BluEye

Thomas Kelly, EE, Krista Lohr, CSE, Stephen Fialli, EE, and Divya Reddy, CSE

Abstract—**BLuEye is a navigation system that will guide the blind and visually impaired in unfamiliar indoor and outdoor environments. The system utilizes a mobile application that communicates with Bluetooth Low Energy beacons to establish user location and provide voice instructions for guidance to a specified destination. The application uses RSSI values from the beacons to determine the distance between the blind user node and predefined reference nodes. These distances are passed to a localization algorithm to provide the location of the user. The user location and desired destination are then passed to a shortest path algorithm. The path will be translated into voice instructions to guide the user to their destination.**

I. INTRODUCTION

Having the ability to navigate from place to place is an essential part of our daily lives. However, navigating in an unfamiliar environment poses a difficult challenge for most individuals. Navigating in an environment while being visually impaired poses a much greater challenge. Currently, there are limited number of systems that help navigate the visually impaired both indoor and outdoor environments, which limit their access to certain environments without the help of others.

It is reported by Lighthouse International that about 285 million people in the world are visually impaired, of which 39 million are blind and 246 million are moderate to severely visually impaired. It is also predicted that the numbers will rise up to 75 million blind and 200 million visually impaired by the year 2020 [1].

Several solutions have been proposed in the recent years to help the blind navigate. For example, PEREPT uses RFID tags that will transmit signals to a Smartphone that will show the current location of the user [2]. Drawbacks to the system are that it can only be used in short-range communication and indoor environments.

There are also a few non-technical solutions that can help the bind navigate, like human aids or guide dogs. These solutions definitely have some downsides for the users. For example, having a human aid eliminates the person's independence. Not only that, but the cost of having a human aid by your side every day is expensive. Having a guide dog is a major inconvenience for the user and prevents the users from going to locations that do not allow animals.

SFO, LightHouse, and an Australian based company called Indoors recently developed a system for the blind. It allows the user to navigate throughout the terminals of the San Francisco Airport. The system required beacons to transmit current location signals to the user's mobile device to give proper directions to the user [3]. BluEye offers similar services, but with some changes.

BluEye allows users to navigate both indoor and outdoor environments, providing a continuous system. Having a continuous system for both environments is necessary,because we want to create a system that helps the user navigate to a building as well as find rooms/facilities within the building.

Since the increase of mobile devices in the recent years, human-technology interactions have changed significantly. With the rapid development of touch screen based technology such as the iPhone and tablets, more and more people are dependent on these devices. The goal of our research is to help the visually impaired population navigate both indoor and outdoor environments through the use of a mobile application that receives signals from RFID based beacons.

BluEye's system uses both short and long-range Bluetooth Low Energy (BLE) beacons. An iPhone application will communicate with the beacons to receive RF signals that will later be translated into the current location of the user. The application uses a visionless user interface that will help the blind operate the application.

Table 1: System Requirements

User and Facilities Requirements	System Requirements
Easy to access and use	Voice instructions every 15 seconds for outdoor Ex: turn right, turn left, open door
Inexpensive	Voice instructions every 10 seconds for indoor
Practical for facilities to deploy	Complete beacon coverage of environment
Compatible to Apple products	Provide location that is accurate to 3 feet
	Beacon battery life of at least 3 months

The system will use Estimote beacons for the outdoor deployment and Gimbal Series 10 beacons for the indoor deployment. The beacons will be deployed throughout the Engineering quad and the first floor of Marcus. When the user enters an environment covered by our system, he will request his destination on his Smartphone device using the Smartphone accessibility features. Through the Smartphone, BLuEye will provide audible instructions to the user until he has reached his destination.

II. Design

A. Overview

BLuEye will provide guidance to the user by implementing a network of wireless tags in communication with an iOS mobile application on the user's device. At this stage in our design we have focused mainly on the indoor implementation. BLE tags were chosen for their low power consumption and their compatibility with new generation mobile devices. There are currently 20 Gimbal Proximity Series 10 beacons deployed on the first floor of Marcus Hall, which will be used for navigation through the main hallway of the first floor. These beacons are a Qualcomm product chosen for their low cost ($5 each) and battery life (up to 1 year)[4]. The tags transmit at a frequency of 2.4 GHz, a power of 0 dBm, and on an interval of 645ms [5][6].

The mobile application used in BLuEye will be on the iOS platform. The user interface will be designed to be compatible with Apple's accessibility application, Voiceover. The visionless user interface will consist of a list of destinations in the system environment. When in the system environment, the user will choose a destination and the system will provide clear voice instructions to guide the user to that destination.

BLuEye is divided into indoor and outdoor subsystems because the separate environments will require different methods of localization, navigation, and RSSI characterization. Each subsystem has its own block diagram, which are divided into real time and offline components. Real-time refers to the parts of the system that depend on various inputs at the current time. Offline refers to the parts of the system that are predefined and determined through data collection. The offline aspect of the system includes a digital space, which will be used to map user and reference node locations. The digital space for the indoor system is a set of coordinates that correspond to locations in Marcus Hall. The digital space for the outdoor system will be obtained in real time using GPS (Google Maps). GPS will be used in outdoor localization but beacons are still necessary due to inadequacy of GPS in certain areas of the quad (near the buildings). RSSI characterization is also determined offline through experimental data and will provide the values necessary to convert the received signal from the beacons into estimated distances in feet.

