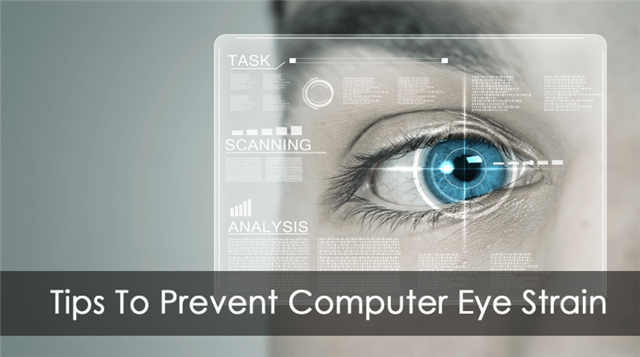
## horizontal line



Strain Analysis Based On Eye Blinking

08.11.2020

**─**

Tanya Garg

Pursuing B.Tech (CSE)

Meerut Institute of engineering and Technology

# Overview

A neural network model is built which alerts the user if eyes are getting strained. This model uses the integrated webcam to capture the face (eyes) of the person. It captures the eye movement and counts the number of times a person blinks. If blink count deviates from the average value (if the number of blinks is less or more), then an alert is initiated by playing an audio message along with a popup message is displayed on the screen appropriately.

# Goals

1. know fundamental Computer vision, google text to speech.
2. Gain a broad understanding of face landmark detection.
3. know how to install necessary packages and set up the environment.
4. Calculate Eye aspect ratio
5. Work with google text to speech
6. Work with Tkinter

# Specifications

To create an eye blink detector, eyes will be the area on the face that we are interested in. We can divide the process of developing an eye blink detector into the following steps:

1. Detecting the face in the video
2. Detecting facial landmarks of interest (the eyes)
3. Calculating eye width and height
4. Calculating eye aspect ratio (EAR) – relation between the width and the height of the eye
5. Displaying the eye blink counter in the output video
6. Based on the blinks, an alert is initiated to the user with an audio message and popup message.

# Project Report

Tsk 572 (Importing Necessary Libraries)

All the necessary libraries have been successfully imported in the main application file “app\_eye.py” . As previously mentioned in the project prerequisites, Jupyter notebook has been used to accomplish this task and later the file has been extracted as a python file with a (.py) extension.

Tsk 573 (Defining Necessary Functions)

All the necessary functions have been successfully defined in the file named “app\_eye.py”. The functions are described below,

**a. playaudio(text)**

In this function, we are translating the text input to a speech by using gTTS and saving the translated speech to the output1.mp3 file. We are returning the output1.mp3 to the calling function.

**b. popupmsg(msg)**

* Creating an instance of Tk initializes the interpreter and creates the root window
* Giving a title and style to the popup window and configuring it using the configure function.
* A label is used to display text messages Generally the content is static, but your program can change the text.
* pack() is geometry manager organizes widgets in blocks before placing them in the parent widget.
* We are creating an “Okay” button and showing the window using mainloop() function .

**c. eye\_aspect\_ratio(eye) :**

We can Calculate Eye Aspect Ration this using the below code

* Compute the Euclidean distances between the two sets of vertical eye landmarks (x, y)-coordinate.
* Compute the Euclidean distance between the horizontal eye landmark (x, y)-coordinates.
* Compute the eye aspect ratio using the above formula and then return ear to the calling function

Tsk 574 (construct argparser)

Our app\_eye.py  script requires a single command-line argument, followed by a second optional one.

We are using argparse library to parse command-line arguments.ArgumentParser() is a predefined class. let us create an object ‘ap’ for it so that we can access it in our program.

Tsk 575 (Defining important constants)

Let us define two important constants

* **EYE\_AR\_THRESH**
  + By using ear value, we determine if a blink is taking place in the video stream.
  + A "blink" is registered when ear value falls below a certain EYE\_AR\_THRESH and then rises above the **EYE\_AR\_THRESH.**
  + We default it to a value of 0.3  as this is what has worked best for my applications, but you may need to tune it for your application.
* **EYE\_AR\_CONSEC\_FRAMES**

This value is set to 3  to indicate that three successive frames with an eye aspect ratio less than EYE\_AR\_THRESH  must happen for a blink to be registered.

### Tsk 576 (Get The Face Land Marks Using Dlib)

We first load the detector using the g**et\_frontal\_face\_detector(**) and facial landmark predictor dlib.shape\_predictor from dlib library.

By using face\_utils.FACIAL\_LANDMARKS\_IDXS  we can determine the starting and ending array slice index values for extracting (x, y)-coordinates for both the left and right eye .

### Tsk 577 (Capturing the input frames)

There are two ways we can capture the input video:

1.using in-built webcam

2. using video file residing on the disk

we use the VideoStream module to capture a live video.

we use the VideoCapture module in OpenCV to capture a video that is residing on the disk.

### Tsk 578 (Converting frames to grayscale channels)

The frame we have captured is a 3-channel RGB colored image. We detect face and eyes in the frame. dlib’s face detection works perfectly fine on grayscale images as well as colored images. As a grayscale image is a single channel image, we convert the frame to grayscale to reduce the processing time required by further steps of the algorithm.

### Tsk 579 (Calculate Eye Aspect Ratio)

Now let us loop over each of the faces in the frame and then apply facial landmark detection to each of them:

We’ll be computing ear value for both the left and right eye, respectively by using the predefined function eye\_aspect\_ratio(eye).

**Computing convex hull for eyes**

Compute the convex hull for the left and right eye using cv3 inbuilt function convexHull, then visualize each of the eyes using the drawCountours.

### Tsk 579 (Detect Blinks)

The aspect ratio will be approximately constant while the eye is open, and it will quickly fall to zero when a blink occurs. We need to determine the threshold for a blinking ratio that is near to zero. We will assume that every blinking ratio below that threshold will be detected as a blink, and the blinking ratio above the threshold will not be detected.

### Tsk 579 (Alerting the users)

We’ll be calculating the average number of blinks per minute . If the total blink count is less or more than the average blink count for the stipulated time(calculated for every minute incrementally), then an alert is initiated using audio and popup messages to take rest by calling the playaudio and popup functions.

### Tsk 579 (Display the result)

The cv2.putText function displays the number of blink and ear  on the OpenCV window once a blink count is detected. The cv2.imshow() function always takes two more functions to load and close the image. These two functions are cv2.waitKey() and cv2.destroyAllWindows(). Inside the cv2.waitKey() function, we can provide any value to close the image and continue with further lines of code.

### Tsk 580(Run the application)

To access the built-in webcam execute the following command in anaconda prompt

**python app\_eye.py --shape-predictor shape\_predictor\_68\_face\_landmarks.dat**

To access video file residing on the disk execute the following command in anaconda prompt

**python app\_eye.py --shape-predictor shape\_predictor\_68\_face\_landmarks.dat –video filename.mp3**

Your output looks like:

