STUDENT EYES CLOSURE AND YAWNING DETECTION FOR DROWSINESS ANALYSIS USING LANDMARK PREDICTOR

A CAPSTONE PROJECT REPORT

Submitted in partial fulfillment of the requirement
for the award of the
Degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

by

VISHAL KUMAR SINGH (21BCE8647)
MARKANDAY SINGH (21BCE7601)
DEEPANSHU DARIYA (21BCE8708)
KHUSHBU RANI (21BCE7468)

Under the Guidance of

DR. VIKASH KUMAR SINGH

SCHOOL OF ELECTRONICS ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237

DECEMBER 2024

CERTIFICATE

This is to certify that the Capstone Project work titled **STUDENT EYES CLOSURE AND YAWNING DETECTION FOR DROWSINESS ANALYSIS USING LANDMARK PREDICTOR** that is being submitted by **VISHAL KUMAR SINGH (21BCE8647), MARKANDAY SINGH (21BCE7601), DEEPANSHU DARIYA (21BCE8708), KHUSHBU RANI (21BCE7468)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Vikash kr. Singh

VIKASH KUMAR SINGH

Guide

The thesis is satisfactory / unsatisfactory

[illegible] [illegible]/12/2024
Hemant Kumar Reddy

Internal Examiner 1

[illegible]
S. Bharath Bhushan

Internal Examiner 2

Approved by

HoD, Department of

School of Computer Science and Engineering

ACKNOWLEDGEMENTS

I am pleased to acknowledge my sincere thanks to **Board of Management** of **VIT AP** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to the Dean, School of Computing and Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. VIKASH KUMAR SINGH** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Diversions can result in hazardous incidents that are challenging to prevent as the world's population rises. Many studies have been conducted in an effort to figure out how to stop the death toll. The designers of the car put the safety of the occupants first. Airbags are made to increase security and save lives of occupants, but they do not prevent accidents from happening. The primary causes are general exhaustion and distraction from phone alerts. In the past, a wide range of methods based on behavioral measurements and machine learning approaches have been examined to identify driver distraction These algorithms are necessary given the rapid expansion of such technology. The greatest frequent danger to their lives and the lives of others nowadays is driving while intoxicated. Although we cannot help someone stop being sleepy, we can make them awake and alert while they are driving. The suggested model was created by a few persons utilizing various technologies. But in order to increase the detection's precision, this model was created using specific technologies, namely OpenCV and Keras. A CNN model developed using Kera's will be trained using the dataset after the driver's face is detected using OpenCV and a Haar Cascade Classifier. This trained model will output whether the driver is alert or tired based on the input of the detected face. Convolution neural networks are used in this study's new methodology to identify the distraction and immediately play a loud siren to provide a sensory shock that restores attentiveness. The aforementioned hybrid strategy would result in the greatest immediate resolution to such problems in the future when combined with appropriate instruction. A Convolutional Neural Network capable of identifying the eye and mouth detection in a given image is built, trained, and tested using Keras and TensorFlow of which 2000 are drowsiness images and 2000 are yawning images, make up this dataset. These photos are loaded, then transformed into a NumPy array and normalized to have the best precision.

TABLE OF CONTENTS

List of Tables List of Figures

CHAPTER 1

INTRODUCTION

Drowsiness is a state of near sleep, where the person has a strong desire for sleep. It has two distinct meanings, referring both to the usual state preceding falling asleep and the chronic condition referring to being in that state independent of a daily rhythm. Sleepiness can be dangerous when performing tasks that require constant concentration, such as driving a vehicle. When a person is sufficiently fatigue while driving, they will experience drowsiness, and this leads to increase the factor of road accident.

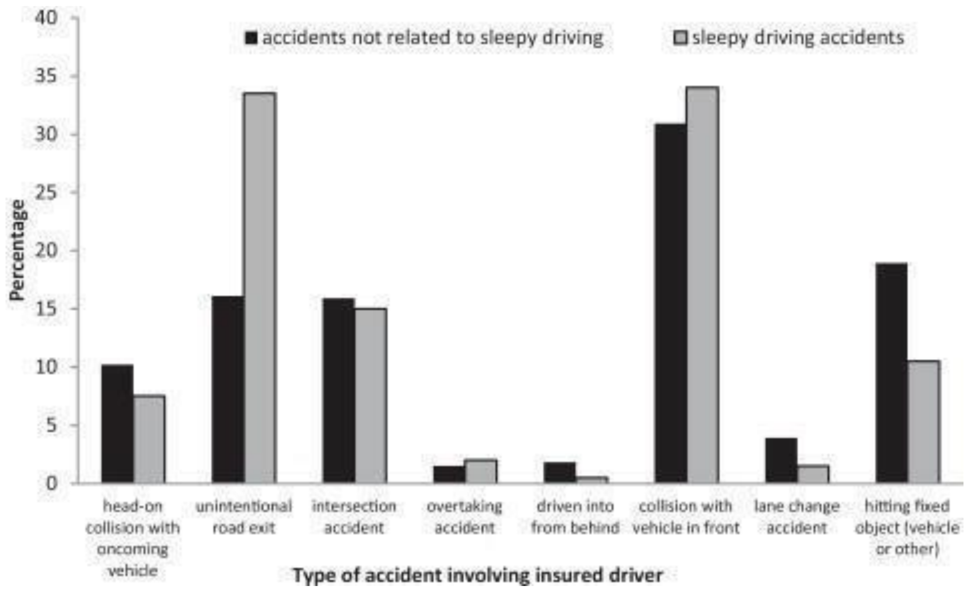


Fig 1.1: STATISTIC OF ROAD ACCIDENT

Figure 1 shows the statistic of road accident in Malaysia from the year 2005 to 2023 provided by MIROS (Malaysia Institute of Road Safety). The numbers of vehicles involved in road accident keep increasing each year. From Figure 1, car and taxi type of vehicles shows about nearly 400,000 cases of road accident has been recorded. It keeps increasing every year and by the year 2009, it shows the number of road accident were recorded by MIROS are nearly 500,000. These are significant and latent dangers responsible for much loss of lives. In recent years, scientists have been trying to prevent any further loss by pre-emptively spotting such symptoms well in advance.

These recognizing methods are characterized as subjective and objective detection. In the subjective detection method, a driver must participate in the evaluation, which is associated with the driver’s subjective perceptions through steps such as self-questioning.

Figure 1.2 shows the difference between fatigue and drowsiness condition.

The development of technologies for detecting or preventing drowsiness while driving is a major challenge in the field of accident-avoidance system. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects.

1.1 Objectives

- To develop an affordable and efficient system for detecting drowsiness based on eye closure and yawning.
- To use Raspberry Pi and camera modules to create a lightweight and portable system for real-time analysis.
- To explore and apply facial landmark detection algorithms such as those found in Dlib and OpenCV to track facial movements and compute the Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR).
- To evaluate the effectiveness of the system through real-time testing and analysis, identifying both strengths and weaknesses.

1.2 Background and Literature Survey

This problem statement has been extensively studied over the past 5 years by researchers and automotive companies in a bid to create a solution, and all their solutions vary from analyzing various patterns of distractive habits to analyzing health vitals of the driver.
The paper titled "IoT based driver drowsiness detection and health Monitoring System" by Tiwari et al[1]. was published in the International Journal of Research and Analytical Reviews in 2019.The paper proposes a system that uses IoT-based technology to monitor the driver's health and detect drowsiness while driving. The system consists of multiple sensors, including a heart rate sensor, a temperature sensor, a humidity sensor, and a camera-based drowsiness detection module.

The paper titled "Driver Drowsiness Detection Model Using Convolutional Neural Networks Techniques for Android Application" by Jabbar et al[2]. was published in the proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT).The paper proposes a driver drowsiness detection model based on convolutional neural networks (CNN) for an Android application. The proposed model uses a dataset of images of drivers to train a CNN model to detect driver drowsiness based on features such as eye closure and head movement.The authors evaluated the performance of the proposed model using an Android application that captures real-time images of the driver and applies the trained CNN model to detect driver drowsiness.

