CHAPTER 1

INTRODUCTION

1.1 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. The basic 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.

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 behaviour of the driver i.e., eye blinking, eye closing, yawn, head bending etc. are analysed to detect drowsiness. This is also nonintrusive measurement as simple camera is used to detect these features. In physiological based method, the physiological signals like Electrooculogram (EOG), Electroencephalogram (EEG), heartbeat, pulse rate etc. are monitored and from these metrics, drowsiness or fatigue level is detected. Depending on the sensors used in the 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.

CHAPTER 2

LITERATURE SURVEY

1. **W. L. Ou, M. H. Shih, C. W. Chang, X. H. Yu, C. P. Fan, "Intelligent Video-Based Drowsy Driver Detection System under Various Illuminations and Embedded Software Implementation", 2015 international Conf. on Consumer Electronics - Taiwan, 2015**

   An intelligent video-based drowsy driver detection system, which is unaffected by various illuminations, is developed in this study. Even if a driver wears glasses, the proposed system detects the drowsy conditions effectively. By a near-infrared-ray (NIR) camera, the proposed system is divided into two cascaded computational procedures: the driver eyes detection and the drowsy driver detection. The average open/closed eyes detection rates without/with glasses are 94% and 78%, respectively, and the accuracy of the drowsy status detection is up to 91%. By implementing on the FPGA-based embedded platform, the processing speed with the 640×480 format video is up to 16 frames per second (fps) after software optimizations.

2. **W. B. Horng, C. Y. Chen, Y. Chang, C. H. Fan, "Driver Fatigue Detection based on Eye Tracking and Dynamic Template Matching", IEEE International Conference on Networking,, Sensing and Control, Taipei, Taiwan, March 21-23, 2004**

   Lazy driving is one of the significant reasons for street mishaps and passing . So it is vital to identify the sleepiness of the driver to save life and property. In spite of the fact that there are a few techniques for estimating the sleepiness however this methodology is totally nonmeddlesome which doesn't influence the driver in any capacity, consequently giving the specific state of the driver. In this framework, an advanced camera persistently records the video catching driver's face and changes occurring over the long haul. In this framework, a webcam records the video and driver's face is identified in each casing utilizing picture preparing strategies. Facial landmarks on the recognized face are called attention to utilizing Support Vector Machine (SVM) based pre-prepared facial milestone indicator. From this, the eye viewpoint proportion is registered. Sleepiness is distinguished from these determined values. Then SVM is utilized to check whether identified item is face or non-face. It

further screens the Eye Aspect Ratio (EAR) and Mouth Opening Ratio (MOR) of the driver up to a fixed number of casings to check the sleepiness and yawning. In the event that sleepiness is distinguished, an warning alert is given to the driver through a caution/alarm

3. **S. Singh, N. P. papanikolopoulos, "Monitoring Driver Fatigue using Facial Analysis Techniques", IEEE Conference on Intelligent Transportation System**
   In this paper, we describe a non-intrusive vision-based system for the detection of driver fatigue. The system uses a colour video camera that points directly towards the driver's face and monitors the driver's eyes in order to detect micro-sleeps (short periods of sleep). The system deals with skin-color information in order to search for the face in the input space. After segmenting the pixels with skin like colour, we perform blob processing in order to determine the exact position of the face. We reduce the search space by analysing the horizontal gradient map of the face, taking into account the knowledge that eye regions in the face present a great change in the horizontal intensity gradient. In order to find and track the location of the pupil, we use gray scale model matching. We also use the same pattern recognition technique to determine whether the eye is open or closed. If the eyes remain closed for an abnormal period of time (5-6 sec), the system draws the conclusion that the person is falling asleep and issues a warning signal.
4. **B. Alshaqaqi, A. S. Baquhaizel, M. E. A. Ouis, M. Bouumehed, A. Ouamri, M. Keche, "Driver Drowsiness Detection System", IEEE International Workshop on Systems, Signal Processing and their Applications, 2013**
   Drowsiness and Fatigue of drivers are amongst the significant causes of road accidents. Every year, they increase the amounts of deaths and fatalities injuries globally. In this paper, a module for Advanced Driver Assistance System (ADAS) is presented to reduce the number of accidents due to drivers fatigue and hence increase the transportation safety; this system deals with automatic driver drowsiness detection based on visual information and Artificial Intelligence. We propose an algorithm to locate, track, and analyse both the drivers face and eyes to measure PERCLOS, a scientifically supported measure of drowsiness associated with slow eye closure.
5. **M. Karchani, A. Mazloumi, G. N. Saraji, A. Nahvi, K. S. Haghighi, B. M. Abadi, A. R. Foroshani, A. Niknezhad, "The Steps of Proposed Drowsiness Detection System Design based on Image Processing in Simulator Driving", International Research Journal of Applied and Basic Sciences**

In our day to day life transportation systems plays an important role in human activities. Anyone can be the victim of road accidents at any time for various reasons but most of the accidents are caused due to drowsiness of the driver. The main reasons for drowsiness are due to lack of rest and sleep 5 which causes tiredness on long journeys. Due to these factors, driver vigilance will reduce which causes serious situations and increases the chances of accidents. Because of this reason yearly, most of the accident is happening all over the world. In this technology advanced era, new technologies can play an important role in providing a solution to this problem.

