# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**"Jnana Sangama", Belagavi-590 018, KARNATAKA, INDIA**

A

PROJECT REPORT

ON

# "DROWSINESS AND DISTRACTION ALERTING SYSTEM"

**Submitted in partial fulfillment of the requirements for the award of the degree of**

## BACHELOR OF ENGINEERING
## IN
## COMPUTER SCIENCE AND ENGINEERING


**Submitted by:**

| | |
|---|---|
| **AKANSHA JAIN** | **1JS16CS005** |
| **KARTIK NESARI** | **1JS15CS047** |
| **PRATIK SHANKAR** | **1JS16CS072** |
| **PRAWALIKA TEWARI** | **1JS16CS073** |

**Under the guidance of**
**Dr. PRABHUDEV JAGADEESH**
**Professor**
**Department of Computer Science and Engineering**
**JSSATE, Bengaluru**




**JSS ACADEMY OF TECHNICAL EDUCATION, BENGALURU**
**Department of Computer Science and Engineering**


**2019 – 2020**

**JSS MAHAVIDYAPEETHA, BENGALURU**

# JSS ACADEMY OF TECHNICAL EDUCATION

JSS Campus, Uttarahalli-Kengeri Main Road, Bengaluru - 560060

## Department of Computer Science and Engineering

# CERTIFICATE

This is to certify that the Project work entitled **"DROWSINESS AND DISTRACTION ALERTING SYSTEM"** is a bonafied work carried out by **MR. KARTIK NESARI (1JS15CS047), MS. AKANSHA JAIN (1JS16CS005), MR. PRATIK SHANKAR (1JS16CS072)** and **MS. PRAWALIKA TEWARI (1JS16CS073)** in partial fulfillment of the degree of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the academic year 2019 - 2020. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report. The project has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

| | | |
|---|---|---|
| **Dr. Prabhudev Jagadeesh** | **Dr. Naveen N C** | **Dr. Mrityunjaya V Latte** |
| **Professor**<br>**Dept. of CSE**<br>**JSSATE, Bengaluru** | **Professor & Head**<br>**Dept. of CSE**<br>**JSSATE, Bengaluru** | **Principal**<br>**JSSATE, Bengaluru** |

**External Viva**

| **Name of the Examiners** | **Signature with Date** |
|---|---|
| **1)……** | **……** |
| **2)……** | **……** |

# ABSTRACT

The goal is to create a device that can detect drowsiness or any signs of distraction from a driver while they're driving. Not only does this help the driver itself but it also helps the people around the driver remain safe and sound. Many luxury vehicle brands are now starting to provide this feature in their vehicles but this system is still something that's not perfected and in fact, still requires much work to be done on. Our project aims at being able to detect the drivers state with a high level of accuracy. This project makes use of image detection, machine learning, neural networks and natural language processing concepts. This project is created by using the python programming language and OpenCV.

# ACKNOWLEDGEMENTS

With utmost joy and satisfaction, we submit this Project Report on "DROWSINESS AND DISTRACTION ALERTING SYSTEM". This has been completed as a part of the curriculum of Visvesvaraya Technological University.

The satisfaction that accompanies the successful completion of our project would be incomplete without mentioning the people who made it possible, whose constant guidance and encouragement crowns all the efforts with success.

We take immense pleasure in thanking Dr. Mrityunjaya V Latte, Principal, JSSATE, Bengaluru, for being kind enough to provide us with an opportunity to work the Project in this institution.

We are also thankful to Dr Naveen N C**,** Professor and Head of Department of Computer Science and Engineering, for his co-operation and encouragement at all moments of approach.

We are thankful to Mrs. Snehalatha N and Mrs. Rajeshwari KS, Assistant Professor, Project Coordinator, for their cooperation and support.

We are thankful to our Project guide Dr. Prabhudev Jagadeesh, Professor, for his constant support and encouragement.

We wish to thank every teaching and non-teaching faculty of out department for always being there to support and guide us.

**AKANSHA JAIN (1JS16CS005),**

**KARTIK NESARI (1JS15CS047),**

**PRATIK SHANKAR (1JS16CS072)**

**PRAWALIKA TEWARI (1JS16CS073)**

# Table of Contents

# List of Figures

# Chapter 1

# INTRODUCTION

## 1.1 Overview

Road fatalities are one of the leading causes of death in the world. Of the road fatalities, India ranks first in the total number of deaths by road accidents in the world. India alone had One Lakh Fifty-One thousand deaths due to road accidents alone last year. India constitutes about Eleven percent of road fatalities across the globe which is a lot considering that there are One Ninety-Five countries in the world. Most road accidents occur on highways when a driver loses his handling over his vehicle. Many a times it's only the driver who can prevent this from happening but there might be other reasons for road accidents too with one of the most probable reasons being drowsy while driving. Unlike occurrences such as brake fail or steering wheel jam which can't be notified on the spot, measures to prevent drowsiness can always be taken to ensure that a driver keeps themselves, the passengers and the ones around them safe.

Many vehicle companies such as Audi, BMW, Bosch, Ford, Honda etc. now incorporate fatigue detection systems in their vehicles to alert driver of drowsiness and microsleep.

Driver drowsiness can be detected by using techniques such as:

- **Steering pattern monitoring**

  Primarily uses steering input from electrical power steering mechanism. Monitoring a driver in this manner only works as long a driver actually steers a vehicle actively instead of an automatic lane-keeping system.

- **Vehicle position in lane monitoring**

  Uses lane monitoring camera. Monitoring a driver in this manner only works as long a driver actually steers a vehicle actively rather than an automatic lane-keeping system.

- **Driver eye/face monitoring**

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

- **Physiological measurement**

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

Our project is created by using the Driver eye/face monitoring technique. Concepts from Machine Learning, Image recognition have been applied to detect any notions of drowsiness in a driver. The advantage of this system is that it is relatively inexpensive to build. All you

need is a IC such as Raspberry PI/Arduino, a dashcam for real time video feed and a speaker to sound the alert sounds. The system works with a high level of accuracy and works quickly.

## 1.2 Scope

A high number of fatalities are a result of road accidents. One of major reasons for road accidents is due to the driver driving whilst being drowsy or distracted. Given this fact, a system that can detect signs of drowsiness or distraction can help prevent a driver from losing his grip over the wheels. This not only helps the driver and his passengers but also to those in his vicinity at any given point of time.

Major automobile manufacturers have already started incorporating this feature into their vehicles, first with their luxury models and then with their economy models. While that's correct, it's also correct that the system is still in its infancy stages with lots of opportunities to grow further. Current technology is not only inefficient but is also not readily available in all the vehicles due to the higher overheads. Different methodologies have been used to detect drowsiness but nothing shows a person getting drowsy more than their face.

As such, the current technology only allows for dashcams which while not being large in size do still cover a drivers' view of the road if placed in front. It'd help much more if there are more cameras placed around the driver to catch a drivers' face from every angle. But this increases not only the total load but also the amount of processing time thus making the drowsiness detection lack in terms of speed. Thus, simplicity is the key here unless there's a breakthrough in hardware capabilities.

In the near future, these systems could be imagined being incorporated into every type of vehicle. Smaller cameras would mean that the driver would have a better view of his surroundings. Better IC chips would allow the system to make use of more peripheral devices to help detect drowsiness or distraction. Current studies into auto drive technology and the drowsiness detection system can be integrated together to create an adaptive auto-drive system that can take over the vehicle if the driver feels drowsy or help the driver manage the steering of the vehicle whilst the drowsiness detection system awakens the driver.

## 1.3 Assumptions

The objective of our project is to process a large amount of real time data to analyze the given features in a frame to detect signs of drowsiness. Our system is based on the following assumptions:

1. There's no lag in the video feed
2. There's no lag in the processing
3. The vehicle has the space to incorporate this system at the drivers' seat.
4. The camera is set up in the position it was trained from originally in any other vehicle.
5. The windows are closed and the car is noise isolated from the inside

## 1.4 Existing System

Numerous automotive manufacturers have implemented the drowsiness detection systems into their vehicles. Some of the most known brands to use this system are Audi, BMW, Bosch, Citroen, Ford, Honda, Hyundai etc.

The current system makes use of one or more of the steering wheels, the pulse of a driver, the face of a driver or the lane of the vehicle to make prediction whether a driver is drowsy or not. Bosch makes use of Steering-angle sensor to determine the steering angle and the steering angle velocity.

The steering-angle sensor is fixed onto the steering shaft. In each measuring gear wheel a magnet is mounted, whose field changes its direction in accordance with the rotational movement. Below each magnet, a GMR sensor element is located to detect the angle position of the magnet above. By means of mathematic functions, it is possible to determine the absolute steering-wheel angle from the position of the measuring gears. In addition, this function enables error correction and a plausibility test of individual signals.

Audi created the rest recommendation function, which analyses, the driver's steering motions among other things at speeds between 65 and 200 km/h (40.39 and 124.27 mph). If the analysis reveals indications that the driver's attentiveness is declining, the system recommends a rest by illuminating an indicator in the driver information system and sounding an acoustic signal.

In Citroen, The Lane Departure Warning System (LDWS) combats drowsiness at the wheel. This is performed by a camera fitted into the upper windscreen to monitor lane markings. If it senses that these lines are being crossed when the vehicle is traveling at speeds in excess of 40mph, without an indicator being activated, it will correct the vehicle's line and alert the driver.

Ford's Driver Alert System constantly monitors the driving behaviour and is designed to detect any changes that could be caused by fatigue. It uses a forward-looking camera to monitor the vehicle position in the lane and calculate a vigilance level for the driver. If the system identifies you're becoming less vigilant, a warning icon appears in the display, suggesting you take a

break. If your driving alertness further declines, the message is repeated and combined with a chime. Driver Alert depends on the driver responding to the warning and taking a sufficiently refreshing break.

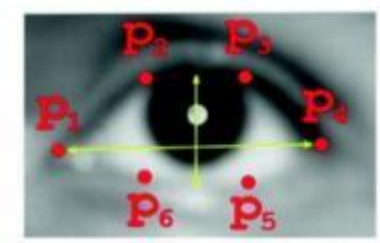
## 1.5 Proposed System

The current systems used by automobiles are more hardware dependent which can cause problems if any of its parts start malfunctioning. Also, the costs of implementing them are high. The proposed system makes the use of vision and audio cues to detect any signs of Drowsiness or distraction. Given the fact that you have a video camera we perform the following tasks:

1. Split the video into individual frames
2. Feature classification
3. Process each frame using algorithms
4. If drowsy, deploy the alerting system.

In the first step, we make the use of OpenCV to split the video in a stream of individual frames. OpenCV (Open Source Computer Vision Library) is a library which is open sourced and is dedicated towards fields such as computer vision and machine learning. OpenCV was built to provide a common platform which could help industries commercial products in a faster in easier manner.

In the second step, each frame is processed using the OpenCV library. The head is detected using the facial landmarks. Then, the Eye-Aspect-Ratio of the eyes is calculated. 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.




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

**Figure 1.1**: Eye-Aspect-Ratio

This is used because of the hypothesis that the eyes squint when a person yawns [21].

In the third step, after feature extraction, a series of modelling techniques are applied to classify the model in the drowsy/not drowsy state.

If the image was classified as drowsy then an alerting system is activated that works based on voice activation protocols.

## 1.6 Problem Statement

Given a person is in the drivers' seat, the problem is to detect the person. The model should then detect whether a person is feeling drowsy or not. On the other hand, the model should also be capable of detecting if the driver is getting distracted while driving. In any of such cases the system should snap the driver out of his sleep or delusions by the use of sound. The system should be able to communicate intelligently with the driver and keep him engaged just enough to keep the driver awake.

# Chapter 2

# LITERATURE SURVEY

## 2.1 Visual Object Tracking

Visual object tracking is a crucial problem in computer vision. It has a wide range of applications in fields such as human-computer interaction, behavior recognition, robotics, and surveillance. Visual object tracking estimates the target position in each frame of the image sequence, given the initial state of the target in the previous frame. Lucas and Kanade proposed that the tracking of the moving target can be realized using the pixel relationship between adjacent frames of the video sequence and displacement changes of the pixels. However, this algorithm can only detect the medium-sized target that shifts between two frames. With the recent advances of the correlation filter in computer vision Bolme proposed the Minimum Output Sum of Squared Error (MOSSE) filter, which can produce stable correlation _lters to track the object. Although the MOSSE's computational efficiency is high, its algorithm precision is low, and it can only process the gray information of a single channel.