The paper titled "IOT based real-time drowsy driving detection system for the prevention of road accidents" by Hossain and George[3] was presented at the 2018 International Conference on Intelligent Informatics and Biomedical Sciences.The paper proposes a realtime drowsy driving detection system that utilizes the Internet of Things (IoT) technology to prevent road accidents. The system consists of a drowsiness detection module that uses a camera to monitor the driver's eyes and facial features for signs of drowsiness, and an alert module that triggers an alarm to alert the driver when drowsiness is detected.

The paper titled "Detecting human driver inattentive and aggressive driving behavior using deep learning: Recent advances, requirements and open challenges" by Alkinani et al[5]. was published in IEEE Access in 2020.The paper provides an overview of recent advances in detecting human driver inattentive and aggressive driving behavior using deep learning. The authors discuss the various types of data that can be used for this purpose, including video, audio, and physiological signals. The paper also discusses the requirements for developing accurate and reliable detection systems, including the need for large datasets, appropriate feature extraction techniques, and effective deep learning models.

The paper titled "Driver drowsiness detection using behavioral measures and machine learning techniques: A review of state-of-art techniques" by Ngxande et al[6]. was presented at the 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech) conference. The paper provides a review of the stateofthe-art techniques for driver drowsiness detection using behavioral measures and machine learning techniques. The authors discuss various methods for measuring driver behavior, including eye tracking, facial expression analysis, and physiological signals such as heart rate variability.

The paper titled "Driver Drowsiness Recognition via 3D Conditional GAN and Two-

Level Attention Bi-LSTM" by Hu et al[9]. was published in IEEE Transactions on Circuits and Systems for Video Technology in 2019.The paper proposes a novel approach to detecting driver drowsiness using a 3D conditional generative adversarial network (GAN) and a two-level attention bidirectional long short-term memory (Bi-LSTM) network. The proposed method captures the spatiotemporal information from multiple modalities, including images and physiological signals, to recognize driver drowsiness.

The paper titled "Internet of Things Based Intelligent Drowsiness Alert System" by

Bala and Sarath[10] was published in the proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES).The paper proposes an Internet of Things (IoT) based intelligent drowsiness alert system that utilizes sensors to detect the driver's drowsiness and alert the driver before an accident occurs. The proposed system consists of a microcontroller, a drowsiness detection unit, a GSM module, and an alert unit. The drowsiness detection unit in the proposed system uses a sensor to detect the driver's eyelid movements and a microphone to detect the driver's snoring sounds. If the driver shows signs of drowsiness, the alert unit in the system emits an alarm and sends an alert message to a preconfigured phone number using the GSM module.

1.3 Organization of the Report

The remaining chapters of the project report are described as follows:

- Chapter 2 contains the proposed system, methodology, hardware and software details.
- Chapter 3 gives the cost involved in the implementation of the project.
- Chapter 4 discusses the results obtained after the project was implemented.
- Chapter 5 concludes the report.
- Chapter 6 consists of codes.
- Chapter 7 gives references.

CHAPTER 2

TITLE OF THE CHAPTER

This Chapter describes the proposed system, working methodology, software and hardware details.

2.1 Proposed System

The following block diagram shows the system architecture of this project.

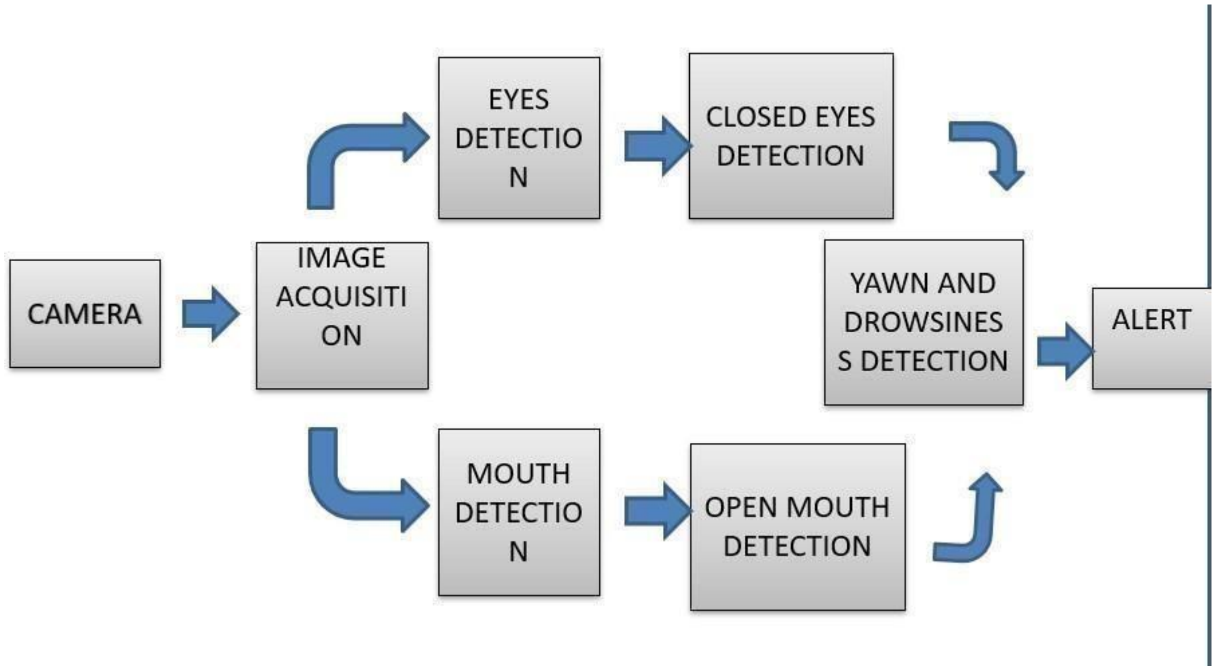
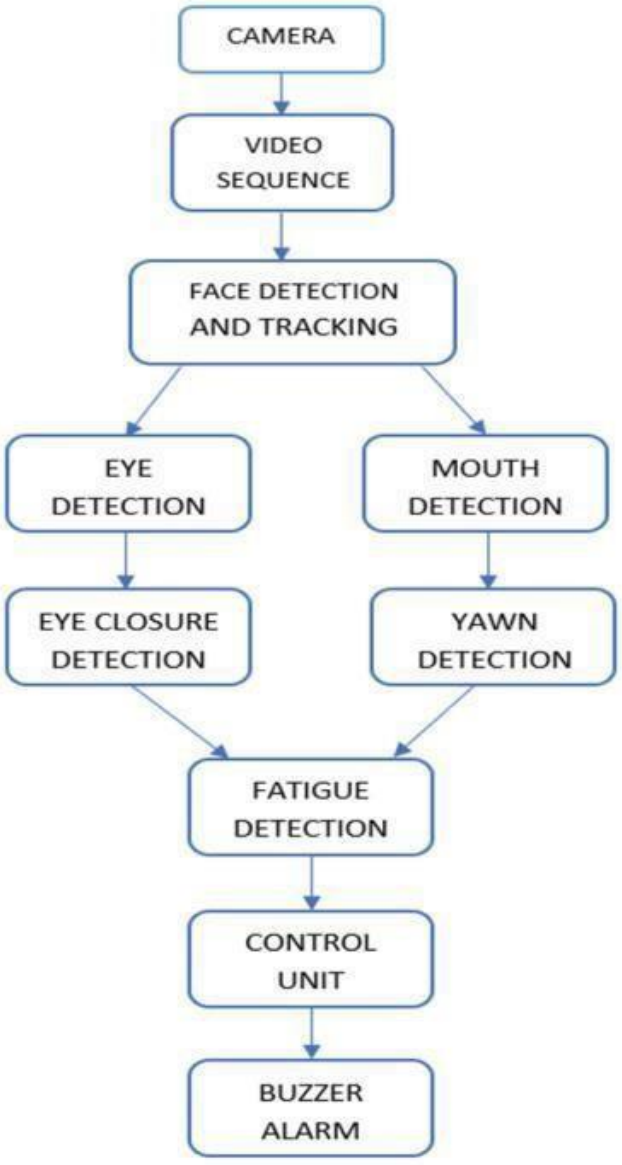


Figure 2.1 block diagram

The camera captures the image of the person inside the car and sends that to the HOG model to train and it detects each feature from the face using facial landmark technique in the system. The next step in the process would be as it starts to find the position and condition of each feature to analyze, it should detect whether the person is sleeping or not. If any of the features especially the eye and the head pose/state of the person are detected to be ab normal then automatically the system start stop produce the siren sound. This is where we will get the final judgment of the state of the driver, whether he is concentrated or distracted.

