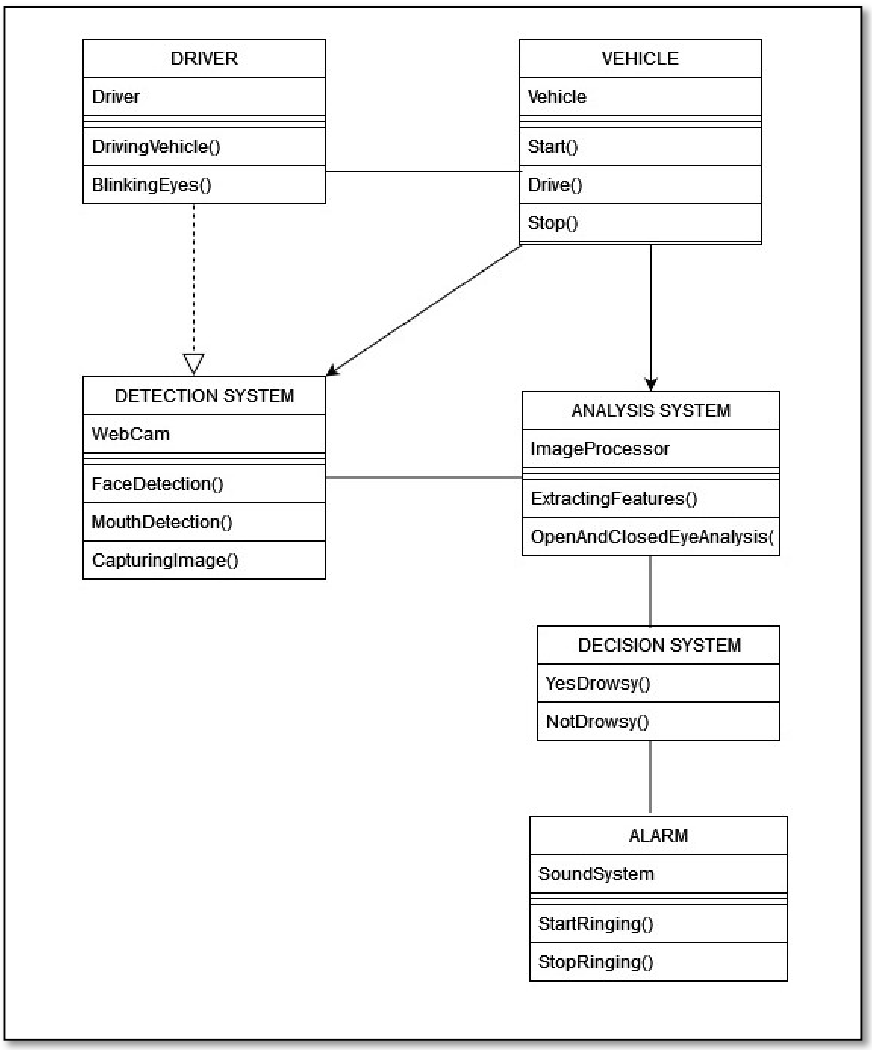

Fig 4.2.3 Class Diagram

4.3 SOURCE CODE

```
from scipy.spatial import distance
from imutils import face_utils
import imutils
import dlib
from threading import Thread
from playsound import playsound
import cv2
import pygame

def eye_aspect_ratio(eye):
A = distance.euclidean(eye[1], eye[5])
B = distance.euclidean(eye[2], eye[4])
C = distance.euclidean(eye[0], eye[3])
ear = (A + B) / (2.0 * C)
return ear

thresh = 0.25
frame_check = 20
detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor("models/shape_predictor_68_face_landmarks.dat")

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
cap=cv2.VideoCapture(0)
flag=0

pygame.mixer.init()
pygame.mixer.music.load('alarm.wav')


while True:
ret, frame=cap.read()
```

```
frame = imutils.resize(frame, width=450)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
subjects = detect(gray, 0)
for subject in subjects:
shape = predict(gray, subject)
shape = face_utils.shape_to_np(shape)
leftEye = shape[lStart:lEnd]
rightEye = shape[rStart:rEnd]
leftEAR = eye_aspect_ratio(leftEye)
rightEAR = eye_aspect_ratio(rightEye)
ear = (leftEAR + rightEAR) / 2.0
leftEyeHull = cv2.convexHull(leftEye)
rightEyeHull = cv2.convexHull(rightEye)

if ear < thresh:
cv2.drawContours(frame, [leftEyeHull],-1, (0,0,255),1)
cv2.drawContours(frame, [rightEyeHull],-1, (0,0,255),1)
flag += 1
if flag >= frame_check:
pygame.mixer.music.play(-1)

else:
cv2.drawContours(frame, [leftEyeHull],-1, (0,255,0),1)
cv2.drawContours(frame, [rightEyeHull],-1, (0,255,0),1)
flag = 0
pygame.mixer.music.stop()

cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
if key == ord("q"):
break
cv2.destroyAllWindows()
cap.release()
```

