Drowsiness Detection Project Report

A report submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
In
Computer Science and Engineering

Submitted by: Submitted to:
Bazgha Razi

Delhi Global Institute of Technology

Bahadurgarh, Jhajjar, Haryana 124201

Affiliated to

Maharshi Dayanand University

CERTIFICATE

This is to certify that the project Drowsiness-Detection submitted by Bazgha Razi, to Maharishi Dayanand University(MDU), Rohtak-124001, Haryana in partial fulfilment of the requirement for the award of semester 7th of the degree of B.Tech in Computer Science Engineering. This project full-fills the requirement as per the regulation of the institute of university in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this time institute or any other institute or university to the best of my knowledge and belief.

Mentor Signature

ACKNOWLEDGEMENT

It has been great honour and privilege to undergo this project.

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all.

I am highly indebted to professors for their guidance and constant supervision as well as for providing necessary information regarding the project and for their support in completing the project. Their constant guidance made me understand this project and its manifestations in great depths helped us to complete the assigned tasks on time.

I am also thankful and grateful to my parents who helped me throughout this project period.

Bazgha Razi

B.Tech(CSE)

<u>**DECLARATION**</u>

I here by declare that the project report entitled "DROWSINESS-DETECTION" submitted by me. Bazgha Razi partially fulfilment the requirements for this project work under the guidance of professors.

I also declare that, the report is only prepared for my academic requirement not for any other purpose.

Bazgha Razi
B.Tech(CSE)

CONTENTS

INTRODUCTION

Driving force exhaustion is a vast variable in an expansive range of car accidents. Past due insights, investigate that every year many deaths and injuries may be credited to weariness associated injuries.

Avenue accidents in the world possess much loss. It can be seen there are around 2, four hundred road accidents always that's one death per each four hours. It has been figured around 20% of vehicle crashes with driving force fatalities are due to driving force’s drowsiness. It became uncovered that using execution fast drop with multiplied tiredness which brings about making extra than 20% of all automobile accidents. [1]

Less attention leads the driver to being distracted and the chance of road twist of fate is going high. Drowsiness related injuries have all the earmarks of being extra serious, due to the higher speeds concerning distraction and the driving force being no longer able to take any keeping off activity, or maybe brake, before the coincidence.

The development of improvements for recognizing or stopping tiredness of the motive force is a tremendous take a look at inside the field of coincidence stopping systems. Due to the danger that that drowsiness presents on the street, techniques need to be created for checking its effects.

Loss of the attention due to the tiredness reasons a few modifications inside the human's frame and sports. Those aspect outcomes and parameters empower us to efficiently degree the drowsiness stage. [2]

Exclusive strategies for drowsiness identity can be partitioned into standard classifications. The techniques inside the first collecting apprehend the extent of the tiredness focused across the physiological modifications in the body. Eye repute, speech houses, time interval between two yawning, head function, sitting carriage, coronary heart price, and mind alerts are absolutely a couple of illustrations of the techniques inside the first classification.

Drowsiness moreover brings about a few changes within the driving style. Techniques within the second category estimate the motive force. Drowsiness level by following these progressions. Storage attitude, distance from the subsequent car, lateral role of the car, longitudinal pace, longitudinal speeding up, and lane departure are applied as a part of the method of the second one class.

A countless number of people drive on the highway day and night. Taxi drivers, bus drivers, truck drivers and people traveling long-distance suffer from lack of sleep. Due to which it becomes very dangerous to drive when feeling sleepy.[3]

Fatigue is a protection problem that has not but been deeply tackled by using any united states in the global in particular due to its nature. Fatigue, in well known, may be very difficult to measure or examine unlike alcohol and capsules, which have clear key indicators and assessments that are to be had without problems.

Likely, the exceptional answers to this problem are recognition of approximately fatigue-associated injuries and selling drivers to confess fatigue while needed. The previous is tough and lots more expensive to reap, and the latter is not feasible without the previous as using for lengthy hours could be very rewarding.

The majority of accidents happen due to the drowsiness of the driver. So, to prevent these accidents we will build a system using Python, and OpenCV which will alert the driver when he feels sleepy.

Driver drowsiness has caused a large number of serious injuries and deaths on public roads and incurred billions of taxpayer dollars in costs. Hence, monitoring of drowsiness is critical to reduce this burden on society

The aim of this program will be to design a system which will, with close proximity, detect the state of the driver's eyes i.e. whether they are open or closed in real time. The principle behind this analogy is that the state of someone's eyes can indicate the symptoms of their fatigue and prevent accidents caused by it by sounding an alarm, ideally loud enough to bring the river back to senses and avoid the imminent accident.

An accident is an incident, occurring suddenly, unexpectedly, and inadvertently under unforeseen circumstances. Each day thousands of road accident victims succumb to injuries and death globally. According to studies, almost one-quarter of all serious motorway accidents are due to drivers handling the vehicle in a state of heavy drowsiness indicating that driver drowsiness tends to cause a greater number of accidents than driving under influence [9].

Long haul drivers who put off taking a break while moving for long-distance often run a high risk of becoming sleepy behind the wheel and fail to identify this early on. By making use of the technologies of computer vision, an interdisciplinary science that deals with how computers can be made more intelligent by developing techniques that help them gather high-level understanding from digital images or videos, a solution can be developed for this problem of detecting and notifying of driver drowsiness.

The main idea behind this project is to develop a computer vision system, using OpenCV, which is non-intrusive that can automatically detect driver drowsiness in a real-time video stream and will alert the driver by playing an alarm if they appear to be drowsy.

This system would make use of an algorithm that detects the eye landmark points of the driver and based on this can determine if the driver is in a drowsy state or not, and appropriately sound an alarm.

This project has been a great learning experience as I have discovered most of the tools and techniques during this project period. Learned about various libraries and also learned about OpenCV for making Drowsiness Detection device.

Prior to this project period, I’ve already experience working with Python. It is a high-level, general-purpose programming language. It is commonly used for developing websites and software, task automation, data analysis, and data visualization.

During the project period, I used OpenCV for making this device.

I have also met many guides during the building period of this project. They are passionate about technology and innovation. They made this journey truly inspiring from a technological point.

METHODOLOGY

The proposed drowsiness detection system deals with eye closure detection, yawn detection and head tilt detection. The project is carried out in OpenCV environment using dlib facial landmark detector. The lane detection is performed by Canny edge and hough transform technique and implemented in both image and video files.

A. Facial landmarks detection

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. Detecting facial landmarks is a subset of the shape prediction problem. Given an input image (and normally an ROI that specifies the object of interest), a shape predictor attempts to localize key points of interest along the shape. Detecting facial landmarks is therefore a two-step process:

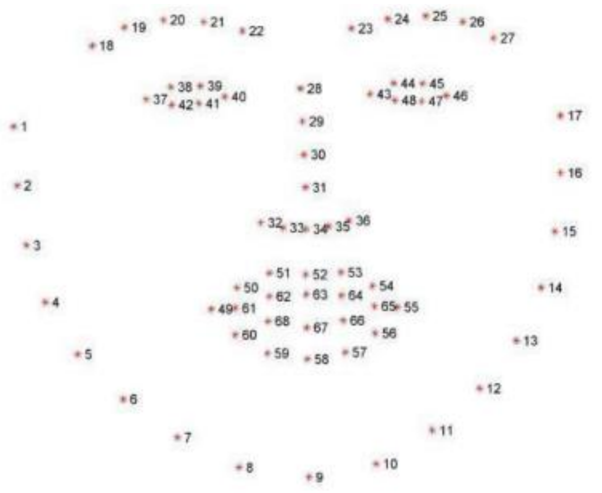
Step 1: Localize the face in the image.