The real time portion of the system begins with communication between the beacons and the mobile device. The RSSI values corresponding to the signals transmitted by the beacons will be converted to distances and used in a localization algorithm that will provide the user's current location node. This node will be passed to a program that computes the shortest path to a destination defined using the visionless user interface. Once the shortest path is computed, instructions for navigating to the destination will be generated and delivered to the user.

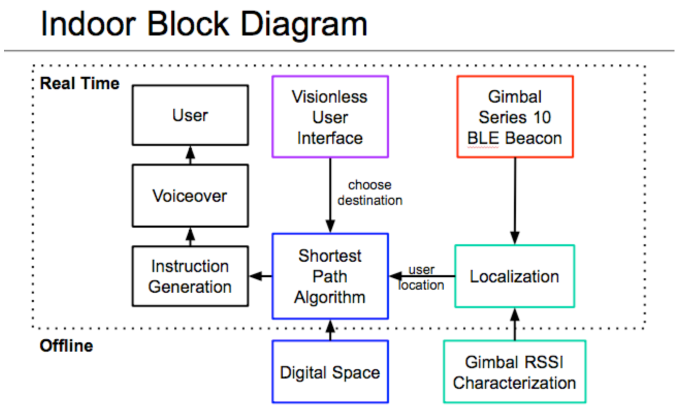


Fig. 1. Indoor Block Diagram

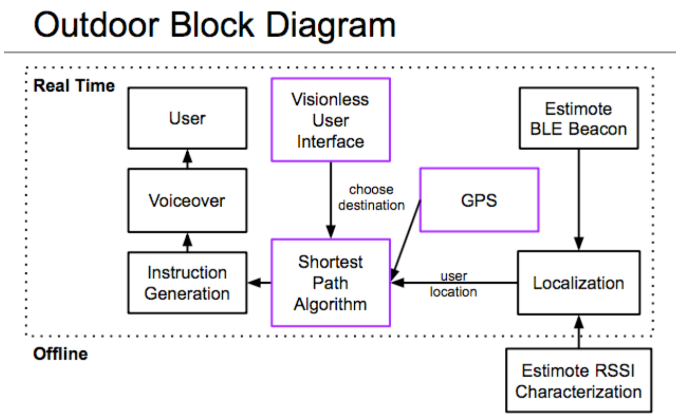
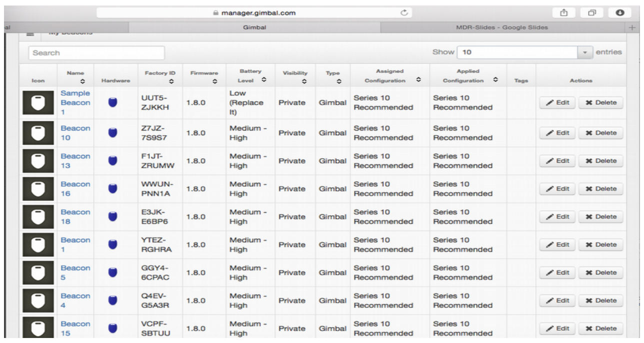


Fig. 2. Outdoor Block Diagram

B. Communication Between Beacons and Mobile Device:

One of the first steps that needed to be completed to start work on our indoor system was connecting the Gimbal Bluetooth beacons to our phone application. This allowed us to be able to read values from the beacons and set what value the beacon transmitted as its ID value. To start this process, each of our beacons was registered with Gimbal to give each beacon its own unique identifier [8]. Using the beacon manager on the Gimbal site, we registered each beacon as Beacon 1 – Beacon 20 to easily identify them.



Icon	Name	Hardware	Factory ID	Firmware	Battery Level	Visibility	Type	Assigned Configuration	Applied Configuration	Tags	Actions
	Sample Beacon 1		UUT5-ZJKKH	1.8.0	Low (Replace It)	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 10		Z7JZ-7S9S7	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 13		F1JT-ZRUMW	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 16		WWUN-PNN1A	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 18		E3JK-E6BP6	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 1		YTEZ-RGHRA	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 5		GGY4-6CPAC	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 4		Q4EV-G5A3R	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete
	Beacon 15		VCPF-SBTUU	1.8.0	Medium - High	Private	Gimbal	Series 10 Recommended	Series 10 Recommended		Edit Delete

Figure 3. Gimbal Beacon Manager

After this, we wrote objective-c code to make our application Gimbal enabled. There is a Software Development Kit for the Gimbal beacons that was used to make this possible [9] and also many guides on the Gimbal website [4]. We had to enable Bluetooth in the application, include Bluetooth frameworks as well as frameworks made by Gimbal, and add objective-C code to start the Gimbal service. Through the Gimbal website, you enter the name of the application you are making in XCode and it gives you an ID and secret that you use in your code to make your application work with the beacons you register. If beacons are active that are not registered, they simply do not show up in your application. This is for security purposes. The application ID as well as the secret is a 32 byte hexadecimal number and are linked directly to the beacons that you add to your account. Without these, the beacons are unusable, even without using Gimbal frameworks.

