VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CSE3999
Technical Answers for Real World Problems

Review III

Attention Span Detection in Online Flip Classroom Sessions

Prof. Vergin Raja Sarobin M

Slot: TDD1

Demo: tinyurl.com/ezlearn-demo

Team Members

Rushvi Jitendar Shah (17BCE1067)

Akash T (17BCE1093)

Ramprasath A (17BCE1098)

Jomith Yarlagadda (17BCE1116)

G Rajagopalan (17BCE1202)

Problem Definition

Most of the people in the world are having easy access to the internet and want to skill up during their free time. Hence the easiest solution for this is to just enroll in an online MOOC course and learn. However, it's the natural human tendency to get distracted to something else while learning and this might result in not being able to understand a concept in the course. Our project aims at minimizing distractions for the student by giving hierarchy of alerts to pay attention to the course content which in turn will enable the student to learn better.

Motivation

We came across the problem of people not paying attention in online classes leading to poor learning and we noticed an unexpected surge in the number of users learning online. Due to the current global pandemic everyone is forced to learn online and it also leads to situations where most people are not paying attention and distracted with something else in the background. Apart from that as per the report form KPMG before the pandemic, it is estimated that online education market in India is going to be around 2 Billion Dollars. In that case, in order to improve the quality of online learning we have come up with the above problem definition.

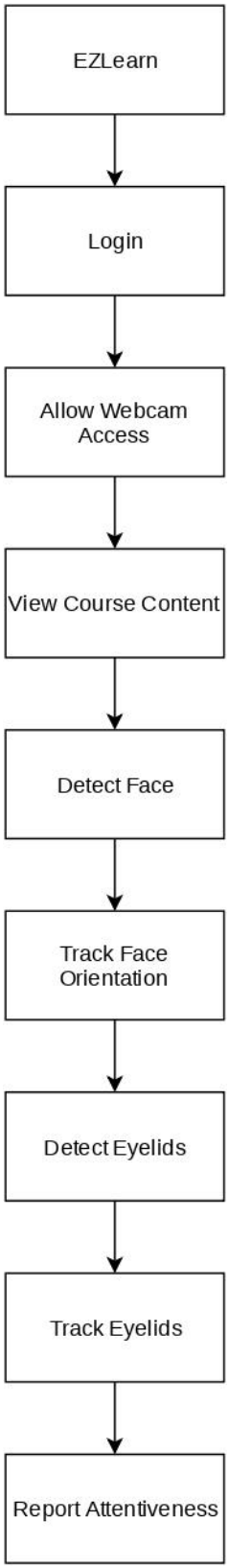
Methodology

Attention span is a normal function of brain. To calculate the time spent on watching videos attentively, various algorithms have been used. Python, openCV, Dlib, Node.js, p5.js, ml5.js are mainly used to achieve the task.

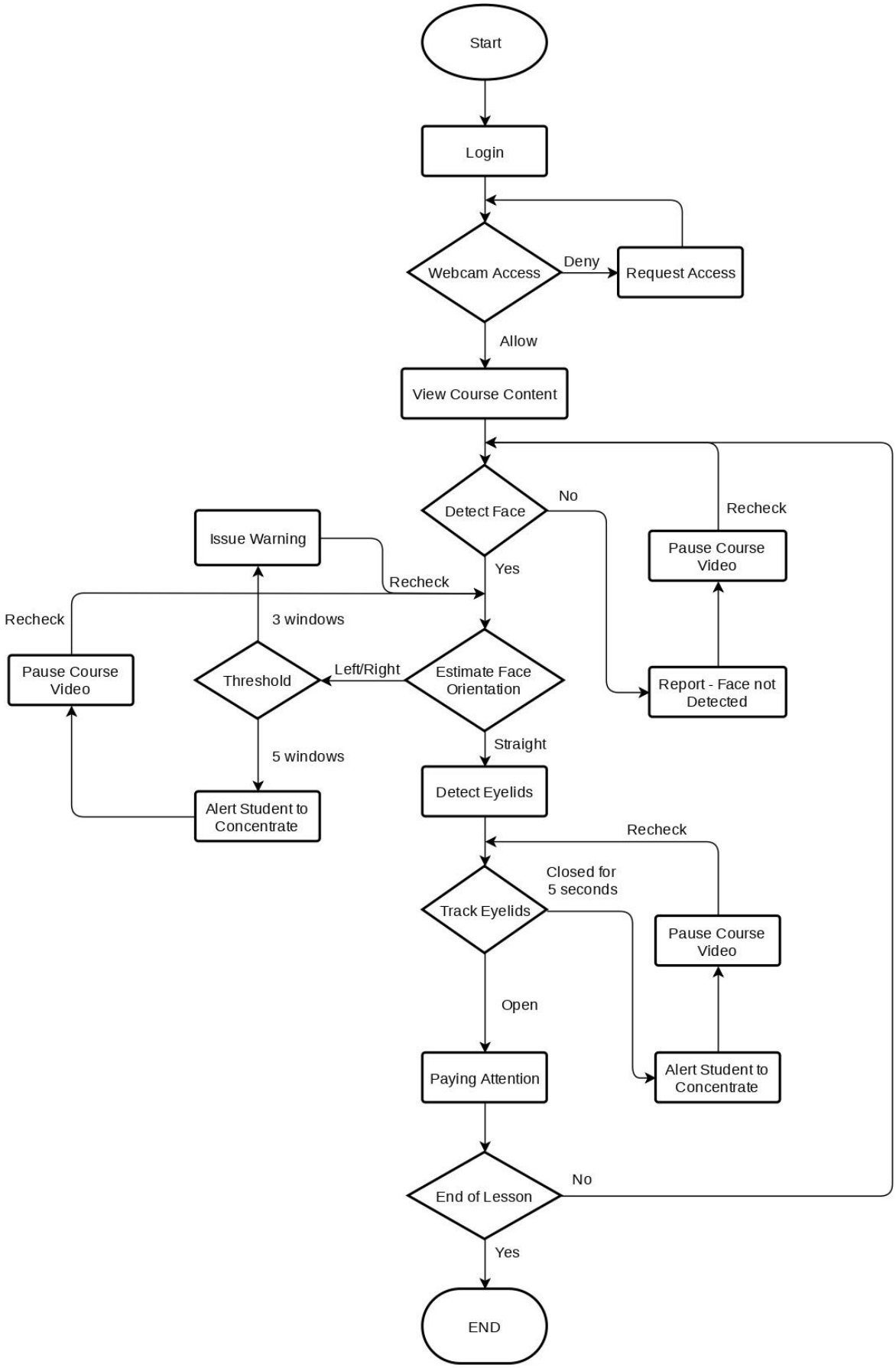
The student's webcam is used to detect his face and detect his attention span while viewing the course content. If no face is detected in the webcam then the video is paused. Once the student's face is detected, then the orientation of the face is estimated. While viewing the course content if the student turns left or right for up-to a threshold limit, at first a warning will be issued. If the student still doesn't respond to the waring and continue to pay attention to the course video, then an alert will be prompted asking the student to pay attention and the course video will be automatically paused as the student was not paying attention. The threshold limit will be explained in detail in the module description.

