iBeacon Applications and Hybrid Wi-Fi Localization

A Major Qualifying Project Report

Submitted to the faculty of

Worcester Polytechnic Institute

In partial fulfillment of the requirements

for the Degree of Bachelor of Science

By:

Yi Sun

Olawole Tunde-Lukan

Teng Weng

Project Advisor: *Professor* Kaveh Pahlavan

Abstract

Apple introduced its new product, iBeacon, to the world in 2013. iBeacon allows you to do indoor geolocation and has features that essential to application development. Since 2007, Wi-Fi localization has become the most popular indoor geolocation technology after used with iPhone. The purpose of this project was to develop an iBeacon application and also use iBeacon's signals to improve the accuracy of Wi-Fi's localization. For the application, we developed an algorithm that allows us to count the number of people that are in a room and broadcast this information based on proximity. While the iBeacons are set up in place for this application, we also integrated it with Wi-Fi to improve the accuracy of Wi-Fi geolocation. We achieved this by developing a new algorithm called path-loss based nearest neighbor.

Acknowledgement

We wish to acknowledge the guidance of Prof. Pahlavan, supportive help of PhD student Yishuang Geng, Master student Julang Ying, Zhouchi Li, Yang Yang, Undergraduate student Hui Zheng, Tianchang Gu, Fangming Ning, Zilu Tian and all the members of CWINS Labs for their cooperation during this project.

Table of Contents

Abstract 1
Acknowledgement 2
Chapter 1: Introduction 5
1.1 Introduction 5
1.2 Motivation of the project 5
1.3 Project Description 5
1.4 Paper Description 7
Chapter 2: Background for Indoor Geolocation 8
2.1 Wi-Fi Localization 8
2.2 RFID Introduction and Development 8
2.3 Development History of iBeacon and the Features of iBeacon 9
2.4 Estimote Development Kit for iBeacon 11
Chapter 3: Methodology 13
3.1 Data Collection and Scenario Development 13
3.1.1 Scenario Development 13
3.1.2 Data Collection 14
3.2 Data Analysis 17
3.2.1 Empirical Model Development 17
3.2.3 Regenerate the Database Using 802.11 Channel Model 18
3.3 Nearest Neighbor Algorithm 19
3.3.1 RSSI Based Nearest Neighbor 20
3.3.2 Distance Based Nearest Neighbor 21
3.4 Cramér–Rao Lower Bound 22
3.4.1 Single Observation 22
3.4.2 Multiple Same Observation 22
3.4.3 Hybrid Observation 23
3.5 Presence Detection Application using iBeacon and Hybrid Wi-Fi with iBeacon Localization Algorithm Development 24
3.5.1 Presence Detection Algorithm Development 24
3.5.2 Hybrid Wi-Fi and iBeacon Localization Algorithm 27
Chapter 4: Results 29
4.1 Applications of iBeacon 29
4.1.1 Application of Broadcasting and Distance Estimation 29

4.1.2 Application on iPhone to Count the Number of People in the Room.......................... 31
4.2 Wi-Fi Localization Results 32
4.2.1 Empirical Channel Model Result.. 32
4.2.2 Wi-Fi Localization Results between Generated Database and Empirical Database.... 36
4.3 Cramér–Rao Lower Bound 38
4.3.1 Cramér–Rao Lower Bound of RSS and Distance Based Preliminary Results 38
4.3.2 CRLB for Single iBeacon 39
4.3.3 CRLB for Hybrid Localization 40
4.4 iBeacon Configuration 42
4.4.1 Percentage Coverage Results 42
4.5 Results of Hybrid Localization of Wi-Fi and iBeacon 44
Chapter 5: Conclusion and Future Work 46
5.1 Conclusion 46
5.2 Future Work 46
Reference 47
Appendices... 49
Appendix A: MATLAB Code for CRLB Calculation 49
Appendix B: MATLAB Code to Generate Distance Based Database and Calculate the DME and CDF 50
Appendix C: MATLAB Code to Count Probability of Coverage for iBeacon......................... 53
Appendix D: Code for the application for iBeacon 54
AppDelegate.m file 54
ViewController.m file 58

Chapter 1: Introduction

1.1 Introduction

While outdoor localization is already very popular due to Global Positioning System (GPS), indoor localization has been gaining a lot of attention because GPS is inadequate indoors. Many companies have shown interest in indoor localization and began to try to implement it for their own applications, such as Google and Apple. These companies are attempting to install Wi-Fi access points inside the desired buildings. Wi-Fi localization is one of the more popular techniques for indoor localization; some of the other techniques that can be used are Radio Frequency Identification (RFID) and iBeacon. Using just Wi-Fi doesn't provide a good enough accuracy to determine the location of a person due to several reasons, one being multipath fading. However the accuracy can be increased when Wi-Fi is complemented by another technique, such as RFID or iBeacon.

1.2 Motivation of the project

While in an outdoor environment, we are able to use GPS as a source of navigation. As of now, there aren't too many applications for indoor navigation. With the increase in popularity in the new technology we decided to include it in our project. Wi-Fi localization alone wouldn't give us the accuracy that we want. With the addition of Apple's new product, iBeacon, we will be able to improve the accuracy. iBeacon also offers intelligence localization, which broadcasts a message to a device based on the location of a person.

1.3 Project Description

In this project, our goal is to show that it is possible to develop an indoor navigating system using Wi-Fi localization combined with iBeacon. iBeacon allows you to broadcast information to a nearby smart device and also offers some localization. The coverage of iBeacons wouldn't be accurate enough unless a large number of iBeacons were set up in the proper orientation. We plan on using Wi-Fi for our main localization technique because it provides better accuracy and is able to cover a larger area than the iBeacon. The project was separated into two main parts, iBeacon application development and hybrid localization. In the iBeacon application part of the project, we spent time working on the proximity broadcasting, in room presence detection, and a distance estimation. The proximity broadcasting broadcasts a message to the smart phone when in the range of the iBeacon. The message that gets broadcasted is relevant information about the professor's office. The in room presence detection allows us to count the number of people that are currently in the office and the distance estimation gives the user an update on their distance from the iBeacon. The second part of the project, hybrid localization, consisted of analyzing our localization technique. We analyzed our performance using Cramér–Rao lower bound to simulate the results in our scenario. We also applied a path-loss based nearest neighbor algorithm to our hybrid localization to improve the accuracy of Wi-Fi localization.

Figure 1.3.1: Example of Proximity Interaction with iBeacon

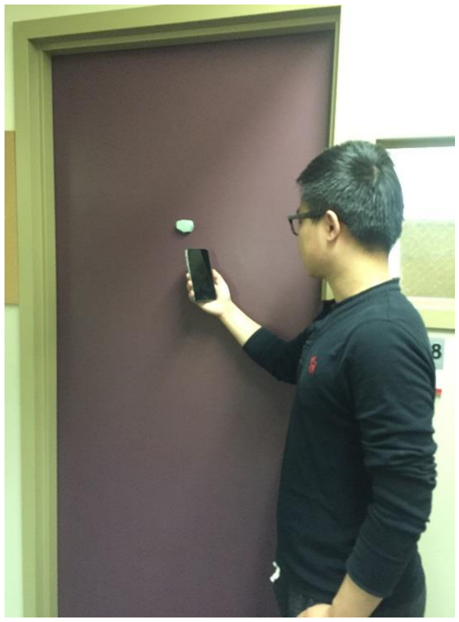


Figure 1.3.1: Scenario of using iBeacon Application to Receive Information

1.4 Paper Description

In the rest of this report we will touch on background of localization, methodology, results, and conclusion and future work of our project. Chapter 2 gives background information on the different localization technologies that were used and relevant to our project. Chapter 3 steps through the methodology used in order to achieve the goal of developing an application for iBeacon and improving Wi-Fi’s geolocation by integrating it with iBeacon. Chapter 4 discusses the results that were obtained from the testing and simulations. Chapter 5 concludes the paper by giving a general review of what our results were and also what that the next step would be to continue the project.

Chapter 2: Background for Indoor Geolocation

This chapter provides the necessary background information of indoor geolocation of Wi-Fi, Radio Frequency Identifier (RFID) and iBeacon. It also offers information on each technology, including its advantages and disadvantages at indoor geolocation in order to help developer to get a better understanding in their abilities and usage. Section 2.1 describes Wi-Fi localization, section 2.2 defines RFID, section 2.3 talks about iBeacon and section 2.4 discusses the estimate development kit for iBeacon.

2.1 Wi-Fi Localization

Wi-Fi localization is used for localization in areas where Global Positioning System (GPS) is insufficient due to different reasons, one being multipath. With the growth of Wi-Fi access points (AP) and mobile Wi-Fi, many applications which use Wi-Fi for localization and connectivity have been introduced. Wi-Fi localization was first introduced to be used for indoor localization. There are two different methods to use Wi-Fi for indoor localization, fingerprinting and channel modeling. Fingerprinting consist of creating a database of coordinate points and their corresponding RSSs, while the locations of the APs are unknown. The points collected in the database are used to find a current location, the RSSs of a current location will be compared to all of the RSSs in the database. After comparing the RSSs of the current location to the RSSs of the database, whichever point's RSS is closer to the current location's RSS is used to determine the current location's coordinate point. Channel modeling consist of knowing where the Aps are located and using the RSS to determine the current location. The RSS of each AP is measured at the current location, the distance from each AP can be calculated using the RSSs and a propagation model. Multilateration is then used to estimate the current location.

2.2 RFID Introduction and Development

Radio-frequency identification, RFID, transfers data with the purpose of identifying and tracking tags attached. RFID have many applications in our technology advanced world today that we may not even realize. RFIDs are present scanning a bus card, pay to tap with credit cards, and card access to buildings to name a couple. The RFID consist of a RFID tag that is attached to the object that is being identified and RFID reader to read the information. There are two different types of RFID tags, active and passive. The active tags contains an on-board battery and every so often transmits its ID signal. The passive tag on the other hand doesn't have batteries, it uses radio energy that is transmitted by the reader. The tags consist of an integrated circuit, that stores and processes information and other specialized functions, and an antenna, for receiving and transmitting the signal. The reader is made of a scanning antenna that puts out radio frequency signals. The signals give the readers and tag a way to communicate between each other. The reader first sends a signal to the tag. The tag receives the signal and modulates the signal before sending it back to the reader. When the reader receives the new modulated signal, it converts it to digital. The reader is then able to display all of the information that is being transmitted and received through a software user interface. The RFID readers mainly use either polarized linear antennas or polarized circular antennas. The polarized linear antennas broadcast on a single plane, this tends to have a higher read range when it is pointed in the direction of the tag. The polarized circular antennas give out electromagnetic fields, in a corkscrew like fashion. This broadcasts electromagnetic waves in two planes, which makes a complete revolution in a single wavelength. Since the power is being split across two separate planes, these antennas lose about 3 dB per read.

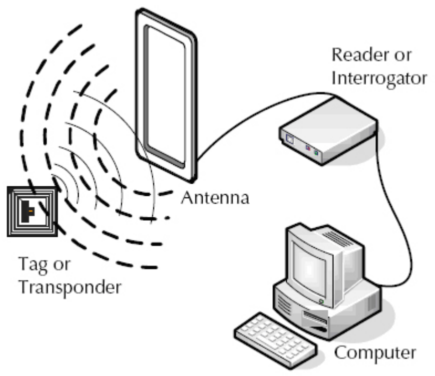


Figure 2.2: Example of RFID Development Kit Interaction with Tag, Reader and Computer

2.3 Development History of iBeacon and the Features of iBeacon

iBeacon is a new technology based on Bluetooth developed by Apple company in 2013. This technology allows devices such as smartphones and tablets to interact with an iBeacon when in close range. Since this technology can be used on smartphones, especially iPhones, various applications have been developed and more and more attention is paid to this new type of technology. iBeacon uses Bluetooth low energy proximity sensing to transmit a universally unique identifier picked up by a compatible app or operating system. The identifier and several bytes sent with it can be used to determine the device's physical location, track customers, or trigger a location-based action on the device such as a check-in on social media or a push notification. The beacons broadcasts tiny packets of data and for each beacon, it has its unique iBeacon ID which is 20 bytes long and divided into three sections: UUID (16 bytes), major number (2 bytes) and minor number (2bytes). All the beacons have a default UUID and the major number and minor number can be modified.

The functions of iBeacon can be explained as region monitoring and ranging. With region monitoring, the number of region is limited to 20 and can function even in the background and has different delegates to notify the listening app (and user) of entry/exit in the region, which means if a user has a locked iPhone with iBeacon, the iBeacon can still receive messages. And this type of monitoring gives a closed app an opportunity to react to the entry of a region. Another function of iBeacon is ranging. Ranging can provide a list of beacons with their UUID which are in the region. An iOS device can approximate the distance from the iBeacon and the distance can be categorized into 3 distinct ranges: immediate (within a few centimeters), near (within a couple of meters), far (greater than 10 meters). The transmission range of an iBeacon will depend on the location and placement. The standard beacons have an approximate range of 70 meters. Long range beacons can reach up to 450 meters.

Compared with normal Bluetooth devices, iBeacon consumes less power because this technology use the Bluetooth low energy protocol. And this LE protocol is significantly power efficient. With the Apple's recommended setting of 100ms advertising interval with a coin cell battery, it has a life time of 1-3 months and this number increases to 2-3 years as advertising interval is increased to 900ms. Battery consumption of the phones is a factor that must be taken into account when deploying beacon enabled apps. A recent report has shown that older phones tend to draw more battery in the vicinity of iBeacons, while the newer phones can be more efficient in the same environment.[19] In addition to the time spent by the phone scanning, number of scans and number of beacons in the vicinity are also significant factors for battery drain, as pointed out by the Aislelabs report.

Compatible devices:

- iOS devices with Bluetooth 4.0 (iPhone 4S and later, iPad (3rd generation) and later, iPad Mini (1st generation) and later, iPod Touch (5th generation).
- Macintosh computers with OS X Mavericks (10.9) and Bluetooth 4.0.
- Android 4.3+.
- Windows Phone devices with the Lumia Cyan update or above.

