**הוראות שימוש \ Instructions**

**\*We suggest using anaconda, it realy helped us with the tensorflow versions.**

Environments setup:

Python 3.7

Tensorflow 1.15.0

Required packages:

Dlib 19.22.0

OpenCV 3.4.2

Numpy 1.19.2

Moviepy 1.0.1

mkl 2020.2

pywin32 227

scipy 1.2.0

cmake 3.19.6

**configuration:**

all the changes that you need to do in order to have your own Demo are in the “proj\_config.py” file.

\*\*Only one change in the "proj\_utils.py" file: in the first function - "concat", you need to change the path to "multi\_frames" folder and in the end of the path write "\\\*.jpg".

**and the changes are:**

1. first of all, you should move the wanted zoom recording to the project file and call it “zoom\_vid.mp4”, make sure its 30 fps
2. at the top of each screen in the demo there is a name and color

those are determined in the variables “names\_arr” , “color\_arr”.

1. Each contestant has his own screen configuration.

screen size, camera position.

The screen size is in the “sw” (screen width) and “sh” (screen height) variables, the camera position is determined in the “pcy” which means the distance from the screen center to the camera in the y-axis (up is negative). also “pcx” and “pcz” for the x and z axis.

1. The variable “focal” is the focal length of the camera, to find out the focal length, the user can run the file “focal\_length\_calibration.py” and look at the focal length in the top right corner of the window.

you should run the file only on the “to\_align.mp4” video that comes out after you run our code with “what\_to\_do” = “ border to\_align”.

1. The variable “coord\_orig” holds the coordination of each image center for the original screen that the video was taken from.  
   the order is: top left, top right, bottom left, bottom right
2. The variable “scr\_orient” is responsible for the contestants order in each screen for the Demo video.  
   [0,1,2,3] means that the contestants order stays the same as it was in the original video.  
   [1,0,2,3] means that the top right and top left contestants changed place

1. Timing:  
   the program doesn’t figure out by itself the position each person is looking at each point of time. ( but it should know it to perform the task)  
   in the config file, the user has to enter the “timing” of the gaze direction.

**how to fill the timing array**:  
Each contestant has its own looking array.  
The looking array contains the top left, top right, bottom left, bottom right looking intervals.  
“intervals” is a two number list that has the starting time and finish time of each looking period.  
for example :   
for person x, he has all directions looking array, we will look at one.

top right looking array:

[ [1,6], [14,20], [31-33] ]

in this example, contestant x is looking to the person on the to right corner in time periods: 1-6, 14-20 and 31-33

**one more thing:**  
the input video that you use should be in the folder and called “zoom\_vid.mp4”

This video should be taken from zoom recording and include exactly 4 contestants.

We ran it on a video with resolution 1280x720, if you use a different resolution you should also change the coord variables in the “proj\_utils.py” file in the “get\_coordinates” function accordingly.(it is well documented)

**after configuration setup, how to run:**

in the “video\_maker.py” file there is a variable called “what\_to\_do”.

it can determine which parts of the code will run.

First of all, in the configuration setup you should have a “to\_align” video for each contestant in his own folder (tl,tr,bl,br). for that, run the code with “what\_to\_do”=”border to\_align”

if you managed to get the configuration setup correctly you can then change

“what\_to\_do” = “all”

and it will run all the code.

(be aware that it takes some time, especially if the video is long)

the output will be a video called “all\_screens.mp4” in the project folder.

**Explanation of the code parts:**

The GitHub code we found (main function = regz\_socket\_MP\_FD) gets as input a video and coordinates, and the output is the video with eye gaze correction to the coordinates we gave to the function.

The output video isn't a video yet, it's a lot of pictures inside the "multi\_frames" folder which we concatenate afterwards to have one smooth video.

"Video\_maker.py":

The code finds all possible eye gaze corrections and save them. (all the gaze corrections are basically (x,y) coordinates)

The code adds a border to look like a real ZOOM conference call, after that the code loops on all of the participants and does the following steps:

1. crops each participant window from the video and saves it.

2. for each participant, we align his gaze to all possible coordinates that we found before. All possible videos are saved under the folders of each participant (top left = tl, top right = tr, bottom left = bl, bottom right = br).

3. the code puts the correct part of video according to the look\_arr\_all. (look\_arr\_all is an array that save who looked at whom at each point of time)

4. the code pastes (overlays) the final videos of each participant in the right position according to the right screen.

5. composes all the final videos together to make the "all\_screen.mp4" video.

**The output:**

**tl (folder):**

all videos related to the top left participant in the original ZOOM meeting:

* "to\_align", the video of the person that we need to align his gaze.
* all the videos of the person aligned to all possible coordinates, example: "aligned\_[380, 190]".
* 4 videos of the person that will appear in the 4 different screens, example: "look\_timing\_2", this video go to the top right screen.
* 4 videos of the screen each one represents the final video after overlaying one participant, example: "final\_1" is the video after 2 participants update the video. "final\_3" is the real video of the screen after gaze correction of all participants.

**tr/bl/br (folders):** is the same like "tl" but for top right/bottom left/bottom right participants.

**Multi\_frames:** folder with all the pictures that represent the aligned video for specific coordinates. (output folder for the regz\_socket\_MP\_FD function)

**"all\_screens":** composed video of all the final videos for all the screen.

**References:**

<https://github.com/chihfanhsu/gaze_correction> -

*Hsu, Chih-Fan and Wang, Yu-Shuen and Lei, Chin-Laung and Chen, Kuan-Ta*

the github page of the code we used for the gaze correction.

\*All code download from this github, add our files under gaze\_correction\_system directory + replace "regz\_socket\_MP\_FD.py" (we made some changes).