Bachelor of Technology (B. Tech.)

Computer and Communication Engineering (CCE)

Amrita School of Engineering

Coimbatore Campus (India)

Academic Year – 2022 - 23

Team 13

S. No	Name	Roll Number
1	KR Abishekraswanth	CB.EN.U4CCE20001
2	Akshara R	CB.EN.U4CCE20003
3	Hemchand R	CB.EN.U4CCE20024
4	R Rangashree Dhanvanth	CB.EN.U4CCE20048

Fifth Semester

19CCE301 Internet of Things (IoT) Project

IoT-Based Driver Drowsiness Detection using Raspberry Pi

Faculty In-charge – Peeyush K P Sir

DRIVER DROWZINESS DETECTION USING RASPBERRY PI

ABSTRACT: This proposed system is used for Driver & Road safety system. Based on computer vision techniques, the driver's face is located from a colour video captured in a car. Then, face detection is employed to locate the regions of the driver's eyes, which are used as the templates for eye tracking in subsequent frames. The tracked eye's images are used for drowsiness detection in order to generate warning alarms. The proposed approach has three phases: Face, Eye detection and drowsiness detection. The role of image processing is to recognize the face of the driver and then extracts the image of the eyes of the driver for detection of drowsiness. Hence, we conclude this approach is a low cost and effective solution to reduce the number of accidents due to driver's Drowsiness to increase the transportation safety.

INTRODUCTION:

Drowsy driving is one of the major causes behind fatal road accidents. One of the recent studies shows that one out of five road accidents are caused by drowsy driving which is roughly around 21% of road accidents, and this percentage is increasing every year as per global status report on road safety 2015, based on the data from 180 different countries. This certainly highlights the fact that across the world the total numbers of road traffic deaths are very high due to driver's drowsiness. Driver fatigue, drink-and-drive and carelessness are coming forward as major reasons behind such road accidents. Many lives and families are getting affected due to this across various countries. Real time drowsy driving detection is one of the best possible majors that can be implemented to assist drivers to make them aware of drowsy driving conditions. Such driver behavioural state detection system can help in catching the driver drowsy conditions early and can possibly avoid mishaps.

With this report, we are presenting technique to detect driver drowsiness using of Open CV, raspberry pi and image processing. Several studies have shown various possible techniques that can detect the driver drowsiness. Among these physiological measure and ocular measure can give more accurate results. But this leads to discomfortable driving conditions. But ocular measure can be done without physical connection. Ocular measure to detect driver eye condition and possible vision based on eye closure is well suited for real world driving conditions, since it can detect the eyes open/ closed state non intrusively using a camera.

In Real Time Driver Drowsiness System using Image Processing, capturing drivers eye state using computer vision-based drowsiness detection systems have been done by analysing the interval of eye closure and developing an algorithm to detect the driver's drowsiness in advance and to warn the driver by in vehicles alarm. This section motivates how face is detected and how eye detection is performed for automotive application and their detection is necessary for assessing driver drowsiness.

METHODOLOGY:

Face landmark detection is a computer vision task where we want to detect and track key points from a human face. This task applies to many problems. The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face.

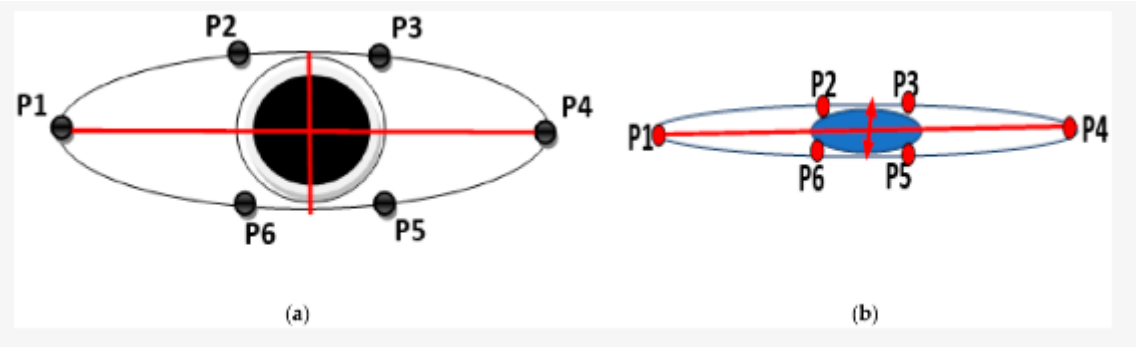
The indexes of the 68 coordinates can be visualized on the image below:

To detect the drowsiness of a person we need to calculate a special type of ratio known as EAR - Eye Aspect Ratio.

