MID-TERM REPORT

(2020-21)

Driver Drowsiness Detection System

Institute of Engineering & Technology

SupervisedBy:	**Submittedby:**
Priya Agrawal	Vivek Kumar Singh(181500819**)**
(Technical Trainer)	Jitendra Parmar(181500299)
	Abhishek Gupta (181500016)

Contents

Abstract

Driver fatigue is one of the major causes of accidents in the world. Detecting the drowsiness of the driver is one of the surest ways of measuring driver fatigue. In this project we aim to develop a prototype drowsiness detection system. This system works by monitoring the eyes of the driver and sounding an alarm when he/she is drowsy. The system so designed is a non-intrusive real-time monitoring system.

The priority is on improving the safety of the driver without being obtrusive. In this project the eye blink of the driver is detected. If the drivers eyes remain closed for more than a certain period of time, the driver is said to be drowsy and an alarm is sounded. The programming for this is done in OpenCV using the Haarcascade library for the detection of facial features.The report proposed the results and solutions on the limited implementation of the various techniques that are introduced in the project. Whereas the implementation of the project give the real world idea of how the system works and what changes can be done in order to improve the utility of the overall .

Furthermore, the paper states the overview of the observations made by the authors in order to help further optimization in the mentioned field to achieve the utility at a better efficiency for a safer road.

Keywords—Driver drowsiness eye detection.

Introduction

1.1General Introduction to the topic:

Driver fatigue is a significant factor in a large number of vehicle accidents. Recent statistics estimate that annually 1,200 deaths and 76,000 injuries can be attributed to fatigue related crashes.

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

The aim of this project is to develop a prototype drowsiness detection system. The focus will be placed on designing a system that will accurately monitor the open or closed state of the driver's eyes in real-time.

By monitoring the eyes, it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident. Detection of fatigue involves the observation of eye movements and blink patterns in a sequence of images of a face.

First we input the facial image using a webcam. Preprocessing was first performed by binarizing the image. The top and sides of the face were detected to narrow down the area where the eyes exist. Using the sides of the face, the center of the face was found which will be used as a reference when computing the left and right eyes. Moving down from the top of the face, horizontal averages of the face area were calculated. Large changes in the averages were used to define the eye area. There was little change in the horizontal average when the eyes were closed which was used to detect a blink.

For our project face and eye classifiers are required. So we used the learning objects method to create our own haarclassifier .xml files.

Around 2000 positive and 3000 negative samples are taken. Training them is a time intensive process. Finally face.xml and haarcascade-eye.xml files are created.

These xml files are directly used for object detection. It detects a sequence of objects (in our case face and eyes). Haarcascade-eye.xml is designed only for open eyes. So when eyes are closed the system doesn't detect anything. This is a blink. When a blink lasts for more than 5 frames, the driver is judged to be drowsy and an alarm is sounded.

What Is OpenCV :

OpenCV [OpenCV] is an open source computer vision library.detection. OpenCV is coded with optimized C and can take work with multicore processors. If we desire more automatic optimization using Intel architectures.

One of OpenCVs goals is to provide a simple-to-use computer vision infrastructure which helps people to build highly sophisticated vision applications fast. The OpenCV library, containing over 500 functions, spans many areas in vision. Because computer vision and machine learning oft en goes hand-in-hand.

This sub library is focused on statistical pattern recognition and clustering. The MLL is very useful for the vision functions that are the basis of OpenCV's usefulness, but is general enough to be used for any machine learning problem.

What Is Computer Vision:

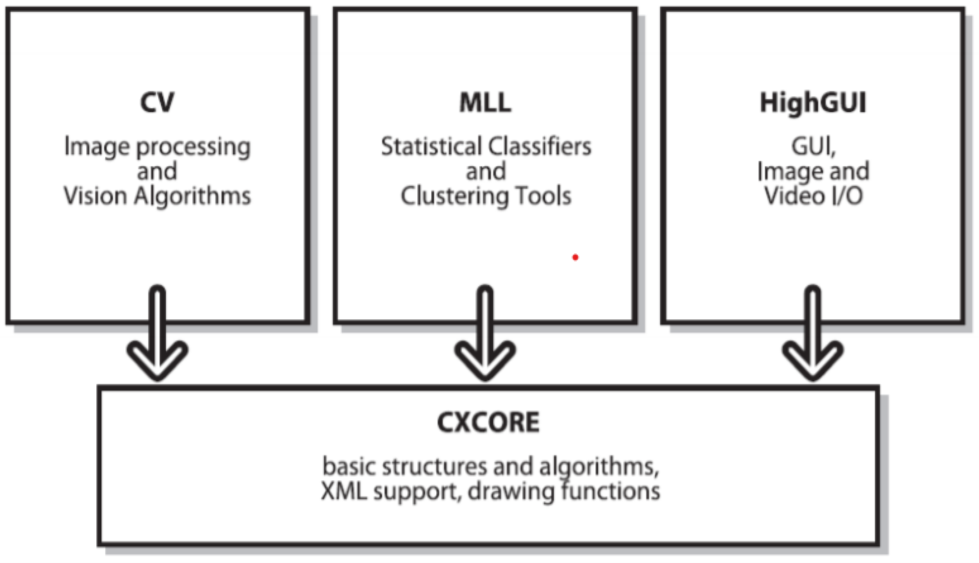
Computer vision is the transforming of data from a still, or video camera into either a representation or a new decision. All such transformations are performed to achieve a particular goal. A computer obtains a grid of numbers from a camera or from the disk, and that's that. Usually, there is no built in pattern recognition or automatic control of focus and aperture, no cross-associations with years of experience. For the most part, vision systems are still fairly .

The Origin of OpenCV :

OpenCV came out of an Intel Research initiative meant to advance CPU-intensive applications. Toward this end, Intel launched various projects that included real-time ray tracing and also 3D display walls. One of the programmers working for Intel at the time was visiting universities. He noticed that a few top university groups, like the MIT Media Lab, used to have well-developed as well as internally open computer vision infrastructures—code that was passed from one student to another and which gave each subsequent student a valuable foundation while developing his own vision application. Instead of having to reinvent the basic functions from beginning, a new student may start by adding to that which came before.

OpenCV Structure and Content :

OpenCV can be broadly structured into five primary components, four of which are shown in the figure. The CV component contains mainly the basic image processing and higher-level computer vision algorithms; MLL the machine learning library includes many statistical classifiers as well as clustering tools. HighGUI component contains I/O routines with functions for storing, loading video & images, while CXCore contains all the basic data structures and content.


CV
Image processing and Vision Algorithms
MLL
Statistical Classifiers and Clustering Tools
HighGUI
GUI, Image and Video I/O
CXCORE
basic structures and algorithms, XML support, drawing functions

WhyOpenCV:

Specific :

OpenCV was designed for image processing. Every function and data structure has been designed with an Image Processing application in mind. You can get almost everything in the world by means of toolboxes. It may be financial toolboxes or specialized DNA toolboxes.

Speedy :

Matlab is just way too slow. Matlab itself was built upon Java. Also Java was built upon C. So when we run a Matlab program, our computer gets busy trying to interpret and compile all that complicated Matlab code. Then it is turned into Java, and finally executes the code.

Efficient :

With OpenCV, we can get away with as little as 10mb RAM for a real-time application. Although with today's computers, the RAM factor isn't a big thing to be worried about. However, our drowsiness detection system is to be used inside a car in a way that is non-intrusive and small; so a low processing requirement is vital.
Thus we can see how OpenCV is a better choice than Matlab for a real-time drowsiness detection system.

Area of computer Science :

With the help of face detection we can easily identify whether a person are sleepy or not.

We know that various accidents are happen day by day it's only due to drowsy of driver. With the help of this technique we can easily detect whether driver is drowsy or not.

OpenCV [OpenCV] is an open source computer vision library.detection. OpenCV is coded with optimized C and can take work with multicore processors. If we desire more automatic optimization using Intel architectures.

One of OpenCVs goals is to provide a simple-to-use computer vision infrastructure which helps people to build highly sophisticated vision applications fast.

Computer vision is the transforming of data from a still, or video camera into either a representation or a new decision. All such transformations are performed to achieve a particular goal. A computer obtains a grid of numbers from a camera or from the disk, and that's that.
Usually, there is no built in pattern recognition or automatic control of focus and aperture, no cross-associations with years of experience. For the most part, vision systems are still fairly .