2.2 Working Methodology



2.2.1 VIDEO SEQUENCE

The input video sequence from the camera is analysed in the first step. The input can be from a webcam or CSI camera in raspberry pi. In this step the video sequence got from the camera is processed in the open-cv environment. The video sequence is converted from colour to black and white, as the black and white images can be processed at a higher speed. This video sequence is now will be processed in the open cv environment.

2.2.2 FACE DETECTION USING EYE ASPECT RATIO

Using dlib and OpenCV we can detect facial landmarks in an image. Our first step in detecting facial landmarks is to localize the face in an image and then detecting the eyes and the mouth region (ROI). Any region of the face can be detected using the 68 facial landmarks coordinates.

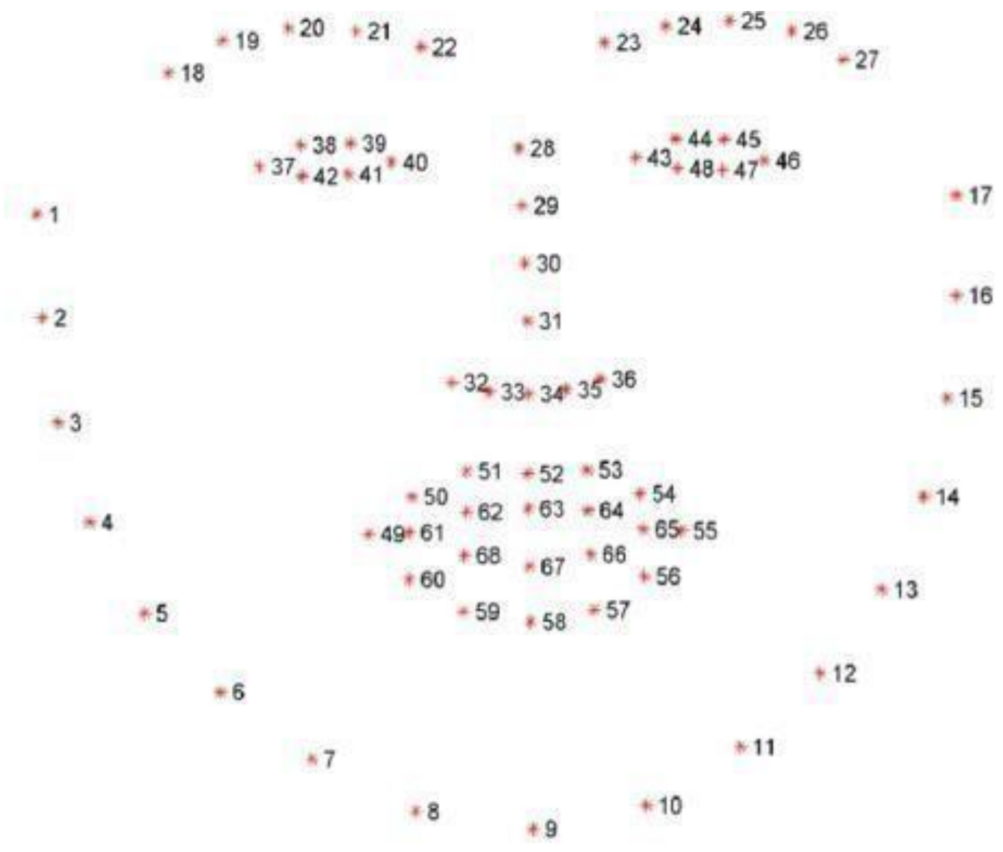


Figure 2.2.1 FACE COORDINATES

2.2.3 EYE CLOSURE DETECTION

The indexes of the 68 coordinates can be visualized on the image based on which the ROI is selected.

Points from 37 to 42 mark the right eye and 43 to 47 mark the left eye. The points from 49 to 60 mark the outer portion of the lips and 61 to 68 mark the inner lips. The face is actively tracked in real-time.

The EAR is given by the following equation as there is a correlation between the width and height of the coordinates, where p1 to p6 are the 2D landmark locations.

The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye. It is partially person and head pose insensitive. EAR of the open eye has a little difference among people and it is completely invariant to a uniform scaling of the picture and in-plane pivot of the face. Since blinking is performed by the two eyes simultaneously, the average of both the EARs of the two eyes is obtained.

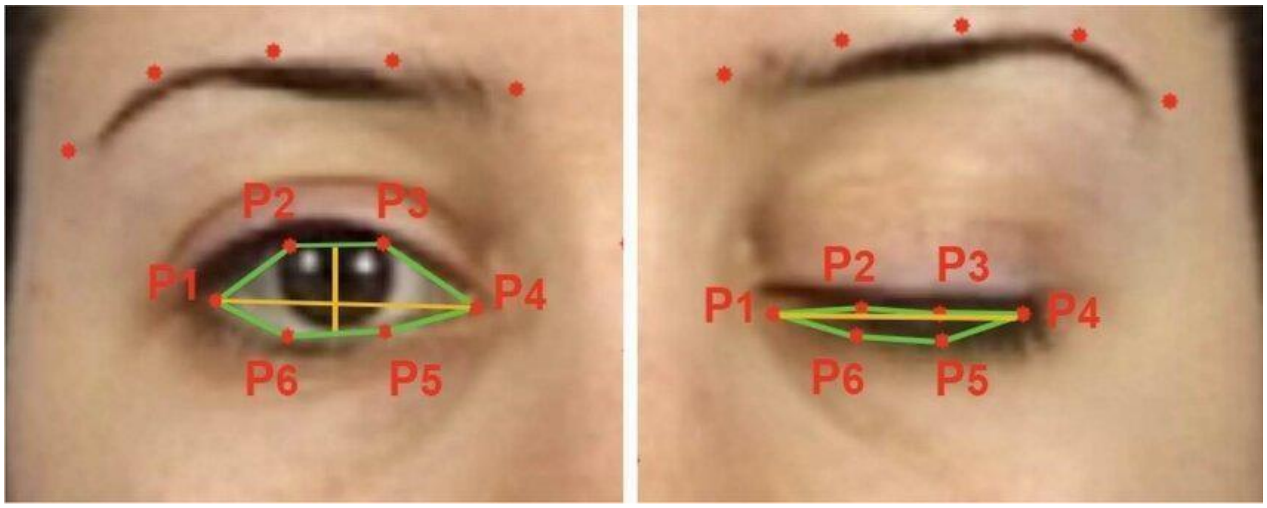
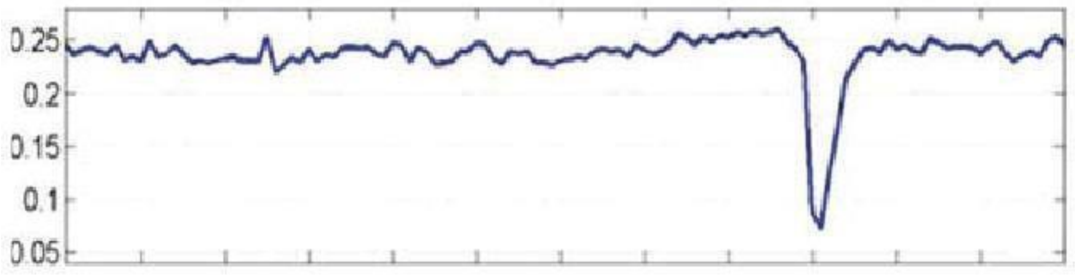


Figure 2.2.2 EYE CO-ORDINATES

left: Eye landmarks when the eye is open.

right: Eye landmarks when the eye is closed.

$$EAR = \frac{|p2-p1|+|p3-p5|}{2|p1-p4|}$$



Plotting the eye aspect ratio over time. The dip in the eye aspect ratio depicts a blink.

2.2.4 YAWN DETECTION

The mouth part is detected by the facial landmark detector and the landmark point ranges from 61 to 68. Here we are only using the innermost landmark points only. These points are displayed below.

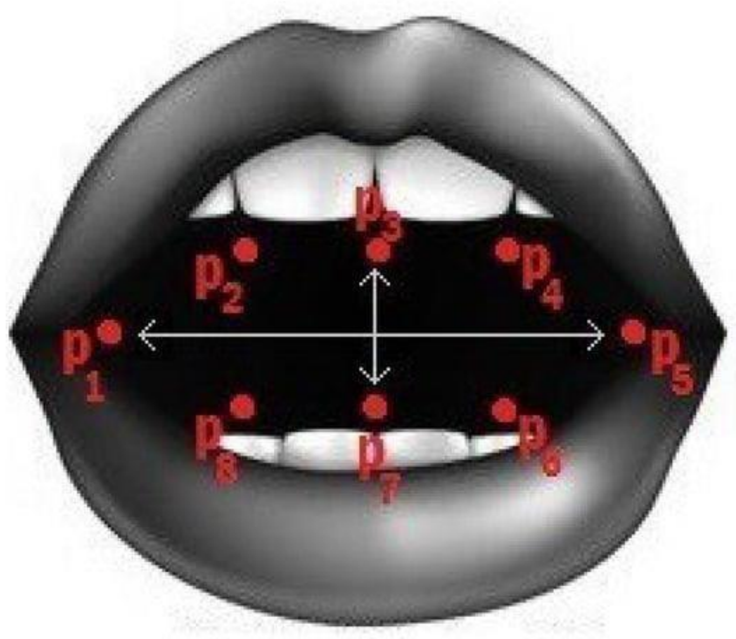


Figure 2.2.3 MOUTH CO-ORDINATES

$$MAR = \frac{|p2 - p8| + |p3 - p7| + |p4 - p6|}{3|p1 - p5|}$$

we are considering both 3 points in the mouth for better accuracy. As we can see if the mouth opened the distance between the points (p2 and p8), (p3 and p7), and (p4 and p6). If the distance between these points increases the MAR ratio also increases.

If it crosses the threshold of 30 the person will be alerted and asked to take some fresh air.

2.3 Standards

Various standards used in this project are:

- **SSL secured connection using firebase**

  The Secure Sockets Layer (SSL)—now technically known as Transport Layer Security (TLS)—is a common building block for encrypted communications between clients and servers. In a typical SSL usage scenario, a server is configured with a certificate containing a public key as well as a matching private key. As part of the handshake between an SSL client and server, the server proves it has the private key by signing its certificate with public-key cryptography.

- **Secure Shell** (**SSH**)

  It is a cryptographic network protocol for operating network services securely over an unsecured network. Typical applications include remote command-line login and remote command execution, but any network service can be secured with SSH. SSH provides a secure channel over an unsecured network in a client–server architecture, connecting an SSH client application with an SSH server. The protocol specification distinguishes between two major versions, referred to as SSH-1 and SSH-2. The standard TCP port for SSH is 22. We are using it for PuTTy as SSH Client here in our project for connection with Raspberry Pi.

2.4 System Details

This section describes the software and hardware details of the system:

2.4.1 Software Details

1. **ANACONDA NAVIGATOR:**

Anaconda Navigator is remembered for the Anaconda distribution and permits clients to send off applications and oversee condo packages, environments, and channels without using command-line commands. Navigator can search for packages, install them in an environment, run the packages and update them. With the help of the anaconda command prompt, all necessary packages are installed.

1. **GIT-HUB:**

Git-hub is used for cloning repositories like UCI and uploading the files into Git- hub and saving them there. GitHub also supports the development of open-source software, which is software with source code that anyone can inspect, modify, and enhance the models. Kaggle might also use for training the model as it supports faster execution as it supports GPU.

2. **OPEN CV:**

OpenCV (Open-Source Computer Vision Library) is a popular open-source computer vision and machine learning library. It provides a wide range of tools and functions for image and video processing, feature detection, object recognition, and machine learning.

Some of the key features of OpenCV include:

a) Image and Video Processing: OpenCV provides tools for loading, manipulating, and saving images and videos. It also supports different image and video file formats.

b) Feature Detection: OpenCV includes functions for detecting and extracting features from images and videos, such as corners, edges, and blobs.

c) Object Recognition: OpenCV includes functions for detecting and recognizing objects in images and videos using machine learning algorithms, such as Haar cascades and deep learning models.

d) Machine Learning: OpenCV includes tools and functions for training and evaluating machine learning models, such as support vector machines (SVMs) and artificial neural networks (ANNs).
e) GUI and Visualization: OpenCV provides functions for creating GUI applications and visualizing data, such as plotting and displaying images and videos.

2.4.2 Hardware Details

As shown in Figure 2 we have various hardware components being used in this system. The details of each component is as follows

i) Raspberry Pi

The Raspberry Pi is a low cost, credit-card sized computer as shown in Figure 4 that plugs into a computer monitor or TV and uses a standard keyboard and mouse.

ii) Camera Module

The Raspberry Pi Camera Module captures real-time video feed, which is processed by the system to detect facial landmarks.

Raspberry Pi Camera
Rev 1.3

CHAPTER 3 COST ANALYSIS

3.1 List of components and their cost

The costs of the various components used in this project are given below in Table 1.

Table 1. List of components and their costs

COMPONENT	COST
Raspberry Pi	₹ 3000
Camera Module V2	₹ 2000
MicroSD Card	₹ 850
Alert Sensor	₹ 350
HDMI Cable	₹ 200
GPS Module	₹ 400
Power Supply (5V 3A)	₹ 680
DC Motor	₹ 150
Adhesive or Mounting Tape	₹ 50
TOTAL	₹ 7680

CHAPTER 4 RESULTS AND DISCUSSIONS

Drowsiness detection using Convolutional Neural Networks (CNNs) has been an active area of research in recent years, with the potential to improve safety in various fields such as transportation and healthcare. In this task, the goal is to develop a model that can accurately detect drowsiness in real-time by analyzing visual cues such as eye and head movements.

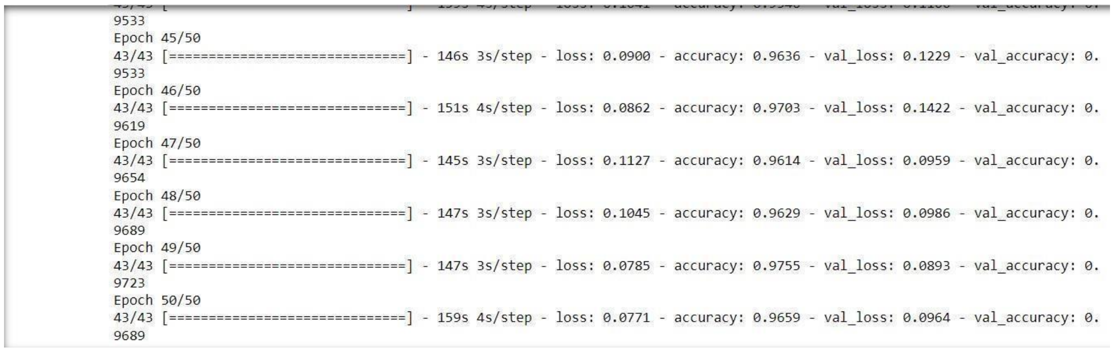
Several studies have demonstrated the effectiveness of CNN-based approaches for drowsiness detection. These approaches typically involve training a CNN on a large dataset of annotated videos or images, where the model learns to automatically extract relevant features from the input data. The model is then able to classify new input as either drowsy or alert based on these learned features.