Based on the correlation filter, Li and Zhu utilized HoG, color-naming features and the scale adaptive scheme to boost object tracking. Danelljan *et al.* used HOG and the discriminative correlation filter to track the object. SAMF and DSST solve the problem of deformation or change in scale when the tracking target is rotating. Further, they solve the problem of the tracker's inability to track object adaptively and the low operation speed. With the development of the deep-learning algorithm, some scholars combine deep learning and the correlation filter to track the mobile target [1]. Although these algorithms have better precision than the track algorithms based on the correlation filter, their training is time-consuming. Hence, these algorithms cannot track the object in real-time in a real environment.

In this study, we propose a MC-KCF algorithm based on the correlation filter and deep learning. This algorithm uses CNN and MTCNN to offset the KCF's limitation and uses the KCF to track objects. Thus, the algorithm can track the driver's face in real-time using our system.

## 2.2 Facial Landmarks Recognition

The purpose of facial key-points recognition is that getting the crucial information about locations of eyebrows, eyes, lips and nose in the face. With the development of deep learning, it is the first time for Sun *et al.* to introduced DCNN based on CNN to detect human facial

keypoints. This algorithm only recognizes 5 facial keypoints, albeit its speed is very fast. To get a higher precision for facial key points recognition, Zhou *et al.* employed FACE++ which optimizes DCNN and it can recognize 68 facial keypoints, but this algorithm includes too much of a model and the operation of this algorithm is very complicated. Wu *et al.* proposed Tweaked Convolutional Neural Networks (TCNN) which is based on Gaussian Mixture Model (GMM) to improve different layers of CNN. However, the robustness of TCNN depends on data excessively. Kowalski *et al.* introduced Deep Alignment Network (DAN) to recognize the facial keypoints, which has better performance than other algorithms. Unfortunately, DAN needs vast models and calculation based on complicated functions. So in order to meet the requirement about real time performance, DriCare uses Dlib [25] to recognize facial keypoints [1].

## 2.3 Driver Drowsiness Detection

Driver drowsiness detection can be divided into two types: contact approaches and non-contact approaches. In contact approaches, drivers wear or touch some devices to get physiological parameters for detecting the level of their fatigue. Warwick *et al.* implemented the BioHarness 3 on the driver's body to collect the data and measure the drowsiness. Li *et al.* used a smartwatch to detect driver drowsiness based on electroencephalographic (EEG) signal. Jung *et al.* reformed the steering wheel and set an embedded sensor to monitor the electrocardiogram (ECG) signal of the driver. However, due to the price of contact approaches and installation, there are some limitations which cannot be implemented ubiquitously. The other method employs a tag-free approaches to detect the driver drowsiness, where the measured object does not need to contact the driver. For example, Omidyeganeh *et al.* used the driver's facial appearance captured by the camera to detect the driver drowsiness, but this method is not real-time. Zhang and Hua used fatigue facial expression reorganization based on Local Binary Pattern (LBP) features and Support Vector Machines (SVM) to estimate the driver fatigue, but the complexity of this algorithm is bigger than our algorithm. Moreover, Picot *et al.* proposed a method that uses electrooculogram (EOG) signal and blinking feature for drowsiness detection. Akrout and Mahdi and Oyini Mbouna *et al.* used a fusion system for drowsiness detection based on eye state and head position. Different from these methods, we employ simple formulae and evaluations, which make the results easier to measure [1].

# Chapter 3

# SYSTEM REQUIREMENTS

System requirements are the configuration that a system must have for a hardware or software application to run easily and proficiently. If these requirements are not satisfied, they can lead to installation or performance problems. Installation problems may prevent a device or an application from getting installed. Performance problems may cause a product to malfunction or perform below expectation or even hang or crash.

## 3.1 Hardware Requirements

The section of hardware configuration is an important task related to the software development insufficient random access memory may affect adversely on the speed and efficiency of the entire system. The process should be powerful to handle the entire operations. The hard disk should have sufficient capacity to store the file and application.

| | | |
|---|---|---|
| GPU | - | Nvidia / AMD |
| Processor | - | Intel Core i3/i5 |
| RAM | - | 8GB (recommended) |
| Peripheral Devices | - | Dash Camera / Camera, Speaker, microphone |

## 3.2 Software Requirements

A major element in building a system is the section of compatible software since the software in the market is experiencing in geometric progression. Selected software should be acceptable by the firm and one user as well as it should be feasible for the system.

This document gives a detailed description of the software requirement specification. The study of requirement specification is focused specially on the functioning of the system. It allows the developer or analyst to understand the system, function to be carried out the performance level to be obtained and corresponding interfaces to be established.

### 3.2.1 OpenCV

OpenCV (Open Source Computer Vision Library) [2] is an open-source library that includes several hundreds of computer vision algorithms. The documentation now describes the so called OpenCV 2.x API, which is essentially a C++ API, as opposed to the C-based OpenCV 1.x API as OpenCV has been upgraded to a newer version to include more sophisticated algorithms and data structures.

The OpenCV library contains a lot of shared and static packages. The following modules are available:

- **Core functionality** (**core**) – The Core Functionality module is a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions which are used by all other modules.
- **Image Processing** (**imgproc**) – As the name suggests, this module is used for calling functions that ease the task of image processing. The image processing module includes linear and non-linear image filtering, geometrical image transformations, colour space conversion, histograms, and so on.
- **Video Analysis** (**video**) – Video analysis means scanning a video (premade or live) to look for specific object(s) in the given viewport. A video analysis module includes motion estimation, background subtraction, and object tracking algorithms.
- **Camera Calibration and 3D Reconstruction** (**calib3d**) - basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.
- **2D Features Framework** (**features2d**) - salient feature detectors, descriptors, and descriptor matchers.
- **Object Detection** (**objdetect**) - detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).
- **High-level GUI** (**highgui**) – This gives the user signals in form of better visuals which are easier to interpret.
- **Video I/O** (**videoio**) – OpenCV provides an easy-to-use interface to video capturing and video codecs.
- ... some other helper modules, such as FLANN and Google test wrappers, Python bindings, and others.

The library has more than Two-Thousand-Five-Hundred optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms are capable of detecting and recognize faces, identifying objects, classifying human actions in videos, track camera movements, track moving objects, extracting 3D models of objects, producing 3D point clouds from stereo cameras, stitching images together to produce a high resolution image of an entire scene, finding similar images from an image database and so on [2].

### 3.2.2 SciPy

SciPy is an open-source Python library used for scientific computing and technical computing which can be taken for free of charge. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

SciPy consists of the NumPy array object and is a component of the NumPy stack which incorporates tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPy stack is analogous in nature and working to other applications such as MATLAB, GNU Octave, and Scilab and features a similar user base. The NumPy stack is additionally sometimes also called as the SciPy stack.

The SciPy library is currently distributed under the BSD license, and is open sourced which suggests that different developers can pitch in to enhance this library. It is also supported by NumFOCUS, a community foundation for supporting reproducible and accessible science.

Available sub-packages include:

- **constants**: physical constants and conversion factors (since version 0.7.0)
- **cluster**: hierarchical clustering, vector quantization, K-means
- **fftpack**: Discrete Fourier Transform algorithms
- **integrate**: numerical integration routines
- **interpolate**: interpolation tools
- **io**: data input and output
- **lib**: Python wrappers to external libraries

- **linalg**: linear algebra routines
- **misc**: miscellaneous utilities (e.g. image reading/writing)
- **ndimage**: various functions for multi-dimensional image processing
- **optimize**: optimization algorithms including linear programming
- **signal**: signal processing tools
- **sparse**: sparse matrix and related algorithms
- **spatial**: KD-trees, nearest neighbors, distance functions
- **special**: special functions
- **stats**: statistical functions
- **weave**: tool for writing C/C++ code as Python multiline strings

The basic data structure employed by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for linear algebra, Fourier transformations, and random number generation, but not with the generality of the equivalent functions in SciPy. NumPy can also be used as an efficient multidimensional container of data with arbitrary datatypes [3]. This enables NumPy to seamlessly and speedily integrate with different types of databases. Newer NumPy array code is used as opposed to using Numeric as an array data type in the newer version of SciPy.

### 3.2.3 Numpy

NumPy 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 targets the CPython reference implementation of Python, which is a nonoptimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops using NumPy.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging. [7]

**The ndarray data structure**

The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for memory-mapped ndarrays.

**Limitations**

Inserting or appending entries to an array is not as trivially possible as it is with Python's lists. The np.pad(...) routine to extend arrays actually creates new arrays of the desired shape and padding values, copies the given array into the new one and returns it. NumPy's np.concatenate([a1,a2]) operation does not actually link the two arrays but returns a new one, filled with the entries from both given arrays in sequence. Reshaping the dimensionality of an array with np.reshape(...) is only possible as long as the number of elements in the array does not change. These circumstances originate from the fact that NumPy's arrays must be views on contiguous memory buffers. A replacement package called Blaze attempts to overcome this limitation.

Algorithms that are not expressible as a vectorized operation will typically run slowly because they must be implemented in "pure Python", while vectorization may increase memory complexity of some operations from constant to linear, because temporary arrays must be created that are as large as the inputs. Runtime compilation of numerical code has been implemented by several groups to avoid these problems; open source solutions that interoperate with NumPy include scipy.weave, numexpr and Numba. Cython and Pythran are static-compiling alternatives to these.

### 3.2.4 Dlib

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open source licensing allows you to use it in any application, free of charge [5].

Core to the development philosophy of dlib is a dedication to portability and ease of use. Therefore, all code in dlib is designed to be as portable as possible and similarly to not require a user to configure or install anything. To help achieve this, all platform specific code is confined inside the API wrappers. Everything else is either layered on top of those wrappers or is written in pure ISO standard C++. Currently the library is known to work on OS X, MS Windows, Linux, Solaris, the BSDs, and HP-UX.

**Organization**

The library can be thought of as a collection of components. Each component always consists of at least two separate files, a specification file and an implementation file. The specification files are the ones that end with _abstract.h. Each of these specification files don't actually contain any code and they even have preprocessor directives that prevent any of their contents from being included. Their purpose is purely to document a component's interface in a file that isn't cluttered with implementation details the user shouldn't need to know about.

The next important concept in dlib organization is multi-implementation components. That is, some components provide more than one implementation of what is defined in their specification. When you use these components you have to identify them with names like dlib::component::kernel_1a. Often these components will have just a debugging and non-

debugging implementation. However, many components provide a large number of alternate implementations.

**Creating Objects**

To create many of the objects in this library you need to choose which kernel implementation you would like and if you want the checking version or any extensions.

To make this easy there are header files which define typedefs of all this stuff. For example, to create a queue of ints using queue kernel implementation 1 you would type **dlib::queue<int>::kernel_1a my_queue;.** Or to get the debugging/checking version you would type **dlib::queue<int>::kernel_1a_c my_queue;.** There can be a lot of different typedefs for each component.

None of the above applies to the single-implementation components, that is, anything that doesn't have an "implementations" section in its documentation. These tools are designed to have only one implementation and thus do not follow the above naming convention. For the purposes of object creation the API components also appear to be single-implementation. That is, there is no need to specify which implementation you want since it is automatically determined by which platform you compile under. Note also that there are no explicit checking versions of these components. However, there are DLIB_ASSERT statements that perform checking and you can enable them by #defining DEBUG or ENABLE_ASSERTS.

**Assumptions**

There are some restrictions on the behavior of certain objects or functions. Rather than replicating these restrictions all over the place in my documentation they are listed here.

- **global swap():** It is assumed that this operator does not throw. Undefined behavior results if it does. Note that std::swap() for all intrinsics and std::string does not throw.

- **operator<():** It is assumed that this operator (or std::less or any similar functor supplied by you to the library) does not throw. Undefined behavior results if it does.

- **dlib::general_hash:** It is assumed that general_hash does not throw. Undefined behavior results if it does. This is actually noted in the general hash spec file but I'm listing it here also for good measure.

### 3.2.5 gTTs

gTTS (Google Text-to-Speech) is a free of cost Python library and CLI tool to interface with Google Translate's text-to-speech API. This tool stores spoken mp3 data to a file, a file-like object (bytestring) for further audio manipulation, or stdout. It contains features such as flexible pre-processing and tokenizing, as well as automatic retrieval of supported languages [6].