4.4 SCREEN SHOTS

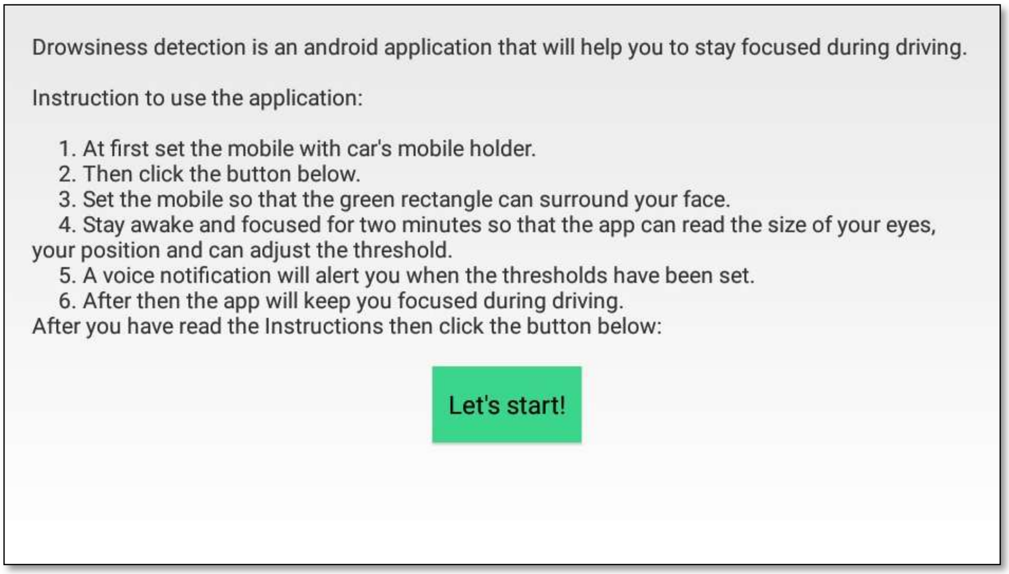

Fig 4.3.1 Instruction Page

Fig 4.3.2 Device Setup

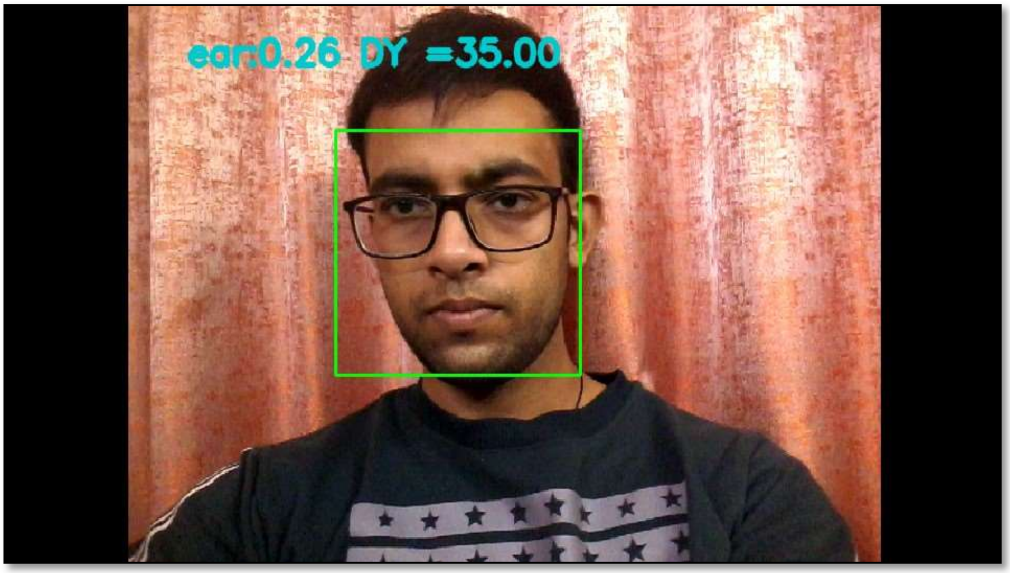

