SOFTWARE PROTOTYPE SPECIFICATION

for

$NAVATAR$

Version 1.0 approved

Prepared by Matthew Berger, Liam Gomez, and Connor Parkinson

Advised by Professor Eelke of the UNR CSE Department

December 16, 2016

Contents

Revision History

Name	Date	Reason For Changes	Version
Matthew Berger	12/14/16	Created Document	1.0.0

1 Abstract

Navatar is an indoor navigation system for visually impaired students. Unlike existing systems that rely on expensive equipment, Navatar only requires an Android smartphone. The phone's GPS, accelerometer, and compass are used to approximate a user's location and movements and the phone's speaker is used to provide auditory directions to rooms within a building. Buildings on University campuses often have the GIS maps and the design required for Navatar's accelerometer and compass based localization, which is why students are the intended audience for the project. With the application's open source status and modular design, crowdsourcing efforts can be used to integrate maps for buildings on any campus. The application is still under development and Team 6 is working on prototyping a variety of new features alongside other open source contributors.

2 Introduction

The main goals and objectives of this project are to help blind people navigate indoor environments using a mobile phone and open source software which allows for a very low overall cost. Most people already have a cell phone, and mobile phones have a wide array of sensors available for environmental information. Accessibility is a major issue for many students with disabilities, and the technology necessary to ease the burden of some tasks that can be needlessly complicated is available and waiting to be utilized. Currently, there is no free and open source software capable of gathering environmental information and assisting students with indoor navigation for an entire college campus.

The high level requirements of this software include being simple to use, low-cost, scalable, maintainable, and efficient in its assistance to blind users. The intended audience of this project, as stated, is blind students and partially blind students. Existing and familiar environments will be conquered through the use of landmarks and bookmarks. A bookmark in this case would be a landmark that the user sets manually, that gets stored specifically for that user. The reason behind this is that many partially-sighted and blind users remember details that sighted people may not utilize or need, such as a metal strip on the floor, that help them remember where they are in an indoor setting relative to where they want to go.

The project itself currently exists as a repository on GitHub that contains a mobile android app. The members Team 6 all currently have a fork and have been working on learning the code base and developing prototypes of a few simple features. The future of this software depends greatly on what the team is capable of achieving in the time-frame provided, but the team plans to implement many features. The first planned enhancements, which the team has prototyped, are the implementation of geofencing to determine what building the user is in, the addition of navigation history to allow users to repeat routes taken previously, and adding gesture based input functionality for allowing the user to declare and input their own landmarks. Following stakeholder interviews, the team has decided to modify the location history feature to include the ability to backtrack on previously taken routes. The team also plans on investigating collecting route history for compiling statistics on if routes are available and completable, and testing the multiple methods for obtaining a user's location for geo-fencing. Possible future enhancements continue to include third party device integration to improve usability and accuracy (i.e. smart watches, Bluetooth earpieces), using additional environment information such as wifi SSIDs, making the app more scalable, improving map conversion and creation, user profile learning, stride detection, and gesture-based UI/UX improvements.

3 Prototype Objectives

The team's primary objective when developing the prototype is to learn as much about the design and functionality of the existing code base as possible. The team has limited experience with developing applications for Android and will have to spend considerable time becoming familiar with Android Studio and Java. Once the team has sufficient understanding of the code and the Android development process, the team can then refine its plans and designs for new features. When developing the prototype, the team plans to integrate modular features which do not require extensively modifying the existing code and functionality. Modular features will allow the team to work around any bugs or limitations in the existing code and test new functionality efficiently.

4 Prototype Functionality

The items the team chose to prototype are small features that mainly focus on improving the overall user experience of the existing application. The items that were chosen to be prototyped are also modular–having minimal interaction with one another.

The first item chosen to prototype is adding the ability to auto-detect the user's current campus and building. In the existing application these two pieces of information have to be manually entered by the user. By automatically detecting these, the number of inputs the user is required enter is cut in half. This prototype can be developed by geo-fencing the campus and buildings that are loaded into Navatar. The devices geolocation API will then be used to provide GPS coordinates which are used to place the user's current location inside of a known geo-fenced area. The second item chosen to prototype is integrating gesture based input methods into Navatar's UI. These gesture based inputs can be used to trigger a number of different actions when using Navatar. For example, by pressing and holding the screen of their device the user could add a new landmark at their current location. The types of gestures that can be used is somewhat limited as shaking or moving the phone would affect the accelerometer and compass which would interfere with Navatar's core path planning and localization algorithms.