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXSISTING SYSTEM

There are three types of general methods to detect driver drowsiness; they are, vehicle-based, behavior based and physiological-based methods. The steering wheel movement, the accelerator of vehicle or pattern of vehicle brakes, vehicle's speed, and deviation in position of lane are monitored continuously in the method which is based on vehicle . If there is any deviation in the values detected, it is considered as driver drowsiness. The sensors are not connected to the driver and this measurement is nonintrusive.

Visual behavior like blinking of eye, closing of eye, yawning, bending of head etc. are examined for drowsiness detection in behavioral based method [1]. A simple camera is used to take images to be sent as input to SVM algorithm to identify the above features and are called nonintrusive measurement. Monitoring the physiological signals like EOG, ECG, the heartbeat, EEG, pulse rate etc. [2,3] helps in detecting driver drowsiness based on physiological method and they are intrusive measurement due to the direct connection of sensors to the driver. The current detection of the drowsiness methods are mainly based on the machine learning algorithms.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM

- Not Reliable
- May damage retina
- Highly expensive
- Intrusive
- Not portable

3.2 PROPOSED SYSTEM

Almost all drivers have experienced this drowsiness problem while driving . Youngsters and professional drivers are mostly affected by this drowsy driving because of continuous hours of driving without any rest. In many cities, auto drivers and cab drivers drive continuously overtime sometimes to complete their targets or at times to get bonus profit. Many of the poor workers in order to meet their daily expenses and for the sake

of their loved families tend to work in night shifts for long time, this can be one of the main reasons for accidents taking place because of drowsy driving.

Therefore, a driver drowsiness monitoring system has been developed in this paper. The block diagram of the proposed method for drowsiness monitoring is shown in figure

A webcam has been used to record the video of the driver. The webcam is arranged in such a way that it captures the front facial image of the driver. Once the video capturing is done, the recorded frames are then pulled out to get the 2-Dimensional images. The object (Face) in the frames is detected by HOG and SVM algorithm.

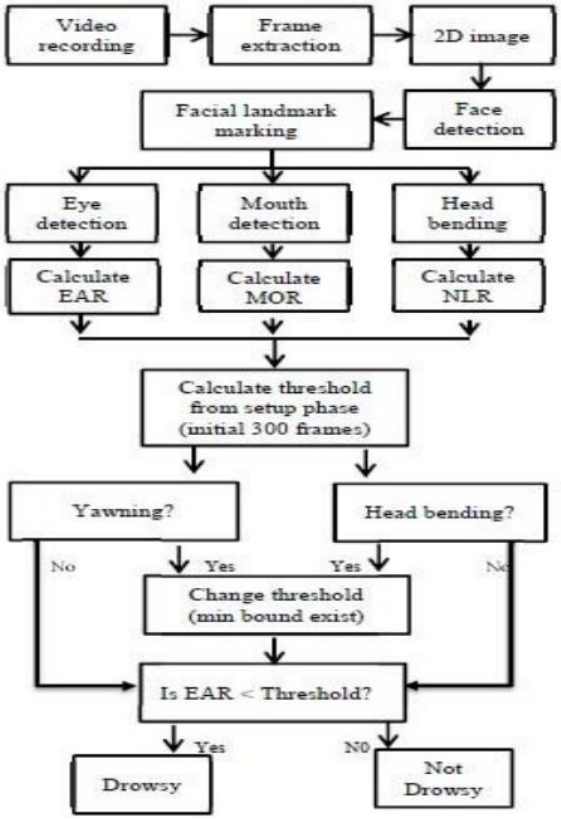

Fig 3.1 The block diagram of proposed drowsiness detection system

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

- The proposed system for drowsiness detection typically offers several advantages :
- Real-time Monitoring
- Safety
- Safety
- Early Warning
- Customizable Alerts
- Non-intrusive

3.2.1 ALGORITHMS

CNN working procedure:

To demonstrate how to build a convolutional neural network based image classifier, we shall build a 6 layer neural network that will identify and separate one image from other. This network that we shall build is a very small network that we can run on a CPU as well. Traditional neural networks that are very good at doing image classification have many more parameters and take a lot of time if trained on normal CPU. However, our objective is to show how to build a real-world convolutional neural network using TENSORFLOW Neural Networks are essentially mathematical models to solve an optimization problem. They are made of neurons, the basic computation unit of neural networks. A neuron takes an input (say x), do some computation on it (say: multiply it with a variable w and adds another variable b) to produce a value (say; z= wx + b). This value is passed to a non-linear function called activation function (f) to produce the final output (activation) of a neuron. There are many kinds of activation functions. One of the popular activation function is Sigmoid. The neuron which uses sigmoid function as an activation function will be called sigmoid neuron. Depending on the activation functions, neurons are named and there are many kinds of them like RELU, TanH.

