A Project Report On

Vehicle Alarm System based on Driver Drowsiness and Speed

Submitted in the partial fulfillment of the

requirement for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

"COMPUTER SCIENCE AND ENGINEERING"

Submitted by:

Anil Kr. Sonkar (2016021014)

Anshika Kankane (2016021019)

Jitendra Chaudhari (2016021118)

Project Guide:

Muzammil Hasan

Assistant Professor

CSED, MMMUT

Department of Computer Science and Engineering

Madan Mohan Malaviya University of Technology

Gorakhpur - 273010 (U.P.)

May 19, 2020

CANDIDATE'S DECLARATION

We declare that this project report presents our work and ideas in our own words and where other ideas or words have been included, we adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misprinted or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom paper permission has not been taken when needed.

Anil Kr. Sonkar (2016021014)

Anshika Kankane (2016021019)

Jitendra Chaudhari (2016021118)

Computer Science & Engineering

Madan Mohan Malaviya University of Technology

Date: _______________

CERTIFICATE

This is to certify that ***Anil Kr. Sonkar***, ***Anshika Kankane*** **and** ***Jitendra Chaudhari*** have carried out the project work presented in this report entitled **—** "***Vehicle Alarm System based on Driver Drowsiness and Speed***" for the partial fulfillment of Bachelor of Technology from Madan Mohan Malaviya University of Technology, Gorakhpur under my supervision. The report embodies results of original work and studies carried out by students themselves to the best of my knowledge and the contents of this report do not form the basis for the award of any other degree to the candidates or to anybody else.

Muzammil Hasan
Assistant Professor
Computer Science & Engineering
M.M.M.U.T. Gorakhpur

Date: ____________

APPROVAL SHEET

This report entitled **“Vehicle Alarm System based on Driver Drowsiness and Speed”** by **Anil Kr. Sonkar (2016021014), Anshika Kankane (2016021019), Jitendra Chaudhari (2016021118)** is approved for the degree of Bachelor of Technology.

Examiner

____

____

____

Supervisor

Muzammil Hasan

Assistant Professor

Head of Department

Dr. P K Singh

Dean, Research & Development or

Other Dean/Professor to be nominated

by the Vice Chancellor in his absence

____

Date: ________________

Place: Gorakhpur

ACKNOWLEDGEMENT

We would like to make our deepest appreciation and gratitude to our supervisor, Mr. Muzammil Hasan for his invaluable guidance, constructive criticism, encouragement, professional knowledge, help and useful suggestions, continuous encouragement and supervision throughout the course of the present work.

I also take this opportunity to express a deep sense of gratitude to Dr. P K Singh (Head of Dept., CSE, MMMUT) for his cordial support, valuable information and guidance which helped me in completing this task through various stages.

Grateful acknowledgement is made to all the staff and faculty members of the Computer Science and Engineering Department for their encouragement and technical help for the successful completion of this project. Our thanks extend to all fellow students for their time, invaluable suggestions and help. In spite of numerous citations above, we accept full responsibility for the content that follows.

ABSTRACT

Driver drowsiness and speeding are the main causes of accidents in the world as well as in India. Due to lack of sleep and tiredness, drowsiness can occur while driving. The best way to avoid such circumstances and accidents caused by drowsiness is to detect drowsiness of the driver and warn him before he/she falls into sleep. On the other hand, speeding occurs when the engine of the vehicle is forced to move above certain specified limit and the best way for this is to warn the driver as soon as he reaches above the certain defined limit on each location. To detect these factors many techniques have been developed separately.

In this project, we propose a method of detecting driver drowsiness by the method of face/object detection and detection of speed on each location by using Google Maps Roads API. We have designed a system that comprises of detection both the features in real time. Further, the changes would be done according to the experimental results obtained.

Keywords: Driver Drowsiness, Speeding, Face detection, Speed detection, Roads API, OpenCV.

LIST OF FIGURES

Fig no.	Figure Title	Page no.
Figure 1	Practical representation of positions of system	10
Figure 2	Drowsiness detection system architecture	16
Figure 3	Flowchart for drowsiness detection	17
Figure 4	Haar type features	18
Figure 5	Image area integration using Integral Image (II)	18
Figure 6	Eye region pixels used for calculating EAR	19
Figure 7	Flowchart for speed detection and calculation	21
Figure 8	When the driver is awake (eyes open)	27
Figure 9	When the driver is sleepy (eyes closed)	27
Figure 10	When the driver is awake (eyes open)	28
Figure 11	When the driver is sleepy (eyes closed)	28
Figure 12	When the driver is awake (eyes open)	29
Figure 13	When the driver is sleepy (eyes closed)	29

TABLE OF CONTENTS

		Candidate's Declaration	iii
		Certificate	iv
		Aproval Sheet	v
		Acknowledgment	vi
		Abstract	vii
		List of Figures	viii
Chapter	**1**	**Introduction**	5
		1.1 Need for Drowsiness Detection Alarm System	5
		1.2 Need for Speed Detection Alarm System	6
		1.3 Drowsiness Detection Technologies in Market	7
Chapter	**2**	**Problem Definition**	8
		2.1 Objective	8
		2.2 Scope	9
Chapter	**3**	**Hardware & Software Requirements**	10
		3.1 Hardware Requirements	10
		3.2 Software Requirements	11
Chapter	**4**	**Methodology**	12
		4.1 Used Technologies/Softwares	12
		4.2 Used Libraries/Packages	14
		4.3 Prototype for drowsiness detection	16
		4.4 Steps to design the ETSDD	17
		4.5 Steps to design the Speed Alert System	21

		4.5 Quality Requirements	26
Chapter	**5**	**Results & Practical Implementation**	27
		5.1 Screenshots of website	27
		5.2 Result	30
Chapter	**6**	**Conclusion & Future Scope**	31
		6.1 Conclusion	31
		6.2 Future Scope	31
		References	32

CHAPTER -1

INTRODUCTION

Driver drowsiness and speeding are the major factors in vehicle accidents. Speed has been identified as a key risk factor in road traffic injuries, influencing both the risk of a road crash as well as the severity of the injuries that result from crashes. According to the Indian road accidents survey, every year there are more than 135,000 incidents of road accidents. Out of these, most of them are due to rash driving. According to Indian Constitution, IPC section 279, rash driving is an offence.

Drowsiness is a state where a person sleep or almost likely. It alludes to a failure to keep awake or a drive to rest. It alludes to a powerlessness to keep wakeful or a drive to sleep. Latest statistics evaluate that every year 1550 deaths and 71,000 injuries can be ascribed to accidents due to the driver fatigue. The advancement of technology for recognizing or forestalling drowsiness in the driver's seat is a noteworthy challenge in the field of accident evasion frameworks. It would in this manner be advantageous to figure out how to recognize drowsiness before it happens and to have the capacity to caution the driver in time. A few systems already had been created, considering recording of head developments, movement of steering wheel, heart rate variability or grip quality. The algorithm of eye detection system integrated with hardware to develop the smart vehicles, which can implement nationwide to avoid the road accidents. Microcontroller and camera are used to make and intelligent hardware and software integrated system. By checking the driver's eyes, the indications of driver drowsiness can be identified in the beginning to protect from vehicle accident.[1]

1.1 Need for Drowsiness Detection Alarm System

Driving while drowsy is like driving under influence of alcohol. The driver is three times more likely to be in a car crash if he is fatigued. Drowsy driving affects everyone, but especially those under age of 25, who make up an estimated 50% or more of drowsy driving crashes.