Step 2: Detect the key facial structures on the face ROI

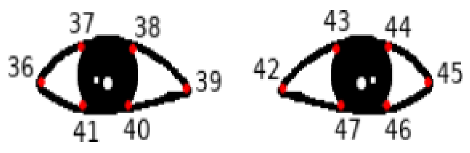
The facial detection is done using dlib python library module, OpenCV in Python IDE. Dlib is a landmark's facial detector with pre-trained models which is used to estimate the location of 68 coordinates (x, y) that map the facial points on a person's face.

B. Eye Closure Detection

The face is continuously captured using web camera and the captured frames are processed in OpenCV environment. The face is detected and key facial features are extracted using dlib facial landmark detector. The landmark indices for both eye regions are highlighted in the frame. The Eye Aspect Ratio (EAR) is estimated continuously and displayed in the frame. When eyes are opened, EAR value will be high. When the eyes are closed, value of EAR will be lesser. When the EAR value goes below a certain threshold, eye closure will be detected. Upon detecting the eye closure driver is alerted by triggering the alarm.



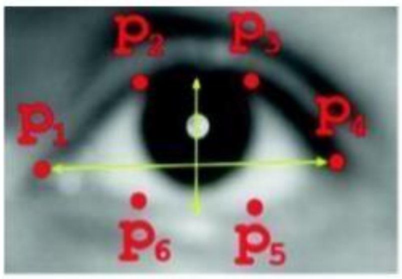
Visualization of the 68 facial landmark co-ordinates [4]



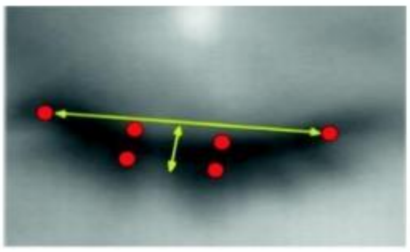
Landmark indices of eyes [4]

C. Eye Aspect Ratio

EAR refers to the aspect ratio of the eye region, which is often used to calculate the temporal consistency and speed of left and right eye blinks and in fatigue detection. These co-ordinates are used to estimate the value of EAR. Landmark indices for right eye → [36,37,38,39,40,41] Landmark indices for left eye → [42,43,44,45,46,47]



Landmarks when eyes opened [4]



Landmarks when eyes closed [4]

The formula to estimate the Eye Aspect Ratio (EAR) isgiven by as: EAR = |p2−p6|+|p3−p5| |p1−p4| The numerator of this equation computes the distance between the vertical eye landmarks while the denominator computes the distance between horizontal eye landmarks, weighting the denominator appropriately since there is only one set of horizontal points but two sets of vertical points.

PROJECT DETAILS

Name of the Project

Drowsiness Detection

Hardware Used

Personal Computer

Camera

Cable Connector

Software Used

Windows (7,10,11)

VS Code (Python IDE)

Libraries and Modules Used

Python

OpenCV

Imutils

Scipy

Dlib

Numpy

Version Control/ IDE

Github

VS Code

Web Browser

Google Chrome

VISION AND MISSION

Vision: To become the best drowsiness detector in industry to prevent accidents on road caused by the driver getting drowsy.

Mission: To aid in the prevention of accidents. The system will detect the early symptoms of drowsiness before the driver has fully lost all attentiveness and warn the driver that they are no longer capable of operating the vehicle safely.

DROWSINESS DETECTION

The drowsiness of a person driving a vehicle is the primary cause of accidents all over the world. Due to lack of sleep and tiredness, fatigue and drowsiness are common among many drivers, which often leads to road accidents. Alerting the driver ahead of time is the best way to avoid road accidents caused by drowsiness. Drowsiness detection is a car safety technology that helps prevent accidents caused by the driver getting drowsy.[1]

Problem Statement

The goal of the project is to detect drowsiness while driving and inform the driver at the appropriate time to avoid any mishaps. The project employs a model to determine whether or not a person is drowsy based on whether their eyes are closed or open. The idea has a direct application in the vehicle sector, making driving safer and lowering the number of people killed in car accidents caused by drowsy driving.

Why Drowsiness Detection?

- It prevents road accidents caused by the driver getting drowsy.
- It is secure for a person as well as the vehicle which damaged due to accidents.
- It is safe, easy to use and less costly.
- It reduces the passenger’s tension while travelling.
- It detects person’s facial expression very quickly and inform them back with the beep sound if he/she is feeling drowsy.
- The device is fast and accurate in detecting the eyes.

Application Area of Drowsiness Detection

Transportation:

Drowsiness detection is used in cars which helps prevent accidents caused by the driver getting drowsy.

Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads.[3]

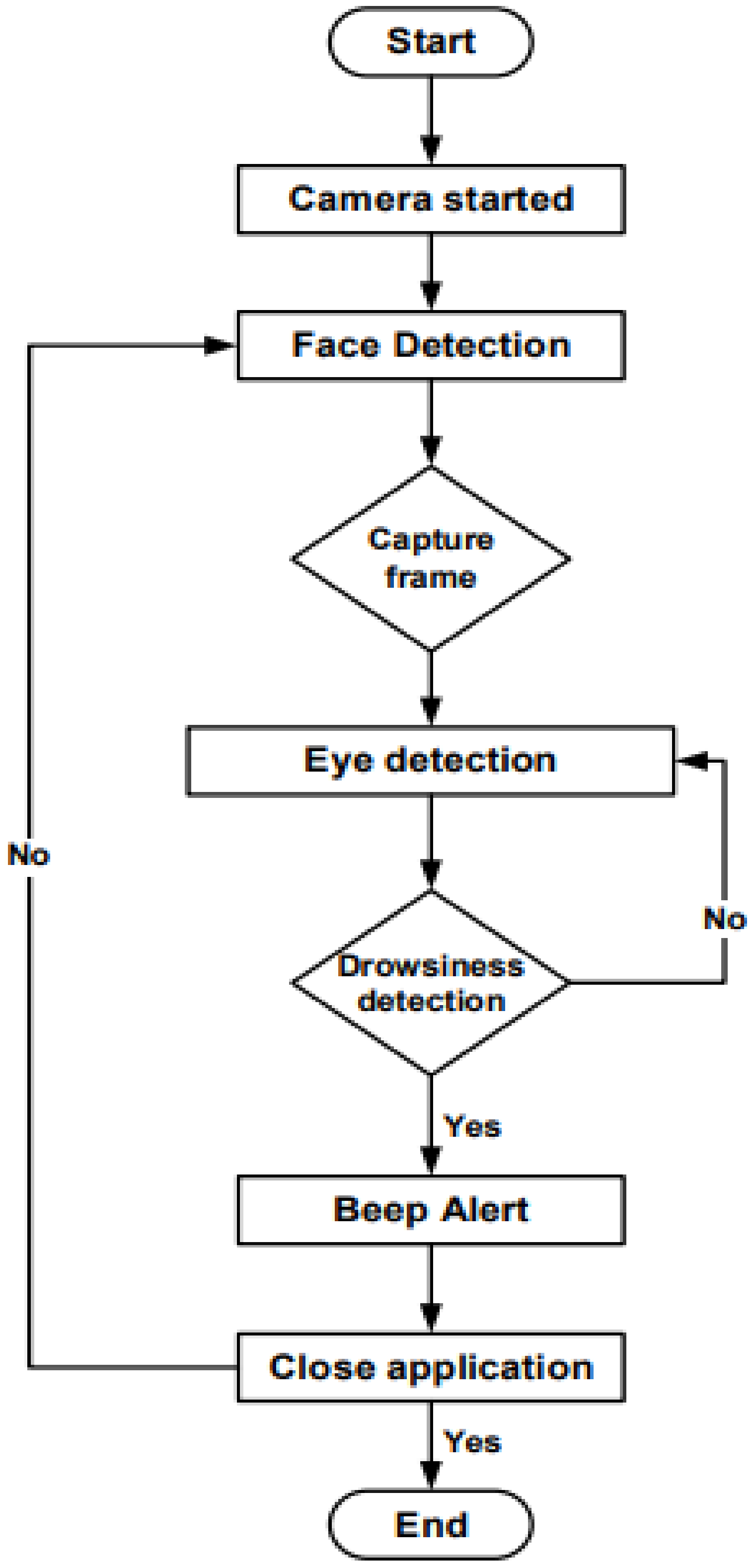
Security Guard’s Cabin:

It is used in a security guard’s cabin; it helps to check if the guard is drowsy while on duty.

