Project report: Gaze-driven app for infants (DGI18)

Jonas Nockert, nockert@kth.se (supervised by Christopher Peters).

Project blog: `https://lemonad.github.io/ui-for-infants/`

August 26, 2018

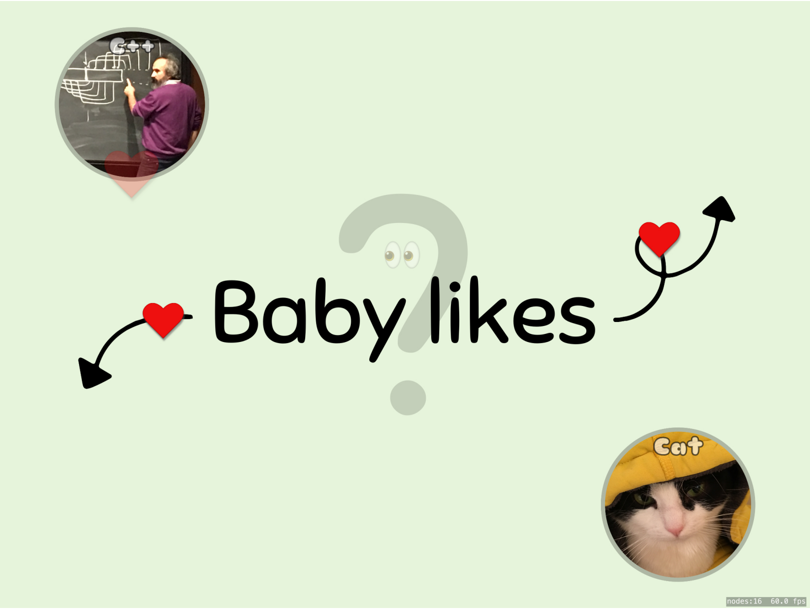

Figure 1: A screenshot of the app running on an 2017 iPad 10.5". For videos, please see the project blog.

Abstract

I present an iPad eye-controlled app intended for use by infants before they acquire the necessary motor skills to use touch-based interfaces. The idea is based on a paper by Vidal et al., 2015, using the correlation between pupil and on-screen movement patterns to conclude that a specific graphical object is being tracked by the user.

The app is written in Swift, Objective-C and C++, using SpriteKit for the user interface, with Dlib and OpenCV for face landmark and pupil detection. The pupil detection algorithm is implemented based on a paper by Asadifard and Shanbezadeh, 2010.

Background

Childrens' fine motor skills take time to develop. At 12–18 months, children begin using their index fingers to point to things, like pictures in books. Vision, on the other hand, develop earlier. Infants begin tracking moving objects soon after they are born and at about five months, their ability, when it comes to tracking horizontally moving objects, is adult-like[1].

Pursuits[2] is a technique that enables interaction with graphical

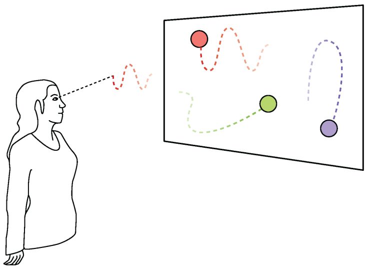

Figure 2: Some examples of Pursuits gaze patterns (ibid., p. 8).

[1] Grönqvist, 2010.

[2] See Vidal et al., 2015; and the related video Vidal, 2013.

devices using only gaze. It introduces a new kind of graphical user interface element that is based on movement (see figure 2). A user selects an element by following its specific movements.

Pursuits utilizes the smooth pursuit movements of the eye, which is a type of movement that only happens when we are following something with our eyes. Most people can not reproduce this movement on their own, which means that triggering false positives while "just looking" can largely be avoided.

As this technique does not depend on having to identify the position on the screen a user is gazing, only that the gaze is moving in a specific pattern, it seems to be less dependent on exact readings and, better yet, calibration is not necessary as only relative eye movements are relied upon.

