2017

STRIDE | Southeastern Transportation Research, Innovation, Development and Education Center

Final Report

Cycle Atlanta SWIFT Development

(Project number 2016-007)

Author: Dr. Kari Watkins, Ph.D., Georgia Institute of Technology

April 2017

DISCLAIMER AND ACKNOWLEDGMENT OF SPONSORSHIP

TABLE OF CONTENTS

LIST OF FIGURES

List of all authors with their affiliations and contact information

Dr. Kari E. Watkins, PE (P.I.), *School of Civil and Environmental Engineering, Georgia Institute of Technology; Email: kari.watkins@ce.gatech.edu*

Dr. Chris LeDantec (co-P.I), *School of Literature, Media and Communication, Georgia Institute of Technology; Email:ledantec@gatech.edu*

ABSTRACT

Cycle Atlanta (http://cycleatlanta.org/) is a mobile application that records a cyclist's bike route using the geolocation data in the phone. Since releasing the app, smartphone operating systems have evolved at a dramatic pace. The accumulated changes to the underlying software platforms and application programming interfaces (APIs) have resulted in the app no longer working reliably, which jeapordizes our ability to continue with our research program to build robust empirically based models of cyclist behavior. The work conducted here directly addressed the need for a substantial revision of the Cycle Atlanta app for Apple's iOS, which accounts for over 80% of our historic and active user base. The updates required a substantial development effort to re-write large portions of the app in order to maintain functional parity on current versions of iOS. Re-writing the components in Swift and re-architecting core functions stabilized the app to make it easier to manage when iOS updates in the future. The app is now compliant with current App Store requirements and Apple's current Best Practices. The code is also much more compact and easy to understand and follow to make it easier for any future work that might be performed by a student.

BACKGROUND

Cycle Atlanta (http://cycleatlanta.org/) is a mobile application that Dr. Chris LeDantec and Dr. Kari Watkins at Georgia Tech created in 2012, based on the original CycleTracks open source codebase from San Francisco County Transportation Authority. The app uses a smartphone's geolocative capabilities to record a cyclist's bike route as she travels to her destination. Images from the app itself are shown in Figure 1.

As the app runs in the background, it acts as a passive form of citizen-sensing: the cyclist gathers route data as she rides, which is later sent to the research team at Georgia Tech to inform transportation planners with the City of Atlanta. For the initial implementation of the app, an Android version was created and substantial code revisions were made to update the iPhone version. The Cycle Atlanta project team has already contributed substantial revisions to the open-source code base and plans to continue to do so throughout the project.

To date, we have been able to record about 20,000 trips by 1700 users and Aditi Misra wrote her dissertation about the route choices that cyclists make. One of the primary applications of Cycle Atlanta is to enable planners to use the origin, destination and route data in conjunction with existing planning data to determine where bike infrastructure is needed most. By collecting these data, we have been able to better understand route-choice among cyclists in the City by developing new route-choice models that take into account cyclist comfort along with demographic clusters. Riders are categorized by gender, age, race, income, cycle frequency, rider type (strong & fearless; enthused & confident, etc.) and rider history (length of time as a cyclist). Deviations from the shortest path are assessed using discrete choice models based on this data as well as infrastructure conditions (topography, traffic volume and speed).

Since releasing the app, smartphone operating systems have evolved at a dramatic pace. The accumulated changes to the underlying software platforms and application programming interfaces (APIs) have resulted in our app no longer working reliably which jeapordizes our ability to continue with our research program to build robust empirically based models of cyclist behavior to better inform transportation planning and infrastrcuture build-out.

Fig. 1(a)

Fig. 1(b)

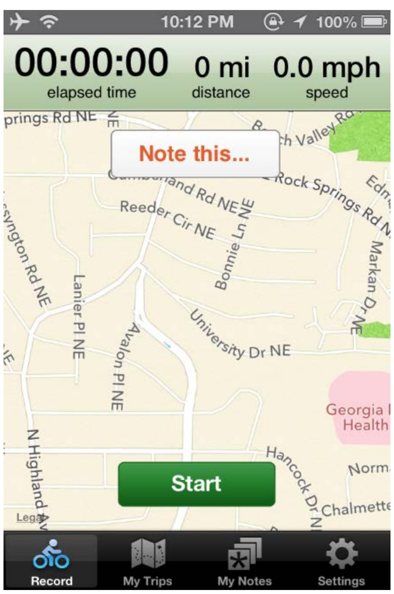

Fig. 1(c)

