Aalto-yliopisto
Aalto-universitetet
Aalto University

ELEC-E7852
Computational Interaction and Design

Assignment A3b

Saliency models

Aitor Urruticoechea Puig
aitor.urruticoecheapuig@aalto.fi
Student N°101444219
November 2024

Notice

Acknowledgement

Parts of the code implemented in this work were generated with the assistance of GitHub Copilot. None of its proposals were, however, employed without refinement and necessary tweaks to adapt it to the nuances of the tasks at hand; making it impossible to identify and subsequently mark which lines of code were human or machine-made, for many were the fruit of the combination of both.

Contents

List of Figures

1 Goals and Method

In A3a, some designs were (manually) iterated and run through the EyeFormer algorithm in a bid to predict the users' scanpath for different configurations of the landing page for the app (Figure 1). Now, however, the question becomes: how well did the algorithm predict these paths?

(a) Initial layout scanpath results (b) Coordinate-iterated scanpath. (c) Plus size-iterated scanpath.

Figure 1: Manual iteration results.

The main idea for this assignment is to compare this obtained scan paths with the ones that one can record using WebGazer. With that objective in mind, a website is going to be hosted as a baseline for the usage of WebGazer. This will first display a blank page for calibration, where the user will move the mouse around and click while looking at the mouse, so the measurements get reasonably exact. After that, the left arrow will serve as the main tool. After pressing it, each time one of the different layouts will be shown, thus having the user see through them as naturally as possible, as it will be the first time seeing them rather than a conscious scan. All this process will be recorded, which will allow us to afterwards do a frame-by-frame analysis of the highlighted points to see if the path is similar or not to the predicted by EyeFormer. See here a snippet of the implemented code to get that data, which will be wrapped inside a pretty standard HTML file for deployment. How this website looks like can be seen in Figure 2.

```
// Keyboard event listener to move through images
window.addEventListener('keydown', function(event) {
    // ArrowRight key (39) to show the next image
    if (event.keyCode === 39) {
        showNextImage();
    }
});
```

```
// Initialize WebGazer
webgazer.begin().then(() => {
    console.log("WebGazer started.");
}).catch((err) => {
    console.error("Error starting WebGazer:", err);
});

// Listen for gaze data
webgazer.setGazeListener(function(data, elapsedTime) {
    if (data == null) {
        return;
    }

    const xprediction = data.x;
    const yprediction = data.y;

    // Update the gaze pointer
    gazeDot.style.left = `${xprediction - 5}px`;
    gazeDot.style.top = `${yprediction - 5}px`;
    gazeDot.style.display = 'block';
});
```