Classroom:

It is also used in the classroom or during the study. It helps to prevent students from dozing off while studying.

Block Diagram



[4]

Weekly Work For Project

Week 1	Defining scope for the project, idea brainstorming.
Week 2	Create Drowsiness Detection device. Add all the necessary components using some libraries and modules.
Week 3	Connect the camera and check the camera connection properly.
Week 4	Evaluation and testing of the Drowsiness Detection device.

What is an IDE?

An integrated development environment (IDE) is software for building applications that combines common developer tools into a single graphical user interface (GUI). An IDE typically consists of:

- **Source code editor**: A text editor that can assist in writing software code with features such as syntax highlighting with visual cues, providing language specific auto-completion, and checking for bugs as code is being written.
- **Local build automation**: Utilities that automate simple, repeatable tasks as part of creating a local build of the software for use by the developer, like compiling computer source code into binary code, packaging binary code, and running automated tests.
- **Debugger**: A program for testing other programs that can graphically display the location of a bug in the original code.

An IDE allows developers to start programming new applications quickly because multiple utilities don’t need to be manually configured and integrated as part of the setup process. Developers also don’t need to spend hours individually learning how to use different tools when every utility is represented in the same workbench. This can be especially useful for onboarding new developers who can rely on an IDE to get up to speed on a team’s standard tools and workflows. In fact, most features of IDEs are meant to save time, like intelligent code completion and automated code generation, which removes the need to type out full character sequences.[7]

NOTE: During my project I used Visual Studio Code for drowsiness detection.

Python

Python is an object-oriented, high-level programming language. The main feature of python is that it comes with dynamic semantics. When its high-level built in data structures combine with dynamic typing and dynamic binding, they make it perfect for Rapid Application Development. Also it can be used as a scripting or glue language to connect existing components together.

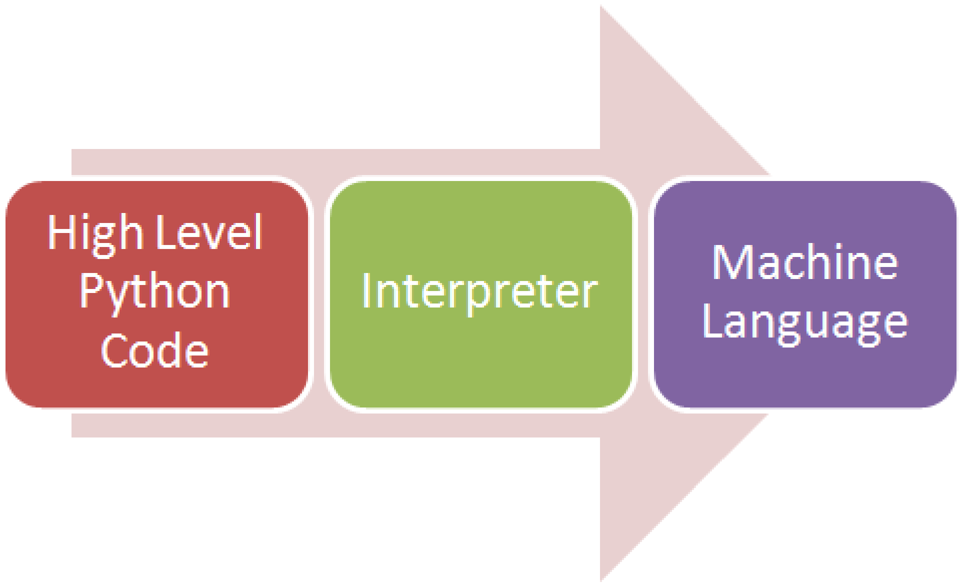
Python is an interpreted language. Because of the higher readability of its syntax, python is known as a very simple language compared to the other computer languages.

Why Python?

- Python is the most popular language due to the fact that it’s easier to code and understand it.
- Python is an object-oriented programming language and can be used to write functional code too.
- It is a suitable language that bridges the gaps between business and developers.
- Subsequently, it takes less time to bring a Python program to market compared to other languages such as C#/Java.
- Additionally, there are a large number of python machine learning and analytical packages.
- A large number of communities and books are available to support Python developers.
- Nearly all types of applications, ranging from forecasting analytical to UI, can be implemented in Python.
- There is no need to declare variable types. Thus, it is quicker to implement a Python application.

How Does Python Work?

This image illustrates how python runs on our machines:



The key here is the Interpreter that is responsible for translating high-level Python language to low-level machine language.

The way Python works is as follows:

1. A Python virtual machine is created where the packages (libraries) are installed. Think of a virtual machine as a container.
2. The python code is then written in .py files
3. CPython compiles the Python code to bytecode. This bytecode is for the Python virtual machine.
4. When you want to execute the bytecode then the code will be interpreted at runtime. The code will then be translated from the bytecode into the machine code. The bytecode is not dependent on the machine on which you are running the code. This makes Python machine-independent.

OpenCV

OpenCV is a library of Programming functions which are used in real-time Computer Vision. It is developed by Intel Corporation, Willow Garage, Itseez .

OpenCV is cross-Platform and Free for use under the license which makes it easy for businesses to utilize and modify the code and supports the Deep Learning Frameworks TensorFlow, Torch/PyTorch and Caffe.

OpenCV is written in C++. supports Windows, Linux, Android and Mac OS.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

OpenCV has more than 47,000 people of user community and estimated number of downloads exceeding 18 million or 180 lakh which makes its more important in world community.[8]

The library is also used by Companies, Research Groups and by Governmental bodies .

Private companies :

Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota and many more.

Startups:

Applied Minds, VideoSurf, and Zeitera , and many more.

Governments:

- OpenCV’s deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel.
- Helping robots navigate and pick up objects at Willow Garage.
- Detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York.
- Inspecting labels on products in factories around the world on to rapid face detection in Japan.

Applications:

OpenCV’s application areas include:

- 2D and 3D features toolkits
- Egomotion estimation
- Facial Recognition system
- Gesture Recognition
- Human-computer interaction (HCI)
- Mobile robotics
- Motion understanding
- Object identification
- Segmentation and recognition
- Structure from motion (SFM)
- Motion tracking
- Augmented reality

Imutils

A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much more easier with OpenCV and both Python 2.7 and Python 3.

Installation

Provided you already have NumPy, SciPy, Matplotlib, and OpenCV already installed, the imutils package is completely pip-installable:

```
$ pip install imutils
```

Finding function OpenCV functions by name

OpenCV can be a big, hard to navigate library, especially if you are just getting started learning computer vision and image processing. The find_function method allows you to quickly search function names across modules (and optionally sub-modules) to find the function you are looking for.

Example:

Let's find all function names that contain the text contour:

```
import imutils

imutils.find_function("contour")
```

Output:

1. contourArea

2. drawContours

3. findContours

4. isContourConvex

The contourArea function could therefore be accessed via: cv2.contourArea

Translation

Translation is the shifting of an image in either the *x* or *y* direction. To translate an image in OpenCV you would need to supply the *(x, y)*-shift, denoted as (t_x, t_y) to construct the translation matrix *M*:

$$M = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$$

And from there, you would need to apply the cv2.warpAffine function.

Instead of manually constructing the translation matrix *M* and calling cv2.warpAffine, you can simply make a call to the translate function of imutils.

Example:

# translate the image x=25 pixels to the right and y=75 pixels up

translated = imutils.translate(workspace, 25, -75)

Rotation

Rotating an image in OpenCV is accomplished by making a call to cv2.getRotationMatrix2D and cv2.warpAffine. Further care has to be taken to supply the *(x, y)*-coordinate of the point the image is to be rotated about. These calculation calls can quickly add up and make your code bulky and less readable. The rotate function in imutils helps resolve this problem.