**Preprocessor**

A Preprocessor is a function that takes text and returns text. Its goal is to switch text (for example correcting pronounciation), and/or to organize text for perfect tokenization (eg. spacing after certain tokens).

You can pass a list of any function to **gtts.tts.gTTS's pre_processor_funcs** attribute to act as pre-processor (as long as it takes a string and returns a string) [6].

By default, **gtts.tts.gTTS** takes a list of the following pre-processors, applied in order:

```
[
    pre_processors.tone_marks,
    pre_processors.end_of_line,
    pre_processors.abbreviations,
    pre_processors.word_sub
]
```

**Tokenizer**

A Tokenizer is a function that takes text and returns it split into a list of *tokens* (strings). In the gTTS context, its goal is to chop the text into smaller segments that don't exceed the utmost character size allowed for every TTS API request, while making the speech sound natural and continuous. It does so by splitting text where speech would naturally pause (for example on ".") while handling where it shouldn't (for example on "10.5" or "U.S.A."). Such rules are called **tokenizer cases**, which it takes a list of.

You can pass any function to **gtts.tts.gTTS's tokenizer_func** attribute to act as tokenizer (as long as it takes a string and returns a list of strings).

By default, gTTS takes the **gtts.tokenizer.core.Tokenizer's gtts.tokenizer.core.Tokenizer.run()**, initialized with default tokenizer cases:

```
Tokenizer([
tokenizer_cases.tone_marks,
tokenizer_cases.period_comma,
tokenizer_cases.other_punctuation
]).run
```

**Minimizing**

The Google Translate text-to-speech API accepts a maximum of **100 characters**. If after tokenization any of the tokens is larger than 100 characters, it will be split in two:

- On the last space character that's closest to, but before the 100th character;
- Between the 100th and 101st characters if there's no space.

### 3.2.6 Selenium

Selenium is a container that houses various tools and packages that enable automation of a web browser.

It provides packages to run virtually simulated user interaction with web clients, a distribution server for scaling browser allocation, and the infrastructure for implementations of the W3C WebDriver specification that allows you to write interchangeable code for all major web clients.

Selenium's primary feature is the WebDriver, an interface to write instruction sets that could be enabled to be run interchangeably in many browsers [7].

**WebDriver**

WebDriver uses either local or remote machines to drive a browser natively as any individual user or users would; marks a breakthrough in terms of browser automation.

Selenium WebDriver refers to both the language bindings and the implementations of the individual browser controlling code. This is commonly referred to as just WebDriver.

Selenium WebDriver is a W3C Recommendation [7].

- WebDriver is designed as a simple and more concise programming interface.
- WebDriver is a compact object-oriented API.
- It drives the browser effectively.

# Chapter 4

# SYSTEM ARCHITECTURE

## 4.1 System Structure

The proposed approach divides the whole system into three modules. The first module is designed to track the facial features of the driver. The video is converted into individual frames using OpenCV that is later processed to detect the face. In the second module, the feature extraction takes place. The eyes are traced and a facial landmark is generated for the eyes. In the third module, the blinking rate, the eye closure distance are analyzed to help figure out if the driver is drowsy or not. A voice bot is activated if we go to the third stage that will help alleviate the drivers drowsiness symptoms.

The following Figure 4.1 shows the overall system structure.

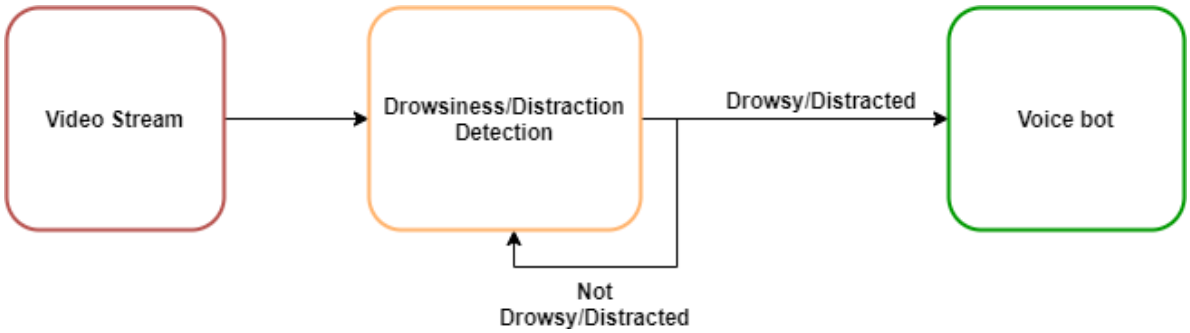



**Figure 4.1**: Drowsiness and distraction detection system structure

## 4.2 Drowsiness Detection System Design

The Drowsiness detection system consists of three modules, them being the face tracking module, the feature extraction module and the evaluation module. As the name suggests, the face tracking module tracks the human face while the video is going on live. The feature extraction module considers only the important features in the face that is, the eyes. In the third module, the eye closure distances are measured for both the eyes to tell if a person is drowsy or not.

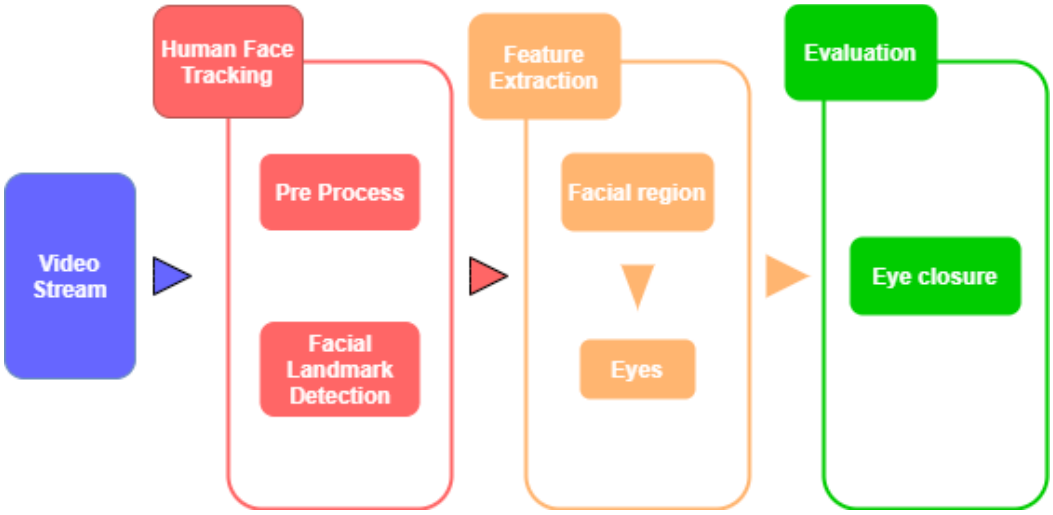



**Figure 4.2:** Drowsiness Detection System Design

Each of these three modules consist of sub-modules as well. A brief description of these modules is mentioned below.

### 4.2.1 Pre-Process

During the detection process, the quality of images is affected and features of the human face become unclear if the illumination intensity within the cab is changed during driving. This usually occurs in case of overcast skylight, rain, and at night. For detection accuracy, we use the illumination enhancement method to pre-process images before tracking the driver's face. Using OpenCV, the video gets converted into frames which are images that can be scanned for feature extraction.

### 4.2.2 Facial Landmark Detection

Face landmark detection is the process of finding points of interest in an image of a human face. The facemark detector can work with any image. Landmark detection starts with face detection, finding faces in the image and their extents (bounding boxes). Two factors are considered essential while calculating the facemark for a human face, pyramid scale factor and number of neighbours. The pyramid scale factor is used to create a pyramid of images within which the detector will try to find faces. Number of neighbors is the amount of facemarks around a particular point(facemark). Having obtained the facial landmarks, we can attempt to find the direction of the face. The 2D face landmark points essentially conform to the shape of the head. So, given a 3D model of a generic human head, we can find approximate corresponding 3D points for a number of facial landmarks. The opacity of the landmarks has been turned to zero to not let show off any face marks other than the eyes [21].

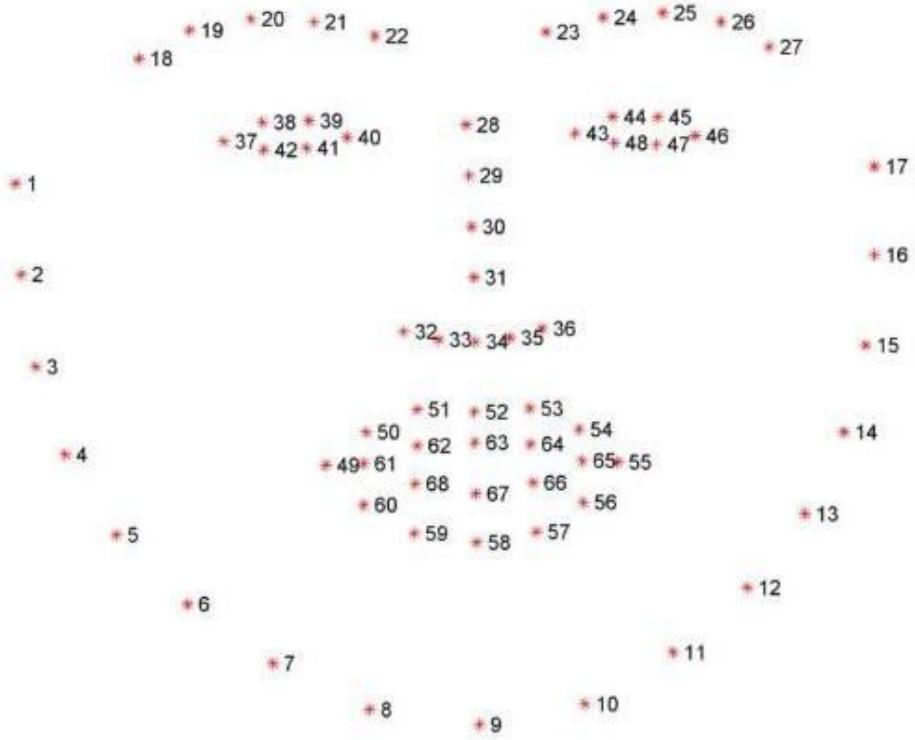



**Figure 4.3**: Facial Landmarks

### 4.2.3 Facial Region : Eyes

The facial landmark detection accurately helps you mark the eyes accurately. First, we offer the solution for locating the eyes' regions. one eye has six key points. However, these points are near the eyeball. By using these points to detect the region of an eye, the region will not include the upper and lower eyelids from the analysis, thereby influencing the result of the subsequent evaluation. Therefore, we use the key points of the eyebrow and nose to find the scope of the eye and eye socket. The equation is as follows [21]:

$$\begin{cases} lex = \dfrac{x_i + x_j}{2} \\ ley = y_m + \dfrac{y_n - y_m}{4} \end{cases}$$

In the above equation, $x_i$ and $x_j$ represent the $X$ coordinate of the $i_{th}$ and $j_{th}$ key points, respectively. $y_n$ and $y_m$ represent the $Y$ coordinate of the $n_{th}$ and $m_{th}$ key points, respectively. lex and ley denote the vertices' coordinates of the rectangular region of the eye. After we obtain the coordinate of the upper left $A$ and lower right vertices of the region $D$, we determine the eye socket region on the driver's face based on rectangular symmetry.

The eyes themselves are not classified as of yet. The feature classification is carried out by the YOLO algorithm. YOLO stands for You Only Look Once. This algorithm is a Single Shot Detection (SSD) Algorithm. YOLO is trained using the COCO dataset which contains 80 labels. The algorithm is shown to have a high accuracy in feature classification and is relatively fast compared to other similar neural networks. After the eyes are classified as the primary feature, only the landmark points for the eyes are considered as necessary and the rest can be ignored.

### 4.2.4 Eye Closure

For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is computed.

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

where $p_1, \ldots, p_6$ are the 2D landmark locations. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye.