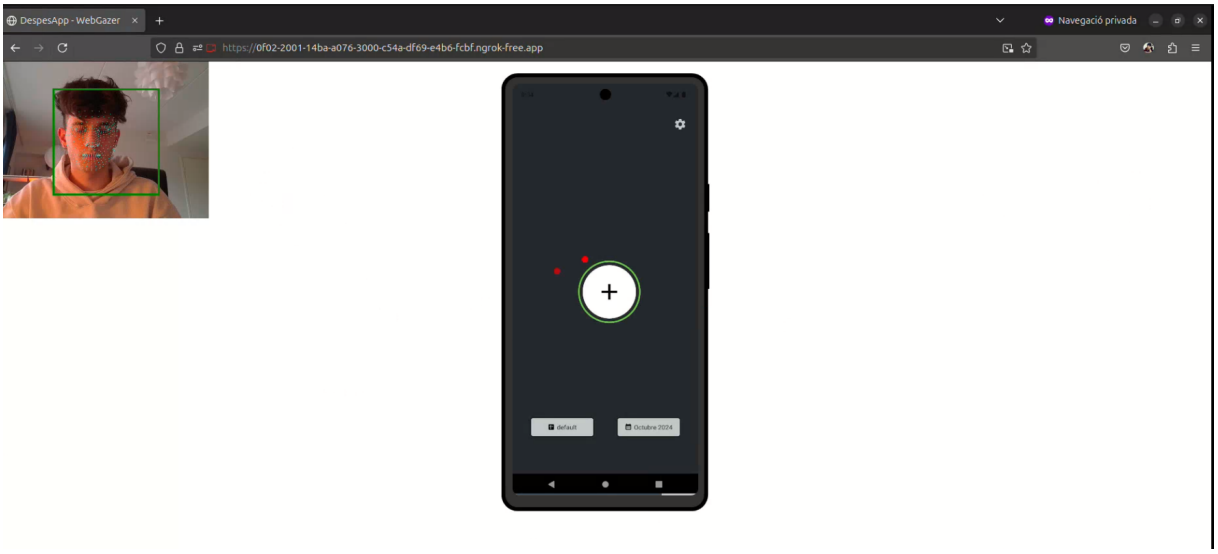
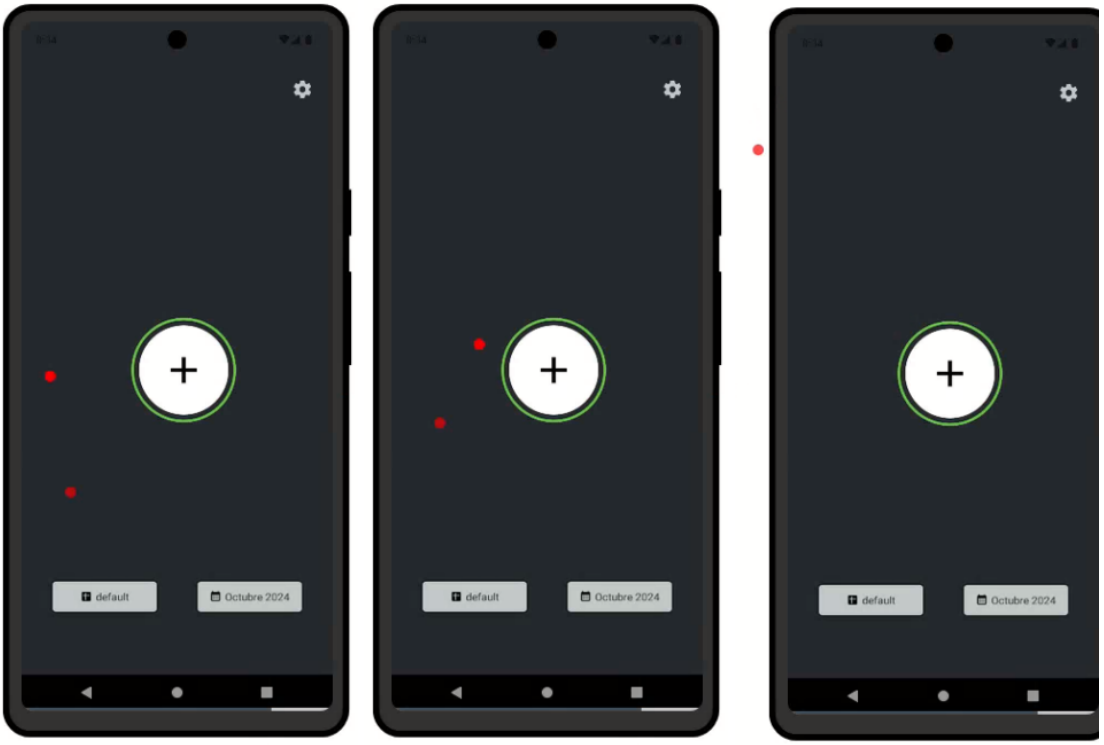


Figure 2: Screenshot of the developed website in action.

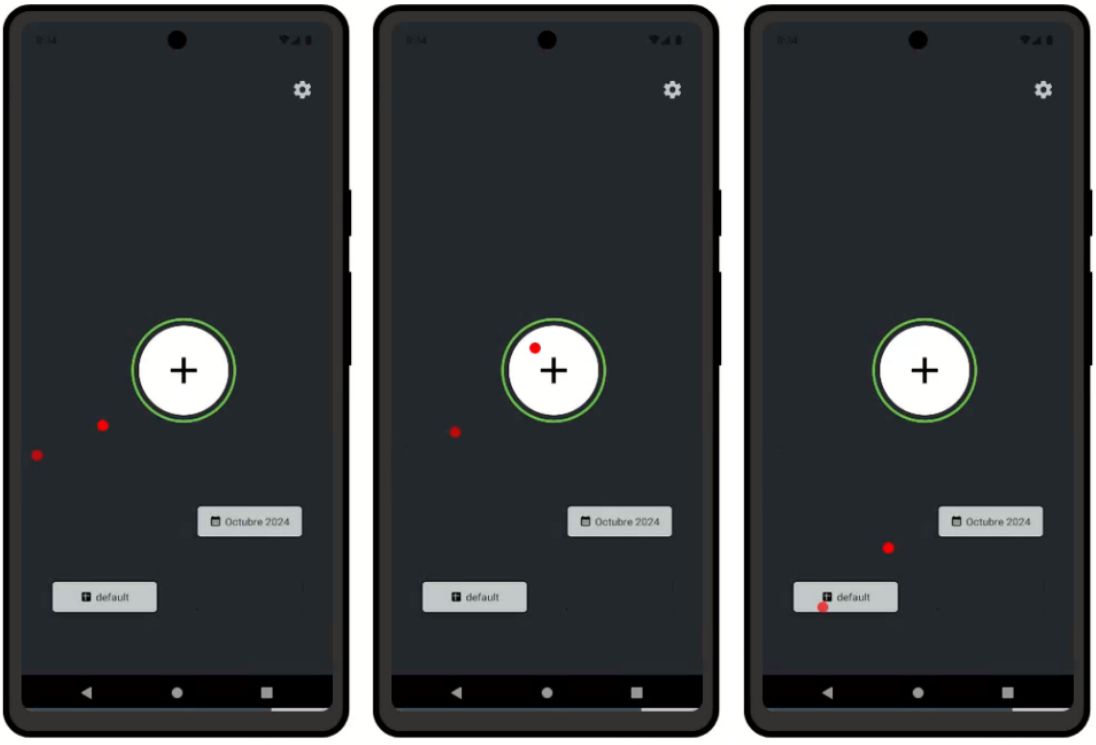
2 Results

After carefully working with the captured video, key frames have been extracted for each of the three designs (see Figures 3, 4 and 5).