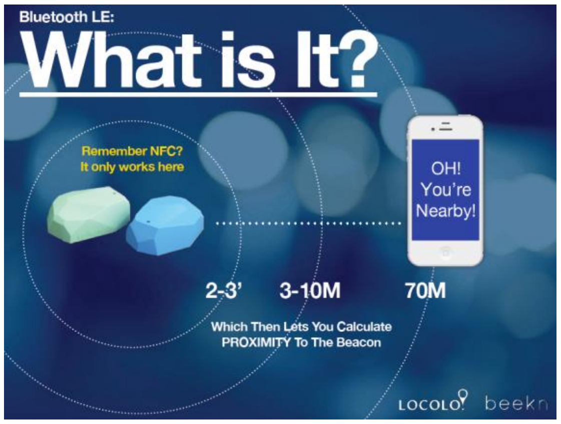

Bluetooth LE:
What is It?
Remember NFC? It only works here
OH! You're Nearby!
2-3'
3-10M
70M
Which Then Lets You Calculate PROXIMITY To The Beacon
LOCOLO
beekn

Figure 2.3: Example of How iBeacon and Smart Devices Interact

2.4 Estimote Development Kit for iBeacon

Estimote Beacons are certified Apple iBeacon compatible devices used for iBeacon development. They are small wireless sensors that can be attached to any object. With a smartphone, you are allowed to unlock mirco-location and appropriate awareness through receiving and interpreting small radio signals that are sent out from the beacons. Using Estimote SDK, apps on your smartphone are allowed to interact with the beacons. The smartphone is able to understand things such as, its proximity to close objects, approximate location, temperature and motion. You can use all of this information and data to develop mobile apps that allows your smartphone to interact with the real world.

The beacons are similar to tiny computers, they have a very powerful RAM processor, memory, Bluetooth Smart module and both temperature and motion sensors. Running on a coin battery, they broadcast radio signals through its built-in antennas. These signals are received by smartphones that are in range. Apps that are installed and compatible can respond to the signals.

The response is flexible to the user's needs, it can be a notification or any other action. Everything gets tied together through Estimote's SDK and Cloud by authorizing access to the metadata. The metadata includes beacon ownership, the object type, and precise location. Security and other services on top of the beacon are also enabled.

Figure 2.4: Estimote Beacon Development Kit (3 iBeacon units shown on the left)

Chapter 3: Methodology

This chapter provides the methods that developer used to achieve the purpose of the project. It first describes the technic of indoor Wi-Fi localization. And then, the chapter introduces the method of Cramer-Rao low bound in simulating the result of localization. At the last part, it provides the algorithm that the developer invented to achieve the indoor hybrid localization goal. The specific design process can be followed as the Figure 3.0 shows.

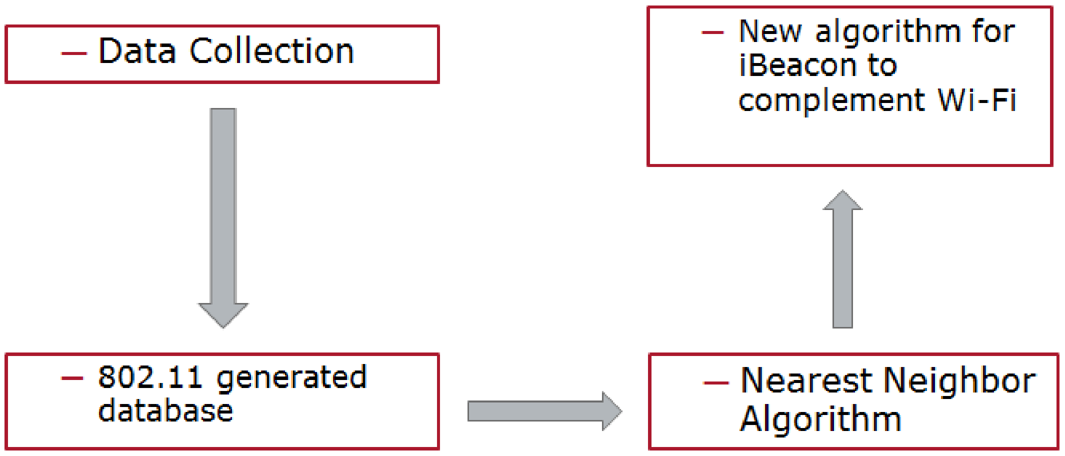

– Data Collection
– New algorithm for iBeacon to complement Wi-Fi
– 802.11 generated database
– Nearest Neighbor Algorithm

Figure 3.0: Hybrid Localization Algorithm Design Procedure

In section 3.1 and 3.2 go through the data collection and analysis. Sections 3.3 and 3.4 talks the simulation for our scenario. Section 3.5 discusses our path-loss based nearest neighbor algorithm design and in room presence detection algorithm.

3.1 Data Collection and Scenario Development

Before we start the whole project, we need to figure out how the project scenario looks like and how we can collect the data we need appropriately.

3.1.1 Scenario Development

As we can see in the figure below, the twenty points in the rectangular area are the data points we picked and the four dark red points are the access points. Since we need to specify the location of each point clearly, we also develop an x-y plan to note down the specific location using x-y coordinate of each point. We chose the black point at the corner as our original point whose coordinate is (0,0) and the x-axis and y-axis are distributed in the normal way. So for each of the data point and access we have an associated coordinate. We can easily to calculate the distance between any two of them in this way.

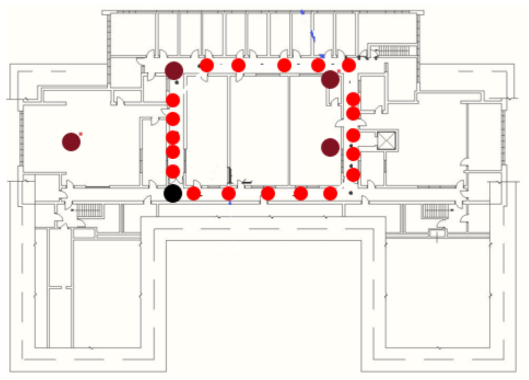


Figure 3.1.1: Map of the Project Scenario with Data Points

3.1.2 Data Collection

For this project, first we need the data configuration of the access point so that we can make nothing will go wrong in the next step because every other data is associated with the access points. Then we need a database which is consisted of a relatively large number of data as the localization database. And the last thing we need is the test data points to test how well our algorism works.

3.1.2.1 Access Point Data Collection

For the study requirement, we need to know the transmit power of each access point and the first meter path loss of them. But before we measure the data, we need to calculate the first meter path loss using the data configuration we got from the website. The equation to calculate the received signal strength at the first meter is below.

$$\mathrm{P}_0 = P_t \times G_t \times G_r \times \left(\frac{\lambda}{4\pi}\right)^2$$

And the calculated first meter path loss of each access point is given in the table below.

\

	Theoretical 1st Meter Path Loss(dBm)
AP1	40.20810682
AP2	39.31868808
AP3	39.2854724
AP4	39.25212921

Table 3.1.2.1.1: Theoretical First meter Path Loss of the APs

And then we went to the actual places to measure the first meter path loss. For the more accurate result, we chose six different positions around the access point to measure the data and after measuring, we take the average of the six numbers to use it as the first meter path loss for the access point. And the actual first meter path loss is in the table below

.

	Actual 1st Meter Path Loss(dBm)
AP1	54
AP2	49
AP3	50.5
AP4	50.833

Table 3.1.2.1.2: Actual First Meter Path Loss of the APs

As we can see in the 2 tables above, we can see that the theoretical first meter path losses of the four access point are smaller than the actual first meter path loss. The explanation we have for this situation is that the scenario is not real free space. Although there was nothing between the access point and the receiver, the access point itself still has a plastic cover and also the case of the receiver can block some of the signal since we use our computer as the receiver. After consideration, we decided to take the measured data for future use.

3.1.2.2 Database Points Data Collection

For the database measurement, we have 20 point around the rectangular space. To make the database more reliable, we decided to measure 10 time at each of the data point, which means we have total of 200 numbers sets in our database.

To achieve that purpose, every time we stand on one selected point, we hold the receiver and turn around in a circle slowly and note down the RSSI associated with all the four access points. So in the database we have a 200*4 matrix. The 200 rows stand for the 200 data sets and the 4 columns stand for the four RSSI associated with each dataset. There is an example of one data set in our data base. The four numbers in the matrix from left to right indicates the RSSI in dBm to the AP1 to AP4 consecutively.
[-64 -75 -69 -50]

Figure 3.1.2.2: Dancing Points Method used to Collect Data

3.1.2.3 Test Points Data Collection

Since we already have 200 sets of points as our database, we still need some points localization accuracy testing.

The way we use to collect theses points is to walk slowly around the rectangular scenario and collect the RSSI to each of the four access point. After one loop of walking, we collected 24 sets of data. This time we did not stop at each location and collect a series of data because test point is collected to simulate the situation that a guest is walking around the scenario continuously. So the data can test accuracy of the algorism we create.

3.2 Data Analysis

When we analysis the data we have, we need to use the channel modeling method because by measuring we can only get the RSSI and the distances between the points. The only way to associate these two variables together is the channel modeling approach. So analyzing the data is most likely to channel model the data.

3.2.1 Empirical Model Development

The existing channel modeling system is the 802.11 standard. However, this standard is not necessarily suitable for every building. So we decided to develop the unique channel modeling standard for the scenario.

The basic channel modeling equation is

$$\begin{cases} Lp = Lo + 20\log(d) & d < d_{bp} \\ Lp = Lo + 20\log(d_{bp}) + 35\log\left(\frac{d}{d_{bp}}\right) & d > d_{bp} \end{cases}$$

So the most important part of defining a channel modeling standard is to find the α_1, α_2, shadow fading and the break point.

As in the 802.11 standard, we chose 5 meters as our break point. Then we chose the 200 points database to develop the model. Since we already measured the data for the access point, the L_0 can be easily figured out. The next step is to plot the distance against the received signal strength. The distance is referred to the distance between the data point and the access point. We can have these distances easily because we have already developed an x-y plan of the scenario and each data point has an associated coordinate.

For all the points we have, we can draw two best fit lines before and after 5 meters, which is our break point. Then we set the distance axis in log based so the plot looks like a straight line. Then the slops of the two lines are the parameters we want.

Then for the shadow fading calculation, we set the best fit line as our average value and then we can calculate the variance of the 200 data points. This variance is the variance of the shadow fading of our model.

3.2.3 Regenerate the Database Using 802.11 Channel Model

After we create the empirical channel model, we need to compare it to the existing channel model system which is the 802.11 channel model standard.

Usually, the nearest neighbor algorism is the most simple and accurate algorism among many different indoor localization methodologies. However, it has its disadvantages. In terms of accuracy, lots of data as reference points has to be collected. In order to take those data, technicians usually need to go to field and record data point by point. At this point, a question had come out: is there a way that do not need to take those many sample data or even do not need to actual go to the place to take data? Based on what we've learned, we had come up with an idea that uses channel-model we developed to regenerate data sets simulated as real points.

All data of points could be stored as received signal strength from each AP define as $i\ X\ j$ matrix R, which i is the number of Aps and j is the number of database points. We uses P to represent RSS. D as distance

$$R = \begin{bmatrix} P_{11} & \cdots & P_{i1} \\ \vdots & \ddots & \vdots \\ P_{1j} & \cdots & P_{ij} \end{bmatrix}$$

We could find the twenty points location Based on the map figure 3.1.1.1. By using the coordinates we have for the twenty points to calculate the distance database as distance to each AP matrix define as $i\ X\ j$ matrix D.

$$D = \begin{bmatrix} D_{11} & \cdots & D_{i1} \\ \vdots & \ddots & \vdots \\ D_{1j} & \cdots & D_{ij} \end{bmatrix}$$

Based on 802.11 model c, we could apply our new dataset of distance to calculate a new database of received signal strength which can be represent as N

$$R_{new} = \begin{bmatrix} N_{11} & \cdots & N_{i1} \\ \vdots & \ddots & \vdots \\ N_{1j} & \cdots & N_{ij} \end{bmatrix}$$

As the equation below

$$\begin{cases} R_{new} = pt - (Lo + 20\log(D)) \\ R_{new} = pt - \left(Lo + 20\log(5) + 35\log\left(\frac{D}{5}\right)\right) \end{cases}$$

Pt is the transmit power of AP and L0 is the first meter path-loss of AP.

Can be derived as

$$\begin{bmatrix} N_{11} & \cdots & N_{i1} \\ \vdots & \ddots & \vdots \\ N_{1j} & \cdots & N_{ij} \end{bmatrix} = Lo + 20log \begin{bmatrix} D_{11} & \cdots & D_{i1} \\ \vdots & \ddots & \vdots \\ D_{1j} & \cdots & D_{ij} \end{bmatrix}$$

Lo is the measured first meter path-loss. Therefore, we have a new matrix of received signal strength reference database called model generated database. This database requires the transmitted power of AP, first meter path loss and a map which can generate coordinated of the simulated database points.

Therefore, we can add a shadow fading with σ to increase the database size to increase the localization accuracy. At this point, we can generate k new data by adding Gaussian white noise with mean $rssn_{ij}$ and empirical σ to have a new three dimensional $i\ X\ j\ X$ k database called new model generated database.

i is the number of Aps, j is the number of database points and k is the number of new data with noise for each point.

Then our dataset can be transformed as i number of $j\ X$ k matrix.

$$\begin{bmatrix} N_{111} & \cdots & N_{11k} \\ \vdots & \ddots & \vdots \\ N_{1j1} & \cdots & N_{1jk} \end{bmatrix}$$

$$\begin{bmatrix} N_{211} & \cdots & N_{21k} \\ \vdots & \ddots & \vdots \\ N_{2j1} & \cdots & N_{2jk} \end{bmatrix}$$

$$\vdots$$

$$\begin{bmatrix} N_{i11} & \cdots & N_{i1k} \\ \vdots & \ddots & \vdots \\ N_{ij1} & \cdots & N_{ijk} \end{bmatrix}$$

We can call the new database as BASE. This is the model generated database with 400 (i = 4, j = 20, k = 10) points as we used as our reference point to apply the nearest neighbor algorism.