Example:

```
# loop over the angles to rotate the image
for angle in xrange(0, 360, 90):
    # rotate the image and display it
    rotated = imutils.rotate(bridge, angle=angle)
    cv2.imshow("Angle=%d" % (angle), rotated)
```

Resizing

Resizing an image in OpenCV is accomplished by calling the cv2.resize function. However, special care needs to be taken to ensure that the aspect ratio is maintained. This resize function of imutils maintains the aspect ratio and provides the keyword arguments width and height so the image can be resized to the intended width/height while (1) maintaining aspect ratio and (2) ensuring the dimensions of the image do not have to be explicitly computed by the developer.

Another optional keyword argument, inter, can be used to specify interpolation method as well.

Example:

```
# loop over varying widths to resize the image to
for width in (400, 300, 200, 100):
    # resize the image and display it
    resized = imutils.resize(workspace, width=width)
    cv2.imshow("Width=%dpx" % (width), resized)
```

Scipy

SciPy is a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python. It adds significant power to the interactive Python session by providing the user with high-level commands and classes for manipulating and visualizing data. With SciPy, an interactive Python session becomes a data-processing and system-prototyping environment rivaling systems, such as MATLAB, IDL, Octave, R-Lab, and SciLab.

SciPy was created by NumPy's creator Travis Olliphant.

SciPy has optimized and added functions that are frequently used in NumPy and Data Science.

SciPy provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics and many other classes of problems.

The algorithms and data structures provided by SciPy are broadly applicable across domains.

Extends NumPy providing additional tools for array computing and provides specialized data structures, such as sparse matrices and k-dimensional trees.

SciPy wraps highly-optimized implementations written in low-level languages like Fortran, C, and C++. Enjoy the flexibility of Python with the speed of compiled code.

SciPy’s high level syntax makes it accessible and productive for programmers from any background or experience level.

The main feature of the **scipy.special** package is the definition of numerous special functions of mathematical physics. Available functions include airy, elliptic, bessel, gamma, beta, hypergeometric, parabolic cylinder, mathieu, spheroidal wave, struve, and kelvin.

There are also some low-level stats functions that are not intended for general use as an easier interface to these functions is provided by the stats module. Most of these functions can take array arguments and return array results following the same broadcasting rules as other math functions in Numerical Python.

Many of these functions also accept complex numbers as input. For a complete list of the available functions with a one-line description type >>> help(special).

Each function also has its own documentation accessible using help. If you don’t see a function you need, consider writing it and contributing it to the library. You can write the function in either C, Fortran, or Python. Look in the source code of the library for examples of each of these kinds of functions.

DLib

Dlib is principally a C++ library, however, you can use a number of its tools from python applications. A toolkit for making real world machine learning and data analysis applications.

We will start by importing the **dlib** module.

After that we will call the **load_rgb_image** function to load an image from the file system. This function receives as input the path to the image, as a string, and returns a numpy_ndarray We will store the result in a variable.

img = dlib.load_rgb_image("C:/Users/N/Desktop/Test.jpg")

After this we will create an object of class image_window, where we will be able to display our original image and also the rectangle around the detected face. As input of the constructor we will pass our previously loaded image, which will be displayed on the window.

Note that the **image_window** class also has a parameterless constructor that allows to initialize the window without any image. This allows us to set the image later, with a call to the set_image method.

win = dlib.image_window(img)

Then we will call the get_frontal_face_detector function from the **dlib** module. This will return to us an object of class fhog_object_detector (the default detector from dlib), which we can use to detect faces.

detector = dlib.get_frontal_face_detector()

Instances of this class are callable (check more about callables in Python and the definition of **__call__** for this class). In our case, to perform the face detection, we simply need to call the instance and pass as input the image where we want to detect faces. As output, it will return an array of objects of class rectangle (it will be empty in case no face is found).

```
face = detector(img)
```

Now that we have our array of rectangles, we can simply call the **add_overlay** method on or **image_window**, passing as input the rectangles. The rectangles will be drawn on the window, over the image we have previously loaded.

```
win.add_overlay(face)
```

To finalize, we will call the **wait_until_closed** method on our **image_window** object, to ensure the window stays opened until we close it.

```
win.wait_until_closed()
```

The complete code can be seen below.

```
import dlib
img = dlib.load_rgb_image("C:/Users/N/Desktop/Test.jpg")
win = dlib.image_window(img)
detector = dlib.get_frontal_face_detector()
face = detector(img)
win.add_overlay(face)
win.wait_until_closed()
```

Before running the code, don’t forget to change the string passed as input of the **load_rgb_image** to point to an image in your file system.

Building drowsiness detection

The libraries need for driver drowsiness detection system are

- Opencv
- Dlib
- Numpy

These are the only packages you will need for this machine learning project.

OpenCV and NumPy installation is using pip install and dlib installation using pip only works if you have cmake and vs build tools 2015 or later (if on python version>=3.7)

The easiest way is to create a python 3.6 env in anaconda and install a dlib wheel supported for python 3.6.

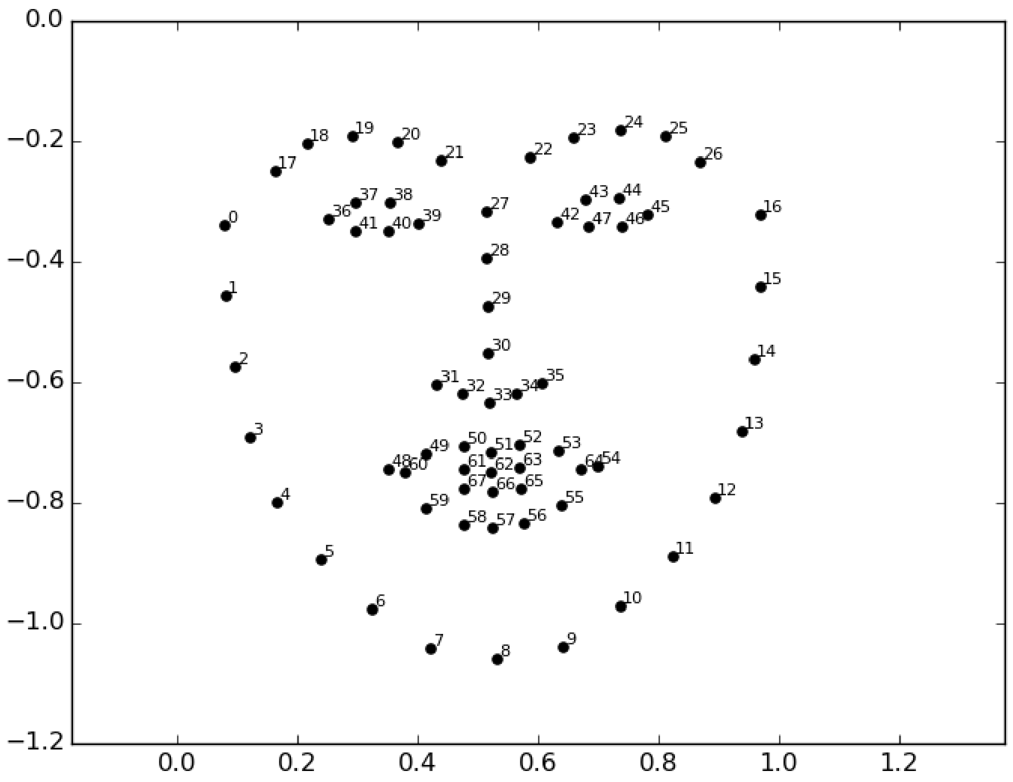
Import the libraries

Numpy is used for handling the data from dlib and mathematical functions. Opencv will help us in gathering the frames from the webcam and writing over them and also displaying the resultant frames.

