A PROJECT REPORT
DROWSINESS DISTRACTION DETECTION USING OPENCV

Submitted In Partial Fulfillment of the Requirement for the Award of

Post Graduate Diploma in Artificial Intelligence

(PG-DAI) Under the Guidance of

MS. SARUTI GUPTA

(Project Guide)

Submitted By

KALYANI SHEWALE PRN: 20220320528018

MANISHA KADAM PRN: 20220320528015

ANKUR DIXIT PRN: 20220320528005

CDAC, B-30, Institutional Area, Sector-62

Noida (Uttar Pradesh)-201307

CERTIFICATE

CDAC, NOIDA

This is to certify that Report entitled "Machine Learning & using open CV Based DRIVER DROWSINESS DISTRACTION DETECTION " which is submitted by

Kalyani Shewale, **Manisha Kadam** and **Ankur Dixit** in partial fulfillment of the requirement for the award of Post Graduate Diploma in Artificial Intelligence (PGDAI) to CDAC, Noida is a record of the candidates own work carried out by them under my supervision.

The documentation embodies results of original work, and studies are carried out by the student themselves and the contents of the report do not from the basis for the award of any other degree to the candidate or to anybody else from this or any other

MS. SARUTI GUPTA (Project Guide)

ABSTRACT

It has been illustrated by experts, that accidents caused by drowsiness are significantly underreported in police crash investigations They estimate that about 24-33% of the severe accidents are related to drowsiness. In order to develop warning systems that detect reduced vigilance based on the driving behaviour, a reliable and accurate drowsiness reference is needed. Studies have shown that measures of the driver's eyes are capable to detect drowsiness under simulator or experiment conditions. In this study, the performance of the latest eye tracking based in-vehicle fatigue prediction measures are evaluated. These measures are assessed statistically and by a classification method based on a large dataset of 90 hours of real road drives. The results show that eye-tracking drowsiness detection works well for some drivers as long as the blinks detection works properly. Even with some proposed improvements, however, there are still problems with bad light conditions and for persons wearing glasses. As a summary, the camera based sleepiness measures provide a valuable contribution for a drowsiness reference, but are not reliable enough to be the only reference. Advanced Driver Assistance System (ADAS) for automatic driver drowsiness detection based on visual information and Artificial Intelligence is presented. The aim of this system is to locate, to track and to analyse the face and the eyes to compute a drowsiness index, working under varying light conditions and in real time. The implementation of systems based on computer vision is presented, and different visual perception modules useful for some ADAS such as Line Keeping System, Adaptive Cruise Control, Pedestrian Protector, or Speed Supervisor, are described. To prevent car crashes that occur due to drowsy driver, it is essential to have an assistive system that monitors the vigilance level of driver and alert the driver in case of drowsy detection. This system presents a drowsy detection system based on eye detection of the driver.

So, we have tried to deal this problem which many of among us face while driving (especially covering long distance) in order to avoid catastrophic outcomes. We were inspired by the ones who have already done some work on it and tried to make our project to match to the standards which they have made or even look forward to set up a milestone in this work. This report is a detailed analysis of our work dealing with high standard methodology

ACKNOWLEDGEMENT

We would like to express our best sense of gratitude &endeavor with respect to Ms. Saruti Gupta (Project Guide) CDAC, Noida for suggesting the problems scholarly guidance and expert supervision during the course of this project. Special thanks to Mr. Ravi Payal (Program Coordinator).

We are very thankful to Ms. Saruti Gupta (Project Guide) the project guide for constant simulated discussion, encouraging new ideas about this project.

KALYANI SHEWALE PRN: **220320528018**

MANISHA KADAM PRN: **220320528015**

ANKUR DIXIT PRN**: 220320528005**

Table of Contents

LIST OF FIGURE

LIST OF TABLE

CHAPTER 1

1.INTRODUCTION

Drowsiness is a human characteristic that is not taken seriously by any individual. But this particular human feature can have grave and fatal consequences if not considered and acted upon especially on roads while driving. Drowsiness is defined as the state of feeling sleepy. A human being requires to have a minimum amount of 6-7 hours of sleep per day for proper functioning and for carrying out his/her day to day activities. When this particular factor is ignored and a person is not getting enough sleep due to any reason it directly leads to a state of drowsiness. This can get dangerous while a person is driving any vehicle. Most of the accidents occur because of the drowsiness of the driver and this is something that can be brought under control. Driver drowsiness detection is a technology that ensures car safety that can in turn help prevent mishap such as accidents when the driver is feeling sleepy. There are various other factors that can cause car accidents like the conditions of the road, weather conditions and mechanical fault/error of the car. But 80% of the mishaps occur due to driver‘s error that includes drinking and driving, fatigue and drowsiness. There are factors that affect the driver‘s ability to control the vehicle such as perception, natural reflexes and recognition. The diminishing of these factors can cause accidents. Our paper aims to evaluate specific activities of the driver to determine the drowsiness level. Various studies have suggested that around 20% of all road accidents are related to fatigue. Driver fatigue is a significant factor in large number of vehicle accidents. The recent advancement in technologies for detecting or preventing drowsiness at the wheel is a major challenge in the field of accident avoidance systems. There are various methods through which drowsiness of the person driving the vehicle could be detected. Drowsiness detection can be divided into categories based on the vehicle, the behavior of the driver and somethings the physiological factors . Possible detection technique based on the vehicle could be deviation from the lane or the pressure on the acceleration portal. Also detection through pulse rate, heartbeat etc. fall under the physiological methods. The Drowsiness detection system provided here focuses on the behavioral factors of the person driving the car. The behaviors shown by the person driving the car in the state of drowsiness include eye closure, eye blinking, yawning and also the head pose . Majorly focus on the factors related to the eyes of the driver and detecting the state of drowsiness The use of image processing in the following system is very important and necessary as it provides one of the best solutions to detect the drowsiness at earliest and spares time to work on avoiding the accidents. Image processing is used in this system to process the images that are collected from the vehicle of the person driving the vehicle. We are also including one more feature in our project module that is Intrusion detection. This is an important feature for ensuring safety of the cars from the burglars.

BACKGROUND :

Driver drowsiness detection is a car safety technology which helps prevent accidents caused by the driver getting drowsy. Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads.

- Some of the current systems learn driver patterns and can detect when a driver is becoming drowsy.
- In 2007, Volvo Cars launched the world‘s first Driver Drowsiness Detection system, Driver Alert Control. The system monitors the car‘s movements and assesses whether the vehicle is being driven in a controlled or uncontrolled way. If the system detects a high risk of the driver being drowsy, the driver is alerted via an audible signal.
- Smart Bluetooth headset that detects signs of drowsiness through the eyes and head motion, and uses a combination of light, sound, and vibration to alert the user

Nowadays, more and more professions require long-term concentration. Drivers must keep a close eye on the road, so they can react to sudden events immediately. Driver fatigue often becomes a direct cause of many traffic accidents. Therefore, there is a need to develop the systems that will detect and notify a driver of her/him bad psychophysical condition, which could significantly reduce the number of fatigue-related car accidents. However, the development of such systems encounters many difficulties related to fast and proper recognition of a driver‘s fatigue symptoms. One of the technical possibilities to implement driver drowsiness detection systems is to use the vision-based approach. This article presents the currently used driver drowsiness detection systems. Here we are detecting the driver drowsiness by estimating vision system of him .

A new approach towards automobile safety and security with autonomous region primarily based automatic automotive system is projected during this conception. We have a tendency to propose 3 distinct however closely connected ideas viz. a Drowsy Driver Detection system and a traffic detection system with external vehicle intrusion dodging primarily based conception. In recent time‘s automobile fatigue connected crashes have very enlarged. So as to attenuate these problems, we‘ve incorporated driver alert system by watching each the driver‘s eyes still as sensing still because the driver state of affairs based primarily based native setting recognition based AI system is projected

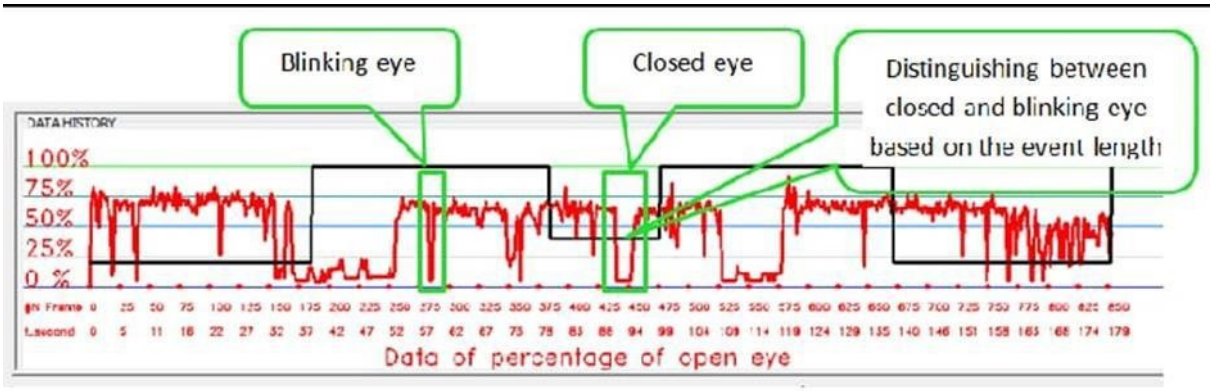


FIG 1.1 Background Process

OBJECTIVE :

