Clare Boothe Luce Final Report

Alana Huitric

Abstract

With augmented reality technology that can accurately track a smartphone's motion through 3D space becoming more prevalent, assistive technology researchers saw the opportunity to utilize it for navigation purposes. For the past few years, Paul Ruvolo and his team have been developing *Clew*, a free iPhone app that helps blind and visually impaired individuals navigate recorded routes indoors. This summer, my work has centered around the addition of rerouting and pausing capabilities for the *Clew* app. This report will conclude with the discussion of future opportunities and limitations of the app.

Introduction

Although there are many existing applications and devices intended for the assistance of blind or visually impaired individuals (B/VI) with navigation such as Blindsquare and Nearby Explorer, these apps are typically used for outdoor navigation and are reliant on GPS technology, and difficult to use in confined, unmapped indoor spaces.[3] The *Clew* app seeks to provide an improved and free indoor navigation experience to supplement Orientation and Mobility (O&M) techniques such as a cane or guide dog, without the usage of GPS.

The purpose of this summer research was to improve the *Clew* app for B/VI. By adding features such as rerouting and pausing the *Clew* app's functionality would be improved. Currently, the version of *Clew* available in the app store allows users to save routes, recall single-use routes, and navigate through existing routes. When the user navigates an existing route, they must stay on the path and continuously use the app until the user has reached the end.

My research this summer aimed to add more functionality by adding a rerouting feature that would allow users to record a temporary detour along a route, and navigate back to the original route or help them get back to their original route if they turned around or unintentionally deviated from the path.

Methods

Clew Overview

Rather than GPS, *Clew* uses ARKit – Apple's augmented reality API [1,2]– to help the user navigate their surroundings. In the app, the user can either follow single-use or saved routes in *Clew*. For the former, the user records a single-use route that can be immediately navigated back in the opposite direction once they reach their destination. This route is only used when the user navigates back to their starting point. For a saved route, the user must first record a route the same way as a single-use route, saving an anchor point – a point that helps the user re-orient to the augmented reality map for later use – at the beginning and end of each route by holding their smartphone against a surface to orient themself to the path for future uses. These routes are then saved with a name and additional voice notes about their anchor points. Then, the saved routes can be navigated from start to end or vice versa. Routes are created by augmented reality 'crumbs' that are dropped while the user is recording along the path. Crumbs are pieces of location data that are created at regular time intervals while recording, and are saved in the same order they were created in. These crumbs are condensed into keypoints that comprise the major turns in the route, with anchor points at the beginning and end of the route. While navigating, the user receives haptic

feedback, and when they reach a keypoint they receive directions to the next keypoint.

Progress

I began the project by learning the Swift programming language within the XCode IDE for creating iOS applications. The first feature to be implemented was adding a toggle-able announcement feature that tells the user which way to turn if facing away from the next keypoint along their route providing they are still within bounds of the path to the next keypoint. This feature was created by formatting pre-existing app methods to tell the user to either turn left or right instead of clock directions until they receive haptic feedback. As seen in **Figure 1** on the next page, the top arrow indicates that the user is facing out of the acceptable trajectory range and that they would receive an announcement to "Turn right until (they) feel haptic feedback." If following the bottom two arrows, the user would not receive the message to turn until they feel haptic feedback as they are within acceptable trajectories to the next keypoint.

This feature was implemented largely by combining older methods to create a new path distance calculator. This new feature aids the user in quickly facing the correct direction, however I believe this feature may be more useful to those less familiar with the app as those who use it more frequently already know to sweep their phone left and right to receive haptic feedback on their position along the path.

Next, the more complex, toggle-able rerouting feature was implemented. This feature allows the user to deviate from the path of a recorded route and return to that route along the same path they took walking away from that original recorded route. The rerouting/detour feature is activated when the user walks outside of a certain threshold distance of the calculated path between route keypoints (currently set at 0.2m) as seen in **Figure 2** on the next page. Once the user is alerted of their deviation from the path, the app records the index of the crumb it last dropped while on the path. At this time, the next keypoint in the route is inaccessible until the user hits the reroute button which appeared when they left the original path. The user must press the reroute button to be guided back to the original path using the crumbs dropped when they first walked away from it. New keypoints are created from the crumbs dropped out of the bounds of the path, seen in purple in **Figure 3** on the next page. This feature was implemented by having the app dropping crumbs while navigating as if it were recording a route. As seen in the same figure on the next page, not all crumbs are used to make a purple detour back to the original route in red. The interesting part about implementing this feature was calculating when the user had walked outside of the threshold of the path width. This was done using vector projection which I had recently learned and was excited to use in a real life application.

The final feature I worked on allowed users to pause the app while navigating. While there were many methods in place to pause the app while recording a route or create anchor points, they were not compatible for this usage. However I learned much more about how sessions, timers, and ARkit in general worked through implementing this feature, by breaking something that had to do with each aspect. This feature is still in development but currently works with a few minor bugs including: unnecessary route relocalization, keypoints now rendered in black, and occasional crashing.

