A

Major Project Report on

"DRIVER DROWSINESS MONITORING SYSTEM USING VISUAL BEHAVIOUR AND MACHINE LEARNING"

submitted in partial fulfillment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

submitted by

CH. SAI SHARANYYA	**16M61A0511**
CH. SRAVANTHI	**16M61A0514**
R. V. RAMA DEVI	**16M61A0540**
SK. HASEENA BEGUM	**16M61A0546**

Under the Guidance of

Mrs. N. CHANDRAKALA, M.Tech.

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SWARNA BHARATHI INSTITUTE OF SCIENCE & TECHNOLOGY

(Approved by AICTE, Affiliated to JNTUH & Accredited by NAAC)

Pakabanda Bazar, KHAMMAM, TS-507 002

April 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SWARNA BHARATHI INSTITUTE OF SCIENCE & TECHNOLOGY

(Approved by AICTE, Affiliated to JNTUH & Accredited by NAAC)

CERTIFICATE

This is certify that the major project entitled **“** Driver Drowsiness Monitoring System using Visual Behaviour and Machine Learning**”** being submitted by

CH. SAI SHARANYYA	**16M61A0511**
CH. SRAVANTHI	**16M61A0514**
R.V. RAMA DEVI	**16M61A0540**
SK. HASEENA BEGUM	**16M61A0546**

in partial fulfillment for the award of degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING to Jawaharlal Nehru Technological University Hyderabad (JNTUH),** during the academic year 2019-20 is a record of bonafied work carried out under my guidance and supervision. The results embodied in this project report have not been submitted to any other institution for the award of any degree.

Mrs. N. Chandrakala **M.Tech.**
Assistant Professor
GUIDE

Dr.M.Senthil **M.Tech. ,Ph.D ,MISTE**
Professor
HEAD OF THE DEPARTMENT

EXTERNAL EXAMINER

Dr.G. Raja kumar, M.Tech, Ph.D, MISTE
PRINCIPAL, SBIT

ACKNOWLEDGEMENT

We take the opportunity to express our deep sense of gratitude to our internal guide, Mrs.N.Chandrakala M.Tech, Assistant Professor, Department of CSE, Swarna Bharathi Institute of Science & Technology, Khammam, for her excellent guidance, technical information and support which helped us in successfully completing this major project.

We take the opportunity to express our heartfelt thanks to the Head of the Department Dr.M.Senthil M.E., PhD., Professor, Department of Computer Science and Engineering, Swarna Bharathi Institute of Science & Technology, Khammam, for his excellent guidance, technical information and support which helped us in successfully completing this Project.

We wish to express our profound thanks to Dr.Raja kumar M.Tech., PhD., Principal, Swarna Bharathi Institute of Science & Technology, Khammam, for providing necessary facilities to make this project a success.

We wish to express our thanks to Management Members of Swarna Bharathi Institute of Science & Technology, Khammam, for providing necessary facilities to make this project a success.

We thank all the members of Teaching and Non-Teaching Staff, Department of CSE, and all those who have helped us directly or indirectly in completing our project successfully.

DECLARATION

We hereby declare that the Major project entitled **“DRIVER DROWSINESS MONITORING SYSTEM USING VISUAL BEHAVIOUR AND MACHINE LEARNING”** recorded in this project is based on our work carried out at the **“SWARNA BHARATHI INSTITUTE OF SCIENCE & TECHNOLOGY”, Khammam** during the B.Tech course.

DATE:

PLACE: Khammam

Reported by:

CH. SAI SHARANYYA	16M61A0511
CH. SRAVANTHI	16M61A0514
R.V. RAMA DEVI	16M61A0540
SK. HASEENA BEGUM	16M61A0546

CONTENTS

ABSTRACT

Drowsy driving is one of the major causes of road accidents and death. Hence, detection of driver’s fatigue and its indication is an active research area. Most of the conventional methods are either vehicle based, or behavioural based or physiological based. Few methods are intrusive and distract the driver, some require expensive sensors and data handling. Therefore, in this study, a low cost, real time driver’s drowsiness detection system is developed with acceptable accuracy. In the developed system, a webcam records the video and driver’s face is detected in each frame employing image processing techniques and machine learning techniques. Facial landmarks on the detected face are pointed and subsequently the eye aspect ratio, mouth opening ratio and nose length ratio are computed and depending on their values, drowsiness is detected based on developed adaptive thresholding and generates an alarm. The system so designed is a non-intrusive real-time monitoring system. The priority is on improving the safety of the driver without being obtrusive.

Keywords—drowsiness detection, visual behaviour, eye aspect ratio, mouth opening ratio.

List of Diagrams

List of Screen Shots

List of Abbreviations

DFD	Data Flow Diagram
OpenCV	Opensource Computer Vision
EAR	Eye Aspect Ratio
MOR	Mouth Opening Ratio

CHAPTER 1

INTRODUCTION

Driver fatigue is a significant factor in a large number of vehicle accidents. Recent statistics estimate that annually 1,200 deaths and 76,000 injuries can be attributed to fatigue related crashes. The development of technologies for detecting or preventing drowsiness at the wheel is a major challenge in the field of accident avoidance systems. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects.

1.1 Scope

The aim of this project is to develop a portable, non-intrusive drowsiness detection system. The focus will be placed on designing a system that will accurately monitor the drowsiness state of the driver in real-time.

1.2 Purpose

Today drowsy driving is a serious problem that leads to thousands of accidents each year. Present automobiles have detection system which uses intrusive methods to detect the drowsiness state of driver also there are some new cars which doesn't have this detection system. Hence this project aims to solve this problem by developing a non-intrusive and portable driver drowsiness detection system. By monitoring the visual behaviour, 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 eyes and mouth in a sequence of images of a face.

1.3 Objectives

The primary objective of the proposed system is to overcome one or more problems of the existing systems.

- To provide a system that concentrates on a non-intrusive based system
- To capture the driver image continuously
- To find eye detection fastly using non-intrusive method
- To find eye state quickly.
- To wake up the driver to avoid accidents
- To protect the people and vehicle
- To provide a Driver Drowsy Detection System that will work well in enough lighting condition
- To provide a Driver Drowsy Detection System that uses pre-trained machine learning models to reduce the chance of misdetection of face or wrong alarm
- To protect the inmates and driver.

1.4 Feasibility Study

A feasibility study is an assessment of the practicality of a proposed project or system. 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. Conducting a feasibility study is always beneficial to the project as it gives you and other stakeholders a clear picture of the proposed project.

During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

In this study project’s potential for success is evaluated; therefore, perceived objectivity is an essential factor in the credibility of the study for potential investors and lending institutions. There are five types of feasibility study—separate areas that a feasibility study examines, described below.

1.4.1 Technical Feasibility

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 involves the evaluation of the hardware, software, and other technical requirements of the proposed system.

1.4.2 Economic Feasibility

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.

1.4.3 Legal Feasibility

This assessment investigates whether any aspect of the proposed project conflicts with legal requirements like zoning laws, data protection acts or social media laws. Let’s say an organization wants to construct a new office building in a specific location. A feasibility study might reveal the organization’s ideal location isn’t zoned for that type of business. That organization has just saved considerable time and effort by learning that their project was not feasible right from the beginning.

1.4.4 Operational Feasibility

This assessment involves undertaking a study to analyze and determine whether—and how well—the organization’s needs can be met by completing the project. Operational feasibility studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

1.4.5 Scheduling Feasibility

This assessment is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, an organization estimates how much time the project will take to complete.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

Drowsy driving is one of the major causes of deaths occurring in road accidents. The truck drivers who drive for continuous long hours (especially at night), bus drivers of long distance route or overnight buses are more susceptible to this problem. Driver drowsiness is an overcast nightmare to passengers in every country. Every year, a large number of injuries and deaths occur due to fatigue related road accidents. Hence, detection of driver's fatigue and its indication is an active area of research due to its immense practical applicability.

Drowsiness is the state of feeling tired or sleepy. We all can be victim of drowsiness while driving, simply after too short night sleep, tired physical condition or during long journeys. Driver fatigue affects the driving ability in the following 3 areas, a) It impairs coordination, b) It causes longer reaction times, and, c) It impairs judgment. The number of accidents as a result of drowsiness is increasing day by day. Recent statistics estimate that annually 76,000 injuries and 1200 deaths can be attributed to drowsiness related crashes. The advancement of technology in detecting the drowsiness of the driver is a noteworthy challenge as it can help reduce the probability of accidents taking place resulting in decrease in the death and injuries caused due to drowsy driving. Considering the hazards, drowsiness presents on the road, it is necessary to develop and efficient system which can work under low ligh environment and with better and faster speed. Nowadays the driver safety in the car is one of the most wanted system to avoid accidents. Our objective of the project is to ensure the safety system. In this manner, a system which can keep a check of driver's condition for drowsiness and alert the driver before it's too late.For this we need a system which will focus on the open or closed state of driver's eyes as by monitoring the state of the eyes detection of drowsiness is easy. Detection in real-time is the major challenge in the field of accident prevention system. The purpose of this study is to provide a real-time monitoring system using video processing, face/eye detection techniques.

2.2 Existing System

Generally, the methods to detect drowsy drivers are classified in three types; vehicle based, behavioural based and physiological based. In vehicle-based method, a number of metrics like steering wheel movement, accelerator or brake pattern, vehicle speed, lateral acceleration, deviations from lane position etc. are monitored continuously. Detection of any abnormal change in these values is considered as driver drowsiness. This is a nonintrusive measurement as the sensors are not attached on the driver. In behavioural based method, the visual behavior of the driver i.e., eye blinking, eye closing, yawn, head bending etc. are analyzed to detect drowsiness. This is intrusive measurement as IR sensor is used to detect these features. In physiological based method, the physiological signals like Electrocardiogram (ECG), Electrooculogram (EOG), Electroencephalogram (EEG), heartbeat, pulse rate etc. are monitored and from these metrics, drowsiness or fatigue level is detected. This is intrusive measurement as the sensors are attached on the driver which will distract the driver.