The main aim of this is to develop a drowsiness detection system by monitoring the eyes; it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident. Detection of fatigue involves the observation of eye movements and blink patterns Nowadays, more and more professions require long-term concentration. Drivers must keep a close eye on the road, so they can react to sudden events immediately. Driver fatigue often becomes a direct cause of many traffic accidents. Therefore, there is a need to develop the systems that will detect and notify a driver of her/him bad psychophysical condition, which could significantly reduce the number of fatigue-related car accidents. However, the development of such systems encounters many difficulties related to fast and proper recognition of a driver‘s fatigue symptoms. One of the technical possibilities to implement driver drowsiness detection systems is to use the vision-based approach. This article presents the currently used drowsiness detection systems

The facts and statistics stated make it obvious that drowsy driving is a significant problem for human beings. The objective of this thesis is to recognize drivers‘ state with high performance. An effective and user-friendly drowsiness detection system will save people‘s lives and make our world a better place to live.

MOTIVATION :

Traffic accidents is a threatening phenomenon for human beings. According to National Road Accident Survey , about 1 lakh. traffic accidents occurred throughout 2019 . Many of these accidents result from drivers‘ being drowsy or asleep. In Table 1.3, total accidents number of persons killed and injured are listed year by year between 2010 and 2019.

Table 1.3: Total accidents, number of persons killed and injured between 2010 and 2019 in India

Years	Total Accidents	Number of persons killed	Number of persons injured
2010	55 637	3946	18 214
2011	57 352	4427	16 437
2012	62 789	4505	15 086
2013	72 755	4633	19 080
2014	85 561	5007	18 057
2015	90 120	4236	18 468
2016	1 03 346	4323	20 380
2017	1 10 201	4045	21 496
2018	1 22 928	3835	23 074
2019	1 29 634	3750	26 079

According to National Road Accident Survey, the number of traffic accidents increases gradually every year. This is not just a problem of our country, it is rather a worldwide problem. According to Association for Safe International Road Travel, about 1 lakh persons die in road crashes each year, which means 287 deaths a day. One of the main reasons of traffic accidents is drowsy driving. According to the estimation made by Indian National Highway Traffic Safety Administration, each year approximately 1000 traffic accidents arise from driver drowsiness and fatigue. According to statistics gathered by federal government, in the U.S. at least 1500 persons die and 40000 persons are injured in drowsy driver crashes each year. These numbers are most likely an underestimation. Unless someone witnesses or survives the car accident and can testify to the driver‘s condition, it is difficult to detect the driver being drowsy . 37% of surveyed American adults said they have dozed off while driving at least once and 27% said that they dozed off while driving in the past year. In the USA, a series of studies by the National Transportation Safety Board (NTSB) have pointed out the significance of sleepiness as being the reason of accidents involving heavy vehicles. According to NTSB, 52% of 107 single vehicle accidents involving heavy trucks were drowsiness-related. Fatigue affects a driver‘s ability to drive. With increasing fatigue, the driver‘s reaction time increases and driver‘s ability to avoid accidents decreases.

PURPOSE AND SCOPE

PURPOSE—

Driver drowsiness detection Driver drowsiness detection is a car safety technology which helps prevent accidents caused by the driver getting drowsy. Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads.

SCOPE—

A non-intrusive system to localize the eyes and monitor fatigue was developed. Information about the head and eyes position are obtained through various self-developed image processing algorithms. During the monitoring, the system is able to decide whether the eyes are opened or closed. When the eyes have been closed for two seconds, a warning signal is issued. In addition during monitoring, the system is able to automatically detect any eye localizing error that might have occurred. In case of this type of error, the system is able to recover and properly localize the eyes.

The proposed system was tested on the real driver images. The video image [480 x 640 pixels] of 75 different test persons has been recorded during several day, night and complex background at different places. The proposed system has two key phases such as preprocessing and detecting eye from video images. In preprocessing, new enhanced technique is used to enhance the contrast of dark regions and tested with existing algorithm. As per the results obtained all the noises in the video image are removed successfully. In second phase new techniques are used to extract eye from the preprocessed image.

CHAPTER 2

2. SURVEY OF TECHNOLOGY

Nowadays we see many of the accident taking place on road and on machines. There are many reason behind these accidents, but major types of accident occurring during night is because of drowsy condition of the driver driving a car and even with worker working on a machine. Drowsiness of the driver and workers may be because of night shift, driving for long distance, medical issue, drunk or it can be any other. To avoid this accident many drowsy detection systems have been developed till date. Drowsy Detection, visual based detection, physiology based detection, vehicle based detection and eye gazing .

There are millions of vehicles travelling across the world and thousands of accidents take place every month. According to the survey on road accident, many accidents take place because of the drowsy condition of the driver. Around 53% of road accident is due to drowsy condition of the driver in India in the year 2014.Many techniques are developed to overcome these problems, but still we see same scenario. Alarm method, alert system, detection system, etc are developed for detecting drowsiness. Drowsy detection system has now become very challenging for the researcher to avoid accident and try to solve these problems. In this survey, we have gone through many technologies to detect drowsy condition and tried to find the best among all. In this survey, we have gone through many technologies and had tried to present it in this paper.

MACHINE LEARNING:

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task.Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that which makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect. Arthur Samuel, a pioneer in the field of artificial intelligence and computer gaming, coined the term ―Machine Learning. He defined machine learning as – Field of study that gives computers the capability to learn without being explicitly

programmed‖. In a very layman manner, Machine Learning(ML) can be explained as automating and improving the learning process of computers based on their experiences without being actually programmed i.e. without any human assistance. The process starts with feeding good quality data and then training our machines(computers) by building machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data do we have and what kind of task we are trying to automate.

MACHINE LEARNING LIBRARIES USED:

SCIPY:

Scipy, a scientific library for Python is an open source, BSD-licensed library for mathematics, science and engineering. The SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The main reason for building the SciPy library is that, it should work with NumPy arrays. SciPy, a scientific library for Python is an open source, BSD-licensed library for mathematics, science and engineering. The SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The main reason for building the SciPy library is that, it should work with NumPy arrays. It provides many user-friendly and efficient numerical practices such as routines for numerical integration and optimization. This is an introductory tutorial, which covers the fundamentals of SciPy and describes how to deal with its various modules.

Together, they run on all popular operating systems, are quick to install and are free of charge. NumPy and SciPy are easy to use, but powerful enough to depend on by some of the world's leading scientists and engineers.The basic data structure used by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for Linear Algebra, Fourier Transforms and Random Number Generation, but not with the generality of the equivalent functions in SciPy.

NUMPY:

IT is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a python library used for working with arrays.It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

DLIB:

Dlib is a general purpose cross-platform software library written in the programming language C++. Its design is heavily influenced by ideas from design by contact and component – based software enginering. Thus it is, first and foremost, a set of independent software components. It is open-source software released under a Boost Software License.Since development began in 2002, Dlib has grown to include a wide variety of tools. As of 2016, it contains software components for dealing with networking, threads, graphical user interfaces, data structure, linear algebra, machine learning, image processing, data mining, XML and text parsing, numerical optimization, Bayesian Network, and many other tasks. In recent years, much of the development has been focused on creating a broad set of statistical machine learning tools and in 2009 Dlib was published in the Journal of Machine Learning Research. Since then it has been used in a wide range of domains.

Dlib contains a wide range of machine learning algorithms. All designed to be highly modular, quick to execute, and simple to use via a clean and modern C++ API. It is used in a wide range of applications including robotics, embedded devices, mobile phones, and large high performance computing environments. DLib includes extensive unit testing coverage and examples using the library. Every class and function in the library is documented. This documentation can be found on the library's home page. DLib provides a good framework for developing machine learning applications in C++.DLib is much like DMTL in that it provides a generic high-performance machine learning toolkit with many different algorithms, but DLib is more recently updated and has more examples. DLib also contains much more supporting functionality.

PYGAME:

Pygame is a cross-platform set of Python modules designed for writing video games . It includes computer graphics and sound libraries designed to be used with the Python . Pygame was originally written by Pete Shinners to replace PySDL after its development stalled. It has been a community project since 2000.and is released under the open source free software GNU Lesser General Public License. Game programming is very rewarding nowadays and it can also be used in advertising and as a teaching tool too. Game development includes mathematics, logic, physics, AI and much more and it can be amazingly fun. In python, game programming is done in pygame and it is one of the best modules for doing so.The pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications. Built on top of the highly portable SDL (Simple DirectMedia Layer) development library, pygame can run across many platforms and operating systems.By using the pygame module, you can control the logic and graphics of your games without worrying about the backend complexities required for working with video and audio.

