DEPARTMENT OF TECHNICAL EDUCATION,AP

GOVERNMENT POLYTECHNIC FOR WOMEN,

BHEEMUNIPATNAM

DEPARTMENT OF COMPUTER ENGINEERING

PROJECT REPORT ON
REAL TIME DROWSINESS DETECTION SYSTEM

A project report submitted to the Government Polytechnic for Women Bheemunipatnam, in the partial fulfilment for the award of

DIPLOMA IN COMPUTER ENGINEERING

Submitted by

P. LAKSHMI	**21045-CM-043**
V.VEERA VARSHINI	**21045-CM-058**
T. HEMALATHA	**21045-CM-055**
S. MANI	**21045-CM-047**
B. NIKITHA	**21045-CM-006**
G. BHAVANA	**21045-CM-018**
S. MADHU MEEKA	**21045-CM-049**
L. DEEPIKA	**21045-CM-035**
B. MANASA	**21045-CM-002**
M.GAYATHRI	**21045-CM-036**
A. ROHITHA	**21045-CM-064**
U. JAHNAVI	**21045-CM-057**
K. SUSHMITHA	**21045-CM-024**

Under the guidance of

Mr. G. VENUGOPAL RAO

(SENIOR LECTURER IN DCME)

DEPARTMENT OF COMPUTER ENGINEERING,

GOVT. POLYTECHNIC FOR WOMEN BHEEMUNIPATNAM,

BHEEMUNIPATNAM, VISAKHAPATAM

ANDHRA PRADESH, INDIA

2023-2024

DEPARTMENT OF COMPUTER ENGINEERING

GOVT POLYTECHNIC FOR WOMEN BHEEMUNIPATNAM-531163

(2021-2024)

CERTIFICATE

This is to certify that the project report entitled "**REAL TIME DRIVER DROWSINESS DETECTION SYSTEM"** is a bonified project work done by:

P. LAKSHMI	**21045-CM-043**
V. VEERA VARSHINI	**21045-CM-058**
T. HEMALATHA	**21045-CM-055**
S. MANI	**21045-CM-047**
B. NIKITHA	**21045-CM-006**
G. BHAVANA	**21045-CM-018**
S. MADHU MEEKA	**21045-CM-049**
L. DEEPIKA	**21045-CM-035**
B. MANASA	**21045-CM-002**
M.GAYATHRI	**21045-CM-036**
A. ROHITHA	**21045-CM-064**
U. JAHNAVI	**21045-CM-057**
K. SUSHMITHA	**21045-CM-024**

For the partial fulfilment of the requirements for the award of the degree of **DIPLOMA IN COMPUTER ENGINEERING GOVERNMENT POLYTECHNIC FOR WOMEN BHEEMUNIPATNAM.**

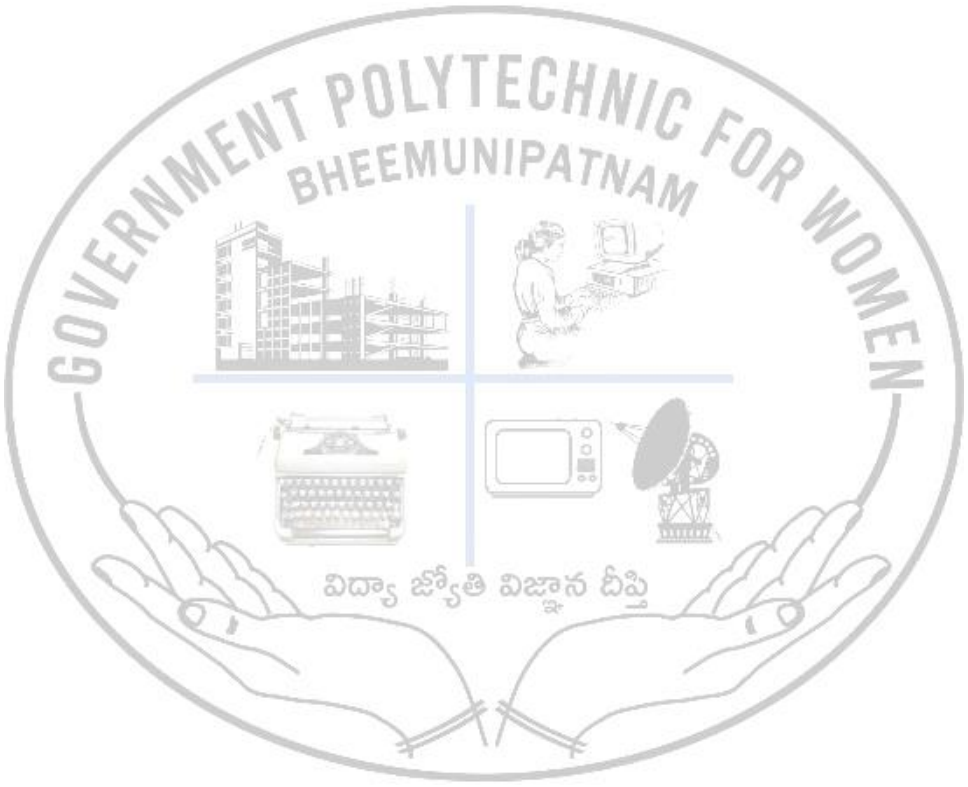
Signature of the project guide
MR. G. VENU GOPALA RAO
Senior Lecturer in CME
Govt Polytechnic for Women
Bheemunipatnam
Visakhapatnam

Signature of the HOD
Mrs. U. PRAMEELA, M. Tech
Head of the Department CME
Govt Polytechnic for Women
Bheemunipatnam
Visakhapatnam

DECLARATION

We, the students under the guidance of **Mr. G. VENUGOPALRAO** , hereby declare that the project entitled **"REAL TIME DRIVER DROWSINESS DETECTION SYSTEM"** is an original work done at **Government Polytechnic of Women Bheemunipatnam,** submitted in partial fulfilment of the requirements for the award of 5th Semester project work.I assure that this project is not submitted in any university or college.

Place: **BHEEMUNIPATNAM**



ACKNOWLEDGEMENT

First, I convey my sincere thanks to my project guide **Mr. G. VENU GOPAL RAO ,** for his valuable guidance throughout the project. His emphasis on the need to thoroughly understand the subject and her zeal for perfection have taught me invaluable lessons. I thank him for the freedom given by him in choosing a topic close to my passion of web designing and the numerousinsights offered, not just on the project, but on other key aspects of communicating ideas and how to think with research bent of mind. My thinking and research have been shaped by thoughtful discussions and interactions with him.