Fig 4.3.3 Setting UP of threshold

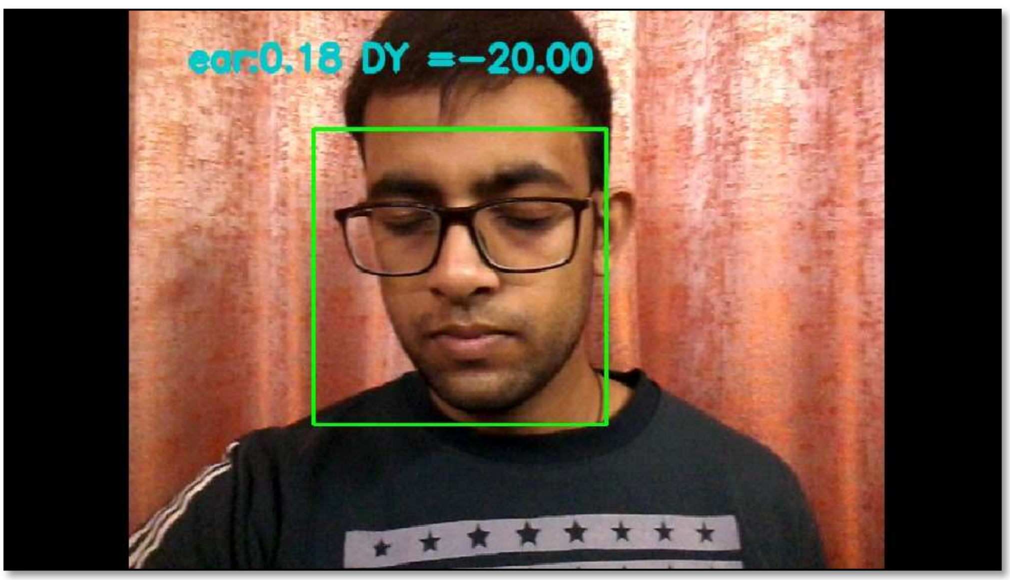

Fig 4.3.4 Threshold Dropped (Drowsiness Detected)