However, there are also some challenges associated with CNN-based approaches for drowsiness detection. One key challenge is the need for large amounts of annotated data to train the model effectively. Collecting and annotating such data can be time-consuming and expensive, particularly if the data needs to be collected in real-world scenarios. Additionally, CNN-based models can be computationally intensive, which can limit their use in resource-constrained environments.

Despite these challenges, several studies have demonstrated promising results for CNN-based approaches to drowsiness detection. For example, a recent study used a CNN to detect drowsiness from eye movement data with an accuracy of 92.5%. Another study used a CNN to detect drowsiness from images of drivers' faces, achieving an accuracy of 86.8%. These results suggest that CNN-based approaches have the potential to improve safety in various applications by accurately detecting drowsiness in real-time.

In conclusion, CNN-based approaches have shown promising results for drowsiness detection in recent years. While there are some challenges associated with these approaches, their ability to automatically learn relevant features from the input data makes them well-suited for real-time applications. With continued research and development, CNN-based approaches to drowsiness detection have the potential to improve safety in a variety of fields.

Accuracy and loss for 50 Epochs



```
9533
Epoch 45/50
43/43 [==============================] - 146s 3s/step - loss: 0.0900 - accuracy: 0.9636 - val_loss: 0.1229 - val_accuracy: 0.
9533
Epoch 46/50
43/43 [==============================] - 151s 4s/step - loss: 0.0862 - accuracy: 0.9703 - val_loss: 0.1422 - val_accuracy: 0.
9619
Epoch 47/50
43/43 [==============================] - 145s 3s/step - loss: 0.1127 - accuracy: 0.9614 - val_loss: 0.0959 - val_accuracy: 0.
9654
Epoch 48/50
43/43 [==============================] - 147s 3s/step - loss: 0.1045 - accuracy: 0.9629 - val_loss: 0.0986 - val_accuracy: 0.
9689
Epoch 49/50
43/43 [==============================] - 147s 3s/step - loss: 0.0785 - accuracy: 0.9755 - val_loss: 0.0893 - val_accuracy: 0.
9723
Epoch 50/50
43/43 [==============================] - 159s 4s/step - loss: 0.0771 - accuracy: 0.9659 - val_loss: 0.0964 - val_accuracy: 0.
9689
```

FIG 4.1 ACCURACY OF EPOCHS

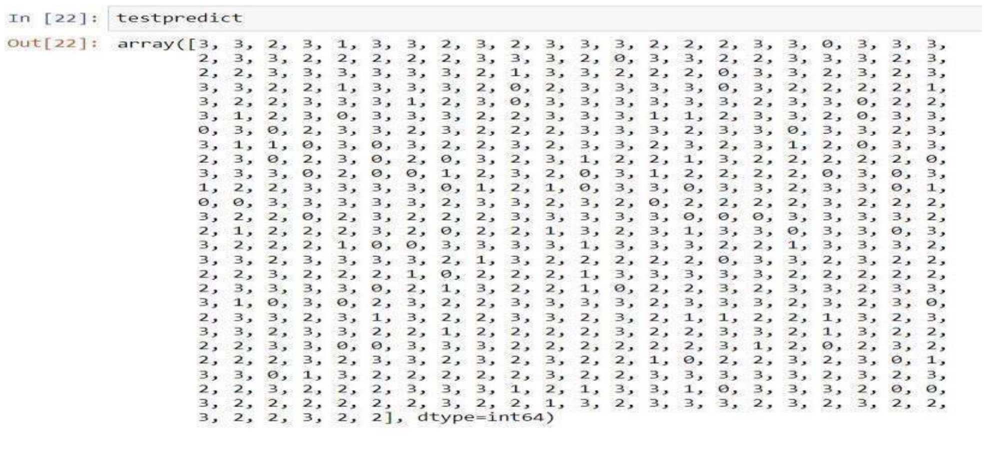
Predictions for test dataset

0 – Yawn

1 – No-yawn

2 – Closed

3 – Opened



```
In [22]: testpredict
Out[22]: array([3, 3, 2, 3, 1, 3, 3, 2, 3, 2, 3, 3, 3, 2, 2, 2, 3, 3, 0, 3, 3, 3,
                2, 3, 3, 2, 2, 2, 2, 2, 3, 3, 3, 2, 0, 3, 3, 2, 2, 3, 3, 3, 2, 3,
                2, 2, 3, 3, 3, 3, 3, 3, 2, 1, 3, 3, 2, 2, 2, 0, 3, 3, 2, 3, 2, 3,
                3, 3, 2, 2, 1, 3, 3, 3, 2, 0, 2, 3, 3, 3, 3, 0, 3, 2, 2, 2, 2, 1,
                3, 2, 2, 3, 3, 3, 1, 2, 3, 0, 3, 3, 3, 3, 3, 3, 2, 3, 3, 0, 2, 2,
                3, 1, 2, 3, 0, 3, 3, 3, 2, 2, 3, 3, 3, 1, 1, 2, 3, 3, 2, 0, 3, 3,
                0, 3, 0, 2, 3, 3, 2, 3, 2, 2, 2, 3, 3, 3, 2, 3, 3, 0, 3, 3, 2, 3,
                3, 1, 1, 0, 3, 0, 3, 2, 2, 3, 2, 3, 3, 2, 3, 2, 3, 1, 2, 0, 3, 3,
                2, 3, 0, 2, 3, 0, 2, 0, 3, 2, 3, 1, 2, 2, 1, 3, 2, 2, 2, 2, 2, 0,
                3, 3, 3, 0, 2, 0, 0, 1, 2, 3, 2, 0, 3, 1, 2, 2, 2, 2, 0, 3, 0, 3,
                1, 2, 2, 3, 3, 3, 3, 0, 1, 2, 1, 0, 3, 3, 0, 3, 3, 2, 3, 3, 0, 1,
                0, 0, 3, 3, 3, 3, 3, 2, 3, 3, 2, 3, 2, 0, 2, 2, 2, 2, 3, 2, 2, 2,
                3, 2, 2, 0, 2, 3, 2, 2, 2, 3, 3, 3, 3, 3, 0, 0, 0, 3, 3, 3, 3, 2,
                2, 1, 2, 2, 2, 3, 2, 0, 2, 2, 1, 3, 2, 3, 1, 3, 3, 0, 3, 3, 0, 3,
                3, 2, 2, 2, 1, 0, 0, 3, 3, 3, 3, 1, 3, 3, 3, 2, 2, 1, 3, 3, 3, 2,
                3, 3, 2, 3, 3, 3, 3, 2, 1, 3, 2, 2, 2, 2, 2, 0, 3, 3, 2, 3, 2, 2,
                2, 2, 3, 2, 2, 2, 1, 0, 2, 2, 2, 1, 3, 3, 3, 3, 3, 2, 2, 2, 2, 2,
                2, 3, 3, 3, 3, 0, 2, 1, 3, 2, 2, 1, 0, 2, 2, 3, 2, 3, 3, 2, 3, 3,
                3, 1, 0, 3, 0, 2, 3, 2, 2, 3, 3, 3, 3, 2, 3, 3, 3, 2, 3, 2, 3, 0,
                2, 3, 3, 2, 3, 1, 3, 2, 2, 3, 3, 2, 3, 2, 1, 1, 2, 2, 1, 3, 2, 3,
                3, 3, 2, 3, 3, 2, 2, 1, 2, 2, 2, 2, 3, 2, 2, 3, 3, 2, 1, 3, 2, 2,
                2, 2, 3, 3, 0, 0, 3, 3, 3, 2, 2, 2, 2, 2, 2, 3, 1, 2, 0, 2, 3, 2,
                2, 2, 2, 3, 2, 3, 3, 2, 3, 2, 3, 2, 2, 1, 0, 2, 2, 3, 2, 3, 0, 1,
                3, 3, 0, 1, 3, 2, 2, 2, 2, 2, 3, 2, 2, 3, 3, 3, 3, 3, 2, 3, 2, 3,
                2, 2, 3, 2, 2, 2, 3, 3, 3, 1, 2, 1, 3, 3, 1, 0, 3, 3, 3, 2, 0, 0,
                3, 2, 2, 2, 2, 2, 2, 3, 2, 2, 1, 3, 2, 3, 3, 3, 2, 3, 2, 3, 2, 2,
                3, 2, 2, 3, 2, 2], dtype=int64)
```