Dlib to extract features from the face and predict the landmark using its pre-trained face landmark detector.

Dlib is an open source toolkit written in c++ that has a variety of machine learning models implemented and optimized. Preference is given to dlib over other libraries and training your own model because it is fairly accurate, fast, well documented, and available for academic, research, and even commercial use.

Dlib's accuracy and speed are comparable with the most state-of-the-art neural networks, and because the scope of this project is not to train one, we'll be using dlib python wrapper.



[5]

Pretrained facial landmark model is available with the code, you can download it from there.

The hypot function from the math library calculates the hypotenuse of a right-angle triangle or the distance between two points (euclidean norm).

```
import numpy as np

import dlib

import cv2

from math import hypot
```

Here we prepare our capture call to OpenCV’s video capture method that will capture the frames from the webcam in an infinite loop till we break it and stop the capture.

```
cap = cv2.VideoCapture(0)
```

Dlib’s face and facial landmark predictors

Keep the downloaded landmark detection .dat file in the same folder as this code file or provide a complete path in the dlib.shape_predictor function.

This will prepare the predictor for further prediction.

```
detector = dlib.get_frontal_face_detector()

predictor =
dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
```

We create a function to calculate the midpoint from two given points.

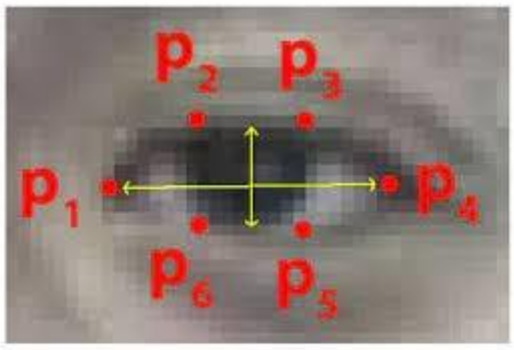
As we are gonna use this more than once in a call we create a separate function for this.

```
def mid(p1 ,p2):

    return int((p1.x + p2.x)/2), int((p1.y + p2.y)/2)
```

Create a function for calculating the blinking ratio

Create a function for calculating the blinking ratio or the eye aspect ratio of the eyes. There are six landmarks for representing each eye.



[5]

Starting from the left corner moving clockwise. We find the ratio of height and width of the eye to infer the open or close state of the eye.blink-ratio=(|p2-p6|+|p3-p5|)(2|p1-p4|). The ratio falls to approximately zero when the eye is close but remains constant when they are open.

```
def eye_aspect_ratio(eye_landmark, face_roi_landmark):

    left_point = (face_roi_landmark.part(eye_landmark[0]).x,
face_roi_landmark.part(eye_landmark[0]).y)

    right_point = (face_roi_landmark.part(eye_landmark[3]).x,
face_roi_landmark.part(eye_landmark[3]).y)

    center_top = mid(face_roi_landmark.part(eye_landmark[1]),
face_roi_landmark.part(eye_landmark[2]))

    center_bottom = mid(face_roi_landmark.part(eye_landmark[5]),
face_roi_landmark.part(eye_landmark[4]))
```

```
    hor_line_length = hypot((left_point[0] - right_point[0]),
(left_point[1] - right_point[1]))

    ver_line_length = hypot((center_top[0] - center_bottom[0]),
(center_top[1] - center_bottom[1]))

    ratio = hor_line_length / ver_line_length

    return ratio
```

Create a function for calculating mouth aspect ratio

Similarly, we define the mouth ratio function for finding out if a person is yawning or not. This function gives the ratio of height to width of mouth. If height is more than width it means that the mouth is wide open.

For this as well we use a series of points from the dlib detector to find the ratio.

```
def mouth_aspect_ratio(lips_landmark, face_roi_landmark):

    left_point = (face_roi_landmark.part(lips_landmark[0]).x,
face_roi_landmark.part(lips_landmark[0]).y)

    right_point = (face_roi_landmark.part(lips_landmark[2]).x,
face_roi_landmark.part(lips_landmark[2]).y)

    center_top = (face_roi_landmark.part(lips_landmark[1]).x,
face_roi_landmark.part(lips_landmark[1]).y)

    center_bottom = (face_roi_landmark.part(lips_landmark[3]).x,
face_roi_landmark.part(lips_landmark[3]).y)
```

```
    hor_line_length = hypot((left_point[0] - right_point[0]),
(left_point[1] - right_point[1]))

    ver_line_length = hypot((center_top[0] - center_bottom[0]),
(center_top[1] - center_bottom[1]))

    if hor_line_length == 0:

        return ver_line_length

    ratio = ver_line_length / hor_line_length

    return ratio
```

We create a counter variable to count the number of frames the eye has been close for or the person is yawning and later use to define drowsiness in driver drowsiness detection system project

Also, we declare the font for writing on images with opencv.

```
count = 0

font = cv2.FONT_HERSHEY_TRIPLEX
```

Begin processing of frames

Creating an infinite loop we receive frames from the opencv capture method.

We flip the frame because mirror image and convert it to grayscale. Then pass it to the face detector.

```
while True:

    _, img = cap.read()

    img = cv2.flip(img,1)

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    faces = detector(gray)
```

We loop if there are more than one face in the frame and calculate for all faces. Passing the face to the landmark predictor we get the facial landmarks for further analysis.

Passing the points of each eye to the compute_blinking_ratio function we calculate the ratio for both the eyes and then take the mean of it.

We also put the ratio on the top of the image.

```
for face_roi in faces:

        landmark_list = predictor(gray, face_roi)

        left_eye_ratio = eye_aspect_ratio([36, 37, 38, 39, 40, 41],
landmark_list)

        right_eye_ratio = eye_aspect_ratio([42, 43, 44, 45, 46, 47],
landmark_list)

        eye_open_ratio = (left_eye_ratio + right_eye_ratio) / 2

        cv2.putText(img, str(eye_open_ratio), (0, 13), font, 0.5, (100,
100, 100))
```

```
        ###print(left_eye_ratio,right_eye_ratio,eye_open_ratio)



//Similarly we calculate the ratio for the mouth to get yawning status,
for both outer and inner lips to be more accurate and calculate its
mean.

        inner_lip_ratio = mouth_aspect_ratio([60,62,64,66],
landmark_list)

        outter_lip_ratio = mouth_aspect_ratio([48,51,54,57],
landmark_list)

        mouth_open_ratio = (inner_lip_ratio + outter_lip_ratio) / 2;

        cv2.putText(img, str(mouth_open_ratio), (448, 13), font, 0.5,
(100, 100, 100))

       ###print(inner_lip_ratio,outter_lip_ratio,mouth_open_ratio)
```

Now that we have our data we check if the mouth is wide open and the eyes are not closed. If we find that either of these situations occurs we increment the counter variable counting the number of frames the situation is persisting.

We also find the coordinates for the face bounding box

If the eyes are close or yawning occurs for more than 10 consecutive frames we infer the driver as drowsy and print that on the image as well as creating the bounding box red, else just create a green bounding box.

```
if mouth_open_ratio > 0.380 and eye_open_ratio > 4.0 or
eye_open_ratio > 4.30:

    count +=1

else:

    count = 0

x,y = face_roi.left(), face_roi.top()

x1,y1 = face_roi.right(), face_roi.bottom()

if count>10:

    cv2.rectangle(img, (x,y), (x1,y1), (0, 0, 255), 2)

    cv2.putText(img, "Sleepy", (x, y-5), font, 0.5, (0, 0, 255))

else:

    cv2.rectangle(img, (x,y), (x1,y1), (0, 255, 0), 2)
```

Finally, we show the frame and wait for the esc keypress to exit the infinite loop.

After we exit the loop we release the webcam capture and close all the windows and exit the program.

```
cap.release()

cv2.destroyAllWindows()
```

Testing and Implementation

Testing

Testing is the process of exercising software with the intent of finding errors and ultimately correcting them. The following testing techniques have been used to make this project free of errors.