3.2.2 METHODOLOGY

Data Acquisition :

The video is recorded using webcam (Sony CMU-BR300) and the frames are extracted and processed in a laptop. After extracting the frames, image processing techniques are applied on these 2D images. Presently, synthetic driver data has been generated. The volunteers are asked to look at the webcam with intermittent eye blinking, eye

closing, yawning and head bending. The video is captured for 30 minutes duration.

Face Detection:

After extracting the frames, first the human faces are detected. Numerous online face detection algorithms are there. In this study, histogram of oriented gradients (HOG) and linear SVM method is used. In this method, positive samples of descriptors are computed on them. Subsequently, negative samples (samples that do not contain the required object to be detected i.e., human face here) of same size are taken and HOG descriptors are calculated. Usually the number of negative samples is very greater than number of positive samples. After obtaining 11 the features for both the classes, a linear SVM is trained for the classification task. To improve the accuracy of VM, hard negative mining is used. In this method, after training, the classifier is tested on the labelled data and the false positive sample feature values are used again for training purpose. For the test image, the fixed size window is translated over the image and the classifier computes the output for each window location. Finally, the maximum value output is considered as the detected face and a bounding box is drawn around the face. This non maximum suppression step removes the redundant and overlapping bounding boxes.

Facial Landmark marking:

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 and so on. Prior to that, the face images should be normalized in order to reduce the effect of distance from the camera, non-uniform illumination and varying image resolution. Therefore, the face image is resized to a width of 500 pixels and converted to grayscale image. After image normalization, ensemble of regression trees is used to estimate the landmark positions on face from a sparse subset of pixel intensities. In this method, the sum of square error loss is optimized using gradient boosting learning. Different priors are used to find different structures. Using this method, the boundary points of eyes, mouth and the central line of the nose are marked and the number of points for eye, mouth and nose are given in Table I. The facial landmarks are shown in Fig 2. The red points are the detected landmarks for further processing.

Feature Extraction:

After detecting the facial landmarks, the features 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 as given by.

Classification:

After computing all the three features, the next task is to detect drowsiness in the extracted frames. In the beginning, adaptive thresholding is considered for classification. Later, machine learning algorithms are used to classify the data. For computing the threshold values for each feature, it is assumed that initially the driver is in complete awake state. This is called setup phase. In the setup phase, the EAR values for first three hundred (for 10s at 30 fps) frames are 12 recorded. Out 4of these three hundred initial frames containing face, average of 150 maximum values is considered as the hard threshold for EAR. The higher values are considered so that no eye closing instances will be present. If the test value is less than this threshold, then eye closing (i.e., drowsiness) is detected. As the size of eye can vary from person to person, this initial setup for each person will reduce this effect. Similarly, for calculating threshold of MOR, since the mouth may not be open to its maximum in initial frames (setup phase) so the threshold is taken experimentally from the observations. If the test value is greater than this threshold then yawn (i.e., drowsiness) is detected. Head bending feature is used to find the angle made by head with respect to vertical axis in terms of ratio of projected nose lengths. Normally, NLR has values rom 0.9 to 1.1 for normal upright position of head and it increases or decreases when head bends down or up in the state of drowsiness. The average nose length is computed as the average of the nose lengths in the setup phase assuming that no head bending is there. After computing the threshold values, the system is used for testing. The system detects the drowsiness if in a test frame drowsiness is detected for at least one feature. To make this thresholding more realistic, the decision for each frame depends on the last 75 frames. If at least 70 frames (out of those 75) satisfy drowsiness conditions for at least one feature, then the system gives drowsiness detection indication and the alarm.

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 HARDWARE REQUIREMENTS

- System : Intel Core i5
- Hard Disk : 10 GB or more
- Ram : 4 GB or more
- Display : Dual XGA (1024 x 768) or higher resolution monitors

4.2 SOFTWARE REQUIREMENTS

- Operating System : Windows 11
- Coding Language : Python 3.7.1 or above
- Library : PIP and NumPy 1.13.1
- Python
- Django

4.3 FUNCTIONAL REQUIREMENTS

- **Eye Monitoring**:

  Feature: Detect and monitor eye movements and blinks.

  Requirement: The system should be able to track the frequency and duration of eye blinks and closures.

- **Visual Alerts:**

  Feature: Provide visual warnings on the dashboard or heads-up display.

  Requirement: The system should issue clear and attention-grabbing visual alerts when drowsiness is detected.

- **Audible Alerts:**

  Feature: Generate sound warnings or alarms**.**

  Requirement: The system should include loud and distinguishable sounds to wake the driver if drowsiness is detected.

- **Head Position Monitoring**:

  Feature: Track the driver's head position and orientation.

  Requirement: The system should detect deviations from a typical driving posture, indicating potential drowsiness.

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

A block diagram of the proposed driver drowsiness monitoring system has been depicted. 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 histogram of oriented gradients (HOG) and linear 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 and position of the head are quantified and using these features and machine learning approach, 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**.**

5.2 UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

Goals:

The Primary goals in the design of the UML are as follows:

1.Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.