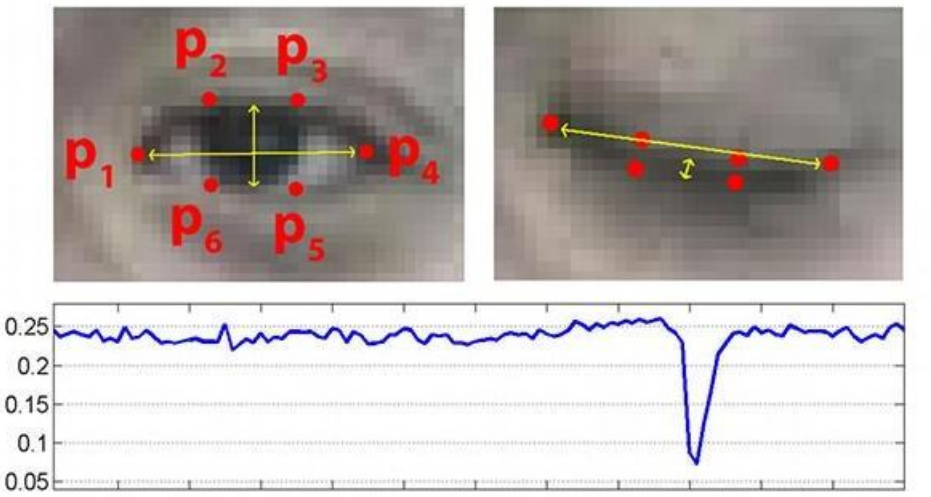



**Figure 4.4**: Open and closed eyes with landmarks and EAR

It is partially person and head pose insensitive. Aspect ratio of the open eye has a small variance among individuals and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged. The EAR is then compared to a specified threshold. If the EAR value decreases beyond the threshold then an alarm is activated to awaken the drowsy driver.

The alarm in this case is an Alerting system enabled with speech recognition and voice to be able to speak to the driver to help alleviate his drowsiness symptoms. The alerting system will repeatedly sound the alarm until the EAR is back above the threshold. The alerting system is

capable of understanding English and is programmed to carry out some other minor tasks aside from alerting the driver

## 4.3 Distraction Detection System Design

The Distraction Detection System consists of 3 modules.

1. The Frame Capture Module
2. Object Detection Module
3. Alerting System Module

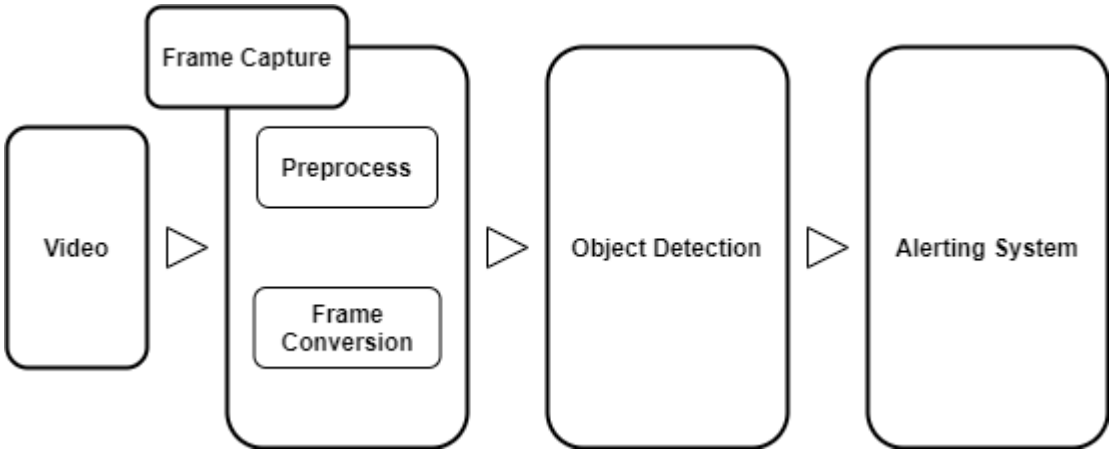



**Figure 4.5**: Distraction Detection System Design

The Frame Capture Module pre-processes the video and captures individual frames of the video.

The object detection module consists of a deep neural network which detects objects. If an object is recognized to being used by the driver then a driver Alerting System is initiated.

A more detailed description of the three modules is given below.

### 4.3.1 Frame Capture Module

The Frame Capture Module is Responsible for capturing frames from a live video. The frame capture module has two sub-modules, Pre-Process Sub-Module and a Frame Conversion Sub-Module.

**Pre-Process**

During the detection process, the quality of images is affected and features of the human face become unclear if the illumination intensity within the cab is changed during driving. This

usually occurs in case of overcast skylight, rain, and at night. For detection accuracy, we use the illumination enhancement method to pre-process images before tracking the driver's face.

**Frame Conversion**

In Frame Conversion module, the video camera which provides frames which are of size 640 x 480 or 480p resolution are down-sampled to 320p resolution frame or a 320 x 320 blob. This is done to reduce the work required to be done from the object detection module later on. This down-sampled frame is then used by the object detection module.

### 4.3.2 Object Detection Module

The object detection module is responsible for detecting if any foreign objects are visible in close vicinity of the driver. The object detection module solely works on the YOLO (You Only Look Once) Algorithm. This algorithm is a Single Shot Detection (SSD) Algorithm. YOLO is trained using the COCO dataset which contains 80 labels. The algorithm is shown to have a high accuracy in feature classification and is relatively fast compared to other similar neural networks.

The project utilized the effectiveness and quick nature of this algorithm to detect and identify the objects a driver might be using that could cause distraction. An object can constitute any food products, water bottle, satchel bags, hand bags, glasses/cups and mobile phones.

Once an item is identified that could lead to a driver losing his attention towards driving for a limited span of time, a driver alerting system is initiated.

### 4.3.3 Alerting System

The alerting system is responsible for alerting the driver about him being distracted. The system is voice enabled which means that it can talk. The system is designed to speak the following sentence when enabled.

Please don’t use [object] while driving.

The object can be any of the predetermined objects such as a mobile phone, food product, hand bag etc.

## 4.4 Alerting System

The alerting system is responsible for alerting the driver in case he gets drowsy or distracted. The Alerting system is in fact a voice bot that can talk. It is programmed to perform a limited amount of functions. The alerting system comprises two modules.

1. Text to Speech Module
2. Speech to Text Module

**Text to Speech (T2S)**

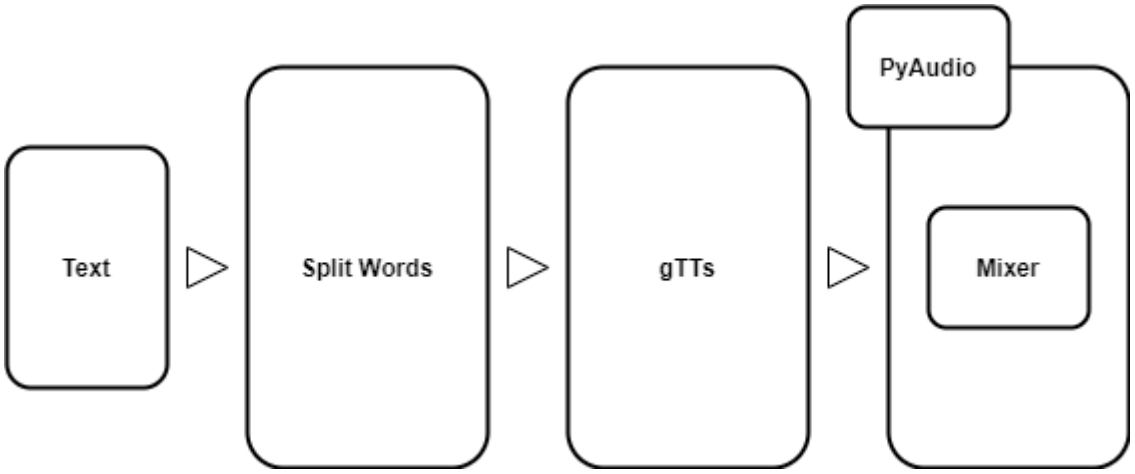



**Figure 4.2**: Text to Speech (T2S) System Structure

The T2S submodule receives **text** as the input. The **Split Words submodule** contains the function splitlines(), which is responsible for splitting the words at line boundaries. Then, **gTTS** will handle to convert all these texts to speech. The speech is saved as an audio file. The **PyAudio submodule** contains a function called as **mixer**. The mixer function is responsible for loading the audio file and then playing this audio file.

**Speech to Text (S2T)**

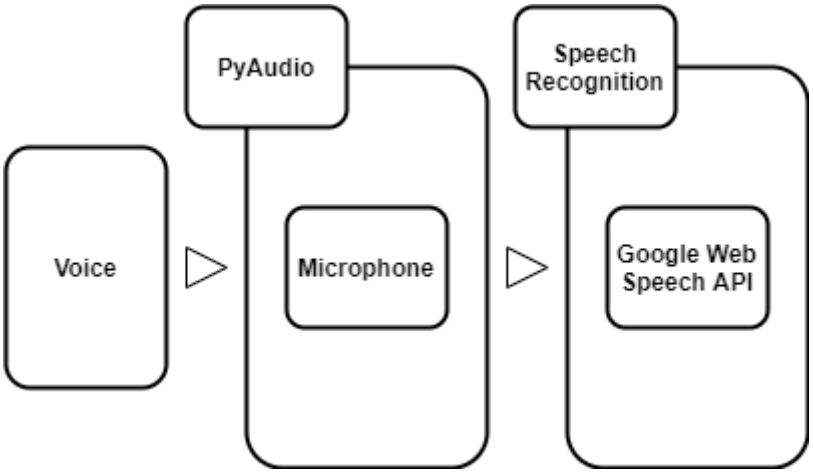



**Figure 4.3**: Speech to Text (S2T) System Structure

The S2T submodule takes human **voice** as the raw input. A microphone is used to collect the audio signals. The **PyAudio library** contains functions that can listen and collect the voice samples. This function is responsible for collecting the audio data using the microphone. The speech to text conversion is done by the **Speech Recognition library**. This library contains a hardcoded **Google Web Speech API** which means that you need not sign up for using this google API. This API converts the given audio input into a string output.

The S2T submodule is further responsible for understanding some special commands given by the driver while driving. The driver can give commands like 'play music' which have to be correctly recognized by the system and in response has to act on the command.

# Chapter 5

# IMPLEMENTATION

## 5.1 Introduction

Implementation is the realization of the technical specification or algorithm as a program. The implementation phase of any project is the most crucial phase as it is here that the project finally takes shape. This phase is implemented keeping the end user in mind. It is the implementation of the project that yields the final solution, which solves the problem in hand. The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document, and the development of the system in a suitable programming language necessary to achieve the final product. Often a product is ruined due to the incorrect choice of the programming language or unsuitable method of programming. It is hence better for the coding phase to be directly linked to the design phase.

The implementation stage consists of proper planning, detailed study of the existing system and its constraints and designing of any alternative methods and their evaluation.

The implementation of the proposed system involves a sequence of simple steps as given below:

First, you take individual frames from OpenCV and then pre process them for the facial marking algorithm.

1. The face mark algorithm generates facial landmarks of a human face.
2. You extract the eyes as the primary feature using the OpenCV algorithms.
3. A ratio of both the eyes is taken, individually. This ratio denotes the amount of the closing of the eyelids.
4. This ratio is compared to a specified threshold. If it ever goes below the threshold then an Alerting System is activated.
5. The Alerting System engages with the driver to keep him awake and to help him revert his attention back to driving.

## 5.2 Programming Language Selection

As described earlier, any mistakes in selection the programming language may lead to the failure of the entire system. Hence, the programming language for any code must be chosen

with proper knowledge about the design of the proposed system. The programming language chosen is Python.

### 5.2.1 Python

Python is an object-oriented scripting language which is very easy to learn. Hence it is also commonly called as a beginner's language. It is an interpreted and interactive language. This language was created by Guido van Rossum and its implementation began in 1989 at Centrum Wiskunde & Informatica (CWI), Netherland. Python is basically developed under and open source license which has been approved by OSI. Hence, this language is free for use and distribution (even for commercial use).

The main reasons for using Python in our proposed system are:

1. Easy to learn and use.
2. Image processing is easier as Python includes libraries that support image processing from the basic to advanced level. Many of the functions for feature extraction, context detection, etc. are implicitly available in the libraries.
3. Python provides interfaces to almost all commercial databases.
4. The Python language is portable and scalable. [9]

## 5.3 Data Flow Diagram

A flowchart is a diagram which depicts the algorithm and/or process as a sequence. The various parts of the algorithm are represented in boxes, where there is a different kind of box for different kinds of statements. The order of the process is shown by arrows connecting the boxes in the appropriate direction. A data flow diagram depicts the flow of data in the system in an organized manner.