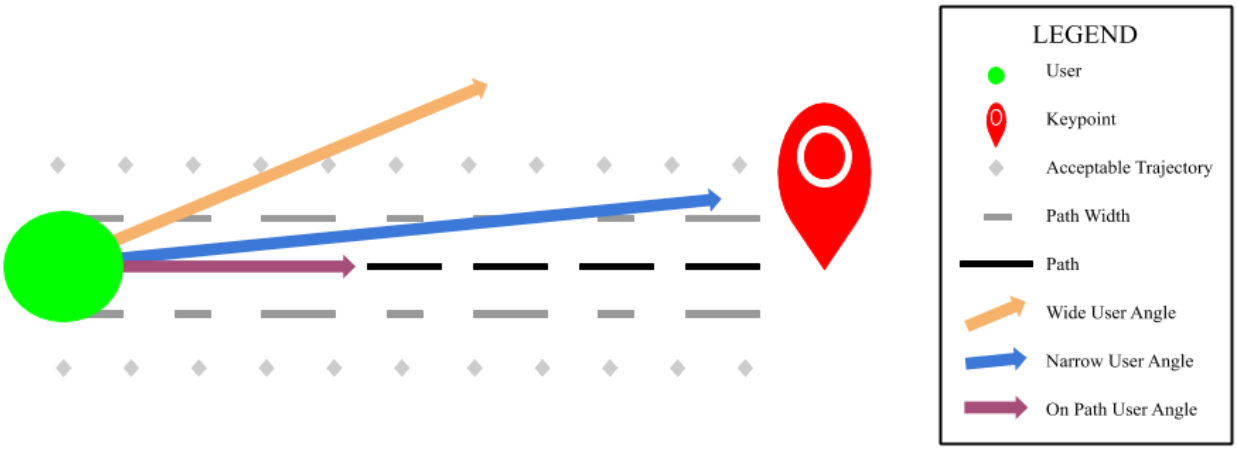

Figure 1. Haptic feedback tolerance and user angles. The purple and blue arrows would trigger haptic feedback whereas the tan arrow would prompt an announcement giving the user directions.

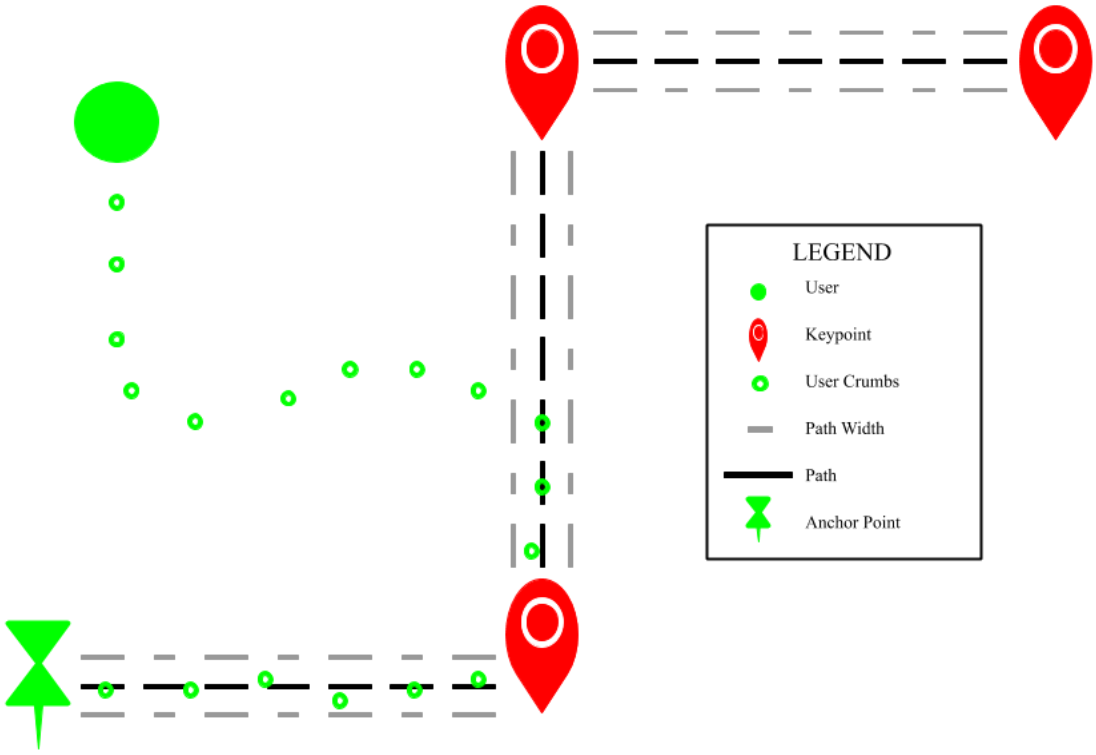

Figure 2. User walking off the path in a complex detour. Before rerouting was implemented, the user would be directed to the second keypoint in a straight line, regardless of any obstacles in the way.