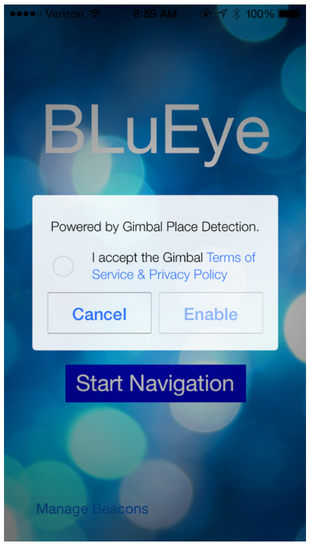
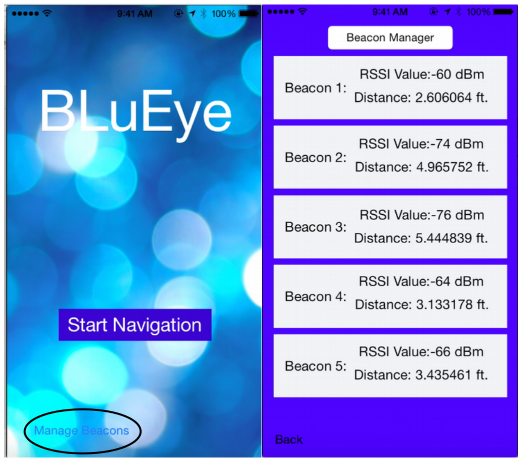

BLuEye
Powered by Gimbal Place Detection.
I accept the Gimbal Terms of Service & Privacy Policy
Cancel
Enable
Start Navigation
Manage Beacons

Fig. 4. Application that is Gimbal Enabled

Now that the application we have created is Gimbal enabled, you can use methods from the software development kits that are triggered each time the Bluetooth antenna on the phone receives a signal from the beacons. The method that was used for our application was receivedSighting, which had parameters of a NSString that was a beacon ID, a NSNumber that was received signal strength (RSSI value), and also the time that the sighting happened. This method acts as a callback, and is called every time a beacon's signal is received. Once this was working, when you ran the application on the phone, the terminal within Xcode showed the values being received from each beacon.

The next task was to make managing our beacons a part of our user interface to be able to read the values on the actual phone without being plugged into XCode. This was done by using UILabel's in Objective-C and extracting values from the receivedSighting method. This was done using the beacon ID's and if statements. If a certain beacon received a sighting, then the corresponding UILabel would be updated to show the beacon values changing in real time. These statements were written inside receivedSighting method so that at each callback the labels are updated. The RSSI characterization algorithms were also added to this, so code was written to convert the RSSI value into a value of feet to visualize the accuracy of our characterization.



Beacon 1: RSSI Value:-60 dBm Distance: 2.606064 ft.
Beacon 2: RSSI Value:-74 dBm Distance: 4.965752 ft.
Beacon 3: RSSI Value:-76 dBm Distance: 5.444839 ft.
Beacon 4: RSSI Value:-64 dBm Distance: 3.133178 ft.
Beacon 5: RSSI Value:-66 dBm Distance: 3.435461 ft.

Fig. 5. Showing Beacon Sightings in the User Interface

C. *Indoor Beacon Deployment*

We determined the best way to deploy our beacons indoors to receive the best signals and have constant coverage. We deployed our beacons starting at the doors entering the hallway from the main entrance of Marcus Hall until the door that enters Marston Hall on the first floor. We placed a beacon at each possible destination that the user of our system would request, and this gave us 100% coverage throughout the hallway showing a minimum of four beacons at all times. One additional beacon was needed at the corner to smooth the transition of turning the corner, because the signals around the corner were blocked. This is much more coverage than what will be needed to complete our project, so addition tests will be made to minimize beacons to minimize cost. At first, we tried deploying the beacons on the ceiling, but we found that the ceiling tiles blocked the signals almost completely if you were not very close to the beacons. In the end, we placed the beacons slightly above each doorway, or on the protruding walls next to the doors so the walls did not block the signals. The beacons were placed up high to reduce disturbances and reflections from by passers, and placed at the same height above the ground for consistency. To place them up on the wall, we uses Velcro on each destination so we can easily put the beacons up for testing and remove them when done. The orientation of the beacon on the wall did not matter in terms of signal strength, but we deploy each beacon facing down for consistency.

D. *RSSI Characterization*

This block of the BLuEye system involves converting RSSI values into estimated distances. These distances are used in the localization algorithm. After establishing communication between the Gimbal beacons and the user device (iPhone 5s), we were able to display RSSI values and corresponding beacons. We used this information during indoor deployment.

Once the beacons were deployed, we collected RSSI data at several locations. This data is displayed in Fig. 6 and Fig. 7.