2.3 Limitations

Depending on the sensors used in the present detection system, system cost as well as size will increase. However, inclusion of more parameters/features will increase the accuracy of the system to a certain extent. These factors motivate us to develop a low-cost, real time driver's drowsiness detection system with acceptable accuracy. Hence, we have proposed a webcam based system to detect driver's fatigue from the face image only using image processing and machine learning techniques to make the system low-cost as well as portable.

2.4 Proposed System

The proposed driver drowsiness monitoring system has been depicted in Fig 1. At first, the video is recorded using a webcam. The camera will be positioned in front of the driver to capture the front face image. From the video, the frames are extracted to obtain 2-D images. Face is detected in the frames using a pre-defined support vector machine (SVM) for object detection. After detecting the face, facial landmarks like positions of eye, nose, and mouth are marked on the images. From the facial landmarks, eye aspect ratio, mouth opening ratio are quantified and using these features a decision is obtained about the drowsiness of the driver. If drowsiness is detected, an alarm will be sent to the driver to alert him/her. The details of each block are discussed below.

-

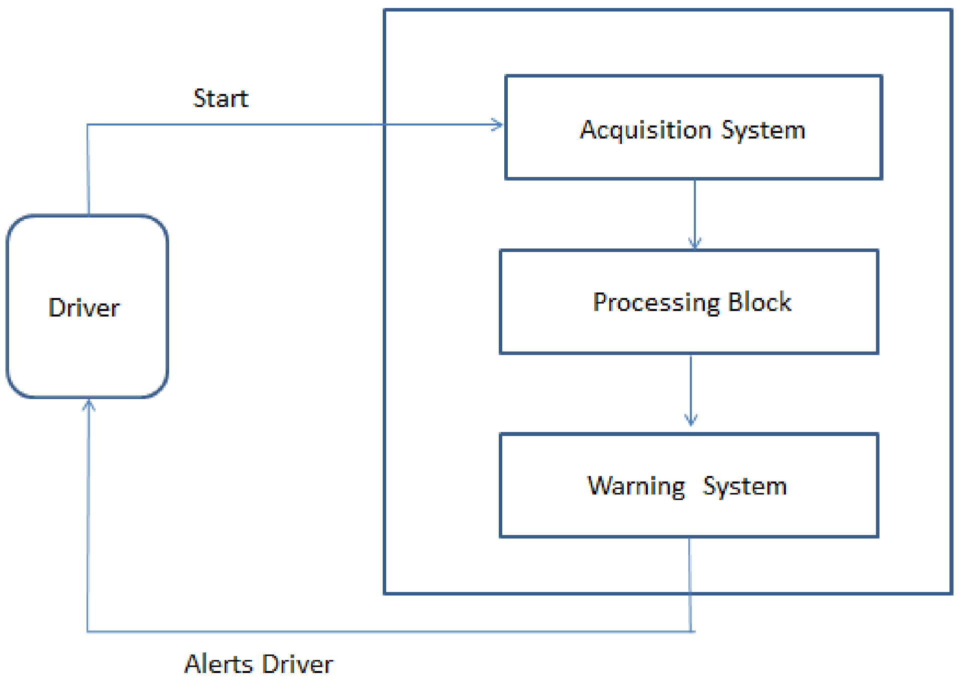


Fig 1. Architecture of Proposed System

The proposed drowsiness detection system has three blocks/modules; acquisition system, processing system and warning system. Here, the video of the driver's frontal face is captured in acquisition system and transferred to the processing block where it is processed online to detect drowsiness. If drowsiness is detected, a warning or alarm is send to the driver from the warning system.

2.5 Advantages

1. Detects drowsiness
2. No wires, sensors are to be attached or aimed at the driver
3. This system is a low-cost as well as portable.

2.6 Conclusion

In this paper, a low cost, real time driver drowsiness monitoring system has been proposed based on visual behavior and machine learning. Here, visual behavior features like eye aspect ratio, mouth opening ratio and nose length ratio are computed from the streaming video, captured by a webcam. An adaptive thresholding technique has been developed to detect driver drowsiness in real time. The developed system works accurately with the generated synthetic data.

CHAPTER 3

ANALYSIS

3.1 Introduction

Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application.

There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, scope, functional and nonfunctional requirements, software and hardware requirements of the project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

Software Requirement specification (SRS) is the starting point of the software development activity. As system grew more complex it become evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirements phase arose. The software project is initiated by the client needs. SRS is means of translating ideas into ideas of the minds of client into document.

An SRS is unambiguous if and only if every requirement stated has one and only one interpretation. Requirements are often written in natural language, which are inherently ambiguous. An SRS is verifiable if and only if every stated requirement is verifiable. A requirement is verifiable if there exists some cost-effective process that can check whether the final software meets that requirement. An SRS is consistent if there is no requirement that conflicts with another.

Problem / Requirement Analysis

The process is order and more nebulous of the two, deals with understanding the problem, the goal and constraint.

Requirement Specification

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools and checking the specifications are addressed during this activity.

The requirements phase terminates with the production of the validate SRS document. Producing the SRS document is the basic of this phase.

Role of SRS

The purpose of SRS is to reduce the communication gap between the client and the developers. SRS is the medium through which the client and user needs are accurately specified. It forms the basis of software development.

3.2 SRS

3.2.1 Software Requirements

1. Operating System : Windows 10
2. Coding Language : Python 3.7 version
3. Packages : openCV, numpy, dlib, imutils, scipy, pygame

3.2.2 Hardware Requirements

1. Processor : intel core i5
2. Hard disk : 100 GB
3. RAM : 4 GB

3.3 SYSTEM MODULES

A module is a collection of source files and build settings that allow you to divide your project into discrete units of functionality. Your project can have one or many modules and one module may use another module as a dependency. Each module can be independently built, tested, and debugged.

A module is a separate unit of software or hardware. Typical characteristics of modular components include portability, which allows them to be used in a variety of systems, and interoperability, which allows them to function with the components of other systems. The term was first used in architecture.

Additional modules are often useful when creating code libraries within your own project or when you want to create different sets of code and resources for different device types, such as phones and wearables, but keep all the files scoped within the same project and share some code.

Modules in the project

1. Video Recording
2. Face detection
3. Facial landmarks detection
4. Calculate
5. Warning

Modules Description

1. **Video Recording**: Using this module we will connect application to webcam using OPENCV built-in function called VideoCapture. And grab frames from webcam. The video is recorded using webcam and the frames are extracted and processed in a laptop.
2. **Face Detection**: After extracting the frames, first the human faces are detected. Numerous online face detection algorithms are there. In this study, linear SVM

method is used. Using SVM pre-trained machine learning drowsiness model, we will detect faces in the frame.

3. **Facial landmarks detection:** After detecting the face, the next task is to find the locations of different facial features like the corners of the eyes and mouth, the tip of the nose, left eyebrow, right eyebrow, left eye, righteye, mouth, jawline are pointed. From these, we extract the ROI(region of interest). Here we need to extract indexes of left eye, right eye and mouth from frames.
4. **Calculate**: After detecting the facial landmarks, In this module we will calculate the Eye Aspect Ratio (EAR) and Mouth Opening Ratio (MOR) with Euclidean Distance formula to check whether given face is closing eyes or yawning, if eyes closes for 10 frames continuously and mouth open as yawn then it will alert driver. The features of EAR and MOR are computed as described below.

   Eye aspect ratio (EAR): From the eye corner points, the eye aspect ratio is calculated as the ratio of height and width of the eye.

   Mouth Opening Ratio (MOR) : From the mouth corner points, the mouth opening ratio is calculated as the ratio of the height and width of the mouth.
5. **Warning**: Here, if driver in picture is blinking his eyes for consecutive 10 frames or with yawning mouth, then application will alert driver with Drowsiness messages. Here, alarm will be generated, if drowsiness is detected

3.4 CONTENT DIAGRAM PROJECT

3.4.1 Use Case Diagram

Use case diagrams show business use cases, actors, and the relationships between them. The relationships between actors and business use cases state that an actor can use a certain functionality of the business system.

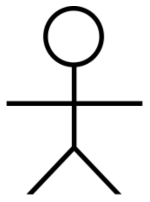
So, A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

We use the following elements in use case diagrams:

Actor

An actor represents a role that an outsider takes on when interacting with the business system. For instance, an actor can be a customer, a business partner, a supplier, or another business system. Every actor has a name

actor



Instead of a stick figure, other symbols can be used as well, if they fit the characteristics of the actor and lead to practical, easy-to-read diagrams.

Association

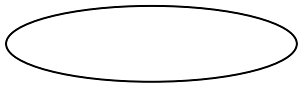
An association is the relationship between an actor and a business use case. It indicates that an actor can use a certain functionality of the business system—the business use case

Association symbol

the association does not give any information about the way in which the functionality is used. If a business use case includes several actors, it is not apparent in the use case diagram if each actor can conduct the business use case alone, or if the actors conduct the business use case together. In fact, association only means that an actor is involved in the business use case.

Use Case

A business use case describes the interaction between an actor and a business system, meaning it describes the functionality of the business system that the actor utilizes



Use case symbol

3.4.2 Sequence Diagram

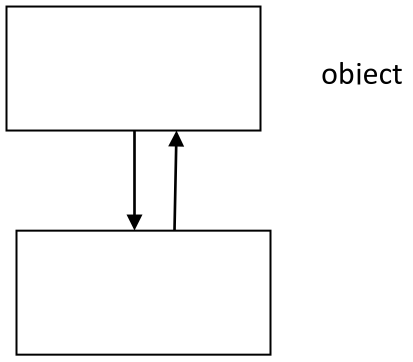
A sequence diagram is the most commonly used interaction diagram. sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

It shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

3.4.3 Collaboration diagram

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.