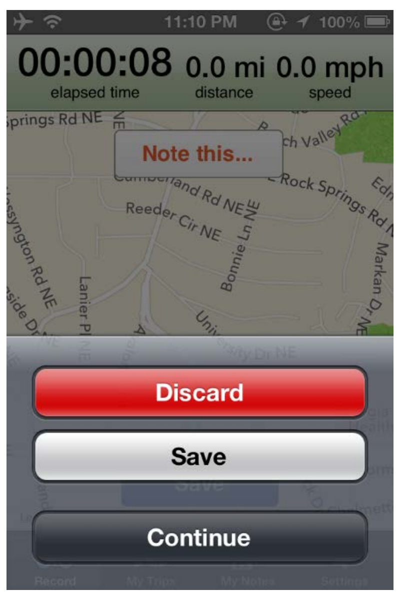

Fig. 1(d)

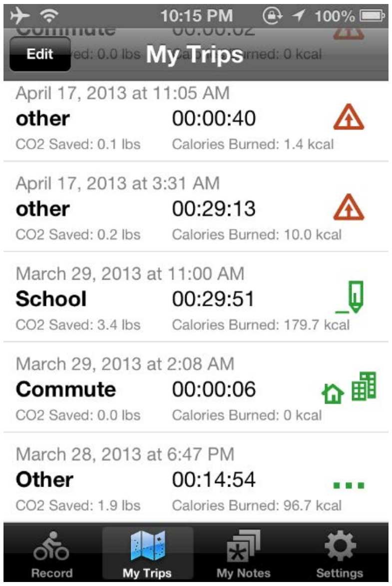

Fig. 1(e)

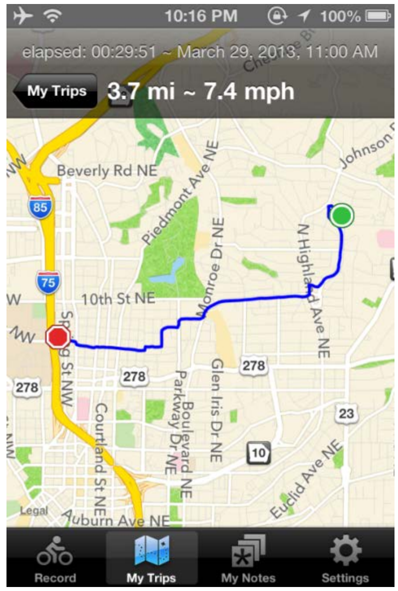

Fig. 1(f)

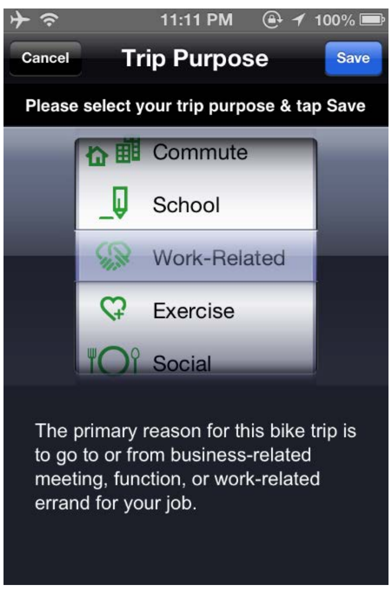

Fig. 1(g)

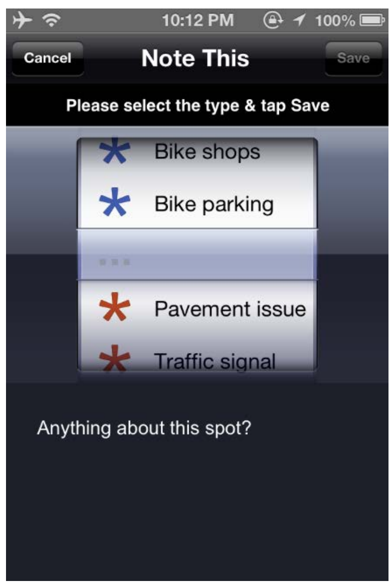

Fig. 1(h)

Figure 1. Cycle Atlanta user interfaces: (a) Socio-demographic information, (b) Riding characteristics information, (c) Trip start screen, (d) Trip end and record screen, (e) &(f) Viewing trips, (g) Choosing a trip purpose, (h) Adding a note on anything about a particular spot

RESEARCH APPROACH AND APPLICATIONS

The work conducted here directly addressed the need for a substantial revision of the Cycle Atlanta app for Apple's iOS, which accounts for over 80% of our historic and active user base. The updates required a substantial development effort to re-write large portions of the app in order to maintain functional parity on current versions of iOS.