3.3 Nearest Neighbor Algorithm

Based on the reference database we have for Atwater Kent third floor, we can simulate the actual scenario by walking around as usual. Then we take twenty-four points as the test database to determine the accuracy of the original algorism and our algorism as we walk through the test field.

The simple example scenario is shown below.

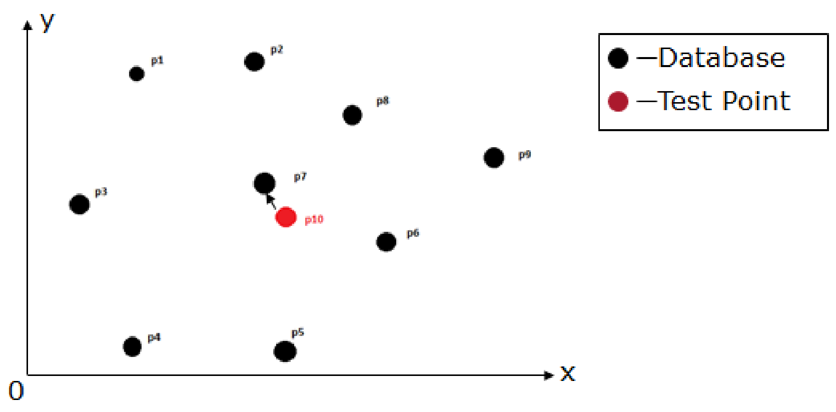


Figure 3.3.0: Figure of The Nearest Neighbor Algorithm Example

$$D_{\min}=\min(\textstyle\sum_{i=1}^{i=9}\sqrt{(x_i-x_{10})^2+(y_i-y_{10})^2}) \Longrightarrow \boxed{\text{Consider p7 as the actual position of p10}}$$

3.3.1 RSSI Based Nearest Neighbor

For RSSI based nearest neighbor, we need a RSS database as reference and take a test point to compare with all points in the database and find the closest one, we can assume that location is the location of test point.

At first, we have the database of i X j X k matrix BASE. Secondly, we have 24 number of test dataset of i X 1 matrix Rt.

We can take each data point matrix i X 1 matrix Rt to compare with all data points j times. For the convenience of calculation, we write Rt as an i X j matrix by repeat the first row j times, and we write Rt as an i X j X k matrix by repeat the first column k times.

We need to apply least squares algorism twice, the first time we find the minimum value of each location.

$$min\sum_{i=1}^{n}(Rt_i - R_i)^2$$

The second time we find the minimum value of all location to find the best location.

We could calculate the first step of least square set of data using the equation to find the i number of matrixes.

$$\begin{bmatrix} Rr_{m1} \\ \vdots \\ Rr_{mj} \end{bmatrix} = \begin{bmatrix} \min((Rt_{11} - R_{11})^2 & \cdots & (Rt_{j1} - R_{j1})^2) \\ \vdots & \ddots & \vdots \\ \min((Rt_{1k} - R_{1k})^2 & \cdots & (Rt_{jk} - R_{jk})^2) \end{bmatrix}$$

Therefore, we can write four matrixes as

$$\mathrm{A} = \begin{bmatrix} Rr_{m11} & \cdots & Rr_{mi1} \\ \vdots & \ddots & \vdots \\ Rr_{m1j} & \cdots & Rr_{mij} \end{bmatrix}$$

The next step is take the least square to find a least value.

$$B = \begin{bmatrix} \min(\sum_{i=1}^{n}(Rr_{m11} \ \ldots Rr_{mi1})) \\ \vdots \\ \min(\sum_{i=1}^{n}(Rr_{mj1} \ \ldots Rr_{mij})) \end{bmatrix}$$

Therefore,

$$LSF = \min(B)$$

Where the 'min()' is a function that can find the least j number of B. The coordinates corresponding to the j is the location to be localized.

3.3.2 Distance Based Nearest Neighbor

Distance based nearest neighbor sharing almost the same algorism as RSS based nearest neighbor. The difference is that the database we had for reference and test are all based on distance values.

At this point, suppose that RSS is already known, we could calculated the distance to AP database by using the equation 3.3.2.1 derived from equa3.2.1.1

$$\begin{cases} D = 10^{\frac{pt-pr-lo}{20}} \\ D = 5 \times 10^{\frac{pt-pr-lo-20\log(5)}{35}} \end{cases}$$

Therefore, we can generate a database by using all Pr values which is matrix D. Then we can use the Rt database to generate a Dt database. By applying the same algorism above, we can find the location by using distance based nearest neighbor.

3.4 Cramér–Rao Lower Bound

Cramér–Rao lower bound (CRLB) is the lowest value of estimator. In our case, we use it to calculate the theoretical least distance measurement error while applying the parameter of each AP.

3.4.1 Single Observation

At first we use Cramer-Rao lower bound to develop the model for each AP to calculate the least DME of each AP.

It can be derive from equation below when our observation is the path-loss model we used for each AP of LOS.

$$f(O/x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{[O-g(x)]^2}{2\sigma^2}} \Rightarrow \quad \hat{x}_{ML} = g^{-1}(O) = 10^{\frac{P_t - P_0}{10\alpha}}$$

After we apply it to Fisher matrix:

$$\mathbf{F} = -E\left[\frac{\partial^2 \ln f(O/x)}{\partial x^2}\right] = E\left[\frac{\partial \ln f(O/x)}{\partial x}\right]^2 = \frac{[g'(x)]^2}{\sigma^2}$$

$$g'(x) = -\frac{10\alpha}{(\ln 10)x} \Rightarrow \mathbf{F} = \frac{(10)^2\alpha^2}{(\ln 10)^2\sigma^2 x^2}$$

The CRLB could be calculated as

$$CRLB = \mathbf{F}^{-1} = \frac{(\ln 10)^2}{100}\frac{\sigma^2}{\alpha^2}x^2 \quad \Rightarrow \quad \sigma_P \geq \frac{\ln 10}{10}\frac{\sigma}{\alpha}x$$

Where α is the path loss gradient σ is the standard deviation and x is the distance. Based on the result, we could see that our CRLB will increases with the distances increases.

3.4.2 Multiple Same Observation

Secondly we use CRLB with N reference APs. Which in our case, there are four APs. Suppose we have a list of observations of path-loss model.

$$P = P0 - 10alog(ri) + x \quad \text{i} = 1,2,3,4;$$

$$r = \sqrt{(x - xi)^2 + (y - yi)^2}$$

Therefore:

$$dPi(x,y) = -\frac{10a}{ln10}\left(\frac{x - xi}{ri^2}dx + \frac{y - yi}{ri^2}dy\right)$$

$$dP = \begin{bmatrix} dP1 \\ \vdots \\ dPn \end{bmatrix}; dr = \begin{bmatrix} dx \\ dy \end{bmatrix}; H = -\frac{10}{ln10} \propto \begin{bmatrix} \frac{x-xi}{ri^2} & \frac{y-yi}{ri^2} \\ \vdots & \vdots \\ \frac{x-xn}{rn^2} & \frac{y-yn}{rn^2} \end{bmatrix}$$

$$dP = Hdr$$

$$dr = (H^T H)^{-1} H^T dP$$

If we have zero mean power variations, which is Gaussian white noise,

$$cov(dpi, dpj) = \begin{cases} \sigma_p^2, i = j \\ 0, i \neq j \end{cases}; i = 1, 2, 3, 4$$

$$cov(dr) = \sigma_p^2 (H^T H)^{-1}$$

$$\sigma_p^2 = \sqrt{\sigma_x^2 + \sigma_y^2}$$

$$CRLB = \sqrt{\sigma_p^2} = \sqrt[4]{\sigma_x^2 + \sigma_y^2}$$

Above is the algorism of Cramer-Rao low bond for multiple observations for LOS. We used it to calculate a theoretical DME to compare with our localization results.

3.4.3 Hybrid Observation

In the scenario of hybrid localization, we have multiple different parameters of path loss models of observations, which are iBeacon and Wi-Fi.

Therefore in this case, we have

$$P = P0 - 10alog(ri) + x \quad \text{i} = 1, 2, 3, 4, 5, 6;$$

And

$$H = -\frac{10}{ln10} [a1 \quad \dots \quad an] \begin{bmatrix} \frac{x-xi}{ri^2} & \frac{y-yi}{ri^2} \\ \vdots & \vdots \\ \frac{x-x6}{r6^2} & \frac{y-y6}{r6^2} \end{bmatrix}$$

Also the parameter of $cov(dpi, dpj)$ will be

$$\sum = cov(dpi, dpj) = \begin{cases} \sigma_p^2, i = j; i = 1,2,3,4 \\ \sigma_i^2, i = j; i = 5,6 \\ 0, i \neq j \end{cases}$$

The first four will be the parameter for Wi-Fi and the last two will be the parameter for iBeacon.

$$cov(dr) = (H^T H)^{-1} H \sum H (H^T H)^{-1}$$

After apply the separate parameter, we could calculate the CRLB by using

$$\sigma_p^2 = \sqrt{\sigma_x^2 + \sigma_y^2}$$

$$CRLB = \sqrt{\sigma_p^2} = \sqrt[4]{\sigma_x^2 + \sigma_y^2}$$

3.5 Presence Detection Application using iBeacon and Hybrid Wi-Fi with iBeacon Localization Algorithm Development

While complementing Wi-Fi, the iBeacon will also be able to interact with smart devices that are in its coverage area. One of the tasks of the iBeacon is to be able to count the number of people that are currently in a professor's office. The RSS will be read both when a person is entering the room and leaving the room. We will use the difference of the RSSs to determine whether or not the person is actually entering or leaving the room.

3.5.1 Presence Detection Algorithm Development

When the person is outside of the room, the RSS will be the strongest because the iBeacon will be on the outside of the door. As soon as the person enters the room, the RSS will no longer be as strong because the device will no longer be right in front of the iBeacon, the door will be in the way of the two. The way that we will keep count of the number of people that are currently in the room will be by comparing the before and after RSSs. We will know a person is entering the room if the RSS decreases and we will know because the RSS will decrease from them being in front of the door and entering the room. We will know a person is leaving the room if the RSS increases from them being inside the room to in front of the door. We simulated different scenarios that can possibly occur to observe the RSS behavior. The first scenario we simulated was someone walking to the professor's office, opening the door and entering the room. The second scenario was if someone is in the professor's office and opens the door, but doesn't leave the office. The last scenario was if someone was just walking by the professor's office and doesn't open the door or enter the room. In all of these scenarios the RSS will fluctuate. We found that when the device is in front of the iBeacon, the RSS would be around -70dBm. We use this as the reference point to determine which of the three scenarios is taking place. The graph below simulates the first scenario of someone entering the room.

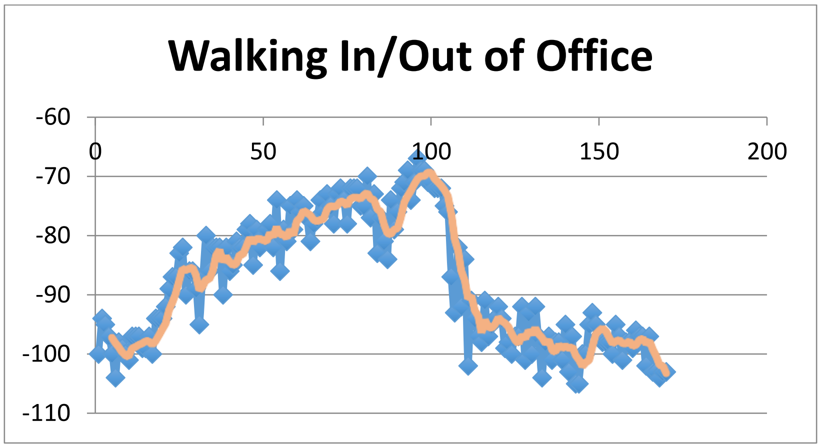


Figure 3.5.1.1: RSSI Changes of iBeacon When Waling In/Out the Office

As shown in the graph above, when the person is further away from the door walking towards it, the RSS starts weak at about -97dBm and slowly increases as the person gets closer to the door. We can see when the person gets to the door because the RSS is the strongest, -70dBm. After the reference point, the RSS drops sharply then remains almost constant at around -97dBm. If the person was then leave the room, the RSS would sharply rise to -70dBm as they are walking by the iBeacon in front of the door, then slowly begin to decrease to around -97dBm as the walk away from the professor's office. If the person was to start in the professor's office, the RSS will be weak until they opened the door and left. The RSS will get significantly stronger quickly, and then gradually get weaker as they are walking away from the iBeacon. The graph below simulates the scenario of someone who is inside the professor's office and opens the door, but doesn't actually exit the office.

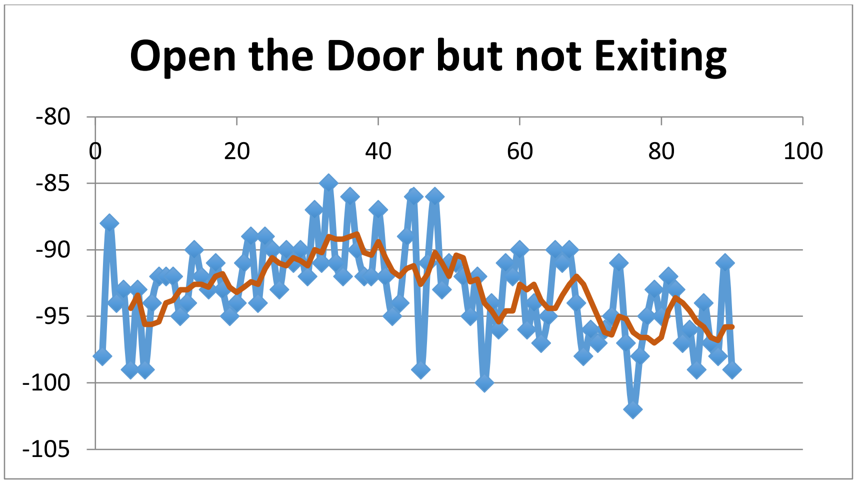


Figure 3.5.1.2: RSSI Changes of iBeacon When Door is Opened but Don't Enter/Exit

As shown in the data above, there is no real significant drop in the RSS. The RSS does fluctuate between -85dBm and -97dBm, but it never reaches the reference point of -70dBm, so we know that the person never actually walks out of the office in front of the iBeacon. The graph below simulates when a person is just passing by the professor's office.