Collaboration diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements. Several vendors offer software for creating and editing collaboration diagrams.



3.4.4 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.Activity diagrams consist of activities that are made up of actions which apply to behavioral modeling technology.

It is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another.

Activity Diagram Notations:

Initial State or Start Point

A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram. For activity diagram using swimlanes, make sure the start point is placed in the top left corner of the first column.

Activity or Action State

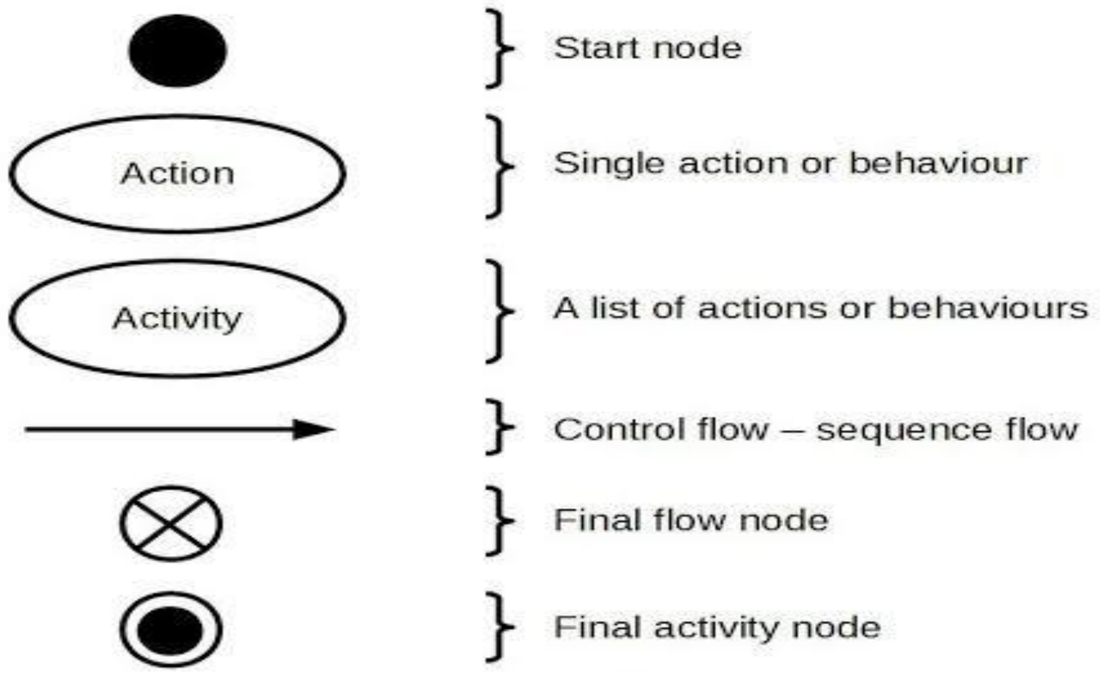
An action state represents the non-interruptible action of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.

Action Flow

Action flows, also called edges and paths, illustrate the transitions from one action state to another. They are usually drawn with an arrowed line.

Decisions and Branching

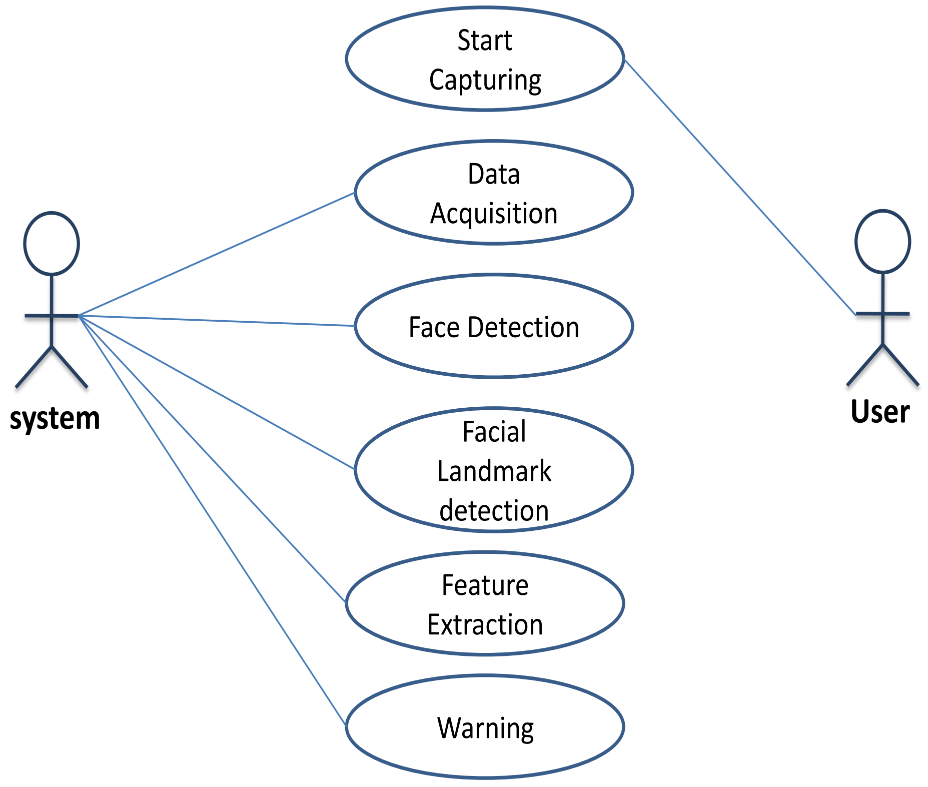
A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."


Start node
Action
Single action or behaviour
Activity
A list of actions or behaviours
Control flow – sequence flow
Final flow node
Final activity node