MIGRATING TO SWIFT

The first step to the proposed work was to migrate the code-base to Swift, which is Apple's new and preferred environment for app development. Parts of this work were completed last year but much of the core components of the app—location tracking, data storage and transport, user interface and presentation layer—needed to be re-written.

Re-writing the remaining components in Swift stabilized the app to make it easier to manage when iOS updates in the future. It will also make integrating new features easier. This is particularly important as handset hardware includes new activity sensing capabilities that are only available through Swift libraries.

UPDATING THE EVENT MODEL

Migrating to Swift was necessary to keep the app compatible with Apple's smartphone operating system. The most important consequence of this was the need to update the location event handling of the app. In the current version, based on the best practices and APIs that were available in iOS6 and 7 (we are now approaching the release of iOS10), when a user started the app and recorded a ride, Cycle Atlanta would run in a 'background' state on the phone. Each second the app would request and receive location data which would be stored until the trip was finished and uploaded to our servers.

Since releasing our app, Apple has introduced a new method of handling apps running in the background. Instead of letting them run on their own, the operating system 'pauses' those apps and only wakes them up periodically so they can update their status. This approach is much better for battery life, because the smartphone can stay in a low-power mode longer. Upon waking the app the operating system would then provide cached data. In our case, that would be a cache of location data sampled at one second intervals.

In its previous form, Cycle Atlanta could not make use of these cached data and as a result no longer collects accurate data about cycling trips. The solution was straight forward and largely came as a result of migrating to Swift and re-architecting core functions in the app according to the current best practices provided by Apple.

PROCESS FOR UPDATING

The initial portion of this project was spent designing, building, and testing a sub-app that just displayed location on a map and recorded it to ensure that no matter what a user does with their phone, it continues to record. The test app loads MapKit, displays a map that tracks user location, and draws the path as a map overlay. It stores all the CoreLocation data (time, speed, accuracy, location, etc). The test app uses the new "deferred location updates" functionality, which allows it to be halted by the OS in the background if the user is doing resource-intensive things with the phone. CoreLocation continues to record data, but does not upload data to the app to conserve battery and processing. Periodically, the OS will “wake up” Cycle Atlanta and give it a big batch of updates all in one go. The app records all of these points as if it had been running in the foreground all along, including drawing the correct overlay path. Therefore, there is no longer any data missed while in the background, no matter what else the user might do with the phone. The test app now caches the current trip to persistent local storage every five seconds (seamlessly). The test app loads any in-progress trip automatically when the app is started, meaning even if the user does stop the app for some reason, we will still not lose the data from their trip up to the point where the app was killed.

The second portion of the project was spent integrating the code into the main CycleAtlanta app. The code from the test app was tested to make sure it is working properly and recording trips. It was then integrated into the main CycleAtlanta app by rewriting the main RecordTripViewController to Swift (from Objective-C). This required restructuring the “map view” portion of the app.

The app is now compliant with current App Store requirements and Apple’s current Best Practices. The code is also much more compact and easy to understand and follow to make it easier for any future work that might be performed by a student. Making the code base easier to follow is necessary as students have limited time and are just learning their coding skills. However, to keep this as a viable research project, we must have predictable results when students are contributing to the code base.

CONCLUSIONS, RECOMMENDATIONS, AND SUGGESTED RESEARCH

STRIDE funds contributed to the initial and continuing development of our Cycle Atlanta app. Doing this maintenance work to the app was an absolute necessity to keep it viable as a research platform.

This work allows the app to stay active for future research in this region and others that have reused our code base for their own versions. We have several future projects in mind that this work will enable with a much quicker development time. One component to the app would allow cyclists to pinpoint near-miss locations. Cyclist safety data is notoriously bad and crashes are infrequent enough that new data is needed. An additional function would directly send notes that we receive in the app about potholes, parking in bike lanes, etc to the appropriate people within a city to deal with them. Both of these are planned as future research projects in crowdsourcing cycling data.