I thank **U. PRAMEELA**, Head of Computer Engineering, who has encouraged us a lot and also for her extremely valuable guidance and advice.

I would also like to convey my gratitude to Dr. **CH. MURALI KRISHNA**, Principal, GPTW, for his valuable advice and timely help.

I also extend my gratitude to all the teaching and non-teaching staff of the Department of Computer Engineering for their support in completing the project.I also thank my friends and family members who supported me for the completion of the project

With Gratitude,

P. LAKSHMI	21045-CM-043
V. VEERA VARSHINI	21045-CM-058
T. HEMALATHA	21045-CM-055
S. MANI	21045-CM-047
B. NIKITHA	21045-CM-006
G. BHAVANA	21045-CM-018
S. MADHU MEEKA	21045-CM-049
L. DEEPIKA	21045-CM-035
B. MANASA	21045-CM-002
M.GAYATHRI	21045-CM-036
A. ROHITHA	21045-CM-064
U. JAHNAVI	21045-CM-057
K. SUSHMITHA	21045-CM-024

INDEX

ABSTRACT

The most common type accident in today's world is the accident occurring due to the sleepiness of the driver irrespective of day and night. The death rate of accidents due to this has spiked to 21% over the world. This shows how serious this problem is. The Drowsiness Detection is a safe technology that can prevent accidents that are caused by drivers who fall asleep while driving. The objective of this python project is to build a Drowsiness Detection Model which will detect that a driver's eyes are closed for a few seconds. The implementation of this project uses a pre-built model of face landmark for easy deployment on edge of computationally less efficient devices. The project has a direct application in the automotive sector. This paper is aimed at designing a Drowsiness Detection Model, which takes the driver's eyes as ROI (Region of Interest) and continuously senses (in real-time) the eye lid to detect whether the driver is feeling sleepy or not. If the driver feels sleepy, the model will generate Sound Alarm to bring the driver back to his/her conscious state. This model is also effective even when the driver is wearing a spectacle. The number of accidents show just how much grave this matter is and that's why we chose to develop this project that is intended to reduce these accidents.

This Project has been done by using:

1.Python

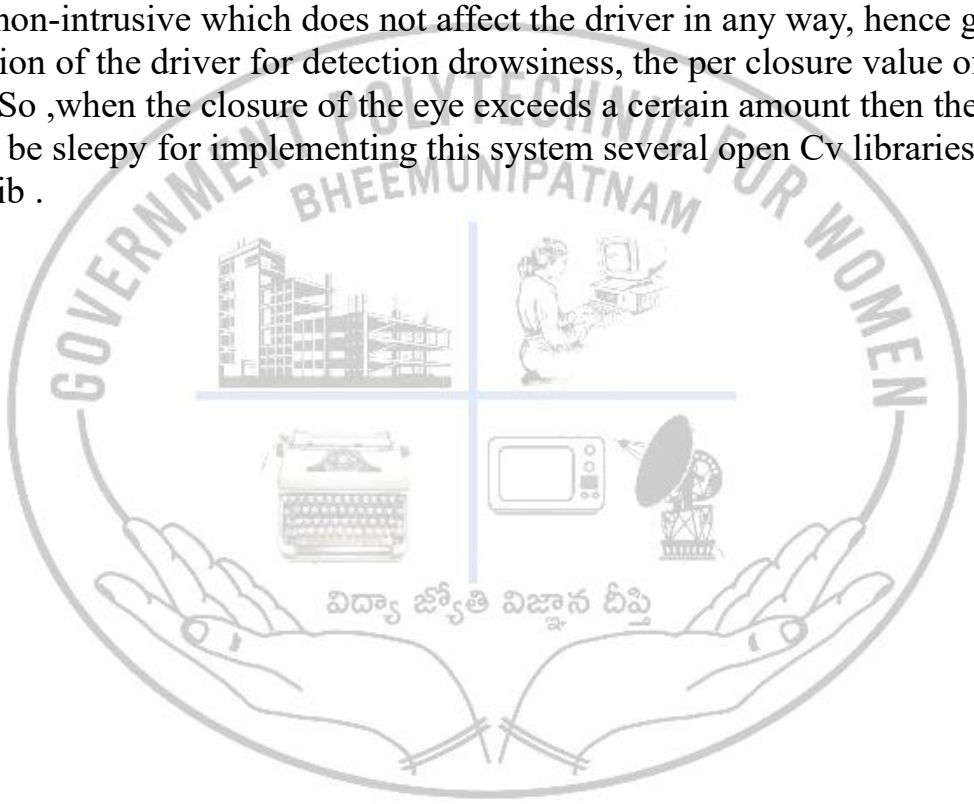
2.Open CV

3.Machine Learning

4.DLib

INTRODUCTION

This project as title "**Real Time Driver Drowsiness Detection System"** is comes under the **computer vision system.** This application is developed with the help of python and OpenCV. This application is also based on image processing methodologies so it can also be called "Image Processing application". In recent years driver fatigue is one of the major cause of the accidents in the world. The direct way of measuring driver fatigue is measuring the state of the driver i.e drowsiness so it is very important to detect the drowsiness of the driver to save life and property this project is aimed towards developing a prototype of a drowsiness detection system. This system is a real time system that captures image continuously and measures the state of the eye according to the specified algorithm and gives a warning if required. Though there are several methods for measuring the drowsiness this approach is completely non-intrusive which does not affect the driver in any way, hence giving the exact condition of the driver for detection drowsiness, the per closure value of the eye is considered. So ,when the closure of the eye exceeds a certain amount then the driver is identified to be sleepy for implementing this system several open Cv libraries are used including dlib .



OBJECTIVES

The objective of this project is to develop a model of detection of drowsiness system and alarm alert.

The total concentration and focus will be placed on designing a drowsiness detection system that will correctly monitor real time state of open and closed eye of driver. By monitoring the eyes constantly, it can be observed that the driver symptoms fatigue can be detected early to avoid an accident.

This project outlines the design and development of a system that focuses on driver's drowsiness detection and prediction.

1. Monitoring the driver behaviour by observing the manoeuvre stability and performance.

2. Validate and measure the progress by using Specific algorithm.

3. Warning the drivers if the behaviour beyond the thresholds.

Here we will employ machine learning methods to classify the data of actual human behaviour during drowsiness.

Devices to detect when drivers are trying to sleep and to provide alert warnings to them of the risk.

