April 2017

Using iBeacon for Navigation and Proximity Awareness in Smart Buildings

Christopher John Connor
Worcester Polytechnic Institute

David John Goodrich
Worcester Polytechnic Institute

Qusai Ahmed Alhumoud
Worcester Polytechnic Institute

Project Number: KZP - AA5Z

Using iBeacon for Navigation and Proximity Awareness in Smart Buildings

A Major Qualifying Project
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science
in
Electrical and Computer Engineering
By

Qusai Alhumoud

Chris Connor

David Goodrich

Advisor: Professor Kaveh Pahlavan

Co-Advisor: Professor Jahangir Rahman

Abstract

Guests in unfamiliar buildings often do not know where to go to or what is around them. This project sought to alleviate this problem using a combination of Android smartphone and Bluetooth iBeacon technology. Released in 2013 by Apple, the iBeacon specification allows an embedded device to broadcast telemetry to various Bluetooth-enabled devices, such as smart phones. This project used iBeacon devices from Estimote, a company that has seen its iBeacon devices used in a variety of real-world applications, such as enhancing the experience of tours at the Guggenheim museum in New York.

This project relied on received signal strength (RSS) information obtained from iBeacon devices deployed on the third floor of Atwater Kent Labs on the WPI campus. This information was used to model Atwater Kent Labs as a wireless channel with established path loss characteristics for bluetooth signals. These path loss characteristics were incorporated into an Android application, which localized the user with live telemetry from the iBeacons using a least mean squares (LMS) algorithm and provided contextual information to the user about their surroundings. The overall goal was to provide a proof of concept platform for indoor navigation and proximity awareness using iBeacon.

In this paper, we discuss the channel model used for modelling bluetooth signals in Atwater Kent Labs and the development of the LMS localization algorithm used in the application. We also provide a performance evaluation of the application's localization functionality where the real-world localization error is examined and compared to a Cramer-Rao Lower Bound (CRLB) analysis for the iBeacon deployment. We also discuss the development and features of the Android application and provide all the code so that future projects can build off of the platform we have developed. We conclude by discussing the feasibility of using iBeacon for indoor navigation and proximity awareness and make recommendations for future work using iBeacon.

Acknowledgements

This project would not have been possible without the help of a number of individuals who provided us with advice, guidance, and resources.

We would first like to thank our project advisors. Professor Kaveh Pahlavan provided us with a huge amount of support and encouragement. He gave great advice and pushed us to make the project better and better as the year went on. Professor Jahangir Rahman gave us a lot of advice on our paper and pushed us to submit our project to conferences and for the Provost's Award. His confidence in our project is one of the biggest reasons that this project was a Provost Award finalist.

We would also like to thank all the students and teaching assistants who helped us through the project. Without the resources and guidance of the students who work in the CWINS lab, especially Julang Ying, our project would not have been able to touch on so many areas of wireless communications research. Thanks as well to Arkar Min Aung, who was instrumental in helping us develop the java version of the LMS localization algorithm used in this project.

Finally, we would like to thank Adriana Hera and Sabah Razavi, who helped us with a lot of the Matlab code used in this project.

Table of Contents

List of Figures

List of Tables

1. Introduction

This project aimed to implement Bluetooth low energy (BLE) proximity sensing within an enclosed, indoor environment. The team designed an application with a variety of features which enhance the experience of anyone who enters the Atwater Kent building on the WPI campus, such as localization and navigation using Bluetooth signals. as well as the ability to access contextual information based on proximity to certain locations. This makes it easier for users to find and orient themselves within the building, even if they are unfamiliar with it. The application also provided users with helpful information about points of interest based on their physical location within the building, including professors' offices, conference rooms, and labs.

1.1 Motivation

When someone walks into a building and is unfamiliar with the layout, they often do not know where they are meant to go, and so need to look around for signs pointing to their destination. If a guest only knows someone's name, looking for that particular person's office can be especially tricky since it could be any room on any floor and typically there are not signs pointing out how to get to a particular office. In an academic setting, office hours are usually just posted online, so even if a student finds an unfamiliar professor's office, the professor may not even be there if the student did not plan to meet with the professor in advance and the student would have no way of knowing when the professor is coming back. This points to a major problem with many large buildings: guests often have a hard time navigating the building and cannot get useful information about the locations and people within it.

1.2 Technology

Recently developed technology, in the form of iBeacon BLE technology, gave the team a way to address this problem. Since its initial development in 2013 by Apple, iBeacon technology has greatly expanded in availability. This technology allows an embedded device to broadcast telemetry to various Bluetooth-enabled devices, specifically smartphones. For this project, the team used Estimote beacons, as they provided a good balance of competitive pricing and a substantial suite of features including a ranging API, large sets of example code, long battery life, and variable broadcast power that the team was able to set to suit the needs of the project. Estimote beacons have also already been used in a variety of real-world applications, such as for displaying contextual information to guests at the Guggenheim museum in New York. This combination of features and a proven track record in real-world applications make Estimote beacons an ideal choice for this project. A picture of the Estimote Location beacons used in this project can be seen in Figure 1.1.

Figure 1.1 Estimote Location iBeacons

1.3 Project Description

This project provides a process and proof of concept for implementing navigation and contextual information broadcasting within future smart buildings. Given the popularity of smartphones and other mobile devices, the team felt that iBeacon technology can be leveraged in many places across the world to provide users with indoor navigation assistance and contextual information about the points of interest near their physical location. This project took the first step in developing an application to meet these need areas by creating an indoor localization and proximity awareness system for the third floor of the Atwater Kent building on the Worcester Polytechnic Institute campus.

The project had three main objectives that built off one another as the project progressed. These objectives were:

1. Model the third floor of Atwater Kent Labs as a wireless channel.
2. Perform localization of a user in Atwater Kent using path loss characteristics derived from our channel model.
3. Create an Android application to localize a user and provide them with contextual information about their surroundings.

In order to fulfill these objectives, the project had to touch on many different research and design areas. First, the team needed to correctly model the third floor of Atwater Kent Labs as a wireless channel. This involved taking hundreds of samples of received signal strength (RSS) data, extrapolating a path loss model from this data, determining the fade margin experienced by bluetooth signals in the channel due to shadow fading, and determining the broadcast coverage of beacons under different circumstances.

Second, the team created a deployment plan for placing the beacons throughout the third floor of Atwater Kent. This involved mapping out the deployment using the coverage analysis the team performed and comparing the real performance of localization in the deployment to theory using a Cramer-Rao lower bound (CRLB) analysis. The deployment map for the iBeacons on Atwater Kent's third floor can be seen in Figure 1.2

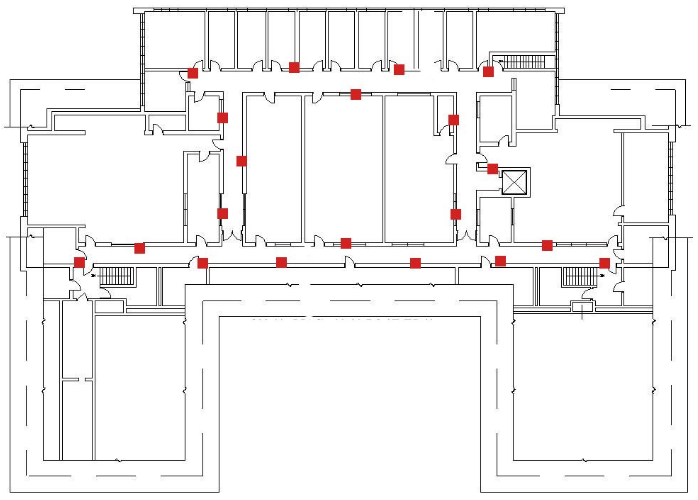
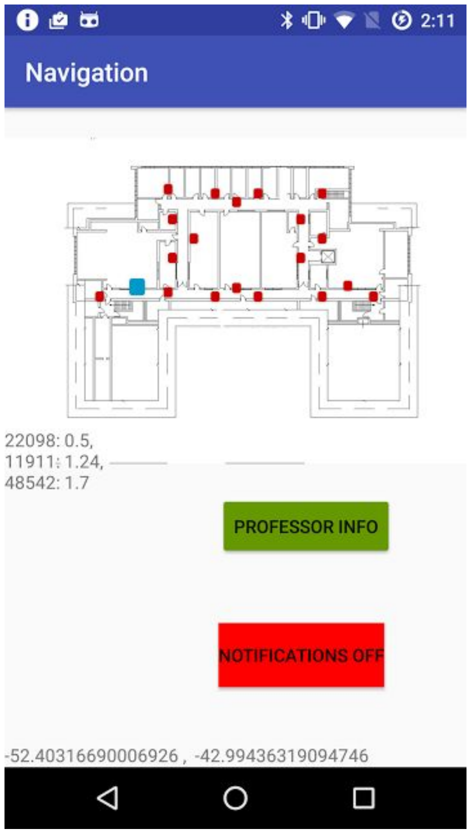


Figure 1.2 iBeacon Deployment in Atwater Kent Labs

Third, the team worked to create a localization scheme that would use the beacons to place a user in their real location on a digital map of Atwater Kent. To do this, the team created a least mean squares (LMS) algorithm in both Matlab and Java that could take distance inputs and reference locations and localize a user on a grid. The team also created a trilateration algorithm in Matlab to use as a comparison point for the LMS algorithm’s performance. Using these two algorithms, the team did an analysis of how effective each was and what degree of error could be expected when using each.

Finally, the team used what it had determined for the channel model, deployment, and localization functionality to build an Android application. To build this application, the team needed to use the channel model and localization algorithm it had built to place a user at a specific location on a map of Atwater Kent. The team also had to build a user-friendly graphical user interface (GUI) for the application, add useful information for the user to look at, and incorporate notification functionality. All together, this application was the culmination of the project because it incorporated everything the team learned and did in the project in order to solve the primary issue of people not being able to navigate effectively indoors. A screenshot of the application’s main screen can be seen in Figure 1.3



22098: 0.5,
11911: 1.24,
48542: 1.7

-52.40316690006926 , -42.99436319094746

Figure 1.3 Application Main Screen with Navigation Map

This type of technology would be able to make it much less frustrating for users to find their way around unfamiliar locations and get useful information that they would normally not have access to. The team hopes this application can be expanded to other buildings across WPI after Atwater Kent, so that it can be useful to all students, faculty, and guests regardless of where they are on campus.

1.4 Report Outline

In the background, the team starts by summarizing previous projects which use iBeacons. The team then researched how we planned to perform the localization by performing path loss modeling, investigating possible algorithms and performing a Cramer-Rao lower bound to find the lowest possible error which would be the theoretical best possible result. To conclude the background, the team goes on to describe the hardware and software which would be required to carry out the project.

For the methodology, the team broke the project into three objectives, first of which was the discussion of their channel modeling process. Second is the localization process through the use of the LMS and trilateration algorithms. Third was the discussion of the smartphone application which would utilize the localization in a form which can be useful and serve users with information. This application was also tested to check how it actually worked to compare with theoretical results.

The results section details what the team actually accomplished with their project and final product. It goes in the same order as the methodology starting with channel modeling, going through localization and then detailing the final application. The performance of the final application was also evaluated based on the tests that were conducted.

Finally, the team concludes the paper with reflections on the design decisions that were made and making recommendations for work which can be done in the future to extend the usefulness of this application.

2. Application Development for Localization

Students and visitors at schools across the country often have appointments and activities in buildings with which they are unfamiliar. They don't usually have a map of the building, they are left with just a room number or room name. If they are looking for a particular person's office it could be even worse as the only clue might be a plaque by the door. Some buildings are even so large it is hard to a person to keep track of their bearings. Schedules and information about people and offices are not always posted in obvious places or easy to find online. All together, this means that there is a need for an indoor geolocation system to help users navigate new buildings and get information on points of interest in those buildings.

The team solved these issues by integrating iBeacon technology with a smartphone application to build an indoor geolocation and information broadcasting system in Atwater Kent. The application provided indoor geolocation in three dimensions so that users could be able to tell where they are in a building and where they need to go to reach their destination. It also provided contextual information on various points of interest in the building depending on the user's location. This was very useful for students who are unfamiliar with Atwater Kent who are looking for a specific office, conference room, or lab. It also gave students access to more detailed information about different professors, including research interests and the classes they teach, that they otherwise would not get by just walking around the building. Finally, with a dynamic information broadcasting feature, users were able to get access to conference room and lab schedules so that they know exactly when these rooms are going to be in use by someone else and when they are free to be used.

Bluetooth (IEEE 802.15.1) is a good platform to use for geolocation and information broadcasting because of its ubiquity in basically every smartphone on the market. Combined with the fact that most people have smartphones or some other mobile device these days, this means that the application has a wide potential user base. The specific type of Bluetooth signal being used in this project is Bluetooth Low Energy (BLE). This wireless specification provides for low power, low range, and a very short connection time. This makes it ideal for high resolution localization and contextual information broadcasting. Some general specifications for BLE can be found in Table 2.1 below [1].

Table 2.1 Bluetooth Low Energy (BLE) Specifications

Topology	Star and broadcast
Modem	GFSK with 1 Mbps at 2 MHz bandwidth
Power	-20 dBm to 10 mW with -70 dBm sensitivity requirement
Channels	40 centered at 2402-2480 MHz
Connection Time	6 ms (100 ms in classic BT)

In order to do localization with BLE, the team had to model the environment in Atwater Kent as a wireless channel. To do this, the team had to closely estimate the path loss, or the amount of power lost when a signal travels between a transmitter and a receiver, of the iBeacon broadcasts to the user smartphones. This is to say, the team must have come up with an algorithm to determine the RSS (received signal strength) that a phone gets from the beacons at various distances from the beacons. Generally, as the phone got further from the beacon, RSS would decrease since the path loss of a signal gets more significant with distance.

The Estimote beacons that the team used, provided the team with a lot of information it need such as RSSI, broadcasting interval, minor and major values and even motion sensing capabilities. Minor and major values could be defined by users, which could be used to distinguish each iBeacon [2]. However, the existing application from Estimote did not allow the team to collect data directly from either android or iPhone devices, which means that the team must have collected data by hand to build a path loss algorithm to use in the project. After this, the team had to develop a specific application to make use of RSS information and this path loss algorithm to determine where a device actually is in Atwater Kent.

2.1 Previous Work on iBeacon Technology

The Center for Wireless Information Network Studies (CWINS) is a compact wireless research laboratory with a rich history of research that have been done with other industrial and academic groups. The research program in wireless information networks at Worcester Polytechnic Institute (WPI) was established in 1985 by Professor Kaveh Pahlavan as the first research program of this sort in the United States. The main focuses of the center are Body Area Network, Indoor Geolocation and Wireless Local Area Network [3]. Recently, the center started focusing on developing projects with iBeacons under the advisement of Professor Kaveh

Pahlavan. Starting in 2016, there have already been two significant projects developed in this area; “Using iBeacon for Newborns Localization in Hospitals” and “Using iBeacon for Intelligent In-Room Presence Detection.”

“Using iBeacon for Newborns Localization in Hospitals” was done by a group of WPI graduate students under Professor Pahlavan. The main idea of the project was to use iBeacons to create an in-room newborn localization system in hospitals to replace systems relying on Radio Frequency Identification (RFID). In order for the group to certify the benefit of using iBeacon over RFID technology for this application, they developed a custom application to receive iBeacon data over BLE because of the difficulty of collecting the needed data by iphone using third party applications. After that, the group had to develop a real-world path loss model for a line-of-sight (LOS) environment to estimate distance using a received signal strength indication (RSSI) analysis. Then, by simulation, the group was able to directly observe the influence of different iBeacon deployment patterns in this hypothetical in-room localization application [4].

“Using iBeacon for Intelligent In-Room Presence Detection” was done by the same WPI graduate students under Professor Pahlavan. The main idea of the project was to create an intelligent in-room presence detection system to record the users in a room by using iBeacon and relying on the fact that the beacon signal is only broadcasted on a certain time interval. The group follows the same procedure for developing an application as was used in the “Using iBeacon for Newborns Localization in Hospitals” project to obtain the necessary data from iBeacon and get RSSI readings directly from iPhone sensors for an LOS situation in a typical indoor office environment. Afterwards, the group tested both a double iBeacon approach for increased coverage, and a single iBeacon approach for lower cost and more convenience [5].

These two projects played an important role in starting this project, as they provided a source for research on iBeacons and channel modelling. Relying on these two projects, the team will have a jumping off point to develop an application to obtain RSSI readings from iBeacons and do localization using smartphones.

2.2 Localization and Navigation research

There are several useful techniques for the localization and tracking of moving objects over networks. These techniques are classified into centralized and decentralized algorithms. The centralized approaches assume the existence of a central processing unit that is responsible for all the processing tasks, such as the least squares localization algorithms. The goal of all localization methods is to minimize the error between where the mobile is and where the mobile should be. The algorithm must be able to follow a preset track or find the least error in the mobile location [6][7][8].

2.2.1 Path Loss Modeling

The first step in localizing a user in an area is to model the path loss experienced by a signal in that area. A path loss algorithm determines how much signal is loss over a given distance in a given channel, which in wireless applications, is the physical space being used in the application. Given a certain received signal strength indication (RSSI) from an iBeacon, the team used a path loss algorithm to determine how far the receiver device is from that beacon. The general formula for a path loss algorithm, based on IEEE models, takes the form [9]:

$$RSSI(d) = P_r = P_t – L_p(d_0) - 10\alpha \log_{10} d/d_0 + \sigma_{SF}$$

Using an algorithm based on this formula should have allowed the team to determine, within a small error range, how far away a user device is from a given beacon.

2.2.2 Least Mean Squares (LMS) Localization Algorithm

After determining how far away a user device is from a beacon, the team used localization techniques to determine exactly where a user is on the third floor of Atwater Kent. In this project, the team used an LMS algorithm for mobile localization. LMS is a centralized approach that assume the existence of a central processing unit is responsible for all the processing tasks. LMS algorithm is used for mobile localization that utilizes a distributed mechanism to process the data, and uses the received signal strength (RSSI) and the signal propagation time (SPT) to estimate mobile locations and increase the accuracy of the localization [10][11].

LMS algorithm should perform well in small network environments such as buildings where the energy and communication bandwidth are not scarce resources. The LMS algorithm has a simple structure and requires low computational resources. Therefore, it was an attractive solution for the proposed applications. More importantly, the algorithm operated in an adaptive manner and had an agile tracking ability which made it particularly useful for tracking a user in real time [10][12].

2.2.3 Trilateration Localization Algorithm

Trilateration is the process of finding the center of the area of intersection of three spheres which are three beacons in this project case. The center point and radius of each of the three beacons must be known. Therefore, trilateration does have practical applications in navigation, including global positioning systems (GPS) [13].

In three-dimensional geometry, trilateration should narrow the possible locations down to no more than two unless the centers lie on a straight line [13].

2.2.4 Cramer-Rao Lower Bound

As the team builds a localization model, the team needed to determine the accuracy of their model by comparing their localization estimates to theoretical estimates. The team did this by comparing their model to estimates obtained by modelling the Cramer-Rao Lower Bound (CRLB) for this application. The CRLB expressed the lower bound on the estimators of a deterministic parameter. CRLB stated that the variance of an unbiased estimator was at least as high as the inverse of the Fisher information. An unbiased estimator at this lower bound was considered fully efficient [14]. The team calculated this lower bound for their deployment of iBeacons in order to see how close their deployment and localization strategy get to the realistic best performance described by CRLB.

2.3 Hardware Platform

For this project, the team used iBeacons from Estimote. These beacons come in a variety of package types including “Location,” “Proximity,” “Stickers,” and “Video.” A breakdown of all of these package types can be seen in Table 2.1 below. The physical appearance of each package type can be seen in Figure 2.1 [2].