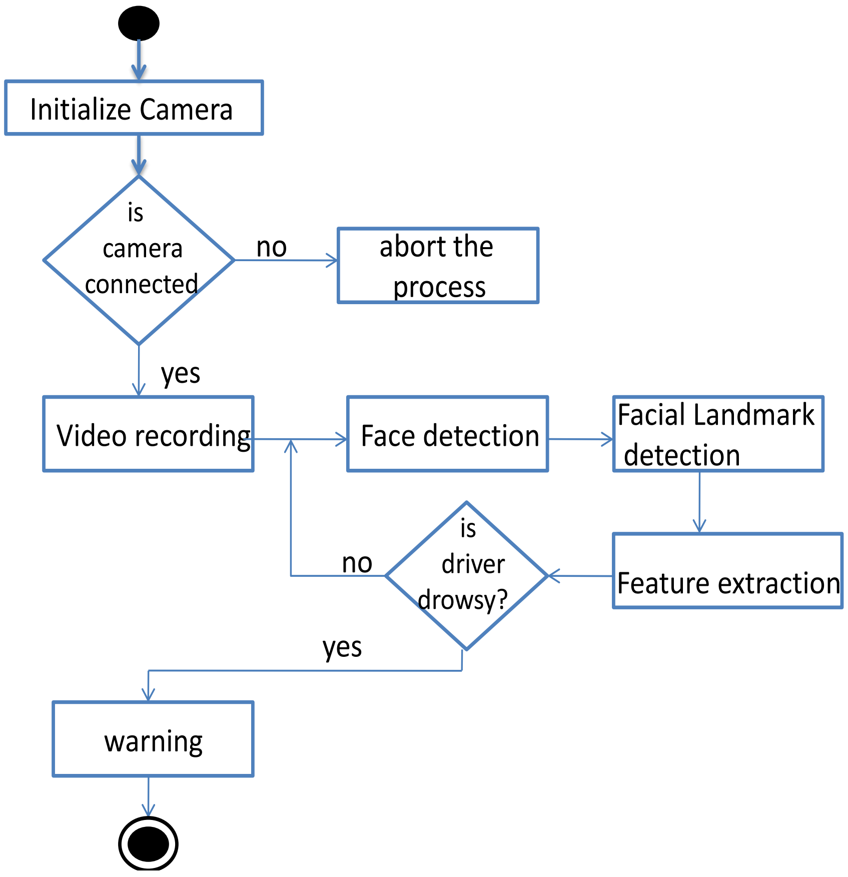
CHAPTER 4

DESIGN

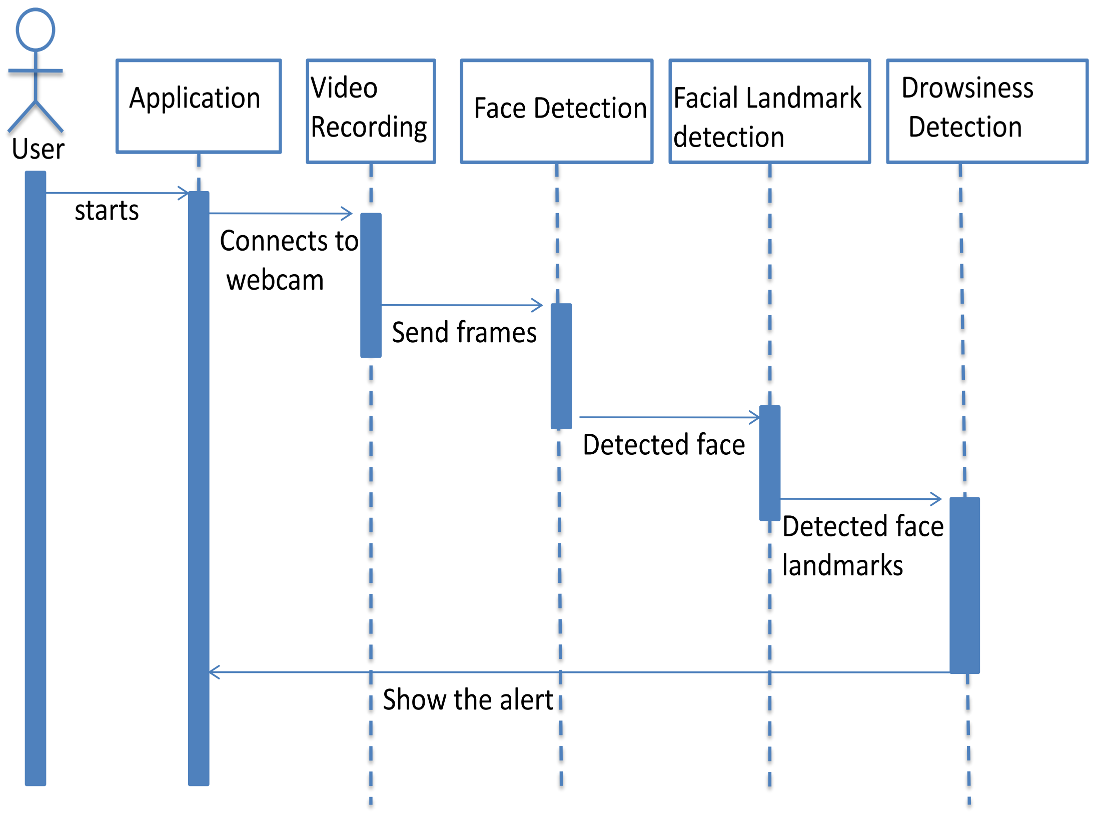
4.1 Use Case Diagram



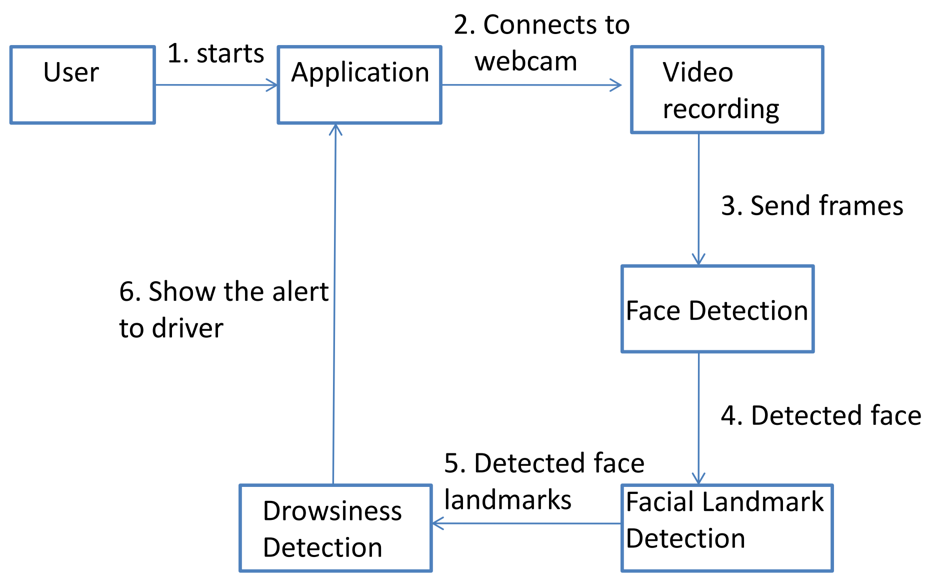
4.2 Activity Diagram



4.3 Sequence Diagram



4.4 Collaboration Diagram



4.5 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model.

The symbols depict the four components of data flow diagrams.

External entity: an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.

Process: any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as “Submit payment.”

Data store: files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as “Orders.”

Data flow: the route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like “Billing details.”

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

Rules Governing the DFD’S

a) Process:

i. No process can have only outputs.

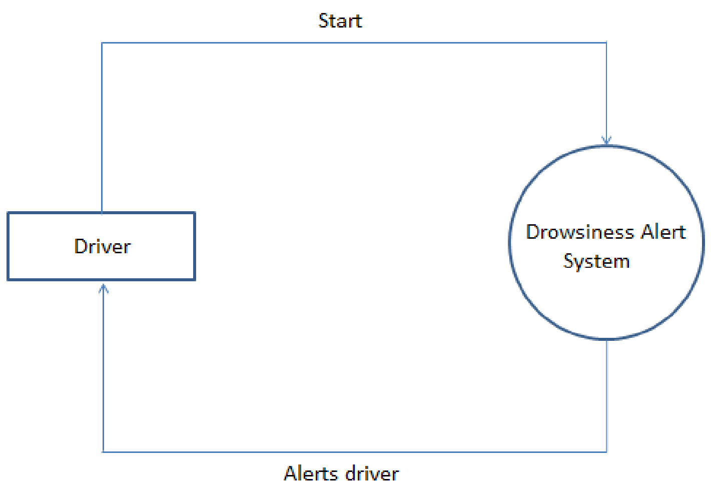
ii. No process can have only inputs. If an object has only inputs than it must be a sink.

b) Data Store:

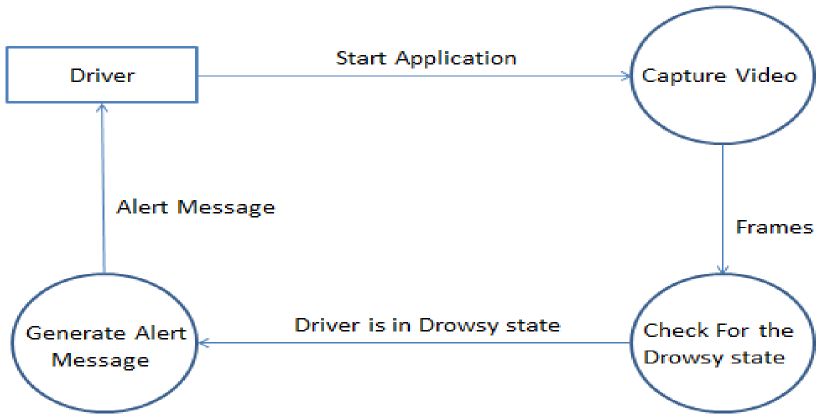
i. Data cannot move directly from one data store to another data store, a process must move data.

ii. The origin and /or destination of data. Data cannot move direly from a source to sink it must be moved by a process.

DFD Level 0 is also called a Context Diagram. It’s a basic overview of the whole system or process being analyzed or modeled. It’s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

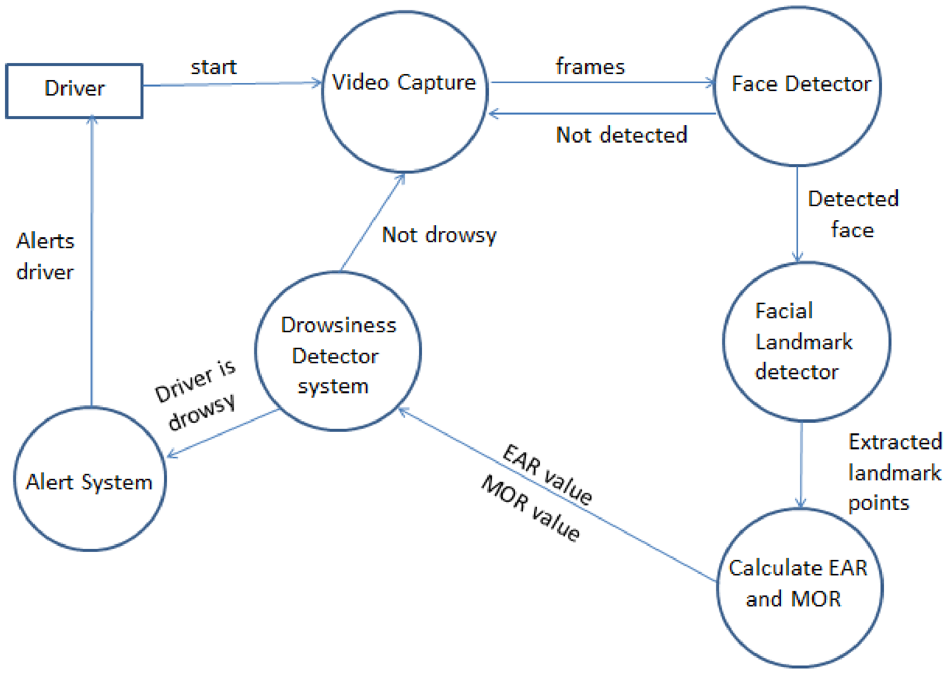


DFD Level 0



DFD Level 1

DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail about the system’s functioning.



DFD Level 2

CHAPTER 5

TECHNOLOGY DESCRIPTION

5.1 Python Programming Language

5.1.1 Introduction

Python Programming Language is a high-level and interpreted programming language which was created by Guido Van Rossum in 1989. It was first released in 1991, which results in a great general purpose language capable of creating anything from desktop software to web applications and frameworks.

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development.

Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.

Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc.

Python 2 vs. Python 3

In most of the programming languages, whenever a new version releases, it supports the features and syntax of the existing version of the language, therefore, it is easier for the projects to switch in the newer version. However, in the case of Python, the two versions Python 2 and Python 3 are very much different from each other.

A list of differences between Python 2 and Python 3 are given below:

Python 2 uses print as a statement and used as print "something" to print some string on the console. On the other hand, Python 3 uses print as a function and used as print("something") to print something on the console.

Python 2 uses the function raw input () to accept the user's input. It returns the string representing the value, which is typed by the user. To convert it into the integer, we need to use the int () function in Python. On the other hand, Python 3 uses input () function which automatically interpreted the type of input entered by the user. However, we can cast this value to any type by using primitive functions (int (), str (), etc.).

In Python 2, the implicit string type is ASCII, whereas, in Python 3, the implicit string type is Unicode.

Python 3 doesn't contain the xrange () function of Python 2. The xrange () is the variant of range () function which returns a xrange object that works similar to Java iterator. The range () returns a list for example the function range (0,3) contains 0, 1, 2.

There is also a small change made in Exception handling in Python 3. It defines a keyword as which is necessary to be used. We will discuss it in Exception handling section of Python programming tutorial.