FIG 4.2 TESTING OF DATA SET

Graph of training and validation accuracies

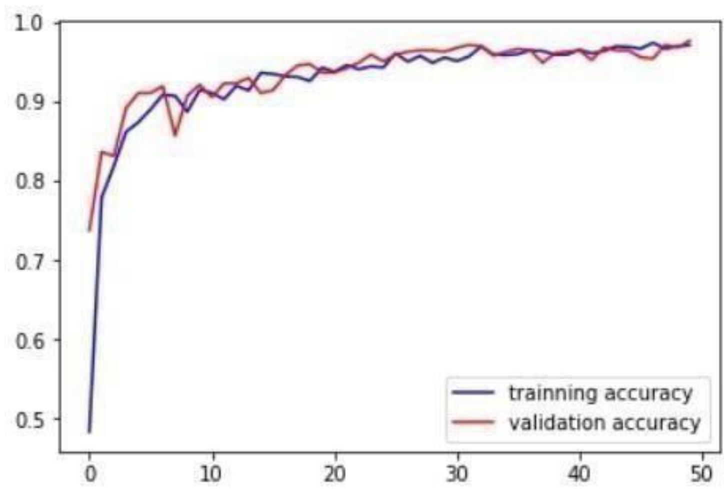


FIG 4.3 GRAPHY OF ACCURACY

Graph of training and validation losses

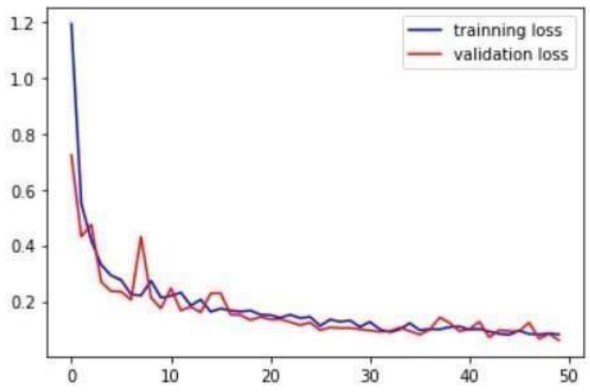


FIG 4.4 GRAPH OF LOSS

Live Detected Images

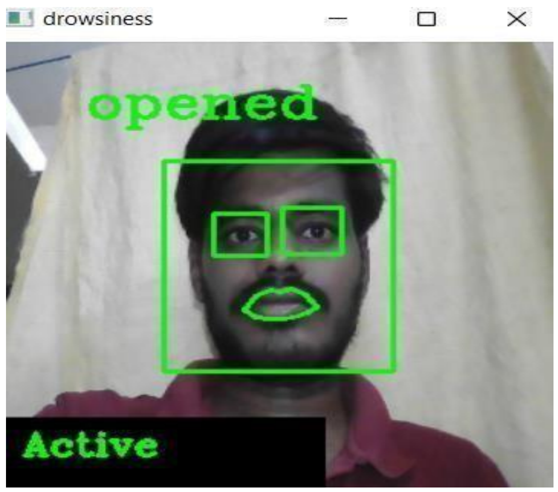


FIG 4.5 OUTPUT OF ACTIVE STATUS

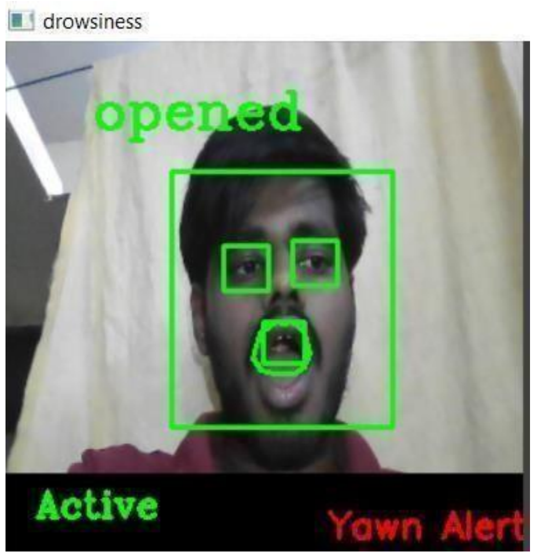


FIG 4.6 OUTPUT OF YAWN ALERT

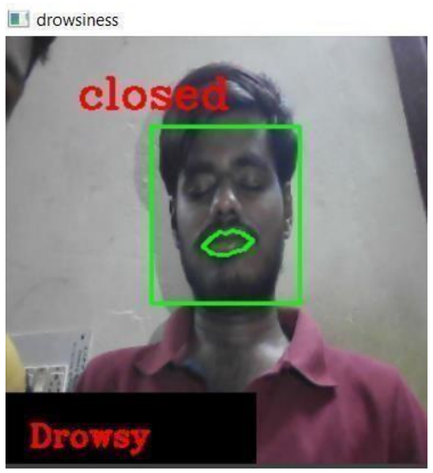


FIG 4.7 OUTPUT OF DROWSY STATUS

ACCURATE YAWNING AND EYE LASHES DETETCTION

SNO.	**Absolute Positive Rating(Eye)**	**Absolute Negative Rating(Eye)**	**Inaccurate Positive Rate(Mouth)**	**Inaccurate Negative Rating(Mouth)**
1	32.2 %	96.3 %	50.5 %	63.1 %
2	92.6 %	97.9 %	96.5 %	96.2 %
3	75.4 %	77.1 %	77.1 %	69.5 %
4	99.2 %	96.9 %	96.9 %	97.0 %
Accuracy	97.0 %	91.7 %	91.7 %	94.5 %

CHAPTER 5 CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

In conclusion, the use of Convolutional Neural Networks (CNN) for drowsiness detection has shown promising results. CNN models can effectively extract relevant features from facial images, such as eye closure and head position, to predict whether an individual is drowsy or not.

However, the performance of CNN models can be influenced by various factors such as lighting conditions, camera quality, and variability in facial expressions. To overcome these challenges, researchers have explored various techniques such as data augmentation and transfer learning.

Overall, CNN-based drowsiness detection systems have the potential to improve safety in various industries such as transportation and healthcare by providing real-time monitoring of drowsiness in individuals. Future research can focus on improving the accuracy and reliability of these systems under various environmental conditions.

5.2 FUTURE WORK

There are several potential future directions for drowsiness detection using convolutional neural networks (CNNs). Here are a few ideas:

a) Incorporating temporal information: Currently, most CNN-based approaches for drowsiness detection use a single image or frame as input. However, drowsiness is a temporal phenomenon, and it would be useful to incorporate temporal information into the model. This could be done by using multiple frames or by using a recurrent neural network (RNN) to model the temporal dynamics.

b) Exploring new CNN architectures: While CNNs have been very successful in many computer vision tasks, there may be more specialized architectures that could be better suited for drowsiness detection. For example, recent work has explored the use of attention mechanisms in CNNs, which could be used to focus on regions of the image that are most informative for drowsiness detection.

c) Developing new datasets: While there are several datasets available for drowsiness detection, they are relatively small and may not fully capture the diversity of real-world scenarios. Developing new datasets with a wider range of subjects, lighting conditions, and driving scenarios could help to improve the robustness of CNN-based drowsiness detection models.

d) Multi-modal approaches: While vision-based approaches have been successful for drowsiness detection, incorporating other modalities such as audio or physiological signals (e.g., heart rate, respiration) could improve performance. Multi-modal approaches could also help to mitigate the impact of environmental factors such as lighting conditions.

e) Real-time implementation: While many CNN-based approaches for drowsiness detection have shown promising results, they are often computationally expensive and may not be suitable for real-time implementation on resource- constrained devices. Developing more efficient models that can be implemented in real-time could enable wider adoption of drowsiness detection technology in practical applications such as driver assistance systems.