Some of the statistical data regarding accidents involving driver drowsiness/fatigue is enlisted below-

- According to a study by the Central Road Research Institute (CRRI), in July 2019 there were 40% of road accidents that occurred due to exhausted drivers who doze off at the wheel on the 300 km Agra-Lucknow Expressway.
- NHTSA reports in 2017 there were 795 fatalities in motor vehicle crashes that involved drowsy drivers.
- In 2017, there were 91,000 police-reported crashes that involved drowsy drivers. Those crashes led to about 50,000 people being injured.
- The Governors Highway Safety Association issued a report in August 2016 concluding that the estimated annual societal cost of fatigue-related fatal and injury crashes was $109 billion.
- In 2017, 1,306 drivers who were involved in fatal crashes (or almost 3 percent) were reported as being drowsy.[2]

1.2 Need for Speed Detection Alarm System

Excess and inappropriate speed are responsible for a high proportion of the mortality and morbidity that result from road crashes. In high-income countries, speed contributes to about 30% of deaths on the road, while in some low-income and middle-income countries, speed is estimated to be the main contributory factor in about half of all road crashes.

Below are some factors on how speed affects traffic collisions and injuries:

- The higher the speed of a vehicle, the shorter the time a driver has to stop and avoid a crash. A car travelling at 50 km/h will typically require 13 metres in which to stop, while a car travelling at 40 km/h will stop in less than 8.5 metres.
- An increase in average speed of 1 km/h typically results in a 3% higher risk of a crash involving injury, with a 4–5% increase for crashes that result in fatalities.
- Speed also contributes to the severity of the impact when a collision does occur. For car occupants in a crash with an impact speed of 80 km/h, the likelihood of death is 20 times what it would have been at an impact speed of 30 km/h. [3]

Some of the data regarding road accidents due to over speeding is mentioned below:

- Speeding was responsible for 116 fatal road accidents in the State capital during the first nine months of year 2019, while 12 were caused by drunk drivers.
- Till the end of September 2018, lives lost in the accidents was 232 in Hyderabad. [4]

- More than 97k people died due to accidents caused by **speeding** in 2018.
- The figure accounts for 64.4% of the total deaths in **India**, according to road transport and highways ministry report on 'Road Accidents in **India**, 2018'. [5]

1.3 Drowsiness Detection Technologies in Market

A driver who falls asleep at the wheel loses control of the vehicle, an action which often results in a crash with either another vehicle or stationary objects. In order to prevent these devastating accidents, the state of drowsiness of the driver should be monitored. The following measures have been used widely for monitoring drowsiness:

- Vehicle-based measures—A number of metrics, including deviations from lane position, movement of the steering wheel, pressure on the acceleration pedal, *etc.*, are constantly monitored and any change in these that crosses a specified threshold indicates a significantly increased probability that the driver is drowsy.
- Behavioral measures—The behavior of the driver, including yawning, eye closure, eye blinking, head pose, *etc.*, is monitored through a camera and the driver is alerted if any of these drowsiness symptoms are detected.
- Physiological measures—The correlation between physiological signals (electrocardiogram (ECG), electromyogram (EMG), electrooculogram (EoG) and electroencephalogram (EEG)) and driver drowsiness has been studied by many researchers.

Other than these three, researchers have also used subjective measures where drivers are asked to rate their level of drowsiness either verbally or through a questionnaire. The intensity of drowsiness is determined based on the rating.[6]

CHAPTER -2

PROBLEM DEFINITION

The number of traffic accidents has increased from year to year in India. Factors affecting road traffic accidents vary, ranging from driver factors, vehicle factors, and environmental factors. Most researchers have agreed that driving is activities that can cause both physical and mental fatigue for drivers. Fatigue due to transportation can be caused by many factors, including factors related condition in the transportation, health factors, psychosocial factors and even psychological factors. Some approaches use the measurement of physiological activities of the human body such as brainwave (using electroencephalogram - EEG), heart rate (using electrocardiogram – ECG) and electric signals from muscle cells (using electromyogram – EMG). The measured signal is usually reliable enough to detect drowsiness because its correlation with driver alertness is relatively strong. However, this technique has some limitations due to the necessity to apply tools to the user's body which is somewhat intrusive. Driver's fatigue-related behavior status can also be detected on the basis of vehicle behaviors, such as steering movements, accelerations and braking, a speed of vehicles, and lane crossings. Unfortunately, this approach can only be used for very limited conditions, because it could be a sudden movement that occurs due to road conditions, not because of the sleepy condition of the driver.[7]

2.1 Objective

This project aims at an approach to estimate driver drowsiness by employing a set of cameras and image processing technique to detect changes in driver behavior, such as movement of eyes and blinking of eyes, along with an additional feature of estimating the speed of the vehicle.

The system generates an alarm in two cases:

- if driver closes his/her eyes for more than 3 sec or if the blinking is at a faster rate.
- if the current speed of the vehicle is above the speed limit allowed on the particular location.

2.2 Scope

The scope of this project is to determine the eye behaviour and facial expressions of the driver to detect if the driver is drowsy or not. Also, this system in future will be designed to detect the speed of the vehicle and compare this speed with the maximum speed limit based on the position of the vehicle. This system is designed to generate an alarm when either of the above cases are fulfilled. This can result in the decrement of the number of accidents happening in our day to day life and thus improve in road safety.

This alarm system will result in the decrement in the number of incidents and crashes on the roads which will result in the improved road safety. This system is highly economic and easy to install saving space, money and most importantly human lives. Also, the system is very user friendly and easy to operate.

This system is very useful for the drivers who have to drive for long durations and without taking much breaks. In the future, this system may be expanded to trains and buses all over the world as these are the transports used for long journeys and for longer durations.

CHAPTER -3

HARDWARE & SOFTWARE REQUIREMENTS

3.1 Hardware Requirements

- Processor – Intel i3 or above/ Smartphone
- Hard disk – 128 GB or more
- Memory – 4 GB or above
- Web Camera
- System with GPS

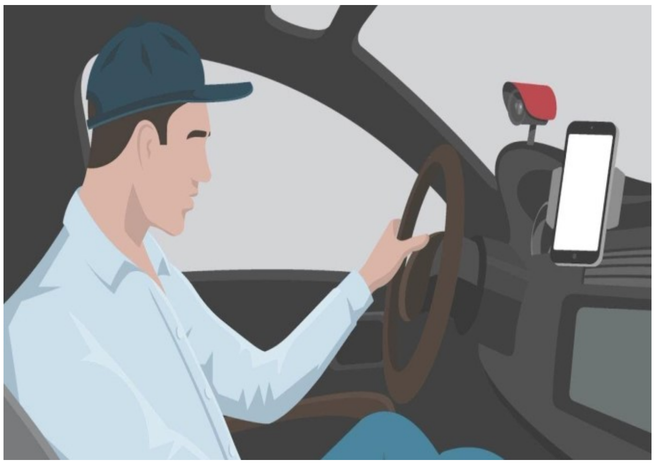


Figure 1: Practical representation of positions of system

3.2 Software Requirements

3.2.1. Web Browser

A web browser is a software application for accessing information on the World Wide Web. Each individual web page, image, and video is identified by a distinct URL, enabling browsers to retrieve and display them on the user's device.