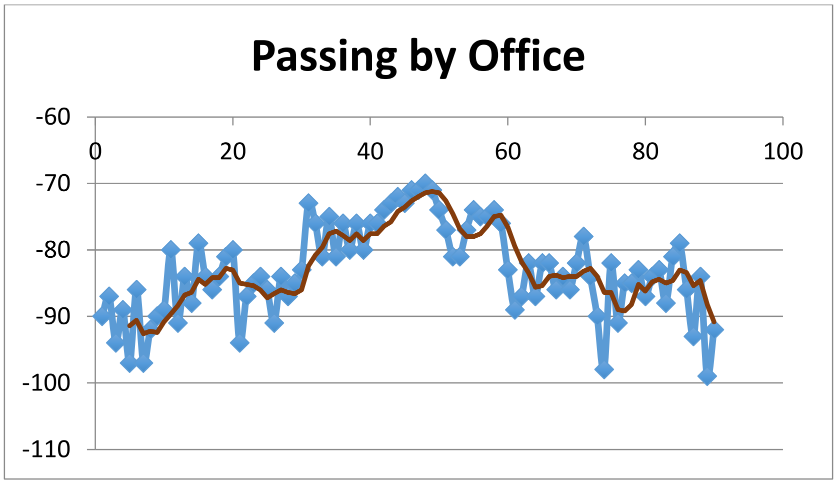


Figure 3.5.1.3: RSSI Changes of iBeacon When a Passing by the Office

As shown in the data above, the RSS starts weak at around -95dBm and slowly increases to -70dBm as the person gets in front of the door. After the person is at the door, the RSS begins to get weaker again and drop back down to about -95dBm. The trend line is similar to when the person is walking into the professor's office. The difference is that when the person is entering the room, there is a significant drop right after the reference point compared to when they are passing by where the drop is more gradual.

Therefore, we could apply the algorithm below to achieve the status of in/out room presence.

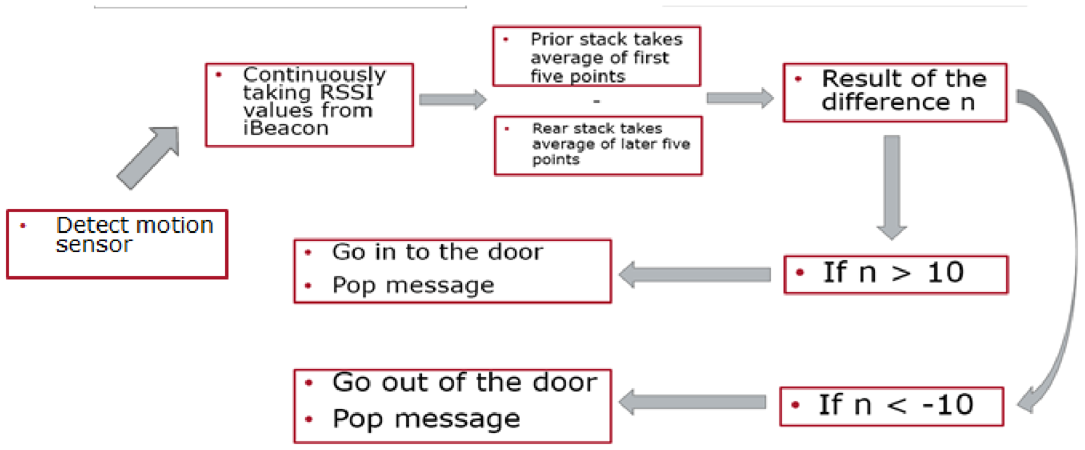

Detect motion sensor
Continuously taking RSSI values from iBeacon
Prior stack takes average of first five points
-
Rear stack takes average of later five points
Result of the difference n
If n > 10
Go in to the door
Pop message
If n < -10
Go out of the door
Pop message

Figure 3.5.1.4: Presence Detection Algorithm Flow Chart

3.5.2 Hybrid Wi-Fi and iBeacon Localization Algorithm

To increase the accuracy of Wi-Fi localization, we decided to add iBeacon in the scenario. And the method we use to hybrid these two devices is given as followed.

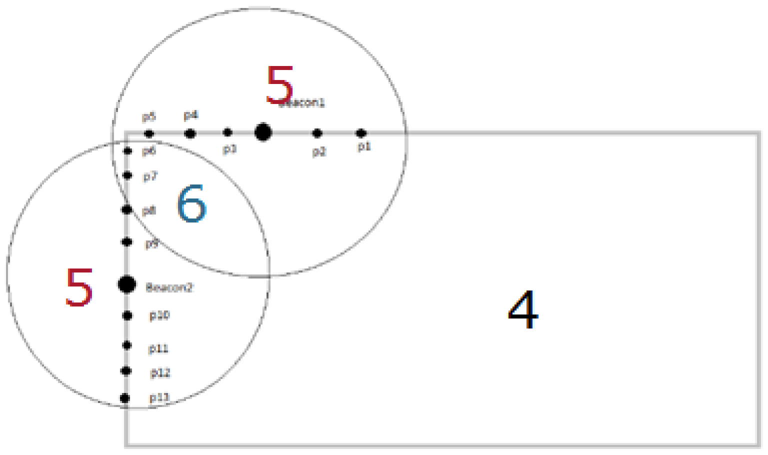


Figure 3.5.2.1: Diagram of the Hybrid Localization Algorithm Design

As shown in the figure above, the gray rectangle is the Wi-Fi database we have, the two bigger dots are the two iBeacons and the circles are the range of the iBeacon. At the same time, the smaller dots are the test points.

For those test points which are only in the range of Beacon1, when we do the nearest neighbor algorithm, we only use the data points within the range of the iBeacon1 instead of the whole rectangular database. And for those points only within the range of Beacon2, we only use the database in the circle of Beacon2. But for the test points which are in the both iBeacons' range, we use the database that the two circles cover. This means for the P6, P7 and P8 in the figure, we use the database within the two circles when we do the localization. If you not in the range of the iBeacon, 4 Wi-Fi aps would be used for the localization. But if you are in the range of iBeacon, 5 or 6 aps get used depending on where you are. 5 aps will get used if you are only in the range of one iBeacon and 6 will get used if you are in the range of both iBeacons.

Chapter 4: Results

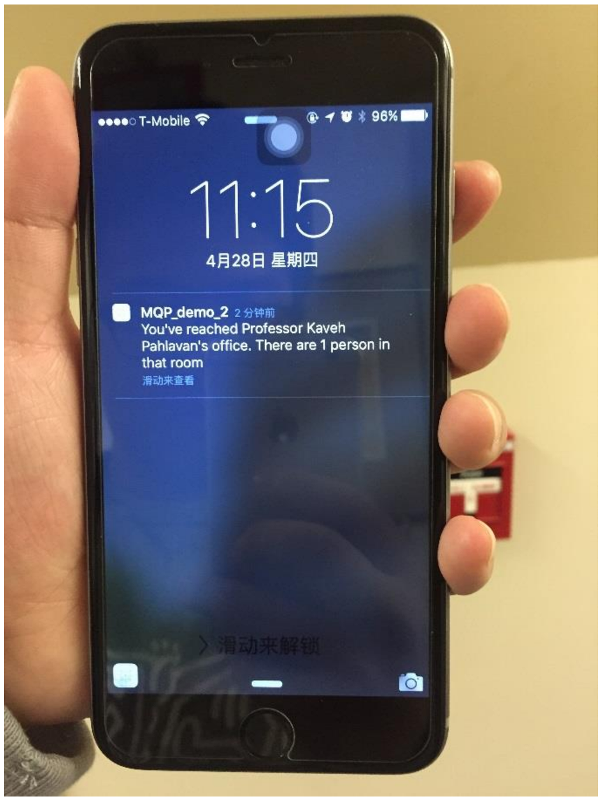
This chapter provides the results and discussion of application development and Hybrid indoor localization and it also provides the progress results in detail. Section 4.1 talks about the results from our application. In section 4.2 we discuss the results from our analysis on just Wi-Fi localization. Section 4.3 and 4.4 go through the analysis on iBeacon and section 4.5 is about the results from the hybrid localization.

4.1 Applications of iBeacon

In our project, there are two parts. The first one is to find algorithm that could increase iBeacon and Wi-Fi accuracy, the second one is to use iBeacon's advantage such as tracking and broadcasting to develop an application which could achieve three functions ---Distance estimation, in room presence and pushing information.

4.1.1 Application of Broadcasting and Distance Estimation

When the application finds that the phone is in the range of iBeacon's coverage, it will pop up the message 'You've entered Professor Kaveh Pahlavan's office. There are 0 person in that room'. And when the phone leaves the range of iBeacon, the application will tell the user that 'You've entered Professor Kaveh Pahlavan's office. There are 1 person in that room'.



MQP_demo_2 2 分钟前
You've reached Professor Kaveh Pahlavan's office. There are 1 person in that room

Figure 4.1.1.1: Example of Proximity Broadcasting Application from iBeacon

Another function of the application is using the ‘immediate’, ‘near’, ‘far’ and ‘unknown’ to indicate the distance between the iBeacon and the phone.

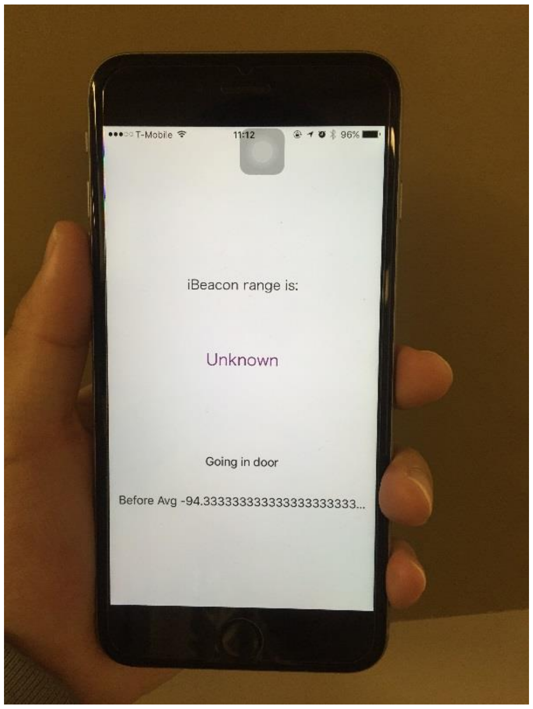


Figure 4.1.1.2: Example of the Distance Estimation and In Room Presence Detection Application from iBeacon

By using this application, we could apply it to the scenario that when a student passes by a professor's office, he/she could receive a message that contains information indicates the professor's name, his/her in/out room presence and information about professor's office hour inside the app.

4.1.2 Application on iPhone to Count the Number of People in the Room

To achieve the goal of counting people in the room, we need to know whether a person is entering the room or leaving the room.

The first function of our app is to read the RSSI of the iBeacon and mesmerize the RSSI over the period. The app will note down the peak point of RSSI over the period. Since the iBeacon is placed outside the door, if a person is entering the room, the average RSSI before the peak point will be about 10 dBm bigger than the average RSSI after the peak point. And if a person is leaving the room, the results will be reversed. At the same time, the message that indicates the status of the person will be broadcasted on the screen. And the application demo is given in the figure below.

As long as we can discriminate the status of the person, we can upload the information to an online server and the server can get the number of people in the room by simply doing addition and subtraction.

4.2 Wi-Fi Localization Results

4.2.1 Empirical Channel Model Result

For the empirical channel model, we arranged all the into one diagram and find the best fit line to find the two alphas and then we calculated the variance of the whole dataset to find out the variance of shadow fading. And in this case, we take the best fit line as the average value of the data.

Since we wanted to be more specific, we developed a channel model standard for each of the access point. The graphs and the parameters are given as followed.

4.2.1.1 Analyzing for Access Point 1

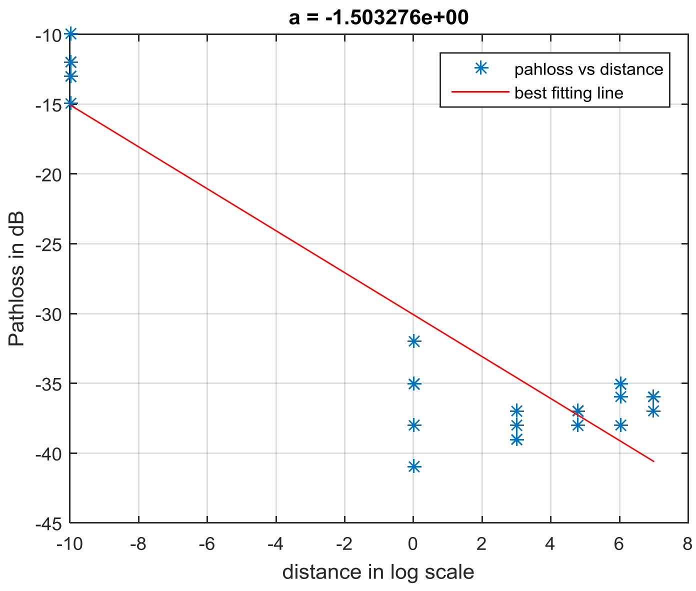


Figure 4.2.1.1.1: Data in 5 Meters for AP1

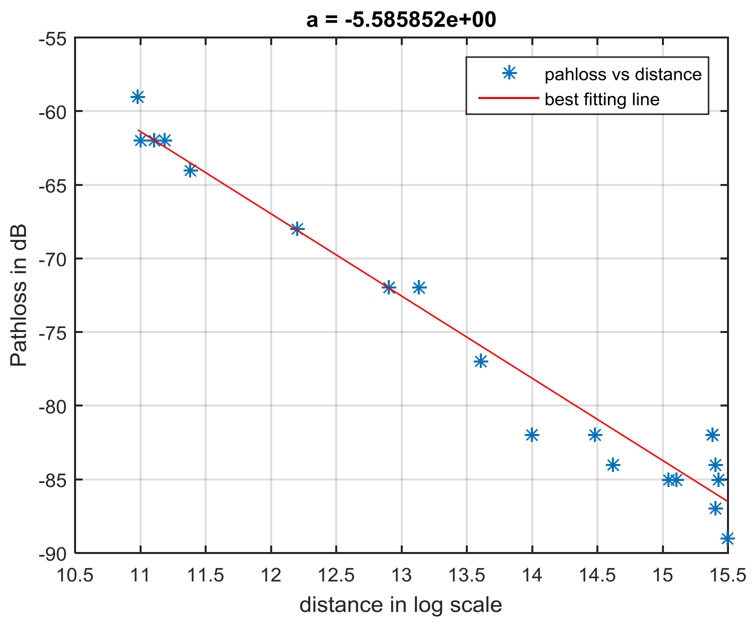


Figure 4.2.1.1.2: Data out of 5 meters for AP1

As we can see in the two graphs above, when within the range of 5 meters, the alpha1 for access point1 is about 1.5 and when the range is larger than 5 meters, the alpha for access point 1 is about 5.6. Then the calculated variance of shadow fading is 7.6.