2. Provide extendibility and specialization mechanisms to extend the core concepts.

3. Be independent of particular programming languages and development process.

4. Provide a formal basis for understanding the modelling language.

5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

5.2.1 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information**.**

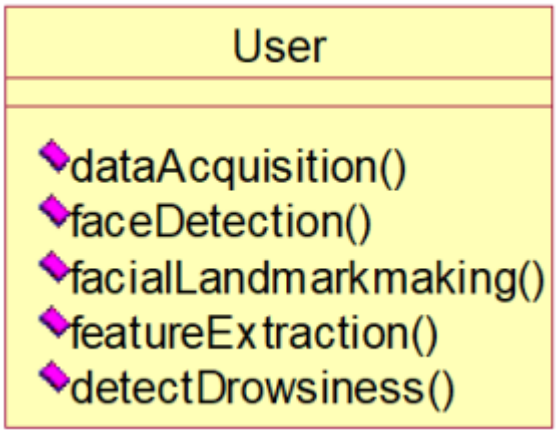

Fig5.1 Class Diagram

5.2.2 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

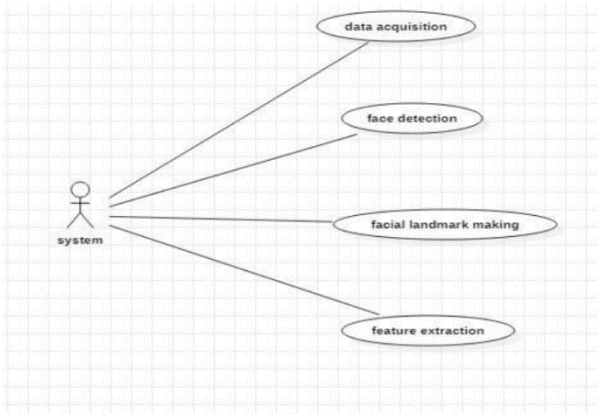

Fig 5.2Use case Diagram

5.2.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

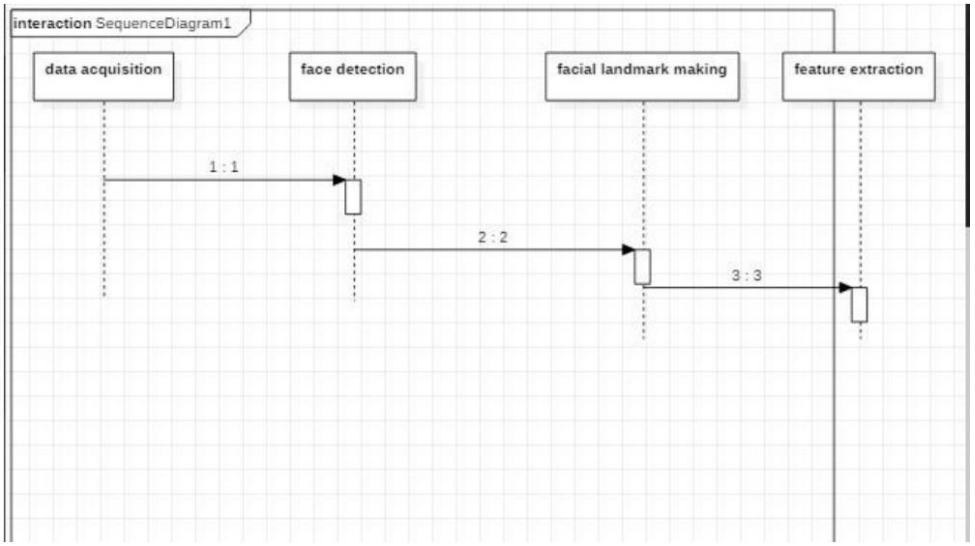

Fig 5.3 Sequence Diagram

5.2.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities .Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.[citation needed]Activity diagrams are graphical

representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities.[2][3] Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.[citation needed]

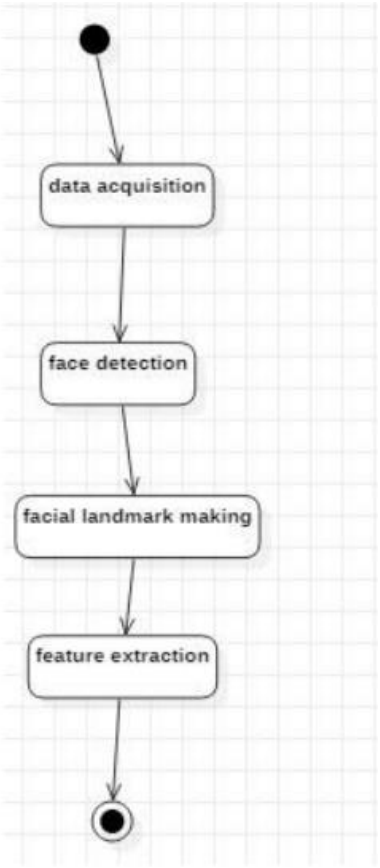

Fig 5.4 Activity Diagram

5.2.5 COMPONENT DIAGRAM

The component diagram extends the information given in a component notation element. One way of illustrating the provided and required interfaces by the specified component is in the form of a rectangular compartment attached to the component element.