Content Review

The whole content of the project has been reviewed thoroughly to uncover typographical errors, grammatical error and ambiguous sentences.

Navigation Errors

Different users were allowed to navigate through the project to uncover the navigation errors. The views of the user regarding the navigation flexibility and user friendliness were taken into account and implemented in the project.

Unit Testing

Focuses on individual software units, groups of related units.

• Unit – smallest testable piece of software.

• A unit can be compiled /assembled / linked/loaded; and put under a test harness.

• Unit testing done to show that the unit does not satisfy the application and /or its implemented software does not match the intended designed structure.

Integration Testing

Focuses on combining units to evaluate the interaction among them

• Integration is the process of aggregating components to create larger components.

• Integration testing done to show that even though components were individually satisfactory, the combination is incorrect and inconsistent.

System testing

Focuses on a complete integrated system to evaluate compliance with specified requirements (test characteristics that are only present when entire system is run)

• A system is a big component.

• System testing is aimed at revealing bugs that cannot be attributed to a component as such, to inconsistencies between components or planned interactions between components.

• Concern: issues, behaviours that can only be exposed by testing the entire integrated system (e.g., performance, security, recovery)each form encapsulates (labels, texts, grid etc.). Hence in case of project in V.B. form are the basic units. Each form is tested thoroughly in term of calculation, display etc.

Regression Testing

Each time a new form is added to the project the whole project is tested thoroughly to rectify any side effects. That might have occurred due to the addition of the new form. Thus regression testing has been performed.

White-Box testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

Techniques used in white-box testing include:

API testing (application programming interface) – testing of the application using public and private APIs.

Code coverage – creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once).

Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies.

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. Code coverage as a software metric can be reported as a percentage for:

Function coverage, which reports on functions executed Statement coverage, which reports on the number of lines executed to complete the test 100% statement coverage ensures that all code paths, or branches (in terms of control flow) are executed at least once. This is helpful in ensuring correct functionality, but not sufficient since the same code may process different inputs correctly or incorrectly.

Black-box testing

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing. Specification-based testing aims to test the functionality of software according to the applicable requirements. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behaviour), either "is" or "is not" the same as the expected value specified in the test case. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. It uses external descriptions of the software, including specifications, requirements, and designs to derive test cases. These tests can be functional or non-functional, though usually functional.

Specification-based testing may be necessary to assure correct functionality, but it is insufficient to guard against complex or high-risk situations.

One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality. On the other hand, black-box testing has been said to be "like a walk in a dark labyrinth without a flashlight." Because they do not examine the source code, there are situations when a tester writes many test cases to check something that could have been tested by only one test case, or leaves some parts of the program untested.

This method of test can be applied to all levels of software testing: unit, integration, system and acceptance. It typically comprises most if not all testing at higher levels, but can also dominate unit testing as well.

Alpha Testing

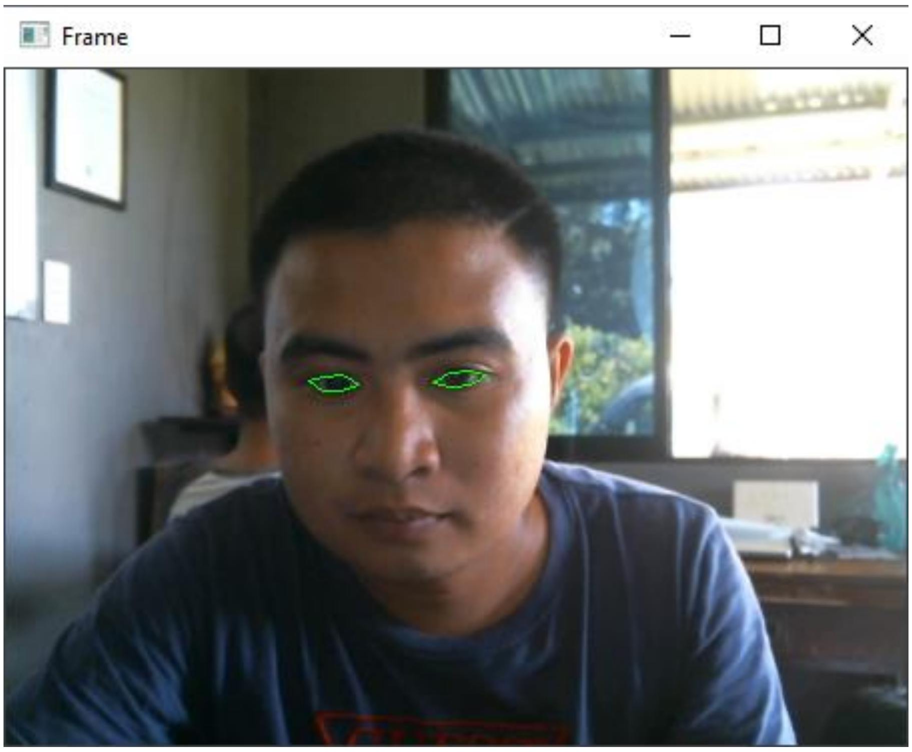
Alpha testing is simulated or actual operational testing by potential users/customers or an independent test team at the developers' site. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing, before the software goes to beta testing.

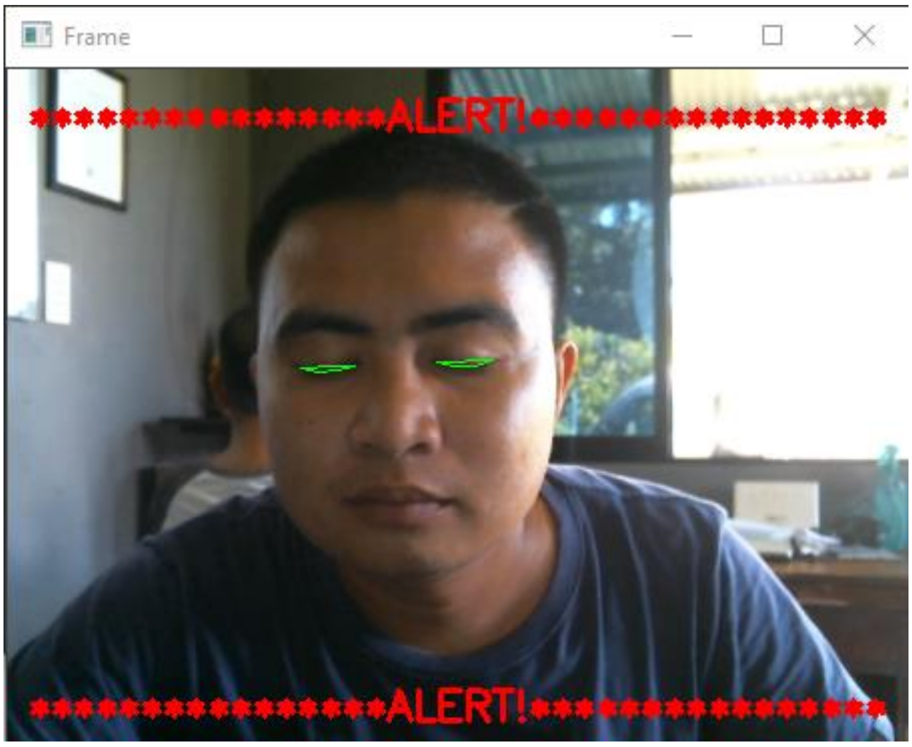
Beta Testing

Beta testing comes after alpha testing and can be considered a form of external user acceptance testing. Versions of the software, known as beta versions, are released to a limited audience outside of the programming team. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