Table 2.2 Estimote iBeacon Package Options

Package Type	Location	Proximity	Sticker	Video
Battery life	5 years	2 years	1 year	Endless (USB‑powered)
Range	70 meters	70 meters	7 meters	10 meters
Thickness	24 mm	17 mm	6 mm	14 mm
iBeacon™ or Eddystone™ packets	8 simultaneously	1 at a time	1 at a time	2 simultaneously
Additional packets	connectivity, telemetry, user‑defined	connectivity, telemetry	Connectivity, nearable with telemetry	connectivity, telemetry, user‑defined
Built-in sensors	motion, temperature, ambient light, magnetometer	motion, temperature	motion, temperature	n/a
Additional tech	GPIO, RTC, LED, 1Mb EEPROM	Programmable NFC	n/a	WiFi, HDMI, USB, 1GB eMMC Storage

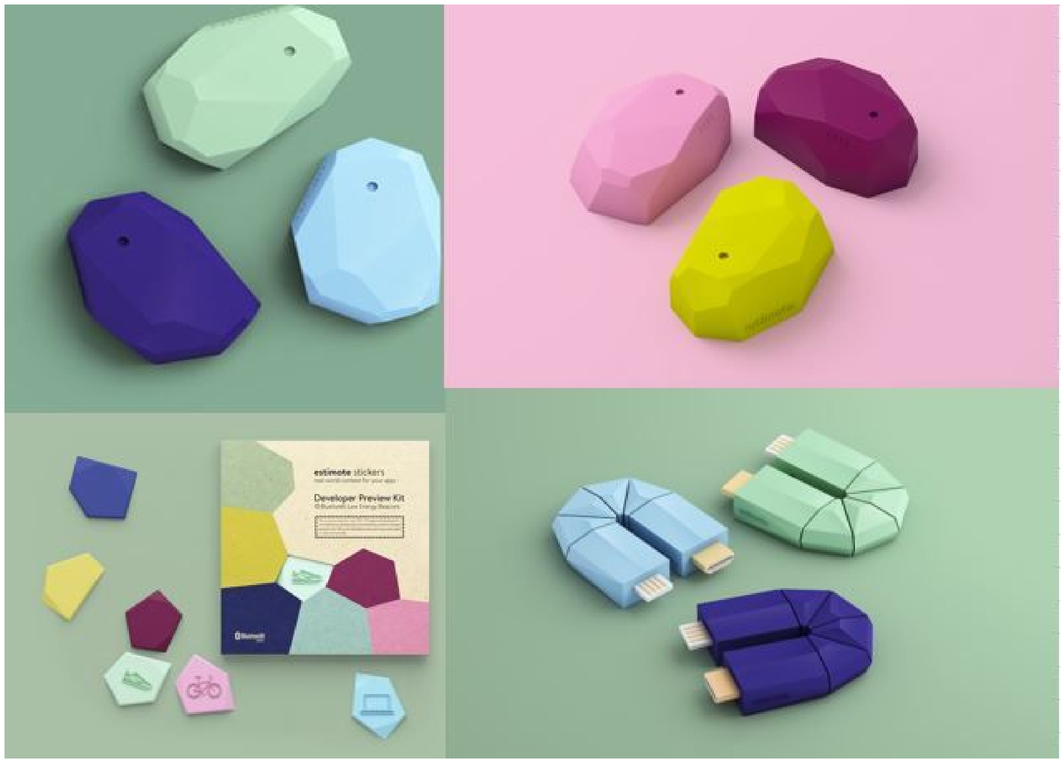


Figure 2.1 iBeacon Packages. Upper Right to Lower Left: Proximity, Location, Video, Sticker

The Location beacons are Estimote’s most robust package type with its highest available broadcasting power, largest range, and longest battery life. The Proximity beacons share many of the same features as the Location beacons, but are somewhat cheaper and lack the additional sensor functionality that the Location beacons have. The Stickers have a much lower broadcast range and limited features, but are much smaller in size. Finally,the Video beacons are usb-powered and have a feature set centered around video content. For this project, the team used with Location beacons, as they have the largest available broadcast range and longest battery life.

Estimote beacons have previously been used in a variety of applications, including in world landmarks such as the Guggenheim museum in New York City where they were used to display contextual information about nearby paintings and sculptures. They were also used in Camp Nou, stadium for FC Barcelona, to create an app which contains information about what is happening with each of the professionals teams of FCB including football, basketball, hockey, handball, and more. The app welcomes thousands of visitors and fans every day, triggers specials promotions depending on the user’s location and allows people to buy tickets for Camp Nou museum tour. Hamad International Airport is using beacons to enable navigation that leads passengers to their gate and checks them in [2].

In addition to the iBeacons, the team used a OnePlus One, an Android phone running a version of Android 6 (Marshmallow), to test the application with. This is the phone that all internal testing was being done on. Further testing details can be found in Section 3.5. The pertinent specifications for this phone can be found in Table 2.2 below [15].

Table 2.3 Oneplus One Test Phone Specifications

Operating System	CyanogenMod 12.1.2 (Based on Android 6.01)
Chipset	Qualcomm MSM8974AC Snapdragon 801
CPU	Quad-core 2.5 GHz Krait 400
GPU	Adreno 330
RAM	3 GB LDDR3
WLAN	Wi-Fi 802.11 a/b/g/n/ac, Wi-Fi Direct, DLNA, hotspot
Bluetooth	v4.1, A2DP
GPS	Yes, with A-GPS, GLONASS
Other Sensors	Accelerometer, gyro, proximity, compass

2.4 Software Platform

The team opted to use Android instead of iOS as the primary development platform for a few reasons. The first reason was that, while Apple phones are certainly very popular, Android has taken a large market share in the US, so the team would not be losing out on too many potential users by going with Android. Secondly, because all of the Android development tools are free and generally open source under the Apache 2.0 license or Creative Commons Attribution 2.5 [16], the team did not have to worry about paying for a development license or operating in a locked-down development environment. Instead, the team could use whatever development tools it chooses. Finally, the team already had had access to several Android phones of various specifications to test the application with.

The team's smartphone application would be interfacing with the beacons using Estimote's development API and by leveraging the Estimote application ecosystem. Estimote has an extensive software ecosystem for running example programs and editing the settings of the beacons. On mobile, Estimote provides an application for running a number of example applications, such as an example ranging app and a simple notification app. A screenshot of the mobile app can be seen in Figure 2.2.

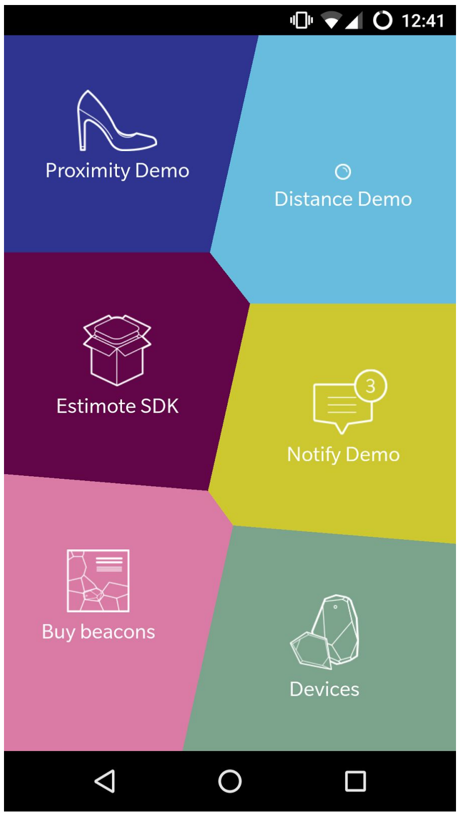
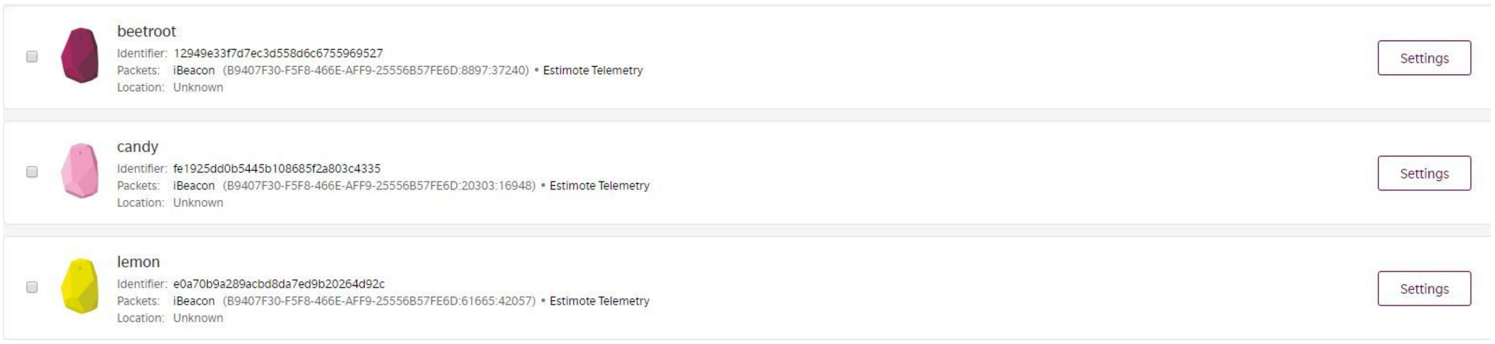


Figure 2.2 Estimote Example Android Application Menu

On the Estimote website, Estimote has a webapp for editing the settings of all the beacons owned by the account. The interface allows a user to select any of the beacons they own, then go in and see each beacon's settings. A screenshot of the beacon selection menu can be seen in Figure 2.3 [2].



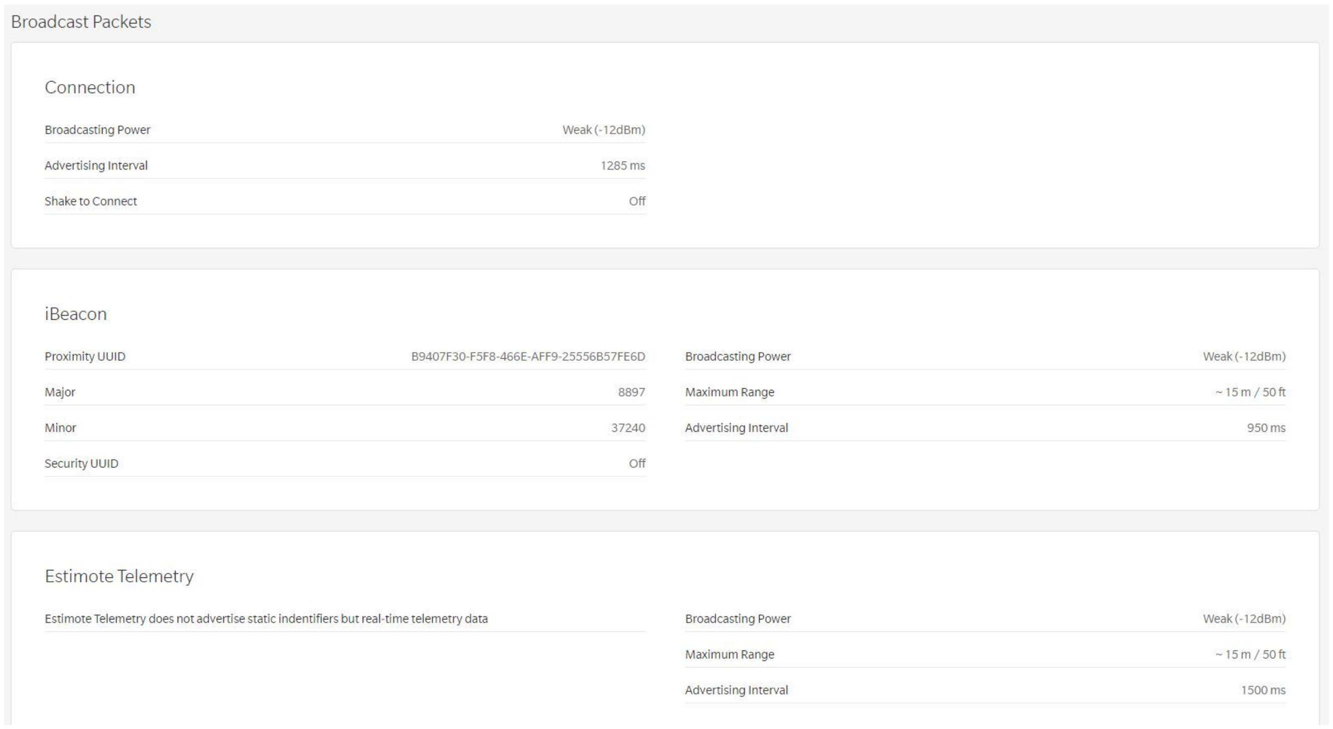
beetroot
Identifier: 12949e33f7d7ec3d558d6c6755969527
Packets: iBeacon (B9407F30-F5F8-466E-AFF9-25556B57FE6D:8897:37240) • Estimote Telemetry
Location: Unknown

candy
Identifier: fe1925dd0b5445b108685f2a803c4335
Packets: iBeacon (B9407F30-F5F8-466E-AFF9-25556B57FE6D:20303:16948) • Estimote Telemetry
Location: Unknown

lemon
Identifier: e0a70b9a289acbd8da7ed9b20264d92c
Packets: iBeacon (B9407F30-F5F8-466E-AFF9-25556B57FE6D:61665:42057) • Estimote Telemetry
Location: Unknown

Figure 2.3 Estimote Online iBeacon Settings Menu

From the beacon selection menu, a user can go in and see the current settings of any beacon they own. From there, the user can edit any settings without doing any code changes. Instead, all a user has to do is edit this page and the beacons will be automatically updated with the new settings. This made it very easy to program all the project's beacons with the settings that the team wanted. The interface for viewing the settings for the team's "beetroot" beacon can be seen in Figure 2.4 [2].



Broadcast Packets

Connection

Broadcasting Power	Weak (-12dBm)
Advertising Interval	1285 ms
Shake to Connect	Off

iBeacon

Proximity UUID	B9407F30-F5F8-466E-AFF9-25556B57FE6D
Major	8897
Minor	37240
Security UUID	Off
Broadcasting Power	Weak (-12dBm)
Maximum Range	~ 15 m / 50 ft
Advertising Interval	950 ms

Estimote Telemetry

Estimote Telemetry does not advertise static indentifiers but real-time telemetry data

Broadcasting Power	Weak (-12dBm)
Maximum Range	~ 15 m / 50 ft
Advertising Interval	1500 ms

Figure 2.4 Estimote Webapp "Beetroot" iBeacon Specifications

2.5 Software Development and Coding

The team used Android Studio, the official Google Android development environment and Java IDE [16], for writing code and uploading programs to the test phones. Using Android Studio for development would cut down on any potential compatibility issues with the project code and would allow the team to quickly and easily test new builds. Additionally, since Android Studio is a full Java IDE, the team would be able to check syntax and compile code directly in the Android Studio program.

Java is an object-oriented coding language used on many different platforms from PCs to smartphones to Internet of Things devices. In fact, it is so ubiquitous, it is the world's number one coding language [17]. Java was the language of choice for the project because it is the main language used for Android development and all of Estimote's example code for Android was written in Java.

In order to maintain proper source control and store project code, the team used Github. Github is a source control program that is often used for open source projects, so there was a lot of Android example code on the service. Additionally, Estimote kept all of their example code repositories on Github, so using Github had made getting started with the project very easy. Github, available in both web and desktop versions [18], allowed the team to track coding progress, document all code revisions, and track which team members have written which segments of code. In this way, in the case of a bug or unexpected problem, the team would be able to track down exactly when that problem started happening, what new code caused it, and even revert to an older build if necessary.

3. Project Objectives and Methodology

When a new student or non ECE student visits Atwater Kent, they often get lost when trying to find a professor's office, a conference room, a lab, or a bathroom. iBeacons can be used to do localization and as monitors of movement in order to track users in Atwater Kent, guide them to their desired location, and prevent them from getting lost. The team also aimed to serve these users with information that relates to their location in the building. The essence of the project and where the team wanted to go with it is summed up in the project mission statement:

Use wireless technologies to make indoor localization simpler and more user-friendly, and to serve specific information to users depending on their location and interests.

In order to complete the project mission, the team had the following three objectives:

1. Use channel modelling techniques to model path loss of bluetooth signals and determine distance from a beacon to a user device using received signal strength (RSS).
2. Apply information derived from channel modelling to perform localization of a user device in Atwater Kent.
3. Create a smartphone application to locate a user device in Atwater Kent and provide contextual information about a user's surroundings.

3.1 Objective 1: Channel Modeling

In this section, we discuss how we collected RSSI data from iBeacon devices. We also discuss how we interpreted and used that information to build a path loss model for Bluetooth signals in our wireless channel, the third floor of Atwater Kent Labs.

3.1.1 Channel Modeling Basics

By determining the path loss, or how much signal strength is lost over a certain distance, the team was able to tell how far away from an iBeacon each user is. In order to compute a location from RSSI data, the team needed to create an algorithm for path loss of Bluetooth signals in Atwater Kent. This algorithm would tell the team how far away a receiver (a phone) was from the transmitter (an iBeacon) given the RSSI readings from the receiver.

The team worked to model Atwater Kent as a transmission channel by estimating the path loss experienced by the Bluetooth signals. For the final result, the team wanted an algorithm for path loss that as closely as possible resembled the actual behavior of the Bluetooth transmission.

The team used IEEE 802.11 Model C [19] as a starting point for this model since this model (residential or small office environment, LOS/NLOS conditions) closely approximates the environment in the hallways of Atwater Kent. Using this model, the team's statistical path loss model was represented by the equation:

$$RSSI(d) = P_r = P_t - L_p(d_0) - 10\alpha \log_{10} d/d_0 + \sigma_{SF}$$

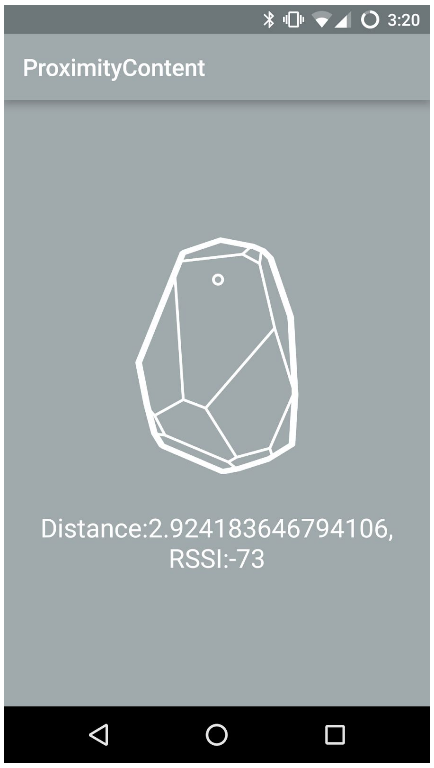
Where d denotes to the actual distance between the smartphone and a specific iBeacon, P_t is the constant transmit power of iBeacons, $L_p(d_0)$ is the path-loss at reference distance d_0. α denotes to the distance power gradient and σ denotes to the shadow fading effect. The ranging estimate (to estimate distance) can be shown as:

$$d = 10^{(RSSI(d) - Pt + Lp(d0) / 10\,\alpha)} \times d_0$$

3.1.2 Collecting Path Loss Data

In order to determine the inputs for this path loss model, the team took measurements of

RSSI at several different distances away from a beacon. In order to do this, the team built a simple Android application to read RSSI and the estimated distance to the beacon that Estimote provides from a test beacon, then used this application to take readings at different distances from the beacon. The team used a tape measure to measure exactly how far away the test device was from the beacon, then recorded the RSSI obtained at a given location, the Estimote estimated distance, and the actual distance to that location. A screenshot of the test application's interface can be found in Figure 3.1.



Distance:2.924183646794106,
RSSI:-73

Figure 3.1 RSSI and Distance Test Application Interface

The relevant code snippets where the balls to access the RSSI and distance information are made can be found below. The rest of the code for the test application can be found in Appendix B.

```
private String placesNearBeacon(Beacon beacon) {
   String rssi = String.format("%d", beacon.getRssi());
   String res= String.format("Distance:%s, RSSI:%s",
Utils.computeAccuracy(beacon),rssi);
```

```
   return res;
}

beaconManager = new BeaconManager(this);
beaconManager.setRangingListener(new BeaconManager.RangingListener() {
   @Override
   public void onBeaconsDiscovered(Region region, List<Beacon> list) {
       String text;
       Integer backgroundColor;
       if (!list.isEmpty()) {
           Beacon nearestBeacon = list.get(0);
           String rssiVals = placesNearBeacon(nearestBeacon);

           text = (rssiVals);
           backgroundColor = null;
       } else {
           text = "No beacons in range.";
           backgroundColor = null;
       }
       ((TextView) findViewById(R.id.textView)).setText(text);
       findViewById(R.id.relativeLayout).setBackgroundColor(
               backgroundColor != null ? backgroundColor : BACKGROUND_COLOR_NEUTRAL);
   }
});
```

This code works by identifying a beacon "region" defined by the broadcast range of a certain beacon (specified by UUID, major, and minor). When the device enters this region, it accesses the beacon's broadcasted information and characteristics as an object of class "Beacon." The application then simply accesses the RSSI and distance estimates from this object, converts them to a text string, then displays the text on the screen to be viewed by a user.