The third item chosen to prototype is recording a user's navigational history giving the user the ability to repeat navigations they had previously made. This feature eliminates the tedious task of inputting information for routes a user frequently navigates. It was also a suggested as a potential improvement by participants in Navatar's early user studies. Recording a user's navigational history is also beneficial for the purpose of analytics. By analyzing a user's route history along with the success or failure of the navigation potential improvements to the application might be realized.

The items not chosen to be prototyped are more complex in nature, and would require extensive development efforts. The items not prototyped included third party device integration, accuracy improvements, streamlining map creation, and user profile learning. With the team's limited experience with the existing codebase and android development techniques, the team felt more comfortable prototyping smaller features and enhancements.

5 Prototype Development

Attached in the prototype document submission is a zip file which includes all three prototypes. The zip file contains the source code for each prototyped feature along with a pre-built apk to be installed on any Android smartphone. It can also be found online using the link below:

Click here for the finished prototypes of Navatar Features.

Because these are prototypes, their stability is somewhat fragile. Below are some notes to prevent known errors that may arise while using these prototypes.

5.1 Navatar-AutoLocate

- The application launches and immediately request location permission from the user. This is currently set to loop if the user rejects permissions. Once the app has the necessary permissions, the user can then select "Auto Locate" and the GPS is polled for the user's location.
- If the GPS is disabled, the settings page is automatically opened allowing the user to enable it.
- Locations are polled until one is collected within 11 meter accuracy at 68% confidence, or five inaccurate locations are collected. If a location is found be on the UNR campus and in a building, the user is automatically taken to the appropriate room selection screen.

5.2 Navatar-Gestures

- Gesture-Detection is controlled through a rectangular element on the navigation screen. This is toward the bottom half of the screen and will respond to many gestures, such as flinging, dragging, scrolling, tapping, and long presses. Distinct, exaggerated motions will be detected more easily than ambiguous motions.
- The application's gesture-detection is context-aware. This means that the same gestures on different screens will trigger different actions in the application. For instance, landmarks can be added on the navigation screen, however, on the building selection screen or navigation history screen the gesture detection will allow for selection of items in the menus. This may be superseded by voice commands in the future.

5.3 Navatar-RouteHistory

- Before using this prototype be sure to give it file access in android settings, if this is not done the application will crash when pressing "Previous Navigations" button for the first time. Pressing the back button or restarting the app after the crash should prompt the user for permissions, but it is best to give it file permissions in android settings prior to opening.
- To have previous navigations appear in the Previous Navigations page you must start a previous navigation in Scrugham building. This is currently the only functional building in Navatar. The navigation does not need to be completed to appear in a user's history only started.

6 Prototype Evaluation

6.1 Feedback Session 1 - Eelke Folmer

The team met with Eelke Folmer, the advisor for this project, to demo the prototypes and receive feedback. The team explained to Folmer the difficulties encountered including getting the existing codebase to compile properly and understanding the multiple sections of undocumented code. Each member had their respective prototype on their personal android device and demoed the features to Folmer.

Connor presented his auto-locate feature to Folmer, demonstrating the ability for a user to press an auto-detect button on the initial application screen. Pressing the button triggers a GPS location request. The request was unsuccessful the first time because the meeting took place inside a building, which interfered significantly with the device's GPS signal. After placing the phone next to the window a location was successfully received, and map and building were auto selected. Folmer suggested that the GPS request should be made automatically on application launch instead of having the user press the button to initiate the request. Folmer also noted that detecting the building for a navigation should be done as a user enters a building not while inside, and that the team should investigate using the last known location of the user if a current location cannot be detected.

Liam presented his navigation history and navigation repeat prototype; demonstrating a new button on the application home screen, which takes the user to a new page containing past navigations they had made. From this page the user can select an old navigation and immediately start the receiving navigation directions. Liam described to Folmer how the data was simply stored on the user's local device storage. Folmer suggested an additional feature related to this, where upon completing a route the user is given the ability to reverse the navigation they just made. Folmer also noted the potential benefit of storing route history along with the navigations success or failure. Explaining that this data could be extremely useful for analytical purposes and future research.

Finally Matthew presented his integration of gesture based inputs into Navatar's UI, explaining that these gestures could be used to trigger a variety of different actions inside of Navatar. Matthew stated that these gesture based inputs could help improve the applications user accessibility and overall user experience. Folmer liked the idea of using gestures and avoiding text inputs which are hard for those with visual impairments to navigate. Folmer also noted that some gestures could not be used, such as shaking the phone to trigger some action which would interfere with Navatar's core path planning and stride detection algorithms. He also noted that some gestures are already utilized by accessibility software and which we must avoid interfering with.