Drowsiness driving has become a serious concern where even researches are unable to decide which factor lead to the accident. Hence the main objective of the proposed system is to detect drowsiness of the driver using image processing techniques to detect open and closed eye.

TOOLS & PLATFORMS

PYTHON INTRODUCTION

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming. Python has become a staple in data science, allowing data analysts and other professionals to use the language to conduct complex statistical calculations, create data visualizations, build machine learning algorithms, manipulate and analyse data, detection of driver drowsiness and complete other data-related tasks. Python also has a number of libraries that enable coders to write programs for data analysis and machine learning more quickly and efficiently, like TensorFlow.

FEATURES OF PYTHON:

- Easy to learn and readable language
- Object oriented programming language
- Dynamically typed language
- Large standard library
- Large community support

ADVANTAGES:

- It can quickly detect drowsiness.
- The system can distinguish between normal eye blinks and drowsy eye blinks.
- It can operate in low-light conditions and while the driver is wearing spectacles.

USE OF PYTHON IN PROJECT:

Thus, python allows the model of deep learning algorithm via including the use of OpenCV. Drowsiness Detection is the detection of a person to check whether the person is feeling sleepy while performing a significant task. This detection has many applications in medical and safety fields. Installing and importing all the modules needed for the code. pip install NumPy, pip install open cv -python, pip install time, pip install pyttsx3, pip install Dlib, pip install SciPy. Access the camera and mark the landmarks from the (.Dat) file to predict the location of the eyes. Mark the eye points in a face so that it will be really easy for the user to get the detection.

OPENCV INTRODUCTION

OpenCV is a library of programming functions mainly for real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage, then ItSeez. The library is cross-platform and licensed as free and open-source software under Apache License. The proposed OpenCV algorithms effectively find and help to normalize human faces while causing the majority of accidents related to vehicle crashes. The algorithm begins by detecting heads on colour images using colour and structure deviations in the human face and background. Several faces and body gestures. The user has to register their account and log in using a username and password. Using Open CV, the system will detect eye closure or yawning actions in real-time. If it finds any, it will draw a red rectangle and add a log to the table.

FEATURES OF OPENCV LIBRARY:

- Using OpenCV library, you can −
- Read and write images
- Capture and save videos
- Process images (filter, transform)
- Perform feature detection
- Detect specific objects such as faces, eyes, cars, in the videos or images.

USAGE OF OPENCV IN PROJECT:

OpenCV is a Python library that allows you to perform image processing and computer vision tasks. It provides a wide range of features, including object detection, face recognition, and tracking. It is also an open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. It will be using OpenCV for gathering the images from webcam and feed them into a Deep Learning model which will classify whether the person's eyes are 'Open' or 'Closed'.

MACHINE LEARNING INTRODUCTION

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more. Machine learning techniques such as Supervised, Unsupervised, and Reinforcement learning. You will learn about regression and classification models, clustering methods, hidden Markov models, and various sequential models.

NEED OF MACHINE LEARNING:

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.

FEATURES OF ML:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

USAGE OF MACHINE LEARNING IN PROJECT:

Machine learning is used in driver drowsiness detection by analyzing facial expressions and movements to identity signs of Drowsiness. Machine learning algorithms are trained on large datasets of facial expressions and movements to recognise patterns associated with Drowsiness. These algorithms can then analyse real-time video feeds from cameras in vehicles to detect signs of driver fatigue.

DLIB INTRODUCTION

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. Dlibs opensource licensing allows you to use it in any application, free of charge.

FEATURES OF DLIB:

- Documentation
- Unlike a lot of opensource 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.
- High Quality Portable Code
- Good unit test coverage. The ratio of unit test lines of code to library lines of code is about 1 to 4.
- The library is tested regularly on MS Windows, Linux, and Mac OS X systems.
- Machine Learning Algorithms
- Deep Learning
- Conventional SMO based Support Vector Machines for classification and regression.
- Numerical Algorithms
- A fast matrix object implemented using the expression templates technique and capable of using BLAS and LAPACK libraries when available.
- Graphical Model Inference Algorithms
- Join tree algorithm for exact inference in a Bayesian network.

USAGE OF DLIB IN PROJECT:

Dlib is a popular library for computer vision tasks, including driver drowsiness detection. It provides tools for facial landmark detection, which can be used to track facial features and detect signs of drowsiness. identify signs of drowsiness based on facial expressions and movements.In driver drowsiness detection, DLib is used to analyse facial landmarks like eye movements and head positions to determine if the driver is getting drowsy. It's a helpful tool for ensuring road safety!

ALGORITHMS

1.HAARCASCADE ALGORITHM:

HaarCascade classifier is another object detection algorithm which was proposed by Paul Viola and Michael Jones. It is a machine learning based model where a rich dataset of positive and negative images is fed to the classifier for it to be trained. Positive images are the objects of interest which our classifier is trained to detect and negative images are images of without the object of interest. HaarCascade classifier in comparison to other object detection techniques requires a significant amount of time to process an input image and give a desired output. Considering the processing time taken ideally the RAM consumption is on the higher side when compared to other methods like LBP, CNN. The output of the HaarCascade classifier is very accurate which makes it a widely preferred method for object detection. This model takes lesser time to be trained compared to the CNN model.

The technique used to identify a drowsy driver's eyes is HaarCasacades. HaarCasacades is an object detection algorithm that helps in the detection of objects from a picture, a video, or from a live stream. Open CV library has an inbuilt class that helps us use the HaarCascade feature. This feature is very widely used in the Face detection which can be used in the detection of any object if properly qualified.

This approach processes the image in certain stages-

Haar Feature Selection

This section involves the extraction of a region of interest or the selection of the object to be detected in other images. This is done by giving a large number of positive Images (an interest object) and negative images (images without an interest object) to train the model accordingly. It consists of a rectangle on the region of interest and also an adjacent rectangle.

Creation of integral images

This process is to boost the process of selecting the Region of Interest.

Adaboost Training

This section is responsible for selecting the required characteristics from all the positive and negative images provided by the user. It selects the best features and trains the model. In this stage, it also groups the objects to be detected in “strong classifiers” and the unwanted parts as a “weak classifier”. The output of this is stored as an XML file.