Common user interface features of browsers:

- Back and forward buttons to go back to the previous page visited or forward to the next one.
- A refresh or reload button to reload the current page.
- A stop button to cancel loading the page. (In some browsers, the stop button is merged with the reload button.)
- A home button to return to the user's home page.

- An address bar to input the URL of a page and display it.
- A search bar to input terms into a search engine

3.2.2. GPU

A graphics processing unit (GPU) is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPUs are used in embedded systems, mobile phones, personal computers, workstations, and game consoles. Modern GPUs are very efficient at manipulating computer graphics and image processing, and their highly parallel structure makes them more efficient than general-purpose CPUs for algorithms where the processing of large blocks of data is done in parallel. In a personal computer, a GPU can be present on a video card, or it can be embedded on the motherboard or—in certain CPUs—on the CPU die.

3.2.3. GPS Receiver

Global Positioning System **(GPS)** is a navigation system based on satellite. It has created the revolution in navigation and position location. It is mainly used in positioning, navigation, monitoring and surveying applications. GPS receivers have been miniaturized to just a few integrated circuits and so are becoming very economical. And that makes the technology accessible to virtually everyone. There exists only one-way transmission from satellite to users in GPS system. Hence, the individual user does not need the transmitter, but only a **GPS receiver**. It is mainly used to find the accurate location of an object. It performs this task by using the signals received from satellites.

CHAPTER -4

METHODOLOGY

In this chapter, we will be discussing the design procedure of the system which involves the basic flowcharts to describe the various stages of the system. It focuses particularly on vehicle-mounted solutions that perform non-invasive monitoring of the driver's head and face for behavioral signs of potential drowsiness and for speed tracking system it focuses on the vehicle's speed.

4.1. Used Technologies/Softwares

4.1.1. Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. It can be used for developing desktop GUI applications, websites and web applications. Like other modern programming languages, Python also supports several programming paradigms. It supports object oriented and structured programming fully. As an open source programming language, Python helps in curtailing software development cost significantly. When it comes to backend development, Python is among the most popular choices that businesses, as well as developers, make. It has various libraries like- numpy, pandas, scipy, scikit-learn, etc. We will use OpenCV which is used for all sorts of image and video analysis, like facial recognition and detection, license plate reading, photo editing, advanced robotic vision, optical character recognition, and a whole lot more.

4.1.2. Jupyter Notebook

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.

The Jupyter Notebook combines two components:

- The web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.
- Notebook documents: Self-contained documents that contain a representation of all content visible in the notebook web application, including inputs and outputs of the computations, narrative text, equations, images, and rich media representations of objects. Each notebook document has its own kernel.

The Jupyter notebook can have different environments that can be made by user. One such environment that we have used is Django which is used for connecting the python script with HTML and CSS. It is a framework to create python based interactive web applications.

4.1.3. Google Maps API

Google Map's API is a robust tool that can be used to create a custom map, a searchable map, check-in functions, display live data synching with location, plan routes, or create a mashup just to name a few. Google Places API allows web and mobile app developers to do customize their search by:

1. apply a keyword search or filters; and
2. rank by distance and popularity.

Google Map's APIs are very powerful, but what is even more powerful is the mashup of more than one API. Google allows a website to call any Google API for free, thousands of times a day.

4.1.4. HTML & Javascript

HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language. HTML is a must specially if the working is in Web Development domain.

HTML tags create objects and JavaScript lets you manipulate those objects. JavaScript's main purpose is to manipulate HTML and CSS. JavaScript is used to implement the dynamic and interactive actions on elements of the page. It determines what happens when users click, hover or type within certain elements. Without

JavaScript a website will still be functional, but in a limited way. JavaScript is what animates HTML and CSS, and it is what brings a website to life.

4.1.5. Django

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. By using Django, we can build web applications in very less time. Django is designed in such a manner that it handles much of configure things automatically, so we can focus on application development only. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement.

4.2. Used Libraries/Packages

4.2.1. cv2.PyPi

OpenCV is an open source computer vision and machine learning software library. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

4.2.2. imutils.PyPi

It's a series of convenience functions to make basic image processing operations such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and Python. It has many submodules like –

- face_utils
- video
- feature
- io

In this project, we are using videostream and face_utils submodules of imutils for processing of live video frames and detection of facial features from these frames.

4.2.3. pyglet.PyPi

Pyglet is a python module used for visuals and sound. Pyglet can play WAV files, and if FFmpeg is installed, many other audio and video formats. We are using this module to play the alarm sound which is used to warn and wake the driver if he closes his eyes or he over speeds the vehicle.

Playback is handled by the Player class, which reads raw data from Source objects and provides methods for pausing, seeking, adjusting the volume, and so on. The Player class implements the best available audio device. A Source is used to decode arbitrary audio and video files.

4.2.4. dlib.PyPi

dlib is a toolkit for making real world machine learning and data analysis applications in C++. While the library is originally written in C++, it has good, easy to use Python bindings. In this project, dlib is majorly used for face detection and facial landmark detection. The frontal face detector in dlib is simple and just works out of the box. This detector is based on histogram of oriented gradients (HOG) and linear SVM**.**

4.2.5. pyinstaller.PyPi

PyInstaller reads a Python script written by you. It analyzes your code to discover every other module and library your script needs in order to execute. Then it collects copies of all those files (including the active Python interpreter) and puts them with your script in a single folder, or optionally in a single executable file.

Its main advantages over similar tools are that PyInstaller works with any version of Python since 2.3, it builds smaller executables thanks to transparent compression, it is fully multi-platform, and uses the OS support to load the dynamic libraries, thus ensuring full compatibility.

4.2.6. haversine. PyPi

Haversine package is used to calculate the distance (in various units) between two points on the Earth using their latitude and longitude. It can be installed by using pip3 install haversine command.

4.2.7. Geolocation API

The Geolocation is one of the best HTML5 API which is used to identify the user's geographic location for the web application. This new feature of HTML5 allows you to navigate the latitude and longitude coordinates of the current website's visitor. These coordinates can be captured by JavaScript and send to the server which can show your current location on the website. Most of the geolocation services use Network routing addresses such as IP addresses, RFID, WIFI and MAC addresses or internal GPS devices to identify the user's location.

4.2.8. Roads API

The Roads API allows you to map GPS coordinates to the geometry of the road, and to determine the speed limit along those road segments. The API is available via a simple HTTPS interface, and exposes the following services:

- **Snap to roads** This service returns the best-fit road geometry for a given set of GPS coordinates. This service takes up to 100 GPS points collected along a route, and returns a similar set of data with the points snapped to the most likely roads the vehicle was traveling along. Optionally, you can request that the points be interpolated, resulting in a path that smoothly follows the geometry of the road.
- **Nearest roads** This service returns individual road segments for a given set of GPS coordinates. This service takes up to 100 GPS points and returns the closest road segment for each point. The points passed do not need to be part of a continuous path.
- **Speed limits** This service returns the posted speed limit for a road segment. The Speed Limit service is available to all customers with an Asset Tracking license. For Google Maps Platform Premium Plan customers who transitioned to pay-as-you-go pricing, the feature remains active.[4]

4.3. Prototype for drowsiness detection

The prototype Eye Tracking System for Drowsiness Detection (ETSDD) includes a dashboard mounted commodity camera, simple alarm board and processor (laptop) equipped with the developed software.