3.1.3 Interpreting Path Loss Data

The team mapped the path loss obtained in these measurements versus distance on a logarithmic scale. These graphs were then used to extrapolate the observed path loss characteristics, along with the algorithm that Estimote uses to do its distance estimates. Finally, the team compared the Estimote distance estimates to the real world distances to determine how much of an error was associated with these distance estimates and whether the error was small enough that these estimates could be used in our localization model.

The path loss models that the team extrapolated from the data were a good starting point for determining the path loss experienced by bluetooth signals in Atwater Kent. However, by themselves they were not good enough to determine the complete path loss experienced by signals and the signal coverage associated with each beacon because they did not take shadow fading into account. In order to model the random shadow fading experienced by signals in Atwater Kent, the team determined the variance in path loss experienced by the iBeacon signals using the data the team had collected. This variance allowed the team to determine a gaussian random variable that predicted the variance of shadow fading. In turn, this shadow fading variable allowed the team to model the actual coverage that each beacon would be capable of at a given broadcast power. In order to do this, the team utilized the following script in Matlab:

```
percentCoverage= .5*(erfc((f/(sqrt(2)*sigma))))
```

Where sigma is the standard deviation of the RSSI from the team's data and f is the shadow fading. Using this information, the team determined how beacons can be deployed on the third floor of Atwater Kent to obtain coverage levels of 50%, 70%, 80%, 90%, 95%, and 99% at different broadcast powers.

3.2 Objective 2: Localization

In order to make navigation in the Atwater Kent building more convenient, especially for guests and new students, the team wanted to be able to localize users very specifically in the building in three dimensions. In other words, users should be able to tell which floor they are on and where they are on that floor. Given path loss information from the previous objective, the team planned on localizing users in Atwater Kent using a least mean squares algorithm.

3.2.1 LMS Localization Algorithm

Using the received signals from the beacons by mobile software, the location of the mobile could be known by triggering their own push notifications (each beacon referred to as a node in this work). All the device need was to measure its distance to beacons. With reliable Time Of Arrival TOA- based distance measurements from landmarks with known locations, the location of a device could be found more accurately. The team used path loss data model to relate mobile locations with the received signal power, and presume that the received signal and mobiles are synchronized such that the SPT can be measured using a time of arrival (TOA) mechanism with low error. The synchronization assumption, however, could be relaxed if the team employed a time difference of arrival (TDOA) technique to measure SPT [6][20].

The team determined the pass loss by measuring RSSI values and inputting them into the following formula:

$$RSSI(d) = P_r = P_t - L_p(d_0) - 10\alpha \log_{10} d/d_0 + \sigma_{SF} \quad (1)$$

Where d denotes to the actual distance between the smartphone and a specific iBeacon, P_t is the constant transmit power of iBeacons, $L_p(d_0)$ is the path-loss at reference distance d_0. α denotes to the distance power gradient and σ denotes to the shadow fading effect.

The time varying Euclidean distance between node k and the mobile terminal is given by $d_k(i) = w^o{}_i - s_k$, where s_k is the known location of node k in two dimensional space and w_o i is the location of the mobile user at time instant i [21].

In addition to RSS measurements, each node records the SPT, $t_k(i)$, which is the signal propagation time from the mobile to node k. If we denote the speed of light by c, $t_k(i)$ can be expressed as:

$$t_k(i) = (d_k(i)/c) + b_k(i) + n^{(t)}{}_k(i) \quad (2)$$

where $b_k(i)$ is a random error with exponential distribution caused by NLOS, and $n^{(t)}_k(i)$ is zero mean measurement noise.

To model the mobile motion over time, the team considered the mobility function Gauss–Markov motion model with constant velocity as following:

$$w^o_i = w^o_{i-1} + v\,[\cos(\varphi(i))\ \sin(\varphi(i))]^T\,\Delta T \qquad (3)$$

where v denotes the mobile speed, $\varphi(i)$ represents the mobile direction at time i and ΔT is the sampling time. The time-varying mobile direction, $\varphi(i)$, is random and changes according to:

$$\varphi(i) = \beta\varphi(i-1) + (1-\beta)\varphi^{-} + [\,2\pi\,(1-\beta^{2})\,]\,u(i) \qquad (4)$$

where φ^{-} is the average direction angle and u(i) is a zero mean random Gaussian variable with variance σ^2_u.

To maintain simplicity in the derivation of the algorithm, the team omitted the index i from w^o_i and work with w^o instead. Let us first assume there exists a fusion center where the measurements by N sectorized base stations (nodes) are sent to for localization. Then, w^o can be found by minimizing the following hybrid global cost function over w:

$$J^{ctrl}(w) = {}_{k=1}{}^{N}\,[\,(1-\eta)\,J^{(p)}_k(w) + \eta\,\nu\,J^{(t)}_k(w)\,] \qquad (5)$$

where $J^{(p)}_k(w)$ and $J^{(t)}_k(w)$ are the local costs associated with node k and related to RSSI and time interval measurements, respectively. The variable $\eta \in [0, 1]$ specifies the amount of the participation of RSSI and SPT measurements in locating the mobile terminal. Parameter ν magnifies $J^{(t)}_k(w)$ to be approximately in the same numerical range as $J^{(p)}_k(w)$. The local cost functions are defined as:

$$J^{(p)}_k(w) = E\,\mathrm{l}p_k(i) + 10\alpha\log \mathrm{ll}w - s_k\mathrm{ll} - h_k(i)\mathrm{l}^2 \qquad (6)$$

$$J^{(t)}_k(w) = E\,\mathrm{l}\,t_k(i) - [(\mathrm{ll}w - s_k\mathrm{ll})/c]\,\mathrm{l}^2 \qquad (7)$$

where:

$$h_k(i) = P_t - L_p(d_0) - 10\alpha\log_{10} d_0$$

The gradient vector of the global objective function (5) can be expressed as:

$$\nabla_W J^{ctrl}(w) = {}^{N}{}_{k=1}\,[\,(1-\eta)\nabla_W J^{(p)}_k + \eta\,\nu\nabla_W J^{(t)}_k\,] \qquad (8)$$

where:

$$\nabla_W J^{(p)}{}_k = [(20\alpha)/(\ln 10)]\ E\ \{ [(w - s_k)/(\,llw - s_k\, ll^2)]\, e^{(p)}{}_k(i) \} \qquad (9)$$

$$\nabla_W J^{(t)}{}_k = -(2/c)\ E\ \{ [(w - s_k)/(\,llw - s_k\, ll)]\, e^{(t)}{}_k(i) \} \qquad (10)$$

with the error functions:

$$e^{(p)}{}_k(i) = p_k(i) + 10\alpha \log llw - skll - h_k(i) \qquad (11)$$

$$e^{(t)}{}_k(i) = t_k(i) - [(llw - s_k ll)/c] \qquad (12)$$

For minimization of (5):

$$w_i = w_{i-1} - \mu \nabla_W J^{\,ctrl}(w_{i-1}) \qquad (13)$$

where parameter $\mu > 0$ is the step size, and w_i is the estimate of the mobile location at iteration i.

Equations (9) and (10) can become smaller by multiplying $\nabla_W J^{(p)}{}_k$ by $llw{-}s_k ll^2$ ln 10/20 and scaling $\nabla_W J^{(t)}{}_k$ with $(c\ llw - s_k ll)/2$. Doing so, lead the team to the final LMS localization which is [14]:

$$\nabla_W J_k(w_{i-1}) = [\alpha(1-\eta)e^{(p)}{}_k(i) - \nu\,\eta\, e^{(t)}{}_k(i)]\,(w_{i-1} - s_k) \qquad (14)$$

$$w_i = w_{i-1} - \mu\ {}^{N}{}_{k=1}\ \nabla_W J_k(w_{i-1}) \qquad (15)$$

Applying the equations above to Matlab using the least squares technique; the least squares technique provided a method of estimating x and y when there are errors in the estimates. Then the team constructed the jacobian matrix as [15]:

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial f_1(x,y)}{\partial x} & \dfrac{\partial f_1(x,y)}{\partial y} \\ \cdots & \cdots \\ \dfrac{\partial f_N(x,y)}{\partial x} & \dfrac{\partial f_N(x,y)}{\partial y} \end{bmatrix}$$

Then the team picked an arbitrary location which was the initial guess and then the team determined the error in the solution where [15]:

$$E_n = -\left(J^T J\right)^{-1} J^T F.$$

The team adapted a Matlab script written by a previous CWINS project to implement LMS [15]. The code takes in beacon coordinates and distances to those beacons to run the algorithm. Running the Matlab code, which is attached in Appendix C-1, gave the team the location of the mobile user in x and y coordinates.

Since the project including created an Android application, the Matlab code needed to be ported to Java. Writing the code in Java was a major issue, since Java does not have robust native matrix manipulation functionality. The team needed to find Java libraries that supported matrix manipulation and write custom functions in order for the Java code to do what the Matlab code does. Luckily, the team was able to implement LMS in Java by leveraging the Efficient Java Matrix Library (EJML). The code for the Java version of the LMS code can be found in Appendix C-2.

3.2.2 Trilateration Localization Algorithm

The team wanted to be able to compare the LMS algorithm to another common localization algorithm. The team chose to implement a trilateration algorithm because it was relatively simple (relying almost primarily on geometric triangulation methods) and easy to implement. It also shared LMS' restriction of only being useful when in range of three beacons, which means the team could use the same datasets to compare the trilateration algorithm to LMS.

In order to implement the algorithm, the team researched trilateration methods using Matlab. The team found that trilateration was very simple to implement graphically, as one only had to draw circles around each reference point, where the radius of each circle was how far away the user was from that reference point. The user location was the intersection of all three of the circles. In order to perform this operation mathematically, the team utilized trigonometric methods rather than drawing circles, but the result remained the same, where the user location was the intersection between the three reference points. The geometry used in trilateration can be described by [13]:

$$r_1^2 = x^2 + y^2 + z^2 r_2^2 = (x - x_2)^2 + y^2 + z^2 r_3^2 = (x - x_3)^2 + (y - y_3)^2 + z^2$$

Simplifying, we find that the trilateration solution is [13]:

$$x = \frac{r_1^2 - r_2^2 + x_2^2}{2x_2} y = \frac{r_1^2 - r_3^2 + x_3^2 + y_3^2 - (2x_3x)}{2y_3} z = \sqrt{r_1^2 - x^2 - y^2}$$

As one can see from the solution, the algorithm is capable of outputting X, Y, and Z coordinates. However, this project only deals with two-dimensional localization, so we only used the X and Y coordinates.

3.2.3 iBeacon Deployment in Atwater Kent Labs

After the team determined the variance of shadow fading from path loss measurements and did the coverage analysis, the team began to think about what sort of deployment pattern would be best. The team wanted to map out a grid deployment for the third floor of Atwater Kent, but this deployment must bear in mind that the team would need a user device to be in range of at least three beacons at all times in order to use LMS for localization. The team deployment would need to account for this restriction, and the team would also have to determine whether the team need to provide coverage for the entire floor or just for the hallways where users would be walking.

In order to check whether the team localization algorithm performed well given our deployment strategy, the team compared their localization technique with the Cramer-Rao Lower Bound (CRLB). This comparison let the team know how effective the localization algorithm they were using was compared to the theoretical bound performance provided by CRLB.

3.2.4 Cramer-Rao Lower Bound (CRLB) Analysis

In its most basic form, a CRLB analysis showed one how much variation to expect in a given region based on a given deployment of access points. In the case of this project, the region was Atwater Kent's third floor, the variation was the localization error, and the access points were the iBeacons. In order to perform this analysis, the team took the deployment maps that were generated earlier in the project and translated the specifics of those deployments to Matlab code. The team then adapted a Matlab script from Professor Pahlavan's ECE 5307 class [22] to generate the CRLB for the deployments. These scripts rely on matrix mathematics [22] as described below:

With N reference APs we have

$$P(r_i) = P_0 - 10\alpha \lg r_i + X; \quad i = 1,\dots N \quad \text{and} \quad r_i = \sqrt{(x-x_i)^2 + (y-y_i)^2}$$

$$dP_i(x,y) = -\frac{10\alpha_i}{\ln 10}\left(\frac{x-x_i}{r_i^2}dx + \frac{y-y_i}{r_i^2}dy\right); \; i = 1,\dots N$$

$$d\mathbf{P} = \begin{bmatrix} dP_1 \\ dP_2 \\ . \\ dP_N \end{bmatrix}; \quad d\mathbf{r} = \begin{bmatrix} dx \\ dy \end{bmatrix}; \quad \mathbf{H} = -\frac{10}{\ln 10}\mathbf{I}_N[\alpha_1 \dots\dots \alpha_N]\begin{bmatrix} \frac{x-x_1}{r_i^2} & \frac{y-y_1}{r_i^2} \\ . & . \\ . & . \\ \frac{x-x_N}{r_N^2} & \frac{y-y_i}{r_N^2} \end{bmatrix}$$

$$d\mathbf{P} = \mathbf{H}d\mathbf{r} \Rightarrow d\mathbf{r} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T d\mathbf{P}$$

Assuming zero mean power variations.

$$E\left\{|d\mathbf{P}|^2\right\} = \operatorname{cov}\left(dP_i, dP_j\right) = \begin{cases} \sigma_P^2, & i = j \\ 0, & i \neq j \end{cases}; \quad i,j = 1,2,\dots N$$

$$E\left\{|d\mathbf{r}|^2\right\} = \operatorname{cov}\left(d\mathbf{r}\right) = \sigma_P^2\left(\mathbf{H}^T\mathbf{H}\right)^{-1} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{bmatrix} \Rightarrow CRLB = Trace\left[\sigma_P^2\left(\mathbf{H}^T\mathbf{H}\right)^{-1}\right] = \sigma_x^2 + \sigma_y^2 = \sigma_r^2$$

$$F = E\left\{|d\mathbf{r}|^2\right\}^{-1} = \frac{\mathbf{H}^T\mathbf{H}}{\sigma_P^2}$$

Figure 3.2 Matrix Mathematics for CRLB Analysis

The Matlab code takes in information about the access point locations (iBeacons in this case), the distance-power gradient of the wireless channel, and the standard deviation of shadow fading in the channel. The code then applies these variables in the matrix math described in Figure 3.2. Finally, the code generates contour plots of the standard deviation of location error in our deployment according to CRLB.

3.3 Objective 3: Smartphone Application

The team developed an application where the user's phone is used to determine the distance from the beacons in its vicinity. This information is then processed using an LMS localization algorithm to determine the user's location on a map. After this, the application overlays the user's location on a map of Atwater Kent so that user would be able to tell where they need to go from their current location to reach their destination.

Once the team had established the ability to localize users based on proximity to an iBeacon, we used this information to serve users with information based on where they are in Atwater Kent. More specifically, the team was trying to leverage user location information to tell when a user is near a certain professor's office, different labs, conference rooms, and lecture halls. If a user is near a particular location, the user would get a notification from the application that they are in a particular spot in Atwater Kent and that information is available for that location They will then either be able to dismiss the notification or click on it for details, which may include more text or an html link. For instance, a user may be walking by Professor Pahlavan's office and will get a notification on their phone that they are right near his office door. Once the user has read the notification, if they are interested they will be able to click on it for more information. In the case of Professor Pahalavan's office, this information might take the form of an html link to his biography on the WPI site. A mockup for what the main window of the application will look like can be seen below, in Figure 3.1. In this figure, red dots are beacon locations, stars are points of interest, and the blue dot is the user's current location.

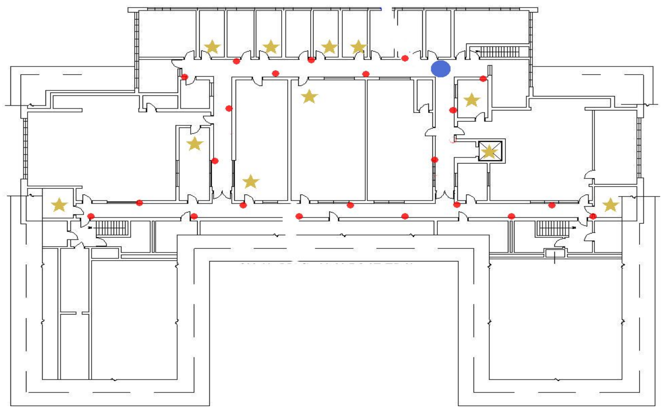


Figure 3.3 Early Application Graphical User Interface (GUI) Mockup

Overall, the essential goal of this application was to make it easier to quickly obtain information about the professors, labs, and conference rooms in Atwater Kent. In the past, people have attempted to serve users with this sort of simple information using things like physical signs and QR codes. The problem with signs is that the type and amount of information they can display is limited. QR codes are a little better in that they can provide information such as html links, but ultimately QR codes are not a very good solution for quickly serving users with information because they require a lot of interaction on the part of the user to actually be effective. For instance, if a user wanted information about a particular office, they would have to find the QR code nearby that pertained to that office, pull out their phone, scan the code, and read the information. Most people do not even have QR code reader applications on their phones. Therefore, having an application that automatically serves useful information to users depending on where they are is a much better solution than QR codes because most everyone has a smartphone and this application would require almost no additional user interaction.

3.4 Application Testing

The team tested the application by going to the third floor of Atwater Kent Labs and taking readings on the application. The test area is called out in Figure 3.3.

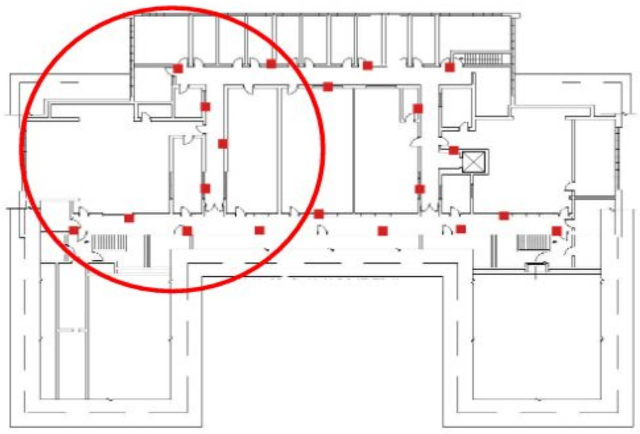


Figure 3.4 Application Test Area on the Third Floor of Atwater Kent Labs

The coordinates given by the algorithm were compared with the actual coordinates calculated using a tape measure. Sixty different readings were taken and compared to the actual locations. The results of this testing can be found in Appendix J.

4. Results

This section outlines the results of the project objective by objective. We begin by discussing our channel model, then touch on our localization model, and then discuss the Android application that incorporates the channel model and localization algorithm. Finally, we evaluate the performance of our application in a real-world test. We also describe how we did our statistical coverage analysis to account for shadow fading in the wireless channel.

4.1 Channel Modeling

The channel modeling subsection discusses the team's results related to our first objective. We discuss our findings from our path loss testing and compare those results to the path loss algorithm that Estimote uses to approximate distance.

4.1.1 Determining Path Loss

In order to determine values for this path loss model, the team measured the path loss of the iBeacon signal versus real world distance. The standard deviation of this data, which is used for the CRLB analysis, was 9.6 dBm. Plotting the path loss in dBm versus distance in meters on a logarithmic scale, the team obtained Figure 4.1. The full set of data for this figure can be found in Appendix K.

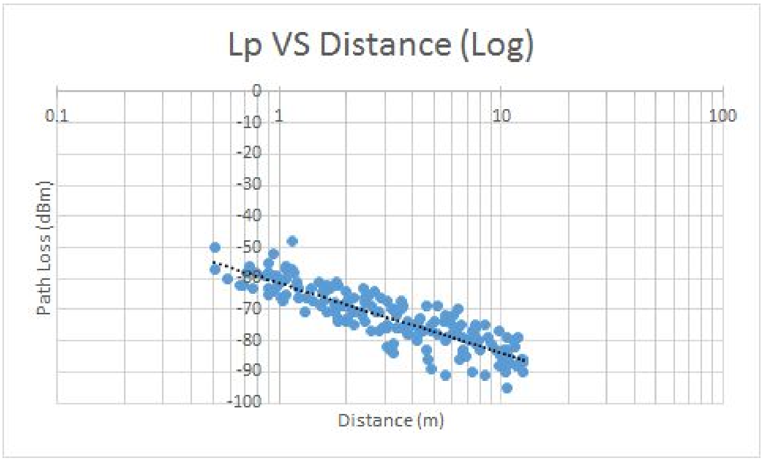


Figure 4.2 Observed Bluetooth Signal Path Loss Vs Real World Distance

Using this graph, the team determined that the path loss that the bluetooth signals experienced was described by a first meter path loss of 60 dBm and an alpha of 3.3. This yields the path loss equation found equation (16):

$$Lp= 60 +33\log(d) \tag{16}$$

The team then did measurements to determine the path loss algorithm that Estimote uses for its built-in distance estimates. The graph of this data can be seen in Figure 4.2.