Cascade classifier

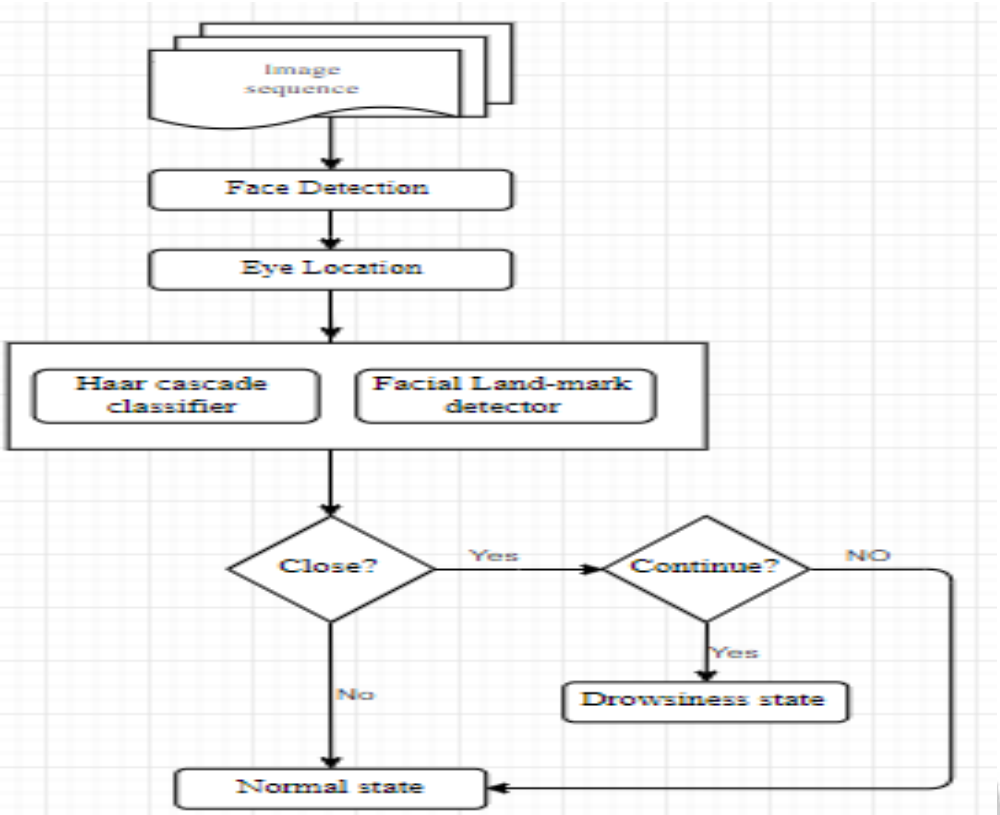
This is the final stage where it consists of many stages where each stage is made to slide over the positive and negative images. The classifier slides over a particular location and if the object is found it returns a true and indicates that the required object was found else when it returns a false it indicates that no object was detected and jumps to the next location and continues this process.

2. SVM ALGORITHM:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

In SVM-based real-time drowsiness detection, various tools and libraries can be used for implementation. Some commonly used tools include OpenCV, dlib, and scikit-learn. These tools provide functionalities for face detection, feature extraction, and SVM model training and prediction. They offer a convenient way to develop real-time drowsiness detection systems using SVM algorithms.

SYSTEM DESIGN



1.Face Detection: For the face Detection it uses Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle. Fig. 3.4 represents five haar like features & example is shown in Fig.3.5

Fig 3.4

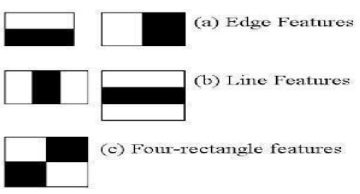


Fig 3.5

Eye detection: In the system we have used facial landmark prediction for eye detection Facial landmarks are used to localize and represent salient regions of the face, such as:

- Eyes
- Eyebrows
- Nose
- Mouth
- Jawline

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. In the context of facial landmarks, our goal is detecting important facial structures on the face using shape prediction methods. Detecting facial landmarks is therefore a twostep process:

- Localize the face in the image.
- Detect the key facial structures on the face ROI.

Localize the face in the image: The face image is localized by Haar feature-based cascade classifiers which was discussed in the first step of our algorithm i.e. face detection.

Detect the key facial structures on the face ROI: There are a variety of facial landmark detectors, but all methods essentially try to localize and label the following facial regions:

- Mouth
- Right eyebrow
- Left eyebrow
- Right eye
- Left eye
- Nose

The facial landmark detector included in the dlib library is an implementation of the One Millisecond Face Alignment with an

Ensemble of Regression Trees paper by Kazemi andSullivan (2014).

This method starts by using:

1. A training set of labelled facial landmarks on an image. These images are manually labelled, specifying specific (x, y)-coordinates of regions surrounding each facial structure.
2. Priors, of more specifically, the probability on distance between pairs of input pixels.

The pre-trained facial landmark detector inside the dlib library is used to estimate the locationof 68 (x, y)-coordinates that map to facial structures on the face.
The indexes of the 68 coordinates can be visualized on the image below:

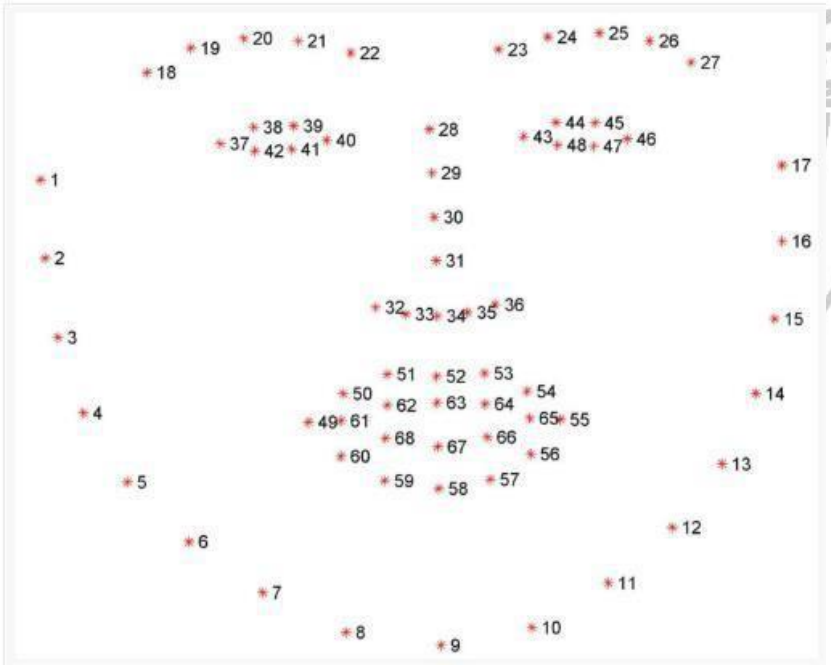


Fig.3.6: Visualizing the 68 facial landmark coordinates

We can detect and access both the eye region by the following facial landmark index showbelow

