Mini Project Report On

Drowsiness Detection System

BY

PRAVEEN KUMAR A (111421104091)

VIJAYA KUMAR C (111421104118)

VINAY B L (111421104119)

SIVASANKAR J M (111421104107)

IN PARTIAL FULFILLMENT FOR THE AWARD OF
DEGREE OF BACHELOR OF ENGINEERING
IN

COMPUTER SCIENCE AND ENGINEERING

PRATHUSHA ENGINEERING COLLEGE
(AUTONOMOUS INSTITUTE)
THIRUVALLUR - 602 025

Table of contents:

1.Abstract:

Drowsiness detection is crucial for preventing road accidents caused by driver fatigue. This paper presents a real-time drowsiness detection system using eye aspect ratio (EAR). The system utilizes the dlib library to detect facial landmarks and the scipy library to calculate distances between points. It continuously monitors the EAR of the driver's eyes and triggers an audio alert when drowsiness is detected. The system effectively identifies drowsiness and provides timely warnings to help enhance road safety.

2.Introduction:

Drowsiness is a major cause of road accidents, accounting for a significant portion of crashes each year. To address this issue, we developed a real-time drowsiness detection system that utilizes eye aspect ratio (EAR) as the primary indicator of fatigue. EAR is a simple and effective measure of eye openness, with values close to zero indicating closed eyes and values closer to one indicating open eyes. By continuously monitoring the EAR of the driver's eyes, our system can effectively detect drowsiness and provide timely alerts to help prevent accidents.

3.System Requirements:

The system's primary requirements include:

- Real-time monitoring: Continuous tracking of the driver's eye movements is essential for timely drowsiness detection.

- Accurate EAR calculation: Precise evaluation of EAR values is critical to distinguish between open and closed eyes.

- Drowsiness detection threshold: A reliable threshold should be established to differentiate between normal eye blinking and prolonged drowsiness.

- Alert mechanism: An effective alert system, such as an audio signal, should be implemented to notify the driver of drowsiness.

4.Importing Libraries:

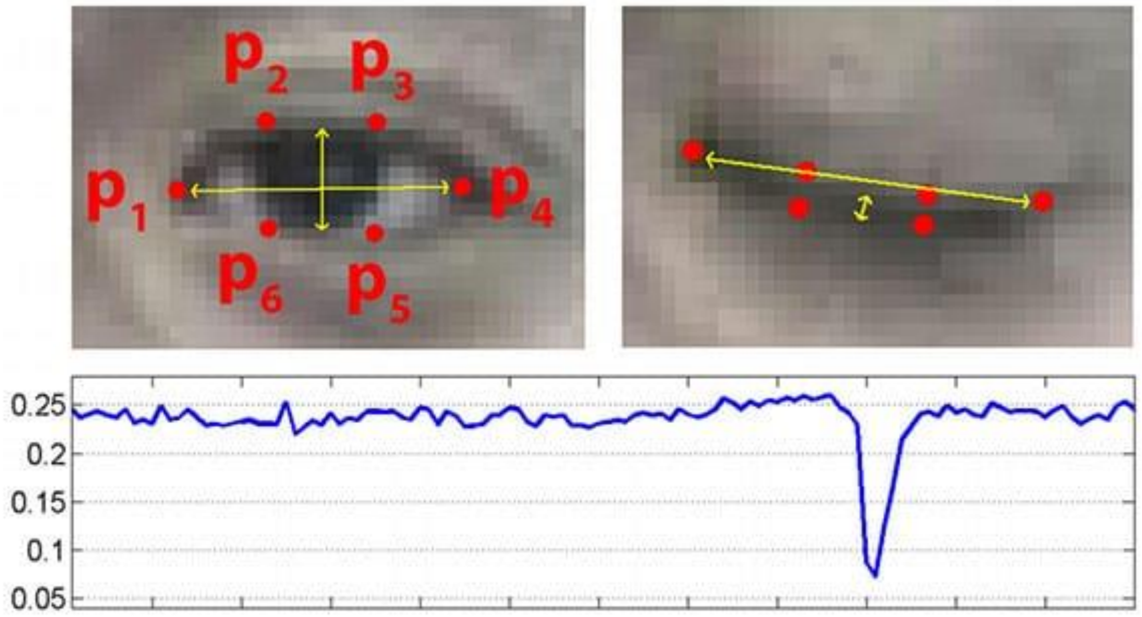
- **Eye aspect ratio:**
  Function to calculate the EAR of an eye.
- **thresh:**
  Threshold value to determine drowsiness (EAR < 0.25).
- **frame check:**
  Number of frames to consider for drowsiness detection.
- **detect:**
  dlib's frontal face detector.
- **predict:**
  dlib's shape predictor for facial landmarks.
- **(lStart, lEnd) and (rStart, rEnd):**
  Index ranges for left and right eye landmarks.