Next, after estimating the student's face orientation, the eyelids of the student will be detected and tracked. If the student closes his eyelids for 5 or more seconds, then an alert will be prompted asking the student to pay attention and the video will be automatically paused as the student was not paying attention. Moreover, the eyelid tracking also enables us to ensure that the student is viewing the course video and not a picture placed in front of the camera.

Block Diagram



Flowchart

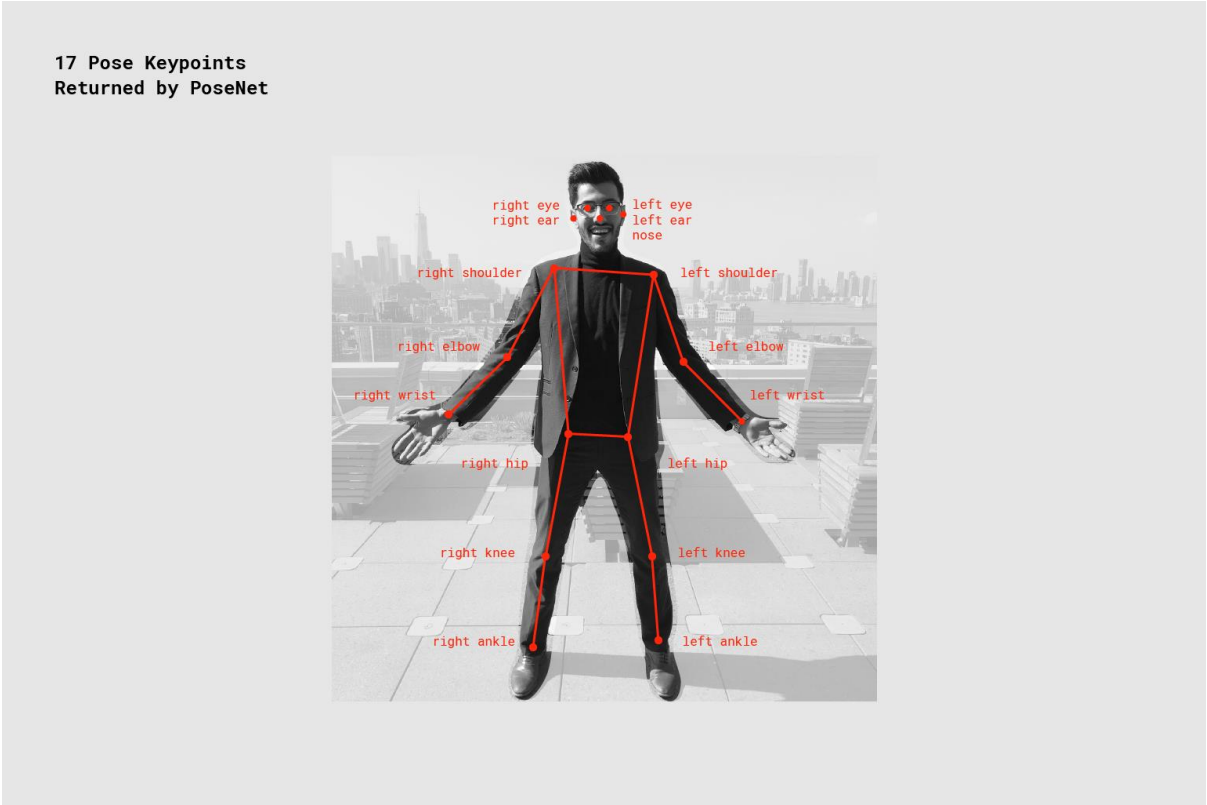

Start
Login
Webcam Access
Deny
Request Access
Allow
View Course Content
Detect Face
No
Recheck
Pause Course Video
Issue Warning
Yes
Recheck
Recheck
3 windows
Pause Course Video
Threshold
Left/Right
Estimate Face Orientation
Report - Face not Detected
Straight
5 windows
Detect Eyelids
Alert Student to Concentrate
Recheck
Closed for 5 seconds
Track Eyelids
Pause Course Video
Open
Paying Attention
Alert Student to Concentrate
No
End of Lesson
Yes
END

Module Description

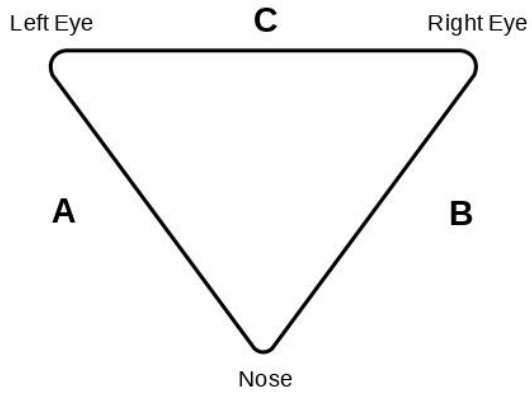
1. Face Detection and Face Orientation Estimation

The student's webcam is used to detect his face and detect his attention span while viewing the course content. If no face is detected in the webcam then the video is paused. Once the student's face is detected, then the orientation of the face is estimated. For this module we use javascript and two other javascript libraries namely p5.js and ml5.js.

The orientation of the face is estimated with the help of the key points provided by PoseNet model. PoseNet is a vision model that can be used to estimate the pose of a person in an image or video by estimating where key body joints are. PoseNet returns 17 key-points for all persons who are present in the given image, out of which we are going to work with the three key-points present in the face namely: left eye, right eye and the nose.



Now, using the 3 key points i.e. left eye, right eye and nose, we can detect whether the student is looking left or right, assume that the 3 key points are the three vertices of triangle, the distance between left eye and right eye is denoted as C, the distance between right eye and nose is denoted as B and the distance between nose and left eye as A, to find the angle formed by line A and line C.



Now to find the angle formed by line A and line C,

$$\angle a = acos((B^2 + C^2 - A^2)/(2BC))$$

And to find the angle formed by line B and line C,

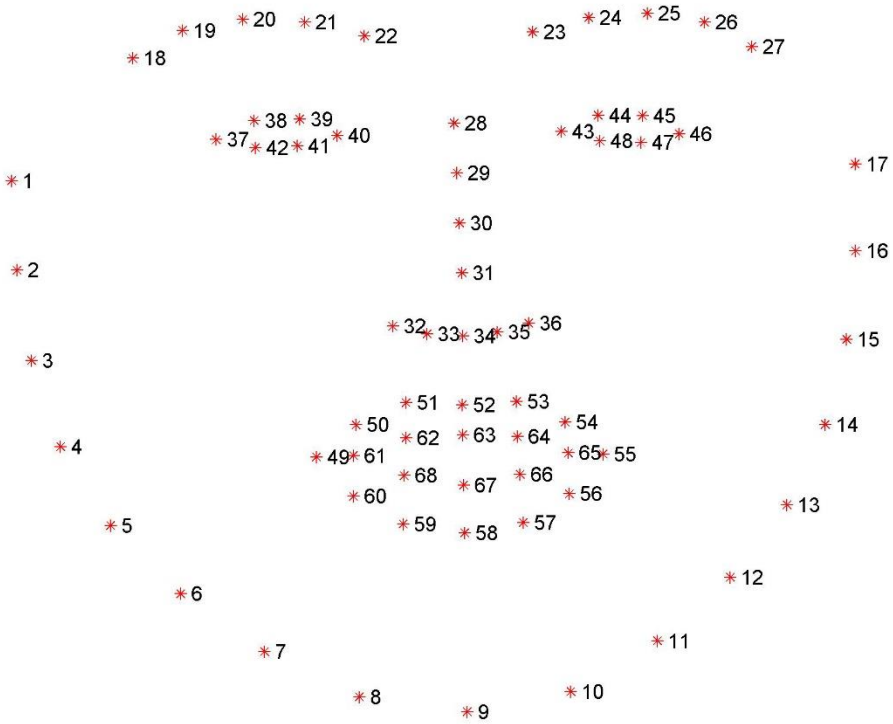
$$\angle b = acos((C^2 + A^2 - B^2)/(2CA))$$

If the student turns his/her face to left with respect to the camera then angle 'a' is greater than angle 'b' and when he/she turns right with respect to the camera then angle 'b' is greater than angle 'a'.

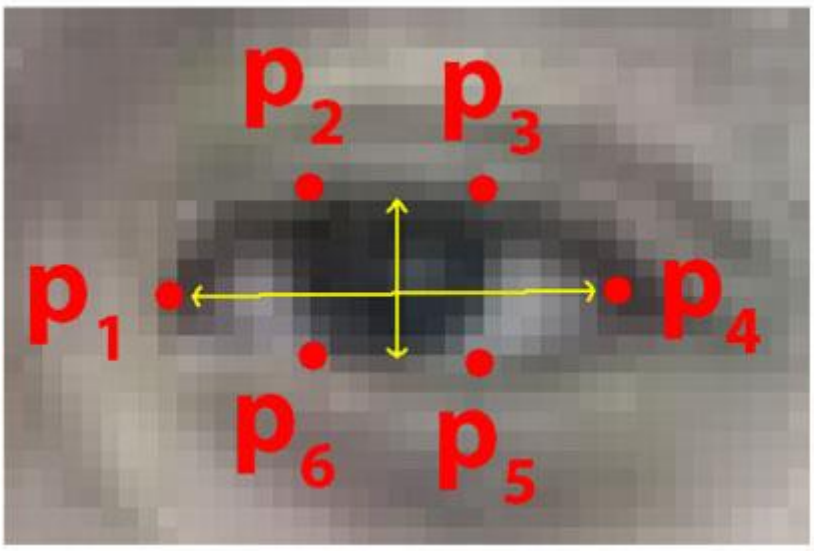
Now regarding the threshold limit for warning and alert, for a warning to be issued, the student must have turned left/right for at least 3 consecutive windows and for an alert to be prompted, the student must have turned left/right for at least 5 consecutive windows. Now each window is calculated by taking the mode of the face orientation in the last 300 windows. Although we know that the video feed from webcam is at 30 fps on an average, it is still not sufficient to estimate the duration from the fps and the range of windows.

2. Eyelid Detection and Eyelid Tracking

We mainly use Python, openCV and Dlib to detect whether the student is looking at the camera with open eyes. To do this we use 12 points out of 68 points in a pre-trained model called facial training set. The library outputs a 68-point plot on a given input image. In order to detect eye blinks, we need to pay attention to points 37-46, the points that describe the eyes. The Eye Aspect Ratio is an estimate of the eye-opening state.