Why an app?

The choice of creating an iPad app was based on, first, that the screen size enables large moving objects while still having trajectories resulting in eye-movements that are big enough to be accurately picked up by a front-facing camera. Second, a larger interface would make it easier for an infant to learn to use the interface, compared to, say, the small screen of a mobile phone. Third, tablets, in my experience, represent a type of digital device that parents are most comfortable with their children using. Fourth, developing an app makes it easy to distribute the application to other parents. Fifth, the later generation iPads have front-facing cameras with good resolution as well as the processor power to handle computer vision tasks while simultaneously displaying smooth movements.

Research question

The overarching research question is: suppose that it is the fine motor skills of infants that limit their use of touch-based tablets, not their cognitive abilities. Then, what if an alternative non-touch user interface based on Pursuits was designed, would they be able — and perhaps more importantly — would they want to use that earlier?

The minor resarch question is: could such an interface be constructed, given that the original Pursuits paper is based on a head-mounted head-tracker with good precison and not on images from the front-camera of a mobile device?

Implementation

The starting point was the minimal sketch from the project description from which a more detailed sketch emerged (figure 3). The completed app consists of two different horizontally moving images acting as the main user interface controls, each one having an indicator, in the form of a heart, showing when the user is tracking the control. There is also two progress indicators, also in the form

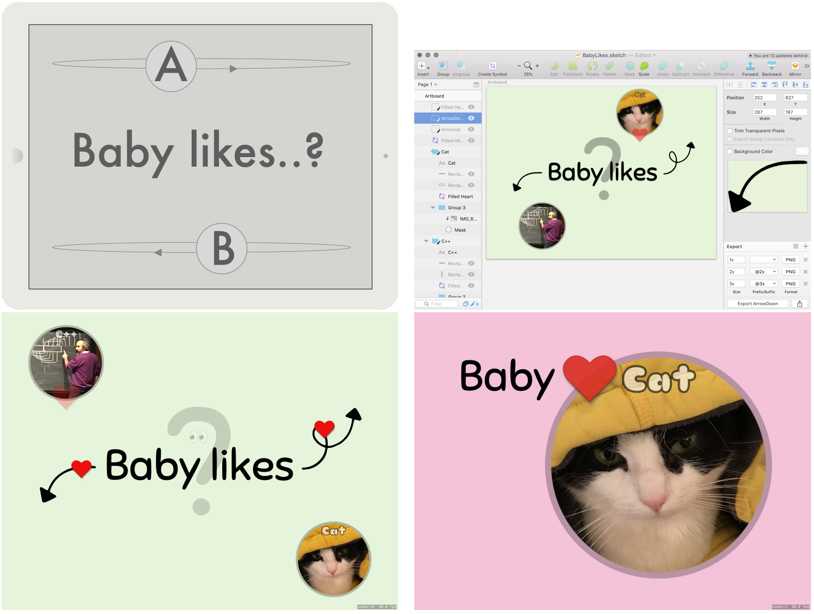

Figure 3: Sketches of the app interface as well as images from the final app with two Pursuits controls.

of hearts, moving along the two arrows. When a heart reaches the arrow tip, the image which the arrow points to "wins" and the user is treated to a screen with a larger version of the image that won (figure 3).

The idea here is that the parent leaves the app running so the infant will use the app, over time, in short intervals. The child will be more drawn to one image than the other and that image will be shown to the parent.

The main interface of the iOS iPad app was programmed in Swift, with Objective-C classes providing a bridge between Swift and the C++ libraries used (OpenCV and Dlib). It uses the SpriteKit framework to provide 60 fps smooth motion while doing heavy image processing in the background. Smooth motion is needed partly because Pursuits is based on smooth pursuits but mainly due to it is tiresome for a user to follow a stuttering object on-screen.