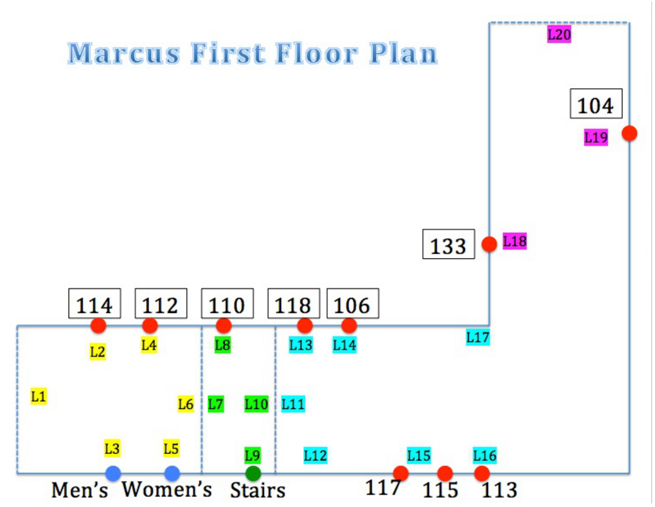
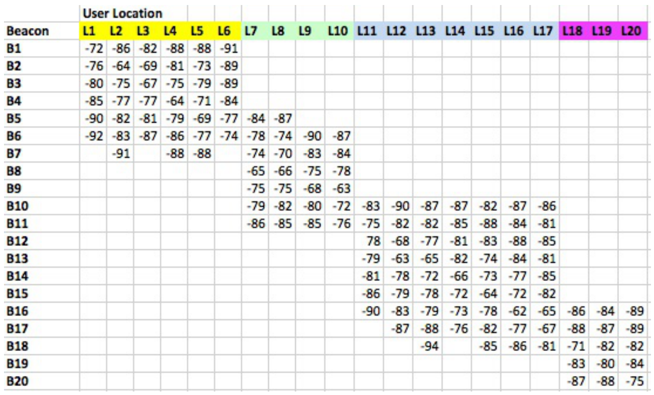


Fig. 6. Marcus First Floor Plan. This map shows the locations at which RSSI data in Fig. 4. was collected.



	User Location																			
Beacon	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15	L16	L17	L18	L19	L20
B1	-72	-86	-82	-88	-88	-91														
B2	-76	-64	-69	-81	-73	-89														
B3	-80	-75	-67	-75	-79	-89														
B4	-85	-77	-77	-64	-71	-84														
B5	-90	-82	-81	-79	-69	-77	-84	-87												
B6	-92	-83	-87	-86	-77	-74	-78	-74	-90	-87										
B7		-91		-88	-88		-74	-70	-83	-84										
B8							-65	-66	-75	-78										
B9							-75	-75	-68	-63										
B10							-79	-82	-80	-72	-83	-90	-87	-87	-82	-87	-86			
B11							-86	-85	-85	-76	-75	-82	-82	-85	-88	-84	-81			
B12											78	-68	-77	-81	-83	-88	-85			
B13											-79	-63	-65	-82	-74	-84	-81			
B14											-81	-78	-72	-66	-73	-77	-85			
B15											-86	-79	-78	-72	-64	-72	-82			
B16											-90	-83	-79	-73	-78	-62	-65	-86	-84	-89
B17												-87	-88	-76	-82	-77	-67	-88	-87	-89
B18													-94		-85	-86	-81	-71	-82	-82
B19																		-83	-80	-84
B20																		-87	-88	-75

Fig. 7. RSSI data. L1-L20 denote the locations at which the data was taken.

The RSSI values varied within a range of about 6 dBm at each location, so it was necessary to take the average values. The model we are using is exponential, so small variations in RSSI can cause a large variation in the distance estimation. For example a difference of 5 dBm in the RSSI values can cause a difference of 90 ft in the distance estimation. Therefore we will need to develop methods to stabilize the RSSI values we use in our calculations. Fig 4. shows that the device is in communication with at least four beacons at all of the locations. At this stage we are using the general path loss model in order to convert RSSI values into estimated distances. This model is shown below.[7]

$$\mathrm{RSSI[dBm]} = -n\log_{10}(\mathrm{d})+\mathrm{A[dBm]} \quad (1)$$

RSSI[dBm] = Received Signal Strength Indicator from beacon
n= path loss exponent
d = estimated distance (meters)
A[dBm] = RSSI value obtained when tag is 1 meter from device

We can manipulate this equation to solve for the unknown value d.

$$\mathrm{d} = 10^{(\mathrm{A[dBm]}-\mathrm{RSSI[dBm]})/n} \quad (2)$$

Using this model, we can characterize the conversion between the RSSI value and distance by determining values for A[dBm] and *n*. A[dBm] was easily obtained by recording the RSSI value while the device was 1 meter from a beacon. We are using the value -65. The value for *n* is the most important aspect of the RSSI characterization. This value is known as the path loss exponent and is highly dependent on the environment. In free space, the value for *n* is 2. The value for *n* in a shadowed urban area can be in the range of 2.7 to 5. Inside a building with the tag in line of sight, *n* can be in the range of 1.6 to 1.8. With obstruction, *n* can be from 4 to 6 [8]. We collected data in an area that we hoped to be close to free space in order to test the model. The results after applying an exponential regression curve to the data are shown in Fig. 5.

