Gazepoint GP3 Eye Tracker

Protocol: Determining Systematic Error in Recorded Fixation Points *and* Correcting the Recorded Data Points using Python

Alexandria Battison
alexandria.battison@uconn.edu

Contents

0.1 Determining Expected Coordinates

The specific values of the expected coordinates will vary depending on the size and resolution of the screen you are recording data from. This first section walks through how to calculate the expected coordinates for each expected fixation point on the screen. Both the expected coordinates and the reported coordinates that come from the eye tracker are expressed as a percentage of the total screen resolution. The origin (0,0) is in the top left corner, and 100 percent of the screen is the bottom right.

Briefly, determining the pixels per inch value for your particular screen, determining the physical location (in inches) on the screen the expected coordinates fall and converting that to a pixel value, and finally determining what percentage out of the total pixel resolution that value is. The percentage is converted to a decimal, and that is your expected coordinate.

0.1.1 Determining Pixels Per Inch (PPI)

In order to determine the expected coordinates, you need to first determine the pixels per inch (PPI) value for the screen(s) that you gathered data with.

1. *To determine screen resolution:* View your computer's screen resolution by going to Settings>Display>Advanced Display Settings. Screen resolution is given in the form widthXheight

2. *To find the size of your screen (in inches):* Either physically (and carefully) measure the diagonal dimension of your computer screen, or lookup online the computer you are using, and see what the reported screen size is (Anything along the lines of '13" display' or '15" monitor' is referring to the diagonal length of the screen)

3. *Calculation for finding PPI value:*

$$d_r = \sqrt{r_w^2 + r_h^2} \tag{1}$$

$$PPI = \frac{d_r}{i} \tag{2}$$

where d_r the number of pixels across the diagonal

r_w = the screen resolution width value (screen's horizontal measure)

r_h = the screen resolution height value (screen's vertical measure)

i = screen size in inches

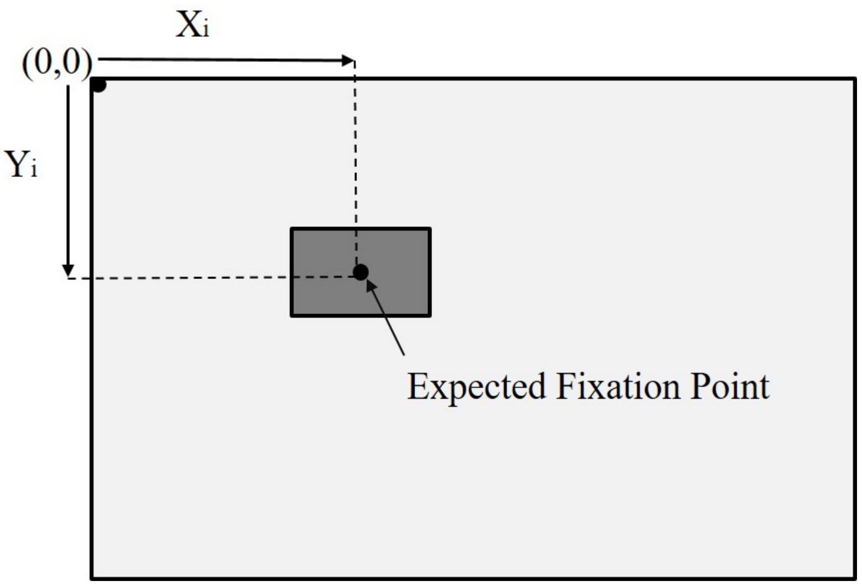
4. Or, instead of doing the above calculation, you can find an online PPI calculator. You will need to provide the screen resolution and screen size in inches.

0.1.2 Determining Expected Coordinates on the Screen

For each point on the screen that you are interested in obtaining an expected coordinate for:

1. Using a ruler, carefully measure how far the point is horizontally from the top left corner. *see diagram below* **This is your x_i value**

2. Using a ruler, do the same, this time for the vertical distance the point is downward from the top left corner (even though the value can be thought of as being 'below' the origin in the corner, still report it as a positive value) *see diagram below* **This is your y_i value**



3. *To find your expected X Coordinate:* Divide x_i by r_w

$$x_i = \frac{x_i}{r_w} \tag{3}$$

4. *To find your expected Y Coordinate:* Divide y_i by r_h

$$y_i = \frac{y_i}{r_h} \tag{4}$$

5. *Convert x_i and y_i from inches to pixels*

$$x_{pixels} = x_i * PPI \tag{5}$$
$$y_{pixels} = y_i * PPI \tag{6}$$

6. *Determine expected coordinates:* The expected coordinates are a percentage of the total screen size. The value can therefore be found by dividing the pixel value of the expected coordinate by the total pixels possible which is set by the screen resolution. **X and Y are your expected coordinates**

$$X = \frac{x_{pixels}}{x_w} \tag{7}$$
$$Y = \frac{y_{pixels}}{y_w} \tag{8}$$

Your expected coordinate is (X,Y).