Correlation between pupil and on-screen movement

While the app is running, on-screen movement as well as pupil movements are collected and tested for correlation using the Pearson product-moment correlation coefficient method, a statistical measure $[-1, 1]$ of how similar two variables are to each other. That is, if the pupils and an on-screen object moves in the same way, the Pearson correlation coefficient will approach 1, if the pupils move in the opposite direction, the coefficient will approach -1. If there is no correlation, the coefficient is zero.

It is important to note that the coordinates compared are only relative, which means that the gaze is actually not taken into account. As long as the pupils have a similar movement pattern (invariant to scale and origin) as something else, the correlation will be high. In this case, since the moving objects have opposite movement patterns (sin vs. cos), the correlation will be high for both objects at the endpoints. This because the pupil movement will be small at the same time that both objects are moving very slowly. Thus care has to be taken to include enough previous positions in the correlation calculation so that tracking of the wrong object is not registered.

Specifically, pupil data from the last 30 frames (1 second) of camera captures is collected and correlated to the on-screen movements (every other frame at 60 fps). Left and right pupils offsets relative to the inner eye corners are averaged and if the correlation with an on-screen object is above 0.6, the interface registers that object as tracked. This causes the progress indicator for that object to move along the arrow. Once correlation goes below 0.6, the progress indicator stops again.

Face and landmarks detection

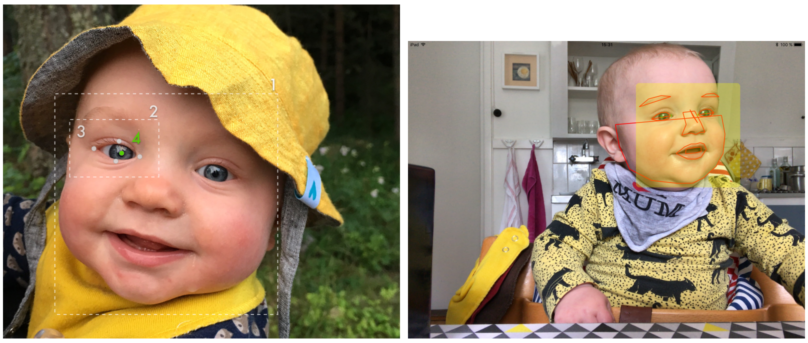

Figure 4: Different levels of detection usually needed in order to extract pupil positions. On the left; face, eye and pupil detection. On the right; face and face landmark detection, including eye region (from Apple's Vision framework) Authors own images..

The process of obtaining the pupil position starts with extracting the area of the face. Apple's Vision framework, Dlib and OpenCV were tested as a basis for face detection but eventually Apple's oldest framework AVCapture was used (fast but with low accuracy).

Once the face is located, Dlib five-point landmark detection is used to obtain approximate positions of the eye corners as well as a point below the nose. These five points are then used to transform the image of the face so the eyes are as level as possible. This makes it easy to get both horizontal and vertical position of a pupil relative to the eye corners even if the face is tilted to the side.

Kalman filtering

The problem is that the pupil movements are very small and face and landmark detections are not very stabile so even if the relative pupil position is easy to obtain, the readings will have substantial noise.

Thus, Kalman filtering is used to try to minimize the amount of noise while still allowing for actual movement of the landmarks and pupils. It is important that as little lag as possible is introduced in the process of sampling the pupil positions as that will have a negative effect on correlation.

Pupil detection

From the eye corners, an approximation of the eye area can be deduced (Baggio et al., 2017). The pupil tracking algorithm is implemented based on a paper by Asadifard and Shanbezadeh, 2010. At a high level, the histogram of the image is used to create CDF for the probability of each individual gray levels. This is then used to create a mask of the 5% darkest pixels, i.e. the subset of pixels likely to correspond to the pupil (cf. figure 5).

