203109004 SELF PROJECT REPORT

GITHUB LINK for SELF PROJECT

https://github.com/apurvmishra22/DrowsinessDetection01

DROWSINESS DETECTION using Haar-Cascades OpenCV Machine Learning Algorithm

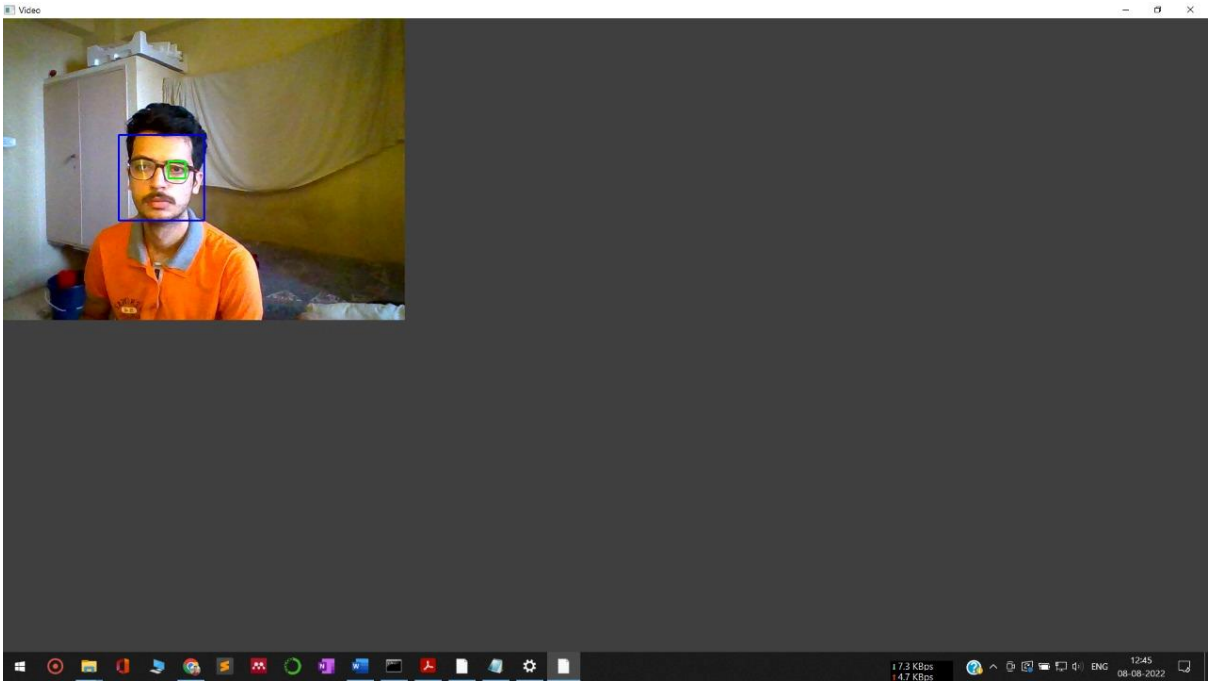
Computer vision is a technology that facilitates the computer to see the outside world. Computer vision is widely used for its security applications but nowadays finding its use in many other sectors. According to DOCTOR'S magazine, the most common method to detect the drowsiness is by monitoring the eye blink pattern of the driver. A webcam is kept in front of the driver in a way it doesn't obstruct the driver while driving. The webcam detects his face and then starts monitoring the eye blink pattern of the driver, it also indicates that his eyes are open

With the growth in population, the occurrence of automobile accidents has also seen an increase. A detailed analysis shows that, around half million accidents occur in a year , in India alone. Further , around 60% of these accidents are caused due to driver fatigue. Driver fatigue affects the driving ability in the following 3 areas, a) It impairs coordination, b) It causes longer reaction times, and, c)It impairs judgment. Through this project, we provide a real time monitoring system using image processing, face/eye detection techniques. Further, to ensure real-time computation, Haarcascade samples are used to differentiate between an eye blink and drowsy/fatigue detection. If the driver is detected to be drowsy for continuous 50 frames, an alert sound is played which will alert the driver instantaneously.