Dlib Facial Landmark Plot

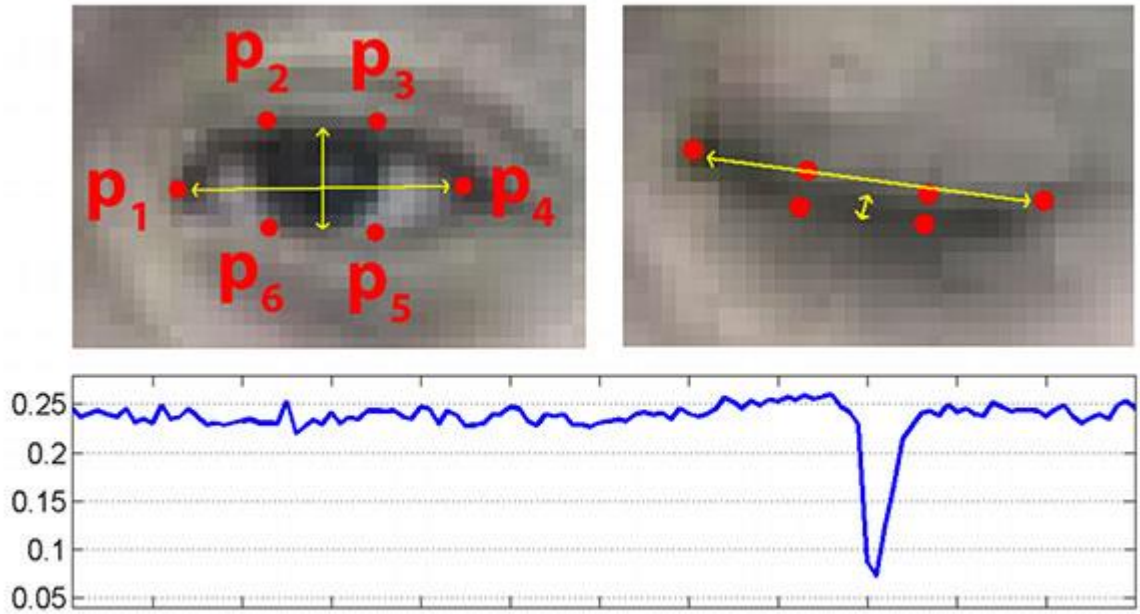


The eye facial landmarks required for calculating eye aspect ratio

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

Equation for calculating eye aspect ratio

"The Eye Aspect Ratio is a constant value when the eye is open, but rapidly falls to 0 when the eye is closed."

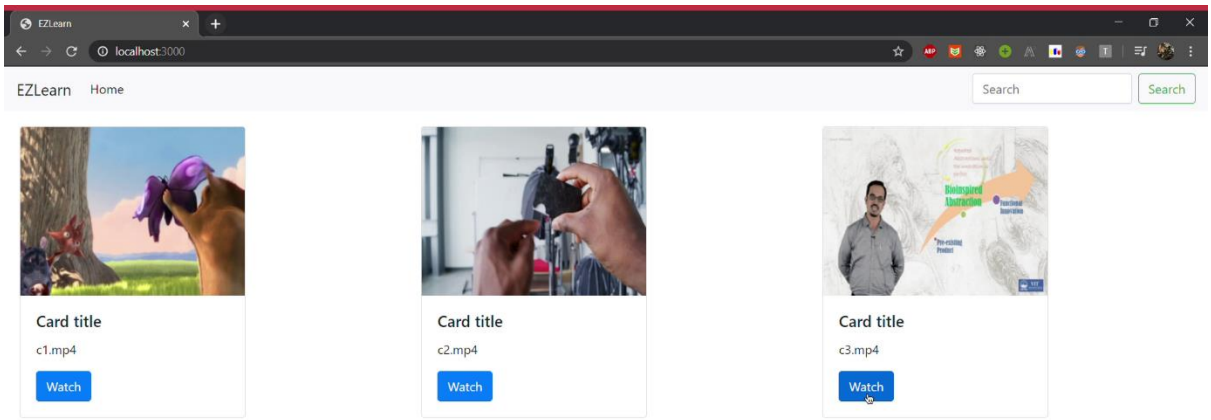


Eye aspect ratio vs Time

Our program aims at determining whether a student's eyes are closed if the Eye Aspect Ratio falls below a certain threshold. The webcam captures student's face at 30 fps, so there are total 150 frames in 5 secs, if the student is not watching continuously for 5 seconds then the video is paused, if the user opens the eye within 5 seconds then the alert is averted. To make sure that the user is watching the video and not a picture placed in front of the camera, we check whether the student blinks his eyes.