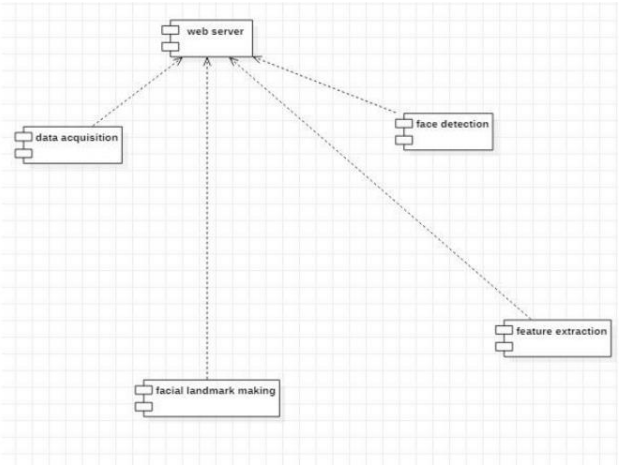

Fig 5.5 Component Diagram

5.2.6 DEPLOYMENT DIAGRAM

A deployment diagram in the Unified Modelling Language models the physical deployment of artifacts on nodes.The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

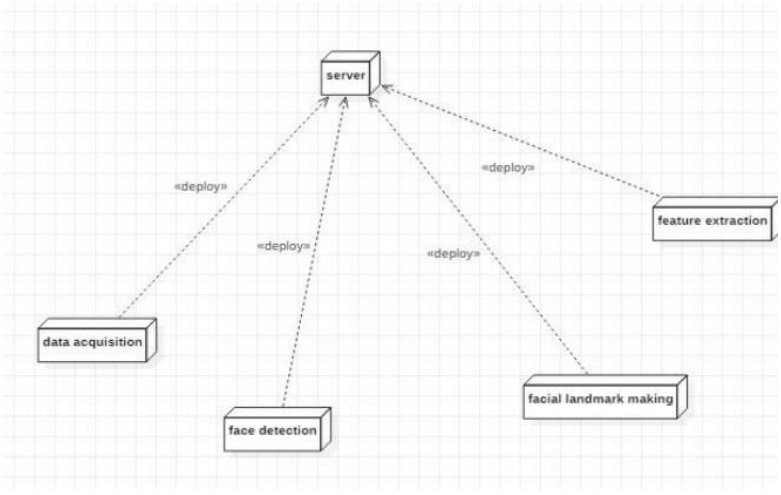

Fig 5.6 Deployment Diagram

5.3 SYSTEM SPECIFICATIONS

The project involved analysing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code 19 than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. C Python is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

DJANGO

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes re usability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models.

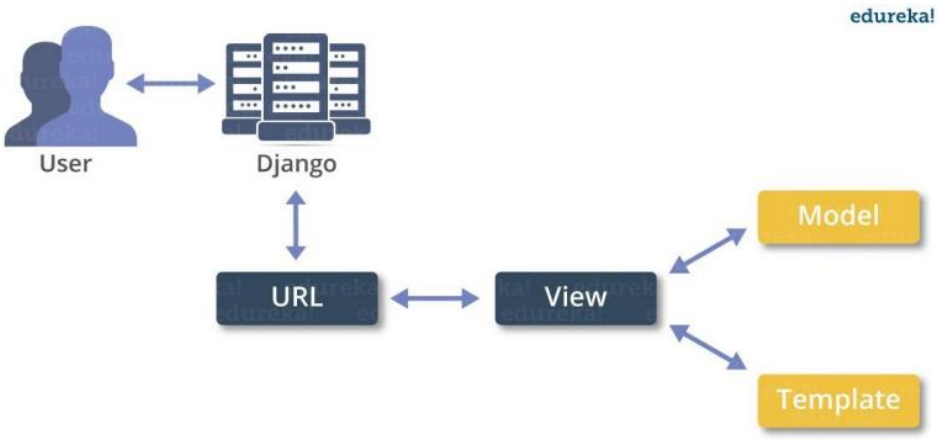

Fig 5.7 Architecture of Django

CHAPTER-6

SYSTEM STUDY

6.1 FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. 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.

Three key considerations involved in the feasibility analysis are

1. Economical Feasibility
2. Technical Feasibility
3. Social Feasibility

6.1.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased

6.1.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

6.1.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make 32 him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

CHAPTER-7

SYSTEM TESTING

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements. System testing takes, as its input, all of the integrated components that have passed integration testing.

7.1 TYPES OF TESTING

7.1.1 UNIT TESTING

Unit testing is a type of software testing that focuses on individual units or components of a software system. The purpose of unit testing is to validate that each unit of the software works as intended and meets the requirements. Unit testing is typically performed by developers, and it is performed early in the development process before the code is integrated and tested as a whole system.