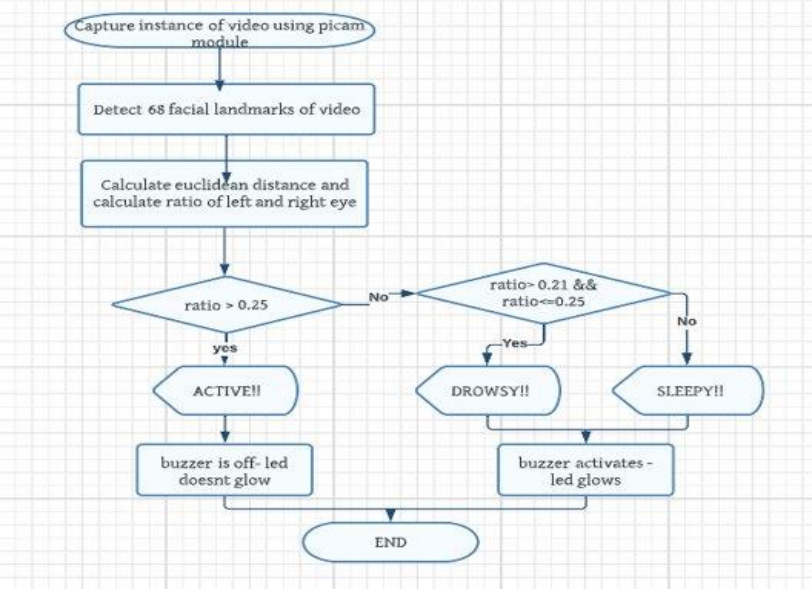
The Eye Aspect Ratio, or EAR, is a scalar value that responds, particularly for opening and closing the eyes. Each frame of the video stream is used to estimate the EAR. Furthermore, when the user shuts their eyes, the EAR drops and then returns to a regular level when the eyes are opened again. This technique is used to determine both blinks and eye opening. The EAR value can be calculated by entering six coordinates surrounding the eyes.

$$EAR = \frac{\|P2 - P6\| + \|P3 - P5\|}{2\|P1 - P4\|}$$

$$AVG\ EAR = \frac{1}{2}(EAR_{Left} + EAR_{Right})$$



The working of the model is shown in the below figure. In this the video of the driver is captured using picamera module in raspberry pi, using the 68 landmark detection the eye movements are captured and EAR is calculated, if the EAR ratio is beyond the threshold it means the person is active, else it is concluded that the driver is drowsy.



HARDWARE COMPONENTS:

A. Raspberry Pi 3:

The Raspberry Pi 3 is equipped with a quad-core 64-bit Broadcom BCM2837 ARM Cortex-A53 SoC processor running at 1.2 GHz.

SPECIFICATIONS:

1. Clock frequency: 1.2 GHz
2. Chipset (SoC): Broadcom BCM2837
3. Processor: 64-bit quad-core ARM Cortex-A53
4. Graphics processor: Broadcom Dual Core VideoCore IV (OpenGL ES 2.0, H.264 Full HD @ 30 fps)
5. Memory (SDRAM): 1 GB LPDDR2
6. Number of USB 2.0 ports: 4
7. Port extension: 40-pin GPIO
8. Video outputs: HDMI and RCA, plus 1 CSI camera connector
9. Audio outputs: 3.5 mm stereo jack or HDMI
10. Data storage: MicroSD card
11. Network connection: 10/100 Ethernet, 802.11n WiFi and Bluetooth 4.1 (BLE - Low Energy)
12. Peripherals: 17 x GPIO
13. Supply: 5V 2.5A via micro USB

B. Pi Camera Module:

The Raspberry Pi Camera Board is a custom designed add-on module for Raspberry Pi hardware. It attaches to Raspberry Pi hardware through a custom CSI interface. The sensor has a 5-megapixel native resolution in still capture mode.

C. Micro USB Cable:

The first, recommended and easiest way to power the Raspberry Pi is via the Micro USB port on the side of the unit. The recommended input voltage is 5V, and the recommended input current is 2A.

D. Buzzer:

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.

E. LCD – 16x2 character lcd:

A 16x2 LCD display is a very basic module and is very commonly used in various devices and circuits. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines.