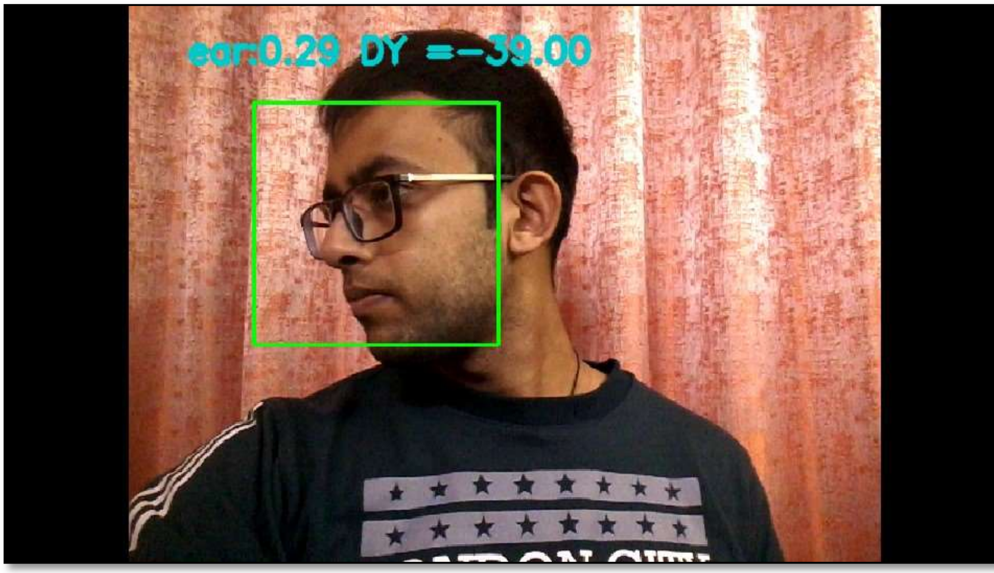

Fig 4.3.5 Eye detection on partial face

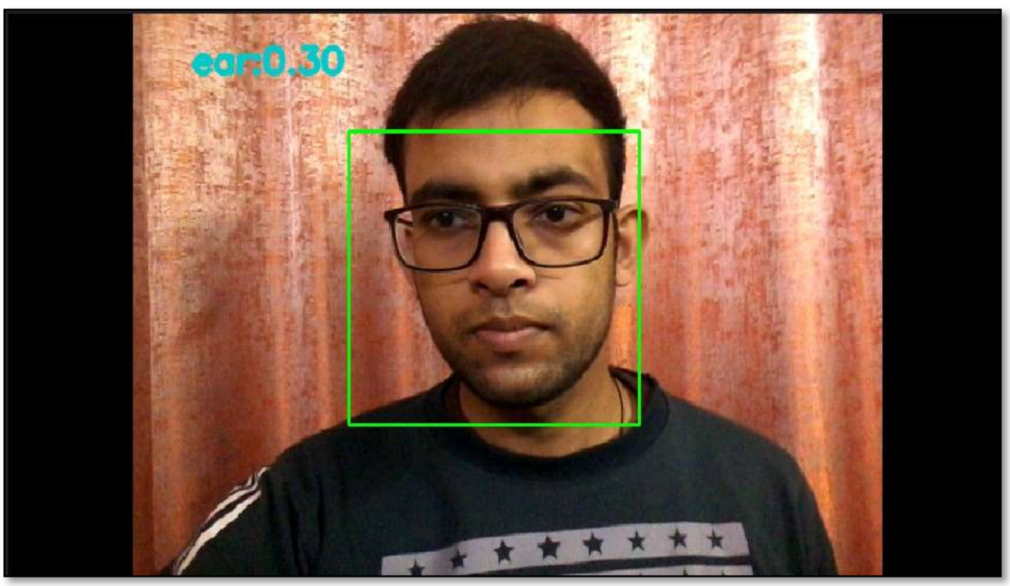

Fig 4.3.6 Face Detection

Fig 4.3.7 Open Eye Detection

Fig 4.3.8 Closed Eye Detection

4.5 TESTING

Testing is done to look for mistakes. Testing is the process of looking for any flaws or weaknesses in a piece of work. It offers a way to test whether parts, subassemblies, assemblies, and/or a finished product are functional. In order to make sure that the software system satisfies its requirements and user expectations, it must be put through a process called software testing. Different test types exist. Each type of test responds to a particular requirement.

4.5.1 UNIT TESTING

The creation of test cases for unit testing verifies that the internal programme logic is working correctly and that programme inputs lead to valid outputs. It is important to verify the internal code flow and all decision branches. It is the testing of the application's individual software components. Before integration, it is done following the completion of each individual unit. This type of intrusive structural testing requires an understanding of how it was built. Unit tests run basic tests at the component level and test a particular setup for a system, application, or business. Each individual Princess path is tested as a unit to make sure it adheres exactly to the documented specifications and has inputs and outputs that are well-defined.

4.5.2 INTEGRATION TESTING

Integration tests are created to check whether integrated software parts function as a single program. Event-driven testing is more focused on the fundamental outcome of screens or fields. Integration tests show that even though the individual components were satisfactory, the combination of the components is accurate and consistent, as successfully demonstrated by unit testing. Integration testing is especially designed to highlight issues that result from combining components.

4.5.3 FUNCTIONAL TESTING

Functional test provides systematic demonstrations that function tests are available as specified by the business and technical requirement, system documentation and user manuals.
Functional testing is centered on the following items:

Valid input: identify classes of valid input must be accepted,

Invalid Input: Identify classes of invalid inputs must be rejected,

Functions: Identified be exercised identities function must be exercised,

Output: identify classes of application outputs must be exercised,

Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functions test is focused on requirements, key functions or special test cases. In addition, systematic coverage pertaining to identify business process flow; data fields, predefined process and successive process must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current test is determined.

4.5.4 SYSTEM TEST

System testing makes sure that the integrated software system as a whole complies with specifications. In order to ensure known and predictable outcomes, it tests a configuration. Configuration-oriented system integration testing is an illustration of on-site testing.