Unit tests are automated and are run each time the code is changed to ensure that new code does not break existing functionality. Unit tests are designed to validate the smallest possible unit of code, such as a function or a method, and test it in isolation from the rest of the system. This allows developers to quickly identify and fix any issues early in the development process, improving the overall quality of the software and reducing the time required for later testing.

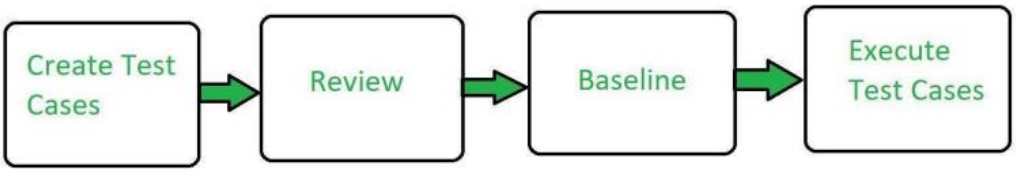

Fig 7.1 Unit Testing

Unit Testing Techniques:

There are 3 types of Unit Testing Techniques. They are

1. Black Box Testing
2. White Box Testing
3. Gray Box Testing

7.1.2 INTEGRATION TESTING

Integration testing is the process of testing the interface between two software units or modules. It focuses on determining the correctness of the interface. The purpose of integration testing is to expose faults in the interaction between integrated units. Once all the modules have been unit tested, integration testing is performed.

Integration testing is a software testing technique that focuses on verifying the interactions and data exchange between different components or modules of a software application. The goal of integration testing is to identify any problems or bugs that arise when different components are combined and interact with each other. Integration testing is typically performed after unit testing and before system testing. It helps to identify and resolve integration issues early in the development cycle, reducing the risk of more severe and costly problems later on.

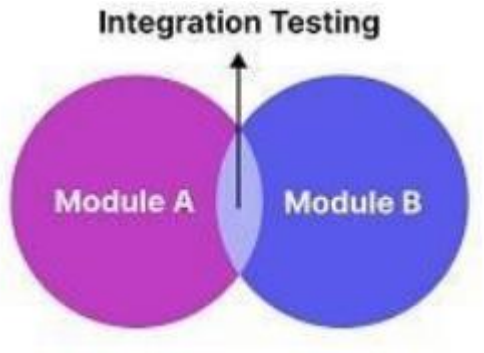

Fig 7.2 Integration Testing

There are four types of integration testing approaches:

1. Big-Bang Integration Testing
2. Bottom-Up Integration Testing
3. Top-Down Integration Testing
4. Mixed Integration Testing

7.1.3 FUNCTIONAL TESTING

Functional testing is basically defined as a type of testing that verifies that each function of the software application works in conformance with the requirement and specification. This testing is not concerned with the source code of the application. Each functionality of the software application is tested by providing appropriate test input, expecting the output, and comparing the actual output with the expected output. This testing focuses on checking the user interface, APIs, database, security, client or server application, and functionality of the Application Under Test. Functional testing can be manual or

automated.

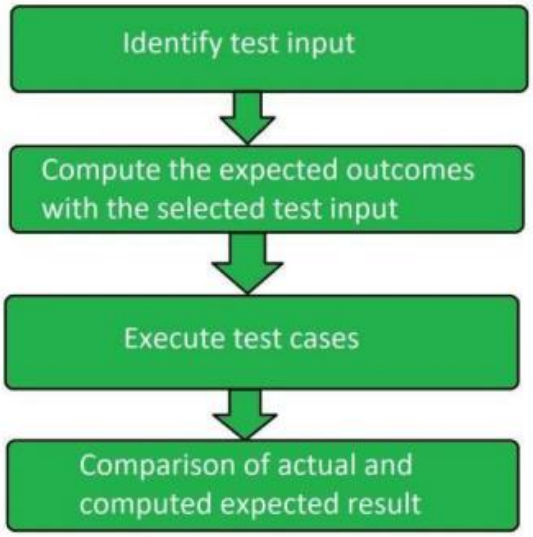

Fig 7.3 Functional Testing

7.1.4 SYSTEM TESTING

System testing is a type of software testing that evaluates the overall functionality and performance of a complete and fully integrated software solution. It tests if the system meets the specified requirements and if it is suitable for delivery to the end-users. This type of testing is performed after the integration testing and before the acceptance testing.

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behaviour of a component or a system when it is tested.

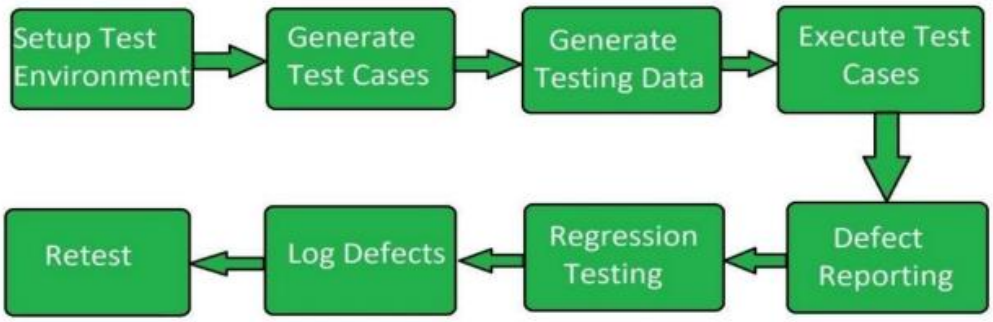

Fig 7.4 System Testing

7.1.5 ACCEPTANCE TESTING

Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system's compliance with the business requirements and verify if it is has met the required criteria for delivery to end users.