The following Figures 5.1 to 5.4 shows the dataflow diagram.[16] These flowcharts explains the working of the system in brief.

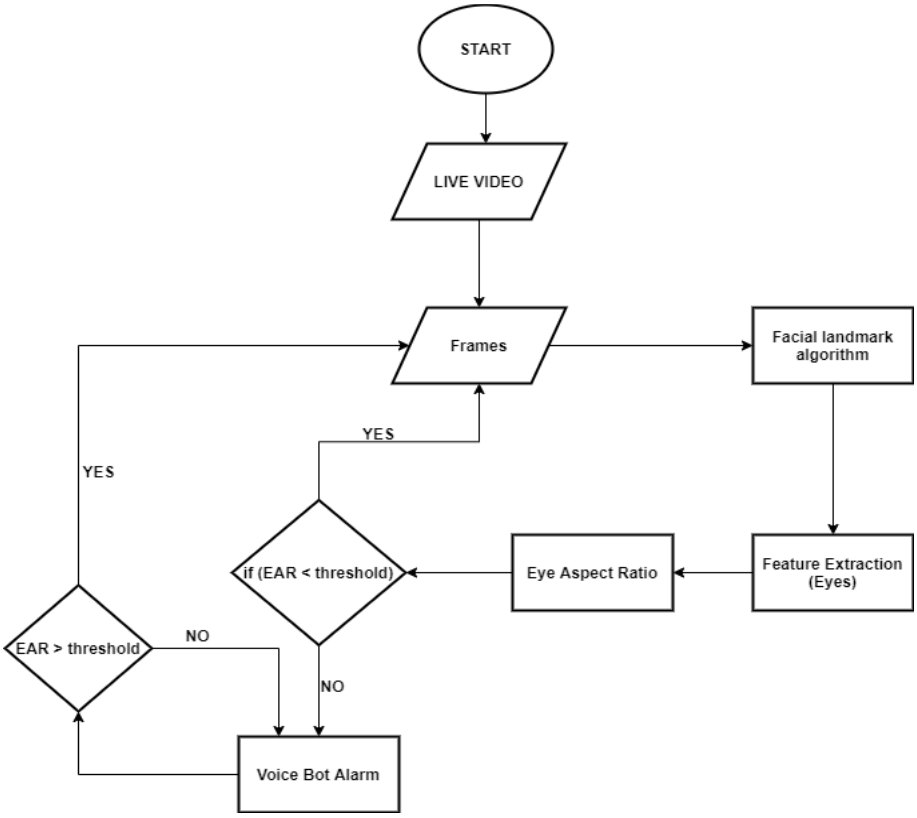



**Figure 5.1:** Drowsiness Detection System Data Flow Diagram

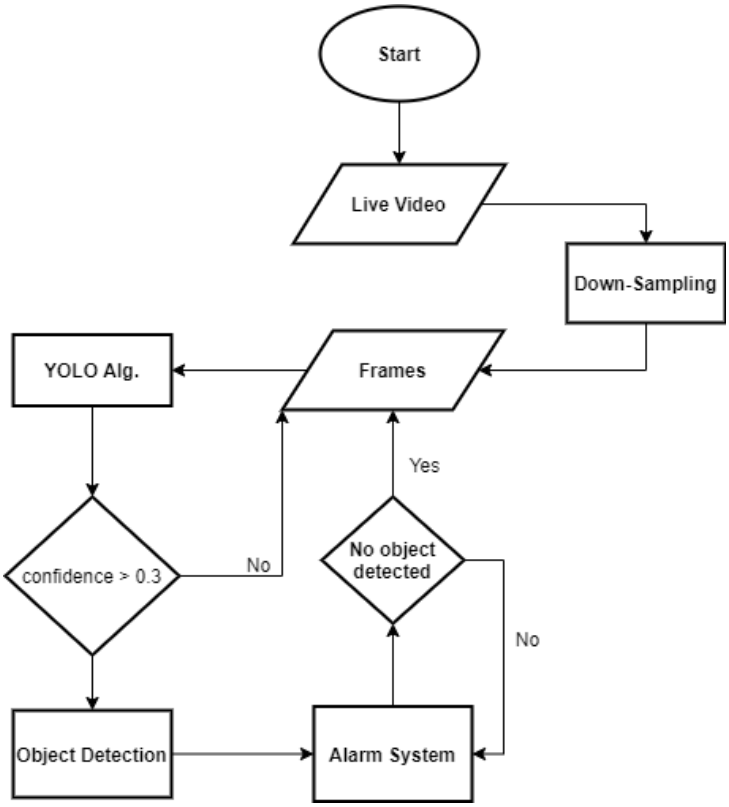



**Figure 5.2**: Distraction Detection System Data Flow Diagram

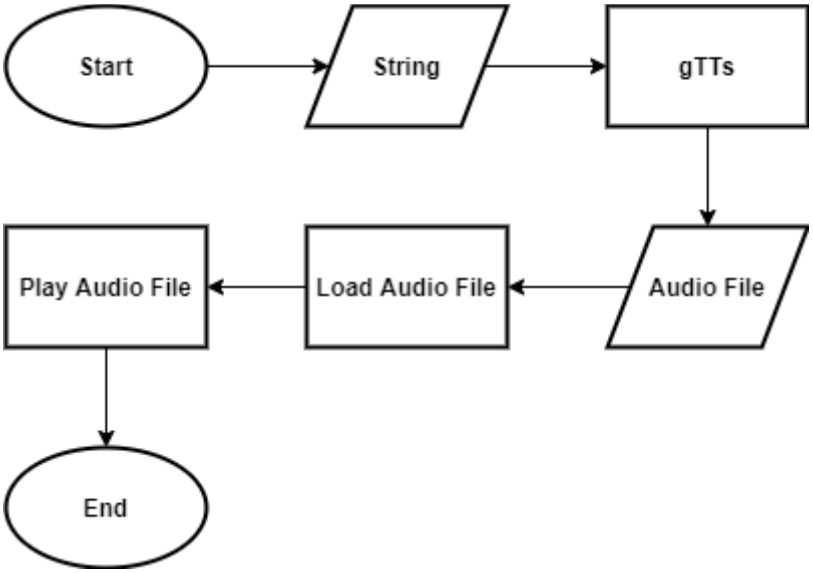



**Figure 5.3**: Alerting System (Text to Speech) Dataflow Diagram

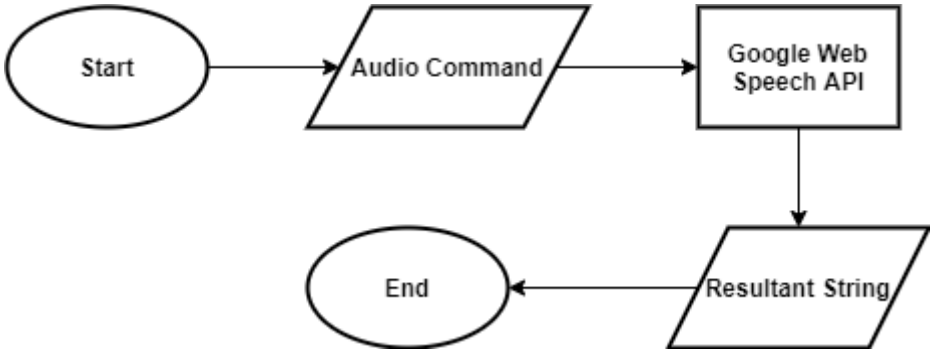



**Figure 5.4**: Alerting System (Speech to Text) Dataflow Diagram

Note that figures 5.3 and 5.4 are parts of the same system.

## 5.4 Activity Diagram

An activity diagram is similar to a flowchart but here the flow is represented from one activity to another activity. An activity is nothing but an operation of the system. This diagram describes the dynamic behaviour of the system. These are usually constructed using forward and reverse engineering techniques. The following Figure 5.2 describes the activity diagram of our proposed system. [17] .