4.2.1.2 Analyzing for Access Point 2

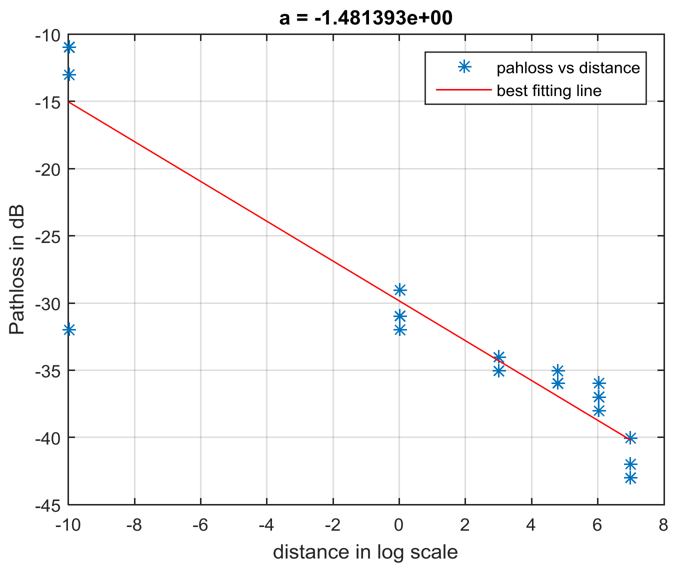


Figure 4.2.1.2.1: Data in 5 Meters for AP2

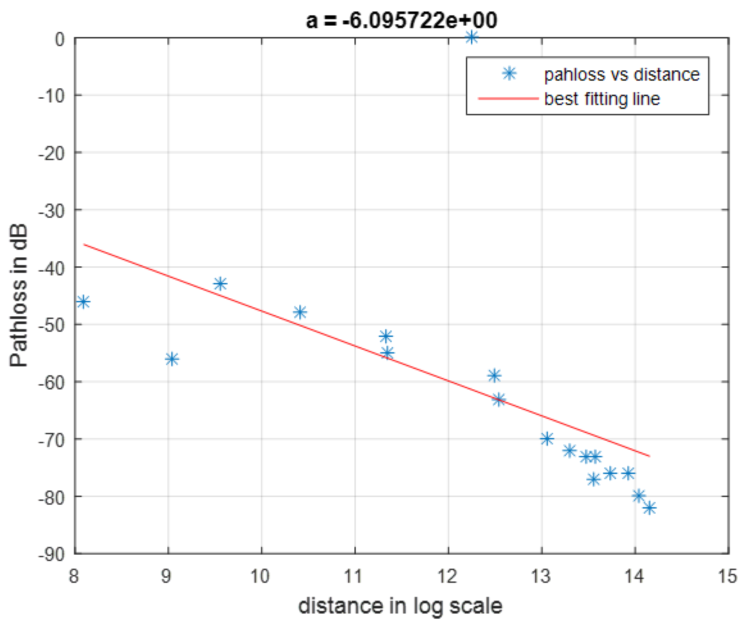


Figure 4.2.1.2.2: Data out of 5 meters for AP2

As we can see in the two graphs above, when within the range of 5 meters, the alpha1 for access point2 is about 1.48 and when the range is larger than 5 meters, the alpha for access point2 is about 6.1. Then the calculated variance of shadow fading is 7.95.

4.2.1.3 Analyzing for Access Point 3

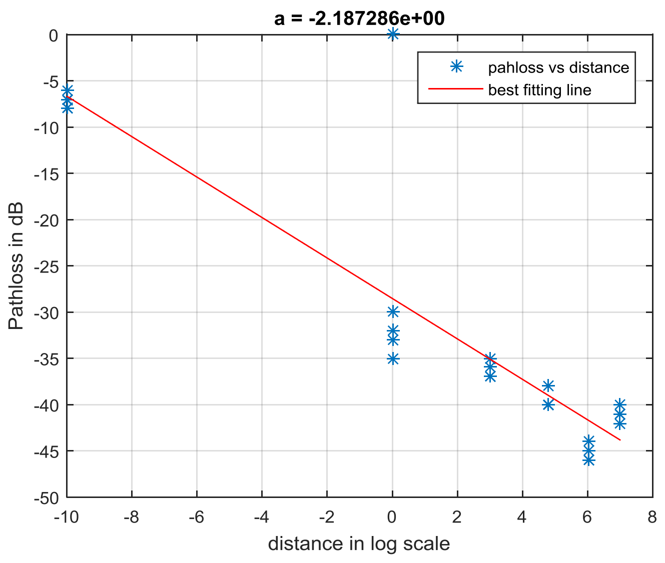


Figure 4.2.1.3.1: Data in 5 meters for AP3

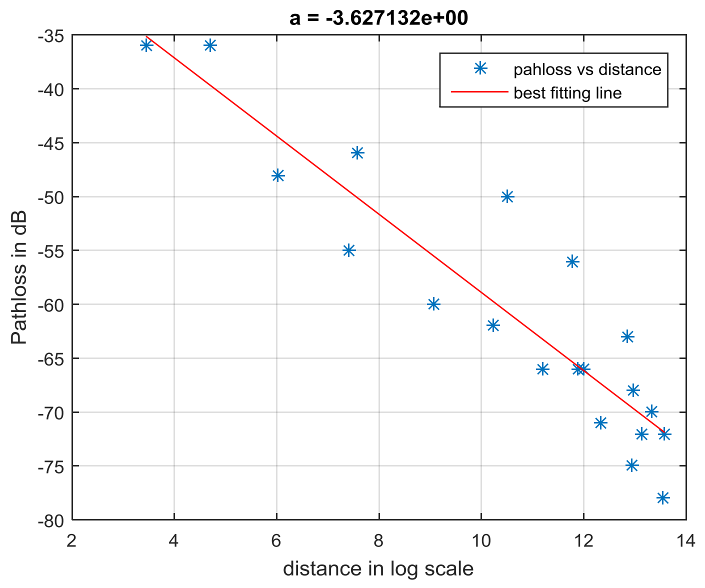


Figure 4.2.1.3.2: Data out of 5 Meters for AP3

As we can see in the two graphs above, when within the range of 5 meters, the alpha1 for access point3 is about 2.19 and when the range is larger than 5 meters, the alpha for access point3 is about 3.63. Then the calculated variance of shadow fading is 4.17.

And in this case, the parameters and the variance are almost the same as those in the 802.11 standard.

4.2.1.4 Analyzing for Access Point 4

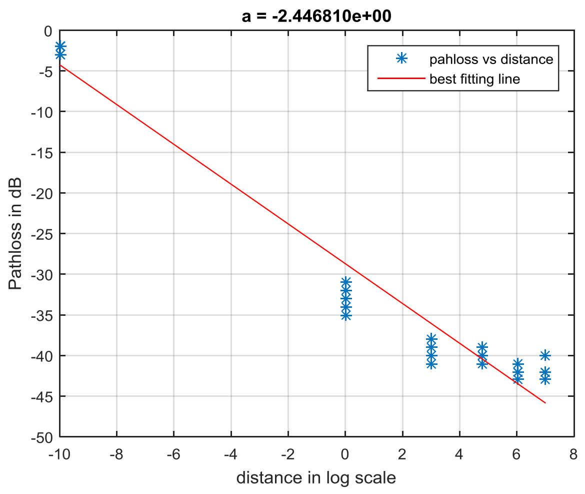


Figure 4.2.1.4.1: Data in 5 Meters for AP4

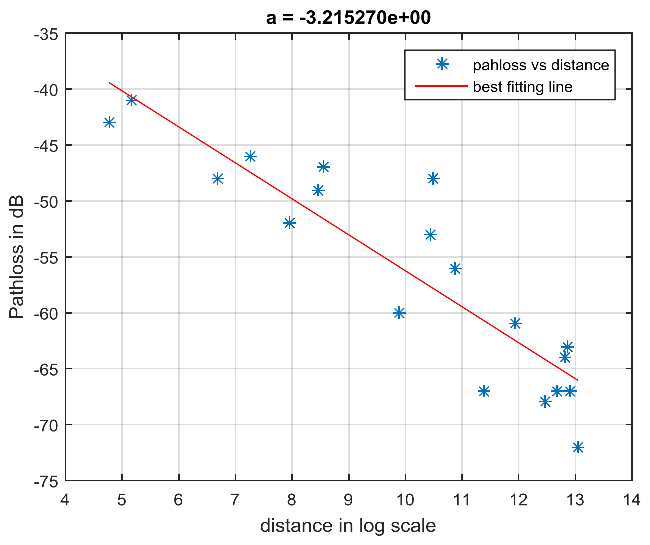


Figure 4.2.1.4.2: Data out of 5 Meters for AP4

As we can see in the two graphs above, when within the range of 5 meters, the alpha1 for access point4 is about 2.4 and when the range is larger than 5 meters, the alpha for access point3 is about 3.2. Then the calculated variance of shadow fading is 3.827.

Also in the case of access point 4, we found that the alphas and variance are similar to the 802.11 standard.

4.2.1.5 Conclusion on the Empirical Channel Model

The data from the four access points seems different from each other although they are the same type of access points produced by the same company. And after taking the position of the access point into consideration, it may cause some of the problems. As we can see in the figure of the scenario, the access point is far away from other points and this may cause some measurement error. And at last we decided to compare the empirical standard and the 802.11 standard to determine which one we should use for future calculation.

4.2.2 Wi-Fi Localization Results between Generated Database and Empirical Database

After calculation, we have three sigma generated from 3 different sources. The sigma of 8 is from the 802.11 channel model. The sigma of 4.377 is generated from the 200 sets of database and the sigma of 5.648 is generated from the 24 test points. The CDF plot of the three sigma is given as followed.

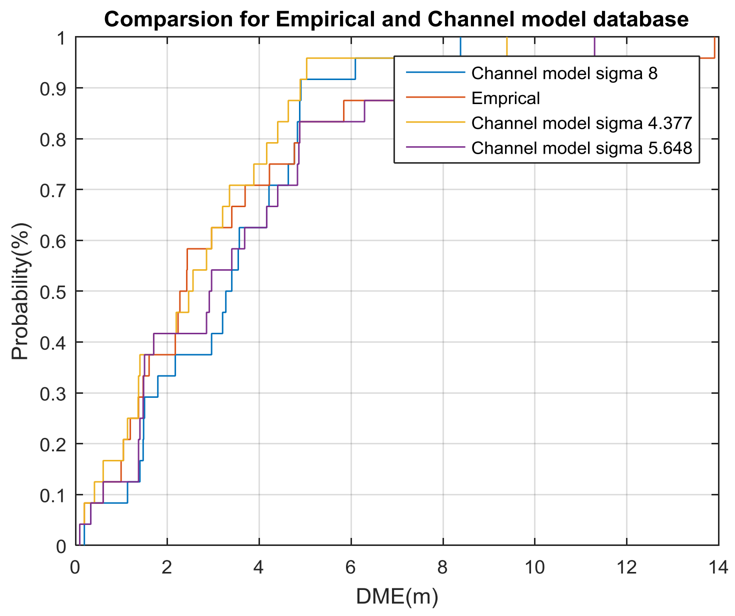


Figure 4.2.2: Distance Measurement Error Comparison of Generated Database and Empirical Database

Scenarios(m)	Mean	Median	Range of Error	Standard Deviation
Empirical	3.26	3.335	8.19	1.9446
8	3.3272	2.345	13.82	3.153
4.377	2.7181	2.5078	9.3	2.0899
5.648	3.5352	2.935	11.208	2.8416

Table 4.2.2: Data for the Different Sigma

As we can see from the figure above, not matter which sigma we use, the CDF plot is almost the same. So we can assume that the localization is almost independent of the variance of shadow fading, which means shadow fading in the building is not a critical problem that affects accuracy. Also as the result shows, out generated database is as good as empirical database. We could apply this method into the future use to save more time in data collecting.

4.3 Cramér–Rao Lower Bound

4.3.1 Cramér–Rao Lower Bound of RSS and Distance Based Preliminary Results

The CRLB of the test points for the Wi-Fi localization can be determined using MATLAB and the results are given as followed.