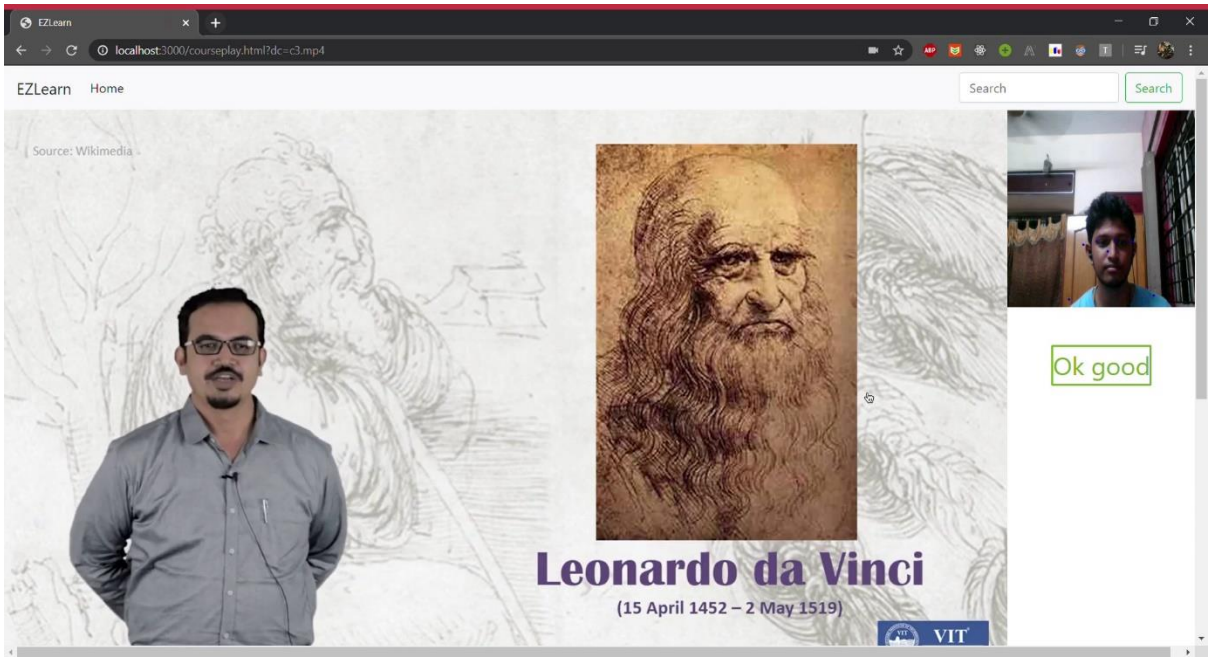
Output Screenshots

1. EZLearn Homepage



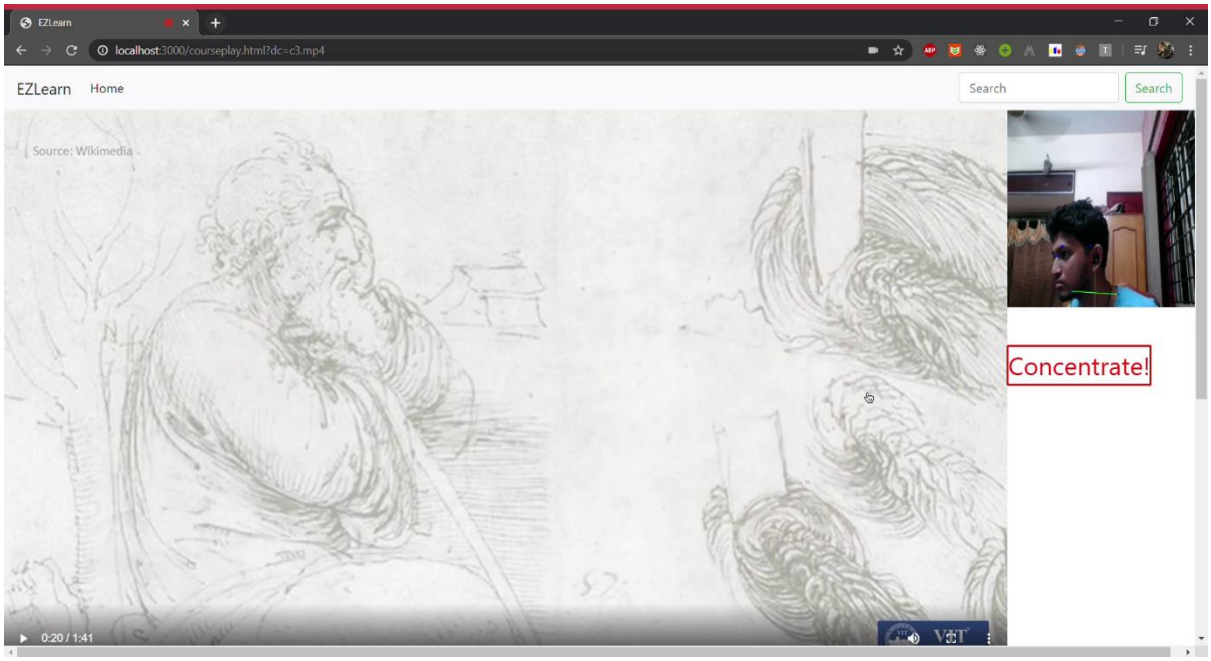
2. Playing course content

3. When student pays attention

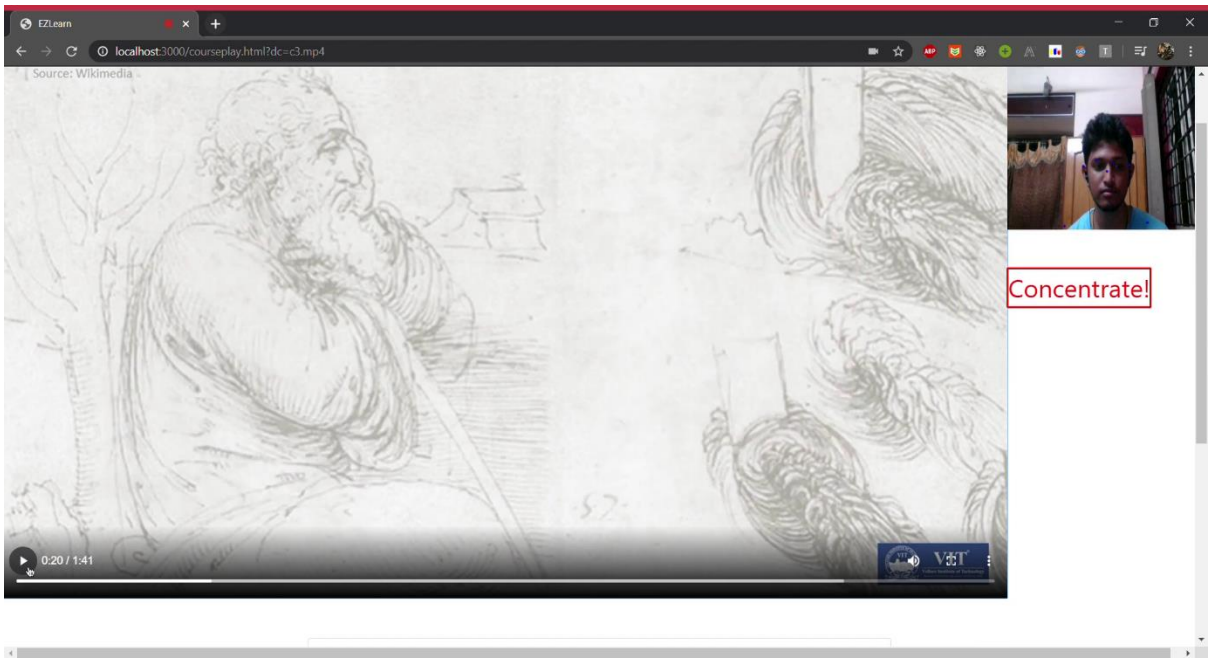


4. Warning Threshold Reached

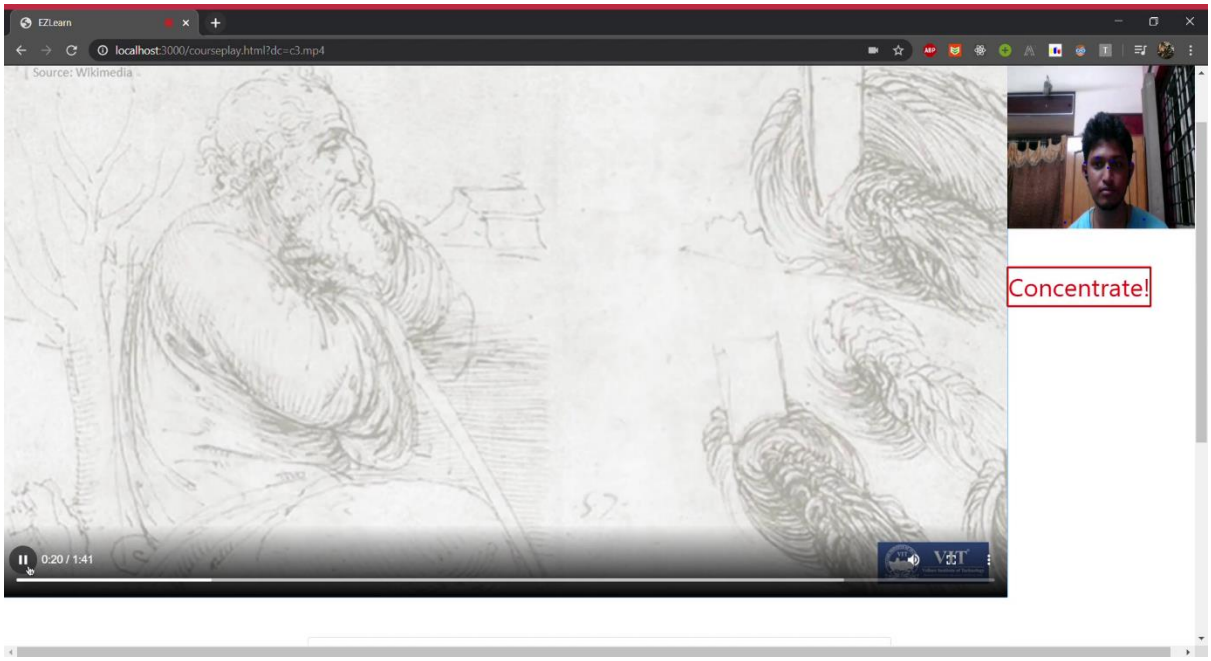
5. Alert asking to ‘Concentrate’ and video has been paused.