4.5.5 WHITE BOX TESTING

It is a type of testing where the software tester is familiar with the inner workings, language, and structure of the software, or at the very least its intended use. It is employed to test regions that are inaccessible from a black box level.

4.5.6 BLACK BOX TESTING

White Box without having any prior knowledge of the inner workings, structure, or language of the module being tested, testing the software. Black box tests, like the majority of other types of tests, must be created from a clear source document, like a set of specification requirements. The software being tested is treated as a "black box" that you cannot see inside. Without considering how the software functions, the test generates inputs and responds to outputs.

4.6 TEST PLAN

A report outlining the objectives, strategy, available tools, and timetable for the planned testing activities. It lists test items, features to be tested, testing tasks, who will complete each task, degree of tester independence, test environment, test design techniques, entry and exit criteria to be used, and the justification for their selection, among other things. It also lists any risks that call for contingency planning. It serves as a log of the test preparation procedure.

To create a test plan in accordance with IEEE 829, follow the steps below.

4.6.1 ANALYZE THE SYSTEM

Only when the tester is familiar with a system or product—that is, when they know how it functions, who its users are, what software or hardware it uses, what it is used for, etc.—can they analyze it.

4.6.2 DESIGN THE TEST STRATEGY

Figuring out the costs and efforts necessary to accomplish the system's goals in order to design a test strategy for every type of functioning hardware.

The test strategy can be created by anyone for any project.

i. Defining the scope of the testing.
ii. Identifying the type of testing required.
iii. Risks and issues
iv. Creating test logistics.

4.6.3 DEFINE THE TEST OBJECTIVES

The overall purpose and success of the test execution is the test objective. The system must be free of bugs and ready for use by end users before objectives can be defined. The goal of the test and the software features that must be tested in order for it to be successful can be used to define the test objective.

4.6.4 DEFINE TEST CRITERIA

A test procedure or test judgement can be based on test criteria, which are standards or rules. Two such test criteria are as follows: Exit criteria, which outlines the requirements that signify a test phase's successful conclusion, and suspension criteria, which state that if a predetermined number of test cases fail, the tester should halt all ongoing test cycles until the criteria is met.

4.6.5 RESOURCE PLANNING

The project task's resource requirements are listed in detail in the resource plan. Resources include people, tools, and supplies needed to finish a project.

4.6.6 PLAN TEST ENVIRONMENT

A testing environment is a configuration of hardware and software where the testing team will run test cases.

4.6.7 SCHEDULE AND ESTIMATION

To reduce the chance of failing to finish the project by the deadline, it is essential to create a schedule for the various testing phases and estimate the amount of time and labor required to test the system. The creation of the test specification, test execution, test reporting, and test delivery is all included.

4.6.8 DETERMINE TEST DELIVERABLES

Deliverables are the materials that must be created, updated, and maintained to support the testing effort. Deliverables for testing are offered prior to, during, and following the testing phase.

The following test plan and various scenarios should be taken into account for this project:

A. This project was created with the intention of identifying driver distraction and sleepiness while operating a non-two-wheeled vehicle.

B. Various project modules, such as face, eye, and face-and-eye tracking, drowsiness, and distraction detection, are all put through unit and functional testing. Through functional testing, determine whether the system is able to extract the necessary features of the face after the face has been detected. API testing and database testing are not covered by the test plan because this project lacks any kind of API. Currently, there will be no testing of nonfunctional aspects like stress, performance, or logical databases. At last, the entire system is tested through system testing for the alert while the driver is drowsy/distracted.

C. Every team member in the project group is introduced to the testing process to reduce the risks of anyone on the team being unable to understand the testing. Two members of this project group have been given the task of testing this system in order to expedite the process. The functional and system testing of this system were carried out by Yash Chaturvedi, Tanishka Gupta, Prakhar Chaurasia and Rashi Pandey, respectively.

D. The testing cycle for this project is suspended if 50% of the test cases are deemed unsuccessful, meaning that Yash Chaturvedi and Prakhar Chaurasia, who make up the development team, have made the necessary code improvements. 95% of the test cases need to pass in order to pass the exit criteria. Under both poor and good lighting conditions, we were successful.

E. Two members of the project team made up the resource planning for testing: Tanishka Gupta, who identified various testing scenarios and can be regarded as the test developer or administrator, and Rashi Pandey, who carried out the tests, recorded the results, and alerted the project team to any flaws.

F. The schedule for this project is as follows:

   a) The test specification for this project is created by Yash Chaturvedi which comprised of testing the different modules of the project under different conditions and excluding certain scenarios.

   b) The test execution is performed by Prakhar Chaurasia which included executing all various test cases provided by Yash Chaturvedi. Logging the results and reporting the outcomes along with defects is performed by her carefully.

   c) The test reports are generated accordingly by both the members of the testing team. Defects are dealt accordingly by the development team which included Tanishka Gupta and Rashi Pandey. Later again these test cases have been tested and are proved successful in achieving the correct results.

4.7 TEST REPORT

The system is tested for a variety of test cases under various circumstances, considering the majority of the potential scenarios. Some of the test reports have been listed below, just to name a few.

Table 4.7.1 (Unit testing)

Test Case:	UTC-1
Name of the Test:	Face Detection
Item Tested:	Face
Sample Input:	Video of the person with face included
Expected Output:	Face Detection in the frame
Actual Output:	Same as expected Output
Remarks:	Successful

Table 4.7.2 (Unit testing)

Test Case:	UTC-2
Name of the Test:	Eye Detection
Item Tested:	Eye
Sample Input:	Driver's Face
Expected Output:	Eye Detection
Actual Output:	Same as expected Output
Remarks:	Successful

Table 4.7.3 (Functional testing)

Test Case:	FTC-1
Name of the Test:	Drowsiness Detection
Item Tested:	Driver's Eyes
Sample Input:	Face and eyes of the driver
Expected Output:	Drowsiness Detection
Actual Output:	Same as expected Output
Remarks:	Successful

Table 4.7.4 (Integration Testing)

Test Case:	ITC-1
Name of the Test:	System Alert
Item Tested:	Alarm
Sample Input:	Drivers face in drowsy condition
Expected Output:	Alert in form of alarm
Actual Output:	Same as expected Output
Remarks:	Successful

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

An automated system for detecting driver drowsiness was developed in the current study. Drowsiness is determined by reading the continuous video stream from the system. Using the HAAR cascade algorithm, it is discovered. HAAR features are used by the HAAR cascade algorithm to find faces and eyes. Predefined HAAR features are employed to detect a variety of things. The HAAR features are applied to the image, and the perclos algorithm is used to determine blink frequency. If the value stays at 0, it recognizes that the user is sleepy and alerts the driver by sounding an alarm. The driver is considered to be distracted and an alarm is set off if the value stays the same for extended periods of time.

5.2 FUTURE WORK

By removing the features of the mouth where the driver can be identified as being sleepy through yawning, the work can be furthered. The driver is in a sleepy mode if he or she yawns frequently and for an extended period of time. We can notify the driver if the number goes over a certain threshold. Utilizing an IR webcam to implement a full night light will further this work. The camera uses infrared radiation to determine whether or not the subject is sleepy. Even though this is a research project, there is potential for it to fully develop into an application that end users can run independently for their own purposes on their own systems.

CHAPTER 6

APPENDIX

1. PYTHON

Guido Van Rossum's Python is an interpreted, high-level, general-purpose programming language. It was first released in 1991, and its design philosophy emphasizes code readability through the use of notable amounts of whitespace. Its language constructs and object-oriented methodology are designed to aid programmers in creating clean, comprehensible code for both little and big projects. Python has garbage collection and dynamic typing. Programming paradigms like procedural, object-oriented, and functional programming are all supported.

2. JUPYTER

An open-source web application called the Jupyter Notebook enables you to create and share documents with real-time code, equations, visuals, and text. It can be used for a variety of things, such as machine learning, statistical modelling, data visualization, and data cleaning and transformation. Jupyter notebooks are the best environment for capturing programming style. You can work on individual lines of code, write out your ideas, and then put everything together once you're finished. They are excellent for experimental, iterative programming.

3. OPENCV

OpenCV is a collection of prioritized functions primarily aimed at real-time dream computing. OpenCV was initially developed by Intel and later by emerald in color office before it was seen. The open-source BSD license allows for free use of the study across all platforms. According to the list of supporting coatings, it supports a few models from deep knowledge foundations, such as TensorFlow, PyTorch, and Caffe. OpenCV promotes Open Vision Capsules, a practical layout, and agreeable supplementary additional layouts.