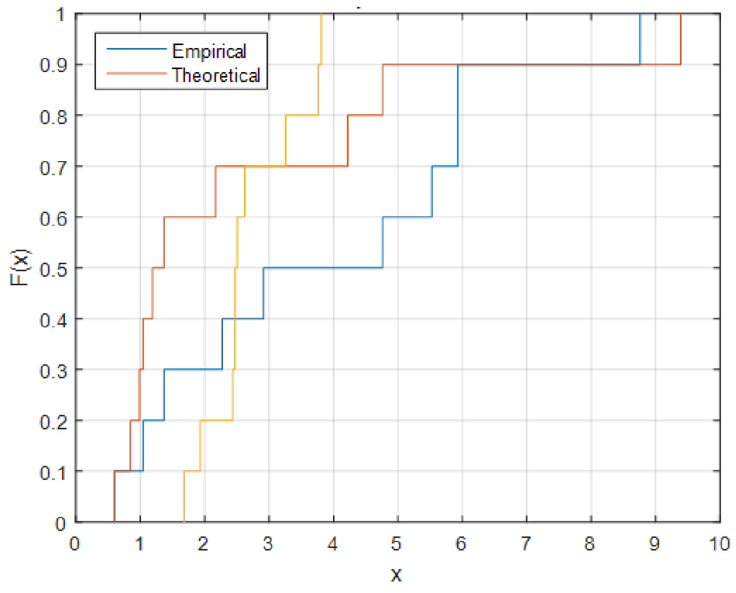


Figure 4.3.1.1: RSS Based Comparison of 2.4GHz and 5GHz and CRLB

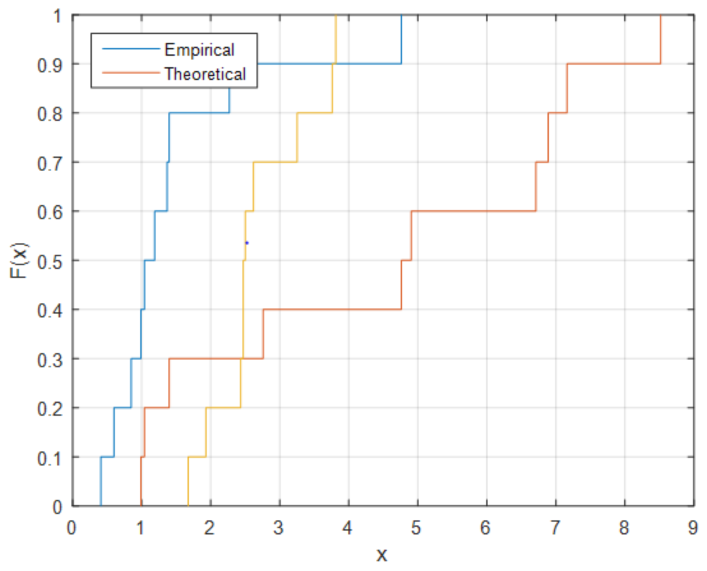


Figure 4.3.1.2: Distance Based Comparison of 2.4GHz and 5GHz and CRLB

The yellow line above indicates the Cramer-Rao lower bound, which is the least DME (Distance Measurement Error). It shows that the DME should be between two meters to four meters. We can see that the RSS generated base result is more close to the Cramer-Rao lower bound. On the other hand, the result of distance based, either empirical or model based, is not optimized. We choose to use RSS based nearest neighbor as our primary localization algorism. That means in order to generating an RSS database, we need to know the reference points' location. And for test points, we need to know the RSS. In our case, it is easy to know the selected points' location as long as we have the map. We don't even need to have a field trip. As for localization, we just need to collect the RSS to compare with the database. It is a convenient way of generating and collecting data.

4.3.2 CRLB for Single iBeacon

With iBeacon, you are giving the option to change the transmit power. Each transmit power offers a different range of coverage. We tested four different iBeacon transmit powers to determine which transmit power would be best to use. The graph below shows the results of the testing of different transmit powers.

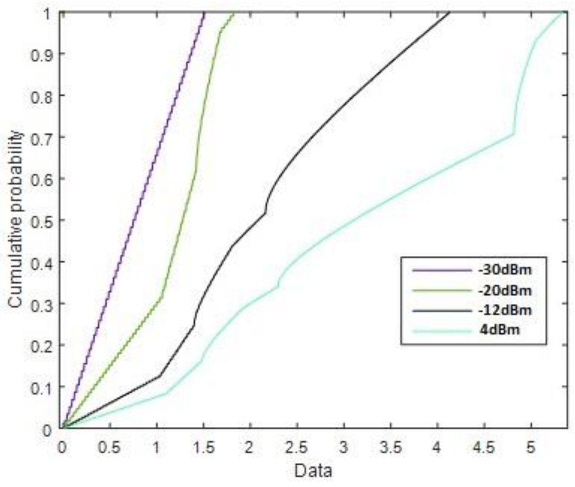


Figure 4.3.2: CRLB for Single iBeacon with Different Transmit Powers

In the graph above, the blue line represents 4dBm, the black line represents -12dBm, the green line represents -20dBm and the purple line represents -30dBm. When the transmit power was set to 4dBm, the iBeacon was able to cover our whole database, compared to when the transmit power was -30dBm, which only covered about 10% of our database. This shows that the higher the transmit power is, the more area the iBeacon is able to cover. In the graph above, we can see that the distance measurement error decreases as the transmit power decrease.

4.3.3 CRLB for Hybrid Localization

Additionally to this testing, we also tested the behavior of the different transmit powers when iBeacon was used for localization with Wi-Fi. The results from this testing is shown in the graph below.

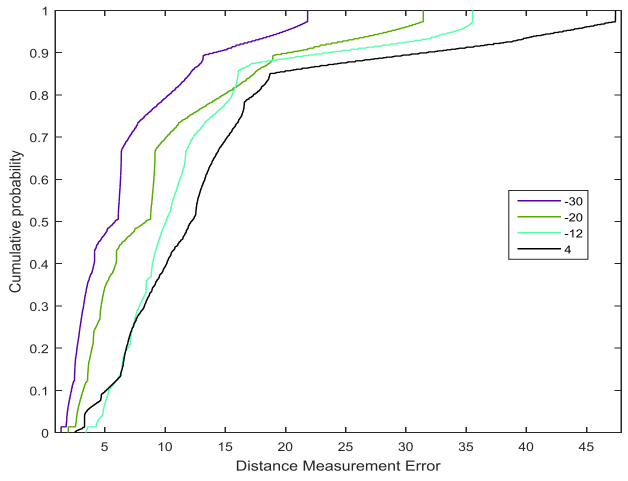


Figure 4.3.3.1: One iBeacon with Different Transmit Powers Combined with Four Wi-Fi APs

In the graph, 4dBm is represented by the black line, -12dBm is represented by the blue line, -20dBm is represented by the green line, and -30dBm is represented by the purple line. The results followed the same trend from the previous testing. The higher the transmit power was, the more accurate it was. Using this information, we decided that setting the iBeacon's transmit power to -20dBm would give us the best results. With this transmit power; we would be able to cover a good portion of our database, while still remaining pretty accurate.

We tested the different scenarios to see the effect that iBeacon had on Wi-Fi localization. We did the testing in a small enough area that we would be in the coverage area of two iBeacons. In this area we used CRLB to simulate the results of localization of Wi-Fi by itself, Wi-Fi with one iBeacon and Wi-Fi with two iBeacons. The graph below shows the results from our simulation.

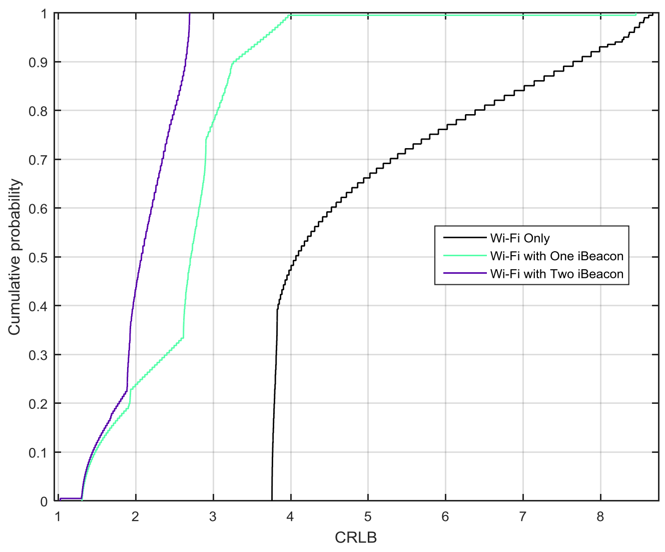


Figure 4.3.3.2: Performance of Hybrid Localization in Ideal Scenario of Only Wi-Fi, One iBeacon with Wi-Fi and Two iBeacons with Wi-Fi

In the graph above, the black line represents Wi-Fi by itself, the green line represents Wi-Fi with one iBeacon, and the purple line represents Wi-Fi with two iBeacons. You can see that when Wi-Fi was the only source of localization, the CRLB is between about 4 to 9 meters. When one iBeacon is introduced to scenario the CRLB decreases to the range of 1 to about 4 meters and improves Wi-Fi's localization by 47%. When a second iBeacon is introduced the CRLB decreases to about 1 to 2 meters and improves Wi-Fi's localization by 57%. When Wi-Fi was the only form of localization, the DME is very large compared to when iBeacons are used with Wi-Fi. From the graph it is clear that the number of iBeacons has a positive effect on the localization of Wi-Fi when they are used with Wi-Fi for localization. As the chart shows

	Mean	Median	Std	Percentage of Improvement to Wi-Fi
Two	2.08	2.07	0.4	57%
One	2.6	2.7	0.77	47%
Wi-Fi only	4.94	4.06	1.53	

Table 4.3.3: Performance of Hybrid Localization in Ideal Scenario

4.4 iBeacon Configuration

4.4.1 Percentage Coverage Results

After we used CRLB tested the accuracy of iBeacon in different transmit power, we want to know the coverage of iBeacon. Since we can adjust transmit power of iBeacon from -30dBm to -4dBm, we chose 3 transmit powers to calculate the percentage of coverage. The three transmit powers are -30dBm, -20dBm and -4dBm and the results are given as followed.

Pt(dBm)	Probability
-4	1
-20	0.3
-30	0.1

Table 4.4.1.1: Empirical Probability of Coverage for Three Transmit Powers

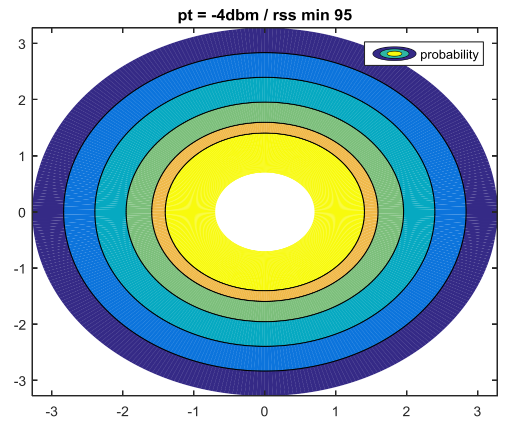


Figure 4.4.1.1: Calculated Probability of Coverage for -4dBm

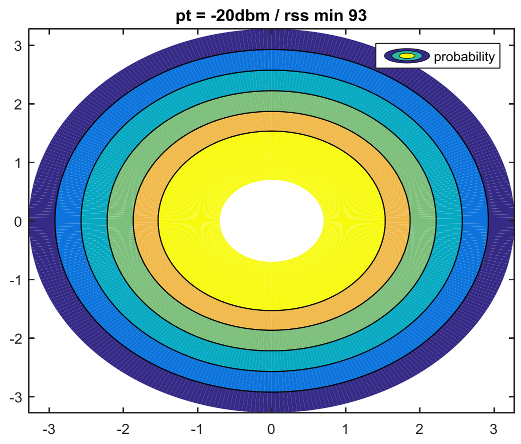


Figure 4.4.1.2: Calculated Probability of Coverage for -20dBm

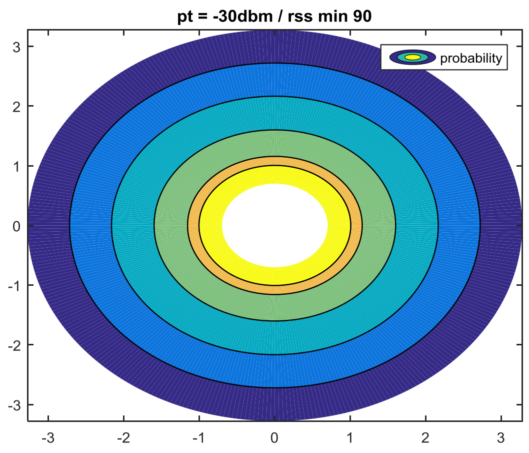


Figure 4.4.1.3: Calculated Probability of Coverage for -30dBm

For the theoretical coverage calculation, we take points from 3 meters to 0 meter gradually. For the empirical coverage, we take 100 points around our test area with around 70 perimeters. It appears that with a higher transmit power, the higher coverage iBeacon can have. One feature of iBeacon is that the higher transmit power iBeacon has, the lower battery life it has. For our project, we want to find a best transmit power that can maximize the accuracy of hybrid localization and save more battery.

Pt(dBm)	Coverage(m)	Std
4	26	3.34
-12	20	3.15
-20	8	3.00
-30	5	2.64

Table 4.4.1.2: Accuracy, Coverage and Transmit Power Relationship

From the chart above, we can find the pattern. As the transmit power increases, the coverage increases, but the accuracy decreases. At this point, based on the calculated percentage of coverage and empirical coverage, we can assume -20 dBm is the choice provisionally. Since 8 meters coverage and variance of 3 is food enough for our purpose.