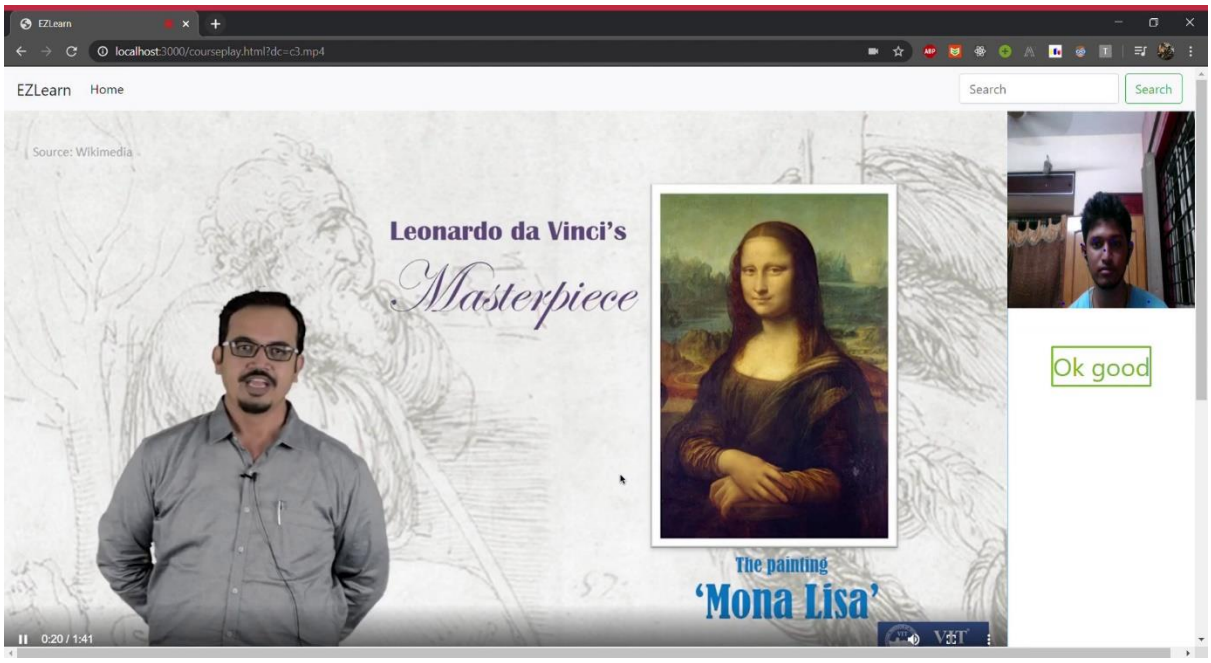
6. Resuming video



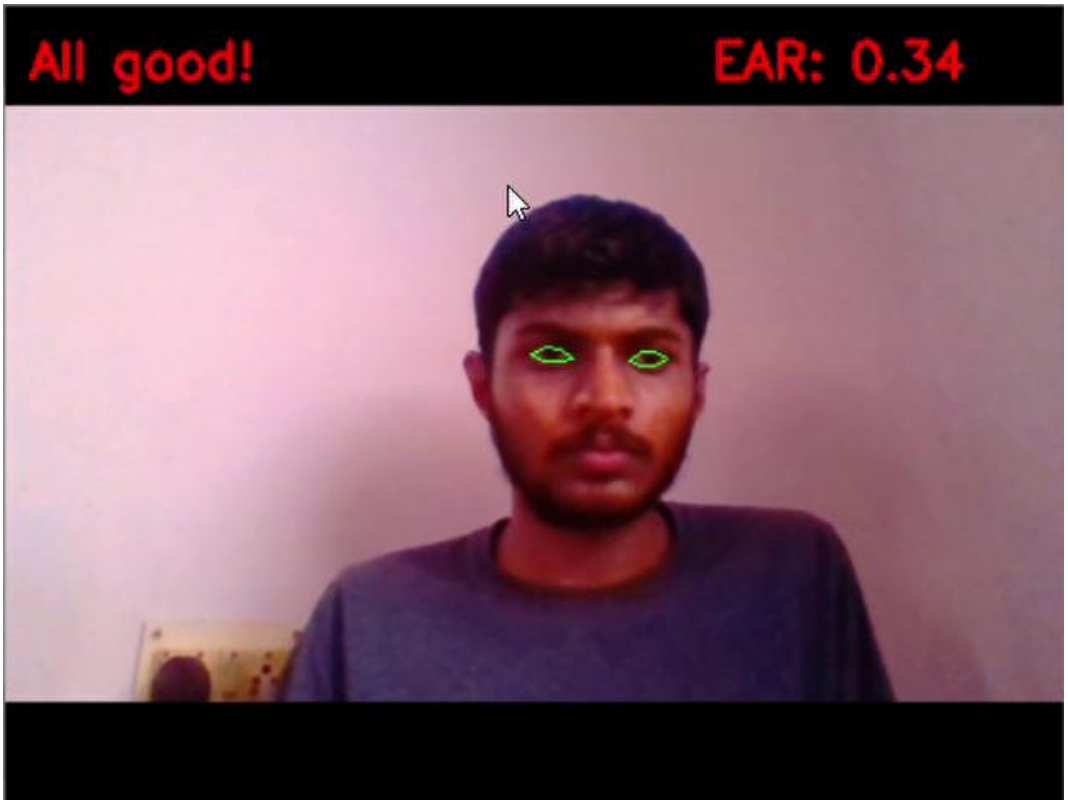
7. Video has been resumed



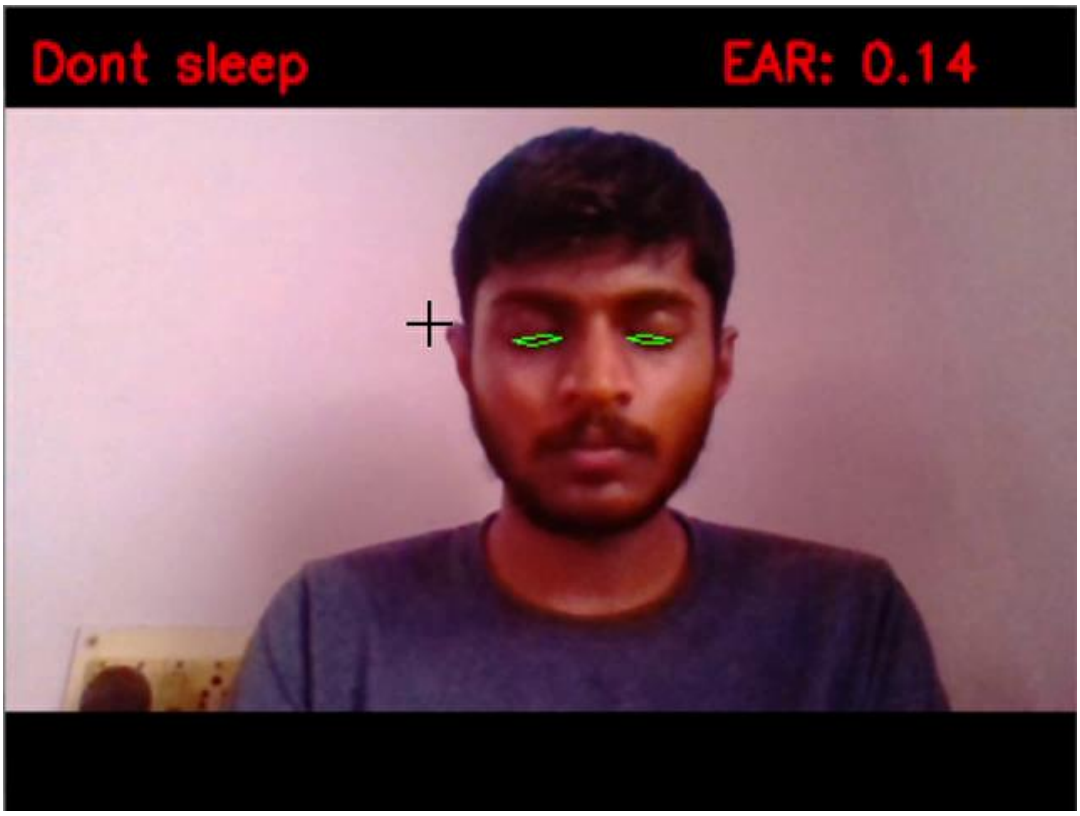
8. Status returns to normal



9. When eyelids are open



10. When eyelids are closed



Sample Coding

Face Detection & Face Orientation Estimation

```javascript
let webcam_output;
let poseNet;
let poses = [];
let wnd = [];
let sz = 300;

let warn = []
class Slicer{
  constructor()
  {
    this.data = []
  }
  add(x)
  {
    if (this.data.length >= sz) {
      for (var i = 0; i < 100;++i)
      this.data.shift();
    }
    this.data.push(x);
  }
  obgetMode()
  {
    return Slicer.getMode(this.data);
  }
```

```javascript
  static getMode(array) {
    let map = array.map((a) => array.filter((b) => a === b).length);
    let mx = array[map.indexOf(Math.max.apply(null, map))];
    return mx;
  }

}

var cnv;
let memory = null;
function setup() {

  memory = new Slicer();
  cnv=createCanvas(250, 250);
  webcam_output = createCapture(VIDEO);
  console.log(width, height);
  webcam_output.size(width, height);

  cnv.position(1080+200, 57);
  poseNet = ml5.poseNet(webcam_output, modelReady);

  poseNet.on('pose', function(results) {
    poses = results;
  });
```

File Edit Selection View Go Run Terminal Help poseNet_webcam.js - Visual Studio Code