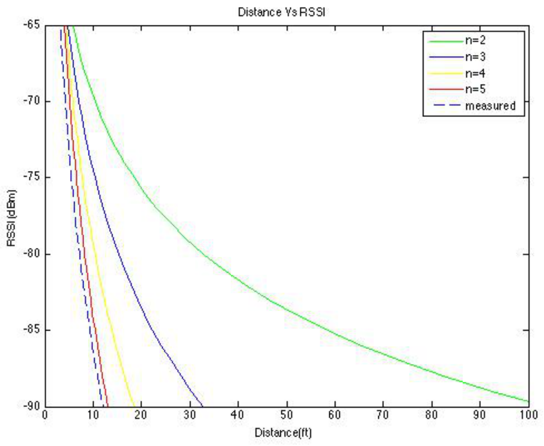


Fig. 8. Distance vs RSSI. The dashed line represents measured data after applying exponential regression.

As Fig. 5. shows, our data seems to have a similar curve as the model when *n*=5. We expect that the environment we tested in is more like a shadowed urban area than free space.

It is clear that our system will not be accurate if we use a single value for *n*. The system environment is full of walls and obstructions that can reflect and attenuate signals. We investigated a method to dynamically calibrate the system so that the value for *n* could be periodically calculated for the current state of the environment. Unfortunately, the adaptive calibration method in [8] requires that the beacons be able to communicate and evaluate RSSI values between each other. Our beacons do not have that capability, so we have moved on to developing a method for static calibration of the system. In order to improve the accuracy of our RSSI to distance conversion, we must section the system environment into several spaces. Each of these spaces will have a corresponding value of *n* that can be calculated during calibration. Each space will have multiple calibration points to ensure that the calculation is not specific to one point in the space. Further measurements and tests will be completed for CDR in order to achieve the best distance estimates.

E. Localization

This block is responsible for providing the current location of the user. The localization algorithm will take the estimated distances from at least three beacons as inputs. The method for localization we are using requires the user's distance from three reference nodes with known coordinates because it utilizes the geometry of triangles [7]. Each beacon location represents a vertex of a triangle, while each estimated distance represents a side of the triangle. This is called trilateration. The method from [7] was confirmed in MATLAB. The coordinates of each beacon were stored in the program using a corner of the hallway as the origin. The MATLAB program was designed to take user input values for beacon identification and the corresponding distance between the user and those beacons. With user input distances to beacon A, B, and C, equations (3.1), (3.2), (3.3), and (3.4) were used to determine the X and Y coordinates of the user.

$$v_a = [(d_b^2-d_c^2)-(x_b^2-x_c^2)-(y_b^2-y_c^2)]/2 \qquad (3.1)$$

$$v_b = [(d_b^2-da\hat{}2)-(x_b^2-xa\hat{}2)-(y_b^2-y_c^2)]/2 \qquad (3.2)$$

$$y = [v_b(x_c-x_b)- v_a (x_a-x_b)]/[(y_a-y_b)(x_c-x_b)-(y_c-y_b)(x_a-x_b) \quad (3.3)$$

$$x = [v_a-y(y_c-y_b)]/[x_c-x_b] \qquad (3.4)$$

Where (x_i,y_i) are the coordinates of the beacon and d_i is the distance from the user to the beacon in feet. Completed indoor localization is a deliverable for CDR. In order to achieve desired results this method of localization must be expanded to three dimensions to account for the height at which the user is holding their device. We also must account for the fact that the distances from the user to the beacons will be purely estimation due to inaccuracy of the RSSI to distance conversion. Our current plan is to use a threshold of about -85 dBm to decide what beacons to consider in each trilateration calculation. We will attempt to use multiple iterations of the algorithm and use the average as the result. This sub block will require a significant amount of testing before CDR. Ultimately this block will need to pass the current node of the user to the shortest path algorithm.

F. Visionless User Interface

The main portion of the project, besides the beacons, is the mobile application. An application is nothing without a user interface. We have decided to develop our mobile application on the iPhone. We are using an Apple Developer account and XCode to create our application and run it on our phones. Apple uses a language called Objective C. Although Objective C had never been taught in any of our courses, it is similar to the C language, which was taught in several of our classes. It follows the same basic format, making it fairly easy to learn. In order to create a user interface, tutorials provided by Apple were used and applied to previous knowledge to make a user interface that is appropriate to our application.

The user interface currently has three main menu page. The first is the title page. This page has a button to start the navigation and another button for debugging purposes that will manage the beacons. The other two menu will allow the user to select an outdoor location (a building in the engineering quad) and an indoor location (a room number or facility within or focus building, Marcus).

The hardest part will be making the application compatible with the accessibility feature, Voiceover. Without accessibility, the application is useless to our target user, the visually impaired. Further work is necessary to make the user interface compatible with Voiceover.

G. Outdoor Shortest Path Algorithm

The shortest path algorithms take input of the user's location and the user's destination to determine a path that will bring the user to their destination the fastest. In the outdoor environment, the user's location is determined by GPS or by a localization algorithm using the Estimote beacon signals. The GPS works on the pathways of the engineering quad, but is very inaccurate near the entrances of the four buildings. The localization will use information from the beacons when the user is close to the entrance of their destination.