OPENCV:

OpenCV was started at Intel in 1999 by Gary Bradsky, and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day.OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python languageIt is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel. The library is cross-platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe. Officially launched in 1999 the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.

Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable. Advance vision-based commercial applications by making portable, performance-optimized code available for free – with a license that did not require code to be open or free itself.

The first alpha version of OpenCV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008.

Limitation:

1. In Open CV, memory allocation is a big issue and it does not have its own IDE for execution.

2. Only one eye is considered for monitoring.

THREADING:

Python threading allows you to have different parts of your program run concurrently and can simplify your designA thread is a separate flow of execution. This means that your program will have two things happening at once. But for most Python 3 implementations the different threads do not actually execute at the same time: they merely appear to.It‘s tempting to think of threading as having two (or more) different processors running on your program, each one doing an independent task at the same time. That‘s almost right. The threads may be running on different processors, but they will only be running one at a time.

Getting multiple tasks running simultaneously requires a non-standard implementation of Python, writing some of your code in a different language, or using multiprocessing which comes with some extra overhead.Because of the way CPython implementation of Python works, threading may not speed up all tasks. This is due to interactions with the GIL that essentially limit one Python thread to run at a time.Tasks that spend much of their time waiting for external events are generally good candidates for threading. Problems that require heavy CPU computation and spend little time waiting for external events might not run faster at all.

Architecting your program to use threading can also provide gains in design clarity. Most of the examples you‘ll learn about in this tutorial are not necessarily going to run faster because they use threads. Using threading in them helps to make the design cleaner and easier to reason about.

TIME:

Time module in Python provides various time-related functions. This module comes under Python‘s standard utility modules. time() method of Time module is used to get the time in seconds since epoch. The handling of leap seconds is platform dependent

- Python has defined module, ‖time‖ which allows us to handle various operations regarding time, its conversions and representations, which find its use in various applications in life. The beginning of time is started measuring from 1 January, 12:00 am, 1970 and this very time is termed as ‖epoch‖ in Python. Python has a module named time to handle time-related tasks. The epoch is the point where the time starts. On January 1st of that year, at 0 hours, the ‖time since the epoch‖ is zero. For Unix, the epoch is 1970. To find out what the epoch is, look at gmtime(0).
- The functions in this module do not handle dates and times before the epoch or far in the future. The cut-off point in the future is determined by the C library; for Unix, it is typically in 2038.

There is a popular time module available in Python which provides functions for working with times, and for converting between representations. The function time. time() returns the current system time in ticks since 12:00am, January 1, 1970(epoch).

PLAYSOUND:

Playsound on Python is easy. There are several modules that can play a sound file (.wav).These solutions are cross platform (Windows, Mac, Linux).The main difference is in the ease of use and supported file formats. All of them should work with Python 3. The audio file should be in the same directory as your python program, unless you specify a path. The documentation of playsound states that it has been tested on WAV and MP3 files, but it may work for other file formats as well.

This library was last updated in June 2017. It seems to work well at the time of writing this article, but it‘s not clear whether it will still support newer Python releases. Playsound is the most straightforward package to use if you simply want to play a WAV or MP3 file. It offers no functionality other than simple playback.

If you want to use Python to play or record sound, then you‘ve come to the right place! In this tutorial, you‘ll learn how to play and record sound in Python using some of the most popular audio libraries. You will learn about the most straight-forward methods for playing and recording sound first, and then you‘ll learn about some libraries that offer some more functionality in exchange for a few extra lines of code.

ARGPARSE:

The argparse module includes tools for building command line argument and option processors. It was added to Python 2.7 as a replacement for optparse. The implementation of argparse supports features that would not have been easy to add to optparse, and that would have required backwards-incompatible API changes, so a new module was brought into the library instead. optparse is now deprecated.argparse is a complete argument processing library. Arguments can trigger different actions, specified by the action argument to add_argument(). Supported actions include storing the argument (singly, or as part of a list), storing a constant value when the argument is encountered (including special handling for true/false values for Boolean switches), counting the number of times an argument is seen, and calling a callback to use custom processing instructions.

The default action is to store the argument value. If a type is provided, the value is converted to that type before it is stored. If the dest argument is provided, the value is saved using that name when the command-line arguments are parse The argparse module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and argparse will figure out how to parse those out of sys.argv. The argparse module also automatically generates help and usage messages and issues errors when users give the program invalid arguments

IMUTILS:

A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much more easier with OpenCV and both Python 2.7 and Python 3. This package includes a series of OpenCV + convenience functions that performs basics tasks such as translation, rotation, resizing, and skeletonization.

FACE DETECTION:

Here we deal with a real time situation where in our video gets recorded and needs processing to be done. But the processing can be done only on the image. Hence the captured image has to be divided into frames for further analyzation. In this stage of the process, we deal with identifying the face of the driver. By identifying the face of the driver we mean that detecting facial features or characters through the use of computer. The frame maybe random. Only facial related structures are identified other types of objects are ignored.

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images.Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class. Examples include upper torsos, pedestrians, and cars.

FIG.2.2. Face Detection Image

FACE RECOGNITION:

A facial recognition system is a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a facial database. It is typically used in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Among the different biometric techniques, facial recognition may not be the most reliable and efficient. However, one key advantage is that it does not require aid (or consent) from the test subject. Properly designed systems installed in airports, multiplexes, and other public places can identify individuals among the crowd.

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analyzing patterns based on the person's facial textures and shape While initially a form of computer application, it has seen wider uses in recent times on mobile platforms and in other forms of technology, such as robotics. It is typically used as access control in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems

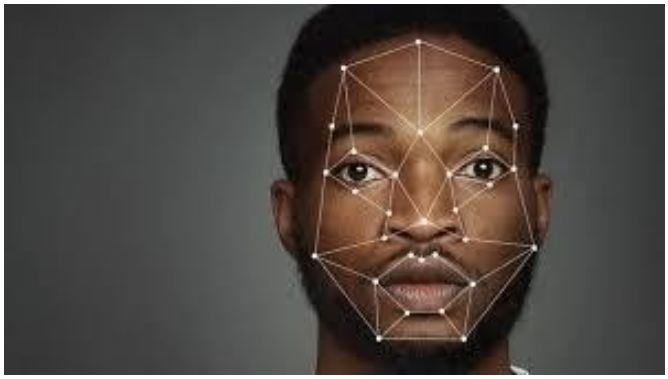


Fig 2.3.Face Recognition Image

ALGORITHM USED IN FACE DETECTION AND FACE RECOGNITION:

SVM (SUPPORT VECTOR MACHINE):

–Support Vector Machine‖ (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

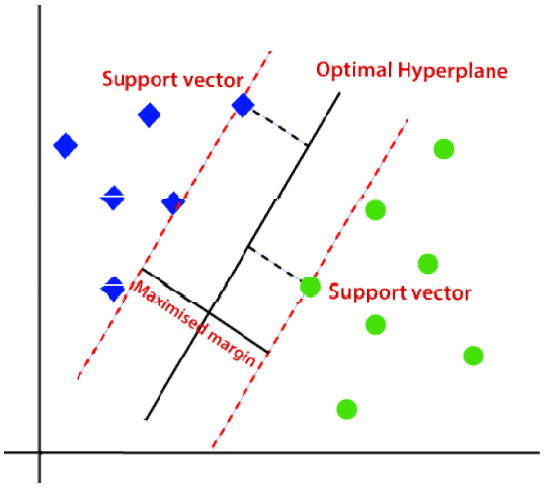


Fig .2.4.1 Figure for SVM

ALGORITHM FOR SVM:

Import the necessary pakages:

```
import pandas as pd
import numpy as np
from sklearn import svm, datasets
import matplotlib.pyplot as plt
```

Load the input data:

```
iris = datasets.load_iris()
```

From input data take two features:

```
X = iris.data[:, :2]
y = iris.target
```

Now plot the SVM boundries with the data:

```
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
h = (x_max / x_min)/100
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
y_max, h))
X_plot = np.c_[xx.ravel(), yy.ravel()]
```

Now we need to provide the value of regularization:

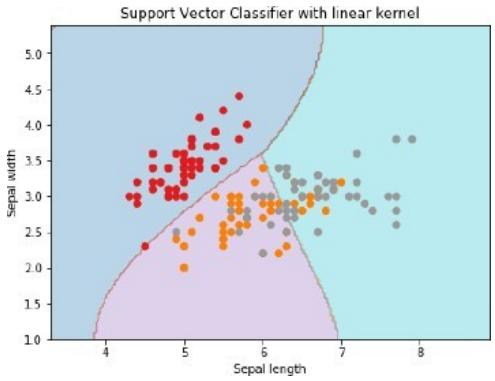
```
C = 1.0
```

Next, SVM classifier object can be created as follows −

Svc_classifier = svm.SVC(kernel='linear', C=C).fit(X, y)

```
Z = svc_classifier.predict(X_plot)
Z = Z.reshape(xx.shape)
plt.figure(figsize=(15, 5))
plt.subplot(121)
plt.contourf(xx, yy, Z, cmap=plt.cm.tab10, alpha=0.3)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(xx.min(), xx.max())
plt.title('Support Vector Classifier with linear kernel')
```

OUTPUT:



HOG (HISTOGRAM ORIENTED GRADIENTS):

The HOG descriptor focuses on the structure or the shape of an object. Now you might ask, how is this different from the edge features we extract for images? In the case of edge features, we only identify if the pixel is an edge or not. HOG is able to provide the edge direction as well. This is done by extracting the gradient and orientation (or you can say magnitude and direction) of the edgesAdditionally, these orientations are calculated in ‗localized' portions. This means that the complete image is broken down into smaller regions and for each region, the gradients and orientation are calculated. We will discuss this in much more detail in the upcoming sections

Finally the HOG would generate a Histogram for each of these regions separately. The histograms are created using the gradients and orientations of the pixel values, hence the name ‗Histogram of Oriented Gradients'

Algorithm for HOG:

Step 1: Preprocessing

Step 2: Calculate the Gradient Images

Visualize the gradient components fxfx and fyfy and the gradient magnitude. function *mag, angle = cart2polar(fx, fy)*.

Step 3: Calculate HOG in 8x8 Cells

A function *HOG8x8(gx, gy)* that takes two a scalar 8x8 images as input where gx is the derivative in x direction and gy the derivative in y direction.

Step 4: Block Normalization

Step 5: Calculate the HOG feature vector

A function *HOGblock(h1, h2, h3, h4)* that takes 4 cell histograms from step 3, concatenates them in one large (36,) vector and normalizes the vector.

Step 6: Visualizing the HOG

In the middle of each 8x8 block we draw a ―rose of directions‖: every bin in the histogram for that block shows the strength of the gradient in that particular orientation. We draw a line in the orientation (center of the line in the middle of the block) with a length that is proportional to the bin count in the histogram.

where,

'$||x_i - v_j||$' is the Euclidean distance between x_i and v_j.

'c_i' is the number of data points in i^{th} cluster.

'c' is the number of cluster centers.

Let X = $\{x_1,x_2,x_3,……..,x_n\}$ be the set of data points and V = $\{v_1,v_2,…….,v_c\}$ be the set of centers.

1) Randomly select 'c' cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..

4) Recalculate the new cluster center using:

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i$$

where, 'c_i' represents the number of data points in i^{th} cluster.

5) Recalculate the distance between each data point and new obtained cluster centers.

6) If no data point was reassigned then stop, otherwise repeat from step 3).

CHAPTER 3

3. FEASIBILITY STUDY:

As the name implies, a feasibility analysis is used to determine the viability of an idea, such as ensuring a project is legally and technically feasible as well as economically justifiable. It tells us whether a project is worth the investment— in some cases, a project may not be doable. There can be many reasons for this, including requiring too many resources, which not only prevents those resources from performing other tasks but also may cost more than an organization would earn back by taking on a project that isn‘t profitable. A well-designed study should offer a historical background of the business or project, such as a description of the product or service, accounting statements, details of operations and management, marketing research and policies, financial data, legal requirements, and tax obligations. Generally, such studies precede technical development and project implementation.

ECONOMIC FEASIBILITY:

For any system if the expected benefits equal or exceed the expected costs, the system can be judged to be economically feasible. In economic feasibility, cost benefit analysis is done in which expected costs and benefits are evaluated. Economic analysis is used for evaluating the effectiveness of the proposed system.
In economic feasibility, the most important is cost-benefit analysis. As the name suggests, it is an analysis of the costs to be incurred in the system and benefits derivable out of the system. Click on the link below which will get you to the page that explains what cost benefit analysis is and how you can perform a cost benefit analysis. It consists of market analysis, economic analysis, technical and strategic analysis. Economic analysis is a method of studying economic processes, which consists in considering the relationships between the various elements of these processes. It can be used both to study economic phenomena and processes occurring on a scale of the whole economy (macroeconomic analysis), as well as phenomena and processes occurring within particular economic units and institutions (microeconomic analysis). Economic analysis makes it possible to make diagnoses, facilitates decision making, as well as facilitates rationalization of economic processes, both on a macro- and microeconomic scale. In economic analysis, mathematical methods are widely applied (e.g. marginal calculus and linear programming

This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility— helping decision-makers determine the positive economic benefits to the organization that the proposed project will provide.

TECHNICAL FEASIBILITY:

In technical feasibility the following issues are taken into consideration.

- Whether the required technology is available or not
- Whether the required resources are available - - Manpower- programmers, testers & debuggers - Software and hardware
- Once the technical feasibility is established, it is important to consider the monetary factors also. Since it might happen that developing a particular system may be technically possible but it may require huge investments and benefits may be less. For evaluating this, economic feasibility of the proposed system is carried out.

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also requirements of the proposed system. As an exaggerated example, an organization wouldn‘t want to try to put Star Trek‘s transporters in their building—currently, this project is not technically feasible.

BEHAVIORAL FEASIBILITY:

Behavioral feasibility.Behavioral feasibility considers human issues. All system development projects introduce change, and people generally resist change. Overt resistance from employees may take the form of sabotaging the new system (e.g., entering data incorrectly) or deriding the new system to anyone who will listen. Covert resistance typically occurs when employees simply do their jobs using their old methods.Behavioral feasibility is concerned with assessing the skills and the training needed touse the new IS. In some organizations, a proposed system may require mathematical or linguistic skills beyond what the workforce currently possesses. In others, a workforce may simply need to improve their skills

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made of how strong a reaction the user staff is likely to have toward the development of a computerized system. [t is common knowledge that computer installations have something to do with turnover, transfers, retraining, and changes in employee job status. Therefore, it is understandable that the introduction of a candidate system requires special effort to educate, sell, and train the staff on new ways of conducting business

CHAPTER 4

4. REQUIREMENT & ANALYSIS:

PROBLEM DEFINITION:

Nowadays, more and more professions require long-term concentration. Drivers must keep a close eye on the road, so they can react to sudden events immediately. Driver fatigue often becomes a direct cause of many traffic accidents. Therefore, there is a need to develop the systems that will detect and notify a driver of her/him bad psychophysical condition, which could significantly reduce the number of fatigue-related car accidents. However, the development of such systems encounters many difficulties related to fast and proper recognition of a driver‘s fatigue symptoms. One of the technical possibilities to implement driver drowsiness detection systems is to use the vision-based approach. This article presents the currently used driver drowsiness detection systems. Here we are detecting the driver drowsiness by estimating vision system of him.

PLANNING AND SCHEDULING (PERT AND GHANTT CHART):

To make this successful we need to make face recognition model with the help of the data of the user and we need to train the model for the same. Face recognition is one of the many wonders that AI research has brought forward to the world. It is a subject of curiosity for many techies — who would like to have a basic understanding of how things work. Let us take a dip into the subject, to see how things work. Training The Model - To train our neural network, we need a huge amount of data. Now, what kind of data do we use when we train a face recognition model? The model we generate should be able to measure similarity between two faces. It should be able to map an image of a face to a vector space such that images of the same person are closer to each other — irrespective of the other parameters. So our data set has to include several images of each person. Face Recognition - When working on problems like this, it is best not to reinvent the wheel — we can‘t! It is best to follow up with models that researchers have provided us. A lot of it is available in the open source as well. One such Python library is face recognition. It works in a few steps:

- Identify a face in a given image Identify specific features in the face
- Generate a face encoding vector of 128 values Based on this encoding, we can measure thesimilarity between two face images — that can tell us if they belong to the same person.

PERT CHART:

The program (or project) evaluation and review technique (PERT) is a statistical tool used in project management, which was designed to analyze and represent the tasks involved in completing a given project.
First developed by the United States Navy in 1958, it is commonly used in conjunction with the critical path method (CPM) that was introduced in 1957.
PERT is a method of analyzing the tasks involved in completing a given project, especially the time needed to complete each task, and to identify the minimum time needed to complete the total project. It incorporates uncertainty by making it possible to schedule a project while not knowing precisely the details and durations of all the activities. It is more of an event-oriented technique rather than start- and completion-oriented, and is used more in those projects where time is the major factor rather than cost. It is applied on very large-scale, one-time, complex, non-routine infrastructure and on Research and Development projects.