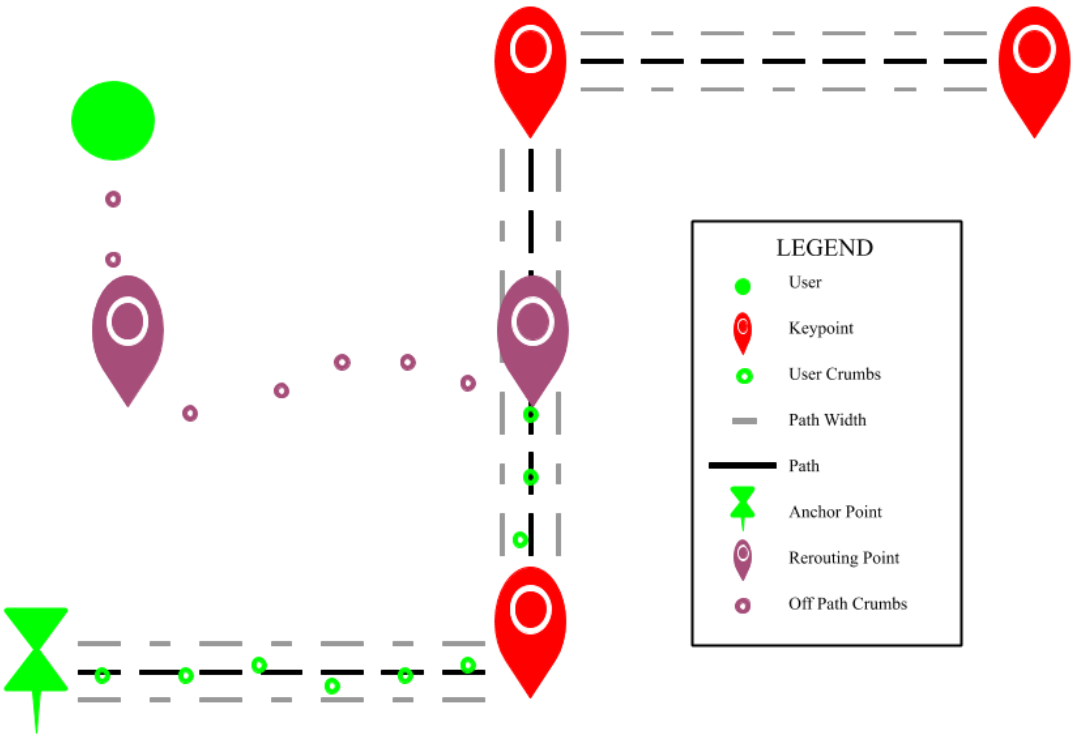

Figure 3. Detour created after the user has pressed the reroute button. The user now has the option to navigate back along the new, purple keypoints.

Reflection

Besides feedback from my advisor, Paul Ruvolo, I was able to get a lot of pointers from an interview with one of our users. I was able to tell them about the new features I was implementing and how they worked. I confirmed that the first feature I created was not useful for an experienced user like them, as the user's main interest was in the more complicated rerouting feature and pausing. However, the feature would have been helpful for an inexperienced user. They thought both of these features would be useful to them in the oddly shaped building they use *Clew* to traverse. In addition to providing feedback on current and planned features, atheylso proposed that in the future, it would be nice to be able to skip keypoints in the route.

This summer I learned a lot more about O&M training as well as the broader world of assistive tech. Though browsing forums and talking to a user of the app I found out more about what makes an assistive app useful and desirable, such as price range, simplicity, and accessibility. Creating expensive and complex devices is not always the most desired solution, and sometimes a free simple app is useful.

Future Work

In future work on the app, further debugging of the pause feature is necessary before it can be released. From user feedback, the ability to skip keypoints in a route may also be a desirable feature as it will allow users to continue on a route that is partially blocked by an obstacle – such as a group of people or new furniture – obstructing a previously accessible keypoint.

Acknowledgements

I would like to acknowledge funding received from the Clare Boothe Luce Research Scholars Program at Olin College, funded by the Clare Boothe Luce Program of the Henry Luce Foundation. Additionally, I would like to acknowledge my advisor, Paul Ruvolo, for his guidance, mentorship, and support during the course of my summer research.

References

1. Inc., Apple. "ARKit - Augmented Reality." *Apple Developer*, developer.apple.com/augmented-reality/arkit/.
2. Inc., Apple. "ARKit - Documentation." *Apple Developer Documentation*, developer.apple.com/documentation/arkit.
3. "Three Blindness-Aware Mobile Navigation Apps." *Three Blindness-Aware Mobile Navigation Apps | American Foundation for the Blind*, www.afb.org/blindness-and-low-vision/using-technology/smartphone-gps-navigation-people-visual-impairments/three.