Hardware and Software Requirements :

a) Hardware:

- Minimum 2GB RAM
- i3 Processor
- Laptop with basic Hardware
- Webcam

b) Software:

- Pycharm
- Anconda
- Operating System(Window)
- Programming Language
  - Python with OpenCv

Problem Defination

Fatigue is a safety problem that has not yet been deeply tackled by any country in the world mainly because of its nature. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily. Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admit fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible
without the former as driving for long hours is very lucrative.
In United states from 1989-1993 approximately 100,000 crashes were reported by police per year, all the crashes are related to drowsiness. Fatality Analysis Reporting System (FARS) reported that around 71,000 of all crashes were non-fatal injuries & 1,357 resulted in mortality. Many of the road accidents were not reported & verified by police, because the problem is very large. Nowadays more accident occurs in trucks and cars than vehicles due to drowsiness. Nearly 97% of crashes of vehicles happen due to drowsiness of driver. It results into loss . for eg: human loss, money loss, medical loss. The accident or crashes not only affect the internal system but also to outside world. 70% injury occurs in internal system and 30% injury happen to the external system. Environmental loss is one of the disadvantage of accident. Accidents results in human as well as non human loss.
In the real time driver fatigue detection system it is required to slow down a vehicle automatically when fatigue level crosses a certain limit. Instead of threshold drowsiness level it is suggested to design a continuous scale driver fatigue detection system. It monitors the level of drowsiness continuously and when this level exceeds a certain value a signal is generated which controls the hydraulic braking system of the vehicle.

Objective

Drowsiness is a process where level of consciousness decrease due to lack of sleep or fatigue and can cause a person falls asleep. When driver is drowsy, the driver could lose control of the car so it was suddenly possible to deviate from the road and crashed into a barrier or a car.
Drowsiness detection techniques, in accordance with the parameters used for detection is divided into two sections i.e. intrusive method and a non-intrusive method. The main difference of these two methods is that the intrusive method.
An instrument connected to the driver and then the value of the instrument are recorded and checked. But intrusive approach has high accuracy, which is proportional to driver discomfort, so this method is rarely used.

Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving .The objective of this Python project is to build a drowsiness detection system that will detect that a person’s eyes are closed for a few seconds. This system will alert the driver when drowsiness is detected.

Driver drowsiness is a significant factor in the increasing number of accidents on today’s roads and has been extensively accepted. This proof has been verified by many researchers that have demonstrated ties between driver drowsiness and road accidents. Although it is hard to decide the exact number of accidents due to drowsiness, it is much likely to be underestimated.

The above statement shows the significance of a research with the objective of reducing the dangers of accidents anticipated to drowsiness. So far, researchers have tried to model the behavior by creating links between drowsiness and certain indications related to the vehicle and to the driver .

This paper represents a new way towards safety as well as security of automobiles. We are using the concept of eye recognition system, Drowsiness detection . Nowadays driver fatigue related crashes has increased. The main objective of our project is to develop nonintrusive system which will detect the fatigue or drowsiness of driver and will issue a warning with the help of alarm. As most of the accidents are caused due to drowsiness so this project will help to decrease the crashes or accidents. In this project we will detect the eye blinking with the help of webcam. If the eyes of the person are closed for more interval of time then this will result into the warning in the form of sound.

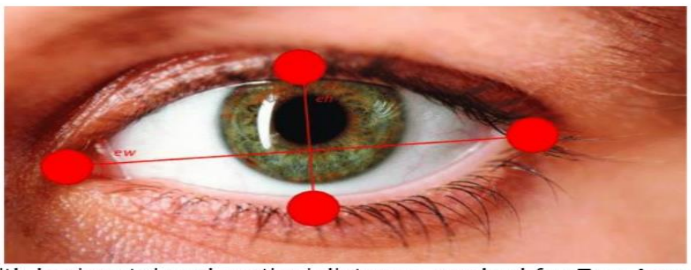
Implementation

In our program we used Dlib, a pre-trained program trained on thedataset to detect human faces using the pre-defined 68 landmarks.

After passing our video feed to the dlib frame by frame, we are able to detect left eye and right eye features of the face

Now, we drew contours around it using OpenCV

Using Scipy’s Euclidean function, we calculated sum of both eyes’ aspect ratio which is the sum of 2 distinct vertical distances between the eyelids divided by its horizontal distance.



Eyes with horizontal and vertical distance marked for Eye Aspect Ratio calculation.