4. SCIPY

The SciPy bundle contains a number of toolboxes devoted to common problems in controlled computing. Its numerous submodules serve a variety of purposes, including introduction, unification, growth, image preparation, enumerations, distinguished functions, and all other possessions. The NumPy library, which offers convenient and quick N- spatial array guidance, is a prerequisite for the SciPy athenaeum.

The SciPy library is built to work with NumPy arrays and supports a wide range of intuitive and skillful mathematical techniques, including grooves for numerical addition and fusion. Together, they are active in establishing and free, gossiping about all popular operating orders. Using NumPy and SciPy is simple. SciPy's basic dossier format is a complex array supported by each NumPy module. Few functions for linear algebra, Fourier transforms, and random number generation are specified in NumPy. These functions are not generalized in the same way as their SciPy counterparts, but they are strong enough to be accepted by a select group of the world's top physicists and engineers.

5. NUMPY

NumPy is a python prioritize terminology athenaeum that adds support for numerous multi-spatial arrays and origins as well as a generous selection of extremely high-level analytical functions to operate on these arrays. Jim originally developed Numeric, the ancestor of NumPy, with assistance from a number of developers.
Travis created NumPy in 2005 by integrating the fighting Numarray into Numeric along with significant modifications. The open-source operating system NumPy has a large user base.

6. DLIB

Dlib is a modern C++ toolkit that includes machine learning algorithms and features for developing complex software in C++ to answer immediate problems.

It has a wide range of applications in both manufacturing and education, including robotics, stationary instruments, mobile phones, and highly advanced computing environments. Dlib’s open-source licensing allows for free use in any application. You can use the study right away; no establishment or construct step is required. This package offers generally to representation processing methods in addition to other applications of Dlib.

7. ANDROID STUDIO

Built on JetBrains' IntelliJ IDEA software and created specifically for Android development, Android Studio is the recognized integrated development environment (IDE) for Google's Android platform.

On Windows, macOS, and Linux-based operating systems, it can be downloaded. You can create apps for Android phones, tablets, Android Wear, Android TV, and Android Auto using Android Studio, which offers a unified development environment. You can divide your project into functional units that you can independently build, test, and debug using structured code modules.

8. PYGAME

A Python covering piece for the SDL combined use of various media studies is called Pygame. It has Python classes and functions that allow you to use SDL's support for various computer storage tasks, radio and television broadcasts of visual and auditory entertainment, as well as input from a keyboard, mouse, and joystick.

CHAPTER 7

REFERENCES

[1] Assari, M. A., & Rahmati, M. (2011). Driver drowsiness detection using face expression recognition. 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA).

[2] Tianyi Hong, Huabiao Qin, & Qianshu Sun. (2007). An Improved Real Time Eye State Identification System in Driver Drowsiness Detection. 2007 IEEE International Conference on Control and Automation.

[3] Warwick, B., Symons, N., Chen, X., & Xiong, K. (2015). Detecting Driver Drowsiness Using Wireless Wearables. 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems.

[4] Dwivedi, K., Biswaranjan, K., & Sethi, A. (2014). Drowsy driver detection using representation learning. 2014 IEEE International Advance Computing Conference (IACC).

[5] Yan, J.-J., Kuo, H.-H., Lin, Y.-F., & Liao, T.-L. (2016). Real-Time Driver Drowsiness Detection System Based on PERCLOS and Grayscale Image Processing. 2016 International Symposium on Computer, Consumer and Control (IS3C).

[6] Alshaqaqi, B., Baquhaizel, A. S., Amine Ouis, M. E., Boumehed, M., Ouamri, A., & Keche, M. (2013). Driver drowsiness detection system. 2013 8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA).

[7] Tripathi, D.P., Rath, N.P. (2009). A novel approach to solve drowsy driver problem by using eye-localization technique using CHT. International Journal of Recent Trends in Engineering.

[8] Subbarao, A., Sahithya, K. (2019) Driver Drowsiness Detection System for Vehicle Safety, International Journal of Innovative Technology and Exploring Engineering (IJITEE).