CHAPTER 6 APPENDIX

Raspberry Pi Code:

```
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
argparse import imutils import time import
dlib import cv2 import os
import serial import
RPi.GPIO as GPIO import
smbus import urllib.request


# GPIO setup
GPIO.setwarnings(False) GPIO.setmode(GPIO.BCM)  m1
= 19
ac =2 buz
=26
GPIO.setup(buz, GPIO.OUT)
GPIO.setup(m1,GPIO.OUT)
GPIO.setup(ac, GPIO.IN)
GPIO.output(buz,0) kk=0 def
GPS_Info():     global
NMEA_buff     global
lat_in_degrees     global
long_in_degrees
nmea_time = []
nmea_latitude = []
nmea_longitude = []
   nmea_time = NMEA_buff[0]                    #extract time from GPGGA string     nmea_latitude
= NMEA_buff[1]                #extract latitude from GPGGA string     nmea_longitude =
NMEA_buff[3]               #extract longitude from GPGGA string


   #print("NMEA Time: ", nmea_time,'\n')
   #print ("NMEA Latitude:", nmea_latitude,"NMEA Longitude:", nmea_longitude,'\n')     try:
      lat = float(nmea_latitude)                  #convert string into float for calculation
longi = float(nmea_longitude)               #convertr string into float for calculation     except:
lat=0         longi=0
   lat_in_degrees = convert_to_degrees(lat)    #get latitude in degree decimal format
   long_in_degrees = convert_to_degrees(longi) #get longitude in degree decimal format
 def convert_to_degrees(raw_value):     decimal_value = raw_value/100.00     degrees =
                                                                       int(decimal_value)
   mm_mmmm = (decimal_value - int(decimal_value))/0.6
position = degrees + mm_mmmm     position = "%.4f"
%(position)     return position


gpgga_info = "$GPGGA,"
ser = serial.Serial ("/dev/ttyS0")             #Open port with baud rate
GPGGA_buffer = 0
NMEA_buff = 0 lat_in_degrees
= 0 long_in_degrees = 0


##def alarm(msg):
##    global alarm_status
##    global alarm_status2
##    global saying
```

```
##
##    while alarm_status:
##        print('call')
##        s = 'espeak "' + msg + '"'
##        os.system(s)
##
##    if alarm_status2:
##        print('call')
##        saying = True
##        s = 'espeak "' + msg + '"'
##        os.system(s)
##        saying = False

def eye_aspect_ratio(eye):     A =
dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])
C = dist.euclidean(eye[0], eye[3])
ear = (A + B) / (2.0 * C)     return ear

def final_ear(shape):
    (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
leftEye = shape[lStart:lEnd]     rightEye = shape[rStart:rEnd]
leftEAR = eye_aspect_ratio(leftEye)     rightEAR =
eye_aspect_ratio(rightEye)     ear = (leftEAR + rightEAR) / 2.0
return (ear, leftEye, rightEye)

def lip_distance(shape):     top_lip
= shape[50:53]
    top_lip = np.concatenate((top_lip, shape[61:64]))     low_lip
= shape[56:59]
    low_lip = np.concatenate((low_lip, shape[65:68]))
top_mean = np.mean(top_lip, axis=0) low_mean = np.mean(low_lip,
axis=0)
    distance = abs(top_mean[1] - low_mean[1])
return distance

# Argument parser setup ap = argparse.ArgumentParser()
ap.add_argument("-w", "--webcam", type=int, default=0, help="index of webcam on system") args =
vars(ap.parse_args())

# Constants  EYE_AR_THRESH
= 0.2
EYE_AR_CONSEC_FRAMES = 1 YAWN_THRESH
= 10  alarm_status = False
alarm_status2
= False
saying = False
COUNTER = 0

print("-> Loading the predictor and detector...")  detector =
cv2.CascadeClassifier("haarcascade_frontalface_default.xml")  # Faster but less accurate predictor =
dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

print("-> Starting Video Stream")
vs = VideoStream(src=args["webcam"]).start()
# vs= VideoStream(usePiCamera=True).start()       #For Raspberry Pi
time.sleep(1.0) GPIO.output(m1,1)  ii = 0
```

```
while True:     received_data = (str)(ser.readline())                  #read NMEA
string received     GPGGA_data_available = received_data.find(gpgga_info)
if(kk==0):          lat_in_degrees=0          lat_in_degrees=0     if
(GPGGA_data_available>0):
        kk=1
        GPGGA_buffer = received_data.split("$GPGGA,",1)[1]  #store data coming after "$GPGGA," string
        NMEA_buff = (GPGGA_buffer.split(','))              #store comma separated data in buffer
GPS_Info()                                          #get time, latitude, longitude
        map_link = 'http://maps.google.com/?q=' + str(lat_in_degrees) + ',' + str(long_in_degrees)    #create link to plot
location on Google map


    map_link = 'http://maps.google.com/?q=' + str(16.5085862) + ',' + str(80.6531247)    #create link to plot location on
Google map
   # print(map_link)     print()
    aval=1-GPIO.input(ac)     print("ACCIDENT:"+
str(aval))
    time.sleep(0.2)     if(aval==1):          print('Accident
occured sending info')
        GPIO.output(m1,0)         GPIO.output(buz,1)
urllib.request.urlopen("https://api.thingspeak.com/update?api_key=43J0P0NIGILXMDDA&field1=1&field2=16.508
5862&field3=80.6531247") frame
    = vs.read()      if frame is None:
print("Error: Frame not
captured correctly.")         continue


    frame = imutils.resize(frame, width=450)      gray =
cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)


    rects = detector.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)


    for (x, y, w, h) in rects:         rect = dlib.rectangle(int(x),
int(y), int(x + w), int(y + h))


        shape = predictor(gray, rect)
shape = face_utils.shape_to_np(shape)
ear, leftEye, rightEye = final_ear(shape)
distance = lip_distance(shape)


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


        lip = shape[48:60]
        cv2.drawContours(frame, [lip], -1, (0, 255, 0), 1)


        if ear < EYE_AR_THRESH:
            COUNTER += 1


            if COUNTER >= EYE_AR_CONSEC_FRAMES:
                #if not alarm_status:
                   # alarm_status = True
                   # t = Thread(target=alarm, args=('Alert and wake up sir',))
```

```
                    GPIO.output(buz, 1)  ##
t.daemon = True
##                  t.start()                   ii += 1
if ii > 4:                        print('drowsiness detected
sending info')
                        GPIO.output(m1,0)
GPIO.output(buz,1)
urllib.request.urlopen("https://api.thingspeak.com/update?api_key=43J0P0NIGILXMDDA&field1=2&field
2=16.5085862&field3=80.6531247")
                         while
True:
                            time.sleep(1)


            cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),                            cv2.FONT_HERSHEY_SIMPLEX,
0.7, (0, 0, 255), 2)


else:
            COUNTER = 0
alarm_status = False


##          if distance > YAWN_THRESH:
##              cv2.putText(frame, "Yawn Alert", (10, 30),
##                          cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2) ##            if
not alarm_status2 and not saying:
##                  alarm_status2 = True
##                  t = Thread(target=alarm, args=('Alert and take some fresh air sir',))
##                  t.daemon = True
##                  t.start()
##                  ii += 1 ##               if
ii > 2:
##                      GPIO.output(m1, 0)
##                      while True: ##                      time.sleep(1)
##          else:
##              alarm_status2 = False
##
##          cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
##                      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
##          cv2.putText(frame, "YAWN: {:.2f}".format(distance), (300, 60),
##                      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)


    cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF


    if key == ord("q"):
break


cv2.destroyAllWindows()
vs.stop()
```

REFERENCES

[1] Tiwari, K.S., Bhagat, S., Patil, N. and Nagare, P., 2019. IOT based driver drowsiness detection and health Monitoring System. *International Journal of Research and Analytical Reviews (IJRAR)*, *6*(02), pp.163-167.