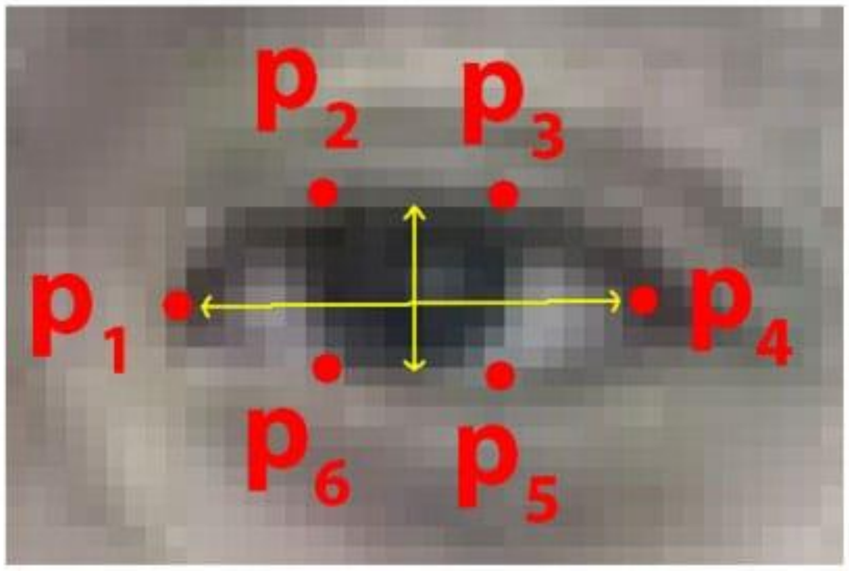
$$\mathrm{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

5.Drowsiness Detection Loop:

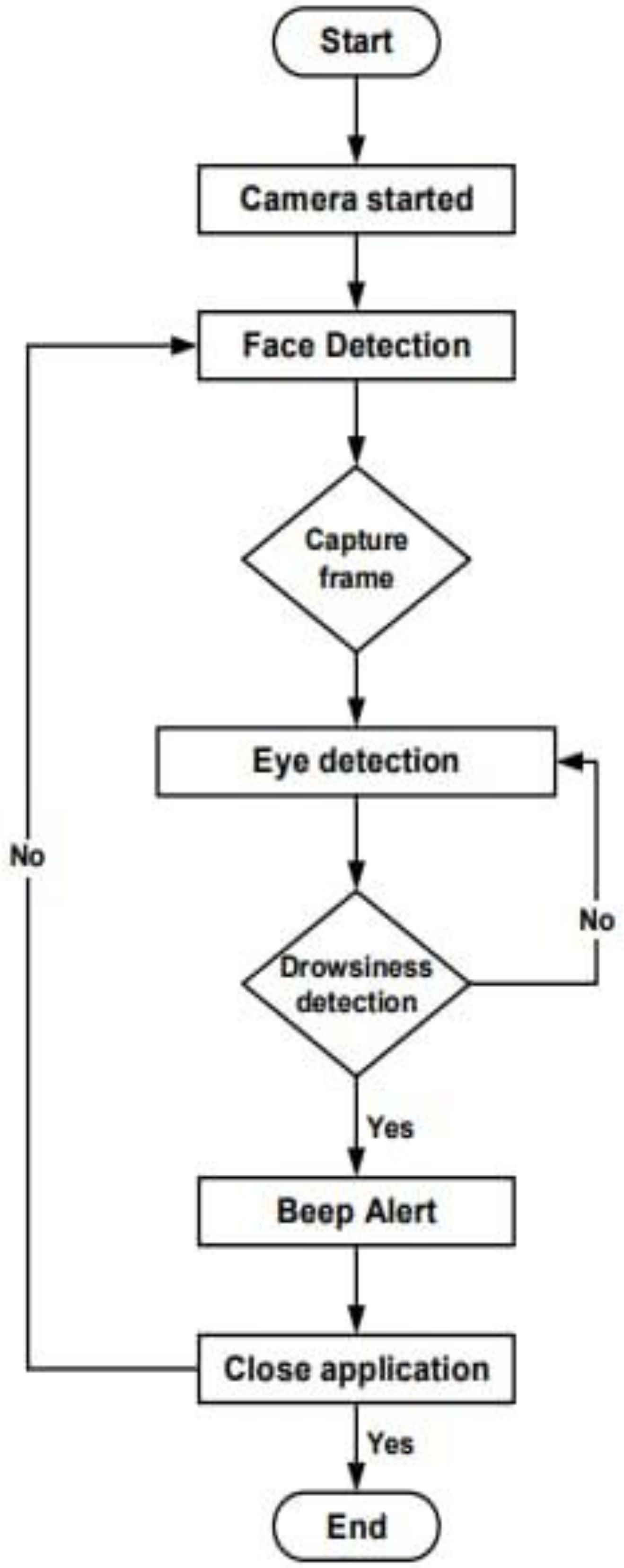
- **cap**: Video capture object.
- **flag**: Counter for consecutive frames with low EAR.
- **while loop**: Captures frames, detects faces, and analyzes eye aspect ratios.
- **ret, frame**: Read a frame from the video capture.
- **frame = imutils.resize(frame, width=450)**: Resize the frame for efficient processing.
- **gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY**): Convert the frame to grayscale.
- **subjects = detect(gray, 0)**: Detect faces in the grayscale frame.
- **for subject in subjects**: Process each detected face.
- **shape = predict(gray, subject)**: Predict facial landmarks for the detected face.
- **shape = face_utils.shape_to_np(shape)**: Convert the shape object to a NumPy array.
- **leftEye and rightEye** : Extract left and right eye landmark points.
- **leftEAR and rightEAR**: Calculate EAR for both eyes.
- **ear = (leftEAR + rightEAR) / 2.0**: Calculate average EAR.
- **leftEyeHull and rightEyeHull**: Create convex hulls around the eye landmarks.
- **cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)**: Draw contours around left eye.
- **cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1**): Draw contours around right eye.
- **if ear < thresh**: Check if average EAR is below the threshold.
- **flag += 1:** Increment the flag counter.
- **if flag >= frame_check:** Check if flag exceeds the threshold for consecutive frames.
- **cv2.putText(frame, ...):** Display drowsiness alert messages.
- **mixer.music.play():**Play the audio alert.
- **else**: Reset the flag counter if EAR is above the threshold.
- **cv2.imshow("Frame", frame):** Display the processed frame.
- **key = cv2.waitKey(1) & 0xFF**: Wait for a key press.

- **if key == ord("q"):** Break the loop if 'q' is pressed.
- **cv2.destroyAllWindows():** Close all windows.
- **cap.release():** Release the video capture object.





6.Flowchart representation:



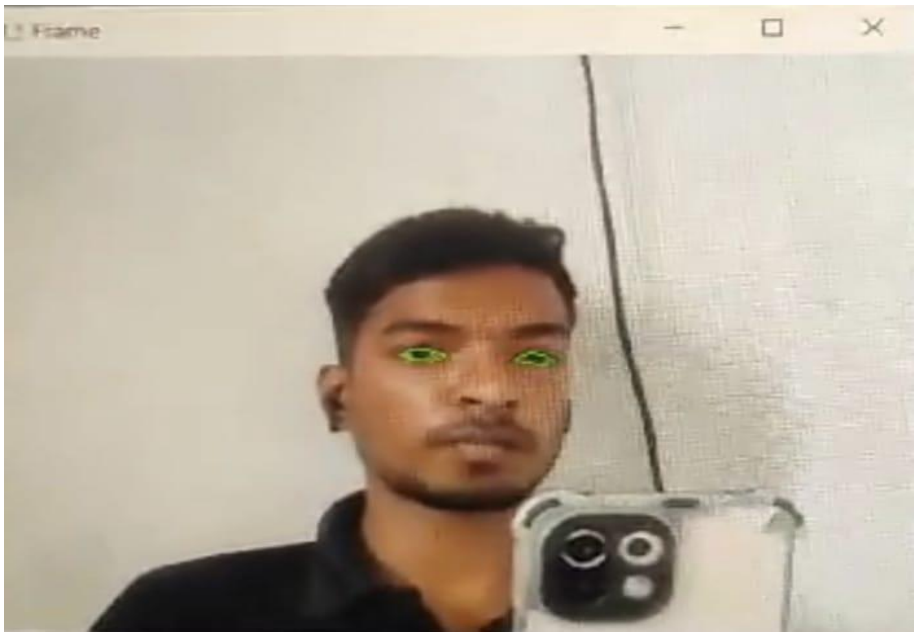
7.Program:

```
from scipy.spatial import distance
from imutils import face_utils
from pygame import mixer
import imutils
import dlib
import cv2


mixer.init()
mixer.music.load("music.wav")

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) / (2.0 * C)
return ear

thresh = 0.25
frame_check = 20
detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor("models/shape_predictor_68_face_landmarks.dat")

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
cap=cv2.VideoCapture(0)
flag=0
while True:
ret, frame=cap.read()
frame = imutils.resize(frame, width=450)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
subjects = detect(gray, 0)
for subject in subjects:
```