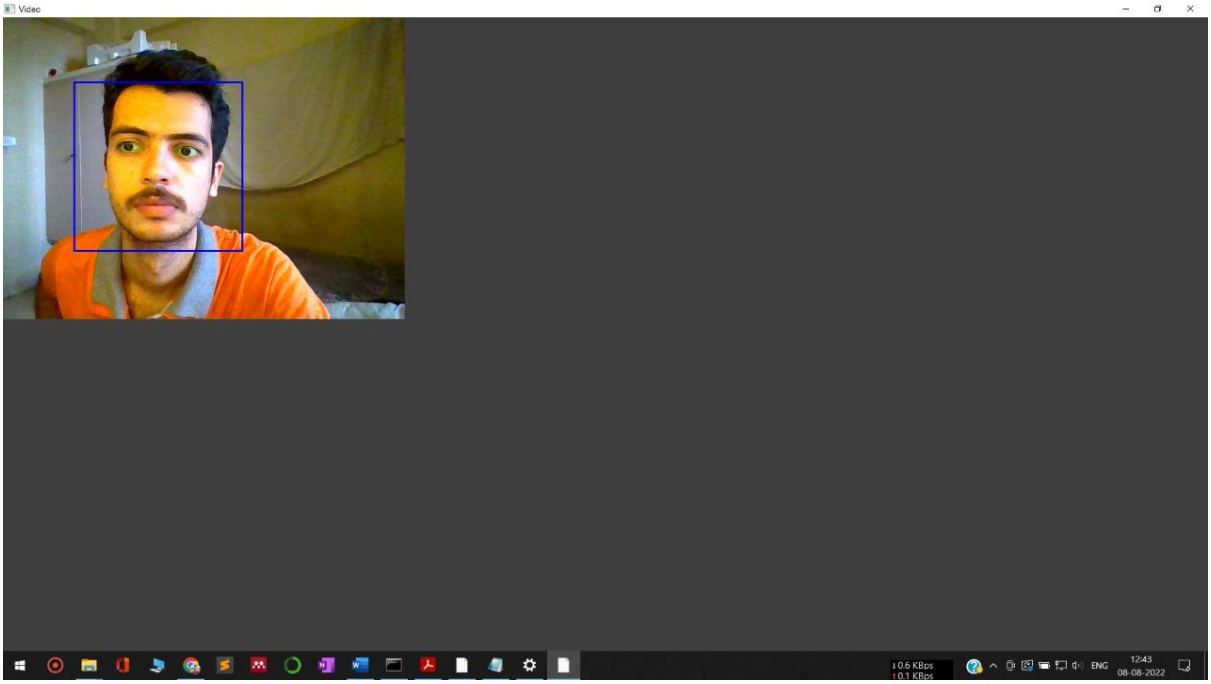
This project uses ML algorithm in order to determine the closeness of eyes and triggers an alarm which senses the drowsiness of alarm and wakes the driver.

OUTPUT

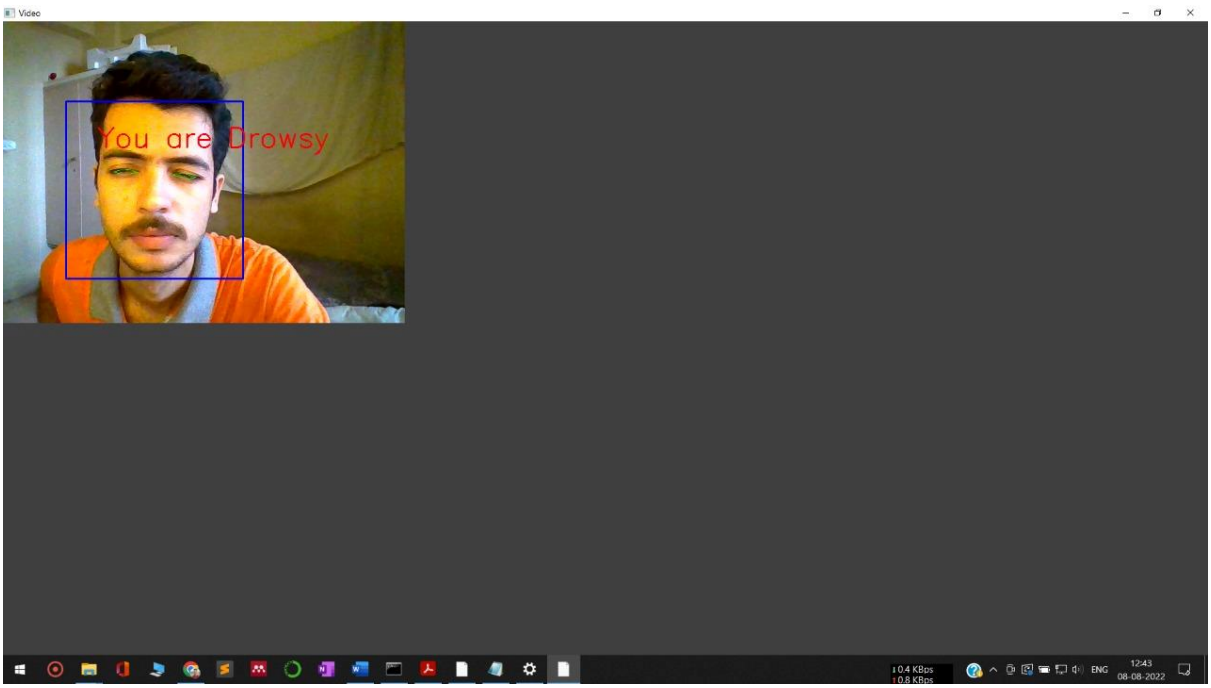
1.) Real time Face and Eye detection using Live Webcam Feed



2.) Since the driver is awake, no alarm is triggered



3.) As the driver becomes drowsy/sleepy, the ML algorithm identifies the sleepy driver, if it remains so for 50 consecutive frames, an alarm will be triggered.