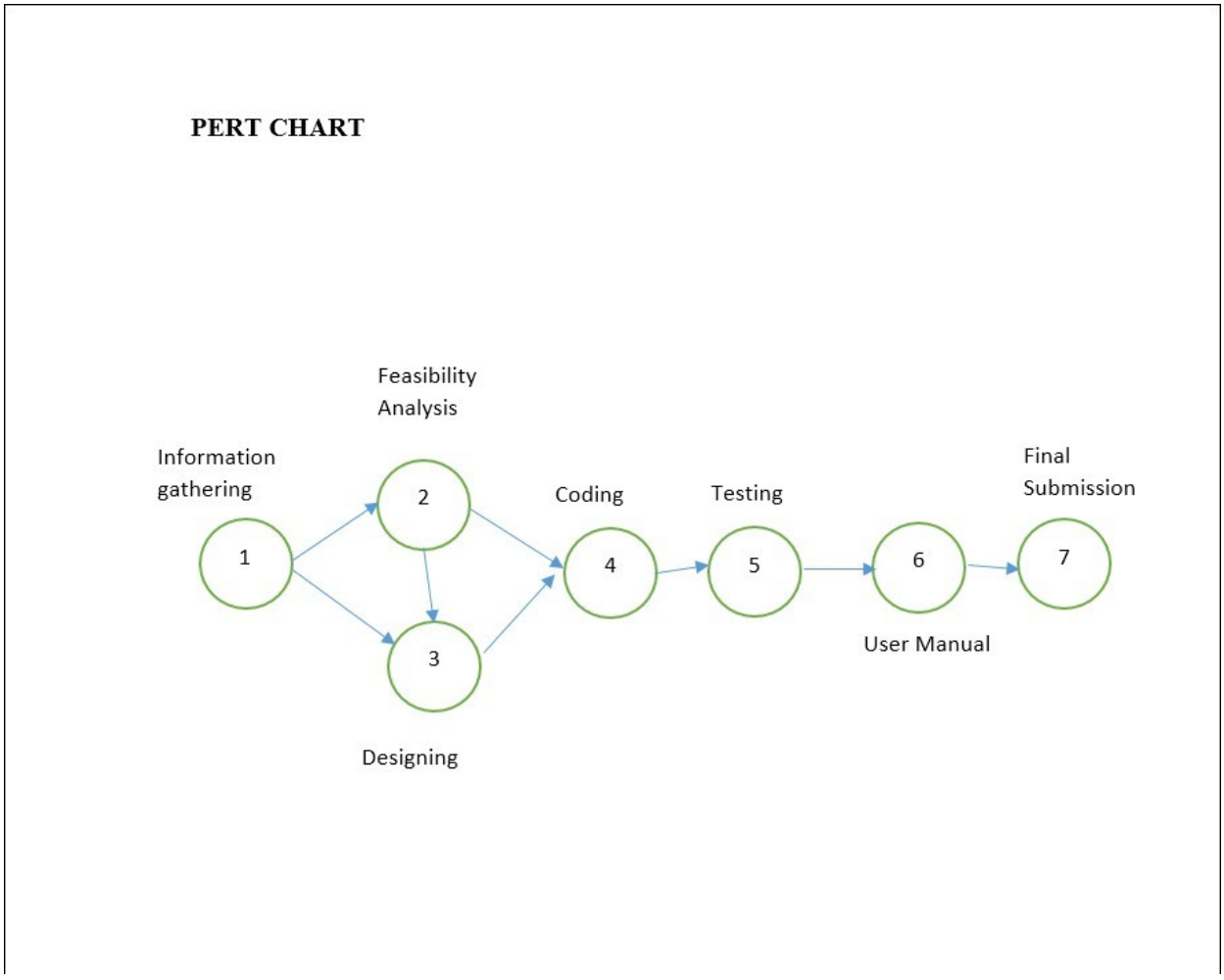


Fig.4.2.Pert Chart for Drowsiness Detection

GANTT CHART:

A Gantt chart is a type of bar chart that illustrates a project schedule, named after its inventor, Henry Gantt (1861–1919), who designed such a chart around the years 1910–1915. Modern Gantt charts also show the dependency relationships between activities and current schedule status.

A Gantt chart, or harmonogram, is a type of bar chart that illustrates a project schedule. This chart lists the tasks to be performed on the vertical axis, and time intervals on the horizontal axis. The width of the horizontal bars in the graph shows the duration of each activity. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements constitute the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities. Gantt charts can be used to show current schedule status using percent-complete shadings and a vertical "TODAY" line as shown here

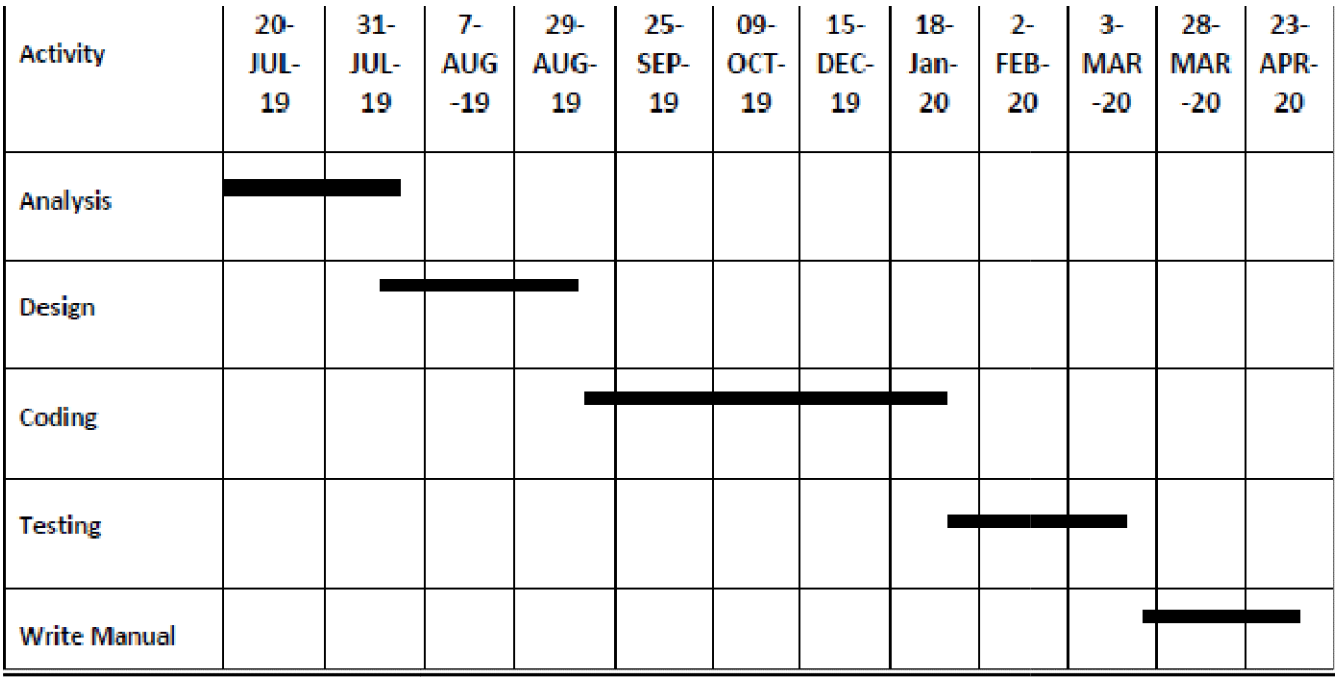


Fig.4.2.2.Ghantt Chart

SOFTWARE / HARDWARE REQUIREMENT:

Python based Machine Leaning and Deep Learning libraries will be exploited for the development and experimentation of the project.
Tools such as Anaconda Python and Libraries such as OpenCV , Tensorflow , keras and pygame will be used for this process.

4.3.1- Software Requirement :

1. Python IDLE
2. Dlib machine learning libraries (for frontal face recognition algorithm)
3. OpenCV for image processing purposes
4. 68 point face predictor data file (for facial parts recognition)
5. Jupyter Notebook
6. Chrome Browser

4.3.2- Hardware Requirement :

1.Processor – The minimum processor required is Intel i3 processor

2.RAM – The minimum RAM requirement is 4GB.

3.GPU – Some of the libraries of the project need to have NVidia GPU installed.

4.Graphic Card – The minimum Graphic Card required is 2 GB

CHAPTER 5

5.PRELIMINARY MODULE DESCRIPTION:

SOFTWARE MODULE:

Face Detection—

Here we deal with a real time situation where in our video gets recorded and needs processing to be done. But the processing can be done only on the image. Hence the captured image has to be divided into frames for further analyzation. In this stage of the process, we deal with identifying the face of the driver. By identifying the face of the driver we mean that detecting facial features or characters through the use of computer. The frame maybe random. Only facial related structures are identified other types of objects are ignored.