• The right eye using [37, 42].
3. The left eye with [43, 48].
4. Recognition of Eye's State:

The eye area can be estimated from optical flow, by sparse tracking or by frame-to-frame intensity differencing and adaptive thresholding. And Finally, a decision is made whether theeyes are or are not covered by eyelids. A different approach is to infer the state of the eye opening from a single image, as e.g. by correlation matching with open and closed eye templates, a heuristic horizontal or vertical image intensity projection over the eye region, a parametric model fitting to find the eyelids, or active shape models. A major drawback of the previous approaches is that they usually implicitly impose too strong requirements on the setup, in the sense of a relative face-camera pose (head orientation), image resolution, illumination, motion dynamics, etc. Especially the heuristic methods that use raw image intensity are likely to be very sensitive despite their real-time performance.

Therefore, we propose a simple but efficient algorithm to detect eye blinks by using a

recentfacial landmark detector. A single scalar quantity that reflects a level of the eye opening is derived from the landmarks. Finally, having a per-frame sequence of the eye-opening estimates, the eye blinks are found by an SVM classifier that is trained on examples of blinking and non-blinking patterns.

Eye Aspect Ratio Calculation:

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

$$EAR = \frac{||p2 - p6|| + ||p3 - p5||}{2||p1 - p4||} \quad (1)$$

where p1, . . ., p6 are the 2D landmark locations, depicted in Fig. 1. 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. Aspect ratio of the open eye has a small variance among individuals, and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged.

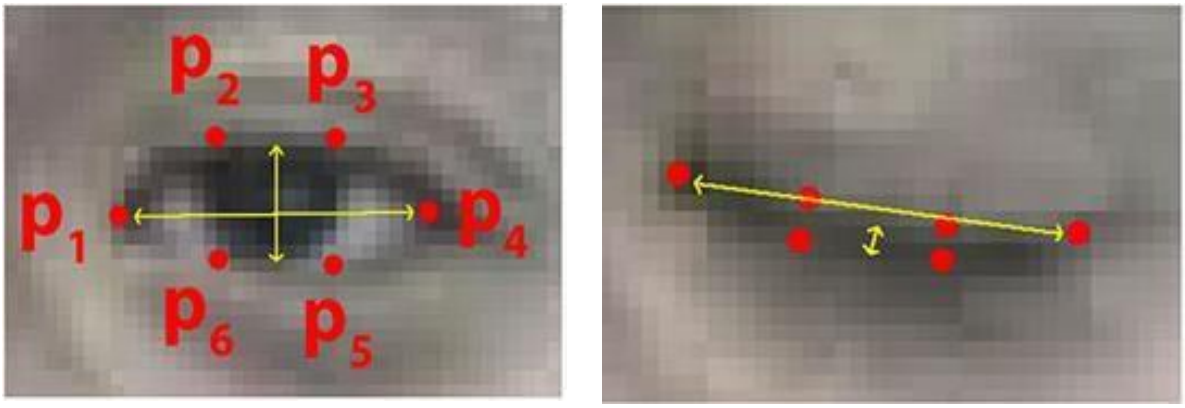


Fig 3.8: Open and closed eyes with landmarks p(i) automatically detected. The eye aspect ratio EAR in Eq. (1) plotted for several frames of a video sequence.

Fig.3.9 EAR for single blink

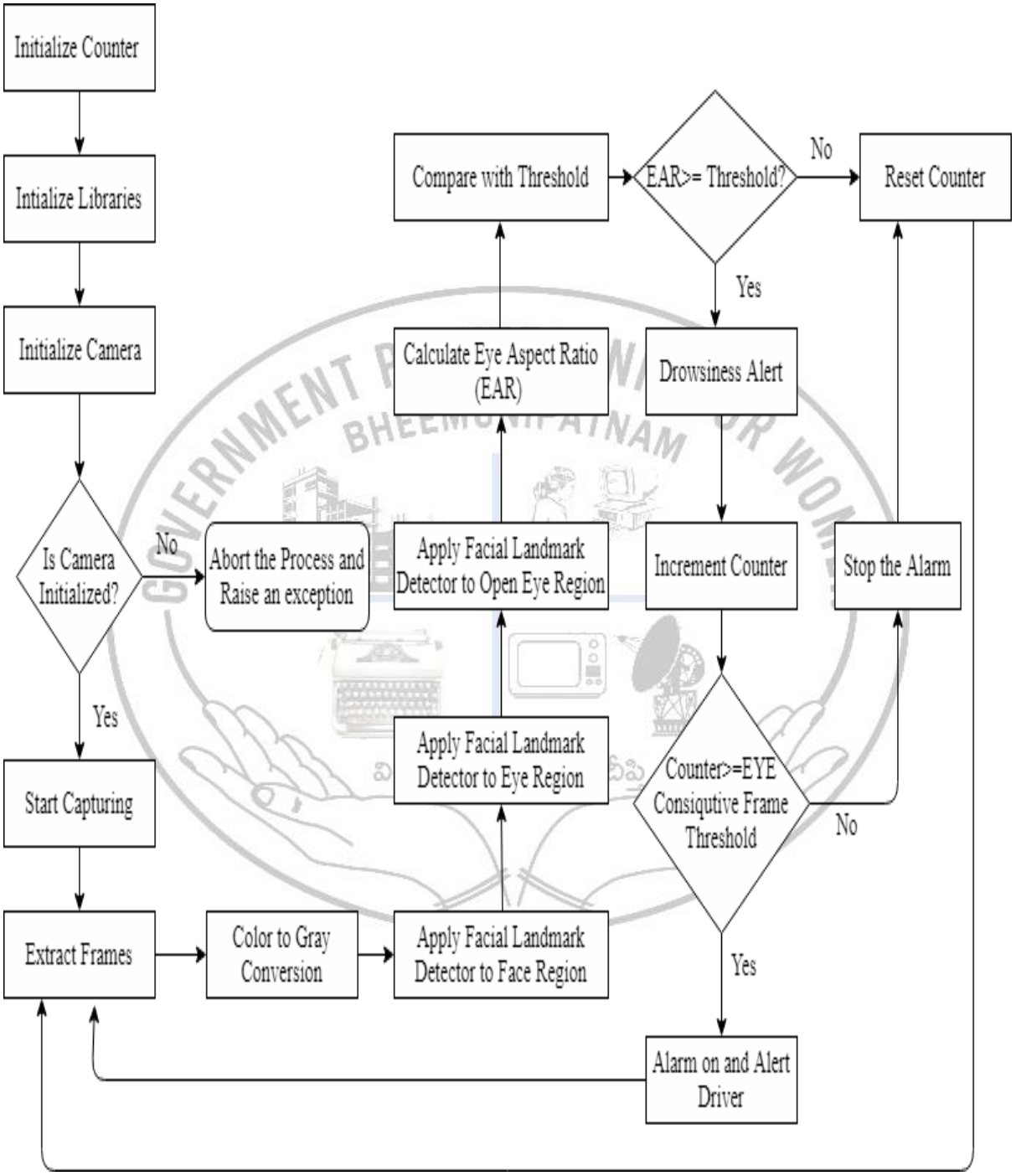
5. Eye State Determination:

Finally, the decision for the eye state is made based on EAR calculated in the previous step.If the distance is zero or is close to zero, the eye state is classified as "closed" otherwise theeye state is identified as "open".

6. Drowsiness Detection:

The last step of the algorithm is to determine the person's condition based on a pre-set condition for drowsiness. The average blink duration of a person is 100-400 milliseconds (i.e.0.1-0.4 of a second). Hence if a person is drowsy his eye closure must be beyond this interval. We set a time frame of 5 seconds. If the eyes remain closed for five or more seconds, drowsiness is detected and alert pop regarding this is triggered.

MODELLING DIAGRAM



MODULES USED IN THE PROJECT

WEBCAM: In a real-time drowsiness detection system, a webcam can be used to capture the driver's face and monitor their eye movements and facial expressions. The webcam continuously captures video frames of the driver's face, which are then processed to analyse various features like eye closure, blinking patterns, and head movements. By using computer vision techniques and machine learning algorithms, the system can detect signs of drowsiness based on these features. For example, if the system detects prolonged eye closure or frequent eye blinking, it can determine that the driver may be getting drowsy and issue an alert to prevent accidents. The webcam plays a crucial role in providing the visual input needed for accurate and real-time drowsiness detection.

DETECT DROWSY: This module includes the following things: image processing, face detection, eye region detection, detect closeness of eye. By tracking features like eye closure, blinking patterns, and head movements, the system can detect signs of drowsiness. Computer vision techniques and machine learning algorithms are used to analyse these features and determine if the driver is getting drowsy. If drowsiness is detected, the system can issue an alert to prevent accidents. The webcam is an essential component in providing the visual input needed for accurate and real-time drowsiness detection.

IMAGE PROCESSING : In a real-time drowsiness detection system, image processing techniques are used to analyze the video frames captured by the webcam. These techniques involve extracting relevant features from the images, such as facial landmarks and eye movements. By tracking these features over time, the system can detect patterns associated with drowsiness, such as prolonged eye closure or changes in facial expressions. Machine learning algorithms are often employed to classify these patterns and determine if the driver is drowsy. The real-time nature of the system allows for immediate detection and timely alerts to prevent accidents.

FACE DETECTION: The face detection technic is used to locate the face from the image. The video, it is a system that uses computer vision algorithms to monitor a person's facial features, such as their eyes and head movements, to determine if they are showing signs of drowsiness or fatigue. It can be used in various applications, such as driver monitoring systems in cars or monitoring operators in critical industries. The system typically uses

a camera to capture the person's face and then analyzes the images or video in real-time to detect specific patterns or behaviors associated with drowsiness, such as eye closure or head nodding. When drowsiness is detected, the system can trigger alerts or notifications to help prevent accidents or improve safety. It's a fascinating technology with great potential.

EYE DETECTION: Eye detection plays a crucial role. The system uses computer vision algorithms to identify and track the position and movements of the eyes. By analyzing factors such as eye closure, blinking patterns, and eye movement, the system can determine if a person's eyes are showing signs of drowsiness or fatigue. This information helps in detecting when a person is becoming drowsy and can trigger alerts or notifications to prevent accidents or improve safety. It's an important component of the system that helps in accurately identifying drowsiness.

DETECT CLOSENESS OF EYE: It also involves detecting the closeness of the eyes. By analyzing the distance between the person's eyes, the system can determine if the eyes are closing or if they are open wide enough. This information helps in identifying drowsiness or fatigue. If the eyes are frequently closing or the distance between them decreases, it can be a sign that the person is becoming drowsy. The system can then trigger alerts or notifications to prevent accidents. It's an important aspect of the system to ensure accurate detection drowsiness.

OUTPUT: The system will provide alerts or notifications when it detects signs of drowsiness or fatigue, such as frequent eye closure, head nodding, or other indicators. These alerts can be in the form of visual cues, auditory signals, or even vibrations to alert the person and help them stay awake and focused. The system's output is designed to prevent accidents and improve safety by proactively addressing drowsiness.

ALERT: The system continuously checks whether the driver is feeling drowsy or not. Real-time drowsiness detection systems use alerts like flashing lights, beeping sounds, or vibrations to keep you awake and prevent accidents.

CODE:

```
#python drowniness_yawn.py --webcam webcam_index


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 playsound
import os


def sound_alarm(path):

    global alarm_status
    global alarm_status2
    global saying

    while alarm_status:
        print('call')
```

```
        playsound.playsound("C:/Users/user/Downloads/Real-Time-Drowsiness-Detection-System-main/Real-Time-Drowsiness-Detection-System-main/Alert.wav")


    if alarm_status2:

        print('call')

        saying = True

        playsound.playsound("C:/Users/user/Downloads/Real-Time-Drowsiness-Detection-System-main/Real-Time-Drowsiness-Detection-System-main/Alert.wav")


        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

ap = argparse.ArgumentParser()
ap.add_argument("-w", "--webcam", type=int, default=0,
```