LIBRARIES USED:

A. OpenCV and Dlib:

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Dlib is used for face detection and facial landmark detection**.**

B. Face recognition:

The face recognition algorithm is used in finding features that are uniquely described in the image. The facial image is already extracted, cropped, resized, and usually converted in the grayscale.

C. Face_utils:

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 .

PYTHON CODE:

```
#Importing OpenCV Library for basic image processing functions
import cv2
# Numpy for array related functions
import numpy as np
# Dlib for deep learning-based Modules and face landmark detection
import dlib
#face_utils for basic operations of conversion
from imutils import face_utils
import time

import board
import digitalio
import adafruit_character_lcd.character_lcd as characterlcd

import RPi.GPIO as GPIO     # Import Library to access GPIO PIN
GPIO.setmode(GPIO.BCM)    # Consider complete raspberry-pi board
GPIO.setwarnings(False)

buzzer_pin = 16             # Define PIN for LED
LED_PIN = 5

lcd_rs = digitalio.DigitalInOut(board.D4)
lcd_en = digitalio.DigitalInOut(board.D17)
lcd_d7 = digitalio.DigitalInOut(board.D26)
lcd_d6 = digitalio.DigitalInOut(board.D22)
lcd_d5 = digitalio.DigitalInOut(board.D27)
lcd_d4 = digitalio.DigitalInOut(board.D18)
GPIO.setup(buzzer_pin,GPIO.OUT)   # Set pin function as output
GPIO.setup(LED_PIN,GPIO.OUT)   # Set pin function as output
# Define some device constants
GPIO.output(buzzer_pin,GPIO.LOW)

lcd_columns = 16
lcd_rows = 2

lcd = characterlcd.Character_LCD_Mono(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns,
lcd_rows)

#Initializing the camera and taking the instance
cap = cv2.VideoCapture(0)
```

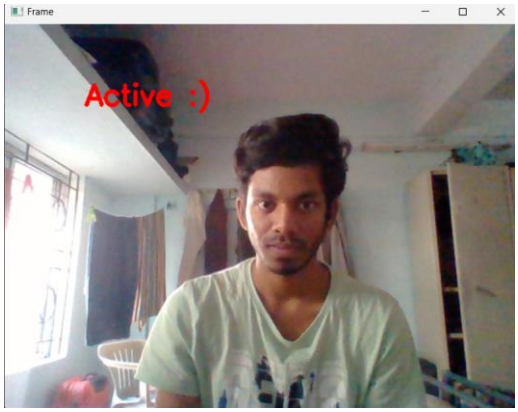
```
#Initializing the face detector and landmark detector
hog_face_detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("/home/pi/Documents/group 13
project/shape_predictor_68_face_landmarks.dat")

#status marking for current state
sleep = 0
drowsy = 0
active = 0
status=""
color=(0,0,0)

def compute(ptA,ptB):
    dist = np.linalg.norm(ptA - ptB)
    return dist

def blinked(a,b,c,d,e,f):
    up = compute(b,d) + compute(c,e)
    down = compute(a,f)
    ratio = up/(2.0*down)

    #Checking if it is blinked
    if(ratio>0.25):
        return 2
    elif(ratio>0.21 and ratio<=0.25):
        return 1
    else:
        return 0


lcd.cursor = True
lcd.message="welcome "#send msg to my LCD
time.sleep(2) #delay5 seconds
lcd.message="Driver Sleep\nDetection System"#send msg to my LCD
time.sleep(2) #delay

while True:

    _, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
faces = hog_face_detector(gray)
#detected face in faces array
for face in faces:
    x1 = face.left()
    y1 = face.top()
    x2 = face.right()
    y2 = face.bottom()
    face_frame = frame.copy()
    cv2.rectangle(face_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

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

    #The numbers are actually the landmarks which will show eye
    left_blink = blinked(landmarks[36],landmarks[37],
    landmarks[38], landmarks[41], landmarks[40], landmarks[39])
    right_blink = blinked(landmarks[42],landmarks[43],
    landmarks[44], landmarks[47], landmarks[46], landmarks[45])

    #Now judge what to do for the eye blinks
    if(left_blink==0 or right_blink==0):
        sleep+=1
        drowsy=0
        active=0
        if(sleep>1):
            status="SLEEPING !!"
            print("SLEEPING !!")
            GPIO.output(buzzer_pin,GPIO.HIGH)
            GPIO.output(LED_PIN,GPIO.HIGH)
            lcd.message='Please Wake up!'
            time.sleep(1)
            color = (255,0,0)
            lcd.clear()

    elif(left_blink==1 or right_blink==1):
        sleep=0
        active=0
        drowsy+=1
        if(drowsy>1):
            status="Drowsy :("
            print("DROWSY :(")
            GPIO.output(buzzer_pin,GPIO.HIGH)
```

```
                GPIO.output(LED_PIN,GPIO.HIGH)
                lcd.message='Dont be drowsy!'
                time.sleep(1)
                color = (0,0,255)
                lcd.clear()

        else:
            drowsy=0
            sleep=0
            active+=1
            if(active>1):
                status="Active :)"
                print("Active :)")
                lcd.message='All okay!\nDriver safe!'
                time.sleep(1)
                GPIO.output(buzzer_pin,GPIO.LOW)
                GPIO.output(LED_PIN,GPIO.LOW)
                color = (0,255,0)
                lcd.clear()

        cv2.putText(frame, status, (100,100), cv2.FONT_HERSHEY_SIMPLEX, 1.2, color,3)

        for n in range(0, 68):
            (x,y) = landmarks[n]
            cv2.circle(face_frame, (x, y), 1, (255, 255, 255), -1)

    cv2.imshow("Frame", frame)
    #cv2.imshow("Result of detector", face_frame)
    key = cv2.waitKey(1)
    if key == 27:
        break
```