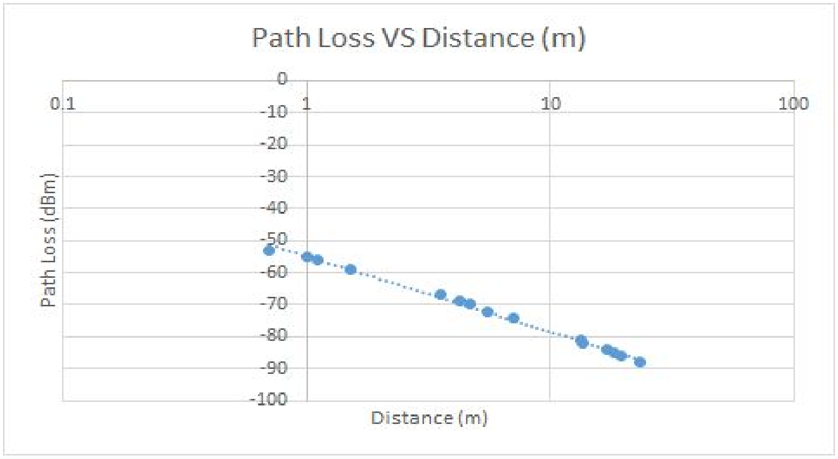


Figure 4.2 Observed Bluetooth Signal Path Loss Vs Estimated Distance

Using this graph, the team determined that the path loss that the bluetooth signals experienced was described by a first meter path loss of 55 dBm and an alpha of 2.5. This yields the path loss equation found equation (17):

$$Lp= 55 + 25\log(d) \tag{17}$$

This path loss equation from Estimote is closer to the theoretical path loss described in the IEEE model, which predicts an alpha of 2 (for line of sight conditions) and a first meter path loss of 40-50 dBm. Therefore, the team decided to investigate whether we could simply use these distance estimates in our localization algorithm. To do this, we took measurements comparing the estimated distance to the real distance. A graph of this data can be found in Figure 4.3.

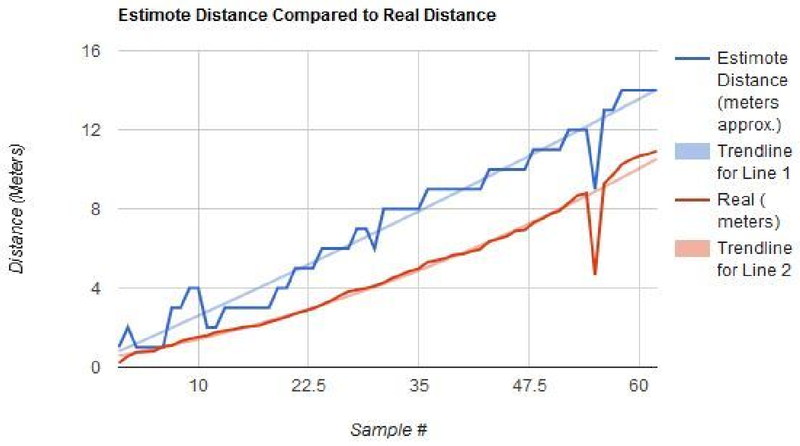


Figure 4.3 Estimote Distance Approximation Compared to Real Distance

This graph shows that the distance estimates provided by Estimote are actually fairly accurate, and look to be able to provide a good estimate of distance to a user device within 2 or 3 meters of the actual distance. Additionally, from these tests, it seems like the estimates are fairly consistently biased towards being slightly above the real distance.

4.1.2 Coverage Analysis

After determining a path loss equation for the iBeacon signals, the team expanded that equation by adding a term signifying the fade margin, or the amount of path loss associate with shadow fading at a given coverage level. This allowed the team to determine the amount of coverage that could be statistically guaranteed for a given radius at a given transmission power. This was done using Matlab as described in section 3.1. The results of this coverage analysis can be seen in Table 4.1. An expanded version of the chart is available in Appendix H.

Table 4.1 Statistical Coverage Analysis with Shadow Fading

			Transmission Power= 4 dBm	Transmission Power= 0 dBm	Transmission Power= -4 dBm
Percent Coverage	Fade Margin (dB)	Equation	coverage radius (m)	coverage radius (m)	coverage radius (m)
50% (0.5)	0	Lp= 55+ 25log(d)	47.86	33.113	22.909
70% (0.699)	5	Lp= 55+ 25log(d)+ 5	30.2	20.893	14.454
80% (0.798)	8	Lp= 55+ 25log(d)+ 8	22.91	15.849	10.965
90% (0.9003)	12.3	Lp= 55+ 25log(d)+ 12.3	15.42	10.666	7.379
95% (0.9503)	15.8	Lp= 55+ 25log(d)+ 15.8	11.169	7.727	5.346
99% (0.99)	22.3	Lp= 55+ 25log(d)+ 22.3	6.138	4.246	2.938

Bearing this analysis in mind, the team chose to build the beacon deployment with the beacons transmitting at 4 dBm, as this statistically guarantees 90% coverage in a 15.42 meter broadcast radius. This should assure that the beacon coverage on the third floor of Atwater Kent is solid enough to reliably perform localization.

4.2 Localization in Atwater Kent

In this section, we discuss the results of our research on localization. We compare the LMS localization algorithm's performance to a simpler trilateration algorithm. We also discuss the specifics of our iBeacon deployment in Atwater Kent. Finally, we perform a CRLB analysis of our iBeacon deployment using information from our path loss testing.

4.2.1 Localization Algorithms

During this project, the team explored two different localization algorithms. The first was least mean squares (LMS). This is an algorithm that relies on three reference points (in this case, beacons) to do successive guessing to find a user position. The mathematics behind it are fairly complex, but this algorithm tends to be accurate enough in practice that it is useful in real-world applications.

The second algorithm the team examined was trilateration, which is a much simpler algorithm based off of geometric triangulation techniques that also requires three reference points to work. This localization method is very simple to implement, but is usually not accurate enough in practice to be used in real-world scenarios. Taking that into account, the team is simply using trilateration as a comparison point to the LMS algorithm's performance to ensure that the LMS algorithm is performing reasonably well, with little error.

4.2.1.1 Least Mean Squares (LMS) Algorithm

As mentioned in chapter 3, the team adapted a Matlab script written by a previous CWINS project to implement LMS. The code basically generate the mobile user location in x and y coordinates. All the code needs is the beacon's location and the distances between the mobile user and the beacons. The code start from an arbitrary location which is initial guess point of the mobile user location; the code keep iterating till the estimation error get so close as 0.01. This can be seen in Figure 4.4.

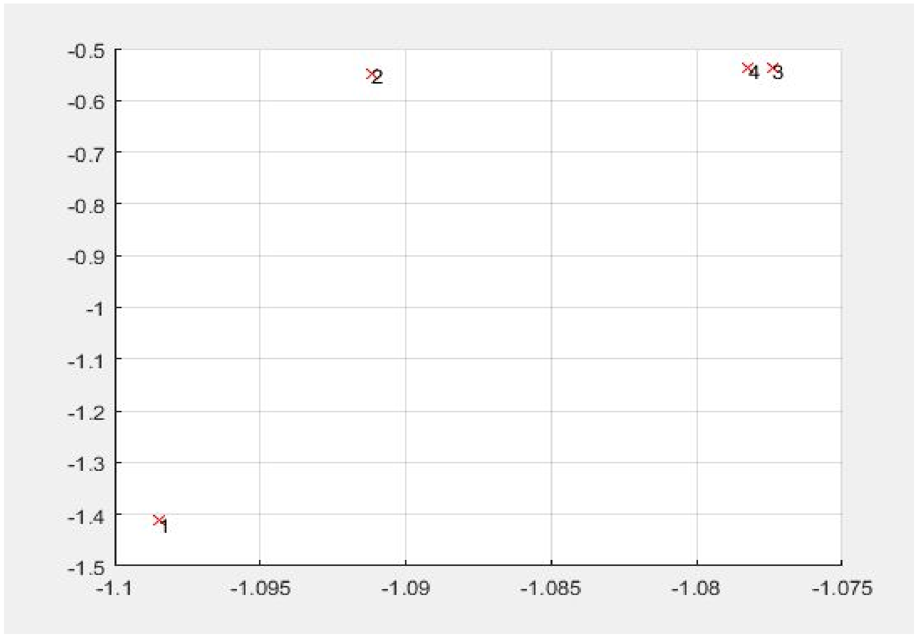
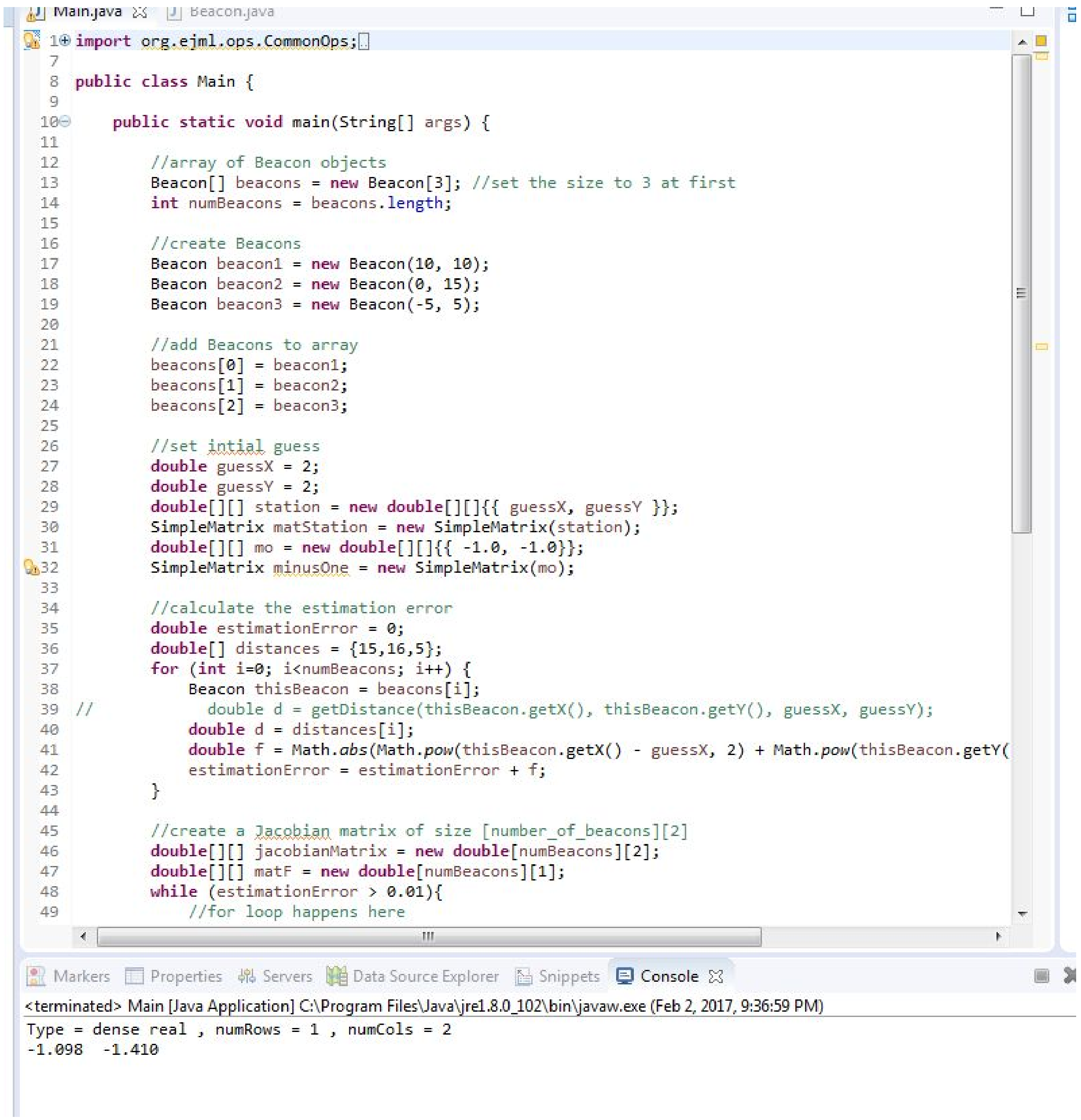


Figure 4.4 Example Output of Matlab LMS Algorithm

In order to make the LMS interact with smartphone application, the team write another LMS code in Java that has the same functionality as Matlab code. This code generates the location of the mobile user in x and y coordinates. Figure 4.5 is an example of the output of the Java code.



Main.java   Beacon.java

```java
import org.ejml.ops.CommonOps;

public class Main {

    public static void main(String[] args) {

        //array of Beacon objects
        Beacon[] beacons = new Beacon[3]; //set the size to 3 at first
        int numBeacons = beacons.length;

        //create Beacons
        Beacon beacon1 = new Beacon(10, 10);
        Beacon beacon2 = new Beacon(0, 15);
        Beacon beacon3 = new Beacon(-5, 5);

        //add Beacons to array
        beacons[0] = beacon1;
        beacons[1] = beacon2;
        beacons[2] = beacon3;

        //set intial guess
        double guessX = 2;
        double guessY = 2;
        double[][] station = new double[][]{{ guessX, guessY }};
        SimpleMatrix matStation = new SimpleMatrix(station);
        double[][] mo = new double[][]{{ -1.0, -1.0}};
        SimpleMatrix minusOne = new SimpleMatrix(mo);

        //calculate the estimation error
        double estimationError = 0;
        double[] distances = {15,16,5};
        for (int i=0; i<numBeacons; i++) {
            Beacon thisBeacon = beacons[i];
//              double d = getDistance(thisBeacon.getX(), thisBeacon.getY(), guessX, guessY);
            double d = distances[i];
            double f = Math.abs(Math.pow(thisBeacon.getX() - guessX, 2) + Math.pow(thisBeacon.getY(
            estimationError = estimationError + f;
        }

        //create a Jacobian matrix of size [number_of_beacons][2]
        double[][] jacobianMatrix = new double[numBeacons][2];
        double[][] matF = new double[numBeacons][1];
        while (estimationError > 0.01){
            //for loop happens here
```

Markers   Properties   Servers   Data Source Explorer   Snippets   Console

```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_102\bin\javaw.exe (Feb 2, 2017, 9:36:59 PM)
Type = dense real , numRows = 1 , numCols = 2
-1.098  -1.410
```

Figure 4.5 Java Version of LMS Code and Example Output

4.2.1.2 Trilateration Algorithm

In order to have another algorithm to compare the LMS algorithm's performance to, the team built a trilateration algorithm in Matlab. This algorithm simply outputs X and Y coordinates for a given set of fixed reference points and distances. Much like LMS, this algorithm requires three beacons because it relies on triangulation methods. This allows the trilateration to be easily compared to LMS because the same data sets can be used for both algorithms. The code for this algorithm can be found in Appendix D.

4.2.1.3 Localization Error

After LMS was generated in Matlab, the team did some measurements manually in the third floor of Atwater Kent to compare the real mobile user location with the location that is generated by the LMS code. The team generated 14 mobile user locations. In all these 14 locations, the mobile user was within the range of three beacons. This measurements can be found later section.

In order to make sure that LMS is a reliable algorithm in this project, the team also wrote code for a trilateration algorithm in Matlab. By comparing the Trilateration to LMS, the team found that the two algorithms had similar performance for some data points, but that trilateration had several points that were way off of the real location. Overall, the LMS algorithm has an average error of 6.18 meters, with a standard deviation of 9.02 meters. Meanwhile, the trilateration algorithm had an average error of 13.47 meters, with a standard deviation of 27.02 meters. This means that, on average, the LMS algorithm will perform with a lower degree of error. The relevant results of this error analysis can be seen in Table 4.2. The full version of the error test table can be found in Appendix I.

Table 4.2 Comparing LMS and Trilateration Localization Accuracy

User Real Location	LMS User Location	Error	Trilateriation Location	Trilat Error
(91 , 37)	(89.37 , 37.28)	(1.63 , -0.28)	(91.70, 37.02)	(-0.70, -0.02)
(90, 50)	(93.75 , 60.75)	(-3.75 , -10.75)	(93.76, 58.62)	(-3.76, -8.62)
(199 , 124)	(195.34 , 98.96)	(3.66 , 25.04)	(240.47, 98.52)	(-41.47, 25.48)
(204 , 81)	(200.44 , 70.94)	(3.56 , 10.06)	(206.85, 78.20)	(-2.85, 2.80)
(201 , 137)	(192.83 , 135.32)	(8.17 , 1.68)	(193.10, 133.71)	(7.90, 3.29)
(64 , 203)	(74.02 , 222.83)	(-10.02 , -19.83)	(192.26, 137.83)	(-128.26, 65.17)
(175 , 189)	(178.36 , 146.92)	(-3.36 , 42.08)	(179.84, 143.62)	(-4.84, 45.38)
(235 , 31)	(234.15 , 31.63)	(0.85 , -0.63)	(234.06, 30.94)	(0.94, 0.06)
(286 , 30)	(287.96 , 30.58)	(-1.96 , -0.58)	(287.90, 29.80)	(-1.90, 0.20)
(175 , 189)	(175.48 , 188.62)	(-0.48 , 0.38)	(171.34, 192.17)	(3.66, -3.17)
(338 , 37)	(335.29 , 34.63)	(2.71 , 2.37)	(337.17, 34.64)	(0.83, 2.36)
(356 , 10)	(349.86 , 11.08)	(6.14 , -1.08)	(355.13, 19.97)	(0.87, -9.97)
(401 , 54)	(401.20 , 52.75)	(-0.20 , 1.25)	(402.56, 54.83)	(-1.56, -0.83)
(340 , 66)	(344.54 , 72.05)	(-4.54 , -6.05)	(344.50, 71.79)	(-4.50, -5.79)

The team graphed this error information in histograms to determine statistically what percentage of calculations could be expected to have little error versus what percentage of calculations could be expected to have large errors. These graphs can be seen in Figure 4.6.

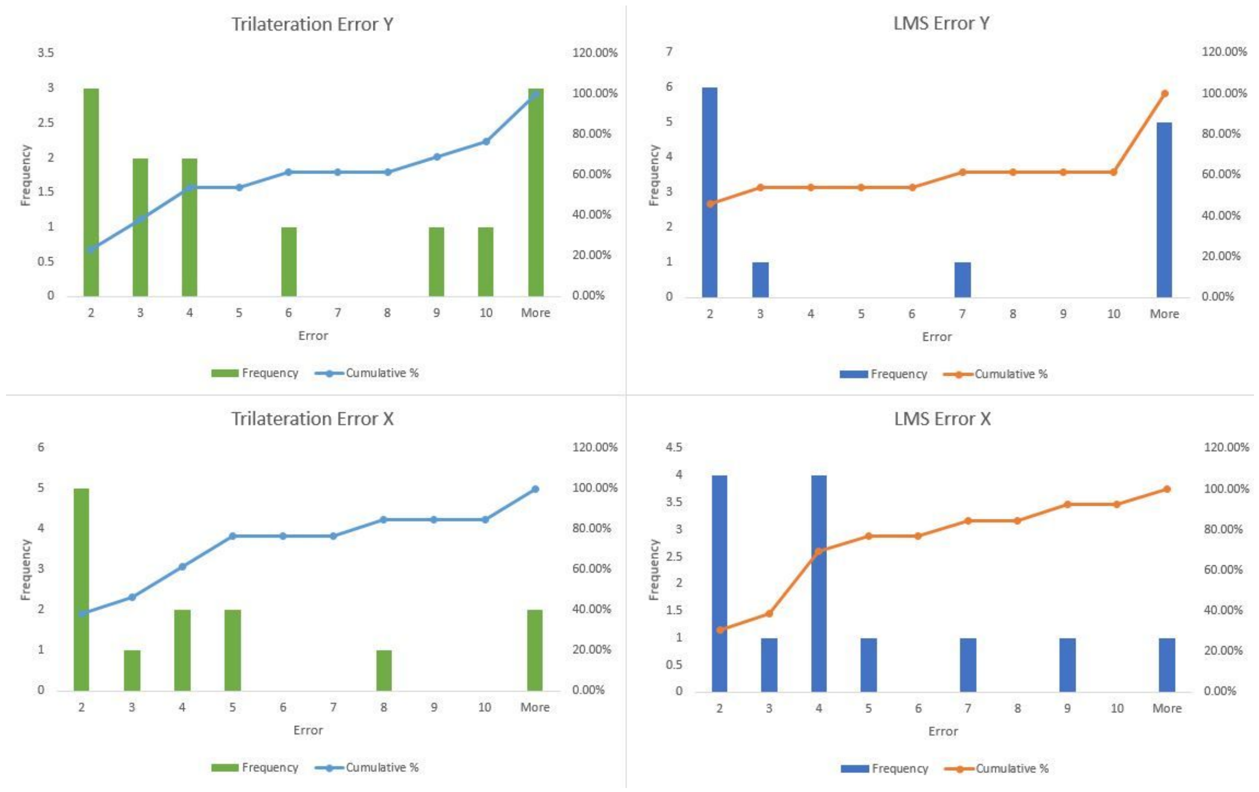


Figure 4.6 Comparing LMS and Trilateration Localization Accuracy for X and Y Coordinates

The major conclusions to be made out of this analysis are that, in the team's testing, LMS had a better average error and that trilateration had a few very large error data points, meaning it had a very high standard deviation. Additionally, the team can expect that 90% of the time, the LMS algorithm will produce an error of 10 units or less for the X coordinate and that 60% of the time, the LMS will produce an error of 10 units or less for the Y coordinate. This gives the algorithm fairly good odds to be close to the true user location and so could most likely be used, with acceptable error, for localization.