0.2 Obtaining Data from Recordings

After a recording session has ended, the Gazepoint Eyetracker produces two excel files filled with data. The file labeled as "User_all_gaze.csv" contains every data point recorded, regardless of if the data represents a fixation or not, and regardless of if it is valid data or not.
The file labeled as 'User_fixations.csv' only contains one data point for each fixation. Because the sampling rate is 60Hz, any fixation point that lasts longer than 1/60th of a second will have multiple data points recorded for it in the "all_gaze.csv" file. Only the last one of these actually gives an accurate representation of the final duration and location of that fixation point. The "fixations.csv" file extracts just those final, cumulative fixation data points. *The "User_fixations.csv" file contains the desired and relevant data in the most concise form*

0.2.1 Extracting Relevant Fixation Data for RFL paradigm with Squares

1. Extract the .avi files from the GazePoint analysis program if you have not done so already

2. Watch the .avi file for the maze and participant of interest. For each square on the PowerPoint, write down on a piece of scratch paper the

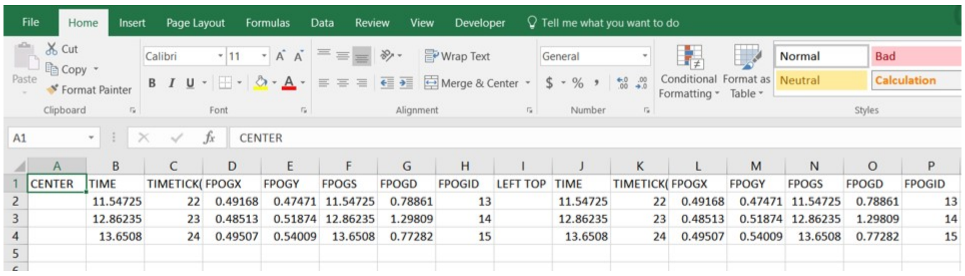
fixation points (or range of fixation points) that occur when that square is present. If this feels tedious and boring then you are doing it right.
note: When multiple squares are present on the screen at the same time, the shift in eye gaze from one square to the next should be drastic enough that you can differentiate the users intent to look at one square vs another. If not, then the eye trackers internal calibration is drastically off, so dont make assumptions that the user is looking elsewhere.
note: If the recorded eye gaze goes off the screen (particularly likely with squares in the corners) try writing down the timestamp in the video that occurs instead, or the next visible fixation point. Even if the recorded eye gaze points are off the screen, the algorithm should be able to correct them back.
Caution: If not data is recorded for a particular square, or if it is unclear when a fixation point belongs to one square or another, dont include it. Some squares may have no data recorded for them, thats ok, thats just the eye tracker.

3. Once you have identified all of the fixation points that belong to each square open the excel file the eye tracker generated that contains just the fixation points
*File to Open:*User_Fixation.csv (for data gathered with Gazepoint Eye Tracker)

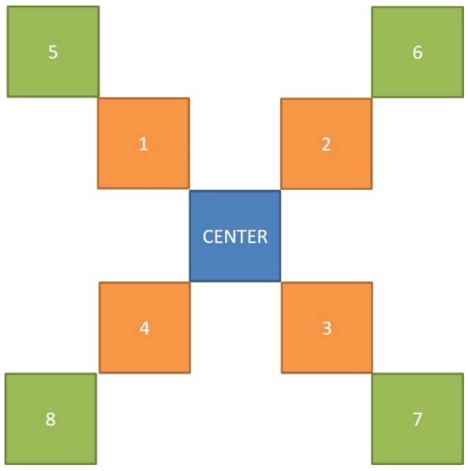
4. Copy and paste columns D through J (labeled as TIME to FPOGD) from the User_Fixations.csv file to the appropriate place in another excel file where you will gather all the desired recorded fixation points
Important Note On Formatting the Excel Document Recorded that the Fixation Points are Collected in: The python file that is used for data analysis uses numerical indexing to iterate through columns and rows when calculating correction factors and expected data points. Please copy and paste the recorded data points that you extract into an excel file that looks *exactly* like the one shown below (image below includes example data). Failure to do so will result in the program either not running at all, or outputting incorrect data.