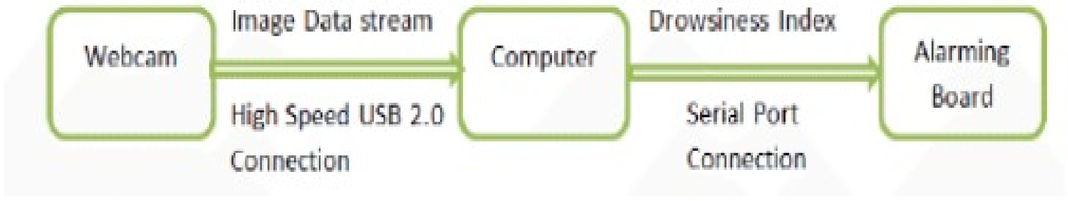


Figure 2: Drowsiness detection system architecture

The system performs a real-time processing of the input image stream so to compute the level of fatigue of the driver. The analysis is based on calculating several frames of the data stream where the driver eyes are closed. The result of the processing is sent to the alarm board, which activates an alarm signal when the drowsiness index exceeds a pre-specified parameter. Because the face and eye tracking depends on light intensity and face illumination, the background should not contain any other high brightness objects or direct light sources. In order to effectively capture the face, the webcam is placed onto the vehicle dashboard and is approximately 20cm away from the driver’s face.

4.4. Steps to design the ETSDD

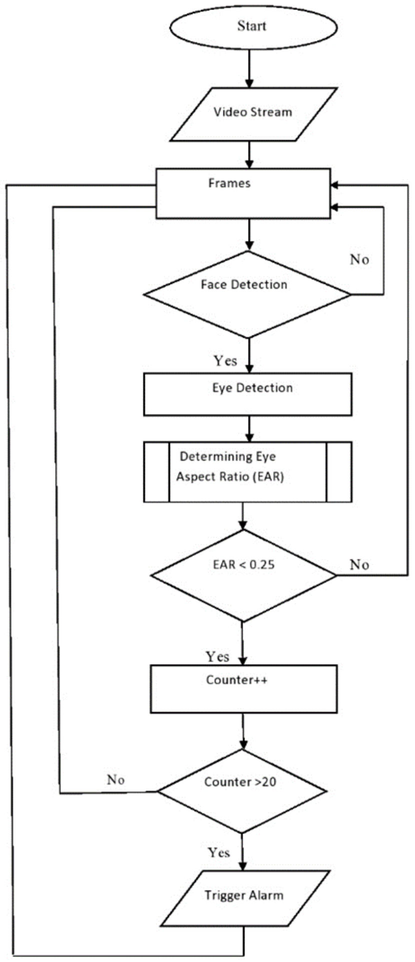


Figure 3: Flowchart for drowsiness detection

In the eye detection stage, the processor receives a real time video, first, then it recognizes the image from the video frames. If the input image does not contain the driver’s face, the program continues to grab new input images from the webcam until the face is detected. From there the eyes region can be extracted. The system employs the Viola-Jones technique and the standard Ada-Boost (Adaptive Boosting) training method to do the fast and effective

eyes detection extraction. In the first stage of the algorithm, a Haar-like features (reminiscent of Haar Basis functions) are applied on a sub image to extract face features.
The Haar cascade algorithm for eye detection contains four stages:

- Haar Feature Selection
- Creating Intergal Images
- Adaboost Training
- Cascading Classifiers [8]

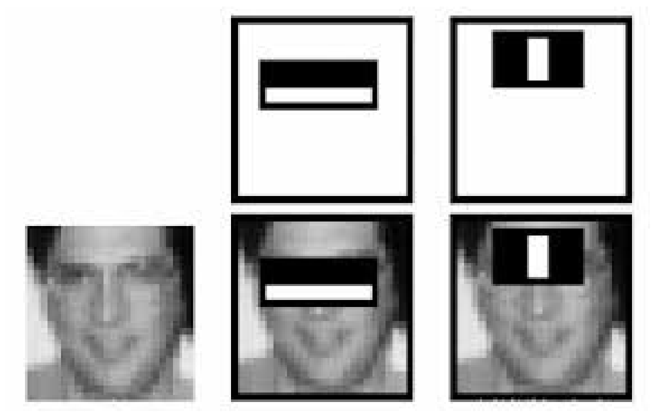
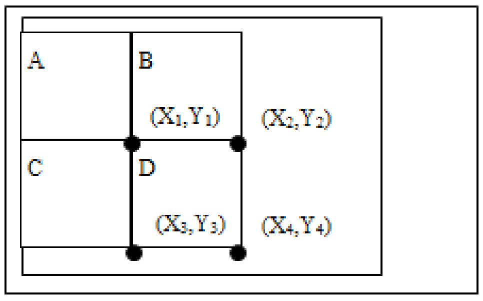
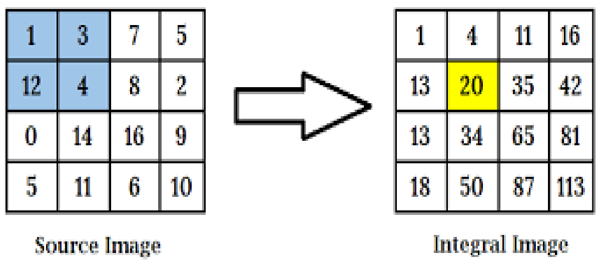


Figure 4: Haar type features

In order to increase efficiency and speed of calculating the sums of pixels inside a rectangle, Viola-Jones algorithm employs a so-called Integral Image technique. The integral image is the matrix where each value of a pixel (X,Y) is a sum of all pixels above and to the left of the coordinate (X,Y) – Fig. 4. This significantly reduces the time and efforts for calculating the sum of pixels in the black and white regions when applying the Haar-like features.





1	3	7	5
12	4	8	2
0	14	16	9
5	11	6	10

1	4	11	16
13	20	35	42
13	34	65	81
18	50	87	113

Figure 5(a) **Figure 5(b)**

Figure 5: Image area integration using Integral Image (II)

The integral image allows integrals for the Haar extractors to be calculated by adding only four numbers. For example, the image integral of area ABCD (Fig.5(a)) is calculated as $II(y_1,x_1) – II(y_2,x_2) – II(y_3,x_3) + II(y_4,x_4)$.

In a sub-window of 24*24 pixel base resolution, up to 160,000 Haar-features may be required to detect elements of interest in a face. However, there are only few sets among them that are actually useful for identifying the target facial areas. Therefore, it is important to choose the best among 160,000+ features to improve the efficiency and reduce the processing time. And this is where the Ada-Boost classifier is applied. Effectively it constructs a linear combination of weak classifiers to create a stronger classifier.