poseNet_webcam.js

C: > Projects > EZLearn-master > public > poseNet_webcam.js > Slicer

```javascript
  webcam_output.hide();
}

function modelReady() {
  singlePose();
}

function draw() {
  image(webcam_output, 0, 0, width, height);
  drawKeypoints();
  drawSkeleton();

}

function drawKeypoints() {
  for (let i = 0; i < poses.length; i++) {
    let pose = poses[i].pose;
    checkValidity(pose);
    for (let j = 0; j < pose.keypoints.length; j++) {
      let keypoint = pose.keypoints[j];
      if (keypoint.score > 0.2) {
        fill(0, 0, 255);
        noStroke();
        ellipse(keypoint.position.x, keypoint.position.y, 3, 3);
```

0 0 Ln 25, Col 3 Spaces: 2 UTF-8 LF JavaScript

File Edit Selection View Go Run Terminal Help poseNet_webcam.js - Visual Studio Code

poseNet_webcam.js

C: > Projects > EZLearn-master > public > poseNet_webcam.js > Slicer

```javascript
      }
    }
  }
}

function checkValidity(pose)
{
  pos = detect(pose);
  memory.add(pos[0]);
  if(pos[1]=='s')
    memory.add(pos[1]);

  wnd.push(memory.obgetMode());
  let res = null;
  if (wnd.length >= sz)
  {
    res = Slicer.getMode(wnd);
    console.log(res,warn);
    if (warn.length == 0)
      warn.push(res);
    else if (warn[warn.length - 1] == res)
      warn.push(res);
    else warn = [];
    for (var i = 0; i < 100;++i)
```