5.1.2 Features

As a programming language, the features of Python brought to the table are many. Some of the most significant features of Python are:

Python is one of the most dynamic and versatile programming languages available in the industry today. Since its inception in the 1990s, Python has become hugely popular and even today there are thousands who are learning this Object-Oriented Programming language. If you are new to the world of programming, you have already heard the buzz it has created in recent times because of the features of Python and must be wondering what makes this programming language special.

Easy to Code

Python is a very developer-friendly language which means that anyone and everyone can learn to code it in a couple of hours or days. As compared to other object-oriented programming languages like Java, C, C++, and C#, Python is one of the easiest to learn.

Open Source and Free

Python is an open-source programming language which means that anyone can create and contribute to its development. Python has an online forum where thousands of coders gather daily to improve this language further. Along with this Python is free to download and use in any operating system, be it Windows, Mac or Linux.

Support for GUI

GUI or Graphical User Interface is one of the key aspects of any programming language because it has the ability to add flair to code and make the results more visual. Python has support for a wide array of GUIs which can easily be imported to the interpreter, thus making this one of the most favorite languages for developers.

Object-Oriented Approach

One of the key aspects of Python is its object-oriented approach. This basically means that Python recognizes the concept of class and object encapsulation thus allowing programs to be efficient in the long run.

High-Level Language

Python has been designed to be a high-level programming language, which means that when you code in Python you don’t need to be aware of the coding structure, architecture as well as memory management.

Integrated by Nature

Python is an integrated language by nature. This means that the python interpreter executes codes one line at a time. Unlike other object-oriented programming languages, we don’t need to compile Python code thus making the debugging process much easier and efficient. Another advantage of this is, that upon execution the Python code is immediately converted into an intermediate form also known as byte-code which makes it easier to execute and also saves runtime in the long run.

Highly Portable

Suppose you are running Python on Windows and you need to shift the same to either a Mac or a Linux system, then you can easily achieve the same in Python without having to worry about changing the code. This is not possible in other programming languages, thus making Python one of the most portable languages available in the industry.

Highly Dynamic

As mentioned in an earlier paragraph, Python is one of the most dynamic languages available in the industry today. What this basically means is that the type of a variable is decided at the run time and not in advance. Due to the presence of this feature, we do not need to specify the type of the variable during coding, thus saving time and increasing efficiency.

Extensive Array of Library

Out of the box, Python comes inbuilt with a large number of libraries that can be imported at any instance and be used in a specific program. The presence of libraries also makes sure that you don’t need to write all the code yourself and can import the same from those that already exist in the libraries.

Support for Other Languages

Being coded in C, Python by default supports the execution of code written in other programming languages such as Java, C, and C#, thus making it one of the versatile in the industry.

5.1.3 Python Byte Code

The source code of a programming language can be executed using an interpreter or a compiler. In a compiled language, a compiler will translate the source code directly into binary machine code. This machine code is specific to that target machine since each machine can have a different operating system and hardware. After compilation, the target machine will directly run the machine code.

In an interpreted language, the source code is not directly run by the target machine. There is another program called the interpreter that reads and executes the source code directly. The interpreter, which is specific to the target machine, translates each statement of the source code into machine code and runs it.

Python doesn't convert its code into machine code, something that hardware can understand. It actually converts it into something called byte code. So within python, compilation happens, but it's just not into a machine language. It is into byte code and this byte code can't be understood by CPU. So we need actually an interpreter called the python virtual machine. The python virtual machine executes the byte codes.

5.1.4 Python Virtual Machine

PVM is nothing but a software/interpreter that converts the byte code to machine code for given operating system.

PVM is also called Python Interpreter and this is the reason Python is called an Interpreted language. The python virtual machine executes the byte codes.

Byte code is sent to the Python Virtual Machine(PVM).Here again the byte code is executed on PVM.If an error occurs during this execution then the execution is halted with an error message.

5.1.5 Compilation of code

Python is usually called an interpreted language, however, it combines compiling and interpreting. When we execute a source code (a file with a .py extension), Python first compiles it into a bytecode. The bytecode is a low-level platform-independent representation of your source code, however, it is not the binary machine code and cannot be run by the target machine directly. In fact, it is a set of instructions for a virtual machine which is called the Python Virtual Machine (PVM).

After compilation, the bytecode is sent for execution to the PVM. The PVM is an interpreter that runs the bytecode and is part of the Python system. The bytecode is platform-independent, but PVM is specific to the target machine. The default implementation of the Python programming language is CPython which is written in the C programming language. CPython compiles the python source code into the bytecode, and this bytecode is then executed by the CPython virtual machine.

5.1.6 Python Applications

Python is known for its general purpose nature that makes it applicable in almost each domain of software development. Python as a whole can be used in any sphere of development.

Here, we are specifying applications areas where python can be applied.

1) Web Applications

We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser etc. It also provides Frameworks such as Django, Pyramid, Flask etc to design and develop web-based applications. Some important developments are: PythonWikiEngines, Pocoo, PythonBlogSoftware etc.

2) Desktop GUI Applications

Python provides Tk GUI library to develop user interface in python based application. Some other useful toolkits wxWidgets, Kivy, pyqt that are useable on several platforms. The Kivy is popular for writing multitouch applications.

3) Software Development

Python is helpful for software development process. It works as a support language and can be used for build control and management, testing etc.

4) Scientific and Numeric

Python is popular and widely used in scientific and numeric computing. Some useful library and package are SciPy, Pandas, IPython etc. SciPy is group of packages of engineering, science and mathematics.

5) Business Applications

Python is used to build Bussiness applications like ERP and e-commerce systems. Tryton is a high level application platform.

6) Console Based Application

We can use Python to develop console based applications. For example: IPython.

7) Audio or Video based Applications

Python is awesome to perform multiple tasks and can be used to develop multimedia applications. Some of real applications are: Tim Player, cplay etc.

5.1.7 Front End - Python Tkinter (Tool-Kit Interface)

a) Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The name Tkinter comes from Tk interface. Tkinter was written by Fredrik Lundh. Tkinter is free software released under a Python license.

b) Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

c) To create tkinter:

a. Importing the module – tkinter
b. Create the main window (container)
c. Add any number of widgets to the main window
d. Apply the event Trigger on the widgets.

d) tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

5.1.8 python PIP

pip is a de facto standard package-management system used to install and manage software packages written in Python. Many packages can be found in the default source for packages and their dependencies — Python Package Index (PyPI)

PIP is a package manager for Python packages, or modules. A package contains all the files you need for a module.

One major advantage of pip is the ease of its command-line interface, which makes installing Python software packages as easy as issuing a command:

pip install some-package-name

Users can also easily remove the package:

pip uninstall some-package-name

Modules are Python code libraries you can include in your project. We can import python modules according to your convenience and need. We will be discussing two ways for installing external python modules to your system, they are:

- using pip
- by downloading package independently from third party.

Installing packages using pip

pip is the preferred installer program. Starting with Python 2.7.9, it is included as default with the Python binary installers i.e Python version > 3.0 will have pre-installed pip installer in it.

For versions < 2.7.9 you have to install pip manually.

Installing third party python packages for this purpose,

1. download the compressed (tar.gz or .zip) file for the required package.
2. Extract the compressed file (to extract tar.gz file in Linux use, tar -xvzf package_name.tar.gz).
3. Now, go to the extracted package directory using.
   cd package name

Now to install the package simply run this on command line

python setup.py install

5.2. Python Packages

5.2.1 OpenCV

a) OpenCV (Open Source Computer Vision) 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 Tensor Flow, Torch/PyTorch and caffe.

b) OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation. Wrappers in other languages such as C#, Perl, Ch, Haskell, and Ruby have been developed to encourage adoption by a wider audience.

c) OpenCV runs on the following desktop operating systems: Windows, Linux, MacOS, FreeBSD, NetBSD, OpenBSD. OpenCV runs on the following mobile operating systems: Android, iOS, Maemo, BlackBerry 10. The user can get official releases from Source Forge or take the latest Sources from GitHub. OpenCV uses Cmake.
d) OpenCV-Python makes use of Numpy, which is a highly optimizedlibrary for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

In short, OpenCV is used in our application to easily load bitmap files that contain landscaping pictures and perform a blend operation between two pictures so that one picture can be seen in the background of another picture. This image manipulation is easily performed in a few lines of code using OpenCV versus other methods.

OpenCV.org is a must if you want to explore and dive deeper into image processing and machine learning in general.

5.2.2 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.

Major Features

Documentation

Unlike a lot of open source projects, this one provides complete and precise documentation for every class and function. There are also debugging modes that check the documented preconditions for functions. When this is enabled it will catch the vast majority of bugs caused by calling functions incorrectly or using objects in an incorrect manner.

Machine Learning Algorithms

- Image Processing
- Implementations of the SURF, HOG, and FHOG feature extraction algorithms.
- Tools for detecting objects in images including frontal face detection and object pose estimation.
- High quality face recognition

To detect and localize facial landmarks we'll need the dlib library. The dlib library ships with pre trained SVM based face detector along with a facial landmark predictor.

5.2.3 Numpy

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.

5.2.4 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. We'll also need the imutils package, my series of computer vision and image processing functions to make working with OpenCV easier.

5.2.5 SciPy

SciPy is an Open Source Python-based library, which is used in mathematics, scientific computing, Engineering, and technical computing.

SciPy contains varieties of sub packages which help to solve the most common issue related to Scientific Computation.

SciPy is the most used Scientific library only second to GNU Scientific Library for C/C++ or Matlab's.

Easy to use and understand as well as fast computational power. It can operate on an array of NumPy library. SciPy is built in top of the NumP. It is a fully-featured version of Linear Algebra while Numpy contains only a few features. Most new Data Science features are available in Scipy rather than Numpy.

We’ll need the SciPy package so we can compute the Euclidean distance between facial landmarks points in the eye aspect ratio calculation

5.2.6 Pygame

In order to actually play our WAV/MP3 alarm, we need the pygame library, a pure Python, cross-platform implementation for playing simple sounds.