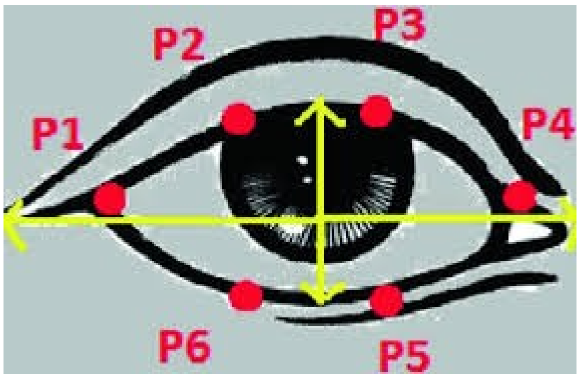


Figure 6: Eye region pixels used for calculating EAR

After detecting the face of the driver, the calculation of drowsiness level of the driver is based on eye blink rate. From the landmarks detected in the image with face, the EAR is used as an estimate of the eye openness state. For every video frame, the eye landmarks are detected between height and width of the eye that had been computed. The eye aspect ratio can be defined by the Equation (1)-

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

Equation (1) shows the eye aspect ratio formula where p1 until p6 are the 2D landmark locations. The p2, p3, p5 and p6 are used to measure the height whereas p1 and p4 are used to measure width of the eyes in meter (m) as shown in Figure 6. The eye aspect ratio is a constant value when the eye is opened, but rapidly falls approximately to 0 when the eye is closed.

We defined two constants, first constant, *EYE_AR_THRESH*, for the eye aspect ratio to indicate eye blink and then a second constant, *EYE_AR_CONSEC_FRAMES,* is defined for the number of consecutive frames which has a certain threshold value. If the eye aspect ratio falls below the first constant value then the counter will be increased to count the number of frames when the eye was closed. If the counter value exceeds the second constant the alarm would be set up and we will start checking the symptoms of drowsiness in the person.

```
EYE_AR_THRESH = 0.25

EYE_AR_CONSEC_FRAMES = 20

COUNTER = 0

ALARM_ON = False

def eye_aspect_ratio(eye):

A = dist.euclidean(eye[1], eye[5])

B = dist.euclidean(eye[2], eye[4])

C = dist.euclidean(eye[0], eye[3])

# compute the eye aspect ratio

ear = (A + B) / (2.0 * C)

return ear
```

We created a separate thread which will be responsible for calling sound alarm to ensure the efficient working of our main program until the drowsiness alert sound finishes playing. If the eyes are open, we reset the counter value and ensure that the alarm is off.

```
if args["alarm"] != "":

t = Thread(target=sound_alarm,

args=(args["alarm"],))

t.deamon = True

t.start()
```

4.5. Steps to design the Speed Alert System

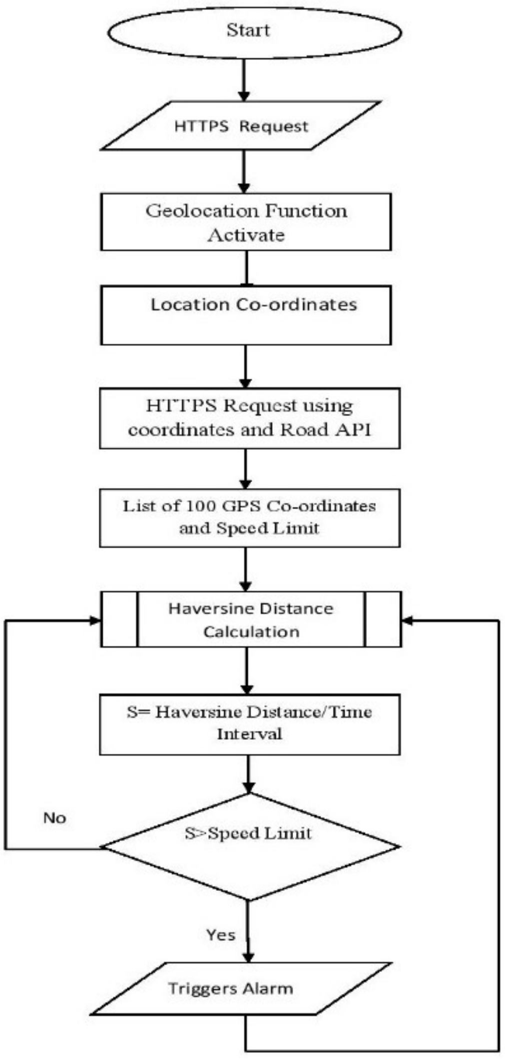

Start
HTTPS Request
Geolocation Function Activate
Location Co-ordinates
HTTPS Request using coordinates and Road API
List of 100 GPS Co-ordinates and Speed Limit
Haversine Distance Calculation
S= Haversine Distance/Time Interval
No
S>Speed Limit
Yes
Triggers Alarm

Figure 7: Flowchart for speed detection and calculation

To tackle the above issues, we propose to build a mobile real time GPS tracker with integrated Google Maps Roads API. In the proposed system, the GPS chip outputs the positioning and speed information which is transferred over a GPRS link to the mobile operator's GGSN (Gateway GPRS Support Node) and then to a remote server over a TCP connection. The TCP server stores the incoming positional data. When a user click on the tracking page, the page serves up an HTML page with an embedded code.[9]

To detect the speed of vehicle Geolocation API of HTML is used. The HTML Geolocation API is used to get the geographical position of a user and the Roads API is used for obtaining a list of about 100 coordinates and speed limits on that road. Using these coordinates, we will be able to calculate the real time current speed of the vehicle in every t seconds of time interval.

For calculating speed we will be using Haversine distance between the two locations on the Earth divided by the fixed time interval of 3 seconds.

The Haversine distance formula is used to calculate the angular distance between two points on the Earth as the Earth has a curved surface area. The first distance of each point is assumed to be the latitude, the second is the longitude, given in radians. The haversine can be expressed in trigonometric function as:

$$haversine(\theta) = sin^2\left(\frac{\theta}{2}\right)$$

The haversine of the central angle (which is d/r) is calculated by the following formula:

$$\left(\frac{d}{r}\right) = haversine(\Phi_2 - \Phi_1) + cos(\Phi_1)cos(\Phi_2)haversine(\lambda_2 - \lambda_1)$$

where r is the radius of Earth (6371 km or 6400 km), d is the distance between two points, Φ_1, Φ_2 are latitude of the two points and λ_1, λ_2 are longitude of the two points respectively.

Solving d by applying the inverse haversine or by using the inverse sine function, we get:

$$d = 2rsin^{-1}\left(\sqrt{sin^2\left(\frac{\Phi_2-\Phi_1}{2}\right) + cos(\Phi_1)cos(\Phi_2)sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\ \right)$$ [10]

Since we are using python for our program we can use the haversine() function of haversine module/package. This package can be installed via pip3 install haversine command. This is the most easy and fastest way to calculate haversine distance in real time. [11]

Geolocation refers to the identification of the geographic location of a user or computing device via a variety of data collection mechanisms. Typically, most geolocation services use network routing addresses or internal GPS devices to determine this location. Geolocation is a device-specific API. This means that browsers or devices must support geolocation in order to use it through web applications. A request via secure contexts such as HTTPS is sent to the geolocation method to get the user's position.