4.5 Results of Hybrid Localization of Wi-Fi and iBeacon

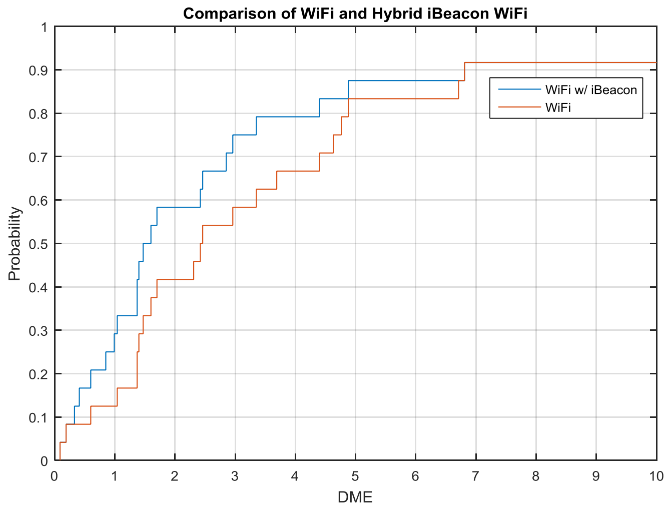


Figure 4.5.1: Localization Distance Measure Error Comparison of Only Wi-Fi and Hybrid Wi-Fi and iBeacon Result

As shown in the figure above, we can see that after adding iBeacon into the scenario, the overall accuracy is increased about 1 to 2 meters. This means the method we use to hybrid iBeacon and Wi-Fi localization is effective. And at the same time, we can see that adding iBeacon to the indoor Wi-Fi localization can improve the accuracy.And then, we apply a new algorithm and the result is

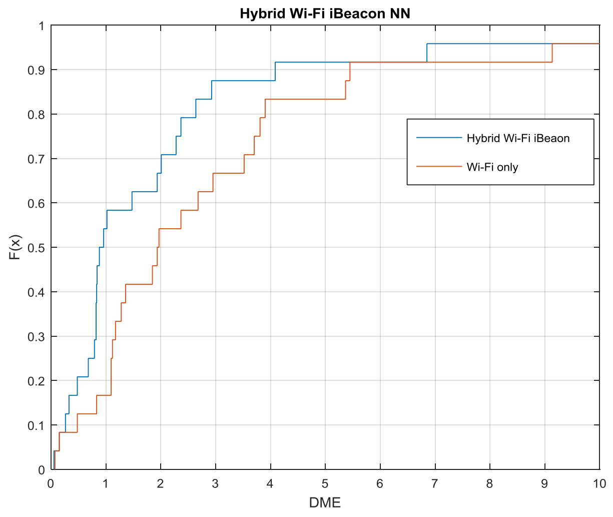


Figure 4.5.2: Weighted Nearest Neighbor Hybrid Localization Comparison of Only Wi-Fi and Hybrid Wi-Fi and iBeacon Result

In the graph above, the orange line represents Wi-Fi by itself and the blue line represents Wi-Fi hybrid with iBeacons. The graph shows the distance measurement error for both forms of localization. Just from looking at the graph, it's clear that the hybrid Wi-Fi and iBeacon there are improvements compared to just Wi-Fi only. When iBeacon was added to Wi-Fi for localization, the accuracy of just Wi-Fi's localization was improved by about 35%. As the chart showing below.

	Mean	Median	Std	Percentage of Improvement to Wi-Fi
Hybrid	1.98	0.92	2.63	0.352941
Wi-Fi only	3.06	1.95	3.46	

Table 4.5: Hybrid Localization Statistic Data

Chapter 5: Conclusion and Future Work

5.1 Conclusion

We developed an algorithm and implemented an application to determine whether or not someone has entered or exited the room. We also implemented distance estimation and push notification functions of iBeacon. Now, it is possible to achieve the scenario of a student finding a professor's office. The student can get distance by using the app. When this student passes by the office, he can know if he enters the office region. He can also get information about how many people in the room to determine if it is a right time to get into the office.

We designed an algorithm for hybrid iBeacon and Wi-Fi localization and showed that iBeacon improves the accuracy of Wi-Fi localization by about 30%. iBeacon has an outstanding performance in improving the accuracy of localization.

5.2 Future Work

In the future, it is needed to construct a web server to store information gathered from iPhone and messages to be broadcasted. At that time, we could include all the functions we implemented into one app which can realize

We have proved that iBeacon can significantly improve Wi-Fi localization theoretically and empirically by using nearest neighbor algorithm with generated database. However, since iBeacon technology is still in developing, iBeacon has some shortage such as unstable transmission. In the future, it is also possible to do some research and improvement on iBeacon to improve its own stability and accuracy. In order to improve the performance of iBeacon in localization.

Reference

[1] K. Pahlavan and P. Krishnamurthy, Principles of Wireless Access and Localization, Wiley, November 2013

[2] "What Is Global Positioning System (GPS)? - Definition from WhatIs.com." SearchMobileComputing. N.p., n.d. Web: http://goo.gl/pj8IUV

[3] "Indoor/Outdoor Localization." Indoor/Outdoor Localization. Web: http://goo.gl/F40lxw

[4] Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". The American Statistician 46 (3): 175–185.

[5] A. Fujihara and T. Yanagizawa, "Proposing an Extended iBeacon System for Indoor Route Guidance," Intelligent Networking and Collaborative Systems (INCOS), 2015 International Conference on, Taipei, 2015, pp. 31-37.

[6] Tao Peng, Xinhong Wang, Chao Wang and Danqing Shi, "Hybrid wireless indoor positioning with iBeacon and Wi-Fi," 11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015), Shanghai, 2015, pp. 1-5.

[7] X. Y. Lin, T. W. Ho, C. C. Fang, Z. S. Yen, B. J. Yang and F. Lai, "A mobile indoor positioning system based on iBeacon technology," 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, 2015, pp. 4970-4973.

[8] Zahradnik, Fred. "How Does a Wi-Fi Positioning System Work?". Web. http://goo.gl/y3uo2t

[9] Estimote, “Developer Docs”. Web. http://developer.estimote.com/

[10] M. Kanaan, B. Alavi, A. Hatami, K. Pahlavan, "Channel Modeling and Algorithms for Positioning in Indoor Wireless Networks", to appear in Springer Encyclopedia on Geographical Information Science, 2007.

[11] Kavel Pahlavan, Prashant Krishnamurthy and Yishuang Geng“Localization Challenges for the Emergence of the Smart World” IEEE Access, vol:3, pp 1, January, 2016.

[12] Chuka Ebi, Sebastian Franco, and Tianxiong Wang, ECE, Hybrid Wi-Fi Localization using RFID and UWB sensors , MQP, May 2015

[13] D. Van Merode, G. Tabunshchyk, K. Patrakhalko and G. Yuriy, "Flexible technologies for smart campus," *2016 13th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Madrid, 2016, pp. 64-68.

[14] E. Nakamori *et al*., "A new indoor position estimation method of RFID tags for continuous moving navigation systems," *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, Sydney, NSW, 2012, pp. 1-8.

[15] Yang, Yang, Zhouchi Li, and Kaveh Pahlavan. "Using iBeacon for intelligent in-room presence detection.” *In Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2016 IEEE International Multi-Disciplinary Conference on*, pp. 1-5. IEEE, 2016.

[16] Li, Zhouchi, Yang Yang, and Kaveh Pahlavan. “Using iBeacon for Newborns Localization in Hospitals.” *In Medical Information and Communication Technology (ISMICT), 2016 IEEE 10th International Symposium on*, pp. 1-5. IEEE, 2016.

Appendices

In the appendix we provide the MATLAB code used for CRLB calculation, generated distance based database and calculated DME and CDF, and counting probability of coverage for iBeacon. We also include the Objective C code used to develop our iBeacon application.

Appendix A: MATLAB Code for CRLB Calculation

```
x=[0,0;15.31,17.49;32.04,17.49;45.37,-10.59];
SD=[16.37,17.21];
bp=5;
e=8.3
for i=[1:1:4]
   r(i)=sqrt((SD(1)-x(i,1)).^2+(SD(2)-x(i,1)).^2);
   if r(i)<5;
       a1=2;
       P(i)=-10*a1/log(10)*((SD(1)-x(i,1))/(r(i).^2));
       Q(i)=-10*a1/log(10)*((SD(2)-x(i,2))/(r(i).^2));
       H(i,1)=P(i);
       H(i,2)=Q(i);
   end
  if r(i)>=5;
       a2=3.5;
       P(i)=-10*a2/log(10)*((SD(1)-x(i,1))/(r(i).^2));
       Q(i)=-10*a2/log(10)*((SD(2)-x(i,2))/(r(i).^2));
       H(i,1)=P(i);
       H(i,2)=Q(i);
end
end
C=e^2*inv(H'*H)
```

The x represents the coordinates of the access point and the SD represents the coordinates of the test point. bp is the break point distance and the e is the variance.

Appendix B: MATLAB Code to Generate Distance Based Database and Calculate the DME and CDF

```
clear all;
% Database declare
measured_data = [-62,-49,-72,-63;-62,-52,-70,-64;-59,-48,-72,-64;-62,-43,-68,-67;-64,-34,-63,-72;-72,-41,-56,-67;-
77,-46,-50,-61;-82,-56,-46,-56;-85,-55,-36,-53;-89,-59,-36,-48;-85,-72,-48,-49;-84,-73,-55,-46;-82,-77,-60,-41;-82,-
76,-62,-43;-87,-76,-66,-48;-85,-82,-66,-52;-84,-80,-66,-47;-82,-73,-71,-60;-72,-70,-75,-67;-68,-63,-78,-68];
test_db = [-58,-50,-73,-63;-62,-33,-60,-67;-70,-37,-50,-63;-77,-47,-48,-57;-77,-52,-46,-52;-80,-53,-37,-51;-80,-53,-
39,-50;-76,-53,-52,-44;-75,-53,-63,-34;-72,-53,-63,-50;-70,-75,-63,-47;-72,-72,-74,-54;-68,-63,-76,-61;-60,-47,-78,-
68;-63,-46,-75,-67;-62,-50,-75,-65;-58,-47,-67,-61;-60,-45,-69,-61;-63,-37,-64,-67;-70,-33,-64,-67;-73,-38,-50,-67;-
77,-46,-45,-60;-77,-47,-44,-57;-77,-53,-39,-49];
database_loc                                                                                              =
[0,3;0,5.56;0,7.54;0,10.12;0,12.62;4.12,17.21;7.99,17.21;13.61,17.21;17.74,17.21;21.4,17.21;21.93,12.66;21.93,10.
75;21.93,7.89;21.93,5.28;21.93,2.55;19.13,0;15.56,0;11.62,0;6.85,0;2.55,0];
test_db_loc                                                                                               =
[0,5.23;0,15.58;4.31,17.21;10.41,17.21;15.31,17.21;19.21,17.21;21.93,14.71;21.93,9.39;21.93,3.49;19.44,0;13.1,0;8
.22,0;2.64,0;0,1.4;0,0.73;0,4.19;0,6.69;0,9.71;0,13.66;1.27,17.21;4.72,17.21;8.98,17.21;12.21,17.21;16.37,17.21];
generated_data         =         [-64.6621559800000,-60.1557446000000,-69.5075210600000,-67.3594911300000;-
64.0275458300000,-56.9780061000000,-68.6189873100000,-67.1532946700000;-63.9455796700000,-
53.9645212000000,-68.0150203800000,-67.1788209300000;-64.3940762900000,-48.8500399500000,-
67.3804595100000,-67.4513056100000;-65.3348081100000,-42.9319419100000,-66.9753664400000,-
67.9483989000000;-70.6917881200000,-43.4014971300000,-63.2148122300000,-66.7149773100000;-
73.1221740500000,-52.1531942600000,-58.7412306500000,-64.0807703600000;-76.1986528400000,-
60.2172718300000,-48.5344320400000,-60.4158366400000;-78.1678898500000,-64.2387118400000,-
39.3882011400000,-58.8369193800000;-79.7426323100000,-67.0867265500000,-41.8906475300000,-
59.0524265300000;-79.5204898500000,-67.6896423500000,-44.5323148500000,-51.9374956100000;-
79.4071140100000,-67.9690844900000,-47.9147660800000,-47.7500461200000;-79.3219331400000,-
68.5579117200000,-53.7584856500000,-43.1294496300000;-79.3351174400000,-69.2386236200000,-
57.8302692400000,-42.3362440700000;-79.4412867400000,-70.0561977700000,-61.2333039100000,-
46.1609333100000;-78.3890098200000,-69.6297934600000,-63.6084616500000,-50.1472228700000;-
76.6645131800000,-67.9968067600000,-63.9874519800000,-52.2570827200000;-74.5209211300000,-
66.2315504800000,-65.1931467200000,-56.9046637600000;-71.4921753300000,-64.3908191700000,-
67.2989168200000,-62.1436811200000;-68.2257161700000,-63.3696332700000,-69.4029171500000,-
65.9755490600000];

tb_points = size(test_db_loc,1);
db_points = size(measured_data, 1);
```

```
replication_num = 10;
sigma = 5.648;

% 4 separate dataset, with Gaussian noise
data_set1 = measured_data(:,1); % the first column
data_set2 = measured_data(:,2);
data_set3 = measured_data(:,3);
data_set4 = measured_data(:,4);



matrix_1=[];
for j=1:db_points,
   tmp_row = normrnd(data_set1(j), sigma, [1,replication_num]);  %generate 10 random numbers for each element
   matrix_1 = [matrix_1; tmp_row]; %populate the array
end

matrix_2=[];
for j=1:db_points,
   tmp_row = normrnd(data_set2(j), sigma, [1,replication_num]);  %generate 10 random numbers for each element
   matrix_2 = [matrix_2; tmp_row]; %populate the array
end

matrix_3=[];
for j=1:db_points,
   tmp_row = normrnd(data_set3(j), sigma, [1,replication_num]);  %generate 10 random numbers for each element
   matrix_3 = [matrix_3; tmp_row]; %populate the array
end

matrix_4=[];
for j=1:db_points,
   tmp_row = normrnd(data_set4(j), sigma, [1,replication_num]);  %generate 10 random numbers for each element
   matrix_4 = [matrix_4; tmp_row]; %populate the array
end

i=1; j=1; k=1;
sum_result_tmp = []; returned_loc=[];dme=[]; sum_result_mat=[];  %initialize empty arrays
```

```
sum_result_mat=zeros(200,24);
for k=1:tb_points,
    % the test database, 10x4 matrix
    tmp = test_db(k,:); % get the first row
    for i=1:db_points,
        for j = 1:replication_num,
            % extract the corresponding (i,j) element in each matrix. 1x4 matrix
            % compare row wise
            foo = [matrix_1(i,j),matrix_2(i,j), matrix_3(i,j),matrix_4(i,j)];
            sum_compare = sum((foo - tmp).^2); %The sum of square of difference
            sum_result_mat((i-1)*10+j,k) = sum_compare; %200x1 row matrix
        end


    end


%     [val, loc] = min(sum_result_mat(k,:));  % val is the min value, loc is row number
%     row_tmp = 1+ floor(loc/replication_num); % the row in the original matrix database


%     %returned_loc = [returned_loc; database_loc(row_tmp,:)]; % a 10x1 matrix containing the locations
end
 [M,I]=min(sum_result_mat);
    row_tmp = ceil(I/10);
for k=1:tb_points,
    display(row_tmp);


    loc = row_tmp(:,k);
    display(loc);
    dme_foo = sqrt(sum((database_loc(loc,:)-test_db_loc(k,:)).^2));
    dme = [dme; dme_foo];
end
display(dme);
cdfplot(dme);
```