5. The order in which the data for the recorded fixation points should be input is as follows: 4 corners center X 4 corners center Y Top Left X Top Left Y Top Right X Top Right Y Bottom left X Bottom Left Y Bottom Right X Bottom Right Y xCenter X xCenter Y 1X 1Y 2X 2Y 3X 3Y 4X 4Y 5X 5Y 6X 6Y 7X 7Y 8X 8Y
where xCenter X and xCenter Y refer to the center square when the

x-shaped fixation points are on the screen, and 1X, 1Y, 2X... refer to the required fixation squares that are in an x-shape through the center of the screen See below for numbering of the squares in the x-shape



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	CENTER	TIME	TIMETICK(	FPOGX	FPOGY	FPOGS	FPOGD	FPOGID	LEFT TOP	TIME	TIMETICK(	FPOGX	FPOGY	FPOGS	FPOGD	FPOGID
2		11.54725	22	0.49168	0.47471	11.54725	0.78861	13		11.54725	22	0.49168	0.47471	11.54725	0.78861	13
3		12.86235	23	0.48513	0.51874	12.86235	1.29809	14		12.86235	23	0.48513	0.51874	12.86235	1.29809	14
4		13.6508	24	0.49507	0.54009	13.6508	0.77282	15		13.6508	24	0.49507	0.54009	13.6508	0.77282	15
5																



0.2.2 Extracting Relevant Fixation Data for Minion Game

1. Extract the .avi files from the GazePoint analysis program if you have not done so already

2. Watch the .avi file for the maze and participant of interest. For each minion on the PowerPoint, write down on a piece of scratch paper the fixation points (or range of fixation points) that occur when that square is present. If this feels tedious and boring then you are doing it right.

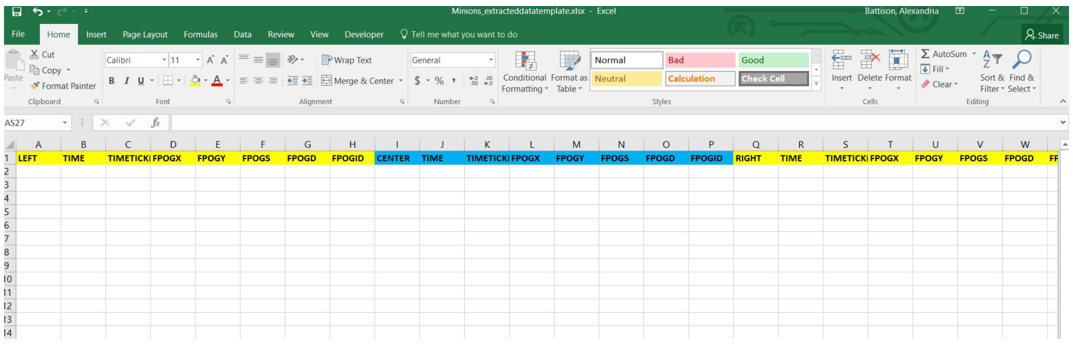
   note: If the recorded eye gaze goes off the screen try writing down the timestamp in the video that occurs instead, or the next visible fixation point. Even if the recorded eye gaze points are off the screen, the algorithm should be able to correct them back.

Caution: Some minions may have no data recorded for them, thats ok, thats just the eye tracker.

3. Once you have identified all of the fixation points that belong to each minion open the excel file the eye tracker generated that contains just the fixation points
*File to Open:*User_Fixation.csv (for data gathered with Gazepoint Eye Tracker)

4. Copy and paste columns D through J (labeled as TIME to FPOGD) from the User_Fixations.csv file to the appropriate place in another excel file where you will gather all the desired recorded fixation points
Important Note On Formatting the Excel Document Recorded that the Fixation Points are Collected in: The python file that is used for data analysis uses numerical indexing to iterate through columns and rows when calculating correction factors and expected data points. Please copy and paste the recorded data points that you extract into an excel file that looks *exactly* like the one shown below (image below includes example data). Failure to do so will result in the program either not running at all, or outputting incorrect data.

5. Enter the data for all of the left, right and center minions, so that each possible position has all of the data from every minion in that position.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	LEFT	TIME	TIMETICK	FPOGX	FPOGY	FPOGS	FPOGD	FPOGID	CENTER	TIME	TIMETICK	FPOGX	FPOGY	FPOGS	FPOGD	FPOGID	RIGHT	TIME	TIMETICK	FPOGX	FPOGY	FPOGS	FPOGD

0.3 Running the Python Files

0.3.1 Choosing Which File to Run

Run the *RFL.py* file if you want to analyze the data from the 4corners and centerX required fixation points
Run the *MinonGameAnalysis.py* file if you want to determine the correction

factors based on data gathered during the minion game
Both python files are written to be fairly transparent, to account for the fact that not everyone who uses them will have a computer science background.

0.3.2 Initializing Parameters in Python

You need to tell python what the expected coordinates are, as well as what files it should be reading.
At the top of the program, there should be a clearly indicated region that has the parameters to edit. In this space, enter in the appropriate coordinates.
For the **RFL program**, these are the expected coordinates that you cal-cualted for each square.
For the **Minion Game**, the expected coordinates are as follows:

leftMinionX = 0.25
centerMinionX = 0.5
rightMinionX = 0.75
minionY = 0.5

You also need to tell python what file to read the data from. This is a few lines down from where the parameters are initialized. In the line

with open('filename.csv') as f:

change *"filename"* to be the name of the file where you collected the extracted fixation data.