4.2.2 iBeacon Deployment on Atwater Kent Labs Third Floor

In order for the team to use the LMS localization algorithm in the application, the team needed to deploy actual beacons in Atwater Kent. To do this, the team leveraged the coverage analysis described in section 4.1.3 to determine what broadcast ranges are available at different broadcast powers. The team came up with two deployment options for the third floor of Atwater Kent: a purely grid-type deployment and a deployment where beacons are only placed in hallways. The team then did a Cramer-Rao Lower Bound (CRLB) analysis on these deployment options to determine their expected performance for localization and to give a performance reference for the real localization in the smartphone application.

4.2.2.1 Deployment Maps

The team ultimately decided to deploy the beacons assuming a 15 meter broadcast range on all the beacons and a transmission power of 4 dBm. This statistically guarantees with 90% confidence that a user will receive signal from a beacon even at the very edge of this broadcast radius. Given this broadcast radius, the team identified two main deployment plan options. The first is a grid-based deployment. This deployment can be seen in figure 4.7, which is a map of Atwater Kent's third floor split into one meter grids. The red dots on the map represent beacon locations.

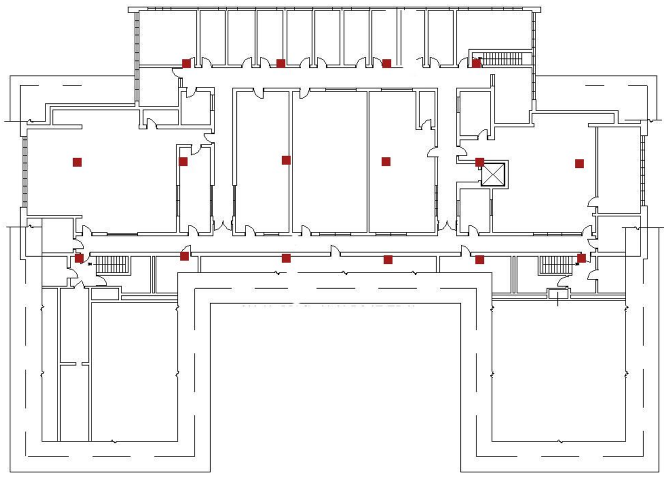


Figure 4.7 Grid Deployment of iBeacons for Atwater Kent Labs Third Floor

While the grid deployment is fairly simply geometrically speaking, it has a number of practical issues. One issue is that the deployment calls for placing beacons in offices and labs where people may not want them placed, they could interfere with experiments, and they would be difficult to access for maintenance. Another issue is that this deployment has many NLOS conditions. Since the Estimote distance algorithm is based off of LOS conditions, having a lot of NLOS conditions in the deployment is not good.

In order to solve some of these problems, the team developed a deployment that only calls for placing beacons in hallways. Since users will almost always be in hallways while using the application, it should be safe to focus the deployment on hallways. A map of this deployment can be seen in Figure 4.8.

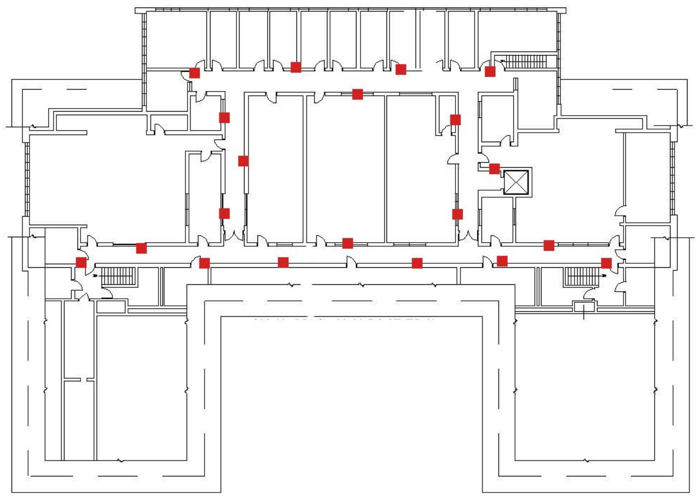


Figure 4.8 Hall Deployment of iBeacons for Atwater Kent Labs Third Floor

Much like the grid deployment, the hall deployment should guarantee three beacon coverage for all the hallways. While the hall deployment calls for more beacons (20 total as opposed to 16 total for the grid deployment), the hall deployment is better in that it allows more freedom for where the team places the beacons and beacons do not have to be placed in hard to access places like labs. Additionally, this deployment assures LOS conditions for almost all points, which should make the application's localization functionality more accurate.

4.2.2.2 Cramer-Rao Lower Bound (CRLB) Analysis

The team performed a CRLB analysis in Matlab to determine what sort of error range to expect for different areas in the deployment. The code for this analysis can be found in Appendix G. Figure 4.9 shows the CRLB for the entire grid deployment. Figure 4.10 shows just the lower left-hand section of the grid deployment. Figure 4.11 shows the CRLB analysis for the entire hall deployment. Figure 4.12 shows just the lower left-hand section of the hall deployment.

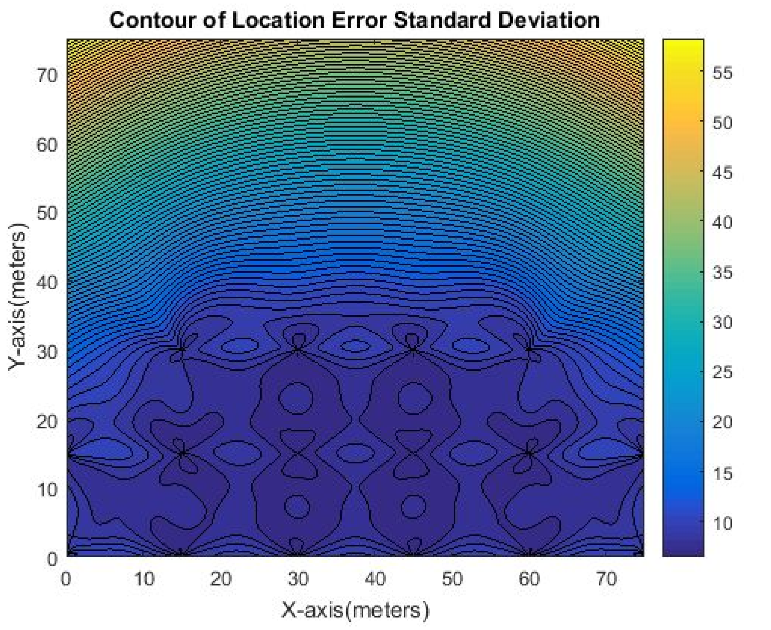


Figure 4.9 CRLB for Grid Deployment

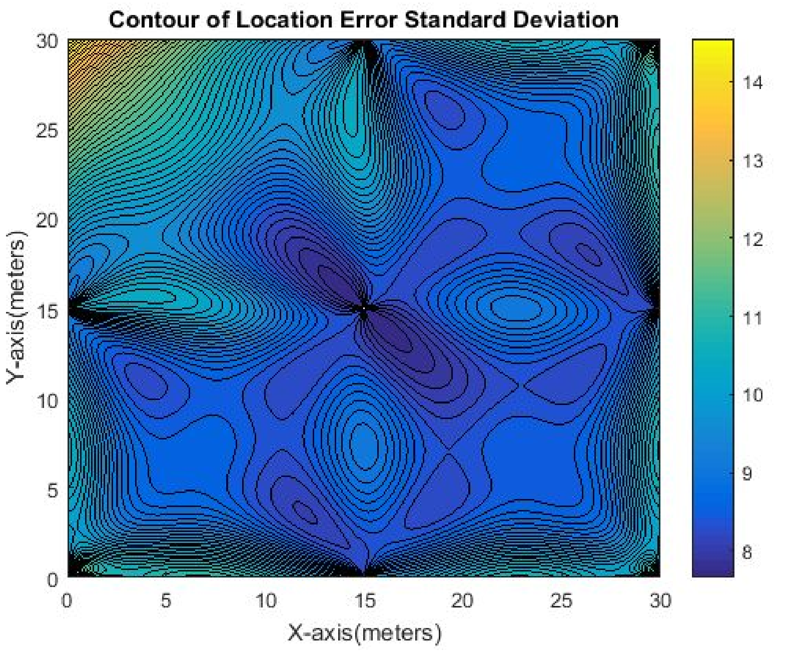


Figure 4.10 Smaller Section of CRLB for Grid Deployment

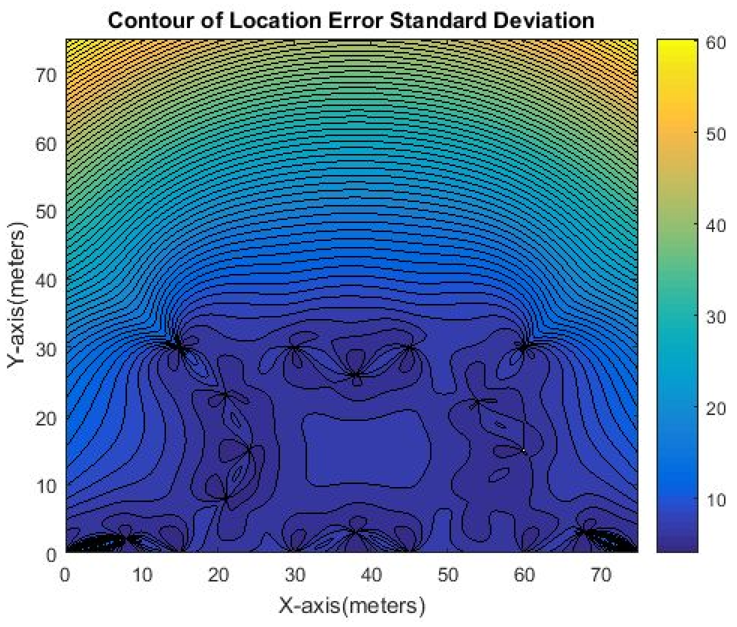


Figure 4.11 CRLB for Hall Deployment

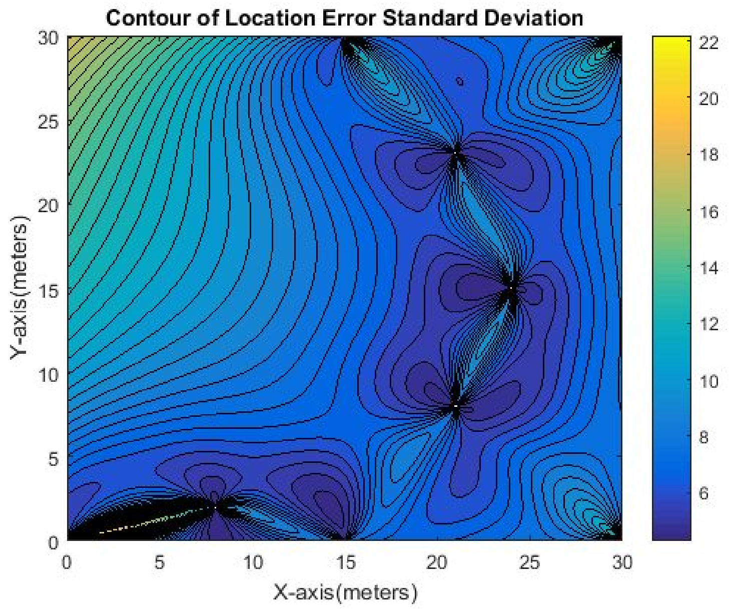


Figure 4.12 Smaller Section of CRLB for Hall Deployment

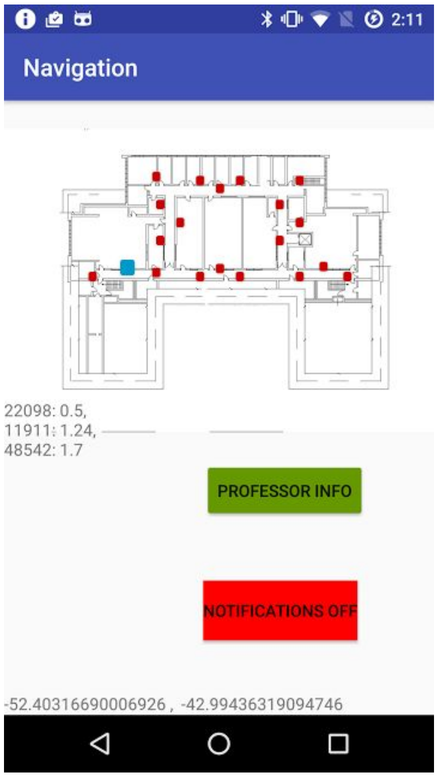
This analysis shows that the team can expect a standard deviation of location error of between 4 and 12 for the hallways in the hall deployment and between 7 and 10 for the hallways in the grid deployment. This means that both these deployments have potential errors associated with them, but they are not so large as to be hugely significant in a theoretical scenario.

4.3 Smartphone Application

In this section, we discuss our smartphone application, which incorporates information from our channel model and the LMS localization algorithm we built. We discuss the functionality of the application. We also evaluate the application’s real-world localization performance.

4.3.1 Graphical User Interface (GUI) and Main Navigation Screen

The main screen consists of a map of atwater kent which shows the user location as a blue dot based on input from the beacons. The screen also shows the beacons as red dots. For testing purposes the distances to each of the beacons are also shown, as well as output from the LMS algorithm. There is a button to proceed to the contextual information screen (Professor Info) as well as a button to toggle whether notifications are given or not based on close beacons. The numbers in the middle left are the beacon majors of the three closest beacons, as well as the distances (in meters) to those beacons. In the bottom left are the X and Y coordinate returned by the LMS algorithm based on those beacons, where the 0,0 point is in the upper right.



```
22098: 0.5,
11911: 1.24,
48542: 1.7
```

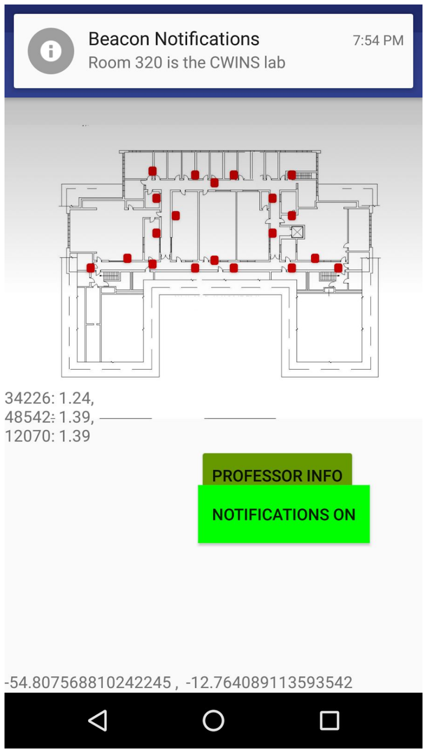
```
-52.40316690006926 , -42.99436319094746
```

Figure 4.13 Android Application Main Activity Window

The relevant code can be found in Appendix B.

4.3.2 Notification Functionality

The team implemented notifications into the application with the user receiving a notification whenever they enter the range of a beacon (Assuming they have notifications on). These notifications tell the user about the closest room to the beacon. An example notification can be seen in Figure 4.14.


Beacon Notifications 7:54 PM
Room 320 is the CWINS lab

34226: 1.24,
48542: 1.39,
12070: 1.39

-54.807568810242245 , -12.764089113593542

Figure 4.14 Android Application Example iBeacon Proximity Notification

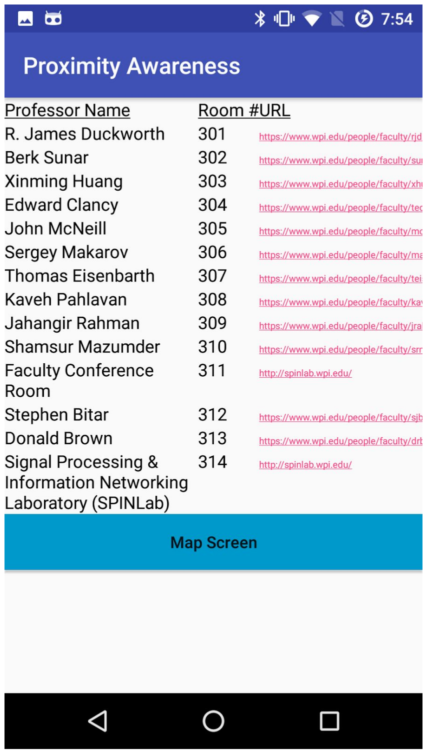
The code for the beacon notifications can be found in Appendix A. Below is a code snippet for creating a new notification with a specific beacon.

```
beaconNotificationsManager.addNotification(//Beetroot 3
        new BeaconID("6EE4D6A9-DD8E-550E-FF81-783E445F9C5B",48542,60126),
        "Room 320 is the CWINS lab",
        null);
```

This code specifies the ID of the beacon that the team is communicating with, as well as major and minor. These three values together identify a specific iBeacon. Next, the messages for when the phone is entering transmission range and exiting transmission range are specified. Using this template, the team added functionality to the application to notify a user about useful information when they are entering the transmission range of any beacon. This forms one of the core elements of the applications end functionality, as the team wants to be able to notify users when they are near certain points of interest and deliver information to them about those locations.

4.3.3 Contextual Information

When a user clicks the Professor Info button, the application displays a second window with information about the rooms on the floor. This window can be seen in Figure 4.15.



Professor Name	Room #URL	
R. James Duckworth	301	https://www.wpi.edu/people/faculty/rjd
Berk Sunar	302	https://www.wpi.edu/people/faculty/su
Xinming Huang	303	https://www.wpi.edu/people/faculty/xh
Edward Clancy	304	https://www.wpi.edu/people/faculty/te
John McNeill	305	https://www.wpi.edu/people/faculty/m
Sergey Makarov	306	https://www.wpi.edu/people/faculty/ma
Thomas Eisenbarth	307	https://www.wpi.edu/people/faculty/tei
Kaveh Pahlavan	308	https://www.wpi.edu/people/faculty/ka
Jahangir Rahman	309	https://www.wpi.edu/people/faculty/jra
Shamsur Mazumder	310	https://www.wpi.edu/people/faculty/sr
Faculty Conference Room	311	http://spinlab.wpi.edu/
Stephen Bitar	312	https://www.wpi.edu/people/faculty/sjb
Donald Brown	313	https://www.wpi.edu/people/faculty/dr
Signal Processing & Information Networking Laboratory (SPINLab)	314	http://spinlab.wpi.edu/

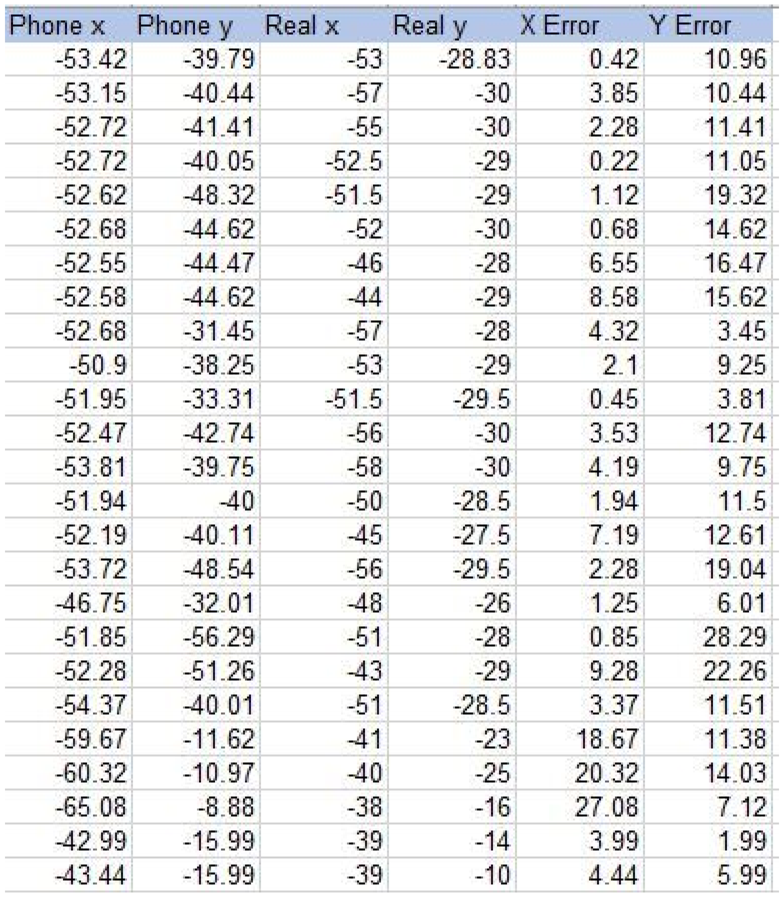
Figure 4.15 Android Application Contextual Information Window

The team has also built a database of the information that will be displayed to a user through the app. A table describing the contents of the database so far can be seen in Appendix D.