Now we check if the aspect ratio value is less than 0.25 (0.25 waschosen as a base case after some tests). If it is less an alarm is sounded and user is warned

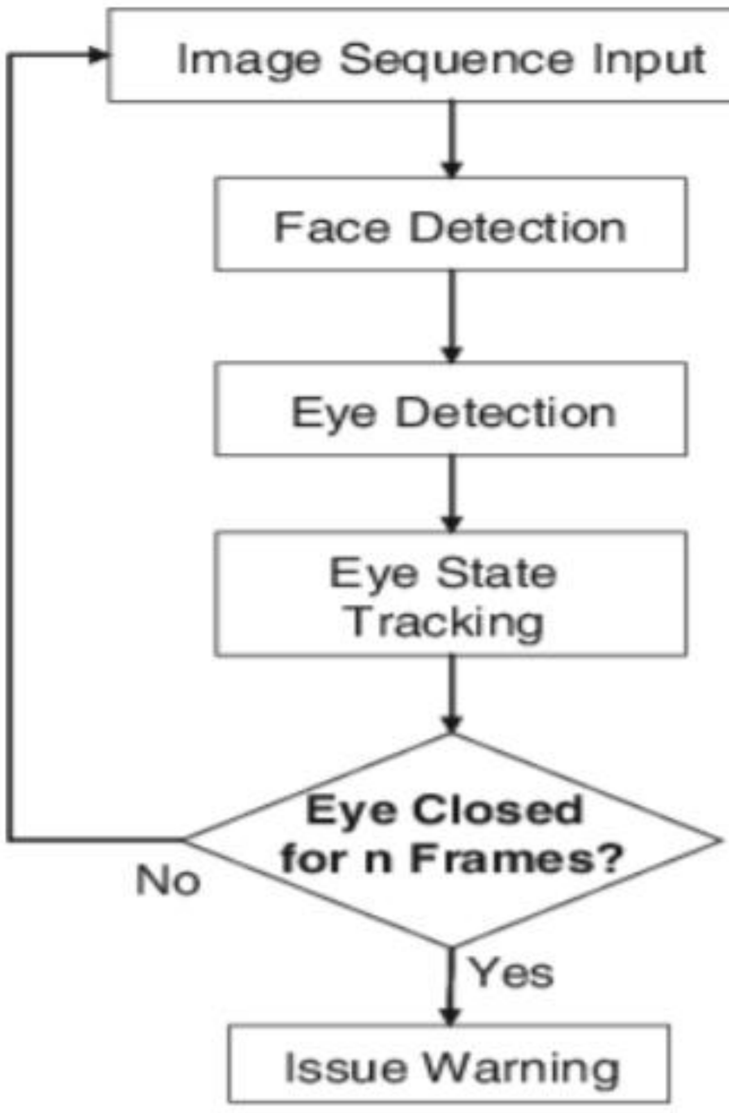


Image Sequence Input
Face Detection
Eye Detection
Eye State Tracking
Eye Closed for n Frames?
No
Yes
Issue Warning

Progress

1) Part 1 is completed :

- Importing Libraries:
  1) Numpay
  2) Scipy
  3) Playsound
  4) Dlib
  5) Imutils
  6) Opencv

- Starting to build the detector system with OpenCv
- Facial landmarks and eye aspect ratio calculation
- Important variables in the script
- dlib library for face detection

2) Part 2 is Remaining:

- Count the Number of eye blinks
- If eye closed the generate an alarm
- Test Cases to check the drowsiness

