Eye Blinking Detection System

Computer Graphics
Group 18

Introduction

What is Eye Blinking Detection?

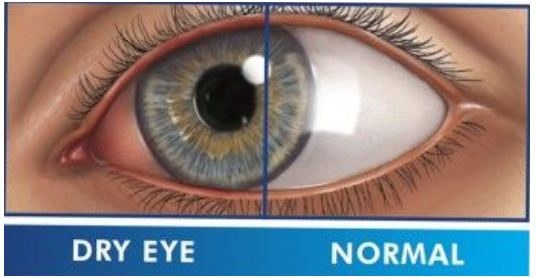
A real-time algorithm to detect eye blinks in a video sequence from a standard camera

Applications

Systems that monitor a human operator vigilance.

e.g. driver drowsiness

Systems that warn a computer user staring at the screen without blinking for a long time to prevent the dry eye and the computer vision syndromes

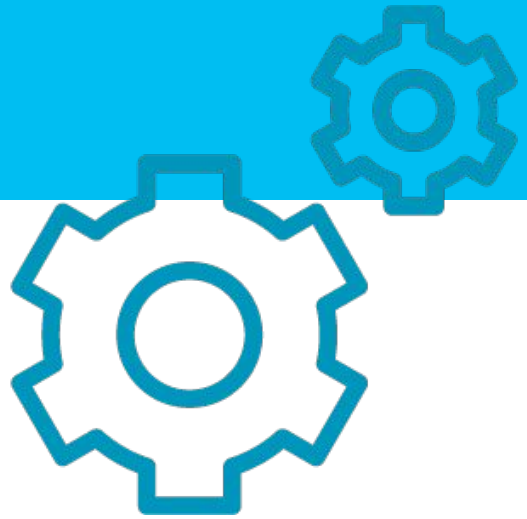


Applications

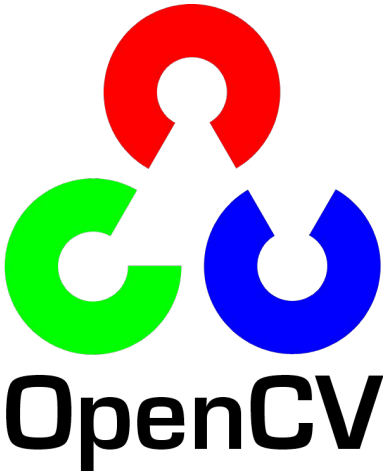
Anti-spoofing protection in face recognition systems

Human computer interfaces that ease communication for disabled people

Methodology



Environment



Recognition of the Eyes

Dlib's facial landmark detectors incorporates the facility to extract only the landmarks relating to the eyes.

```
# initialize dlib's facial landmark detector
print("[INFO] loading facial landmark predictor...")
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])

# extract the landmarks related to both eyes
(lBegin, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rBegin, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
```

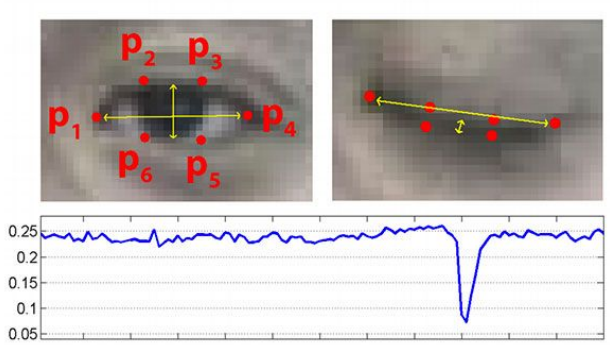
Preliminary Steps

Before detecting the facial landmarks each frame is resized and converted into grayscale.

```
# grab the frame, resize
# it, and convert it to grayscale
frame = vs.read()
frame = imutils.resize(frame, width=450)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Basis to the Algorithm

Eye Aspect Ratio (EAR) is used to detect blinks



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

Algorithm

A blink is identified using the EAR values calculated from the series of frames in the video stream according to the following two variables;

- EAR threshold value
- Number of consecutive frames between a blink

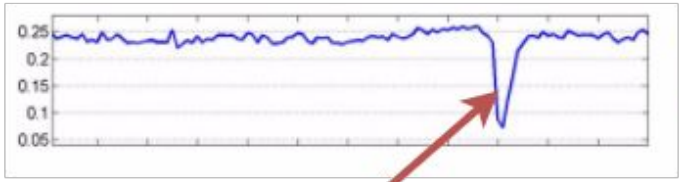
The above process is done for each face in each frame.

How it Works

Consecutive Frames

Resized Grey Scale Frames

Calculate EAR



Blink Detected

```
# loop over the face detections
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
        # coordinates to compute the eye aspect ratio for both eyes
        leftEye = shape[lBegin:lEnd]
        rightEye = shape[rBegin: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
        # compute the convex hull for the left and right eye, then
        # visualize each of the 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)

        # check to see if the eye aspect ratio is below the blink
        # threshold, and if so, increment the blink frame counter
        if ear < EAR_THRESHOLD:
                COUNTER += 1
```

Previously Used Method

Before AER traditional approach was used to identify blinks

1. Eye localization.
2. Thresholding to find the whites of the eyes.
3. Determining if the “white” region of the eyes disappears for a period of time (indicating a blink).

Why EAR?

- Unlike traditional methods AER is an easily calculable simple approach.
- OpenCV and dlib makes our lives more easy when applying EAR approach.

References

- Blog post on Eye blink detection using OpenCV and dlib

https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/

- Research paper on real time eye blink detection using facial landmarks:

http://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf?spm=a2c4e.11153940.blogcont336184.6.28a771e8bHtjbJ&file=05.pdf

Thank You!