Implementation

Drowsiness Detection

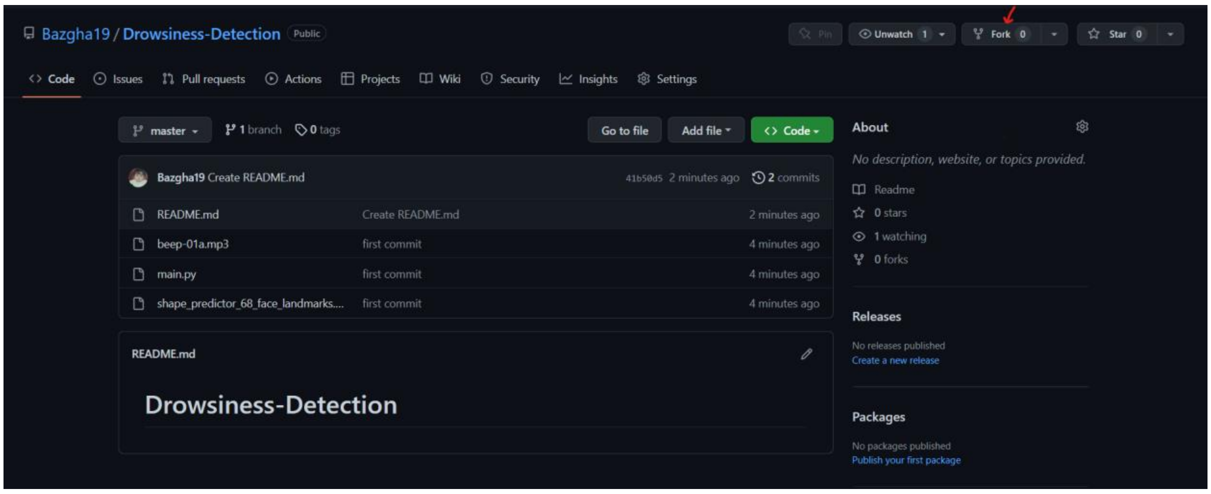



Frame
******************ALERT!******************
******************ALERT!******************

How to use Drowsiness Detection

Fork the repository

Fork the repository by clicking on the fork button on the top of this page. It will create a copy of this project in your github account.



Bazgha19 Create README.md 41b50d5 2 minutes ago 2 commits

README.md	Create README.md	2 minutes ago
beep-01a.mp3	first commit	4 minutes ago
main.py	first commit	4 minutes ago
shape_predictor_68_face_landmarks....	first commit	4 minutes ago

README.md

Drowsiness-Detection

No description, website, or topics provided.

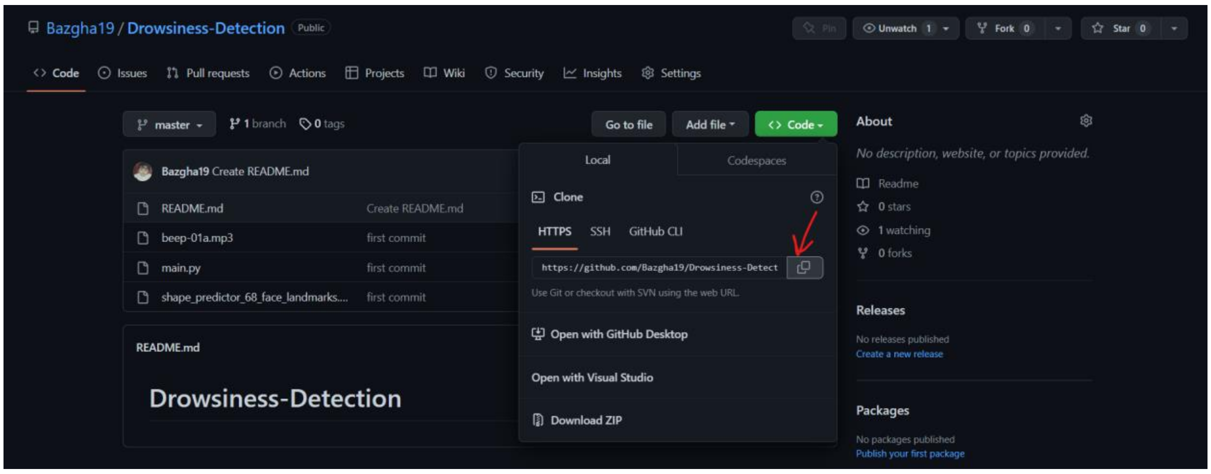
Readme
0 stars
1 watching
0 forks

No releases published
Create a new release

No packages published
Publish your first package

Clone the repository

Open the forked repository, click on the code button and copy the url.



https://github.com/Bazgha19/Drowsiness-Detect

Use Git or checkout with SVN using the web URL.

Install libraries

Install the required libraries using the below command
pip install scipy

```
pip install imutils

pip install dlib

pip install opencv-python
```

<u>Connect the Camera</u>

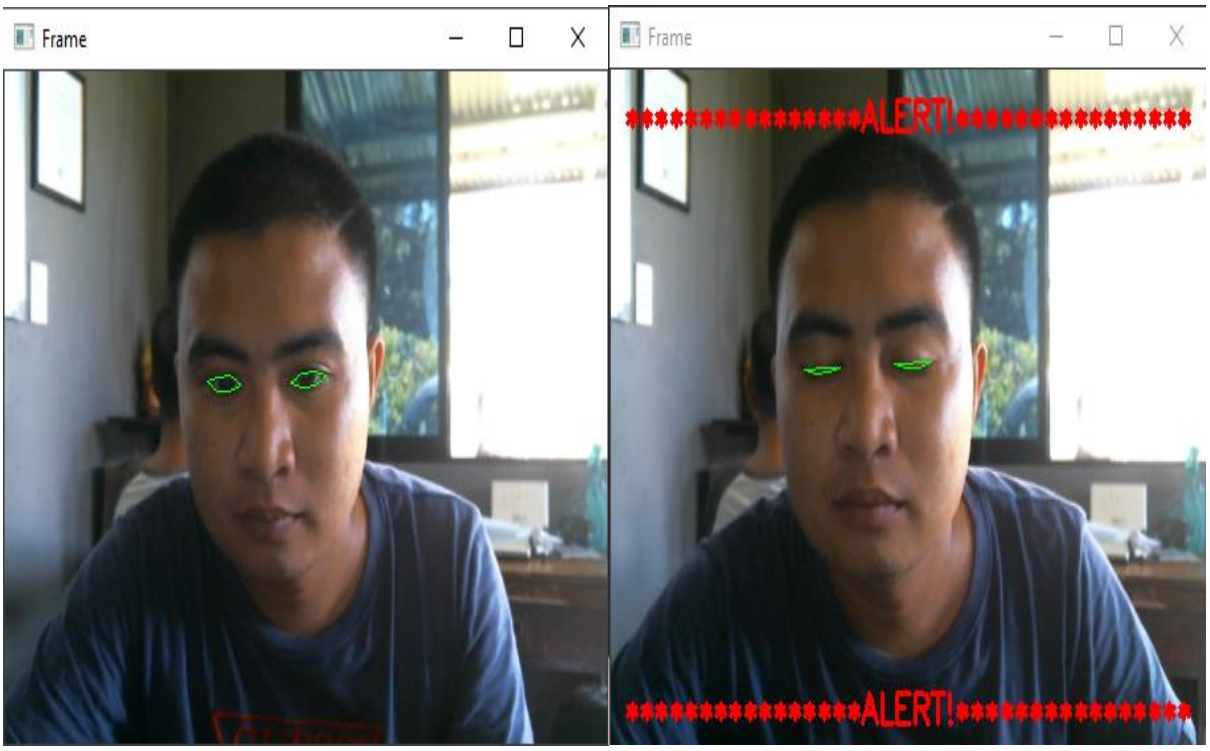
Connect the camera using cable connector for face detection

<u>Done</u>

It's all set, now you're ready to use drowsiness detection by running the below command

```
py main.py
```

Enjoy Drowsiness Detector!!


Frame
Frame
****************ALERT!****************
****************ALERT!****************

Learning’s and Value Addition

A project is a learning experience of its own kind. Neither is it spoon-fed school learning, nor pressure filled workload. It is in between; I not only learn the basics of work life but also the skills required for a brighter professional career.

During my project I had learned lots of thing and few of them I’m mentioning here:

Teamwork: It is important because it enables your team to share ideas and responsibilities, which helps reduce stress on everyone, allowing them to be meticulous and thorough when completing tasks.

Problem Solving Skills: It allow you to find candidates who are cognitively equipped to handle anything their jobs throw at them. Problem solvers can observe, judge, and act quickly when difficulties arise when they inevitably do.

Work Ethics: Workplace ethics ensures positive ambience at the workplace. Workplace ethics leads to happy and satisfied employees who enjoy coming to work rather than treating it as a mere source of burden.

Adaptability Skills: It expands your capacity to handle change, no matter how serious it might be. Instead of throwing away your energy trying to change your circumstance, you will change yourself right from within, thus making you thrive in whatever situation you find yourself.

Communication Skills: It is fundamental to the existence and survival of humans as well as to an organization. It is a process of creating and sharing ideas, information, views, facts, feelings, etc. among the people to reach a common understanding.

Responsibility: Each step we take towards being responsible and productive helps to raise our self-esteem and our relationships with friends, family and co-workers improve ten-fold.

Time Management: It helps you prioritize your tasks so that you ensure you have enough time available to complete every project. The quality of your work increases when you're not rushing to complete it ahead of a fast-approaching deadline.

Theoretical v/s Practical Knowledge

Practical knowledge is knowledge that is acquired by day-to-day hands-on experiences. In other words, practical knowledge is gained through doing things; it is very much based on real-life endeavours and tasks. On the other hand, theoretical knowledge teaches the reasoning, techniques, and theory of knowledge. While practical knowledge is gained by doing things, theoretical knowledge is gained, for example, by reading a manual.

While theoretical knowledge may guarantee that you understand the fundamental concepts and have know-how about how something works and its mechanism, it will only get you so far, as, without practice, one is not able to perform the activity as well as he could.

During this project period, I had only theoretical knowledge about some programming language and never build any thing like this before. But during this project period mentor assigned me a task for Drowsiness Detection and then I try to make that first I made mock-up and yes, it is my first mock up I had made ever. Then some research is important to work practically on a project. So, I did that and share that with our team members and get some review. We told each other some of our mistakes and then we correct them accordingly because this detector is used by many people, so some feedback is important, and we really get some positive feedback as well as some negative from our mentors too and all that are honest. So, on the negative feedback, we research again for that and end up with good user experience. By doing this we got lots of practical knowledge with theoretical knowledge.

Theoretical and practical knowledge are interconnected and complement each other — if one knows exactly HOW to do something, one must be able to apply these skills and therefore succeed in practical knowledge.

LIMITATIONS

- It did not detect the vehicle's state whether it is running or not. If the vehicle is not in a running state, then also detector detects the eyes of the driver.
- With helpless lighting conditions despite the fact that the face is effectively identified, now and again the framework can’t identify the eyes. So, it gives an erroneous result.
- At the point when a face is away from the webcam (more than 70cm) at that point, the blacklight is deficient to light up the face appropriately. So, eyes are not identified with high exactness which shows mistakes in the detection of drowsy.
- Problem with multiple faces. While detecting the face of the driver it may happen that the face of the person sitting in the back also detected which may cause problems.
- It is difficult to detect the eyes of that person who is wearing spectacles.

CONCLUSION

The project “Drowsiness Detection” aims to prevent road accidents.

Drowsy driving is a key factor in thousands of traffic accidents around the world. Driver drowsiness detection is a vehicle safety feature that helps to reduce accidents caused by drowsy driving. The project’s goal is to develop a solution for detecting driver drowsiness using OpenCV.

The Haar cascade classifier are used for locating the eyes. It is a machine learning approach for visual object detection.

Using OpenCV execution speed is fast than Matlab. It consumes less memory and interpretation time is less.

FUTURE SCOPE

- Night vision or near infrared camera can be used instead standard camera that can work even in low light condition and during night.
- An improvement in the algorithm can be made to detect the eyes much faster and results would be more accurate.
- This system can be further extended to have security like only certain people can access the vehicle.
- In case of theft, the vehicle does not start and an mms of the burglar could be sent to the owner of the vehicle.

REFERENCES

[1] Manu, B. N. 2016. Facial features monitoring for real time drowsiness detection. 2016 12th International Conference on Innovations in Information Technology (IIT).

[2] Alshaqaqi, B., Baquhaizel, A. S., Amine Ouis, M. E., Boumehed, M., Ouamri, A., & Keche, M. 2013. Driver drowsiness detection system. 2013 28th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA).

[3] Baek, J. W., Han, B.-G., Kim, K.-J., Chung, Y.-S., & Lee, S.-I. 2018. Real-Time Drowsiness Detection Algorithm for Driver State Monitoring Systems. 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN).

[4] You, F., Li, X., Gong, Y., Wang, H., & Li, H. 2019. A Real-time Driving Drowsiness Detection Algorithm with Individual Differences Consideration. IEEE Access, vol. 7, pp. 179396-179408.

[5] Dasgupta, A., Rahman, D., & Routray, A. 2018. A Smartphone-Based Drowsiness Detection and Warning System for Automotive Drivers. IEEE Transactions on Intelligent Transportation Systems.

[6] Eraldo, B., Quispe, G., Chavez-Arias, H., Raymundo-Ibanez, C., & Dominguez, F. 2019. Design of a control and monitoring system to reduce traffic accidents due to drowsiness through image processing. 2019 IEEE 39th Central America and Panama Convention (CONCAPAN XXXIX).

[7] Lashkov, I., Kashevnik, A., Shilov, N., Parfenov, V., & Shabaev, A. 2019. Driver Dangerous State Detection Based on OpenCV & Dlib Libraries Using Mobile Video Processing. 2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC).

[8] Zhang, S., & Wang, X. 2013. Human detection and object tracking based on Histograms of Oriented Gradients. 2013 Ninth International Conference on Natural Computation (ICNC).

[9] Kamran, M., Mannan, M., & Jeong, Y. 2019. Drowsiness, Fatigue and Poor Sleep's Causes and Detection: A Comprehensive Study.IEEE Access, vol. 7, pp. 167172-167186.

[10] Zhang, W., Cheng, B., & Lin, Y. 2012. Driver drowsiness recognition based on computer vision technology. Tsinghua Science and Technology, vol. 17, no. 3, pp. 354-362.

[11] S. Kaplan, M. A. Guvensan, A. G. Yavuz, and Y. Karalurt, "Driver behavior analysis for safe driving: A survey," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 6, pp. 3017–3032, 2015.

[12] M. Doudou, A. Bouabdallah, and V. Berge-Cherfaoui, "Driver drowsiness measurement technologies: Current research, market solutions, and challenges," International Journal of Intelligent Transportation Systems Research, pp. 1–23, 2019.

[13] M. Ramzan, H. U. Khan, S. M. Awan, A. Ismail, M. Ilyas, and A. Mahmood, "A survey on state-of-the-art drowsiness detection techniques," IEEE Access, vol. 7, pp. 61 904–61 919, 2019.

[14] X. Hu and G. Lodewijks, "Detecting fatigue in car drivers and aircraft pilots by using non-invasive measures: The value of differentiation of sleepiness and mental fatigue," Journal of Safety Research, vol. 72, pp. 173–187, 2020.

[15] A. Nemcov ˇ a, V. Svozilov ´ a, K. Bucsuh ´ azy, R. Sm ´ ´ısek, M. M ˇ ezl, ´ B. Hesko, M. Belak, M. Bil ´ ´ık, P. Maxera, and M. Seitl, "Multimodal features for detection of driver stress and fatigue," IEEE Transactions on Intelligent Transportation Systems, 2020.