The Geolocation API is work with the navigation.geolocation object. Its read-only property returns a Geolocation object which identifies the location of the user and can generate a customized result based on user location. The getCurrentPosition() method is used then to get the coordinates of location. The main steps included are -

- Check if Geolocation is supported

The geolocation property of navigator.geolcation object helps to determine the browser support for the Geolocation API.

```
<script>
   var x= document.getElementById("location");
   function getlocation() {
       if(navigator.geolocation){
           alert("your browser is supporting Geolocation API")
       }
       else
       {
           alert("Sorry! your browser is not supporting")
       }
      }
      }

</script>
```

- If supported, run the getCurrentPosition() method. If not, display a message to the user

To get the user's current location, getCurrentPosition() method of the navigator.geolocation object is used. This method accepts three parameters:

- **success:** A success callback function to get the location of the user
- **error:** An error callback function which takes "Position Error" object as input.
- **options:** It defines various options for getting the location.

```
<script>
      var x= document.getElementById("location");
         function getlocation() {
             if(navigator.geolocation){
                 navigator.geolocation.getCurrentPosition(showPosition)
             }
             else
             {
                 alert("Sorry! your browser is not supporting")
             }
         }

</script>
```

- If the getCurrentPosition() method is successful, it returns a coordinates object to the function specified in the parameter (showPosition)

- The showPosition() function outputs the Latitude and Longitude coordinates along with speed in meters per second (in this case).

```
<script>

    function showPosition(position){
            var x = "Your current location is (" + "Latitude: " + position.coords.latitude
            + ", " + "Longitude: " +   position.coords.longitude + ")";
            var y = "Your current speed is ("  + position.coords.speed+ ")";

          document.getElementById("location").innerHTML = x;
         document.getElementById("location").innerHTML = y;
    }
</script> [12]
```

A request for speed limits must be sent via HTTPS, and takes the following form:

https://roads.googleapis.com/v1/speedLimits?parameters&key=YOUR_API_KEY?

Required parameters for this request:

- Enter a path parameter.
- Your application's API key.
- Optional parameter can be units of speed.

Responses:

- speedLimit — An array of road metadata. Each element consists of the following fields:
  - placeId
  - speedLimit
  - units
- snappedpoint — an array of snapped points. This array is present only if the request contained a path parameter. Each point consists of the following fields:
  - location
  - originalIndex
  - placeId
- warning_message — A string containing a user-visible warning.

The following elements may be present in a speedLimit response-

```
{
 speedLimits:
 [
  {
   placeId: "ChIJX12duJAwGQ0Ra0d4Oi4jOGE",
   speedLimit: 105,
   units: "KPH"
  }
 ],
 snappedPoints:
 [
  {
   location:
   {
    latitude: 38.75807927603043,
    longitude: -9.037417546438084
   },
   originalIndex: 0,
   placeId: "ChIJX12duJAwGQ0Ra0d4Oi4jOGE"
  },
 ],
 warningMessage: "Input path is too sparse. You should provide a path where consecutive points are closer to each other. Refer to the 'path' parameter in Google Roads API documentation."
}[13]
```

- To generate an alarm sound we have created a sound_alarm() function which accepts a path to an audio file that will be the alarm tone which is stored or available on the disk. This function includes methods of pyglet package for implementation of sound.

```
def sound_alarm(path):
music = pyglet.resource.media(&#39;alarm.wav&#39;)
music.play()
pyglet.app.run()
```

4.6. Quality Requirements

- Reliability:

  The information on the system should be readily available to the users at their disposal.

- Efficiency:

  System will be efficient.

- Security:
  - All passwords will be hashed and then stored in the database.
  - Pages of the web site must be access in the way they were intended to be accessed.
  - User will not be allowed to perform any action on the website without login.
- Maintainability:

  Administrators will have the ability to edit every aspect of the system.

- Easy to use:

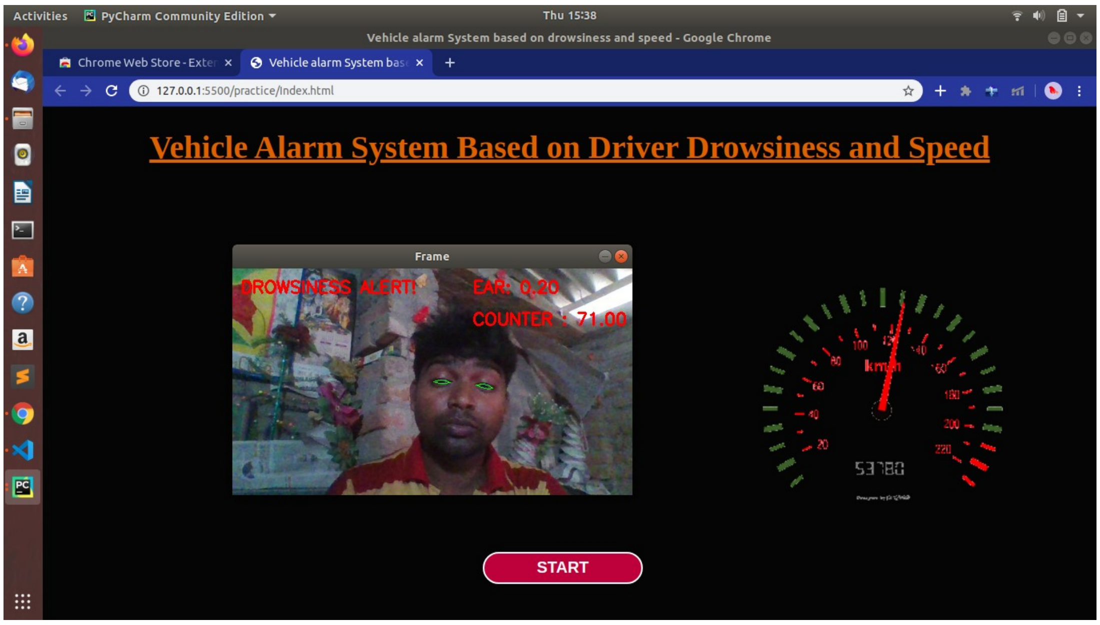
  The system is designed keeping in mind user's requirements and to be user friendly.

CHAPTER -5

RESULTS & PRACTICAL IMPLEMENTATION

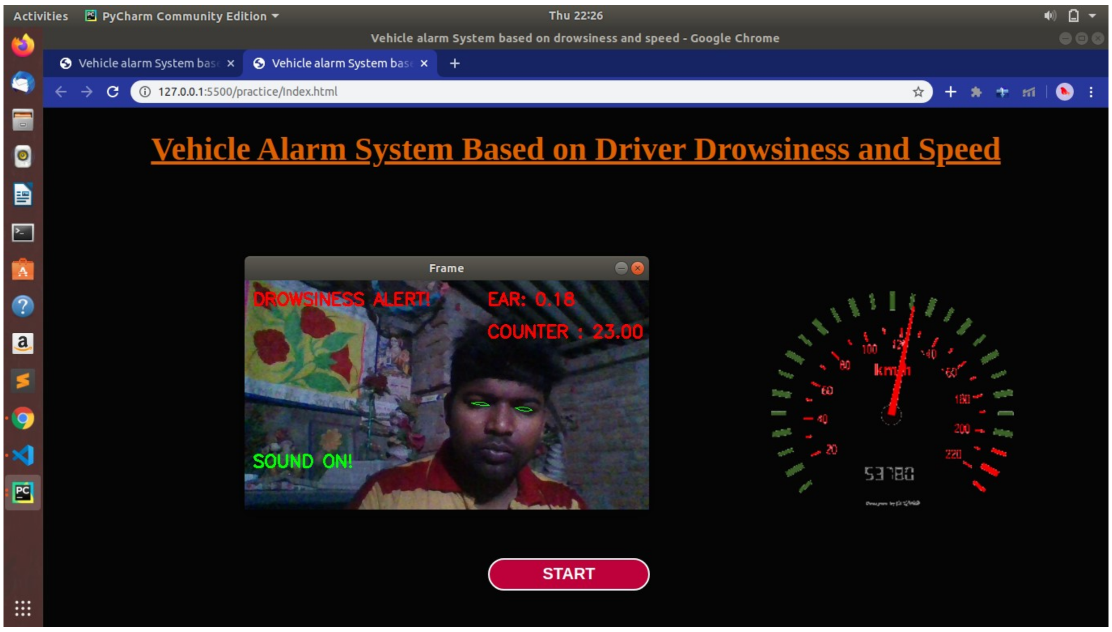
5.1. Snapshots of website

5.1.1. Driver is without spectacles



Activities
PyCharm Community Edition
Thu 15:38
Vehicle alarm System based on drowsiness and speed - Google Chrome
Chrome Web Store - Exte
Vehicle alarm System bas
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
DROWSINESS ALERT!
EAR: 0.20
COUNTER : 71.00
km/h
START