[2] Jabbar, R., Shinoy, M., Kharbeche, M., Al-Khalifa, K., Krichen, M. and Barkaoui, K., 2020, February. Driver drowsiness detection model using convolutional neural networks techniques for android application. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)* (pp. 237- 242). IEEE.

[3] Hossain, M.Y. and George, F.P., 2018, October. IOT based real-time drowsy driving detection system for the prevention of road accidents. In *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)* (Vol. 3, pp. 190-195). IEEE.

[4] Sunagawa, M., Shikii, S.I., Nakai, W., Mochizuki, M., Kusukame, K. and Kitajima, H., 2019. Comprehensive drowsiness level detection model combining multimodal information. *IEEE Sensors Journal*, *20*(7), pp.37093717.

[5] Alkinani, M.H., Khan, W.Z. and Arshad, Q., 2020. Detecting human driver inattentive and aggressive driving behavior using deep learning: Recent advances, requirements and open challenges. *Ieee Access*, *8*, pp.105008-105030.

[6] Ngxande, M., Tapamo, J.R. and Burke, M., 2017. Driver drowsiness detection using behavioral measures and machine learning techniques: A review of state- of-art techniques. *2017 pattern recognition Association of South Africa and Robotics and mechatronics (PRASA-RobMech)*, pp.156-161.

[7] Paulo, J.R., Pires, G. and Nunes, U.J., 2021. Cross-subject zero calibration driver’s drowsiness detection: Exploring spatiotemporal image encoding of EEG signals for convolutional neural network classification. *IEEE transactions on neural systems and rehabilitation engineering*, *29*, pp.905-915.

[8] Bala, U.C. and Sarath, T.V., 2020, June. Internet of things based intelligent drowsiness alert system. In *2020 5th international conference on communication and electronics systems (ICCES)* (pp. 594-598). IEEE.

[9] Karuppusamy, N.S. and Kang, B.Y., 2020. Multimodal system to detect driver fatigue using EEG, gyroscope, and image processing. *IEEE Access*, *8*, pp.129645-129667.

[10] Hu, Y., Lu, M., Xie, C. and Lu, X., 2019. Driver drowsiness recognition via 3D conditional GAN and two-level attention Bi-LSTM. *IEEE Transactions on Circuits and Systems for Video Technology*,
30(12), pp.4755-4768

BIODATA

Name: Vishal Kumar Singh Mob-8178348121

E-Mail: vishal.21bce8647@vitapstudent.ac.in

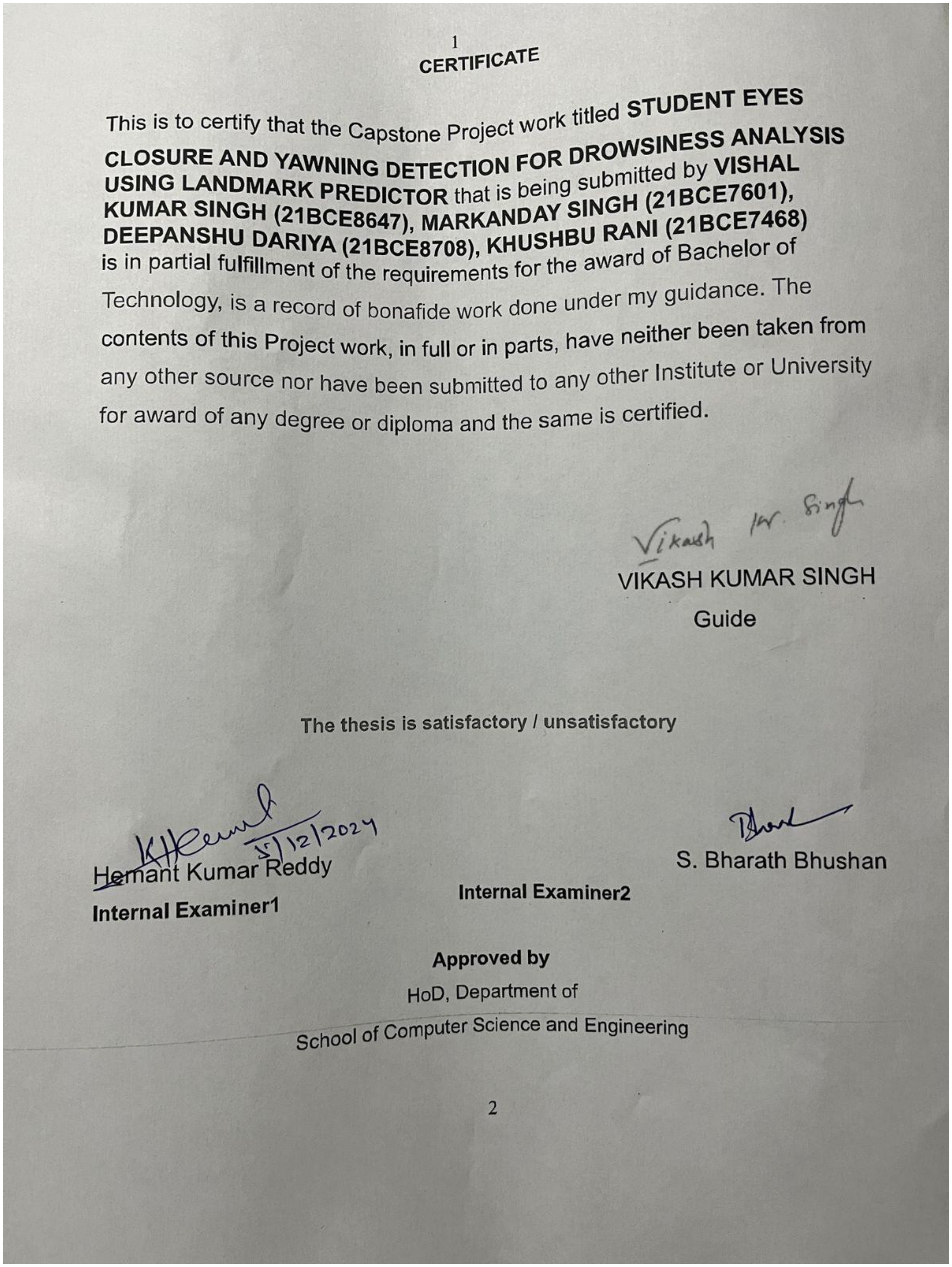
Name: Markanday Singh Mob-9897968286

E-Mail: markanday.21bce7601@vitapstudent.ac.in

Name: Deepanshu Dariya Mob-8905313543

E-Mail:
deepanshu.21bce8708@vitapstudent.ac.in

Name: Khusbu Rani Mob-8307734061 E-Mail:
khusbu.21bce7468@vitapstudent.ac.in



1

CERTIFICATE

This is to certify that the Capstone Project work titled **STUDENT EYES CLOSURE AND YAWNING DETECTION FOR DROWSINESS ANALYSIS USING LANDMARK PREDICTOR** that is being submitted by **VISHAL KUMAR SINGH (21BCE8647), MARKANDAY SINGH (21BCE7601), DEEPANSHU DARIYA (21BCE8708), KHUSHBU RANI (21BCE7468)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Vikash kr. Singh

VIKASH KUMAR SINGH

Guide

The thesis is satisfactory / unsatisfactory

[illegible] 5/12/2024

Hemant Kumar Reddy

Internal Examiner1

[illegible]

S. Bharath Bhushan

Internal Examiner2

Approved by

HoD, Department of

School of Computer Science and Engineering

2