Appendix C: MATLAB Code to Count Probability of Coverage for iBeacon

```
%sigma
std_4 = std(rss_4');
std_20 = std(rss_20');
std_30 = std(rss_30');

foo = pt(1)*ones(5,10);

sita=0:1:360;
x=cosd(sita)'*d';
y=sind(sita)'*d';

%probobility
p1 = mean((1-0.5*erfc((lp_max(1)*ones(5,10) -(pt(1)*ones(5,10)-rss_4))./(sqrt(2)*std_4'*ones(1,10))))');
p2 = mean((1-0.5*erfc((lp_max(2)*ones(5,10) -(pt(2)*ones(5,10)-rss_20))./(sqrt(2)*std_20'*ones(1,10))))');
p3 = mean((1-0.5*erfc((lp_max(3)*ones(5,10) -(pt(3)*ones(5,10)-rss_30))./(sqrt(2)*std_30'*ones(1,10))))');
display(p1);
display(p2);
display(p3);
plot(d,p1');
grid on;
%title ();
z1=ones(361,5);
z2=ones(361,5);
z3=ones(361,5);
for i=1:1:5
    z1(:,i)=p1(i);
    z2(:,i)=p2(i);
    z3(:,i)=p3(i);
end

figure(1)
[C1,h1]=contourf(x,y,z1,5);
legend('probability');
title('pt = -30dbm / rss min 90');
figure(2)
```

```
[C2,h2]=contourf(x,y,z2,5);
legend('probability');
title('pt = -20dbm / rss min 93');
figure(3)
[C3,h3]=contourf(x,y,z3,5);
legend('probability');
title('pt = -4dbm / rss min 95');
```

Appendix D: Code for the application for iBeacon

AppDelegate.m file

```
#import <UIKit/UIKit.h>
#import "AppDelegate.h"

@interface AppDelegate ()

@end

@implementation AppDelegate


- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions {

    if ([UIApplication instancesRespondToSelector:@selector(registerUserNotificationSettings:)]){
        [application registerUserNotificationSettings:[UIUserNotificationSettings
settingsForTypes:UIUserNotificationTypeAlert|UIUserNotificationTypeBadge|UIUserNotificationTypeS
ound categories:nil]];
    }

    return YES;
}

- (void)applicationWillResignActive:(UIApplication *)application {
    // Sent when the application is about to move from active to inactive state. This can occur for certain types of temporary interruptions (such as an incoming phone call or SMS message) or when the user quits the application and it begins the transition to the background state.
    // Use this method to pause ongoing tasks, disable timers, and throttle down OpenGL ES frame rates. Games should use this method to pause the game.
}

- (void)applicationDidEnterBackground:(UIApplication *)application {
    // Use this method to release shared resources, save user data, invalidate timers, and store enough application state information to restore your application to its current state in case it is terminated later.
    // If your application supports background execution, this method is called instead of applicationWillTerminate: when the user quits.
}

- (void)applicationWillEnterForeground:(UIApplication *)application {
    // Called as part of the transition from the background to the inactive state; here you can undo many of the changes made on entering the background.
}

- (void)applicationDidBecomeActive:(UIApplication *)application {
    // Restart any tasks that were paused (or not yet started) while the application was inactive. If the application was previously in the background, optionally refresh the user interface.
}

- (void)applicationWillTerminate:(UIApplication *)application {
    // Called when the application is about to terminate. Save data if appropriate. See also applicationDidEnterBackground:.
    // Saves changes in the application's managed object context before the application terminates.
    [self saveContext];
}

#pragma mark - Core Data stack

@synthesize managedObjectContext = _managedObjectContext;
```

```
@synthesize managedObjectModel = _managedObjectModel;
@synthesize persistentStoreCoordinator = _persistentStoreCoordinator;

- (NSURL *)applicationDocumentsDirectory {
    // The directory the application uses to store the Core Data store file. This code uses a directory named
"com.ibeacons.phillydev716.iBeaconsYoutube" in the application's documents directory.
    return [[[NSFileManager defaultManager] URLsForDirectory:NSDocumentDirectory
inDomains:NSUserDomainMask] lastObject];
}

- (NSManagedObjectModel *)managedObjectModel {
    // The managed object model for the application. It is a fatal error for the application not to be able to
find and load its model.
    if (_managedObjectModel != nil) {
        return _managedObjectModel;
    }
    NSURL *modelURL = [[NSBundle mainBundle] URLForResource:@"iBeaconsYoutube"
withExtension:@"momd"];
    _managedObjectModel = [[NSManagedObjectModel alloc] initWithContentsOfURL:modelURL];
    return _managedObjectModel;
}

- (NSPersistentStoreCoordinator *)persistentStoreCoordinator {
    // The persistent store coordinator for the application. This implementation creates and return a
coordinator, having added the store for the application to it.
    if (_persistentStoreCoordinator != nil) {
        return _persistentStoreCoordinator;
    }

    // Create the coordinator and store

    _persistentStoreCoordinator = [[NSPersistentStoreCoordinator alloc]
initWithManagedObjectModel:[self managedObjectModel]];
    NSURL *storeURL = [[self applicationDocumentsDirectory]
URLByAppendingPathComponent:@"iBeaconsYoutube.sqlite"];
    NSError *error = nil;
    NSString *failureReason = @"There was an error creating or loading the application's saved data.";
    if (![_persistentStoreCoordinator addPersistentStoreWithType:NSSQLiteStoreType configuration:nil
URL:storeURL options:nil error:&error]) {
        // Report any error we got.
        NSMutableDictionary *dict = [NSMutableDictionary dictionary];
        dict[NSLocalizedDescriptionKey] = @"Failed to initialize the application's saved data";
        dict[NSLocalizedFailureReasonErrorKey] = failureReason;
        dict[NSUnderlyingErrorKey] = error;
        error = [NSError errorWithDomain:@"YOUR_ERROR_DOMAIN" code:9999 userInfo:dict];
        // Replace this with code to handle the error appropriately.
        // abort() causes the application to generate a crash log and terminate. You should not use this function
in a shipping application, although it may be useful during development.
        NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
        abort();
    }

    return _persistentStoreCoordinator;
}


- (NSManagedObjectContext *)managedObjectContext {
    // Returns the managed object context for the application (which is already bound to the persistent store
coordinator for the application.)
    if (_managedObjectContext != nil) {
        return _managedObjectContext;
    }

    NSPersistentStoreCoordinator *coordinator = [self persistentStoreCoordinator];
    if (!coordinator) {
        return nil;
    }
    _managedObjectContext = [[NSManagedObjectContext alloc] init];
    [_managedObjectContext setPersistentStoreCoordinator:coordinator];
    return _managedObjectContext;
}

#pragma mark - Core Data Saving support

- (void)saveContext {
    NSManagedObjectContext *managedObjectContext = self.managedObjectContext;
    if (managedObjectContext != nil) {
        NSError *error = nil;
        if ([managedObjectContext hasChanges] && ![managedObjectContext save:&error]) {
            // Replace this implementation with code to handle the error appropriately.
            // abort() causes the application to generate a crash log and terminate. You should not use this
function in a shipping application, although it may be useful during development.
            NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
            abort();
        }
    }
}

@end
```

ViewController.m file

```
#import "ViewController.h"
#import <CoreBluetooth/CoreBluetooth.h>
#import <CoreLocation/CoreLocation.h>
#import <EstimoteSDK/ESTBeaconManager.h>




@interface ViewController () <ESTBeaconManagerDelegate>

@property (nonatomic, strong) CLBeacon *beacon;
@property (nonatomic, strong) ESTBeaconManager *beaconManager;
```

```
@property (nonatomic, strong) CLBeaconRegion *beaconRegion;


@end

@implementation ViewController

int notice_num=0;

- (void)viewDidLoad {
    [super viewDidLoad];

    rssiValueBefore = [[NSMutableArray alloc] initWithCapacity:15];
    rssiValueAfter = [[NSMutableArray alloc] initWithCapacity:15];



    self.beaconLabel.text = @"";
    //Create your UUID
    NSUUID *uuid = [[NSUUID alloc] initWithUUIDString:@"E7EACB3A-A38B-8787-5301-
CF49E2661E34"];

    //set up the beacon manager
    self.beaconManager = [[ESTBeaconManager alloc] init];
    self.beaconManager.delegate = self;



    //set up the beacon region
    self.beaconRegion = [[CLBeaconRegion alloc] initWithProximityUUID:uuid
                                                                 major:13579
                                                                 minor:24680
                                                            identifier:@"ice2"];

    //let us know when we exit and enter a region
    self.beaconRegion.notifyEntryStateOnDisplay=YES;
    self.beaconRegion.notifyOnEntry = YES;
    self.beaconRegion.notifyOnExit = YES;
    [self.beaconManager requestAlwaysAuthorization];
    //start  monitorinf
    [self.beaconManager startMonitoringForRegion:self.beaconRegion];

    //start the ranging
    [self.beaconManager startRangingBeaconsInRegion:self.beaconRegion];

    //MUST have for IOS8
    [self.beaconManager requestAlwaysAuthorization];



}

//check for region failure
-(void)beaconManager:(ESTBeaconManager *)manager monitoringDidFailForRegion:(CLBeaconRegion
*)region withError:(NSError *)error
{
    NSLog(@"Region Did Fail: Manager:%@ Region:%@ Error:%@",manager, region, error);
}

//checks permission status
-(void)beaconManager:(ESTBeaconManager                                    *)manager
didChangeAuthorizationStatus:(CLAuthorizationStatus)status
{
    NSLog(@"Status:%d", status);
}

//Beacon manager did enter region
- (void)beaconManager:(ESTBeaconManager *)manager didEnterRegion:(CLBeaconRegion *)region
{
    //Adding a custom local notification to be presented
    UILocalNotification *notification = [[UILocalNotification alloc]init];
    notification.alertBody = @"Youve done it!";
    notification.soundName = @"Default.mp3";
    NSLog(@"Youve entered");
    [[UIApplication sharedApplication] presentLocalNotificationNow:notification];

}

//Beacon Manager did exit the region
- (void)beaconManager:(ESTBeaconManager *)manager didExitRegion:(CLBeaconRegion *)region
{
    //adding a custon local notification
    UILocalNotification *notification = [[UILocalNotification alloc]init];
    notification.alertBody = @"Youve exited!!!";
    NSLog(@"Youve exited");
    [[UIApplication sharedApplication] presentLocalNotificationNow:notification];
}


-(void)beaconManager:(ESTBeaconManager *)manager didRangeBeacons:(NSArray *)beacons
inRegion:(CLBeaconRegion *)region
{
    if (beacons.count > 0) {
        CLBeacon *firstBeacon = [beacons firstObject];
        self.beaconLabel.text = [self textForProximity:firstBeacon.proximity];
        if([rssiValueBefore count] == 3){
            if([rssiValueAfter count] < 10){
                [rssiValueAfter addObject:[NSNumber numberWithInteger: [firstBeacon rssi]]];
            }else{
                [rssiValueBefore removeObjectAtIndex:0];
                [rssiValueBefore addObject:[rssiValueAfter objectAtIndex:0]];
                [rssiValueAfter removeObjectAtIndex:0];
                [rssiValueAfter addObject:[NSNumber numberWithInteger: [firstBeacon rssi]]];
            }
        }else{
            [rssiValueBefore addObject:[NSNumber numberWithInteger: [firstBeacon rssi]]];
        }

        NSNumber *avgBrfore = [rssiValueBefore valueForKeyPath:@"@avg.self"];
        NSNumber *avgAfter = [rssiValueAfter valueForKeyPath:@"@avg.self"];

        NSLog(@"%@  + %@", avgBrfore, avgAfter);

        _status.text  =  [NSString  stringWithFormat:@"Before  Avg  %@  After  Avg  %@",  avgBrfore,
avgAfter];



        if (avgAfter != nil && [rssiValueAfter count] > 2) {
            if ((([avgAfter floatValue] - [avgBrfore floatValue]) >= 10) ) {
                _rangeLabel.text= @"Going in door";

                if(![_rangeLabel.text isEqualToString:@"Going in door"]){
                    UILocalNotification *notification = [[UILocalNotification alloc]init];
                    notification.alertBody = @"Going in door";
                    notification.soundName = @"Default.mp3";
                    [[UIApplication sharedApplication] presentLocalNotificationNow:notification];
                }else{
                    NSLog(@"nah in");
                }

            }else if((([avgAfter floatValue] - [avgBrfore floatValue]) <= -10) ){
                if(![_rangeLabel.text isEqualToString:@"Going out door"]){
                    UILocalNotification *notification = [[UILocalNotification alloc]init];
                    notification.alertBody = @"Going out door";
                    notification.soundName = @"Default.mp3";
                    [[UIApplication sharedApplication] presentLocalNotificationNow:notification];
                }else{
                    NSLog(@"nah out ");
                }

                _rangeLabel.text= @"Going out door";
            }else{
                _rangeLabel.text= @"Not moving";
                notice_num = 0;
            }

        }else{

        }
    }
}

-(NSString *)textForProximity:(CLProximity)proximity
{

    switch (proximity) {
        case CLProximityFar:
            //NSLog(@"far");
            return @"Far";
            break;
        case CLProximityNear:
//          NSLog(@"near");
            self.beaconLabel.textColor = [UIColor purpleColor];
            return @"Near";
            break;
        case CLProximityImmediate:
//          NSLog(@"immediate");
            self.beaconLabel.textColor = [UIColor redColor];
            return @"Immediate";
            break;
        case CLProximityUnknown:
//          NSLog(@"unknown " );
```

```
            return @"Unknown";
            break;
        default:
            break;
    }
}

@end
```