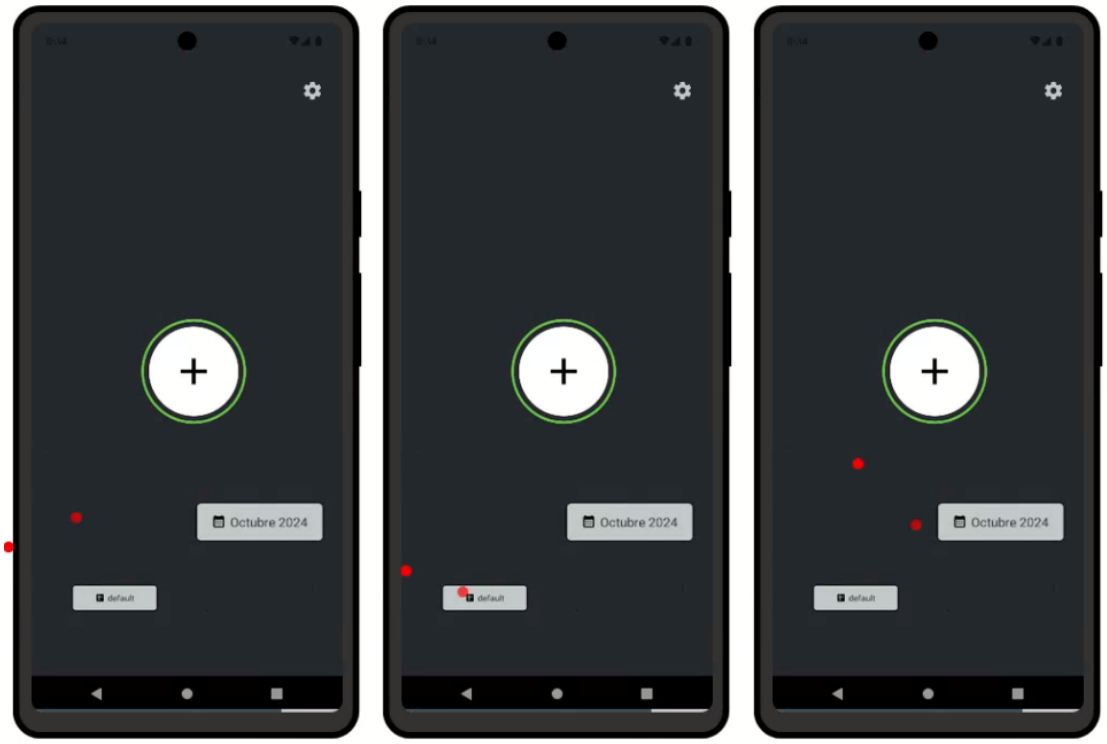
(a) User gaze at $t = 0s$. (b) User gaze at $t = 0.5s$. (c) User gaze at $t = 1s$

Figure 3: User gaze results for base design.



(a) User gaze at $t = 0s$. (b) User gaze at $t = 0.5s$. (c) User gaze at $t = 1s$.

Figure 4: User gaze results for first iteration design.



(a) User gaze at $t = 0s$. (b) User gaze at $t = 0.5s$. (c) User gaze at $t = 1s$.

Figure 5: User gaze results for second iteration design.

It is clear that there exist quite a difference between these results and Figure 1. For the base case, it seems like focus is mostly put in the center button, rather than center then down as predicted. For the other two, things get more interesting: For only position iteration, it seems like the scanpath goes from center, then down. However, when sizes are scaled differently, the bottom-right button becomes more prominent and the scanpath seems to go the other way around: from bottom to center button. This makes the second case the only one correctly estimated by EyeFormer.

3 Discussion and Conclusion

The sample tested here is but a small peak at what could be studied with these tools, and it is not in any way conclusive. One has to wonder as to how the prediction made by EyeFormer would have resulted if it had been feed more specialized content like mobile UIs, rather than or as an addition to the very diverse database provided. Similarly, the recording had probably a very short timespan, which made comparisons possible only for the first part of the path predicted by EyeFormer. Also, this is a single user, exposed one time to each design, rather than a more extensive study with multiple tests and multiple diverse users; which is only worsened by the fact that the user doing the testing is highly biased as a designer of both designs and testing methods. All in all, however, and based also on the unintuitive results generated by EyeFormer, I would preliminarily venture to suggest that the predicted results are not very reliable for UIs that are this simple, with unremarkable backgrounds and just 3-4 distinguishable features (in this case, buttons); and probably achieves better results in rich environments like posters, websites, etc.