GazeControl Documentation:

Welcome! Refer to the below text (or in the outline):

- imports
- functions
- logic

GazeControl/main.py at main · PrathamMehta08/GazeControl (github.com)

Constants:

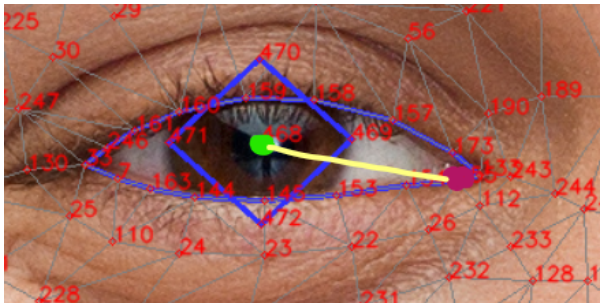
NAME	DEFINITION
minm (int)	The median of left.
maxm (int)	The median of right.
curr (string)	The selected region by the user.
left (list)	The distance between the pupil and the right part of the eye during the 5 second right calibration period.
right (list)	The distance between the pupil and the left part of the eye during the 5 second left calibration period.
calibrating_left calibrating_right asked_question yes_no said_cal said_left said_right said_done said_ask said_all	BOOLEANS: Stores TRUE/FALSE if the mentioned task is finished. Used to activate the next task if the previous one is finished.

Imports:

1. mediapipe (mp)
    - solutions
    - framework.formats
        - landmark_pb2
    - tasks
        - python
            - vision
2. numpy (np)
3. cv2
4. gaze (another file not module)
5. math
6. gtts
    - gTTS
7. statistics
8. pygame
9. os
10. time

Functions:

NAME	DEFINITION
speak()	Uses the PyGame Mixer to play the audio of the user selected option. Note: Does not play the mp3 directly because it causes the Windows Media Player to open up.
dist(x, y, x1, y1)	Finds the distance between the two points (x, y) and (x1, y1) $d = \sqrt{(x1 - x)^2 + (y1 - y)^2}$
ask_question(annotated_image)	Writes “ASK QUESTION” on the screen
calibrate_right(annotated_image)	Writes “LOOK RIGHT” on the screen
calibrate_left(annotated_image)	Writes “LOOK LEFT” on the screen

draw_yes_no(annotated_image, res)	Writes “YES” and “NO” on the screen with a green (if “YES”) and red (if “NO”) overlay
get_x(minm, maxm, d)	Range maps d Screen Range: [0, screen_width] Eye Range: [minm, maxm] ⇒[0, d - minm, maxm - minm] ⇒[0, $\frac{screen\ width}{maxm - minm} * (d - minm)$, screen_width]
draw_landmarks_on_image(rgb_image, detection_result)	Uses MediaPipe’s face_landmarks to annotate the user’s face, and gets the distance from landmark 468 and 155.  Based on the task, it either adds to left or right or decides whether the user is selecting “YES” or “NO”

Logic:

Starts recording
Saves the current frame in “temp_frame.jpg” and processes on that frame
Finite State Machine Implementation:

TIME	ACTION
2 - 5 sec	Prompts the user that calibrating is about to begin
5 - 10 sec	Prompts the user to look left for left calibration
10 - 15 sec	Prompts the user to look right for right calibration
15 - 20 sec	Mentions that calibration is finished and prompts the question to be asked

25+ sec	Prompts user with either “YES” or “NO”