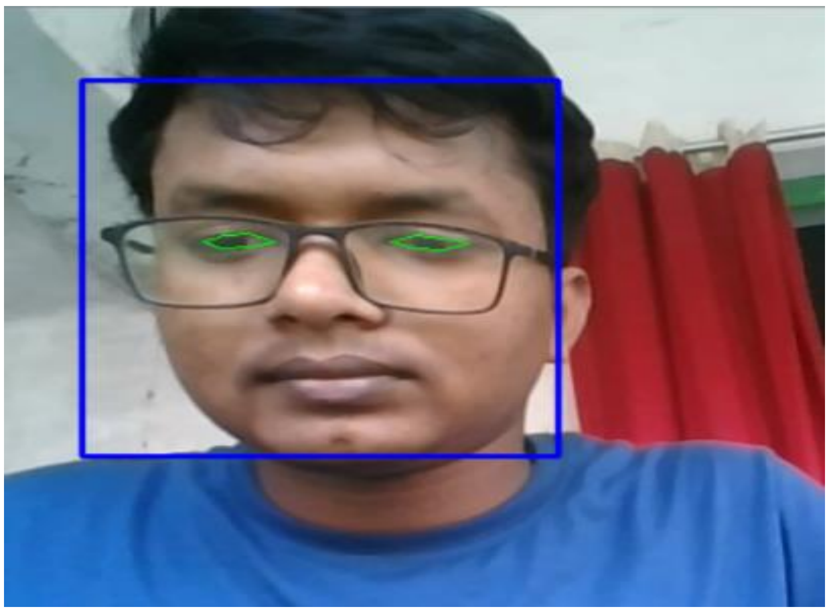
Coading Part

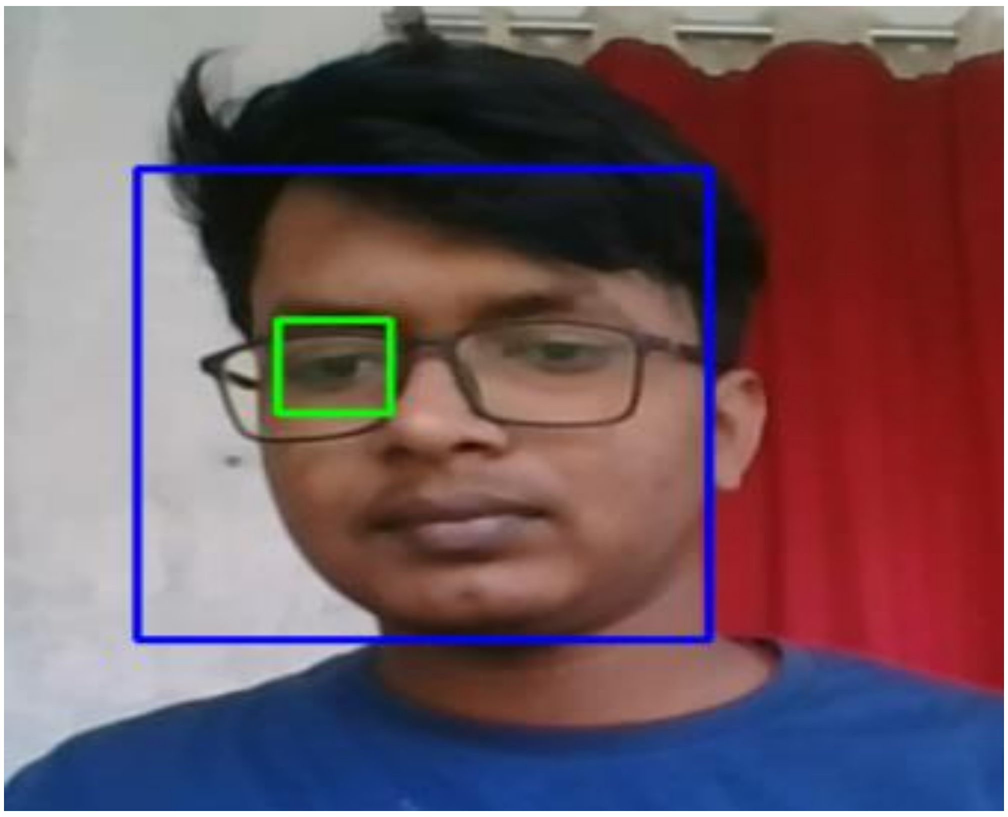
```
'''This Code detects whether a person is drowsy or not,usingdlib and
eye aspect ratio
calculations. Uses webcam video feed as input data.'''

# Imported necessary libraries
fromscipy.spatial import distance
fromimutils import face_utils
importnumpy as np
importpygame  # For playingsoundas alarm
import time
importdlib
import cv2

# Initialize Pygame and load music inourcode
pygame.mixer.init()
pygame.mixer.music.load('audio/alert.wav')

EYE_ASPECT_RATIO_THRESHOLD = 0.3

# Minimum consecutive frames for which eye ratio is below
threshold for alarm to be triggered
EYE_ASPECT_RATIO_CONSEC_FRAMES = 50

# COunts no. of consecutuve frames below threshold value
COUNTER = 0

# Load face cascade which will  used to draw a rectangle around
detected faces.
face_cascade =
cv2.CascadeClassifier("haarcascades/haarcascade_frontalface_defaul
t.xml")


# This function calculates eye aspect rationand return eye aspect
ratio
defeye_aspect_ratio(eye):
    A = distance.euclidean(eye[1], eye[5])
    B = distance.euclidean(eye[2], eye[4])
```