4.4 Application Performance Evaluation

In order for the team to investigate the performance of the application, the team deployed actual beacons on third floor of Atwater Kent. We then measured and collect data on where the application showed us to be on the map versus where we were in real life. Table 4.3 shows a selection of the collected data. The full version of the data table can be found in Appendix J.

Table 4.3 Application Localization Performance Evaluation Data



Phone x	Phone y	Real x	Real y	X Error	Y Error
-53.42	-39.79	-53	-28.83	0.42	10.96
-53.15	-40.44	-57	-30	3.85	10.44
-52.72	-41.41	-55	-30	2.28	11.41
-52.72	-40.05	-52.5	-29	0.22	11.05
-52.62	-48.32	-51.5	-29	1.12	19.32
-52.68	-44.62	-52	-30	0.68	14.62
-52.55	-44.47	-46	-28	6.55	16.47
-52.58	-44.62	-44	-29	8.58	15.62
-52.68	-31.45	-57	-28	4.32	3.45
-50.9	-38.25	-53	-29	2.1	9.25
-51.95	-33.31	-51.5	-29.5	0.45	3.81
-52.47	-42.74	-56	-30	3.53	12.74
-53.81	-39.75	-58	-30	4.19	9.75
-51.94	-40	-50	-28.5	1.94	11.5
-52.19	-40.11	-45	-27.5	7.19	12.61
-53.72	-48.54	-56	-29.5	2.28	19.04
-46.75	-32.01	-48	-26	1.25	6.01
-51.85	-56.29	-51	-28	0.85	28.29
-52.28	-51.26	-43	-29	9.28	22.26
-54.37	-40.01	-51	-28.5	3.37	11.51
-59.67	-11.62	-41	-23	18.67	11.38
-60.32	-10.97	-40	-25	20.32	14.03
-65.08	-8.88	-38	-16	27.08	7.12
-42.99	-15.99	-39	-14	3.99	1.99
-43.44	-15.99	-39	-10	4.44	5.99

By comparing the LMS algorithm's output to real-world positions, the team was able to determine the localization error at several different locations in Atwater Kent. From the CRLB analysis, the team was expecting a good deal of error, as the standard deviation for location error in the hallways ranged from 4 to 12 meters depending on the position. As can be seen in Figure 4.16, the average X coordinate error was 8.67 meters with a standard deviation of 8.94 meters. Figure 4.17 shows the Y coordinate with an average error of 8.88 meters and a standard deviation of 6.43 meters.

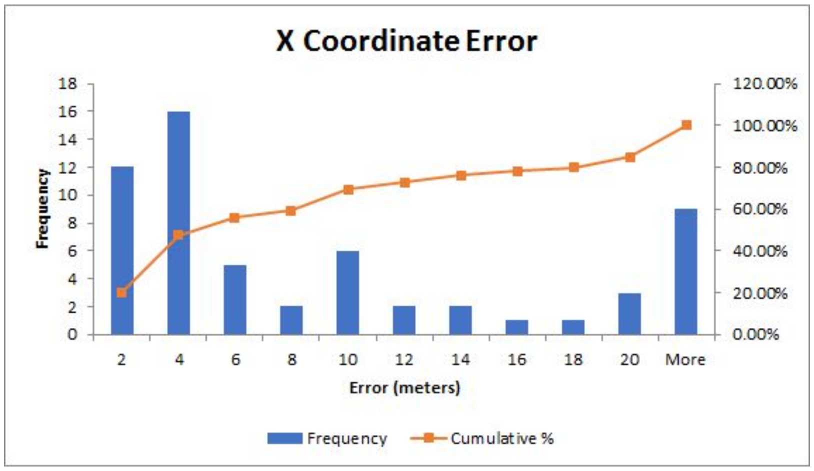


Figure 4.18 X Coordinate Error for Application Localization

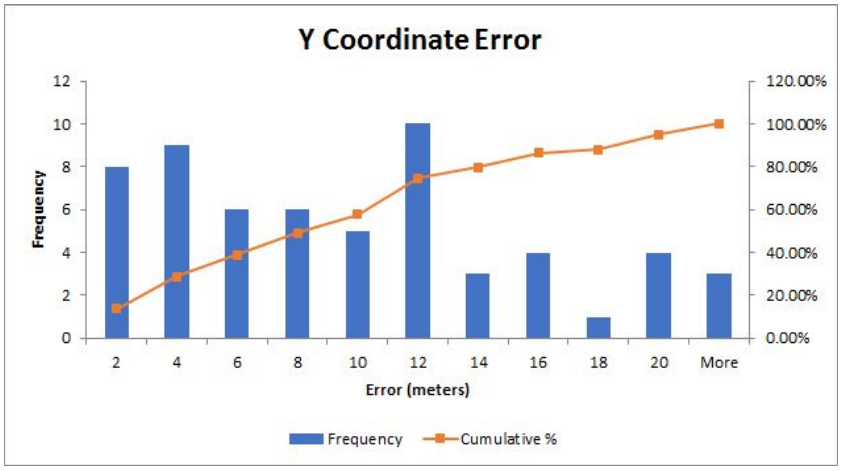


Figure 4.19 Y Coordinate Error for Application Localization

Overall, the average error was 8.78 meters with a standard deviation error of 7.72 meters. The average error was somewhat larger than what the team had expected given the LMS algorithm testing. However, the standard deviation of the results is in line with what the team expected of the deployment given the CRLB analysis the team conducted, which expected an error between 4 and 12 meters of standard deviation based on position. This indicates that the application's performance is in line with theory, but not working as well as it could in ideal conditions.

While this project provides a good proof of concept for indoor localization, the localization error is too large to reasonably be useful for a user in real-time. The team determined that a major source of this error is inaccurate distance estimates from the iBeacons due to fluctuating RSSI readings. For instance, in one test, the distance estimation to an iBeacon was 6 meters, but the actual distance to that iBeacon was 15 meters (representing a 150% ranging error). This seems to indicate that the inaccurate localization results are due to unreliable distance approximations from the iBeacon signals. Moreover, the team also experienced issues with getting reliable signal from three iBeacons at once, as the application was only able to pick up signal from three iBeacons for a few seconds at any one time. This made it difficult to smoothly track a user's position as they were moving through the hallways, and resulted in the user location jumping across the map during use.

5. Conclusions and Future Work

During our testing, the team determined that the RSSI values obtained from the iBeacons can fluctuate significantly. Oftentimes, the team would still have to stand in a location for several seconds in order to get accurate RSSI readings. This resulted in inaccurate distance approximations being fed into the localization algorithm when testing it with real-time data. The team found the localization provided by the LMS algorithm to be fairly accurate in theory, but in practice the application's localization functionality had significant error associated with it because of the inaccurate real-time distance measurements. This indicates that it is difficult to accurately measure distance using telemetry from the iBeacons and that the Estimote distance approximation is not accurate enough to do localization with. A major element of future work on this subject will be determining a way to obtain more stable and accurate RSSI readings from the iBeacons in real-time. Getting accurate distance approximations from the RSSI readings could dramatically increase the performance of the localization algorithm. It would also prevent the user icon from jumping around on the map, which happened quite a lot during our testing. For a real application, the user icon would need to smoothly follow the path that the user is walking so the user could actually navigate using the application.

In our final performance evaluation, we found that the average localization error in our application was 8.88 meters. This error might be good enough for a large outdoor environment, but for an indoor environment, it is too large for the application to be useful to a real user. The team believes that if the average error can be brought down to 4 meters, which is the approximately the distance between doors of adjoining offices in Atwater Kent, the localization functionality in the application could be useful to an actual user. At this level of error, a user would always get a localization estimate that was no more than one room away from where they actually are.

Overall, using iBeacon technology for navigation and proximity awareness shows promise, but the technology is not quite ready for real-world use. In order to develop the technology for future applications, future projects should explore different models to use to approximate distance with and compare the performance of those model to the Estimote algorithm. Additionally, a future team should build more localization algorithms to test against our results to see if a more accurate algorithm for localizing users can be found. Finally, iBeacon localization should be compared to other common indoor localization methods, such as localization using Wifi, which will provide a better idea of how localization using iBeacon performs compared to existing indoor localization technologies.

With future research, we will get a greater understanding of the capabilities of iBeacon technology for indoor navigation and proximity awareness and perhaps finally find a reliable way to accurately localize users indoors, help them find the locations they need to go, and serve contextual information to them about their surroundings. This technology could be used for a number of applications, such as enhancing the experience of museum tours and serving advertisements to potential customers in real-time based on their location. In this way, iBeacon can form a foundation for the smart building technologies of the future that will help us navigate indoors and get better information about the many interesting things around us.

Works Cited

[1] Pahlavan, Kaveh. ECE 3308. Class Lecture, Topic: "Evolution of IoT and 5G." Worcester Polytechnic Institute, October 10, 2016.

[2] "Estimote." Internet: http://estimote.com/, 2016, [Oct.5, 2016].

[3] "CWINS." Internet: http://www.cwins.wpi.edu/, [Oct. 5, 2016].

[4] Z. Li, Y. Yang, and K. Pahlavan. "Using iBeacon for Newborns Localization in Hospitals." Internet: http://www.cwins.wpi.edu/publications/conference%20paper/ZhouchiISMICT.pdf. CWINS Lab, Worcester Polytechnic Institute.

[5] Z. Li, Y. Yang, and K. Pahlavan. "Using iBeacon for Intelligent In-Room Presence Detection." Internet: http://www.cwins.wpi.edu/publications/conference%20paper/Sushi.pdf. CWINS Lab, Worcester Polytechnic Institute.

[6] I. Guvenc and C. Chong, "A survey on TOA based wireless localization and NLOS mitigation techniques," IEEE Communications Surveys & Tutorials, vol. 11, no. 3, pp. 107–124, Mar. 2009.

[7] N. Yousef, A. H. Sayed, and L. Jalloul, "Robust wireless location over fading channels," IEEE Trans. on Vehicular Technology, vol. 52, no. 1, pp. 117–126, Jan. 2003.

[8] X. Li, "Collaborative localization with received-signal strength in wireless sensor networks," IEEE Trans. on Vehicular Technology, vol. 56, no. 6, pp. 3807–3817, June 2007.

[9] Pahlavan, Kaveh, Krishnamurthy, and Prashant. *Principles of Wireless Networks*. N.p.: John Wiley & Sons, 2013. Print.

[10] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," IEEE Trans. on Signal Processing, vol. 56, no. 7, pp. 3122–3136, July 2008.

[11] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network based wireless location: Challenges faced in developing techniques for accurate wireless location information," IEEE Signal Proc. Magazine, vol. 22, no. 4, pp. 24–40, April 2005.

[12] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," IEEE Trans. on Signal Processing, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[13] Pablo Cotera, Miguel Velazquez, David Cruz, Luis Medina and Manuel Bandala. " *Indoor Robot Positioning Using an Enhanced Trilateration Algorithm" - Jun 02, 2016*, 2 June 2016, journals.sagepub.com/doi/full/10.5772/63246. Accessed 20 Feb. 2017.

[14] Y. Yang. “Application Performance Evaluation for iBeacon In-Room Localization Technology Using CRLB.” Internet: http://www.cwins.wpi.edu/publications/Thesis/MS%20Thesis/Yang%20Yang.pdf. Worcester Polytechnic Institute.

[15] “OnePlus One.” Internet: http://www.gsmarena.com/oneplus_one-6327.php, [Sept. 20, 2016].

[16] “Android Studio.” Internet: https://developer.android.com/studio/index.html, [Sept. 20, 2016].

[17] “Create the Future with Java.” Internet: https://www.oracle.com/java/index.html, [Sept. 20, 2016].

[18] “GitHub.” Internet: https://github.com/, 2016, [Sept. 15, 2016].

[19] Pahlavan, Kaveh. ECE 3308. Class Lecture, Topic: “Characteristics of Wireless Medium.” Worcester Polytechnic Institute, August 29, 2016.

[20] A. Pal, “Localization algorithms in wireless sensor networks: Current approaches and future challenges,” Network Protocols and Algorithms, vol. 2, no. 1, pp. 45–73, Jan. 2010.

[21] Ali H. Sayed, Benoit Champagne, Stephan Saur and Reza Abdolee, "DIFFUSION LMS LOCALIZATION AND TRACKING ALGORITHM FOR WIRELESS CELLULAR NETWORKS." *Ece.mcgill.ca*. N.p., 2013. Web. 10 Dec. 2106. <http://www.ece.mcgill.ca/~bchamp/Papers/Conference/ICASSP2013c.pdf>.

[22] Pahlavan, Kevah. ECE 5307. Class Lecture, Topic: “Performance of RSS Based Localization Systems.” Worcester Polytechnic Institute, February 2017.

Appendices

Appendix A: Notification Code

```
//Notification toggling code is included in here
@Override
protected void onResume() {
   super.onResume();

   SystemRequirementsChecker.checkWithDefaultDialogs(this);

   //Button to toggle notifications
   final Button notifsOff = (Button)findViewById(R.id.notifsOff);
   notifsOff.setX(500);
   notifsOff.setY(1200);
   notifsOff.setBackgroundColor(0xFFFF0000);
   notifsOff.setOnClickListener(new View.OnClickListener() {
       @Override
       public void onClick(View v) {
           beaconNotificationsEnabled = !beaconNotificationsEnabled;
           if(beaconNotificationsEnabled){
               notifsOff.setText("Notifications Off");
               notifsOff.setBackgroundColor(0xFFFF0000);
           }
           else {
               notifsOff.setText("Notifications On");
               notifsOff.setBackgroundColor(0xFF00FF00);
           }
       }
   });

   if (!isBeaconNotificationsEnabled()) {
       Log.d(TAG, "Enabling beacon notifications");
       enableBeaconNotifications();
   }
   beaconManager.connect(new BeaconManager.ServiceReadyCallback() {
       @Override
```

```
        public void onServiceReady() {
            beaconManager.startRanging(region);

        }
    });
}

@Override
protected void onPause() {
    beaconManager.stopRanging(region);

    super.onPause();
}

//Creates android notifications when the user enters the range of beacons
public void enableBeaconNotifications() {
    if (beaconNotificationsEnabled) { return; }

    BeaconNotificationsManager beaconNotificationsManager = new
BeaconNotificationsManager(this);
    beaconNotificationsManager.addNotification(//Beetroot 3
            new BeaconID("6EE4D6A9-DD8E-550E-FF81-783E445F9C5B",48542,60126),
            "Room 320 is the CWINS lab",
            null);
    beaconNotificationsManager.startMonitoring();

    beaconNotificationsEnabled = true;
}

public boolean isBeaconNotificationsEnabled() {
    return beaconNotificationsEnabled;
}
```

Appendix B: RSSI and Distance Measurement Application Code

```
public class MainActivity extends AppCompatActivity {
   private static final String TAG = "MainActivity";
   private static final Map<Color, Integer> BACKGROUND_COLORS = new HashMap<>();
   static {
       BACKGROUND_COLORS.put(Color.ICY_MARSHMALLOW, android.graphics.Color.rgb(109,
170, 199));
       BACKGROUND_COLORS.put(Color.BLUEBERRY_PIE, android.graphics.Color.rgb(98, 84,
158));
       BACKGROUND_COLORS.put(Color.MINT_COCKTAIL, android.graphics.Color.rgb(155, 186,
160));
   }
   private static final int BACKGROUND_COLOR_NEUTRAL = android.graphics.Color.rgb(160,
169, 172);
   private String placesNearBeacon(Beacon beacon) {
       String rssi = String.format("%d", beacon.getRssi());
       String res= String.format("Distance:%s, RSSI:%s",
Utils.computeAccuracy(beacon),rssi);
       return res;
   }
   private BeaconManager beaconManager;
   private Region region;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);
       beaconManager = new BeaconManager(this);
       beaconManager.setRangingListener(new BeaconManager.RangingListener() {
           @Override
           public void onBeaconsDiscovered(Region region, List<Beacon> list) {
               String text;
               Integer backgroundColor;
               if (!list.isEmpty()) {
                   Beacon nearestBeacon = list.get(0);
                   String rssiVals = placesNearBeacon(nearestBeacon);

                   text = (rssiVals);
                   backgroundColor = null;
               } else {
                   text = "No beacons in range.";
                   backgroundColor = null;
               }
               ((TextView) findViewById(R.id.textView)).setText(text);
               findViewById(R.id.relativeLayout).setBackgroundColor(
                       backgroundColor != null ? backgroundColor :
BACKGROUND_COLOR_NEUTRAL);
           }
```

```
        });
        region = new Region("ranged region",
UUID.fromString("B9407F30-F5F8-466E-AFF9-25556B57FE6D"), 61665, 42057);
    }
@Override
    protected void onResume() {
        super.onResume();

        SystemRequirementsChecker.checkWithDefaultDialogs(this);

        beaconManager.connect(new BeaconManager.ServiceReadyCallback() {
            @Override
            public void onServiceReady() {
                beaconManager.startRanging(region);
            }
        });
    }
    @Override
    protected void onPause() {
        beaconManager.stopRanging(region);

        super.onPause();
    }
}
```

Appendix C: LMS Code

C-1. Matlab Version:

```
function [final_x,final_y] = ch12_p2(known_references,initial_guess,distances)
known_references = [10,10;0,15;-5,5];
initial_guess = [2,2];
distances = [15,16,5];
if size(known_references,2) ~= 2
error('location of known reference points should be entered as Nx2 matrix');
end
figure(1);
hold on
grid on
i=1;
temp_location(i,:) = initial_guess ;
temp_error = 0 ;
for j = 1 : size(known_references,1)
temp_error = temp_error + abs((known_references(j,1) - temp_location(i,1))^2 +
(known_references(j,2) - temp_location(i,2))^2 - distances(j)^2) ;
end
estimated_error = temp_error ;
% new_matrix = [ ];
while norm(estimated_error) > 1e-2 %iterative process for LS algorithm
for j = 1 : size(known_references,1) %Jacobian has been calculated in advance
jacobian_matrix(j,:) = -2*(known_references(j,:) - temp_location(i,:)) ; %partial
derivative is i.e. -2(x_1-x)
f(j) = (known_references(j,1) - temp_location(i,1))^2 + (known_references(j,2) -
temp_location(i,2))^2 - distances(j)^2 ;
end
estimated_error = -inv(jacobian_matrix' * jacobian_matrix) * (jacobian_matrix') * f' ;
%update the U and E
temp_location(i+1,:) = temp_location(i,:) + estimated_error' ;
% current_point = [temp_location(i+1,1),temp_location(i+1,2)];
% new_matrix = [ new_matrix; current_point];
plot(temp_location(i+1,1),temp_location(i+1,2),'rx') ; % plot
text(temp_location(i+1,1), temp_location(i+1,2)*(1 + 0.005) , num2str(i));
i = i + 1;
end
final_x = temp_location(i,1) ;
disp(final_x);
final_y = temp_location(i,2) ;
disp(final_y);
```

C- 2. Java Version:

```
/**
* Created by qusai on 2/1/17.
*/

public class Beacon {
   //instance variables
   private int x;
   private int y;

   public Beacon(int x, int y) {
       this.x = x;
       this.y = y;
   }

   public int getX() {
       return this.x;
   }

   public int getY() {
       return this.y;
   }
}
```