Overall, Folmer liked the teams prototypes and suggested the continued development of smaller features and enhancements to Navatar similar to the prototypes the team had presented. The team brought up the potential need for refactoring parts of the existing code base to accommodate features in added in future development. Folmer also suggested fixing any pre-existing bugs we find during our development to make the application more stable.

6.2 Feedback Session 2 - Development Team

The members of Team 6 held a meeting to evaluate each other's prototyped features and exchange feedback. The meeting was extremely positive and allowed the team to discuss potential changes to improve the design and functionality of the features.

The location history feature was originally designed for users to repeat routes, however, the team discussed that it could be used as the foundation for multiple other features such as reversing user routes or collecting statistics, as Folmer suggested. When users enter a building the first time, they can use the application to navigate to a specific room, and they will often use it again to exit the building. In this scenario, reversing a previous route will allow users to begin the exit navigation with much fewer inputs than currently possible. Collecting statistics on previously taken or failed routes would allow for multiple enhancements to the application including pruning impossible routes from the room selection screens and promoting frequently used locations such as building entrances and exits. Overall, the team agreed this feature is very useful and holds great potential for future features.

Geo-fencing is a feature the team agreed was useful for reducing the required user input, but there was extensive debate regarding the most effective way to implement it. In the feature's current state, the user's GPS is enabled upon pressing a button and locations are received until a reading is accurate enough to locate the user, or five readings are obtained. The accuracy threshold and time limit can both be easily adjusted, but the team discussed how vital it is that their values are optimized for many different devices and scenarios. If an accurate location is received by the app, the campus and building are automatically selected for the user. The team debated multiple use cases including the user starting navigation from outside of a building where GPS would be accurate, and the user starting inside a building where GPS reception is horrible. The team decided more testing would have to be done to compare the various accuracies of using GPS, WiFi, and cellular towers for locating the user, and develop the feature to use one or a combination. It was also decided that the geo-fences for campuses and buildings would be circular and thus each stored as a coordinate and a radius. Storing locations with this method, as opposed to using multiple points for polygons, will allow for map location information to be easily generated and input by open source contributors.

Adding gesture based input functionality to application was developed as a stepping stone for future functionality such as the user declaring custom landmarks. The team agreed its implementation was executed very well as its modular design allows for using the feature anywhere in the application. After experimenting with the various gestures, the team confirmed a long press action on the navigation screen would be ideal for the user indicating they want to add a custom landmark. Other gestures prompted ideas for new features such as swiping back on the navigation screen to trigger repeating the current navigation direction, however, the team also discussed investigating if this is already a functionality offered by stock accessibility programs on Android. The team was very happy with the implementation of gestures and is excited to improve the user experience with their use.

7 Future Design Changes

It became evident after designing and prototyping desired features that some ideas are better in theory than in implementation. One such example was the gesture-detection as a form of interaction with the application. Gestures that are context-aware lead to the same application interactions eliciting separate and highly distinct behaviors that may lead to unexpected results for the user. The context of the application may be difficult to discern for a blind user. In this sense, gesture-detection should be superseded and brought to obsolescence by voice commands. We plan to use Google voice to implement the actual voice recognition and voice commands. Simply speaking to the application through a microphone on a headset or set of ear-buds will allow for much more fine-grained control and potentially limitless interactions with the app that gestures simply would not allow for. There are only a limited number of gestures, and the overlap would make the usability very poor. Voice commands are the chosen options here.

After designing, prototyping, and testing the GPS features we found out that reliably detecting a user's location indoors was a bit difficult. If this feature is still necessary in future versions, WiFi or cellular towers may be used to pinpoint a location. In reality, it was determined that the only real use-case for auto-location would be detecting when a user enters a building. This is done using Geo-Fencing currently, and can be improved by using more sources to pinpoint location as stated above. As for navigation history, it is a nearly complete and functional feature. The only change necessary for this feature is that the users will need to be able to navigate the selections with easily spoken identifiers using voice commands. As stated in the first paragraph, this is a desired future change that will be developed sometime in the foreseeable future.

Overall there is much to work on and many features to develop, and it is also a high-priority to refactor the codebase and generally improve the quality of the source code.

8 Team Contributions

8.1 Matthew Berger

- Created LaTeX document
- Developed gesture detection prototype
- Documented changes needed to software design

8.2 Connor Parkinson

- Updated the abstract and the introduction
- Documented the prototype objectives
- Developed Geo-Fencing prototype

8.3 Liam Gomez

- Completed descriptions of the prototyped functionality
- Developed location history prototype
- Compiled the prototype branches and apks for submission

8.4 All Team Members

- Documented stakeholder feedback sessions