0 0 Ln 25, Col 3 Spaces: 2 UTF-8 LF JavaScript

File Edit Selection View Go Run Terminal Help poseNet_webcam.js - Visual Studio Code

poseNet_webcam.js

C: > Projects > EZLearn-master > public > poseNet_webcam.js > Slicer

```javascript
      wnd.shift();
    }
    if (warn.length >= 5 && warn[0] == 's')
      warn = [];
    if (warn.length == 3 && warn[0] != 's') {
      console.log("WARN");
      var el = document.getElementById('status');
      el.style.borderColor = "yellow";
      el.style.color = "yellow";
      select('#status').html('Warning');
    }
    if (warn.length >= 5 && warn[0] != 'l') {
      select('#status').html('Concentrate!');
      var el = document.getElementById('status');
      el.style.borderColor = "red";
      el.style.color = "red";
      select('#videoPlayer').pause();
    }
    //wait till certain number of poses generate say 20000
  }

  const detect = (pose) => {
    var dc = []
    var points = []
```

0 0 Ln 25, Col 3 Spaces: 2 UTF-8 LF JavaScript

File Edit Selection View Go Run Terminal Help poseNet_webcam.js - Visual Studio Code

poseNet_webcam.js

C: > Projects > EZLearn-master > public > poseNet_webcam.js > detect

```javascript
    var points = []
    var parts = []
    for (var i = 0; i < 5; i++) {
      var data = null
      const pk = pose.keypoints[i]
      if (pk.score > 0.7) {
        data = [pk.part, pk.score, pk.position.x, pk.position.y]
        parts.push(pk.part)
      }
      points.push(data)
    }

    try {
      var a = Math.sqrt(Math.pow((points[0][2] - points[1][2]), 2) + Math.pow((points[0][3] - points[1][3]),
      var b = Math.sqrt(Math.pow((points[0][2] - points[2][2]), 2) + Math.pow((points[0][3] - points[2][3]),
      var c = Math.sqrt(Math.pow((points[1][2] - points[2][2]), 2) + Math.pow((points[1][3] - points[2][3]),
      var B = Math.acos((Math.pow(b, 2) + Math.pow(c, 2) - Math.pow(a, 2)) / (2 * b * c)) * 180 / Math.PI
      var A = Math.acos((Math.pow(c, 2) + Math.pow(a, 2) - Math.pow(b, 2)) / (2 * c * a)) * 180 / Math.PI
      if (A < 30 && B < 30) {
        dc.push("l");
      }
      else if (A > B) {
        if ((B + 7.5) < A) {
          //console.log("Turned left")
```

0 0 Ln 132, Col 3 Spaces: 2 UTF-8 LF JavaScript

```javascript
            dc.push("l")
          }
          else {
            //console.log("Straight")
            dc.push("s")
          }
        } else if (B > A) {
          if ((A + 7.5) < B) {
            //console.log("Turned right!")
            dc.push("r")
          }
          else {
            //console.log("Straight")
            dc.push("s")
          }
        }
        score = 0
        if (dc[0] == "s") score += 0.5
        if (dc[1] == "s") score += 0.5
      } catch (err) {
        if (dc[0] == "s") score = 0.5
        else score = 0
      }
      return [dc[0],dc[1]];
```

Eyelid Detection & Eyelid Tracking

```python
from scipy.spatial import distance as dist
from imutils.video import FileVideoStream
from imutils.video import VideoStream
from imutils import face_utils
import numpy as np
import argparse
import imutils
import time
import dlib
import cv2

def eye_aspect_ratio(eye):
  # compute the euclidean distances between the two
  # sets of vertical eye landmarks (x, y)-coordinates
  A = dist.euclidean(eye[1], eye[5])
  B = dist.euclidean(eye[2], eye[4])
  # compute the euclidean distance between the
  # horizontal eye landmark (x, y)-coordinates
  C = dist.euclidean(eye[0], eye[3])
  # compute the eye aspect ratio
  ear = (A + B) / (2.0 * C)
  # return the eye aspect ratio
  return ear
```

```python
ap = argparse.ArgumentParser()
ap.add_argument("-p", "--shape-predictor",required=True,help="path to facial landmark predictor")
ap.add_argument("-v", "--video", type=str,default="",help="path to input video file")
args = vars(ap.parse_args())

EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 150
COUNTER = 0
TOTAL = 0

print("[INFO] loading facial landmark predictor...")
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

print("[INFO] starting video stream thread...")
# vs = FileVideoStream(args["video"]).start()
# fileStream = True
vs = VideoStream(src=0).start()
# vs = VideoStream(usePiCamera=True).start()
fileStream = False
time.sleep(1.0)
```

```python
fileStream = False
time.sleep(1.0)

while True:
  # if this is a file video stream, then we need to check if
  # there any more frames left in the buffer to process
  if fileStream and not vs.more():
    break
  # grab the frame from the threaded video file stream, resize
  # it, and convert it to grayscale
  # channels)
  frame = vs.read()
  frame = imutils.resize(frame, width=450)
  gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
  # detect faces in the grayscale frame
  rects = detector(gray, 0)
  for rect in rects:
    # determine the facial landmarks for the face region, then
    # convert the facial landmark (x, y)-coordinates to a NumPy
    # array
    shape = predictor(gray, rect)
    shape = face_utils.shape_to_np(shape)
    # extract the left and right eye coordinates, then use the
```

```python
    shape = face_utils.shape_to_np(shape)
    # extract the left and right eye coordinates, then use the
    # coordinates to compute the eye aspect ratio for both eyes
    leftEye = shape[lStart:lEnd]
    rightEye = shape[rStart:rEnd]
    leftEAR = eye_aspect_ratio(leftEye)
    rightEAR = eye_aspect_ratio(rightEye)
    # average the eye aspect ratio together for both eyes
    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 < EYE_AR_THRESH:
      COUNTER += 1
    # otherwise, the eye aspect ratio is not below the blink
    # threshold
    else:
      # if the eyes were closed for a sufficient number of
      # then increment the total number of blinks
      if COUNTER >= EYE_AR_CONSEC_FRAMES:
        TOTAL += 1
```

```python
      if COUNTER >= EYE_AR_CONSEC_FRAMES:
        TOTAL += 1
      # reset the eye frame counter
      COUNTER = 0

    cv2.putText(frame, "Blinks: {}".format(TOTAL), (10, 30),
      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

  # show the frame
  cv2.imshow("Frame", frame)
  key = cv2.waitKey(1) & 0xFF

  # if the `q` key was pressed, break from the loop
  if key == ord("q"):
    break
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```