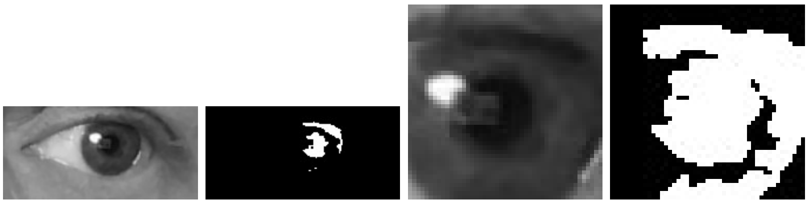

Figure 5: Input image and subimages in the process of getting a position of the pupil. The far left image is the input. The center-left image is the mask containing the 5% darkest pixels. The image to the right is the window around the location of the darkest pixel in the (masked) original image. The centroid of the mask to the far right is finally returned as the pupil position.

Next, the mask is eroded using a 2×2 kernel and applied to the original eye image while looking for the location of the darkest pixel. Again, this is likely in the area of iris or pupil. A window around this location is used to calculate the average intensity, from which a mask is created for any pixels below this intensity.

Finally, the centroid of this mask is calculated, which should correspond to the center of the pupil. However, in my case, reflections from windows and lights were clearly visible as bright regions in the pupil so the detected pupil center were always offset. As the Pursuits method is only concerned with relative position, this issue does not matter here but would be a problem for gaze-detection.

Conclusion

The completed app and face landmark detection testing on infants (cf. figure 4) is a clear indication that this concept could work on infants on a technical and physical level. The minor research question could definitely be answered with a yes.

If it works on a cognitive level is, however, a completely different question and needs to be addressed in a perceptual study with infants having not yet learned touch-control. Thus, the major research question is left unanswered.

Potential perceptual study

A user-study with infants and their parents could give tremendous insight here. Perhaps first in terms of obtaining some form of indication that the underlying idea of apps for infants being sound or not, but also

- qualitatively: parents' general attitude towards apps for infants and toddlers, their expectations beforehand, reflections on their children using the app as well as their thoughts on directions the app could take in order to enable more interesting experiments.

- quantitatively: do the results differ much between children, how much do the settings and surrounding matter, if the parents use the app on their own, over time, do they get different results?

Specifically, one would want to know if this app has potential to be developed into something truly useful for parents as well as infants.

Generally, one would want to learn more about how to design applications for infants. Very few intuitions we have from user interface design for adults likely hold when it comes to infants so foundational questions such as how much movement should be used is not yet known. Both in terms of infants being able to process and track movements and, equally important, keep them interested long enough to be able to collect useful data.

Additionally, infant-oriented pursuits-based user interface design needs to take into account both parent and child, weighing how much parent information distracts the child with the need for the parent to get some idea of what is going on on-screen.

With that said, a user study with infants is probably very difficult to perform, especially since the timing for an infant is so important (not hungry, not sleepy, etc.) A more realistic, albeit uncontrolled, user study would be to get parents to use the app in the comfort of their homes, at the best time for their child. If just *one* child can learn to use this kind of interface, it seems reasonable to expect that with additional work, the design could be improved to enable it to work for more children.

References

Asadifard, M. and J. Shanbezadeh (2010). “Automatic Adaptive Center of Pupil Detection Using Face Detection and CDF Analysis.” In: *Proceedings of the International MultiConference of Engineers and Computer Scientists , Vol I.*

Baggio, D. L. et al. (2017). *Mastering OpenCV 3 - Second Edition*. 2nd. Packt Publishing. ISBN: 1786467178, 9781786467171.

Grönqvist, H. (2010). “Visual motor development in full term and preterm infants.” PhD thesis. Uppsala University. ISBN: 978-91-554-7892-6.

Vidal, M. (Mar. 2013). *Pursuits: Spontaneous Interaction with Displays*. Youtube. URL: https://www.youtube.com/watch?v=TTVMB59KvGA.

Vidal, M. et al. (Jan. 2015). “Pursuits: Spontaneous Eye-Based Interaction for Dynamic Interfaces.” In: *GetMobile: Mobile Comp. and Comm.* 18.4, pp. 8–10. ISSN: 2375-0529.