5.3 Machine Learning

Machine learning is the kind of programming which gives computers the capability to automatically learn from data without being explicitly programmed. This means in other words that these programs change their behavior by learning from data. Python is clearly one of the best languages for machine learning. Python does contain special libraries for machine learning namely scipy, pandas and numpy which great for linear algebra and getting to know kernel methods of machine learning. The language is great to use when working with machine learning algorithms and has easy syntax relatively.

Machine Learning is said as a subset of **artificial intelligence** that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by **Arthur Samuel** in **1959**.

With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance. **A machine has the ability to learn if it can improve its performance by gaining more data.**

A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it**. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

The need for machine learning is increasing day by day. The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly. As a human, we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.

We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically. The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function. With the help of machine learning, we can save both time and money.

The importance of machine learning can be easily understood by its uses cases, Currently, machine learning is used in **self-driving cars**, **cyber fraud detection**, **face recognition**, and **friend suggestion by Facebook**, etc. Various top companies such as Netflix and Amazon have build machine learning models that are using a vast amount of data to analyze the user interest and recommend product accordingly.

At a broad level, machine learning can be classified into three types:

1. Supervised Model
2. Unsupervised Model
3. Reinforcement Model

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

Unsupervised learning is a learning method in which a machine learns without any supervision.

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classifieds into two categories of algorithms:

- **Clustering**
- **Association**

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

CHAPTER 6

SAMPLE CODE

detect_drowsiness.py

```
# 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

pygame.mixer.init()
pygame.mixer.music.load('alarm.wav')
main = tkinter.Tk()
main.title("Driver Drowsiness Monitoring")


main.geometry("500x400")
```

```
def EAR(drivereye):
    point1 = dist.euclidean(drivereye[1], drivereye[5])
    point2 = dist.euclidean(drivereye[2], drivereye[4])
    # compute the euclidean distance between the horizontal
    distance = dist.euclidean(drivereye[0], drivereye[3])
    # compute the eye aspect ratio
   ear_aspect_ratio = (point1 + point2) / (2.0 * distance)
    return ear_aspect_ratio
    def MOR(drivermouth):
    # compute the euclidean distances between the horizontal
    point   = dist.euclidean(drivermouth[0], drivermouth[6])
    # compute the euclidean distances between the vertical
    point1  = dist.euclidean(drivermouth[2], drivermouth[10])
    point2  = dist.euclidean(drivermouth[4], drivermouth[8])
    # taking average
    Ypoint   = (point1+point2)/2.0
    # compute mouth aspect ratio
   mouth_aspect_ratio = Ypoint/point
    return mouth_aspect_ratio
defstartMonitoring():
pathlabel.config(text=" Webcam Connected Successfully")
webcamera = cv2.VideoCapture(0)


svm_predictor_path = 'SVMclassifier.dat'
```

```
    EYE_AR_THRESH = 0.25
    EYE_AR_CONSEC_FRAMES = 10
    MOU_AR_THRESH = 0.75
    COUNTER = 0
    yawnStatus = False
    yawns = 0
svm_detector = dlib.get_frontal_face_detector()
svm_predictor = dlib.shape_predictor(svm_predictor_path)
 (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
 (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
 (mStart, mEnd) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]
  while True:
        ret, frame = webcamera.read()
        frame = imutils.resize(frame, width=640)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
prev_yawn_status = yawnStatus
rects = svm_detector(gray, 0)
        for rect in rects:
            shape = svm_predictor(gray, rect)
            shape = face_utils.shape_to_np(shape)
leftEye = shape[lStart:lEnd]
rightEye = shape[rStart:rEnd]
 mouth = shape[mStart:mEnd]
leftEAR = EAR(leftEye)
```

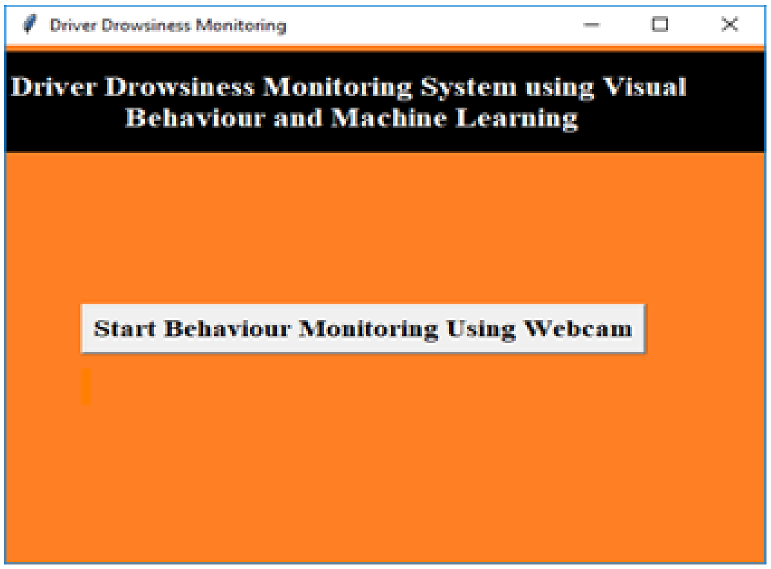
```
rightEAR = EAR(rightEye)

mouEAR = MOR(mouth)

 ear = (leftEAR + rightEAR) / 2.0

leftEyeHull = cv2.convexHull(leftEye)

rightEyeHull = cv2.convexHull(rightEye)

mouthHull = cv2.convexHull(mouth)

  cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 255), 1)

   cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 255), 1)

    cv2.drawContours(frame, [mouthHull], -1, (0, 255, 0), 1)

            if ear < EYE_AR_THRESH:

                COUNTER += 1

                cv2.putText(frame, "Eyes Closed ", (10,
30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

                if COUNTER >= EYE_AR_CONSEC_FRAMES:

pygame.mixer.music.play(-1)

                    cv2.putText(frame, "DROWSINESS ALERT!", (10,
50),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

            else:

                pygame.mixer.music.stop()

                COUNTER = 0

                cv2.putText(frame, "Eyes Open ", (10, 30),cv2.FONT_HERSHEY_SIMPLEX,
0.7, (0, 255, 0), 2)

      cv2.putText(frame, "EAR: {:.2f}".format(ear), (480,
30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

            if mouEAR> MOU_AR_THRESH:
                cv2.putText(frame, "Yawning, DROWSINESS ALERT! ", (10,
70),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

pygame.mixer.music.play(-1)

yawnStatus = True

output_text = "Yawn Count: " + str(yawns + 1)

                cv2.putText(frame, output_text, (10,100),cv2.FONT_HERSHEY_SIMPLEX,
0.7,(255,0,0),2)

            else:

                  pygame.mixer.music.stop()

                   yawnStatus = False

            if prev_yawn_status == True and yawnStatus == False:

                yawns+=1

            cv2.putText(frame, "MAR: {:.2f}".format(mouEAR), (480,
60),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

            cv2.putText(frame,"Visual Behaviour & Machine Learning Drowsiness Detection
@ Drowsiness",(370,470),cv2.FONT_HERSHEY_COMPLEX,0.6,(153,51,102),1)

        cv2.imshow("Frame", frame)

        key = cv2.waitKey(1) & 0xFF

        if key == ord("q"):

            break

    cv2.destroyAllWindows()

webcamera.release()

font = ('times', 16, 'bold')

title = Label(main, text='Driver Drowsiness Monitoring System using Visual\n
Behaviour and Machine Learning',anchor=W, justify=LEFT)

title.config(bg='black', fg='white')
title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 14, 'bold')

upload = Button(main, text="Start Behaviour Monitoring Using Webcam",
command=startMonitoring)

upload.place(x=50,y=200)

upload.config(font=font1)

pathlabel = Label(main)

pathlabel.config(bg='DarkOrange1', fg='white')

pathlabel.config(font=font1)

pathlabel.place(x=50,y=250)

main.config(bg='chocolate1')

main.mainloop()
```

CHAPTER 7

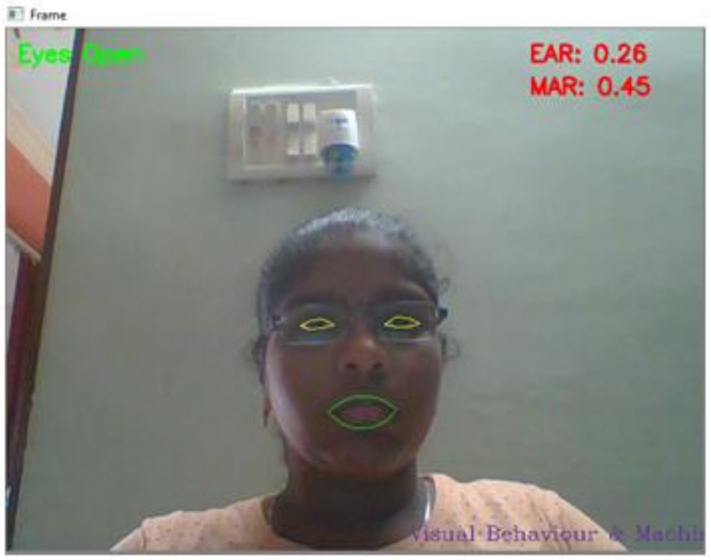
RESULTS AND DISCUSSION

The proposed system has been developed and tested with the generated data. Implementation of drowsiness detection with Python and Machine Learning was done which includes the following steps: Successful runtime capturing of video with camera. Captured video was divided into frames and each frame were analyzed. Successful detection of face followed by detection of eye and mouth. If closure of eye for successive frames were detected and If open of eye for successive frames were detected, then it is classified as drowsy condition else it is regarded as normal blink and normal open mouth the loop of capturing image and analyzing the state of driver is carried out again and again.