Acceptance tests are designed to replicate the anticipated real-life use of the product to verify that it is fully functional and complies with the specifications agreed between the customer and the manufacturer. These may involve chemical tests, physical tests, or performance tests, which may be refined and iterated if needed. If the actual results match the expected results for each test case, the product will pass and be considered adequate. It will then either be rejected or accepted by the customer. If it is rejected, it may be fixed or abandoned entirely if the required fixes will prove too expensive or time-consuming.

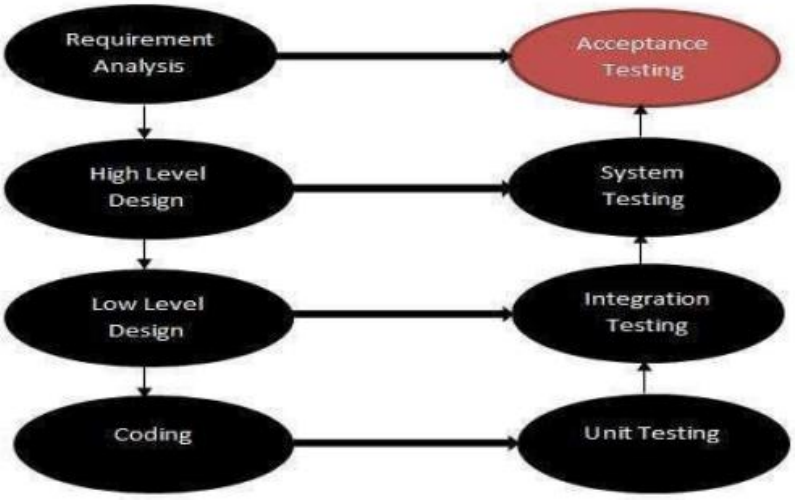

Fig 7.5 Acceptance Testing

7.2 TEST CASES

Test Case1	
Test Case Name	Empty login fields testing
Description	In the login screen if the username and password fields are empty
Output	Login fails showing an alert box asking to enter username and password.

Table 7.2.1 Test Case for Empty Login Fields

Test Case 2	
Test Case Name	Wrong login fields testing
Description	A unique username and password are set by administrator. On entering wrong username or password gives.
Output	Login fails showing an alert box username or password incorrect.

Table 7.2:2 Test Case for Wrong Login Fields

Test Case 3	
Test Case Name	User Signup Fails.
Description	User signup need to provide all data
Output	Signup Fails and an alert message appears asking to enter valid email and name.

Table 7.2.3 Test Case for Signup fail

CHAPTER-8

RESULT SCREENSHOTS

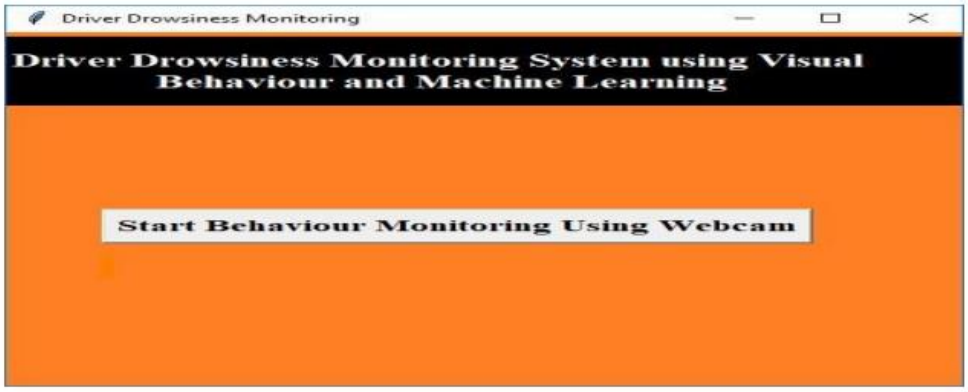

Fig 8.1 Interface of Driver's Monitoring System

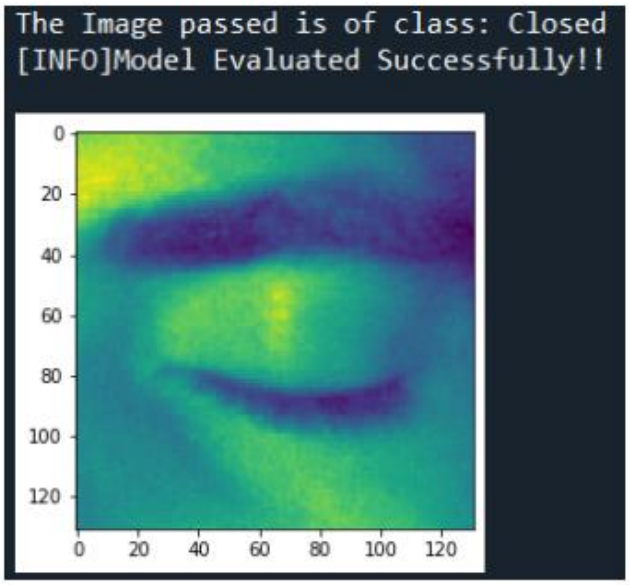

Fig 8.2Closed eye successfully being detected closed.