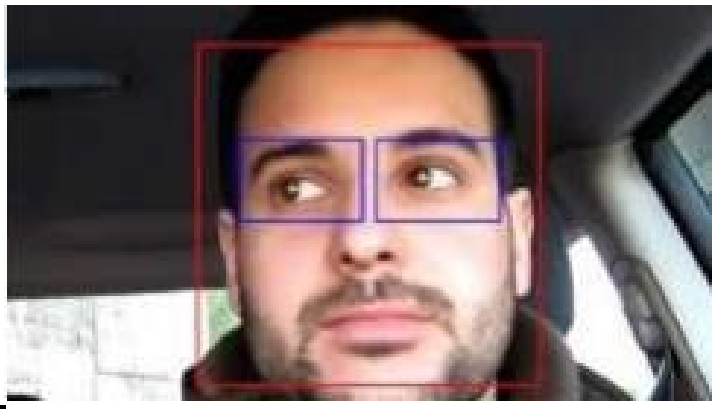


Fig.5.1.1 Image of Face Detection

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analyzing patterns based on the person's facial textures and shape While initially a form of computer application, it has seen wider uses in recent times on mobile platforms and in other forms of technology, such as robotics. It is typically used as access control in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems

Eye Detection—

After successfully detecting the face, we now focus on detecting the eye for further processing. This parameter is essential that helps in detecting the state of the driver whether the driver is active, sleepy or drowsy. The eye detection is done by analyzing the rate at which the driver blinks his eye. There is a particular threshold value set that decides the drowsiness. It is a time taking process and once the detection is done the result is matched with the particular threshold value set. After the successful eye detection , if the system detects the driver drowsy or sleepy, an alert alarm is sent to the driver to prevent accidents or any other mishap. The further demonstration for the same is given in the paper
for the same. We use a library called _dlib' that helps in easy detection of face as well as the eye. We use a method called _Euclidean Distance' to identify the distance between the eye lids that helps in detecting the drowsiness. A driver with small eyes, with a spectacles or any other feature can be easily detected with the help of this library.

Fig.5.1.2 Image for Eye Detection

To detect and track eye images with complex background, distinctive features of user eye are used. Generally, an eye-tracking and detection system can be divided into four steps: Face detection, eye region detection, pupil detection and eye tracking. To find the position of pupil, first, face region must be separated from the rest of the image using mixture of Gaussian, this will cause the images background to be non effective in our next steps. We used the horizontal projection obtained from face region, to separate a region containing eyes and eyebrow. This will result in decreasing the computational complexity and ignoring some factors such as bread
. Finally, in proposed method points with the highest values * of are selected as the eye candidate's. The eye region is well detected among these points. Color entropy in the eye region is used to eliminate the irrelevant candidates. In the next step, we perform eye tracking. In the proposed method, eye detection and tracking are applied on testing sets, gathered from different images of face data with complex backgrounds. Experiments indicate correct detection rate of 94.9%, which is indicative of the method's superiority and high robustness.

ALERT THE DRIVER:

After checking all the 68 facial landmarks as shown in the figure 5.1.3

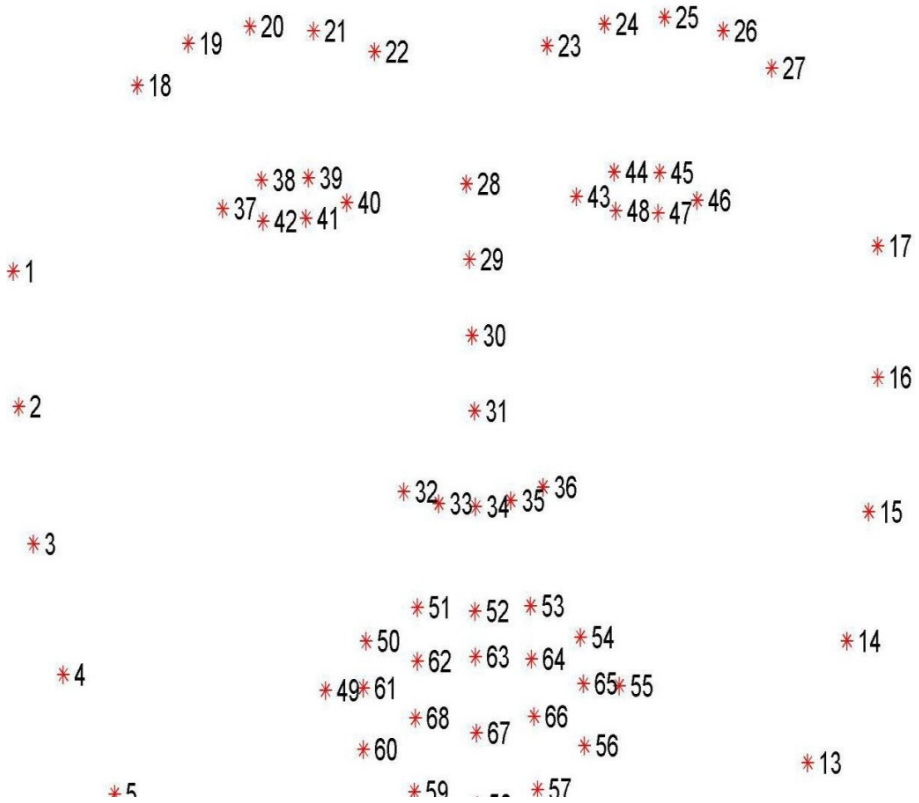


Fig .5.1.3. Image for 68 Facial Landmarks

1. If the eye aspect ratio falls below 0.3 threshold, we'll start counting the number of frames the person has closed their eyes for.

2. If the number of frames the person has closed their eyes in exceeds

3.EYE_AR_CONSEC_FRAMES=48

4.meaning that if a person has closed their eyes for 48 consecutive frames, we'll play the alarm sound.

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

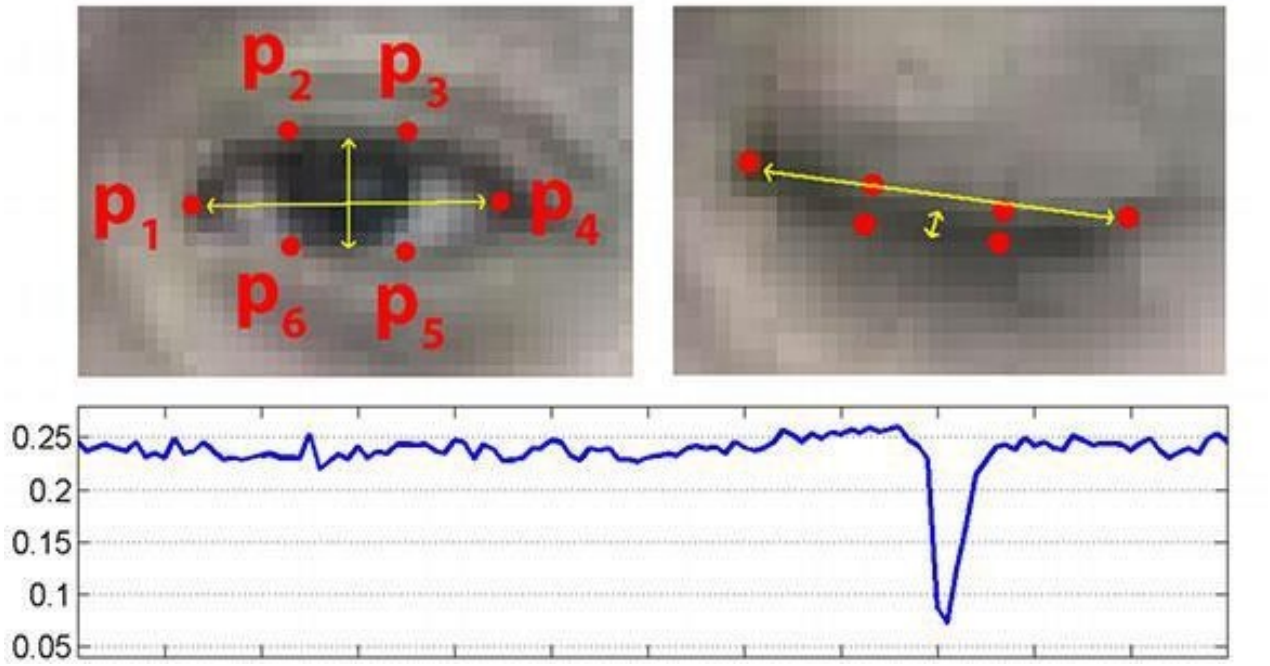


Fig 5.1.4 Image for the Aspect ratio of Eye

PROCESS OF WORKING:

Given below are the major steps involved in the processing of drowsiness detection model over here we have bifurcated into the four featues.These steps have enable us to achieve the more précised and accurate outcome.

The steps are as follows:

OPENING A WEB CAMERA:

The library which we have used over here is Opencv which enable us to open the web camera and extract or grab frames in order to proceed to face mapping step . we use Opencv to extract one frame at a time according to our instruction (may be per second) till the end of the video.