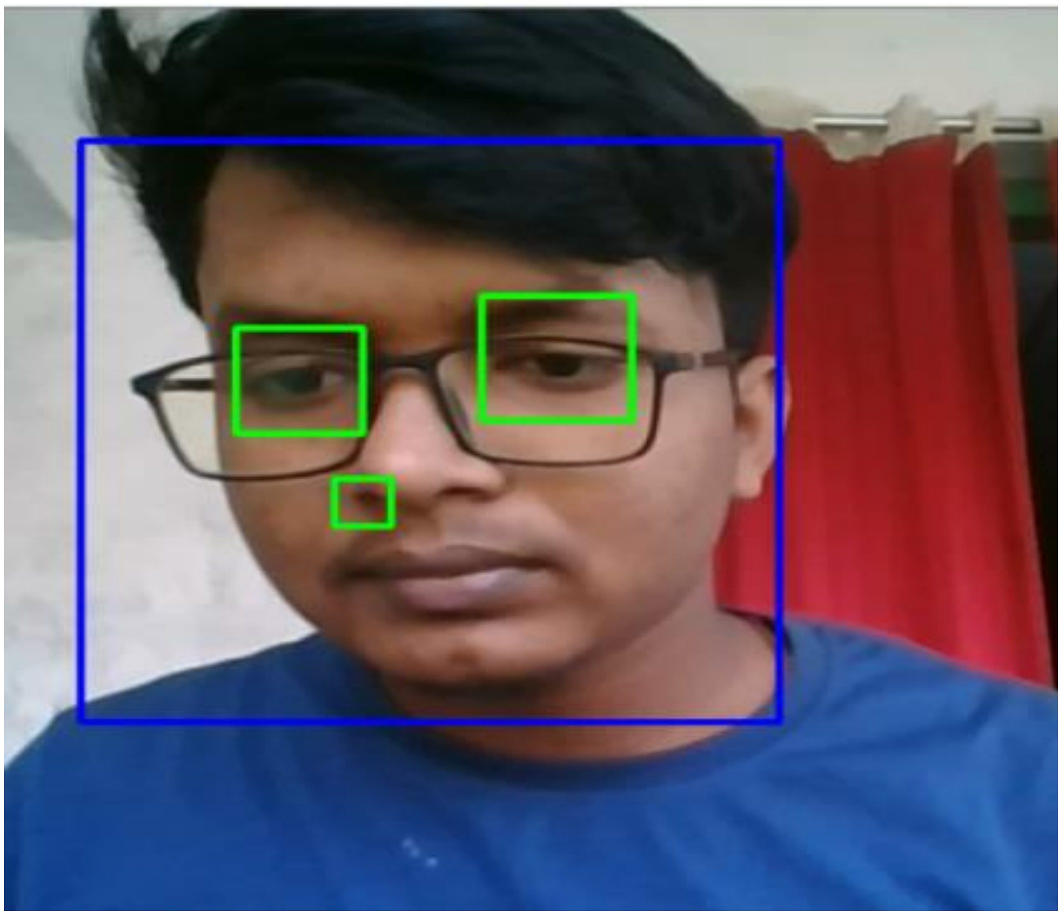
```
   C = distance.euclidean(eye[0], eye[3])

ear = (A + B) / (2 * C)
return ear


# Load face detector and predictor, uses dlib shape predictor file
detector = dlib.get_frontal_face_detector()
predictor =
dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

# Extract indexes of facial landmarks for the left and right eye
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS['left_eye']
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS['right_eye']

# Start webcam video capture
video_capture = cv2.VideoCapture(0)

# Give some time for camera to initialize(not required)
time.sleep(2)

while (True):
   # Read each frame and flip it, and convert to grayscale
ret, frame = video_capture.read()
frame = cv2.flip(frame, 1)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

   # Detect facial points through detector function
faces = detector(gray, 0)

   # Detect faces through haarcascade_frontalface_default.xml
face_rectangle = face_cascade.detectMultiScale(gray, 1.3, 5)

   # Draw rectangle around each face detected
for (x, y, w, h) in face_rectangle:
cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)

   # Detect the facial points
for face in faces:
else:
pygame.mixer.music.stop()
```

```
        COUNTER = 0

   # Show video feed
cv2.imshow('Video', frame)
if(cv2.waitKey(1) & 0xFF == ord('Q')):
break


destroyAllWindows
video_capture.release()
cv2.destroyAllWindows()
```

SCREENSHOT







REFERENCES

- ❖ Book References
  - Think Python: An Introduction to Software Design(Allen B. Downey)
  - OpenCV Computer Vision with Python(Joseph Howse)

- ❖ WWW.w3school.com
- ❖ WWW.javatpoint.com
- ❖ WWW.youtube.com
- ❖ www.tutorialspoint.com
- ❖ Faculty Guidelines
  - Priya Agrawal
  - Mr. Nikhil Govil