Figure 8: When the driver is awake (eyes open)



Activities
PyCharm Community Edition
Thu 22:26
Vehicle alarm System based on drowsiness and speed - Google Chrome
Vehicle alarm System bas
Vehicle alarm System bas
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
DROWSINESS ALERT!
EAR: 0.18
COUNTER : 23.00
SOUND ON!
km/h
START

Figure 9: When the driver is sleepy (eyes closed)

5.1.2. Driver with spectacles

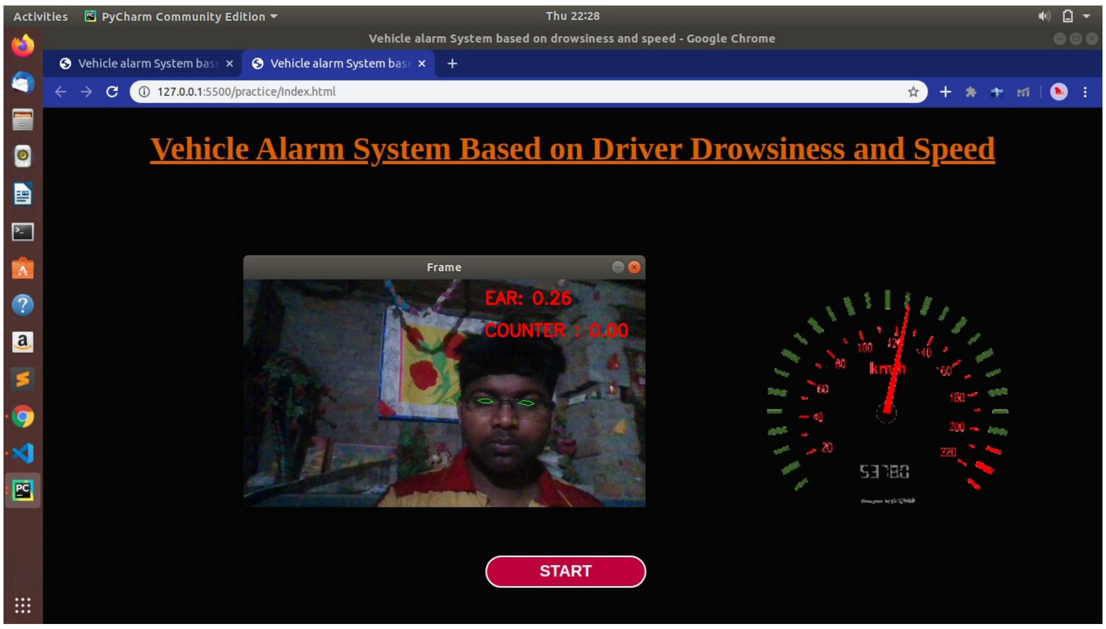

Activities
PyCharm Community Edition
Thu 22:28
Vehicle alarm System based on drowsiness and speed - Google Chrome
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
EAR: 0.26
COUNTER : 0.00
START

Figure 10: When the driver is awake (eyes open)

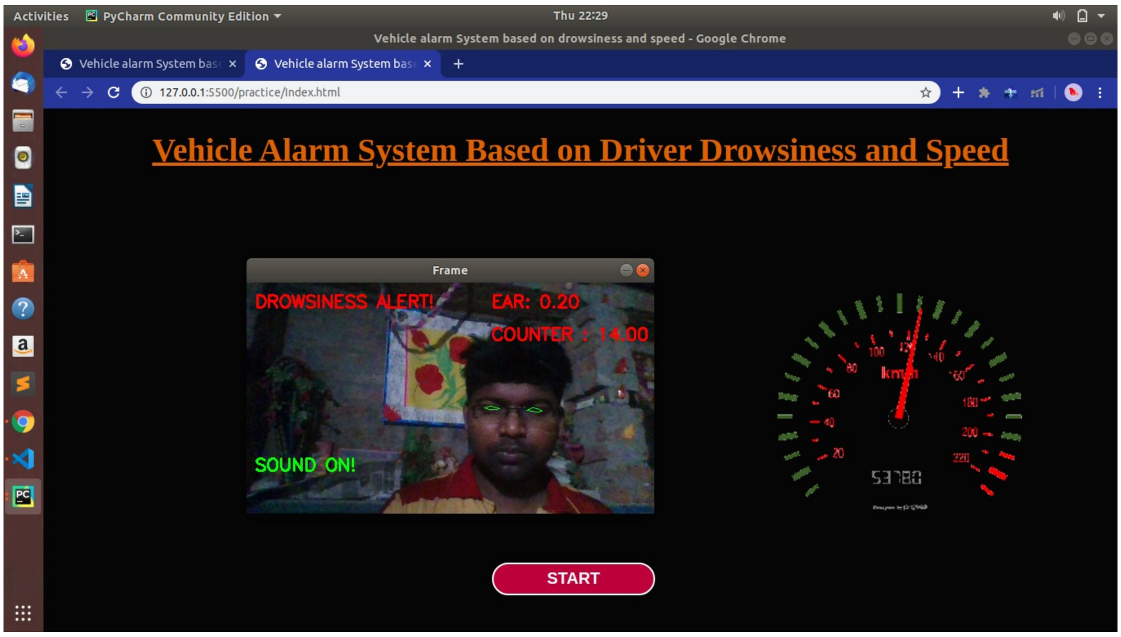

Activities
PyCharm Community Edition
Thu 22:29
Vehicle alarm System based on drowsiness and speed - Google Chrome
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
DROWSINESS ALERT!
EAR: 0.20
COUNTER : 14.00
SOUND ON!
START

Figure 11: When the driver is sleepy (eyes closed)