For example:

```
import cv2
data = []
labels = []
for j in [60]:
    for i in [10]:
    vidcap = cv2.VideoCapture(_drive/My Drive/Fold5_part2/‘ +     str(j) +‘/‘ + str(i) +
_.mp4‘)
            sec = 0
            frameRate = 1
            success, image = getFrame(sec)
            count = 0
            while success and count < 240:
              landmarks = extract_face_landmarks(image)
              if sum(sum(landmarks)) != 0:
                count += 1
                data.append(landmarks)
                labels.append([i])
                sec = sec + frameRate
                sec = round(sec, 2)
                success, image = getFrame(sec)
                print(count)
              else:
                sec = sec + frameRate
                sec = round(sec, 2)
                success, image = getFrame(sec)
                print(-not detected‖)
```

FACIAL LANDMARKING:

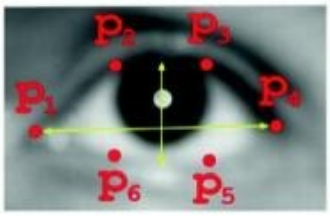
After the frame extractions done via Opencv we were able to extract around 100 odd frames per video and each video was run for sometime.we were able to collect 68 total landmarks per frame.These were the important data points which were used to extract the feature of our model.

FEATURE EXTRACTION:

As briefly alluded to earlier, based on the facial landmarks that we extracted from the frames of the videos, we ventured into developing suitable features for our classification model. While we hypothesized and tested several features, the four core features that we concluded on for our final models were eye aspect ratio, mouth aspect ratio, pupil circularity, and finally, mouth aspect ratio over eye aspect ratio.

Eye Aspect Ratio (EAR)

EAR, as the name suggests, is the ratio of the length of the eyes to the width of the eyes. The length of the eyes is calculated by averaging over two distinct vertical lines across the eyes as illustrated in the figure below.

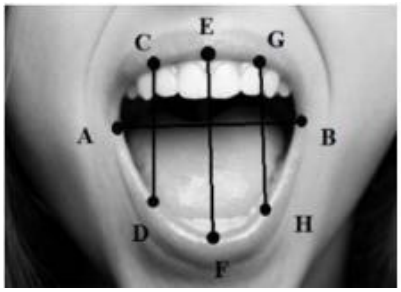


$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Our hypothesis was that when an individual is drowsy, their eyes are likely to get smaller and they are likely to blink more. Based on this hypothesis, we expected our model to predict the class as drowsy if the eye aspect ratio for an individual over successive frames started to decline i.e. their eyes started to be more closed or they were blinking faster.

Mouth Aspect Ratio (MAR)

Computationally similar to the EAR, the MAR, as you would expect, measures the ratio of the length of the mouth to the width of the mouth. Our hypothesis was that as an individual becomes drowsy, they are likely to yawn and lose control over their mouth, making their MAR to be higher than usual in this state.



$$MAR = \frac{|EF|}{|AB|}$$

Pupil Circularity (PUC)

PUC is a measure complementary to EAR, but it places a greater emphasis on the pupil instead of the entire eye.

$$Circularity = \frac{4 * \pi * Area}{perimeter^2} \qquad Area = \left(\frac{\text{Distance}(p2, p5)}{2}\right)^2 * \pi$$

$$Perimeter = Distance(p1, p2) + Distance(p2, p3) + Distance(p3, p4) + Distance(p4, p5) + Distance(p5, p6) + Distance(p6, p1)$$

CHAPTER 6

6. SYSTEM DESIGNING:

SYSTEM ARCHITECTURE:

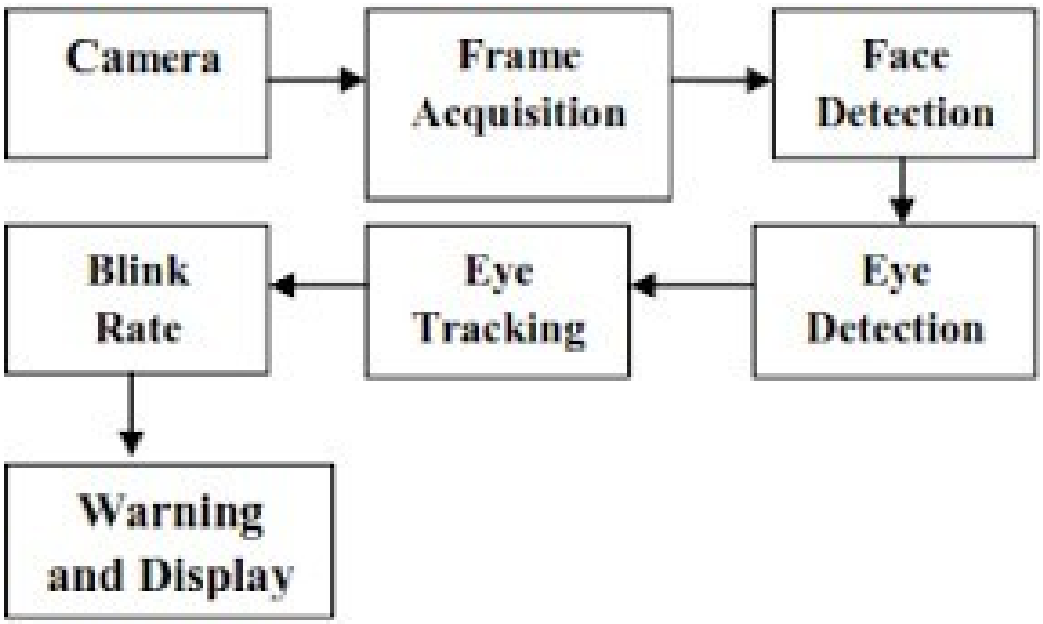


Fig.6.1.System Architecture for Drowsiness Detection

FLOW DAIGRAM:

A flow diagram is a way of representing a flow of data through a process or a system (usually a information system). The FD also provides information about the outputs and inputs of each entity and the process itself. A flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.
There are several notations for displaying data-flow diagrams. The notation presented above was described in 1979 by Tom DeMacro as part of Structured Analysis.
For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

The data-flow diagram is part of the structured-analysis modeling tools. When using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan.

Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.
DFD consists of processes, flows, warehouses, and terminators. There are several ways to view these DFD components
.

Process –

The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.

Data Flow –

Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows. Material shifts are modeled in systems that are not merely informative. Flow should only transmit one type of information (material). The arrow shows the flow direction (it can also be bi-directional if the information to/from the entity is logically dependent - e.g. question and answer). Flows link processes, warehouses and terminators

Terminator –

The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations (eg a bank), groups of people (e.g. customers), authorities (e.g. a tax office) or a department (e.g. a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modeled system communicates.

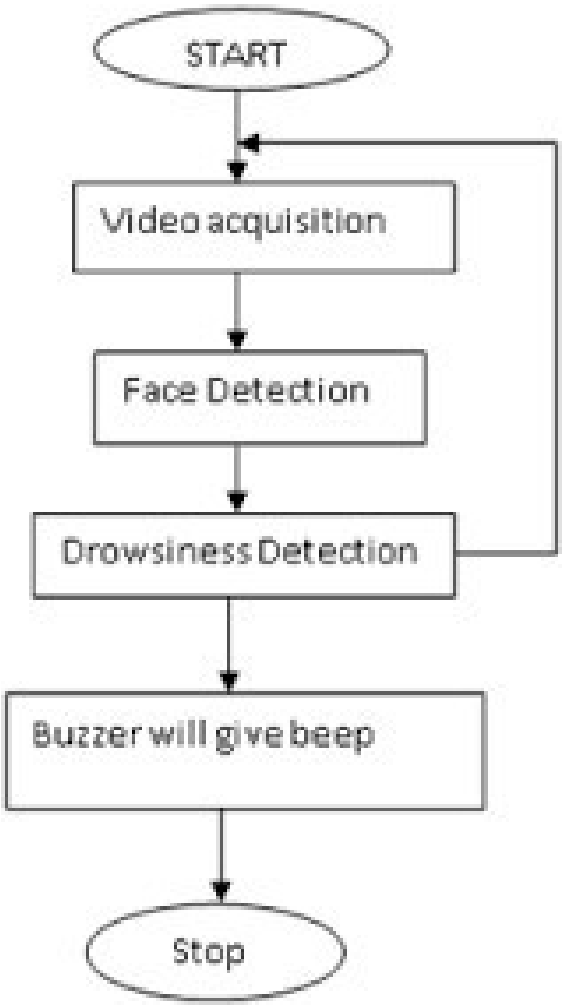


Fig.6.2. Flow daigram for Drowsiness Detection

CHAPTER 7

7. CODING:

```
# USAGE

# python detect_drowsiness.py --shape-predictor shape_predictor_68_face_landmarks.dat

# python detect_drowsiness.py --shape-predictor shape_predictor_68_face_landmarks.dat --
alarm alarm.wav


# import the necessary packages

from scipy.spatial import distance as dist

from imutils.video import VideoStream

from imutils import face_utils

from threading import Thread

import numpy as np

import playsound

import argparse

import imutils

import time

import dlib

import cv2


pather="alarm.wav"

def sound_alarm(path):

        # play an alarm sound

        playsound.playsound("alarm.wav")
```

```
def eye_aspect_ratio(eye):
        # compute the euclidean distances between the two sets of
        # vertical eye landmarks (x, y)-coordinates
        A = dist.euclidean(eye[1], eye[5])
        B = dist.euclidean(eye[2], eye[4])
        # compute the euclidean distance between the horizontal
        # eye landmark (x, y)-coordinates
        C = dist.euclidean(eye[0], eye[3])
        # compute the eye aspect ratio
        ear = (A + B) / (2.0 * C)
        # return the eye aspect ratio
        return ear

# construct the argument parse and parse the arguments
# define two constants, one for the eye aspect ratio to indicate
# blink and then a second constant for the number of consecutive
# frames the eye must be below the threshold for to set off the
# alarm
EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 48

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

