EYE DETECTION PROJECT USING NLP ¶

In [ ]:

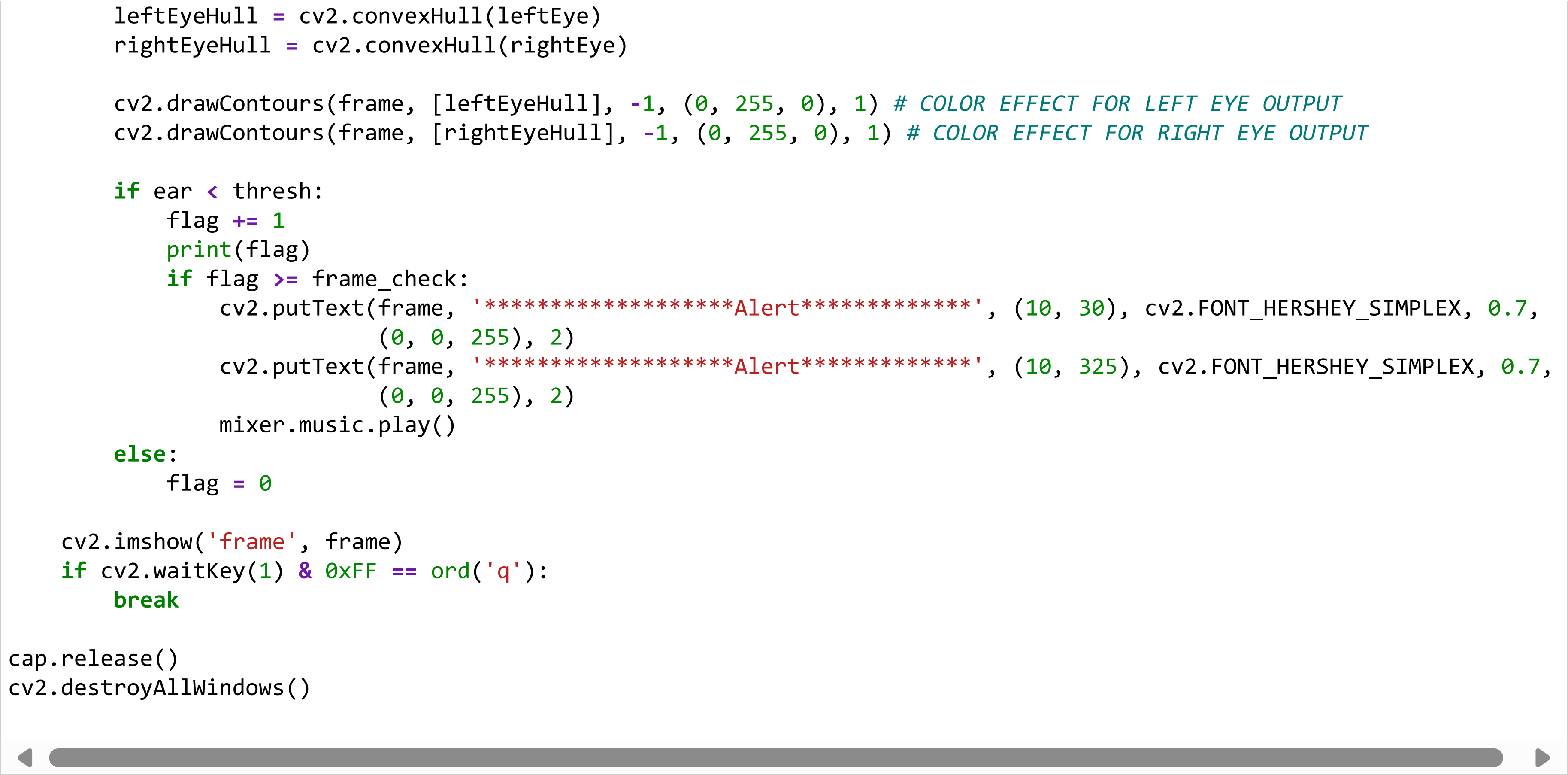
```
import cv2
import imutils
from imutils import face_utils
import dlib
from scipy.spatial import distance
from pygame import mixer
from colorama import init, Fore

# Initialize colorama
init(autoreset=True)
```

In [ ]:

```
mixer.init()
mixer.music.load('music.wav')
```

In [ ]:

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

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 # EAR = EAR ASPECT RATIO

thresh = 0.25
flag = 0
frame_check = 20
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS['left_eye'] #LEFT SIDE EYE CHECK
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS['right_eye'] #RIGHT SIDE EYW CHECK

detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

cap = cv2.VideoCapture(0) #(0) IS FOR SINGLE DEFAULT CAMERA WHERE AS WE CAN PUT 1,2,3, & SO ON TO CONNECT MULTIPLE CAM

mixer.init()
mixer.music.load('music.wav')  # Provide the correct path to your alert sound

while True:
    ret, frame = cap.read()
    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 # FORMULA FOR EAR
```



```
        leftEyeHull = cv2.convexHull(leftEye)
        rightEyeHull = cv2.convexHull(rightEye)

        cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1) # COLOR EFFECT FOR LEFT EYE OUTPUT
        cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1) # COLOR EFFECT FOR RIGHT EYE OUTPUT

        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)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Title: Real-time Eye Movement Detection System for Safe Driving and Productive Work Environments

****

Introduction:

In today's fast-paced world, ensuring safety during activities like driving and promoting focus during classes or office work is crucial. This project aims to develop a real-time eye movement detection system to enhance safety and productivity in these scenarios.

****

Objectives:

1. To implement a robust eye movement detection algorithm.
2. To integrate the system into real-world applications such as driving, attending classes, and office work.
3. To enhance safety by detecting signs of driver drowsiness and inattentiveness.
4. To improve focus and productivity by providing feedback on attentiveness during classes or office tasks

****

Key Features:

1. Real-time Eye Tracking:

- Utilize advanced computer vision techniques to track eye movements in real-time.
- Implement algorithms for accurate detection of gaze direction and blink frequency.

2. Application Integration:

- Develop modules tailored for specific scenarios: driving, class attendance, and office work.
- Ensure seamless integration with existing systems or as a standalone application

3. Class/Office Attentiveness Monitoring:

- Create mechanisms to assess attentiveness during classes or office tasks.
- Provide feedback or notifications to users to maintain focus and productivity

****

Technology Stack:

1. Computer Vision:

- OpenCV for real-time image processing and feature extraction.
- Machine learning models for eye movement prediction.

2. User Interface:

- Design a user-friendly interface for visualizing eye movement data and receiving alerts.

****

Expected Outcome

- A reliable and accurate eye movement detection system.
- Integration with driving systems, classroom settings, and office environments.
- Improved safety on the road through driver drowsiness detection.
- Enhanced productivity by promoting attentiveness during classes and work.

****

Conclusion

- This project seeks to address safety concerns and boost productivity in various scenarios by implementing a sophisticated real-time eye movement detection system. The integration of this system into educational, and office environments holds the potential to make a significant impact on user safety and focus.

****