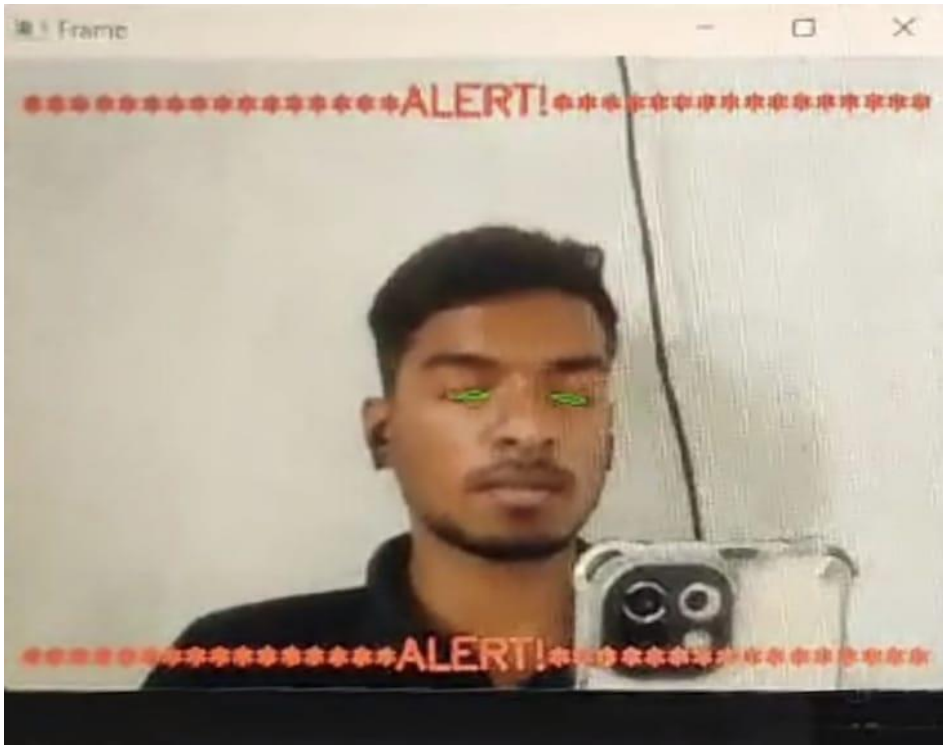
```
shape = predict(gray, subject)
shape = face_utils.shape_to_np(shape)
leftEye = shape[lStart:lEnd]
rightEye = shape[rStart:rEnd]
leftEAR = eye_aspect_ratio(leftEye)
rightEAR = eye_aspect_ratio(rightEye)
ear = (leftEAR + rightEAR) / 2.0
leftEyeHull = cv2.convexHull(leftEye)
rightEyeHull = cv2.convexHull(rightEye)
cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
if ear < thresh:
flag += 1
print (flag)
if flag >= frame_check:
cv2.putText(frame, "****************ALERT!****************", (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
cv2.putText(frame, "****************ALERT!****************", (10,325),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
mixer.music.play()
else:
flag = 0
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
if key == ord("q"):
break
cv2.destroyAllWindows()
cap.release()
```

8.Output:

Before detection:



Detecting alert message:


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

9.Conclusion:

The real-time drowsiness detection system using eye aspect ratio presents a promising solution to address the issue of driver fatigue and contribute to improved road safety. Its continuous monitoring, accurate EAR calculation, and effective alert mechanism make it a valuable tool for preventing drowsy driving accidents.