```
# initialize dlib's face detector (HOG-based) and then create
# the facial landmark predictor
print("[INFO] loading facial landmark predictor...")
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

# grab the indexes of the facial landmarks for the left and
# right eye, respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

# start the video stream thread
print("[INFO] starting video stream thread...")
vs = VideoStream(src=0).start()
time.sleep(1.0)

# loop over frames from the video stream
while True:
	# grab the frame from the threaded video file stream, resize
	# it, and convert it to grayscale
	# channels)
	frame = vs.read()
	frame = imutils.resize(frame, width=450)
	gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
# detect faces in the grayscale frame
rects = detector(gray, 0)

# loop over the face detections
for rect in rects:
	# determine the facial landmarks for the face region, then
	# convert the facial landmark (x, y)-coordinates to a NumPy
	# array
	shape = predictor(gray, rect)
	shape = face_utils.shape_to_np(shape)

	# extract the left and right eye coordinates, then use the
	# coordinates to compute the eye aspect ratio for both eyes
	leftEye = shape[lStart:lEnd]
	rightEye = shape[rStart:rEnd]
	leftEAR = eye_aspect_ratio(leftEye)
	rightEAR = eye_aspect_ratio(rightEye)

	# average the eye aspect ratio together for both eyes
	ear = (leftEAR + rightEAR) / 2.0

	# compute the convex hull for the left and right eye, then
	# visualize each of the eyes
	leftEyeHull = cv2.convexHull(leftEye)
```

```
rightEyeHull = cv2.convexHull(rightEye)
cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)

# check to see if the eye aspect ratio is below the blink
# threshold, and if so, increment the blink frame counter
if ear < EYE_AR_THRESH:
        COUNTER += 1

        # if the eyes were closed for a sufficient number of
        # then sound the alarm
        if COUNTER >= EYE_AR_CONSEC_FRAMES:
                # if the alarm is not on, turn it on
                if not ALARM_ON:
                        ALARM_ON = True

                        # check to see if an alarm file was supplied,
                        # and if so, start a thread to have the alarm
                        # sound played in the background
                        if pather!= "":
                                t = Thread(target=sound_alarm,
                                        args=("alarm.wav",))
                                t.deamon = True
                                t.start()
```

```
                        # draw an alarm on the frame
                        cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),
                                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

                # otherwise, the eye aspect ratio is not below the blink
                # threshold, so reset the counter and alarm
                else:
                        COUNTER = 0
                        ALARM_ON = False
                # draw the computed eye aspect ratio on the frame to help
                # with debugging and setting the correct eye aspect ratio
                # thresholds and frame counters
                cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        # show the frame
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF
        # if the `q` key was pressed, break from the loop
        if key == ord("q"):
                break
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

8. RESULT:

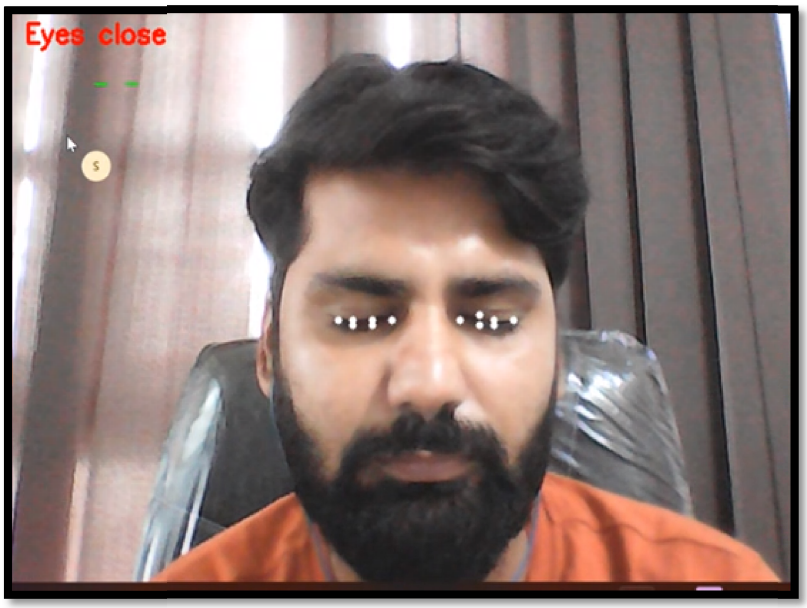
After we compile the source code of the project we get the following output:

- Checking the aspect ratio of eyes as shown in the figure 8.1

Fig.81.Checking aspect ratio

- Generating Drowsiness alert while the aspect ratio is less then 0.3 threshold as shown in the figure 8.2

Fig.8.2.Generating Drowsiness Alert



CHAPTER 9

CONCLUSION:

The proposed system helps the driver to stay awake while driving and minimizes sleep-induced accidents. The model has proved to be efficient and since it has been facilitated with camera module is one the most compact and economic system that is adaptable for different motor vehicles. The best models (those whose rates of successful detection or prediction are the highest) use information about the eyelid closure, gaze and head movements and driving time. Performance on the prediction is very promising, since the model can predict to within5 min when the driver‘s state will become impaired. Moreover, modelling drowsiness as continuum can lead to more precise detection system offering refined results beyond simply detecting whether the driver is alert or drowsy

FUTURE WORK:

In this work an inbuilt camera has been used for capturing video of the driver instead of this a wireless camera can be used whose video is transferred to control system wirelessly, processed and give alert to a wireless buzzer system. This work includes only yawn detection and sleep detection, Head movement detection can also be added to this system to give and additional feature. A night vision camera can be used for the same system to increase the usability in all light conditions whether dark or dim.

REFERNCES:

[1] Lee Boon Leng, Lee Boon Giin and Wan-Young Chung –Wearable Driver Drowsiness Detection System Based on Biomedical and Motion Sensors‖, 2015 IEEE

[2] Jaeik Jo, Sung Joo Lee, Kang Ryoung Park, Ig-Jae Kim and Jaihie Kim, –Detecting driver drowsiness using feature-level fusion and user-specific classification‖, 2013 Elsevier Ltd.

[3] Amna Rahman , Mehreen Sirshar and Aliya Khan , –Real Time Drowsiness Detection using Eye Blink Monitoring‖, Software Engineering Conference (NSEC), 2015 National, 2016, IEEE

[4] Anjali K U, Athiramol K Thampi, Athira Vijayaraman, Franiya Francis M, Jeffy James N and Bindhu K Rajan –Real-Time Nonintrusive Monitoring and Detection of Eye Blinking in view of Accident Prevention due to Drowsiness‖, 2016 International Conference on Circuit, Power and Computing Technologies [ICCPCT], IEEE

[5] Brandy Warwick1, Nicholas Symons1, Xiao Chen1, Kaiqi Xiong , –Detecting Driver Drowsiness using Wireless Wearables‖, 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems

[6]M. Stork, J. Skala, P. Weissar, R. Holota, Z. Kubik , –Various Approaches to Driver Fatigue Detection: A Review‖, University of West Bohemia, 2015

[7] Kai-Wei Ke, Muhammad R. Zulman, Ho-Ting Wu, Yu-Fu Huang and Jayasakthi Thiagarajan, –Drowsiness Detection System Using Heartbeat Rate in Android-based Handheld Devices‖, 2016 First International Conference on Multimedia and Image Processing 978-1-4673-8940-2/16 $31.00 © 2016 IEEE 100 2016 First International Conference on Multimedia and Image Processing

[8] Duy Tran, Eyosiyas Tadesse and Weihua Sheng, Yuge Sun, Meiqin Liu and Senlin Zhang –A Driver Assistance Framework based on Driver Drowsiness Detection‖, The 6th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems June 19-22, 2016, Chengdu, China

[9] Suparna Sahabiswas and Sourav Saha –Drunken driving detection and prevention models using Internet of things‖, 2016 IEEE