The second part of the navigation algorithm is a map that will be a page of the user interface. When the user chooses an outdoor destination, it will bring him/her to this map. Using the Google maps SDK, the map focuses on the engineering quad and has a drawn path from the user's position to their destination. This path can only be created if the user is within the quad and the path always follows the actual pathways of the quad.

To create the navigation algorithm, several nodes were determined. The nodes were placed in front of the entrances of the buildings, at the intersections of pathways, and at the ends of pathways. To use these nodes in the code, I gathered several coordinates from the engineering quad. These coordinates were used to determine the user's location. They also were used to draw the path from the user's location to each node that in the path.

To test that the algorithm was working correctly, we went out to the engineering quad with the application already downloaded onto my phone. I then stood on all of the pathways and determined if the correct path was given to me after choosing each of the building destinations. It was during this testing that I realized that the GPS is not as accurate when close to the buildings. This is why the beacons are essential to the navigation, not just in the indoor environment, but in the outdoor environment as well.

H. Indoor Shortest Path Algorithm

The main part of the mobile application is the shortest path algorithm. Dijkstra's Algorithm will be implemented to determine the shortest path from the current location of the user to their desired destination. As shown in the block diagram, the inputs to the Shortest Path Algorithm block will be the user's current location, the user's chosen destination and a digital map of the environment. The user's location will be determined via the localization algorithm, and the destination will be a user input. The output of the algorithm will be the distance to destination in feet and the path the user must take to reach their destination.

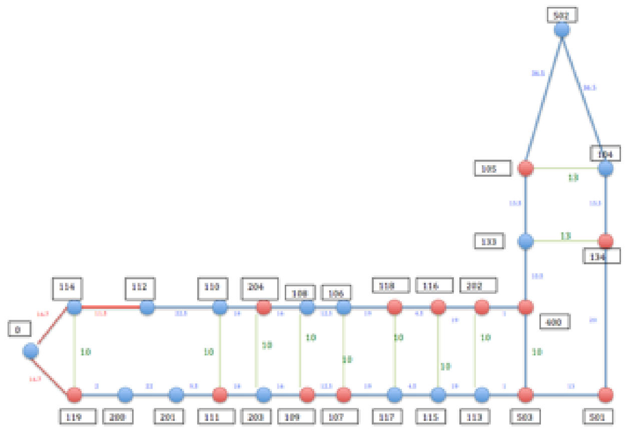
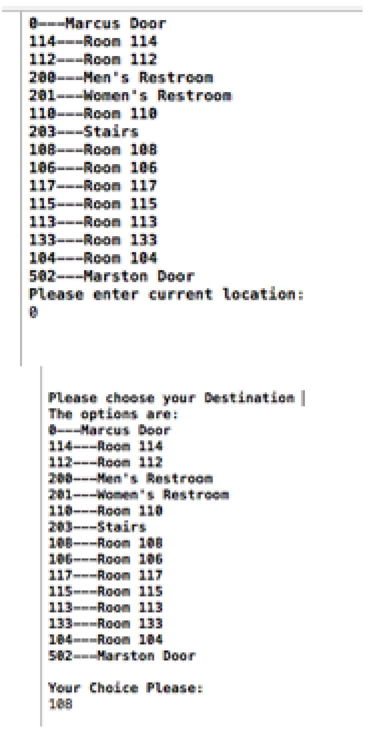


Fig. 9. Dijkstra's map of Marcus First Floor

•0 = Marcus Door
•114 = Room 114
•112 = Room 112
•200 = Men's Restroom
•201 = Women's Restroom
•110 = Room 110
•203 = Stairs
•108 = Room 108
•106 = Room 106
•117 = Room 117
•115 = Room 115
•113 = Room 113
•133 = Room 133
•104 = Room 104
•502 = Marston Door

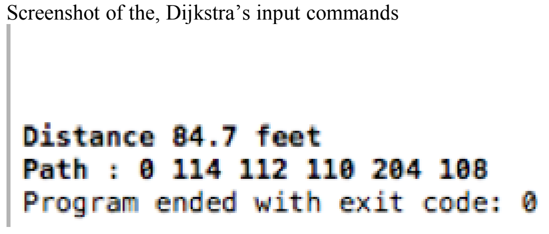
Fig. 10. Codes for Destination Selection

Here, Figure 9 shows the layout the Marcus first Floor as a weighted graph to represent the digital map of the environment. Figure 10 shows the code numbers for each of the locations available.


```
0---Marcus Door
114---Room 114
112---Room 112
200---Men's Restroom
201---Women's Restroom
110---Room 110
203---Stairs
108---Room 108
106---Room 106
117---Room 117
115---Room 115
113---Room 113
133---Room 133
104---Room 104
502---Marston Door
Please enter current location:
0
```

```
Please choose your Destination
The options are:
0---Marcus Door
114---Room 114
112---Room 112
200---Men's Restroom
201---Women's Restroom
110---Room 110
203---Stairs
108---Room 108
106---Room 106
117---Room 117
115---Room 115
113---Room 113
133---Room 133
104---Room 104
502---Marston Door

Your Choice Please:
108
```
Fig. 11. Screenshot of the, Dijkstra's input commands


```
Distance 84.7 feet
Path : 0 114 112 110 204 108
Program ended with exit code: 0
```
Fig. 12. Screenshot of the, Dijkstra's output commands

Here, Figure 11 and 12 shows the input and the output of the algorithm, which will be integrated with the visionless user interface.