//Main Class

```
import org.ejml.ops.CommonOps;
import org.ejml.ops.NormOps;
import org.ejml.simple.*;
import org.ejml.ops.CommonOps.*;
import org.ejml.ops.NormOps.*;
import sun.java2d.pipe.SpanShapeRenderer;
```

```
public class Main {

   public static void main(String[] args) {

        //array of Beacon objects
        Beacon[] beacons = new Beacon[3]; //set the size to 3 at first
        int numBeacons = beacons.length;

        //create Beacons
        Beacon beacon1 = new Beacon(10, 10);
        Beacon beacon2 = new Beacon(0, 15);
        Beacon beacon3 = new Beacon(-5, 5);

        //add Beacons to array
        beacons[0] = beacon1;
        beacons[1] = beacon2;
        beacons[2] = beacon3;

        //set intial guess
        double guessX = 2;
        double guessY = 2;
        double[][] station = new double[][]{{ guessX, guessY }};
        SimpleMatrix matStation = new SimpleMatrix(station);
        double[][] mo = new double[][]{{ -1.0, -1.0}};
        SimpleMatrix minusOne = new SimpleMatrix(mo);

        //calculate the estimation error
        double estimationError = 0;
        double[] distances = {15,16,5};
        for (int i=0; i<numBeacons; i++) {
            Beacon thisBeacon = beacons[i];
//            double d = getDistance(thisBeacon.getX(), thisBeacon.getY(), guessX,
guessY);
            double d = distances[i];
            double f = Math.abs(Math.pow(thisBeacon.getX() - guessX, 2) +
Math.pow(thisBeacon.getY() - guessY, 2) - Math.pow(d, 2));
            estimationError = estimationError + f;
        }

        //create a Jacobian matrix of size [number_of_beacons][2]
        double[][] jacobianMatrix = new double[numBeacons][2];
        double[][] matF = new double[numBeacons][1];
        while (estimationError > 0.01){
            //for loop happens here
            //the condition for, for loop ->
            for (int i=0; i<numBeacons; i++){ //3 is the number of beacons
                //we calculate the jacobian matrix here
                Beacon b = beacons[i];
                for (int j=0; j<2; j++){
                    if (j==0){
                        jacobianMatrix[i][j] = -2*(b.getX()-guessX);
                    }
                    else {
                        jacobianMatrix[i][j] = -2 * (b.getY() - guessY);
                    }
                }
                matF[i][0] = Math.pow(b.getX() - guessX, 2) + Math.pow(b.getY() -
guessY, 2) - Math.pow(distances[i], 2);
            }
            SimpleMatrix matrixJacobian = new SimpleMatrix(jacobianMatrix);
            SimpleMatrix matrixF = new SimpleMatrix(matF);
            //here goes the matrix inverse operation
            // estimationError = -inv(jacobianMatrix' * jacobianMatrix) *
(jacobianMatrix') * F'
            SimpleMatrix first =
(matrixJacobian.transpose().mult(matrixJacobian)).invert();
            SimpleMatrix second = (matrixJacobian.transpose().mult(matrixF));
            SimpleMatrix matrixError = first.mult(second);

            matrixError = matrixError.negative();

            matStation = matStation.plus(matrixError.transpose());

            System.out.println(matStation);

            estimationError = matrixError.elementSum();

        }

    }

    //coordinate 1: x1, y1; coordinate 2: x2, y2

    public static double getDistance(int x1, int y1, int x2, int y2) {

        double d = Math.sqrt(Math.pow(x2-x1, 2) + Math.pow(y2-y1, 2));

        return d;

    }

}
```

Appendix D: Trilateration Matlab Code

```
function [lat, lon] = trilateration(DistA, DistB, DistC, xA, yA, xB, yB, xC, yC )

%
% Adapted from:
% https://www.mathworks.com/matlabcentral/fileexchange/57218-2d-trilateration/content/trilateration.m
% Copyright (c) 2016, Lionel Tailhardat
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
% met:
%
%     * Redistributions of source code must retain the above copyright
%       notice, this list of conditions and the following disclaimer.
%     * Redistributions in binary form must reproduce the above copyright
%       notice, this list of conditions and the following disclaimer in
%       the documentation and/or other materials provided with the distribution
%
% THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
% AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
% IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
% ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
% LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
% CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
% SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
% INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
% CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
% ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
% POSSIBILITY OF SUCH DAMAGE.

distA=122;
distB=32;
distC=32;
```

```
xA=443;
yA=0;
xB=376;
yB=70;
xC=313;
yC=70;

P1 = [xA; yA; 0];
P2 = [xB; yB; 0];
P3 = [xC; yC; 0];

%% Transformation
% from wikipedia https://en.wikipedia.org/wiki/Trilateration
% transform to get circle 1 at origin
% transform to get circle 2 on x axis
ex = (P2 - P1) / (norm(P2 - P1));
i  = dot(ex, (P3 - P1));
ey = (P3 - P1 - i*ex) / (norm(P3 - P1 - i*ex));
d  = norm(P2 - P1);
j  = dot(ey, (P3 - P1));

%Trianglulation Math
x = ((distA^2) - (distB^2) + (d^2))/(2*d);
y = (((distA^2) - (distC^2) + (i^2) + (j^2))/(2*j)) - ((i/j)*x);
% triPt is an array with ECEF x,y,z of trilateration point
triPt = P1 + x*ex + y*ey
end
```

Appendix E: GUI Code

//From the MainActivity class
```
public class MainActivity extends AppCompatActivity {

   private static final String TAG = "MainActivity";

   private Button userLocation;
   //distances to the beacons

   private double distance1,distance2,distance3;

   private BeaconManager beaconManager;
   private Region region;
   private boolean beaconNotificationsEnabled = false;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       //this.setTitle(R.string.instructions);

       setContentView(R.layout.activity_main);

       beaconManager = new BeaconManager(this);
       beaconManager.setRangingListener(new BeaconManager.RangingListener() {
           @Override
           public void onBeaconsDiscovered(Region region, List<Beacon> list) {
               String text;
               Beacon firstBeacon, secondBeacon, thirdBeacon;
               //Only one beacon in range
               if (list.size()==1) {
                   firstBeacon = list.get(0);
                   distance1 = Utils.computeAccuracy(firstBeacon);

                   text = (Integer.toString(firstBeacon.getMajor()) + ": " +
Double.toString((double)Math.round(distance1 * 100d) / 100d) + " Only 1 beacon in
range");
               }

               else if(list.size()==2){
                   //Only two beacons in range
                   firstBeacon = list.get(0);
                   distance1 = Utils.computeAccuracy(firstBeacon);

                   secondBeacon = list.get(1);
                   distance2 = Utils.computeAccuracy(secondBeacon);

                   text = (Integer.toString(firstBeacon.getMajor()) + ": " +
Double.toString((double)Math.round(distance1 * 100d) / 100d) + ", \n" +
                              Integer.toString(secondBeacon.getMajor()) + ": "
+Double.toString((double)Math.round(distance2 * 100d) / 100d) + " Only 2 beacons in
range.");
               }

               else if(list.size()>=3){

                    firstBeacon = list.get(0);
                    distance1 = Utils.computeAccuracy(firstBeacon);

                    secondBeacon = list.get(1);
                    distance2 = Utils.computeAccuracy(secondBeacon);

                    thirdBeacon = list.get(2);
                    distance3 = Utils.computeAccuracy(thirdBeacon);

                    //**************Start LMS
Functionality****************************
                    LMSBeacon[] beacons = new LMSBeacon[3]; //set the size to 3 at
first
                    beacons[0]=null;
                    beacons[1]=null;
                    beacons[2]=null;
                    int numBeacons = beacons.length;

                    //create Beacons
                    LMSBeacon beacon1 = new LMSBeacon(-45, 0);//Set this as candy with
major 20303
                    LMSBeacon beacon2 = new LMSBeacon(-40, -8);//Set this as beetroot
with 8897
                    LMSBeacon beacon3 = new LMSBeacon(-30, 0);//Set this as lemon with
61665
                    LMSBeacon beacon4 = new LMSBeacon(-40, -23);//Set this as candy2
with major 12070
                    LMSBeacon beacon5 = new LMSBeacon(-30, -30);//Set this as beetroot2
with 53500
                    LMSBeacon beacon6 = new LMSBeacon(-37, -15);//Set this as lemon2
with 34226
                    LMSBeacon beacon7 = new LMSBeacon(-53, -28);//Set this as candy3
with major 11911
                    LMSBeacon beacon8 = new LMSBeacon(-60, -30);//Set this as beetroot3
with 48542
                    LMSBeacon beacon9 = new LMSBeacon(-45, -30);//Set this as lemon3
with 22098

                    //array of LMSBeacon objects

                    //add Beacons to array, based on Major and Minor
                    //First closest Beacon
                    int firstMajor = firstBeacon.getMajor();
```

```
if(firstMajor== 20303)
    beacons[0] = beacon1;
else if(firstMajor== 8897)
    beacons[0] = beacon2;
else if(firstMajor== 61665)
    beacons[0] = beacon3;
else if(firstMajor== 12070) {
    beacons[0] = beacon4;
}
else if(firstMajor== 53500) {
    beacons[0] = beacon5;
}
else if(firstMajor== 34226) {
    beacons[0] = beacon6;
}
else if(firstMajor== 11911) {
    beacons[0] = beacon7;
}
else if(firstMajor== 48542) {
    beacons[0] = beacon8;
}
else if(firstMajor== 22098) {
    beacons[0] = beacon9;
}

//Second closest Beacon
int secondMajor = secondBeacon.getMajor();

if(secondMajor== 20303)
    beacons[1] = beacon1;
else if(secondMajor== 8897)
    beacons[1] = beacon2;
else if(secondMajor == 61665)
    beacons[1] = beacon3;
else if(secondMajor == 12070)
    beacons[1] = beacon4;
else if(secondMajor == 53500)
    beacons[1] = beacon5;
else if(secondMajor == 34226)
    beacons[1] = beacon6;
else if(secondMajor== 11911)
    beacons[1] = beacon7;
else if(secondMajor== 48542)
    beacons[1] = beacon8;
else if(secondMajor== 22098)
    beacons[1] = beacon9;

//Third closest beacon
int thirdMajor = thirdBeacon.getMajor();
```

```
                    if(thirdMajor== 20303)
                        beacons[2] = beacon1;
                    else if(thirdMajor== 8897)
                        beacons[2] = beacon2;
                    else if(thirdMajor== 61665)
                        beacons[2] = beacon3;
                    else if(thirdMajor == 12070)
                        beacons[2] = beacon4;
                    else if(thirdMajor== 53500)
                        beacons[2] = beacon5;
                    else if(thirdMajor== 34226)
                        beacons[2] = beacon6;
                    else if(thirdMajor== 11911)
                        beacons[2] = beacon7;
                    else if(thirdMajor== 48542)
                        beacons[2] = beacon8;
                    else if(thirdMajor== 22098)
                        beacons[2] = beacon9;



                        //set initial guess
                        double guessX = beacons[0].getX();
                        double guessY = beacons[0].getY();
                        double[][] station = new double[][]{{guessX, guessY}};
                        SimpleMatrix matStation = new SimpleMatrix(station);
                        double[][] mo = new double[][]{{-1.0, -1.0}};
                        SimpleMatrix minusOne = new SimpleMatrix(mo);

                        //calculate the estimation error
                        double estimationError = 0;
                        double[] distances = {distance1, distance2, distance3};
                        for (int i = 0; i < numBeacons; i++) {
                            LMSBeacon thisBeacon = beacons[i];
                            //double d = getDistance(thisBeacon.getX(),
thisBeacon.getY(), guessX, guessY);
                            double d = distances[i];
                            double f = abs(Math.pow(thisBeacon.getX() - guessX, 2) +
Math.pow(thisBeacon.getY() - guessY, 2) - Math.pow(d, 2));
                            estimationError = estimationError + f;
                        }

                        //create a Jacobian matrix of size [number_of_beacons][2]
                        double[][] jacobianMatrix = new double[numBeacons][2];
                        double[][] matF = new double[numBeacons][1];
                        while (estimationError > 0.01) {
                            //for loop happens here
                            //the condition for, for loop ->
                            for (int i = 0; i < numBeacons; i++) { //3 is the number of
beacons
                                //we calculate the jacobian matrix here
                                LMSBeacon b = beacons[i];
                                for (int j = 0; j < 2; j++) {
                                    if (j == 0) {
                                        jacobianMatrix[i][j] = -2 * (b.getX() -
guessX);
                                    } else {
                                        jacobianMatrix[i][j] = -2 * (b.getY() -
guessY);
                                    }
                                }
                                matF[i][0] = Math.pow(b.getX() - guessX, 2) +
Math.pow(b.getY() - guessY, 2) - Math.pow(distances[i], 2);
                            }
                            SimpleMatrix matrixJacobian = new
SimpleMatrix(jacobianMatrix);
                            SimpleMatrix matrixF = new SimpleMatrix(matF);
                            //here goes the matrix inverse operation
                            // estimationError = -inv(jacobianMatrix' * jacobianMatrix)
* (jacobianMatrix') * F'
                            SimpleMatrix first =
(matrixJacobian.transpose().mult(matrixJacobian)).invert();
                            SimpleMatrix second =
(matrixJacobian.transpose().mult(matrixF));
                            SimpleMatrix matrixError = first.mult(second);
                            matrixError = matrixError.negative();
                            matStation = matStation.plus(matrixError.transpose());
                            estimationError = matrixError.elementSum();
                            if((int)abs(matStation.get(0,0))>80 ||
(int)abs(matStation.get(0,1))>80)//This line avoids nonconvergence
                                break;

                        }
                    //******************End LMS
Functionality*******************************************
                        //Creates a UserPosition to convert map coordinates to a
position on map
                        UserPosition position = new
UserPosition((int)abs(matStation.get(0,0)),(int)abs(matStation.get(0,1)));

                        userLocation = (Button)findViewById(R.id.userLocation);
                        userLocation.setX(position.getXpos());
                        userLocation.setY(position.getYpos());

                        //Displays the coordinates given by algorithm
                        TextView distance = (TextView)findViewById(R.id.math);
                        distance.setText(Double.toString(matStation.get(0,0)) + " ,  "
+
                              Double.toString(matStation.get(0,1)));

                    //Displays the majors of three closest beacons, as well as the
distance to them
                    text = (Integer.toString(firstMajor)+ ": " +
Double.toString((double)Math.round(distance1 * 100d) / 100d)+ ", \n" +
                            Integer.toString(secondMajor)+ ": "
+Double.toString((double)Math.round(distance2 * 100d) / 100d)+ ", \n" +
                            Integer.toString(thirdMajor)+ ": "
+Double.toString((double)Math.round(distance3 * 100d) / 100d));
                    ((TextView) findViewById(R.id.textView)).setText(text);


                }

                else {
                    text = "No beacons in range.";
                }

                ((TextView) findViewById(R.id.textView)).setText(text);
                //((TextView)
findViewById(R.id.distances)).setText(Double.toString(distance1));

            }
        });
        region = new Region("ranged region",
UUID.fromString("6EE4D6A9-DD8E-550E-FF81-783E445F9C5B"), null, null);

        //Set the locations of each of the beacons, moving left to right and top to
bottom
        Button beacon1 = (Button)findViewById(R.id.beacon1);
        beacon1.setX(200);
        beacon1.setY(410);

        Button beacon2 = (Button)findViewById(R.id.beacon2);
        beacon2.setX(295);
        beacon2.setY(385);

        Button beacon3 = (Button)findViewById(R.id.beacon3);
        beacon3.setX(360);
        beacon3.setY(160);

        Button beacon4 = (Button)findViewById(R.id.beacon4);
        beacon4.setX(360);
        beacon4.setY(400);

        Button beacon5 = (Button)findViewById(R.id.beacon5);
        beacon5.setX(370);
        beacon5.setY(230);
```

```
Button beacon6 = (Button)findViewById(R.id.beacon6);
beacon6.setX(370);
beacon6.setY(320);

Button beacon7 = (Button)findViewById(R.id.beacon7);
beacon7.setX(420);
beacon7.setY(275);

Button beacon8 = (Button)findViewById(R.id.beacon8);
beacon8.setX(470);
beacon8.setY(170);

Button beacon9 = (Button)findViewById(R.id.beacon9);
beacon9.setX(470);
beacon9.setY(410);

Button beacon10 = (Button)findViewById(R.id.beacon10);
beacon10.setX(520);
beacon10.setY(190);

Button beacon11 = (Button)findViewById(R.id.beacon11);
beacon11.setX(520);
beacon11.setY(390);

Button beacon12 = (Button)findViewById(R.id.beacon12);
beacon12.setX(570);
beacon12.setY(170);

Button beacon13 = (Button)findViewById(R.id.beacon13);
beacon13.setX(570);
beacon13.setY(410);

Button beacon14 = (Button)findViewById(R.id.beacon14);
beacon14.setX(670);
beacon14.setY(230);

Button beacon15 = (Button)findViewById(R.id.beacon15);
beacon15.setX(670);
beacon15.setY(320);

Button beacon16 = (Button)findViewById(R.id.beacon16);
beacon16.setX(720);
beacon16.setY(170);

Button beacon17 = (Button)findViewById(R.id.beacon17);
beacon17.setX(720);
beacon17.setY(275);

Button beacon18 = (Button)findViewById(R.id.beacon18);
beacon18.setX(720);
```

```
        beacon18.setY(410);

        Button beacon19 = (Button)findViewById(R.id.beacon19);
        beacon19.setX(780);
        beacon19.setY(385);

        Button beacon20 = (Button)findViewById(R.id.beacon20);
        beacon20.setX(840);
        beacon20.setY(410);

        init();

    }

//Method to change screens when a button is clicked
private void init(){
    Button switchScreen = (Button)findViewById(R.id.switchScreen);
    switchScreen.setX(500);
    switchScreen.setY(900);
    switchScreen.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent nextScreen = new Intent(getApplicationContext(),
ProfessorTableActivity.class);
            startActivity(nextScreen);
        }
    });

}
```

//From the ProfessorTableActivity class

```
public class ProfessorTableActivity extends AppCompatActivity {


    private Button mapScreen;

    private void init(){
        mapScreen = (Button)findViewById(R.id.mapScreen);
        mapScreen.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent nextScreen = new Intent(ProfessorTableActivity.this,
MainActivity.class);
                startActivity(nextScreen);
            }
        });
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_professor_table);
        init();
    }
}
```

Appendix F: Point of Interest List

Professor Name	Room Number	URL	Text to Display
R. James Duckworth	301	https://www.wpi.edu/people/faculty/rjduck	Intrested in multiprocessing, parallel computation, logic synthesis, real time systems and rapid prototyping of computer system.
Berk Sunar	302	https://www.wpi.edu/people/faculty/sunar	Intrested in cryptography, network security and high performance computing.
Xinming Huang	303	https://www.wpi.edu/people/faculty/xhuang	Intrested in autonomous cars, integrated circuit design, cyber physical system and wirless communications.
Edward Clancy	304	https://www.wpi.edu/people/faculty/ted	Intrested in biomedical signal processing, modeling and instrumentation.
John McNeill	305	https://www.wpi.edu/people/faculty/mcneill	Intrested in analog microelectronics and high-speed imagining and mixed signal circuit characterization.

Sergey Makarov	306	https://www.wpi.edu/people/faculty/makarov	Intrested in antennas, applied electromagnetics and numerical methods.
Thomas Eisenbarth	307	https://www.wpi.edu/people/faculty/teisenbarth	Intrested in embaded systems security.
Kaveh Pahlavan	308	https://www.wpi.edu/people/faculty/kaveh	Intrested in body area networking, localization, indoor geolocation, WIFI localization, UWB localization and broadband and location aware wireless networks.
Jahangir Rahman	309	https://www.wpi.edu/people/faculty/jrahman	Intrested in signal processing, microelectronics, communication systems and analog and digital systems.
Shamsur Mazumder	310	https://www.wpi.edu/people/faculty/srmazumder	Intrested in simulation-based RF and microwave circuit designs and RF and microwave circuits for radar and wireless communication applications.
Faculty Conference Room	311	http://spinlab.wpi.edu/	Signal Processing & Information Netwiorking Labioratory (SPINLab).
Stephen Bitar	312	https://www.wpi.edu/people/faculty/sjbitar	Intrested in renewable energy, analog, power and atomotive electronics.