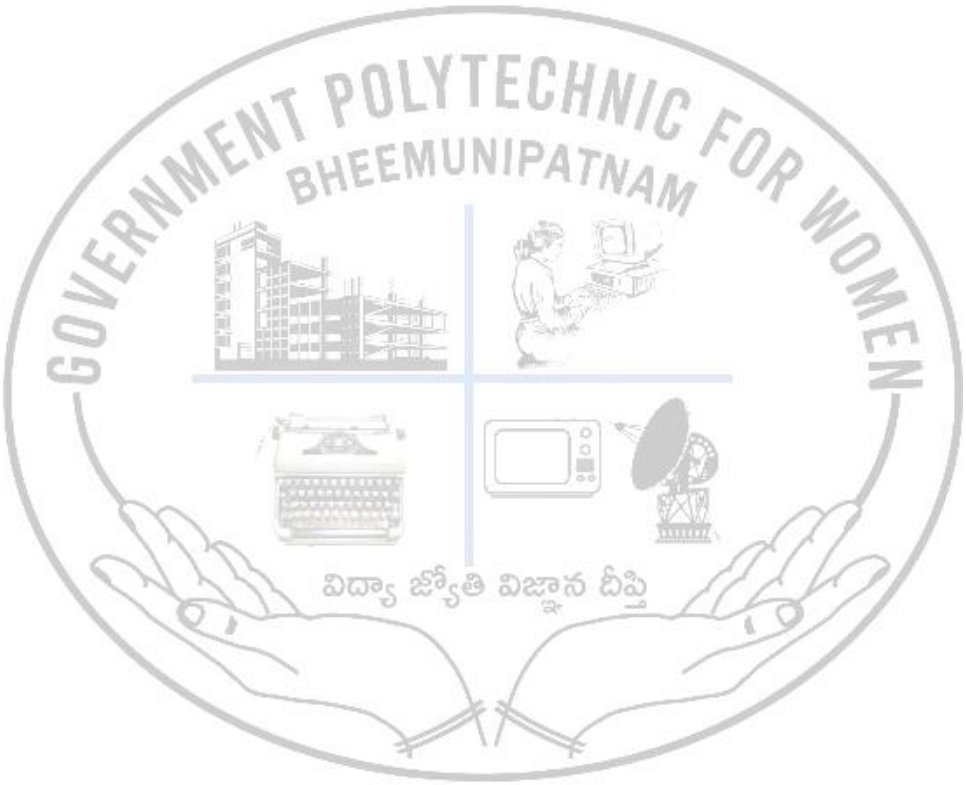
```
            help="index of webcam on system")
ap.add_argument("-a", "--alarm", type=str, default="D:\Files\last
desktop\Drowsiness-Detection-System\Alert.WAV", help="path alarm .WAV
file")
args = vars(ap.parse_args())


EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 30
YAWN_THRESH = 20
alarm_status = False
alarm_status2 = False
saying = False
COUNTER = 0


print("-> Loading the predictor and detector...")
#detector = dlib.get_frontal_face_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)
```

```
while True:

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

    #rects = detector(gray, 0)
    rects = detector.detectMultiScale(gray, scaleFactor=1.1,
                minNeighbors=5, minSize=(30, 30),
                flags=cv2.CASCADE_SCALE_IMAGE)

    #for rect in rects:
    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)

        eye = final_ear(shape)
        ear = eye[0]
        leftEye = eye [1]
        rightEye = eye[2]

        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 alarm_status == False:
         alarm_status = True
         if args["alarm"] != "":
            t = Thread(target=sound_alarm,
                  args=(args["alarm"],))
         t.deamon = True
         t.start()

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

else:
   COUNTER = 0
```

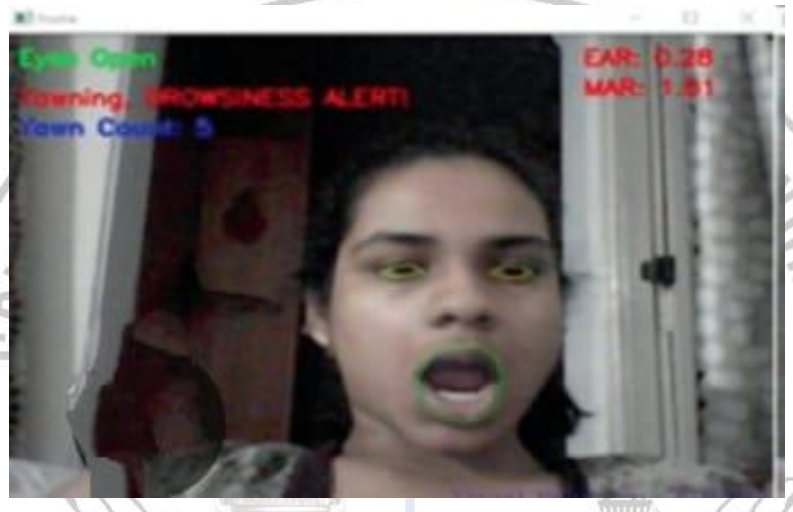
```
        alarm_status = False

    if (distance > YAWN_THRESH):
            cv2.putText(frame, "Yawn Alert", (10, 30),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            if alarm_status2 == False and saying == False:
                alarm_status2 = True
                if args["alarm"] != "":
                    t = Thread(target=sound_alarm,
                               args=(args["alarm"],))
                    t.deamon = True
                    t.start()
    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"):
```