In [2]:

```
#      To identify faces and eyes in a video feed that can be inputted through a webcam,
# this script utilises OpenCV's haarcascade (face and eye cascade).


#Import necessary libraries
import cv2 as cv
import numpy as np

#Load face cascade and hair cascade from haarcascades folder
face_cascade = cv.CascadeClassifier("haarcascades/haarcascade_frontalface_default.xml")
eye_cascade = cv.CascadeClassifier("haarcascades/haarcascade_eye.xml")

#Capture video from webcam
video_capture = cv.VideoCapture(0)

#Read all frames from webcam
while True:
    ret, frame = video_capture.read()
    frame = cv.flip(frame,1) #Flip so that video feed is not flipped, and appears mirror li
    gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:
        cv.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h, x:x+w]

        eyes = eye_cascade.detectMultiScale(roi_gray)

        for (ex,ey,ew,eh) in eyes:
            cv.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

    cv.imshow('Video', frame)

    if(cv.waitKey(1) & 0xFF == ord('q')):
        break

#Finally when video capture is over, release the video capture and destroyAllWindows
video_capture.release()
cv.destroyAllWindows()



# Submitted by 203109004
```

In [2]:

```
pip install imutils
```

```
Collecting imutils
  Downloading imutils-0.5.4.tar.gz (17 kB)
Building wheels for collected packages: imutils
  Building wheel for imutils (setup.py): started
  Building wheel for imutils (setup.py): finished with status 'done'
  Created wheel for imutils: filename=imutils-0.5.4-py3-none-any.whl size=25
862 sha256=9895ff8ea9e61f9a0e5d307d333bb7d2dbab3aaea10930dc41499cdec7e0ed00
  Stored in directory: c:\users\apurv\appdata\local\pip\cache\wheels\59\1b\5
2\0dea905f8278d5514dc4d0be5e251967f8681670cadd3dca89
Successfully built imutils
Installing collected packages: imutils
Successfully installed imutils-0.5.4
Note: you may need to restart the kernel to use updated packages.
```

In [4]:

```
pip install pygame
```

```
Collecting pygame
  Downloading pygame-2.1.2-cp38-cp38-win_amd64.whl (8.4 MB)
Installing collected packages: pygame
Successfully installed pygame-2.1.2
Note: you may need to restart the kernel to use updated packages.
```

In [3]:

```
# This script uses Live webcam feed to track the drowsiness of driver

#Import necessary libraries
from scipy.spatial import distance
from imutils import face_utils
import numpy as np
import pygame #For playing sound
import time
import dlib
import cv2

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

#Minimum threshold of eye aspect ratio below which alarm is triggerd
EYE_ASPECT_RATIO_THRESHOLD = 0.3

#the minimum number of consecutive frames for which the eye ratio is below the alarm-trigge
EYE_ASPECT_RATIO_CONSEC_FRAMES = 50

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

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

#This function calculates and return eye aspect ratio
def eye_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 facial points
    for face in faces:

        shape = predictor(gray, face)
        shape = face_utils.shape_to_np(shape)

        #Get array of coordinates of leftEye and rightEye
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]

        #Calculate aspect ratio of both eyes
        leftEyeAspectRatio = eye_aspect_ratio(leftEye)
        rightEyeAspectRatio = eye_aspect_ratio(rightEye)

        eyeAspectRatio = (leftEyeAspectRatio + rightEyeAspectRatio) / 2

        #Use hull to remove convex contour discrepencies and draw eye shape around eyes
        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)

        #Detect if eye aspect ratio is less than threshold
        if(eyeAspectRatio < EYE_ASPECT_RATIO_THRESHOLD):
            COUNTER += 1
            #If no. of frames is greater than threshold frames,
            if COUNTER >= EYE_ASPECT_RATIO_CONSEC_FRAMES:
                pygame.mixer.music.play(-1)
                cv2.putText(frame, "You are Drowsy", (150,200), cv2.FONT_HERSHEY_SIMPLEX, 1
        else:
            pygame.mixer.music.stop()
            COUNTER = 0

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

#Finally when video capture is over, release the video capture and destroyAllWindows
video_capture.release()
cv2.destroyAllWindows()

# Submitted by 203109004
```