To translate the instructions given to the user, the system will use the Voiceover feature, an assistive technology tool within the iPhone. The Voiceover feature is a gesture based screen reader that lets you use an iOS device without seeing the screen. The Voiceover feature will translate the output of the algorithm so that the user will get detailed instructions on how to navigate through the environment. In order to implement the Voiceover feature into our application, we will need to learn all the features with Voiceover and how to adjust it to continuous commands, which is extremely important.

To test the accuracy of the algorithm, there must be an intense testing procedure to ensure that the instructions given are as accurate as possible. We will put a test user in our environment, blindfold them and have them use our application. We will then go with the user and see if the application gives accurate instructions such as open door, pull door, turn right at the right interval of time, etc. We will repeat this experiment multiple times asking the user to start at different locations and choosing different destinations. Based on our results, we will change the instruction commands and the frequency at which the commands are translated.

III. Project Management

Goal	Status
Communication established between beacons and mobile device	Completed
Indoor beacon deployment	Completed
Gimbal RSSI characterization	Completed with accuracy to be improved
Localization Algorithm	Completed, not integrated
Main function of iPhone user interface	Completed
Outdoor navigation algorithm	Completed with medium accuracy
Indoor navigation algorithm	Completed

Table 2: Goals for MDR

Table 2 shows the goals that were set for our MDR. We decided to split of project into two parts: indoor and outdoor. Although these sections will eventually come together, they are independent of each other, so they can be worked on separately. All of the goals mentioned in the table were accomplished for our MDR. Some of the goals, however were met without a high level of accuracy. Some of the future goals of our project will focus on making these measurements more accurate, which will make the navigation instructions more precise.

With half of the year done, there are still several part of the project that need to be completed. Although the indoor portion of the project is almost complete, there are still a few pieces left. The localization needs to be completed. This entails finishing the algorithm that will give a coordinate of the user's location to the algorithm that determines the navigation path. The navigation algorithm, which is already complete, needs to be added to the phone application. Voice instructions going along with the shortest path need to be generated as well. For the outdoor portion of the project, deployment and RSSI characterization needs to be done, as it was for the indoor environment. Same as indoor, the localization and voice instructions also need to be completed. Finally, the phone application needs to be adjusted for accessibility. Voiceover needs to be compatible with our application. With all of these things finished, the only thing left will be adjusting our measurements for better accuracy.

When dividing the work, we each picked our tasks based on our strengths. Steve and Tom, the electrical engineers, have the most experience with analyzing signals, so they choose tasks that would be working directly with the beacons. The RSSI characterization, beacon deployment, and localization all required analysis of the signals received from the beacons. Divya and Krista, the computer systems engineers, have the most experience with software, so they chose the tasks that required more knowledge of coding. The indoor navigation algorithm and outdoor navigation algorithm required inputs from the localization algorithm but are completely software. The user interface is also a completely software assignment. We all completed our tasks promptly and without any major issues.

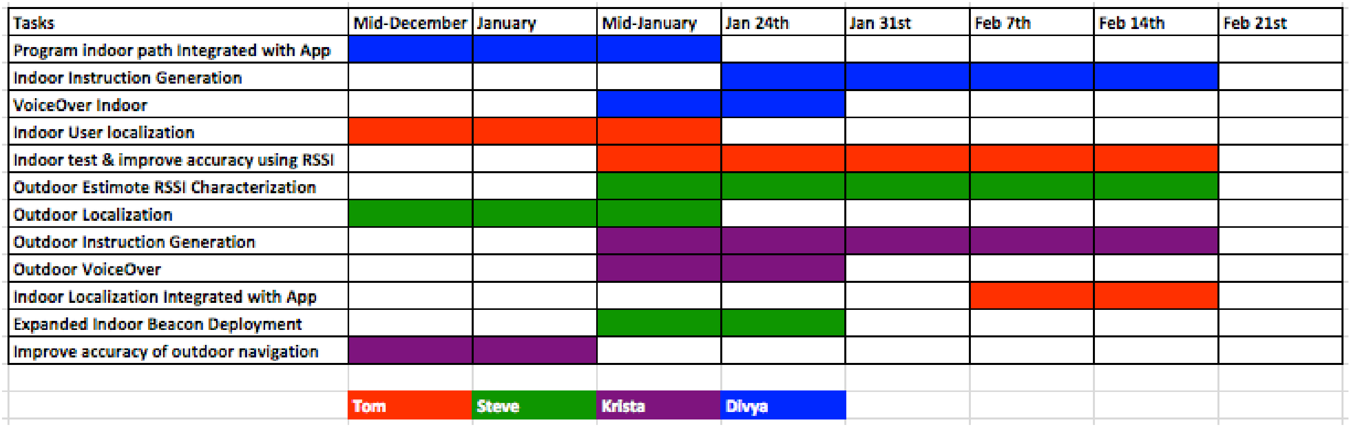
So far, the team is working very well together on our project. Although the goals for MDR were specifically divided among our four team members, we have still managed to work together on parts of the project. When we first started working on our tasks after PDR, we met in Marcus Hall to make measurements. These measurements helped with the RSSI characterization, the beacon deployment, and creating the shortest path algorithm. Besides these measurements, it was important throughout the process to keep in touch with each other. Steve and Tom were both in charge of tasks that worked with the Gimbal beacons, so they met frequently to make sure they were both on the same page with their separate parts of the project. The visionless user interface that Krista was in charge of completing is the piece that ties all of the other parts together. Because of this, communication is needed among all of the team members to successfully integrate the other parts into the application. The team frequently worked on their separate parts in the same room to facilitate communication. We are constantly e-mailing and texting each other to make sure everyone is on track, and providing help when necessary.