5.1.3. Driver is impaired with an eye

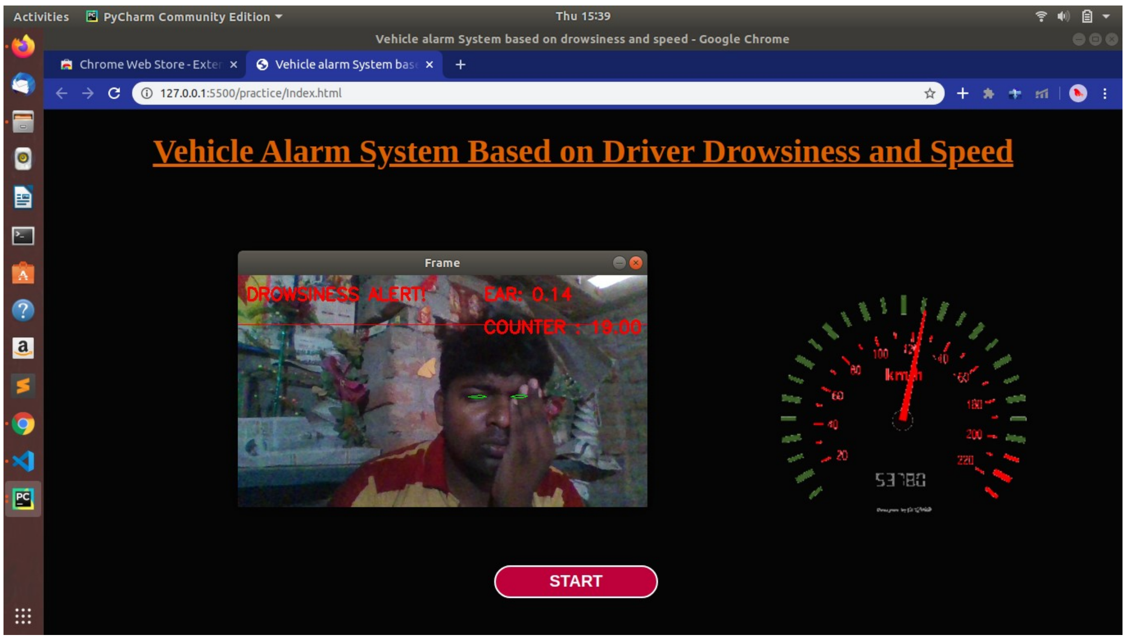

Vehicle alarm System based on drowsiness and speed - Google Chrome
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
DROWSINESS ALERT!
EAR: 0.14
COUNTER : 19.00
km/h
START

Figure 12: When the driver is awake (eyes open)

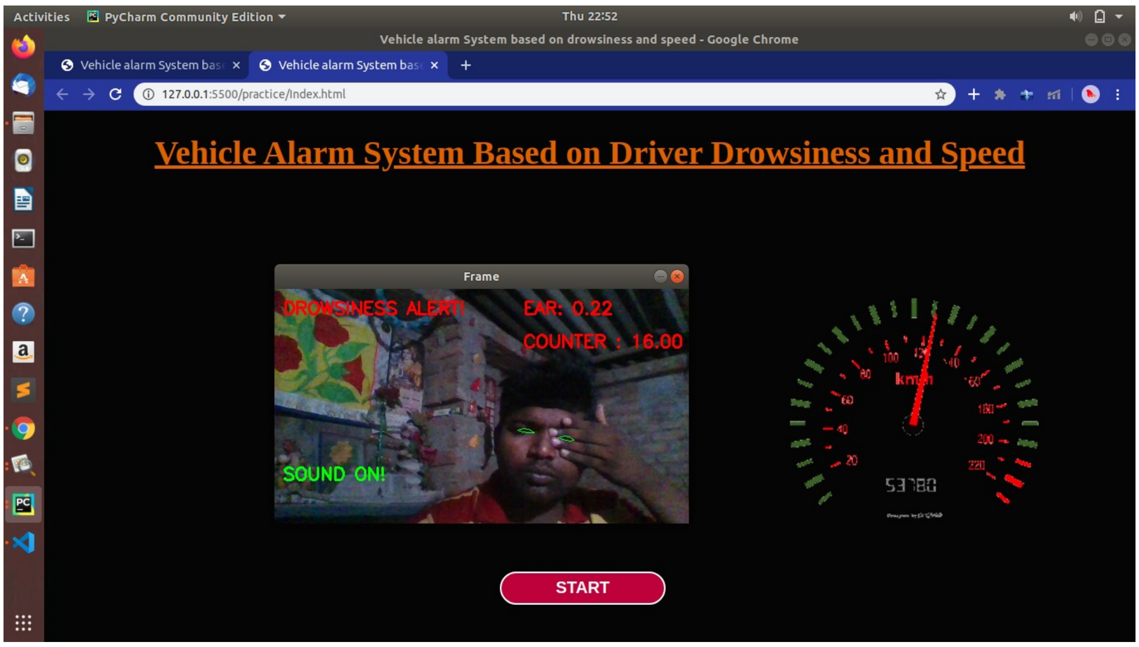

Vehicle alarm System based on drowsiness and speed - Google Chrome
127.0.0.1:5500/practice/index.html
Vehicle Alarm System Based on Driver Drowsiness and Speed
Frame
DROWSINESS ALERT!
EAR: 0.22
COUNTER : 16.00
SOUND ON!
km/h
START

Figure 13: When the driver is sleepy (eyes closed)

5.2. Result

The system works efficiently in all the above cases with an alarming sound. The alarm sounds for both the features is different, that is, the sound for drowsiness alert and the sound for speed system will be different so that the driver can clearly know the difference between both the alerts.

CHAPTER-6

CONCLUSION & FUTURE SCOPE

6.1 Conclusion

The current project work involved the detection of drowsiness of the driver and speed of the vehicle both. This work has been focused on designing of an alarm based system which warns the driver in either of the cases among drowsiness and over speeding. The development has been done by using object detection in image processing technique, Geolocation API and Roads API of Google Maps. We have implemented speed comparison and drowsiness detection (via Object Detection) technique in real time. The alarm generation and the detection of drowsiness in front of the web cam can be done in real time. The alarm is generated by the system using python language library pyglet. From the results obtained in the present experimental work as well as computerised analysis following conclusions can be drawn:

- This system generates a warning alarm either if the driver closes his/her eyes or blinks eyes frequently signifying the drowsiness/fatigue stage.
- There is an additional feature in our system which warns with alarm if the vehicle is over speeding. It compares the current speed of the vehicle with the permitted speed limit on the location in real time.

This concept is highly recommended for its usage in the vehicles and trains. It uses no extra software to download and requires low cost investment.

6.2 Future Scope

The present work on drowsiness and speed based alarm system for vehicles is a promising solution for safety in vehicles and trains. It can prevent further damage to vehicle and its riders in case of an accident. Further work can be done on cost estimation and new versions of this system.

REFERENCES

[1] Javed Ahmed, Jian-Ping Li, Saeed Ahmed Khan, Riaz Ahmed Shaikh "Eye Behavior based Drowsiness Detection System" 12th ICCWAMTIP, Dec 2015.

[2] https://www.iii.org/fact-statistic/facts-statistics-drowsy-driving

[3] https://www.who.int/violence_injury_prevention/publications/road_traffic/world_report/speed_en.pdf

[4] https://www.thehindu.com/news/cities/Hyderabad/overspeeding-a-major-cause-of-road-accidents/article29675683.ece

[5] https://www.livemint.com/news/india/road-accident-data-more-than-97-000-people-killed-due-to-over-speeding-in-2018

[6] Arun Sahayadhas, Kenneth Sundaraj, and Murugappan "Detecting Driver Drowsiness Based on Sensors: A Review" 2012.

[7] Road Accidents in India (2015), Ministry of Road Transport and Highways Transport Research Wing, Government of India.

[8] http://www.willberger.org/cascade-haar-explained

[9] Mihir Garude, Nirmal Haldikar "Real Time Position Tracking System using Google Maps API V3" ISSN, 2014.

[10] https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere

[11] https://pypi.org/project/haversine

[12] https://www.javatpoint.com/html-geolocation

[13] https://developers.google.com/maps/documentation/roads/speed-limits