Donald Brown	313	https://www.wpi.edu/people/faculty/drb	Intrested in communication system and networking, signal processing, information theory, estimationdetection and linear nonlinear dynamical sysyems.
Signal Processing & Information Networking Laboratory (SPINLab)	314	http://spinlab.wpi.edu/	Interested in development and testing of an acoustic timeslotted round-trip carrier synchronization system, development and testing of a battery-free wireless tire pressure monitoring system and smartphone app and bicycle-based low-power lighting demonstration system projects.
Convergent Technology Laboratory and Precision Personal Locator (PPL)	315	????	
Classroom/Laboratory	317A		ECE Laboratory

New England Center for Analog & Mixed Signal IC Design (NECAMSID)	317B	http://ece.wpi.edu/analog/center.html	Interested in projects in mixed signal microelectronics typically involve design, fabrication, and test of application circuitry for mixed signal ICs.
Wireless Innovation Laboratory	318A	http://ecewp.ece.wpi.edu/wordpress/wireless/	WI Lab has had extensive experience working with industry and government sponsors on both fundamental and applied research projects, yielding successful outcomes that often exceed the expected value-add these projects bring to the sponsors.
Conference Room	318B		To book a room please ask the ECE office in the 2nd floor.
Zainalabedin Navabi	319	https://www.wpi.edu/people/faculty/navabi	Intrested in RTL design and test, methodologies for system level design, verilog design and synthesis and verilog elements of system design.

Center for Wireless Information Network System (CWINS)	320	http://www.cwins.wpi.edu/	Interested in body area network, indoor geolocation and wireless local area network projects.
Antenna Laboratory	320C	http://ece.wpi.edu/ant/	Directed by professor Sergey Makarov.

Appendix G: CRLB Analysis Matlab Code

%Note: This code was used to generate Figure ## CRLB Analysis for Partial Grid Deployment with 15m %Broadcast Range. Similar code was used to generate the other CRLB analyses. The only changes made %to generate other plots was the value of variables.

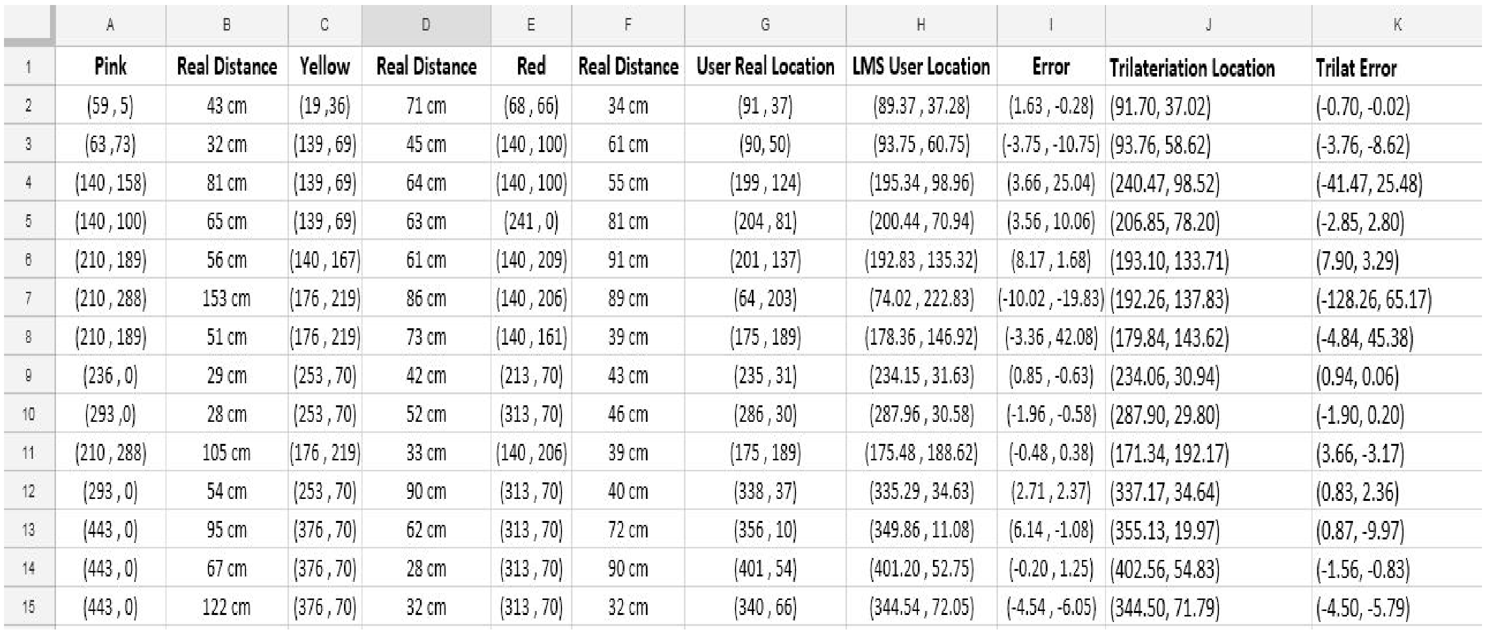
```
close all;clear all;clc;
%% Initialization
% Locations of Access Points
APx(1)=0;APy(1)=0;
APx(2)=15;APy(2)=0;
APx(3)=30;APy(3)=0;
APx(4)=0;APy(4)=15;
APx(5)=15;APy(5)=15;
APx(6)=30;APy(6)=15;
APx(7)=15;APy(7)=30;
APx(8)=30;APy(8)=30;
%APx(9)=45;APy(9)=0;
%APx(10)=45;APy(10)=15;
%APx(11)=45;APy(11)=30;
%APx(12)=60;APy(12)=0;
%APx(13)=60;APy(13)=15;
%APx(14)=60;APy(14)=30;
%APx(15)=75;APy(15)=0;
%APx(16)=75;APy(16)=15;

SD=9.6; % Standard Deviation of Shadow Fading
NUM=8; % Number of Access Points

% Locations of Receivers
pace=0.1;
mx=0:pace:30;
my=0:pace:30;

nxy=length(mx);
```

```
for yi=1:nxy
   for xi=1:nxy
      for i1=1:NUM
         alpha(i1)=2.5; % Power-Distance Gradient
         r(i1,xi,yi)=sqrt((mx(xi)-APx(i1))^2+(my(yi)-APy(i1))^2); % Distance Between Transmitter and
%Receiver
         H1(i1,xi,yi)=-10*alpha(i1)/log(10)*(mx(xi)-APx(i1))/(r(i1,xi,yi))^2; % First Column of H Matrix
         H2(i1,xi,yi)=-10*alpha(i1)/log(10)*(mx(yi)-APy(i1))/(r(i1,xi,yi))^2; % Second Column of H Matrix
      end
      H(:,:,xi,yi)=[H1(:,xi,yi),H2(:,xi,yi)];
      Covv(:,:,xi,yi)=SD^2*((H(:,:,xi,yi))'*H(:,:,xi,yi))^(-1); % Covariance Matrix of Error Estimate
      SDr(xi,yi)=sqrt(Covv(1,1,xi,yi)+Covv(2,2,xi,yi)); % Standard Deviation of Location Error
   end
end
SDr=SDr';
contourf(mx,my,SDr,40,'ShowText', 'On');
xlabel('X-axis(meters)');
ylabel('Y-axis(meters)');
title('Contour of Location Error Standard Deviation')
colorbar;%display plot legend
```

Appendix H: Coverage Analysis

			Transmission Power= 4 dBm	Transmission Power= 0 dBm	Transmission Power= -4 dBm	Transmission Power= -8 dBm	Transmission Power= -12 dBm	Transmission Power= -30 dBm
Percent Coverage	Fade Margin (dB)	Equation	coverage radius (m)	coverage radius (m)	coverage radius (m)	coverage radius (m)	coverage radius (m)	coverage radius (m)
50% (0.5)	0	Lp= 55+ 25log(d)	47.86	33.113	22.909	15.85	10.965	2.089
70% (0.699)	5	Lp= 55+ 25log(d) + 5	30.2	20.893	14.454	10	6.918	1.318
80% (0.798)	8	Lp= 55+ 25log(d) + 8	22.91	15.849	10.965	7.586	5.248	1
90% (0.9003)	12.3	Lp= 55+ 25log(d) + 12.3	15.42	10.666	7.379	5.105	3.532	0.673
95% (0.9503)	15.8	Lp= 55+ 25log(d) + 15.8	11.169	7.727	5.346	3.698	2.559	0.511
99% (0.99)	22.3	Lp= 55+ 25log(d) + 22.3	6.138	4.246	2.938	2.032	1.406	0.268

Appendix I: LMS and Trilateration Performance Comparison



	A	B	C	D	E	F	G	H	I	J	K
1	**Pink**	**Real Distance**	**Yellow**	**Real Distance**	**Red**	**Real Distance**	**User Real Location**	**LMS User Location**	**Error**	**Trilateriation Location**	**Trilat Error**
2	(59 , 5)	43 cm	(19 ,36)	71 cm	(68 , 66)	34 cm	(91 , 37)	(89.37 , 37.28)	(1.63 , -0.28)	(91.70, 37.02)	(-0.70, -0.02)
3	(63 ,73)	32 cm	(139 , 69)	45 cm	(140 , 100)	61 cm	(90, 50)	(93.75 , 60.75)	(-3.75 , -10.75)	(93.76, 58.62)	(-3.76, -8.62)
4	(140 , 158)	81 cm	(139 , 69)	64 cm	(140 , 100)	55 cm	(199 , 124)	(195.34 , 98.96)	(3.66 , 25.04)	(240.47, 98.52)	(-41.47, 25.48)
5	(140 , 100)	65 cm	(139 , 69)	63 cm	(241 , 0)	81 cm	(204 , 81)	(200.44 , 70.94)	(3.56 , 10.06)	(206.85, 78.20)	(-2.85, 2.80)
6	(210 , 189)	56 cm	(140 , 167)	61 cm	(140 , 209)	91 cm	(201 , 137)	(192.83 , 135.32)	(8.17 , 1.68)	(193.10, 133.71)	(7.90, 3.29)
7	(210 , 288)	153 cm	(176 , 219)	86 cm	(140 , 206)	89 cm	(64 , 203)	(74.02 , 222.83)	(-10.02 , -19.83)	(192.26, 137.83)	(-128.26, 65.17)
8	(210 , 189)	51 cm	(176 , 219)	73 cm	(140 , 161)	39 cm	(175 , 189)	(178.36 , 146.92)	(-3.36 , 42.08)	(179.84, 143.62)	(-4.84, 45.38)
9	(236 , 0)	29 cm	(253 , 70)	42 cm	(213 , 70)	43 cm	(235 , 31)	(234.15 , 31.63)	(0.85 , -0.63)	(234.06, 30.94)	(0.94, 0.06)
10	(293 ,0)	28 cm	(253 , 70)	52 cm	(313 , 70)	46 cm	(286 , 30)	(287.96 , 30.58)	(-1.96 , -0.58)	(287.90, 29.80)	(-1.90, 0.20)
11	(210 , 288)	105 cm	(176 , 219)	33 cm	(140 , 206)	39 cm	(175 , 189)	(175.48 , 188.62)	(-0.48 , 0.38)	(171.34, 192.17)	(3.66, -3.17)
12	(293 , 0)	54 cm	(253 , 70)	90 cm	(313 , 70)	40 cm	(338 , 37)	(335.29 , 34.63)	(2.71 , 2.37)	(337.17, 34.64)	(0.83, 2.36)
13	(443 , 0)	95 cm	(376 , 70)	62 cm	(313 , 70)	72 cm	(356 , 10)	(349.86 , 11.08)	(6.14 , -1.08)	(355.13, 19.97)	(0.87, -9.97)
14	(443 , 0)	67 cm	(376 , 70)	28 cm	(313 , 70)	90 cm	(401 , 54)	(401.20 , 52.75)	(-0.20 , 1.25)	(402.56, 54.83)	(-1.56, -0.83)
15	(443 , 0)	122 cm	(376 , 70)	32 cm	(313 , 70)	32 cm	(340 , 66)	(344.54 , 72.05)	(-4.54 , -6.05)	(344.50, 71.79)	(-4.50, -5.79)

Appendix J: Application Performance Evaluation Test Table

Phone x	Phone y	Real x	Real y	X Error	Y Error
-53.42	-39.79	-53	-28.83	0.42	10.96
-53.15	-40.44	-57	-30	3.85	10.44
-52.72	-41.41	-55	-30	2.28	11.41
-52.72	-40.05	-52.5	-29	0.22	11.05
-52.62	-48.32	-51.5	-29	1.12	19.32
-52.68	-44.62	-52	-30	0.68	14.62
-52.55	-44.47	-46	-28	6.55	16.47
-52.58	-44.62	-44	-29	8.58	15.62
-52.68	-31.45	-57	-28	4.32	3.45
-50.9	-38.25	-53	-29	2.1	9.25
-51.95	-33.31	-51.5	-29.5	0.45	3.81
-52.47	-42.74	-56	-30	3.53	12.74
-53.81	-39.75	-58	-30	4.19	9.75
-51.94	-40	-50	-28.5	1.94	11.5
-52.19	-40.11	-45	-27.5	7.19	12.61
-53.72	-48.54	-56	-29.5	2.28	19.04
-46.75	-32.01	-48	-26	1.25	6.01
-51.85	-56.29	-51	-28	0.85	28.29
-52.28	-51.26	-43	-29	9.28	22.26
-54.37	-40.01	-51	-28.5	3.37	11.51
-59.67	-11.62	-41	-23	18.67	11.38
-60.32	-10.97	-40	-25	20.32	14.03

-65.08	-8.88	-38	-16	27.08	7.12
-42.99	-15.99	-39	-14	3.99	1.99
-43.44	-15.99	-39	-10	4.44	5.99
-49.87	-16.04	-37	-5	12.87	11.04
-37.86	-10.58	-40	0	2.14	10.58
-31.29	-8.11	-36	-2	4.71	6.11
-37.14	-0.51	-35	0	2.14	0.51
-29.45	-1.8	-32	-1.5	2.55	0.3
-36.69	-0.53	-40	-5	3.31	4.47
-48.96	-14.42	-39	-8	9.96	6.42
-38.45	-17.31	-39.5	-17.5	1.05	0.19
-65.35	-9.09	-39	-24	26.35	14.91
-72.36	-6.23	-42	-28.5	30.36	22.27
-38.59	-28.99	-38	-26.5	0.59	2.49
-63.74	-15.29	-40	-20	23.74	4.71
-47.48	-16.07	-37	-17	10.48	0.93
-46.16	-15.71	-37	-13	9.16	2.71
-56.58	-14.07	-39	-2	17.58	12.07
-36.76	-5.06	-35	-1	1.76	4.06
-37.14	-1.45	-29	-1	8.14	0.45
-36.69	-1.21	-34	-2	2.69	0.79
-37.28	-4.51	-40	-2	2.72	2.51
-57.55	-15.33	-39	-8.5	18.55	6.83
-46.49	-15	-38	-15	8.49	0
-67.28	-8.33	-39	-27	28.28	18.67
-62.98	-12.3	-37	-21	25.98	8.7
-42.94	-16.75	-40	-13	2.94	3.75

-57.23	-14.58	-38	-8	19.23	6.58
-50.84	-13.02	-39	-2	11.84	11.02
-27.19	-9.13	-32	0	4.81	9.13
-38.04	-8.87	-37	-3	1.04	5.87
-36.61	1.12	-37	-1	0.39	2.12
-53.17	-14.97	-40	-9	13.17	5.97
-64.01	-9.37	-39	-19	25.01	9.63
-65.52	-9.14	-38	-29	27.52	19.86
-53.7	-39.51	-38	-29	15.7	10.51
-37.13	-30.38	-35	-28	2.13	2.38
-36.85	-31.74	-33	-28	3.85	3.74

Appendix K Full Path Loss Testing Dataset

RSSI (dBm)	Distance (in)	Distance (m)	Path Loss
-87	20	0.508	-57
-92	26	0.6604	-62
-95	35	0.889	-65
-95	42	1.0668	-65
-96	48	1.2192	-66
-94	61	1.5494	-64
-92	65	1.651	-62
-101	70	1.778	-71
-100	82	2.0828	-70
-95	100	2.54	-65
-94	106	2.6924	-64
-96	113	2.8702	-66
-100	127	3.2258	-70
-75	30	0.762	-63
-67	35	0.889	-55
-64	37	0.9398	-52
-71	37	0.9398	-59
-74	39	0.9906	-62
-73	40	1.016	-61
-68	42	1.0668	-56
-60	45	1.143	-48
-69	45	1.143	-57
-70	46	1.1684	-58
-74	47	1.1938	-62
-75	55	1.397	-63
-76	63	1.6002	-64
-75	67	1.7018	-63
-80	70	1.778	-68
-86	72	1.8288	-74
-81	75	1.905	-69
-76	79	2.0066	-64
-78	85	2.159	-66
-87	85	2.159	-75
-83	93	2.3622	-71

-78	95	2.413	-66
-81	102	2.5908	-69
-89	102	2.5908	-77
-83	110	2.794	-71
-89	110	2.794	-77
-88	115	2.921	-76
-79	120	3.048	-67
-88	120	3.048	-76
-94	121	3.0734	-82
-95	127	3.2258	-83
-96	128	3.2512	-84
-82	130	3.302	-70
-84	132	3.3528	-72
-88	133	3.3782	-76
-79	140	3.556	-67
-88	140	3.556	-76
-81	142	3.6068	-69
-86	150	3.81	-74
-90	152	3.8608	-78
-87	160	4.064	-75
-90	167	4.2418	-78
-85	170	4.318	-73
-89	176	4.4704	-77
-81	180	4.572	-69
-95	180	4.572	-83
-98	185	4.699	-86
-87	190	4.826	-75
-101	190	4.826	-89
-86	196	4.9784	-74
-90	205	5.207	-78
-86	217	5.5118	-74
-85	232	5.8928	-73
-84	238	6.0452	-72
-90	240	6.096	-78
-88	250	6.35	-76
-98	256	6.5024	-86
-87	262	6.6548	-75
-91	265	6.731	-79
-97	275	6.985	-85

-92	290	7.366	-80
-93	300	7.62	-81
-68	23	0.5842	-60
-70	27	0.6858	-62
-66	28	0.7112	-58
-64	29	0.7366	-56
-66	31	0.7874	-58
-66	35	0.889	-58
-68	38	0.9652	-60
-67	38	0.9652	-59
-68	42	1.0668	-60
-65	42	1.0668	-57
-69	47	1.1938	-61
-74	52	1.3208	-66
-75	56	1.4224	-67
-69	59	1.4986	-61
-77	60	1.524	-69
-75	62	1.5748	-67
-79	65	1.651	-71
-69	71	1.8034	-61
-70	72	1.8288	-62
-77	76	1.9304	-69
-76	78	1.9812	-68
-82	79	2.0066	-74
-75	80	2.032	-67
-71	94	2.3876	-63
-76	95	2.413	-68
-54	20	0.508	-50
-63	34	0.8636	-59
-67	35	0.889	-63
-68	37	0.9398	-64
-70	40	1.016	-66
-71	41	1.0414	-67
-67	48	1.2192	-63
-75	51	1.2954	-71
-73	70	1.778	-69
-77	72	1.8288	-73
-77	80	2.032	-73
-75	85	2.159	-71

-76	93	2.3622	-72
-78	95	2.413	-74
-75	108	2.7432	-71
-79	121	3.0734	-75
-87	125	3.175	-83
-85	128	3.2512	-81
-74	140	3.556	-70
-80	145	3.683	-76
-82	150	3.81	-78
-84	165	4.191	-80
-80	166	4.2164	-76
-78	167	4.2418	-74
-81	190	4.826	-77
-73	205	5.207	-69
-84	220	5.588	-80
-95	221	5.6134	-91
-76	222	5.6388	-72
-83	225	5.715	-79
-79	235	5.969	-75
-82	236	5.9944	-78
-82	248	6.2992	-78
-74	250	6.35	-70
-88	263	6.6802	-84
-87	264	6.7056	-83
-84	280	7.112	-80
-94	290	7.366	-90
-81	291	7.3914	-77
-82	296	7.5184	-78
-79	300	7.62	-75
-80	301	7.6454	-76
-84	311	7.8994	-80
-87	318	8.0772	-83
-95	333	8.4582	-91
-79	335	8.509	-75
-83	345	8.763	-79
-85	355	9.017	-81
-87	371	9.4234	-83
-81	384	9.7536	-77
-92	385	9.779	-88

-89	394	10.0076	-85
-87	407	10.3378	-83
-90	413	10.4902	-86
-94	414	10.5156	-90
-92	415	10.541	-88
-83	415	10.541	-79
-99	420	10.668	-95
-87	421	10.6934	-83
-89	425	10.795	-85
-91	435	11.049	-87
-84	445	11.303	-80
-86	457	11.6078	-82
-92	458	11.6332	-88
-92	471	11.9634	-88
-83	472	11.9888	-79
-90	484	12.2936	-86
-91	489	12.4206	-87
-94	490	12.446	-90
-90	497	12.6238	-86