The Gantt chart show in Fig. 13 illustrates our plan for the next few months. For CDR, Divya will be working on moving her code on the navigation algorithm into the application, generating voice instructions for indoor navigation, and ensuring accessibility of the application for the indoor portion. Krista will be working on generating voice instructions for outdoor navigation and ensuring accessibility of the application for the outdoor portion. Steve will be working on the RSSI characterization of the Estimote (outdoor) beacons and localization of the outdoor environment. Tom will be working on completing indoor localization and improving the accuracy of the RSSI readings.

IV. Conclusion

In conclusion, we are at the point in the project where the most subsystems are complete and ready to be integrated with one another. We are able to communicate with the BLE beacons within our application, we have characterized the RSSI values to real distances, the navigation algorithms that will guide the user are complete,the indoor beacon deployment is complete, and a user interface is complete. We got to this point by working on individual tasks as well as working together and using each others accomplishments.

Our plan for the future is to combine what we have now to create the whole system. We have the majority of the separate parts of the project working on their own, but we need to get them to work together and be more accurate to meet our system specifications. This will be done with a lot of calibrating and testing to get better RSSI values for the localization, to create real coordinates instead of using test inputs. Another large challenge we will face will be making our system completely visionless. This will be done by working with Voiceover. With this, we will discover how often we will need to give instructions and what instructions will be most useful in every situation.



Tasks	Mid-December	January	Mid-January	Jan 24th	Jan 31st	Feb 7th	Feb 14th	Feb 21st
Program indoor path Integrated with App	🟦	🟦	🟦					
Indoor Instruction Generation				🟦	🟦	🟦	🟦	
VoiceOver Indoor			🟦	🟦				
Indoor User localization	🟥	🟥	🟥					
Indoor test & improve accuracy using RSSI			🟥	🟥	🟥	🟥	🟥	
Outdoor Estimote RSSI Characterization			🟩	🟩	🟩	🟩	🟩	
Outdoor Localization	🟩	🟩	🟩					
Outdoor Instruction Generation			🟪	🟪	🟪	🟪	🟪	
Outdoor VoiceOver			🟪	🟪				
Indoor Localization Integrated with App						🟥	🟥	
Expanded Indoor Beacon Deployment			🟩	🟩				
Improve accuracy of outdoor navigation	🟪	🟪						
	🟥 Tom	🟩 Steve	🟪 Krista	🟦 Divya				

Fig. 13. Gantt Chart

References

[1] "Prevalence of Vision Impairment." Lighthouse International -. Web. 3 Dec. 2014. <http://www.lighthouse.org/research/statistics-on-vision-impairment/prevalence-of-vision-impairment/>.

[2] "Media & Press." 5G Mobile Evolution Lab: PERCEPT. Web. 3 Dec. 2014. <http://percept.ecs.umass.edu/>.

[3] "Indoor Mapping Lets the Blind Navigate Airports." Smithsonian. N.p., n.d. Web. 14 Dec. 2014.

[4] Gimbal Store [Online] Available: https://store.gimbal.com/default.aspx?menuoption=homepage [Accessed Web. 20 Sept. 2014.]

[5] Gimbal Proximity Overview [Online] Available: http://gimbal.com/doc/proximity_overview.html [Accessed Web. 20 Sept. 2014]

[6] L. Frenzel. (2012, November 29). *What's the difference between Bluetooth Low Energy and ANT?* Electronic Design [Online] Available: http://electronicdesign.com/mobile/what-s-difference-between-bluetooth-low-energy-and-ant [Accessed Web. 1 Oct. 2014]

[7] O. Oguejiofor, A. Aniedu, H. Ejiofor, and A. Okolibe, (2013) "Trilateration Based localization Algorithm for Wireless Sensor Network" IJISME. ISSN: 2319-6386. Volume-1, Issue-10. Available: http://www.ijisme.org/attachments/File/v1i10/J04470911013.pdf

[8] J. Kang, D. Kim, and Y. Kim. "RSS Self-calibration Protocol for WSN Localization" Information and Communications University, Daejeon, Republic of Korea. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4147056

[9] Gimbal Beacon Manager [Online] Available: https://manager.gimbal.com/transmitters [Accessed Web. 1 Oct. 2014]

[10] Gimbal SDK's [Online] Available: https://manager.gimbal.com/sdk_downloads [Accessed Web. 15 Oct. 2014]