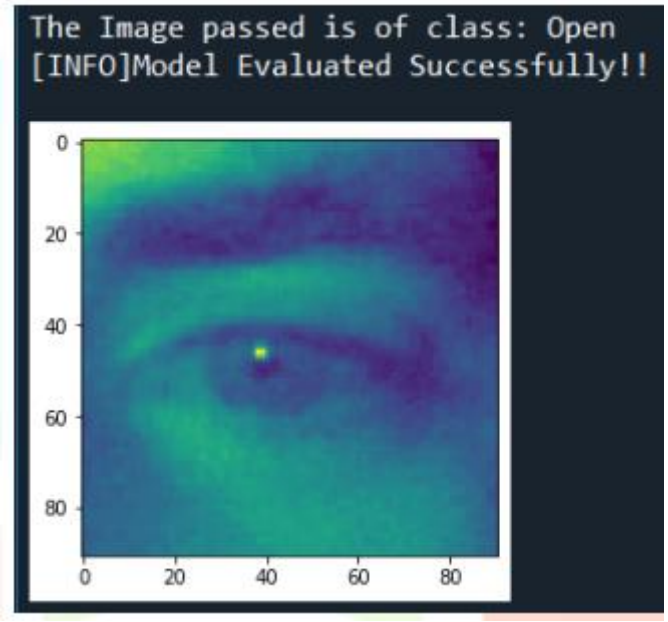

Fig 8.3 Closed eye successfully being detected opened

Fig 8.4 Initial position of Driver

Fig 8.5 Drowsy identification based on Yawning

Fig 8.6 Drowsy identification based on bent head

CHAPTER 9

CONCLUSION

9.1 CONCLUSION

This paper has designed a system for the identification of drowsiness by implementing the machine learning algorithms and visual behavior features like Eye Aspect Ratio, Mouth Opening Ratio and Nose Length Ratio. These features are calculated from the threshold values by recording the video using a webcam on laptop. This flexible technique is used for identifying the threshold value of drowsiness detection. The developed system works perfectly with the given data. Later, the values of threshold are stored and the algorithms of machine learning recused. Here, algorithms like SVM and Bayesian classifier are used, that performs the perception of SVM which is 0.569. SVM algorithm gives more accuracy and the system developed using SVM algorithm gives ideal output. In future, the proposed model can be implemented in real life as hardware in car and bus to validate the developed system.

9.2 FUTURE ENHANCEMENT

- The model can be improved incrementally by using other parameter blinking rate, state of the cars, etc.
- If all these parameter are used it can improve the accuracy by lot.
- We plan to future work on the project by adding a sensor to track heart rate in order to prevent the accident caused due to sudden heart attack to driver.
- Same model and techniques can be used for various other uses like Netflix and other streaming services can detect when is asleep and stop the video accordingly.
- It can also be used in application that prevent user from sleeping.

CHAPTER 10

BIBLIOGRAPHY

[1] **W. L. Ou, M. H. Shih, C. W. Chang, X. H. Yu, C. P. Fan.** "Intelligent Video-Based Drowsy Driver Detection System under Various Illuminations and Embedded Software Implementation", 2015 international Conf. on Consumer Electronics - Taiwan, 2015.

[2] **W. B. Horng, C. Y. Chen, Y. Chang, C. H. Fan.** "Driver Fatigue Detection based on Eye Tracking and Dynamic Template Matching", IEEE International Conference on Networking,, Sensing and Control, Taipei, Taiwan, March 21-23, 2004.

[3] **S. Singh, N. P. papanikolopoulos.** "Monitoring Driver Fatigue using Facial Analysis Techniques", IEEE Conference on Intelligent Transportation System, pp 314-318.

[4] **B. Alshaqaqi, A. S. Baquhaizel, M. E. A. Ouis, M. Bouumehed, A. Ouamri, M. Keche.** "Driver Drowsiness Detection System", IEEE International Workshop on Systems, Signal Processing and their Applications, 2013.

[5] **M. Karchani, A. Mazloumi, G. N. Saraji, A. Nahvi, K. S. Haghighi, B. M. Abadi,A. R. Foroshani, A. Niknezhad**. "The Steps of Proposed Drowsiness Detection System Design based on Image Processing in Simulator Driving", International Research Journal of Applied and Basic Sciences, vol. 9(6), pp 878-887, 2015.