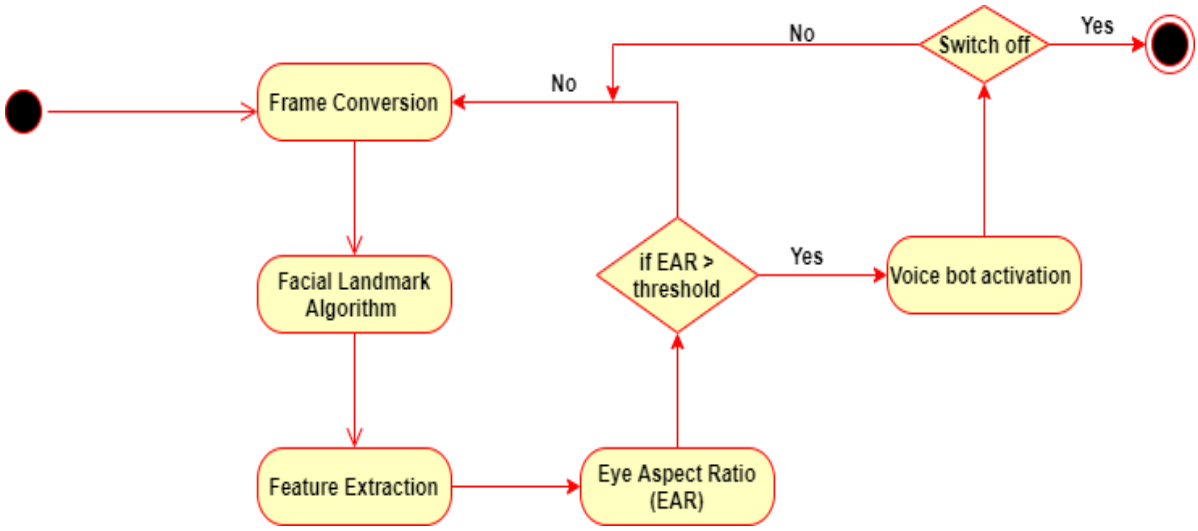



**Figure 5.5:** Drowsiness Detection System Activity diagram

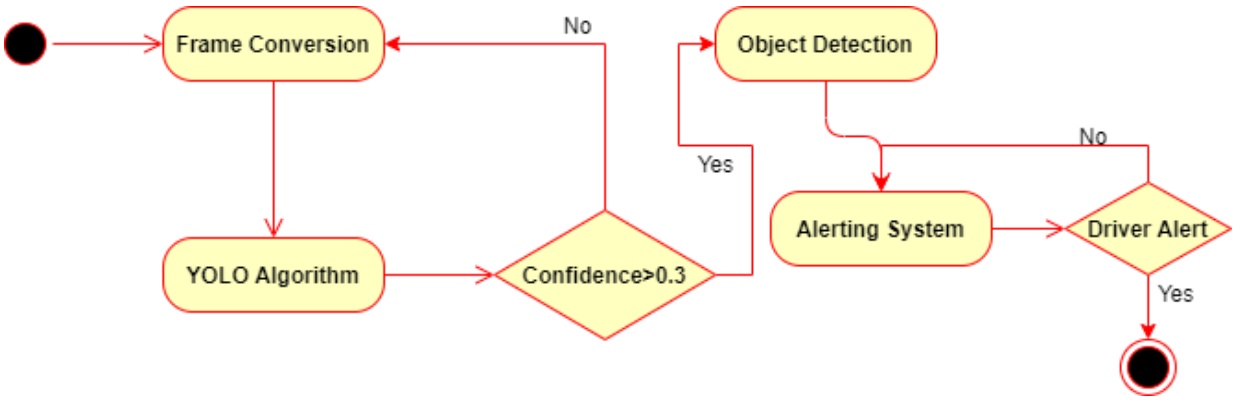



**Figure 5.6**: Distraction Detection System Activity Diagram

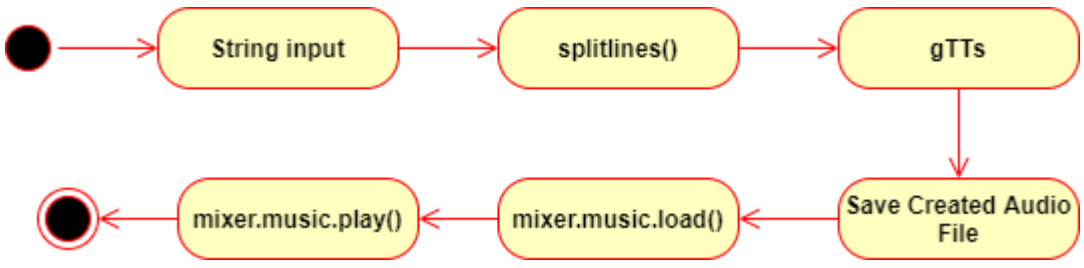



**Figure 5.7**: Text to Speech sub-system Activity Diagram

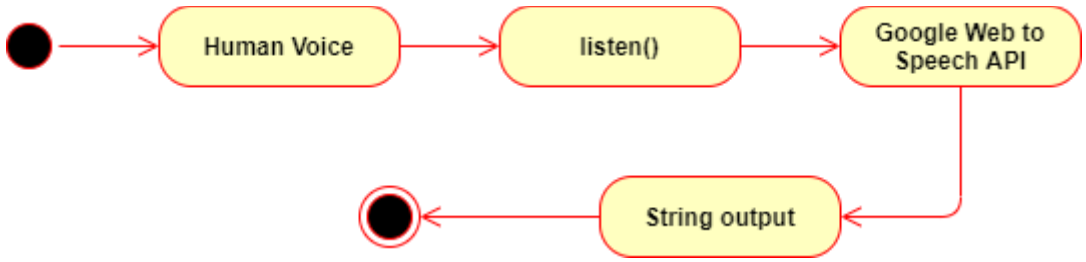



**Figure 5.8**: Speech to Text sub-module Activity Diagram

## 5.5 Use Case Diagram

A use case diagram is a simple way of representing the interaction of a user with the system. It shows the complete relationship between the user and system with the use cases. It is also known as a behaviour diagram as it depicts the behaviour of the system with the external actors (users).The following Figure 5.3 shows the use case diagram for our system. Here the user is considered as the external actor that interact with the system in order to retrieve the answer.

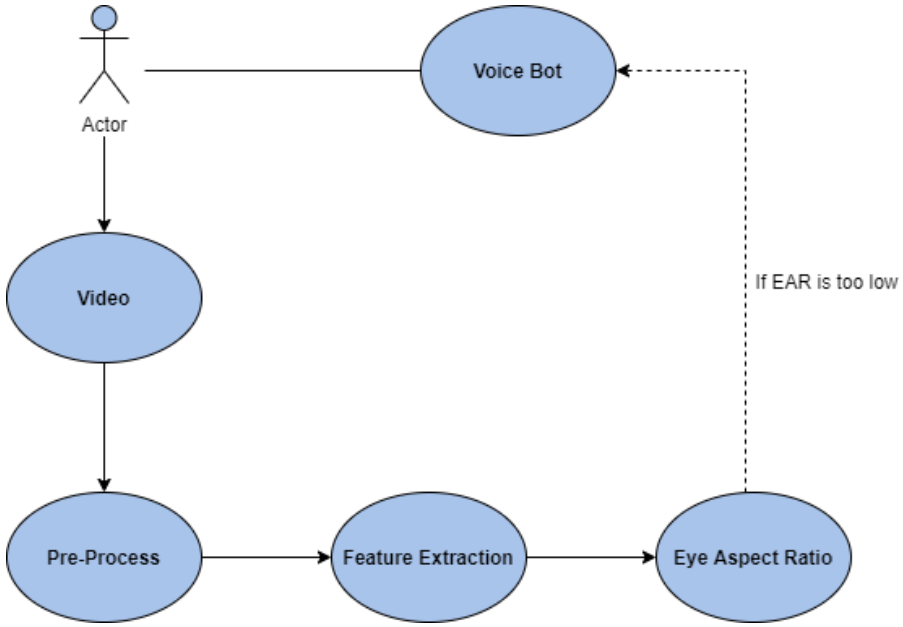



**Figure 5.9:** Drowsiness Detection System Use case Diagram

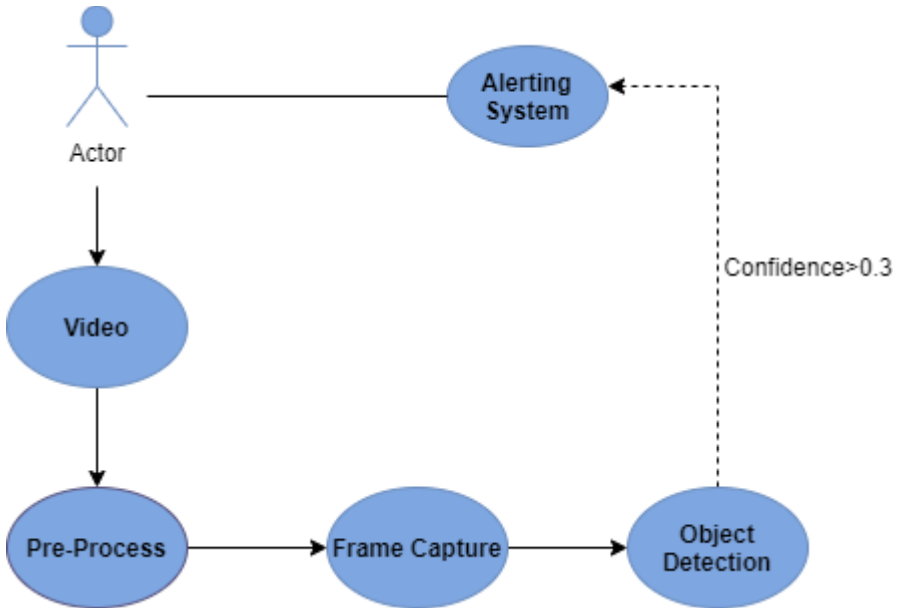



**Figure 5.10**: Distraction Detection System Use Case Diagram

## 5.6 Sequence Diagram

A sequence diagram depicts the active processes that live simultaneously as vertical lines. The horizontal arrows show the messages that are being transferred from one live process or object to another. These messages are given in the order that they are exchanged from the top to the bottom of the sequence diagram. These sequence diagrams are also called event diagrams or event scenarios as they show the various events that occur in the system in the proper order. The Figure 5.4 below shows the sequence diagram for our question answering system.