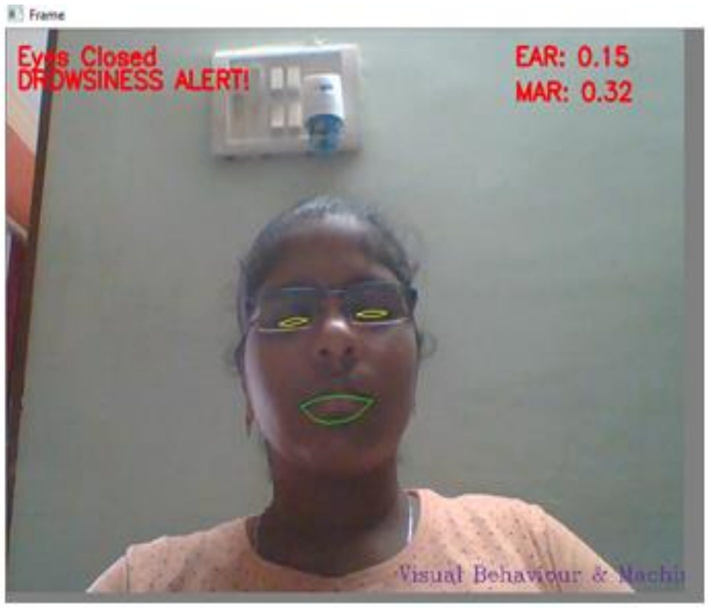
Screenshots of the system shown below:


Driver Drowsiness Monitoring
Driver Drowsiness Monitoring System using Visual Behaviour and Machine Learning
Start Behaviour Monitoring Using Webcam

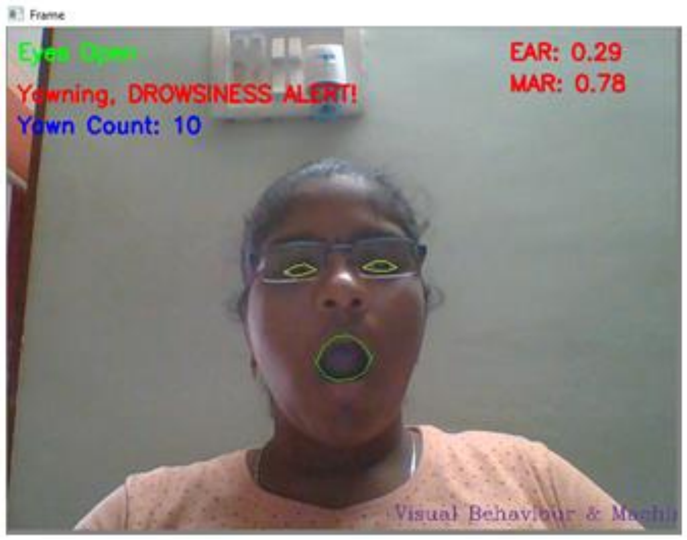
Above screen click on ‘Start Behaviour Monitoring Using Webcam’ button to connect application with webcam, after clicking button will get below screen with webcam streaming

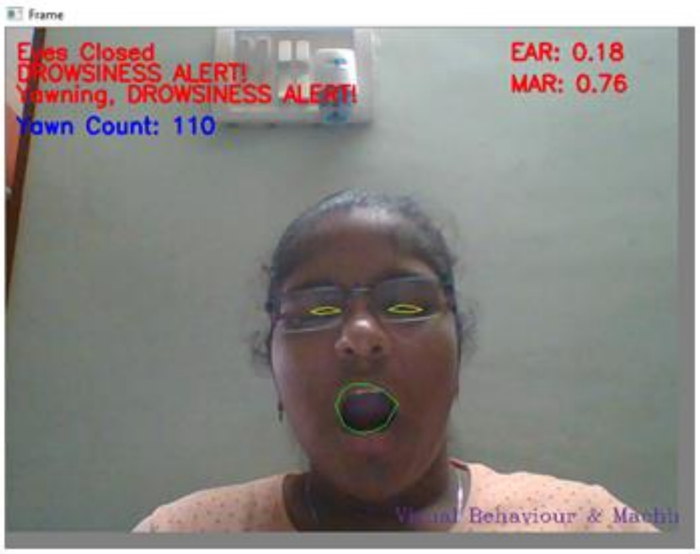


In the above screen we can see web cam streaming, then application monitor all the frames to see the person eyes are open or not, if closed then will get below message



Similarly, if mouth starts yawn then also will get alert message





CHAPTER 8

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 TYPES OF TESTS:

8.1.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.

In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

Functional testing is a form of automated **testing** that deals with how applications functions, or, in other words, its relation to the users and especially to the rest of the system. Traditionally, **functional testing** is implemented by a team of testers, independent of the developers

8.2 System Testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing predriven process links and integration points

8.2.1 White Box Testing:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

8.2.2 Black Box Testing:

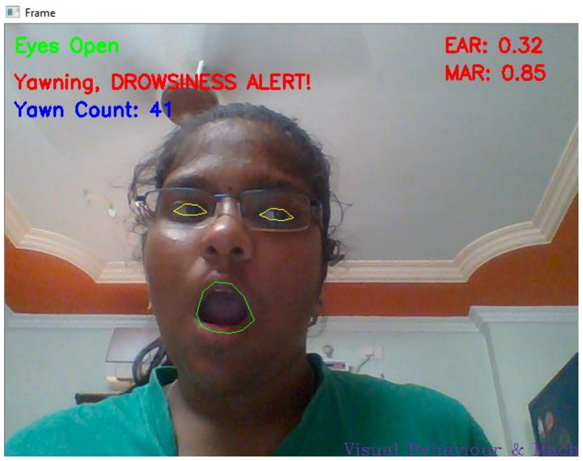
Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

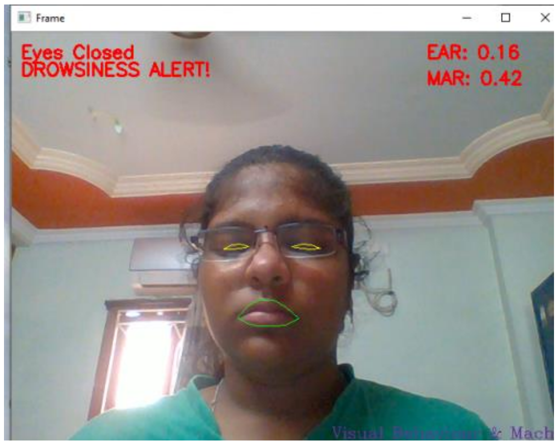
8.3 Design of Test Cases:

The tests were conducted in various conditions including:

1. Different lighting conditions.
2. Drivers posture and position of the automobile drivers face.
3. Drivers with spectacles.

Test case 1: When there is ambient light:

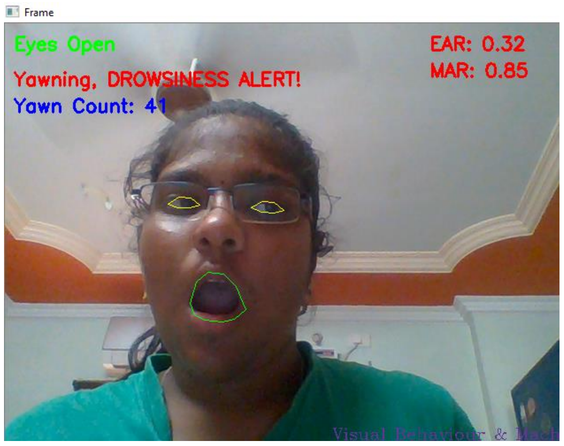


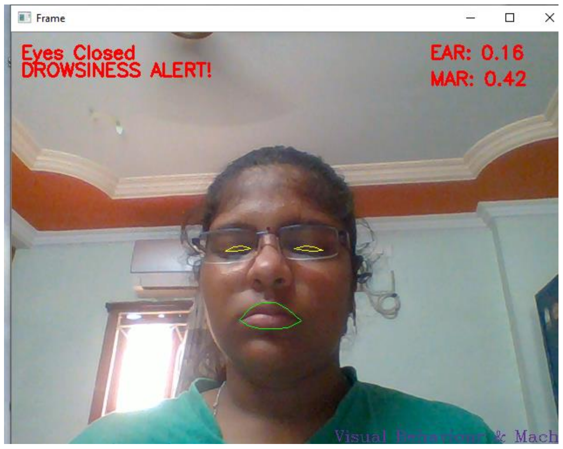


Result: As shown in Fig, when there is ambient amount of light, the automobile driver's face, eyes and mouth are successfully detected.

Test case 2: Position of the automobile drivers face:

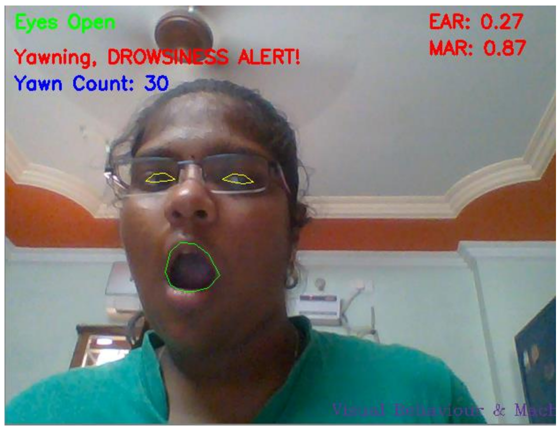
Center Positioned:-

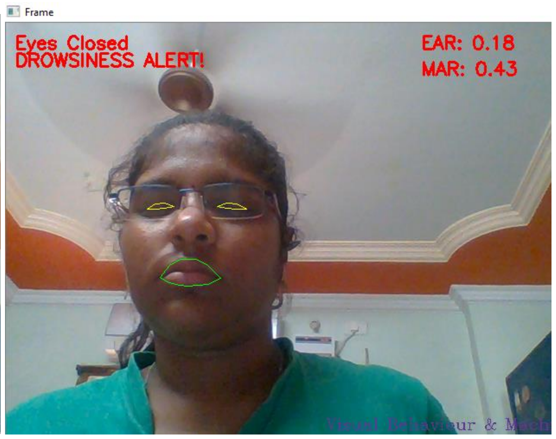




Result: As shown in Fig,When the automobile driver's face is positioned at the centre, the face, eyes, eye blinks, mouth, yawn and drowsiness was successfully detected.