[9] Sukrit Mehta, Sharad Dadhich, Sahil Gumber, Arpita Jadhav Bhatt (2019). Real-Time Driver Drowsiness Detection System Using Eye Aspect Ratio and Eye Closure Ratio International Conference on Sustainable Computing in Science, Technology and Management.

[10] Tayab Khan, M., Anwar, H., Ullah, F., Ur Rehman, A., Ullah, R., Iqbal, A., … Kwak, K. S. (2019). Smart Real-Time Video Surveillance Platform for Drowsiness Detection Based on Eyelid Closure. Wireless Communications and Mobile Computing, 2019.

[11] RamalathaMarimuthu, A. Suresh, M. Alamelu and S.Kanagaraj "Driver fatigue detection using image processing and accident prevention", International journal of pure and applied mathematics, vol. 116, 2017.

[12] Omkar, RevatiBhor, PranjalMahajan, H.V. Kumbhar "Survey on Driver's drowsiness detection system", vol.132, 2015.

[13] Rajasekar.R, Vivek Bharat Pattni, S.Vanangamudi "Drowsy driver sleeping device and driver alert system", IJSR, Vol.3 Issue4,2014.

[14] S. Podder and S. Roy, "Driver's drowsiness detection using eye status to improve the road safety," International Journal of Innovation Research in Computer and Communication Engineering, vol. 1, no. 7, 2013.

[15] I. García, S. Bronte, L. M. Bergasa, J. Almazán, and J. Yebes, (2012). "Vision-based drowsiness detector for real driving conditions," IEEE Intelligent Vehicles Symposium,Proceedings.

PLAGIARISM REPORT

Plagiarism Scan Report

Report Generated on: May 12,2023

0% Plagiarized	100% Unique

Total Words:	324
Total Characters:	1948
Plagiarized Sentences:	0
Unique Sentences:	15 (100%)

Content Checked for Plagiarism

Driver drowsiness is a common cause of accidents It is currently one of the primary causes of road accidents. Many accidents were caused by drowsy driving, according to the most current statistics. Each year, thousands of people are killed in car accidents caused by tired drivers. Drowsiness causes more than 30% of accidents. To avoid this and preserve lives, a system that detects drowsiness and notifies the driver is required.

In this project, we present a method for detecting driver weariness based on an eye aspect ratio (EAR). A smartphone's front-facing camera is used to capture the driver's face and extract facial landmarks using a face detection model and a landmark model, while a webcam is utilized to continually observe the driver. This model employs image processing technologies that highlight the driver's face and eyes in particular When a person with large eyes is drowsy, the size of his or her eyes will be different from the size of the eyes of a person with small eyes since the human eye has a different size in a normal physical state. As a result, under the system of this project, each person's ocular reference was built especially for them.

The model extracts the driver's face and predicts eye blinking based on the location of the eyes. The EAR value from the startup procedure for each driver is used as a baseline for determining whether a driver's eyes are closed or open based on the EAR threshold. If the EAR value (from the driving procedure) goes below the threshold several times in a row (with a reference of 1.5 seconds), the system will detect drowsiness. If the blinking rate becomes excessively rapid, the system alerts the driver with a sound. The project is designed to be lightweight, fast, and accurate, and to function on smartphones without the need for additional hardware or an internet connection.

Keywords — Drowsiness, Distraction, Eye detection, Eye Tracking, Face Detection, Perclos.

No Plagiarism Found

AUTHOR'S DETAILS

Name & Roll No.: Yash Chaturvedi (1903480100128)

Mobile: +91 6307819311

Email: yash.chaturvedi2001@gmail.com

Address: E – 272 Panki, Kanpur, Uttar Pradesh – 208020

Name & Roll No.: Tanishka Gupta (1903480100113)

Mobile: +91 8858460399

Email: thetanishkagupta@gmail.com

Address: 128/541 Y Block Kidwai Nagar, Kanpur, Uttar Pradesh – 208011

Name & Roll No.: Prakhar Chaurasia (1903480100076)

Mobile: +91 7905765424

Email: prakharchaurasia991@gmail.com

Address: C – 284 Swaraj Nagar, Panki, Kanpur, Uttar Pradesh – 208020

Name & Roll No.: Rashi Pandey (1903480100081)

Mobile: +91 8795080746

Email: rashipandey2002@gmail.com

Address: 5/136 Mahesh Marg, Shuklaganj, Unnao, Uttar Pradesh - 209861