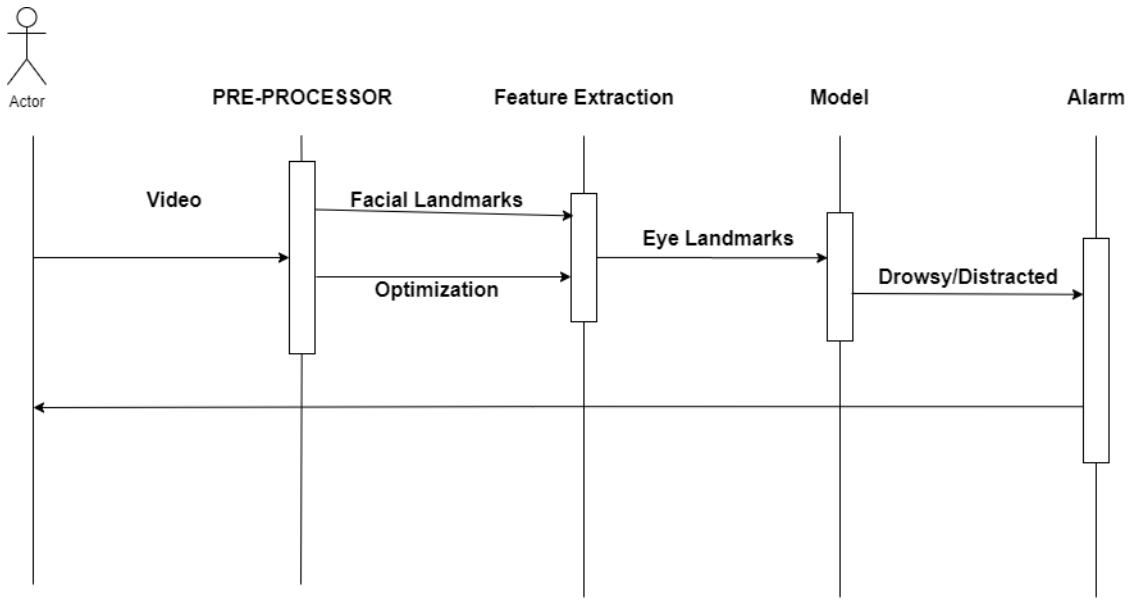



**Figure 5.11:** Drowsiness Detection System Sequence diagram

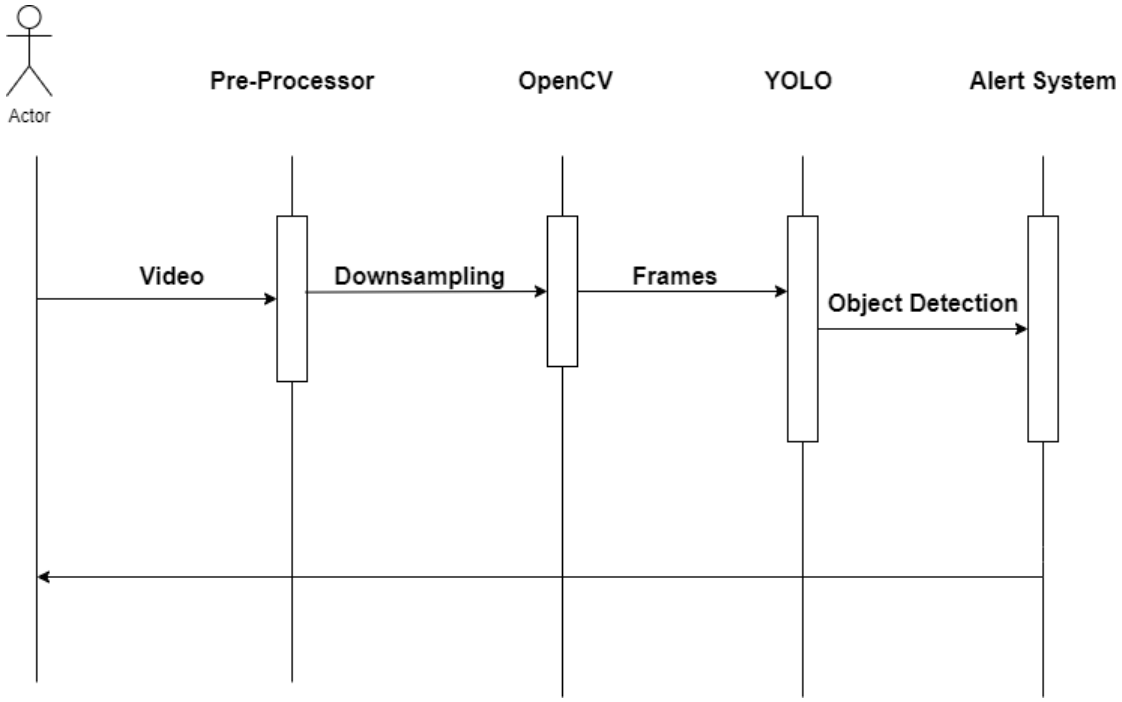



**Figure 5.12**: Distraction Detection System Sequence Diagram

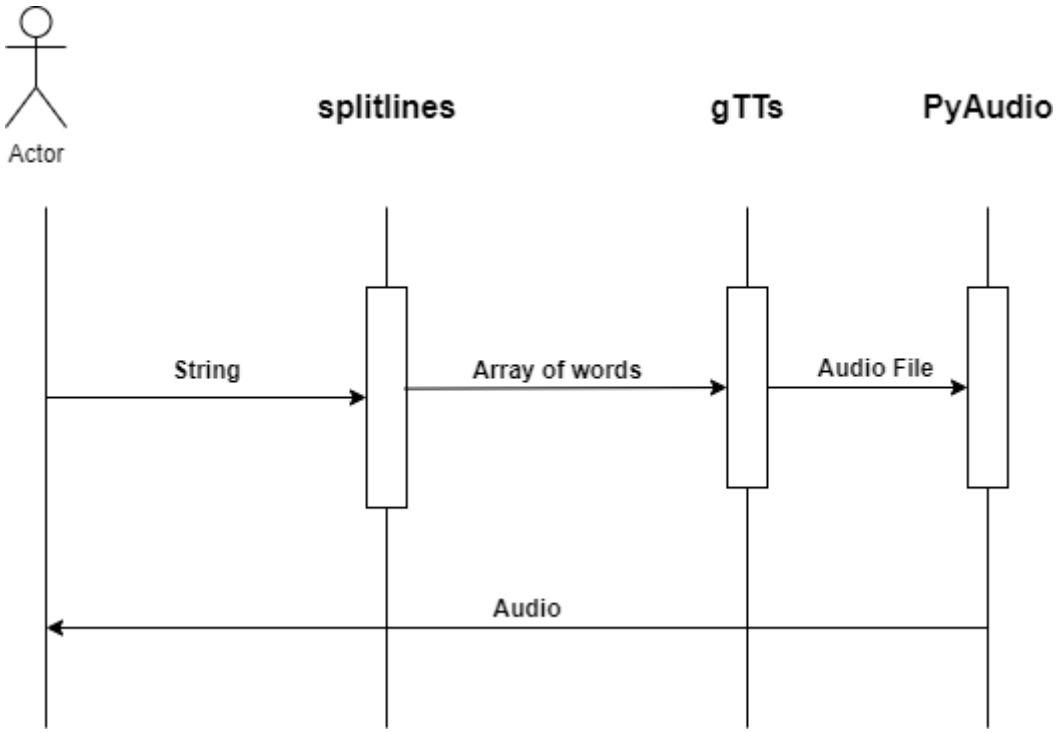



**Figure 5.13**: Text to Speech system Sequence Diagram

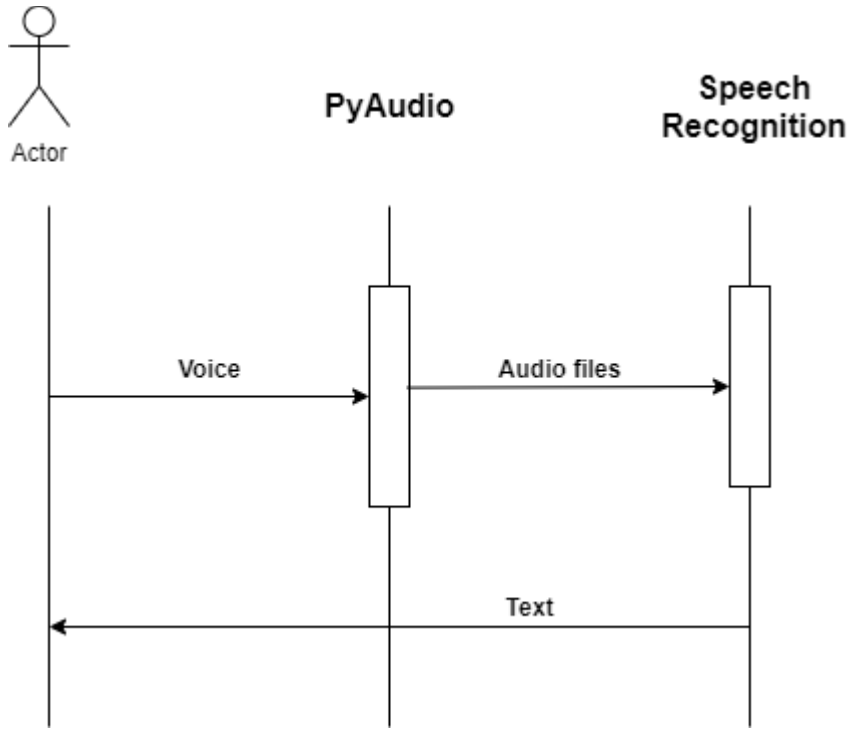



**Figure 5.14**: Speech to Text System Sequence Diagram

# Chapter 6

# SYSTEM STUDY

## 6.1 Feasibility Study

In this phase, the feasibility of the project is evaluated and business scheme is put forth with a general plan for the project and cost estimations. The feasibility study of the proposed system is to be performed during system analysis. This is to ensure that the proposed system is not a burden to the company. It evaluates the project's potential for success.

The feasibility study summarizes and analyzes several methods to achieve success. It helps in tapering the scope of the project and to recognize the best business scenario. The intention of this study is to identify the probability of one or more solutions satisfying the specified business requirements. For feasibility analysis, understanding of the major requirements for the system is necessary. [18]

Three key aspects involved in the feasibility analysis are:

1. Economic Feasibility
2. Technical Feasibility
3. Social Feasibility

### 6.1.1 Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization, that is, the amount of fund that the company can pour into the research and development of the system. It also serves as an independent project assessment and improves system reliability by helping decision makers determine the positive economic benefits to the organization that the proposed system will provide.

The economic feasibility study projects how much start-up capital is needed, sources of capital, returns on investment, and other financial considerations. It looks at how much cash is needed, where it will come from, and how it will be spent. Thus the developed system should be within the budget and this can be achieved because most of the technologies used are freely available. Only the customized products had to be purchased. Thus, the expenditures must be justified. [18]

### 6.1.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is to say the technical necessities of the system. It also concentrates on acquiring and understanding the modern technical resources and their applicability to fulfill the needs of the proposed system. It is an evaluation of the hardware and software potential in meeting the needs of the proposed system.

Any system developed must not have a high requirement for the available technical resources which will lead to high demands being placed on the client. The developed system must have a humble requirement, as only negligible or null changes are required for implementing the system. The technical feasibility study should basically support the financial statistics of an organization. It calculates the aspects of how you intend to furnish a product or service to customers. [18]

### 6.1.3 Social Feasibility

This study is carried out to check the extent of acceptance of the system by the user. This includes the process of training the user to use the system effectively. The user must not feel vulnerable by the system, instead must accept it as a necessity. Social feasibility study includes environmental impacts on the project location and in associated areas to evaluate the effects on environmental resources due to alterations or pollutants.

Social feasibility describes the effect on users from the introduction of the new system taking into account whether there will be a need for retraining the workforce. The level of tolerance by the users solely depends on the means that are employed to educate them about the system. Their level of confidence must be increased so that they are able to make some productive criticism. Social feasibility also describes how you propose to ensure user co-operation before changes are introduced. [18]

# Chapter 7

# SYSTEM TESTING

The purpose of testing is to identify all errors. Testing is the process of trying to discover every likely fault or weakness in the system. It offers a way to inspect the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of implementing the software with the objective of ensuring that the software system meets its requirements and user expectations and does not fail in any improper manner. There are various types of test each addressing a specific testing requirement. [18]

## 7.1 Types of testing

The types of testing are as given in the following section.

### 7.1.1 Unit testing

Unit testing comprises of the design of test cases that validate the internal logic function and also its inputs and outputs. All decision branches and internal code flow should be verified. It is the testing of individual software units of the application, done after the completion of a distinct unit just before integration. This is a structural testing, that relies on the knowledge of its construction and is invasive. Unit tests accomplish basic tests at component level and also test a specific application, business process and/or system configuration. Unit tests ensure that each unique path of a process performs in harmony with the standard specifications and clearly defines the inputs and expected results. [14]

### 7.1.2 Integration testing

Software integration testing is the process of testing two or more integrated software components on a single platform to verify failures caused during interfacing. [14]

Integration tests are designed to test combined software components to determine whether they run as a single program. Integration tests exhibit that although the components were exclusively effective, as shown by the fruitful unit testing, the combination of these are correct and consistent. Integration testing is explicitly aimed at revealing the problems that arise due to assembly of components. [25]

### 7.1.3 Functional testing

Functional tests offer organized demonstrations that indicate whether the functions tested are available as stated by the business and technical requirements, user manuals and system documentation. Functional testing is focused on the following items as shown in Table 7.1.

Table 7.1: Functional Testing items and their corresponding functions

| **Test Items** | **Functions** |
|---|---|
| Input | identifies classes of inputs that must be accepted |
| Functions | identifies functions to be applied |
| Output | identifies classes of outputs that must be executed |
| Systems/Procedures | identifies interfacing systems or procedures that must be invoked |

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, business process flows, data fields, predefined processes, and successive processes must be considered for systematic coverage pertaining to testing. Before functional testing is complete, additional tests are recognized and the effective value of current tests is obtained. [20]

### 7.1.4 System Testing

System testing guarantees that the complete integrated software system meets the requirements. It tests a configuration to ensure pre-determine and expected results. One such example of system testing is the configuration oriented system integration test. System testing works on the basis of process descriptions and flows, highlighting pre-driven process links and integration points. System testing falls into the category of black-box testing hence does not require any knowledge of the inner logic or design of the code. System testing tests not only the design, but also the behavior of the system. It even tests the believed expectations of the customer. It is also expected to test up to and beyond the constraints specified in the software or hardware requirements specification. [16]

### 7.1.5 Black Box Testing

Black Box testing also known as Behavioral testing, is testing the software without prior knowledge of the inner workings, structure or language of the module being tested. Black box test must be written from a completed source document, like specification or requirements document. It is a testing in which the software under consideration is treated as a black box. The test provides inputs and responds to outputs without bearing in mind how the software works. [16]

### 7.1.6 White Box Testing

White Box Testing also known as structural testing, is a testing in which the software tester has information of the inner workings, language and structure of the software, or at least its purpose. White-box testing can be employed at the unit, integration and system levels of the software testing process. It helps in finding errors but has the ability to miss unimplemented sections of the specification or missing requirements. It is used to test regions that cannot be reached from a black box level. [16]

### 7.1.7 Acceptance Testing

User Acceptance Testing is a level of software testing where a system is tested for acceptability. This is a critical phase of any project and requires significant participation by the end user. The main purpose of this test is to assess whether the system is in accordance with business requirements and evaluate whether or not it is acceptable for delivery. It also guarantees that the product meets the functional requirements. The acceptance test needs to be performed several times, as all of the test cases may not be executed within a single test iteration. [16]

## 7.2 Test cases

| Test case ID | Test Case Description/Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|
| TC_ID_01 | Start the command prompt | The command prompt should respond and start running | Command prompt is running successfully | Pass |
| TC_ID_02 | Enter the address of the directory in | The directory of the command | Successfully shift directories | Pass |

| | | | | |
|---|---|---|---|---|
| | which the system is present. | prompt should shift to the entered address | | |
| TC_ID_03 | Enter the command with all the necessary parameters | Accepts the command and starts running the system. | Successfully runs the system and provide output. | Pass |
| TC_ID_04 | Change the model on which the system has to run on in the command. | Accepts and run the system again. | Successfully runs the system and provide output. | Pass |
| TC_ID_05 | WebCam/DashCam working without errors | Live video is seen without any buffering. | Successfully displays a video without any problems | Pass |
| TC_ID_06 | The driver closes their eyes for an extended period of time | Drowsiness Detection System sounds an alarm | An alarm is sound by the alerting system | Pass |
| TC_ID_07 | Driver holds a bag in front of the camera | Distraction Detection System sounds an alarm | An alarm is sound by the alerting system | Pass |
| TC_ID_08 | Driver holds a water bottle in front of the camera | Distraction Detection System sounds an alarm | An alarm is sound by the alerting system | Pass |
| TC_ID_09 | Driver holds a glass in front of the camera | Distraction Detection System sounds an alarm | An alarm is sound by the alerting system | Pass |
| TC_ID_10 | Driver holds a mobile phone in front of the camera | Distraction Detection System sounds an alarm | An alarm is sound by the alerting system | Pass |
| TC_ID_11 | Enter the command to start chatbot with all the necessary parameters | Accepts the command and starts running the chatbot. | Successfully runs the system and provide output. | Pass |

| | | | | |
|---|---|---|---|---|
| TC_ID_12 | Enter a sentence for the chatbot/voicebot to respond to | Chatbot accepts the input and returns a valid response to user. | Chatbot successfully replies to the user. | Pass |
| TC_ID_13 | Speak to the chatbot | The chatbot returns the words spoken in a string format | Chatbot successfully displays users spoken words on the terminal | Pass |
| TC_ID_13 | Tell the chatbot to start playing music | The chatbot starts the music player | Music player is successfully activated | Pass |
| TC_ID_13 | Tell the chatbot to stop playing music | The chatbot stops the music player | Music player is successfully deactivated | Pass |

# Chapter 8

# RESULTS AND DISCUSSIONS

The analysis is done by implementing the three algorithms in Python in virtual Environment. Given below are some of the screenshots which depict the working of the models. Figure 8.1 is on the subject of detecting drowsiness. Notice how a red alert text pops on the video whenever drowsiness is detected.