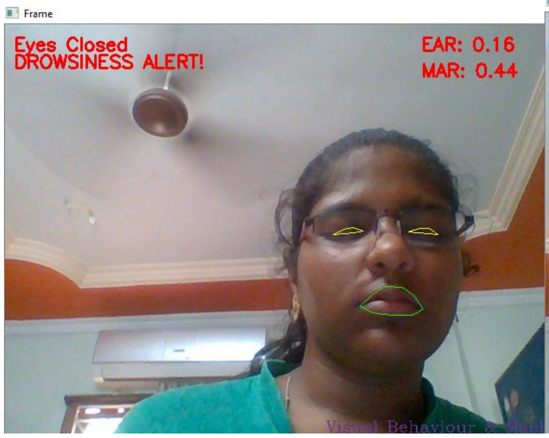
Right Positioned:-

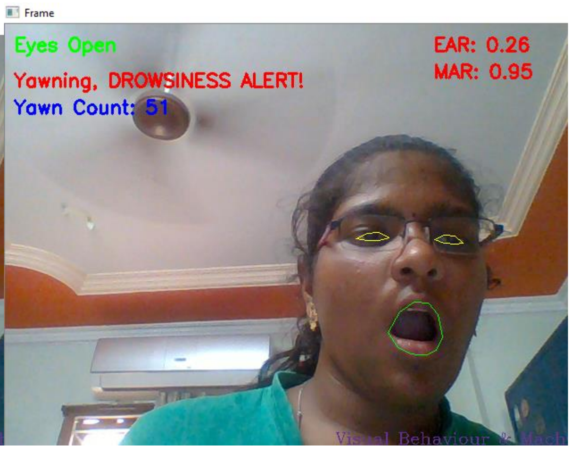




Result: As shown in Fig,When the automobile driver's face is positioned at the right, the face, eyes , eye blinks, mouth, yawn and drowsiness was successfully detected.

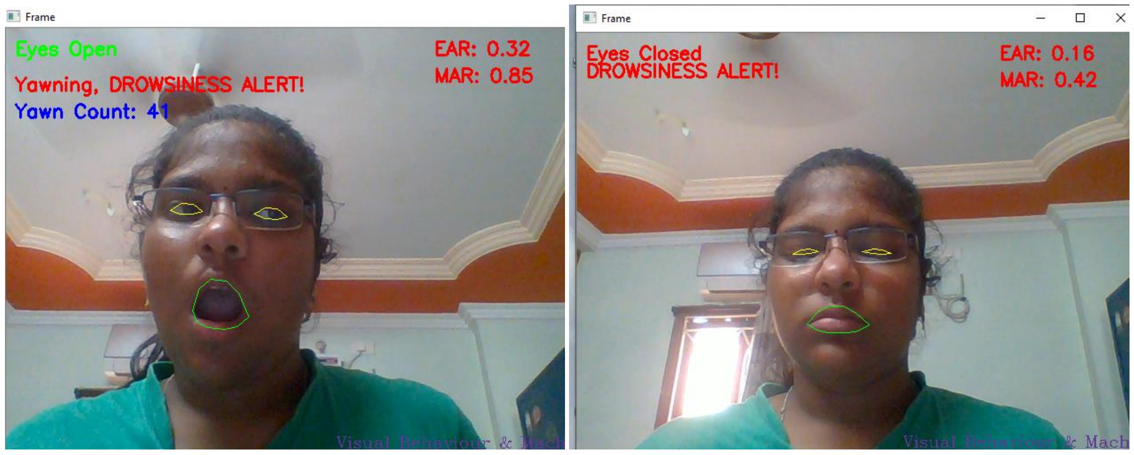
Left Positioned:-





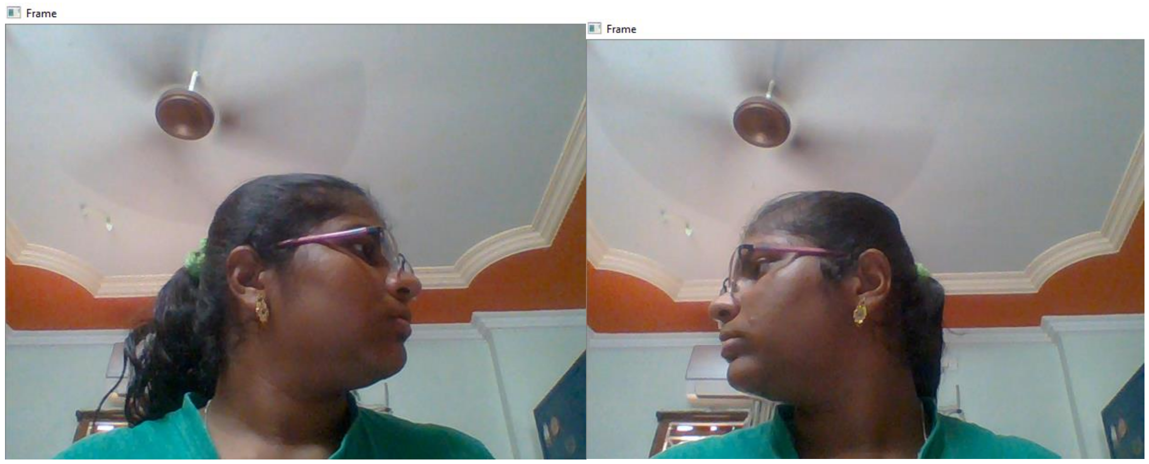
Result: As shown in Fig,When the automobile driver's face is positioned at the left, the face, eyes , eye blinks, mouth, yawn and drowsiness was successfully detected.

Test case 3: When the automobile driver is wearing spectacles



Result: As shown in screen snapshot in Fig,When the automobile driver is wearing spectacles, the face, eyes, eye blinks, and drowsiness was successfully detected.

Test case 4: When the automobile driver’s head s tilted



Result: As shown in screen snapshot in Fig, when the automobile driver's face is tilted for more than 30 degrees from vertical plane, it was observed that the detection of face and eyes failed.

Test case 5: Functional Testing

Test Cases	Eye Closure	Yawning	Result	Pass/Fail
Case -1	No	No	Awake Message	Pass
Case -2	No	Yes	Stop Yawning message alert	Pass
Case -3	Yes	No	Alarm Alert	Pass
Case -4	Yes	Yes	Alarm with message alert	Pass

Test Case 6 : Web Cam Integration

Test Case Id	TC #1
Module	Video recording
Test Action	1. Open the application 2. Click on start capturing button
Expected Result	Once after user clicks on start capturing button, System integrates with the web camera and turns on inside the specified python modules and starts streaming
Actual Result	Same as expected result
Pass/Fail	Pass

Test case 7:

Test Cases	Module Name	Test scenario	Pre-condition	Expected Result	Actual Result	Pass/Fail
Case #1	Face detection	Detecting face	TC #1	Detects the face in the frames	Same as expected	Pass
Case #2	Facial Landmarks detection	Detecting facial landmarks on face	Case #1	Once after the face is detected in frames, then facial landmarks are pointed	Same as expected	Pass
Case #3	Calculate	Calculates eye aspect ratio and mouth opening ratio	Case#2	Eye Aspect Ratio and Mouth opening ratios are calculated. Based on the values, warning module is activated.	Same as expected	Pass
Case #4	Warning	Generates an alert	Case #3	Alarm with message alert	Same as expected	Pass

CHAPTER 9

CONCLUSION

The primary goal of this project is to develop a real time drowsiness monitoring system in automobiles. We developed a simple system consisting of 5 modules namely(a) video acquisition, (b) dividing into frames, (c) face detection, (d) eye detection, and (e)drowsiness detection. Each of these components can be implemented independently thus providing a way to structure them based on the requirements.

Four features that make our system different from existing ones are:

(a) Focus on the driver, which is a direct way of detecting the drowsiness

(b) A real-time system that detects face, blink, yawn and driver drowsiness

(c) A completely non-intrusive system and

(d) Cost effective

Limitations

Multiple face problem: If multiple face arises in the window then the camera may detect more number of faces undesired output may appear. Because of different condition of different faces. So, we need to make sure that only the driver face come within the range of the camera. Also, the speed of detection reduces because of operation on multiple faces.

Future Work:

Our model is designed for detection of drowsy state of eye and give and alert signal or warning in the form of audio alarm.We can also provide the user with an Android application which will provide with the information of his/her drowsiness level during any journey. The user will know Normal state, Drowsy State, the number of times blinked the eyes according to the number of frames captures.

BIBLIOGRAPHY

IEEE standard Journal Paper,

[1] Facial Features Monitoring for Real Time Drowsiness Detection by Manu B.N, 2016 12th International Conference on Innovations in Information Technology (IIT) [Pg. 78-81] https://ieeexplore.ieee.org/document/7880030

[2] Real Time Drowsiness Detection using Eye Blink Monitoring by Amna Rahman Department of Software Engineering Fatima Jinnah Women University 2015 National Software Engineering Conference (NSEC 2015) https://ieeexplore.ieee.org/document/7396336

[3] Implementation of the Driver Drowsiness Detection System by K. Srijayathi International Journal of Science, Engineering and Technology Research (IJSETR) Volume 2, Issue 9, September 2013

Names of Websites referred

https://www.codeproject.com/Articles/26897/TrackEye-Real-Time-Tracking-Of-HumanEyes

https://realpython.com/face-recognition-with-python/
https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/
https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/
https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-dlib-opencv-python

https://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_detection.html
https://www.learnopencv.com/training-better-haar-lbp-cascade-eye-detector-opencv/