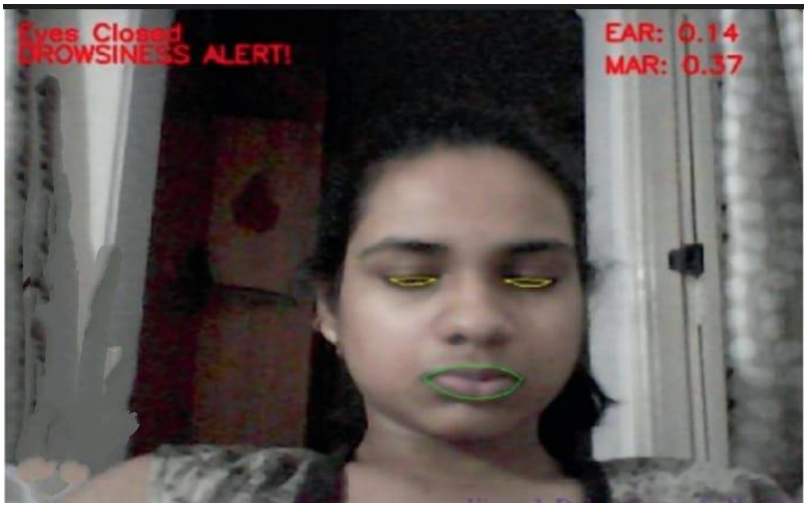
```
        break

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



RESULT

Real time driver drowsiness project is implemented with the algorithms Python and OpenCV. It takes input by capturing the video with the camera. The captured video was divided into frames. The system analyses the each frame and detects patterns associated with drowsiness. patterns are: If the eye drowsiness is detected between 400ms-500ms and if the mouth is around 4-6 centimetres or 1.5-2.5 inches , then it is classified as drowsy condition else it is regarded as normal blink or talking. These alert in the form of sounding an alarm. By monitoring the drivers drowsiness levels and providing the timely alerts, it aims to enhance road safety and prevents accidents caused by drowsy driving.





FUTURE SCOPE

In last decades many researchers worked on drowsiness detection and got significant success. However, still, research is going on in this field to make the system more reliable. There are some challenges with this application. For example, if a driver is wearing spectacles, having a partial face that could not be seen by a camera or sensor. Our model is designed for detection of drowsy state of eye and give and alert signal or warning in the formof audio alarm. But the response of driver after being warned may not be enough to stop causing the accident meaning that if the driver is slow in responding towards the warning signal then accident may occur. Hence to avoid this we can design and fit a motor driven system and synchronize it with the warning signal so that the vehicle will slow down after getting the warning signal automatically.

1) For future work, we will extend our application to detect drowsiness even when the subject is using sunglass or colour spectacles. Another future study can be done to detect drowsiness is to develop a system which can detect drowsiness in the night or with very lowillumination.

2) Real-time data are always unconstrained and with unconstrained nature of the subject/driver, it becomes very difficult to find facial landmarks. Therefore, to overcome thisconstraint a more reliable application can be developed which can detect face even when user/subject is not friendly.

3) To increase the sensitivity of application, a multimodal system can be developed which notuses only eye feature, but also facial expression, mouth feature and combine all features to make the decision where driver/subject is sleeping or not.

4) False alarms and missed detections are common, reducing reliability. To overcome this utilize multi-sensor integration to improve accuracy and reduce false alarms.

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.

CONCLUSION

Now a day, the accident ratio has been increased in large amount. Especially in a country like India, the rate has become more, and the main reason for these accidents to cause is driver fatigue or drowsiness. We should always remember life gives us only one chance. Therefore, this is a device which would help a huge amount of percent from accidents. This system is used to detect drowsiness while driving by giving a buzzer and also alerting people by alert messages. A non-invasive system to localize the eyes and monitor fatigue was developed. Information about the eyes position is obtained through various self-developed image processing algorithms. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for too long, a warning signal is issued. In addition, during monitoring, the system is able to automatically detect any eye localizing error that might have occurred. In case of this type of error, the system is able to recover and properly localize the eyes. The following conclusions were made:

• Image processing achieves highly accurate and reliable detection of drowsiness.

• Image processing offers a non-invasive approach to detecting drowsiness without the annoyance and interference.

• A drowsiness detection system developed around the principle of image processing judges the driver's alertness level on the basis of continuous eye closures. With 80% accuracy, it is obvious that there are limitations to the system

REFERENCES

IEEE standard:

Journal Paper,

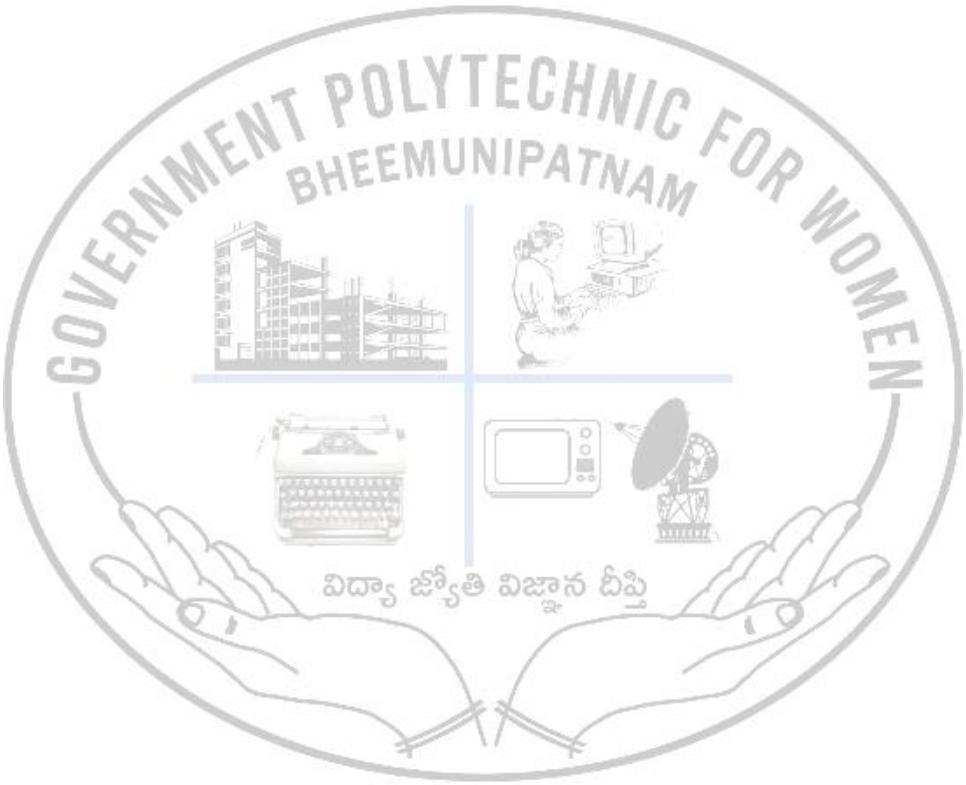
[1] Facial Features Monitoring for Real Time Drowsiness Detection by Manu B.N, 2016 12th International Conference on Innovations in InformationTechnology (IIT) [Pg. 78-81]
https://ieeexplore.ieee.org/document/7880030

[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] International Journal of Science, Engineering and Technology Research (IJSETR) Volume 2, Issue 9, September 2013

Names of Websites referred:

1. https://www.codeproject.com/Articles/26897/TrackEye-Real-Time-Tracking-Of-Human- Eyes- Using-a
2. https://realpython.com/face-recognition-with-python/ https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/
3. https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/
4. https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-dlib-opencv python/
5. https://www.codeproject.com/Articles/26897/Trac kEye-Real-Time- Tracking-Of- HumanEyesUsing-a
6. https://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_detection.html
7. https://www.learnopencv.com/training-better-haar-lbp-cascade-eye-detector-opencv/


GOVERNMENT POLYTECHNIC FOR WOMEN
BHEEMUNIPATNAM
విద్యా జ్యోతి విజ్ఞాన దీప్తి