RESULTS:

WHEN DRIVER IS ACTIVE:

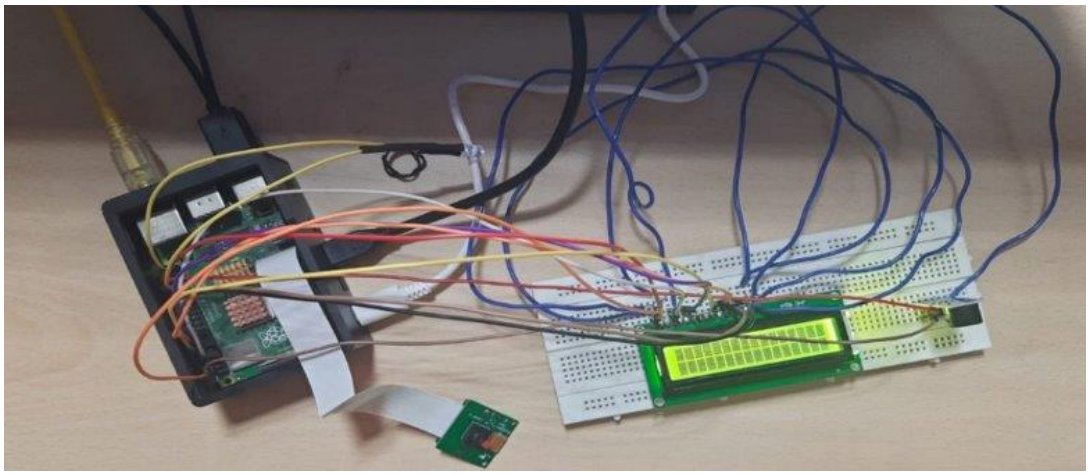


WHEN DRIVER IS DROWSY:

WHEN DRIVER IS SLEEPY:

If the driver is active, the LCD displays the corresponding message and the buzzer is not on. If the driver is drowsy, the LCD displays message that he is drowsy and the buzzer gets active to alert and wake him up, the same applies to when the driver is sleeping

PROJECT DESIGN/MODEL:



CONCLUSION:

The driver abnormality monitoring system developed can detect drowsiness, drunken and reckless behaviours of driver in a short time. The Drowsiness Detection System developed based on eye closure of the driver can differentiate normal eye blink and drowsiness and detect the drowsiness while driving.

The proposed system can prevent the accidents due to the sleepiness while driving. The system works well even in case of drivers wearing spectacles and even under low light conditions if the camera delivers better output. Information about the head and eyes position is obtained through various self-developed image processing algorithms. During the monitoring, the system can decide if the eyes are opened or closed. When the eyes have been closed for too long, a warning signal is issued. Processing judges the driver's alertness level based on continuous eye closures.

REFERENCES:

[1] https://circuitdigest.com/microcontroller-projects/driver-drowsiness-detector-using-raspberry-pi-and-opencv#:~:text=Testing%20the%20Driver%20Drowsiness%20Detection%20System&text=After%20Approx%2010%20seconds%2C%20a,seconds%20to%20test%20the%20alarm.

[2] https://nevonprojects.com/driver-drowsiness-detection-system-for-accident-prevention/

[3] https://www.youtube.com/watch?v=ksi42rwGyas