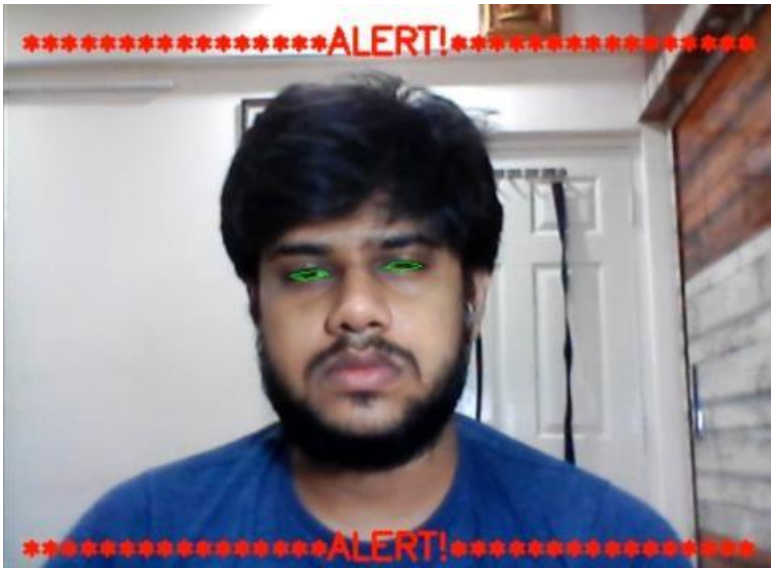



**Figure 8.1**: Drowsiness Detection

While the above figure was related to drowsiness specifically, the next images are specifically related to the driver being distracted from driving.


**Figure 8.2**: Distraction with a cup

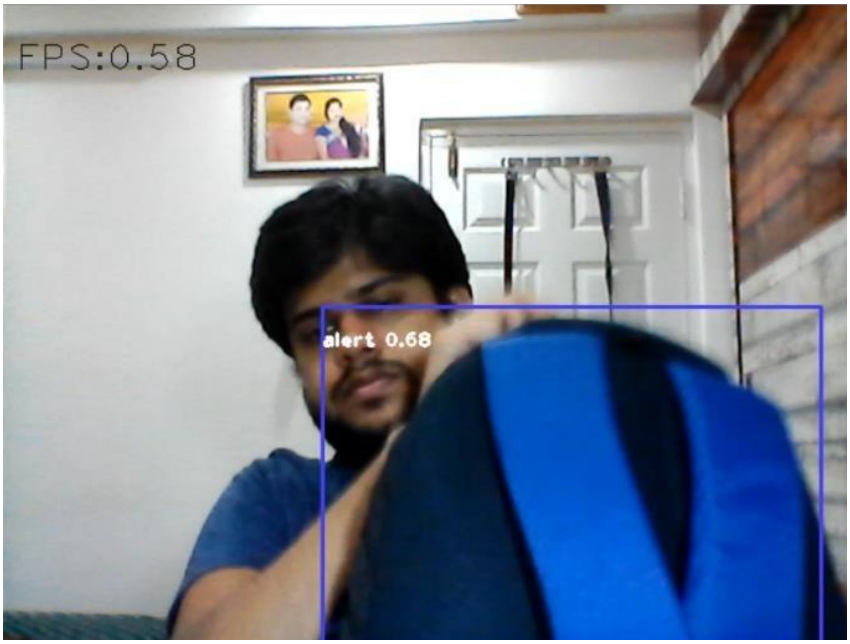



**Figure 8.3**: Distracted with a backpack


**Figure 8.4**: Distracted with a cell phone (texting)


**Figure 8.5**: Distracted with a cell phone (talking)

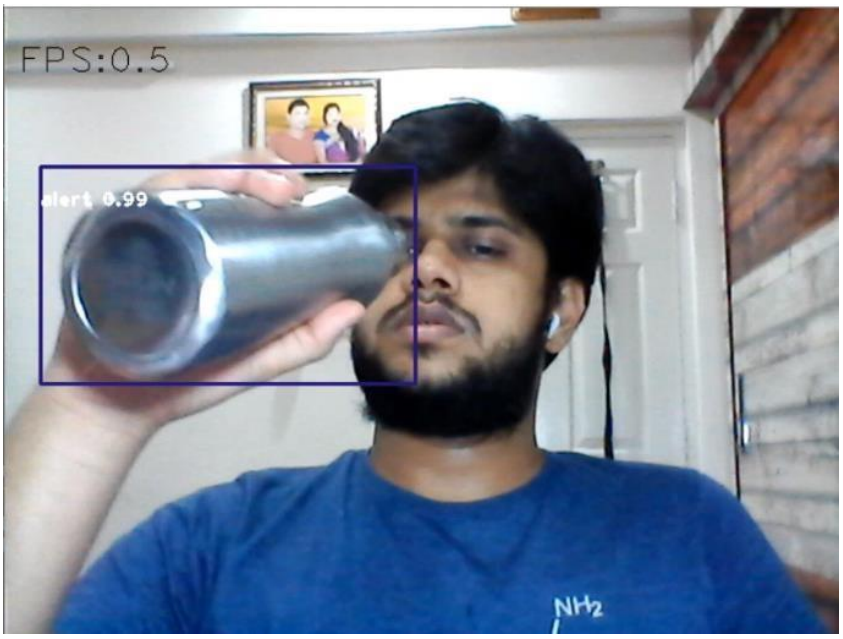



**Figure 8.6**: Distracted with a water bottle

Figure 8.2 to 8.6 depicted different scenarios where a driver is getting distracted and how the system is accurately detecting these objects. A message is given by the system when this happens which is a voice cue.

Figure 8.7 to 8.10 depict the working of the chatbot.

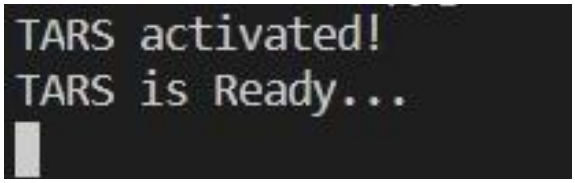



**Figure 8.7**: TARS the ChatBot

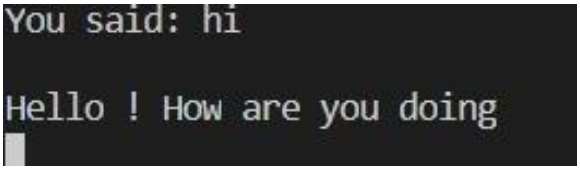



**Figure 8.8**: Chatting with a ChatBot

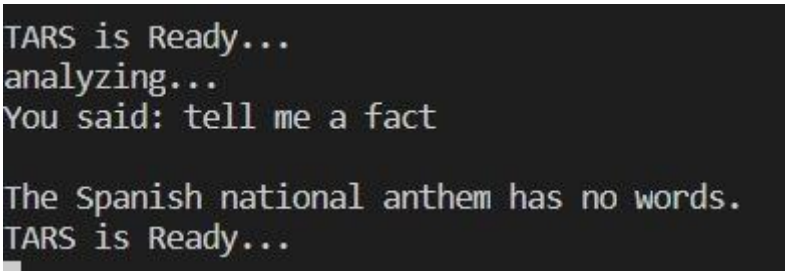



**Figure 8.9**: ChatBot telling unique facts

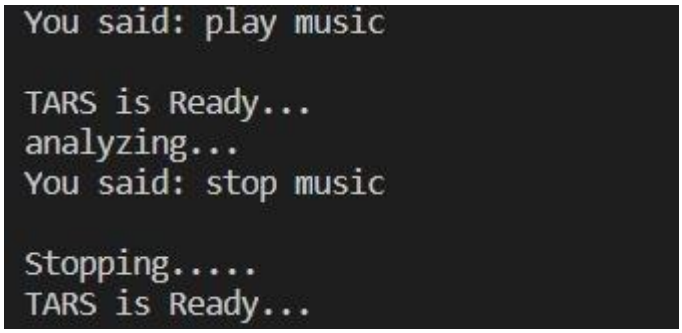



**Figure 8.10**: Chatbot playing music

The chatbot is programmed in such manner that when it hears the command ‘play music’, it starts the music player. It can be instructed with the command ‘stop music’ to stop the music from playing.

# Chapter 9

# CONCLUSION AND FUTURE ENHANCEMENTS

## 8.1 Conclusion

A real-time eye blink detection algorithm was presented. We quantitatively demonstrated that regression-based facial landmark detectors are precise enough to reliably estimate a level of eye openness. While they are robust to low image quality (low image resolution in a large extent) and in-the-wildA real-time eye blink detection algorithm was presented. We quantitatively demonstrated that regression-based facial landmark detectors are precise enough to reliably estimate a level of eye openness. While they are robust to low image quality (low image resolution in a large extent) and in-the-wildWe see a limitation that a fixed blink duration for all subjects was assumed, although everyone's blink lasts differently. The results could be improved by an adaptive approach. Another limitation is in the eye opening estimate. While EAR is estimated from a 2D image, it is fairly insensitive to a head orientation, but may lose discriminability for out of plane rotations. A solution might be to define the EAR in 3D. There are landmark detectors that estimate a 3D pose (position and orientation) of a 3D model of landmarks, e.g. [1, 3].

## 8.2 Future Scope

This project has tremendous potential to gain wide acceptance among people. Through this project we intend to incorporate our system into various industries thus preventing the loss of lives due to driving under drowsiness or getting distracted while driving. The future scope of our project is vast and can be used in extensive ways:

1. The project can be used as a case study.
2. The project can be explored further and newer technologies and hardware can be incorporated into the project provided, the funds are available.
3. The project can be installed into other kinds of vehicles to study the effectiveness of this system under abnormal conditions.
4. This project can be used as a feature in another product that integrates other features into itself.

# References

[1] W. Deng and R. Wu, "Real-Time Driver-Drowsiness Detection System Using Facial Features," in IEEE Access, vol. 7, pp. 118727-118738, 2019, doi: 10.1109/ACCESS.2019.2936663.

[2] OpenCV - https://www.learnopencv.com/

[3] SciPy- https://en.wikipedia.org/wiki/SciPy

[4] Numpy - https://www.numpy.org/

[5] Dlib - http://dlib.net/intro.html

[6] gTTs - https://gtts.readthedocs.io/en/latest/

[7] Selenium - https://www.selenium.dev/documentation/en/

[8] "J. Cech, V. Franc, and J. Matas. A 3D approach to facial landmarks: Detection, refinement, and tracking. In Proc. International Conference on Pattern Recognition, 2014.

[9] Python – https://www.python.org/

[10] ReadAI - https://github.com/ayoungprogrammer/readAI

[11] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding - https://arxiv.org/abs/1810.04805

[12] System Design in Software development - https://medium.com/the-andelaway/system-designin-software-development-f360ce6fcbb9

[13] Designing Use Cases for a Project - https://www.geeksforgeeks.org/designing-use-cases-for-aproject/

[14] Unit Testing - http://softwaretestingfundamentals.com/unit-testing/

[15] Integration Testing - http://softwaretestingfundamentals.com/integrationtesting/

[16] System Testing - http://softwaretestingfundamentals.com/system-testing/

[17] A Practitioner's guide to natural language Processing https://towardsdatascience.com/a-practitioners-guide-to-natural-languageprocessing-part-i-processing-understanding-text-9f4abfd13e72

[18] Feasibility study – https://www.sqa.org.uk/elearning/SDM02CD/page_11.htm

[19] Drowsiness Detection- https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/

[20] Soukupova and Cech's2016 paper- http://vision.fe.unilj.si/cvww2016/proceedings/papers/05.pdf

[21] Chatbot- https://github.com/bhushan-borole/simple-chatbot

[22] H. Singh, J. S. Bhatia and J. Kaur, "Eye tracking based driver fatigue monitoring and warning system," India International Conference on Power Electronics 2010 (IICPE2010), New Delhi, 2011, pp. 1-6, doi: 10.1109/IICPE.2011.5728062.

[23] Introduction To Machine Learning https://towardsdatascience.com/introduction-to machinelearningdb7c668822c4 A. Asthana, S. Zafeoriou, S. Cheng, and M. Pantic. Incremental face alignment in the wild. In Conference on Computer Vision and Pattern Recognition, 2014.