

Learning Indoor Localization using Radio Received Signal Strength

by

Gauri Kulkarni

B.E., University of Pune, India, 2011

A thesis submitted to the Graduate Faculty of the

University of Colorado at Colorado Springs

in partial fulfillment of the

requirements for the degree of

Master of Science in Computer Science

Department of Computer Science

College of Engineering and Applied Science

2016

© Copyright By Gauri Kulkarni 2016

All Rights Reserved

This thesis for Master of Science degree by

Gauri Kulkarni

has been approved for the

Department of Computer Science

by

Dr. Jonathan Ventura

Dr. Edward Chow

Dr. Terrance Boulton

Date

Kulkarni, Gauri (M.S., Computer Science)

Learning Indoor Localization using radio received signal strength

Thesis directed by Dr. Jonathan Ventura.

With this research we will investigate a novel machine learning approach to the prediction of location from received signal strength indicators (RSSI) values obtained from these transmitting access points. Indoor localization has been a long-standing problem in recent times and gaining popularity among researchers. In this research we aim to solve this problem in an indoor environment like office buildings using radio received signals strengths. The most popular approach for positioning has been GPS (Global Positioning System). But we all know that it is inadequate when we consider indoor environments. Hence to solve this issue; we make use of the radio received signal strengths. The most common technology used for indoor positioning is Wi-Fi, which uses radio signals as its signal propagation medium. In this research we are proposing to create an indoor localization system radio signal strengths from as low- energy BLE technology from Bluetooth as access points that were easily available, where the locations of these access points will be unknown. The RSSI obtained from these beacons will be used to predict the locations using machine-learning algorithms. For evaluating our theory we are using the classic fingerprinting method as our baseline for the evaluations. To evaluate this we considered the classic algorithm of nearest neighbor, which is used as a classic method for implementing fingerprinting.

DEDICATION

To

Dad, Mom and Tum

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor Dr. Jonathan Ventura of Department of Computer Science at University of Colorado at Colorado Springs. I would always knock on his office door whenever I came across a hurdle during my thesis and he would tirelessly answer my queries and guide me in the right direction whenever he thought I needed it.

I would also like to thank Dr. Terrance Boult and Dr. Edward Chow for being a part of my committee and providing me with their valuable advice and input, without which the validation and evaluation would not have been efficient.

I would also like to give my thanks to Alessandra Langfels for her constant words of encouragement and guidance towards completing this degree.

I would also like to extend my special thanks to my lab mate Yousef for his guidance and support throughout.

Finally, I must express my heartfelt gratitude to my parents and to my sister, Uma and friends – Brandon, Roser, Marc, Bhakti, Baburao and Dubari for dealing with my constant ups and downs through the process of working on and writing this thesis. Thank you all.

Gauri Kulkarni

Contents

Contents.....	vii
List of Tables.....	viii
List of Figures	x
Chapter 1: Introduction	1
1.1 Overview	1
1.2 Research Goals and Approach.....	3
1.3 Theory.....	4
Chapter 2: Literature survey.....	9
Chapter 3: Machine learning algorithms.....	12
3.1 Selection of algorithms	12
Chapter 4: Experiments.....	14
4.1 Dataset Creation	14
4.2 Data Analysis	21
Chapter 5: Results	23
5.1 Unique RSSI and Unique Locations	23
5.2 Cross validation and ANOVA tests	32
Chapter 6: Conclusion.....	49
6.1 Research Conclusion	49
6.2 Future Work	50
References	51

List of Tables

Table 4-1: Sample of data recorded from Bluetooth beacons	20
Table 5-1: Average Error for Unique RSSI (Near Beacons Dataset)	25
Table 5-2: Average Error for Unique RSSI (Far Beacons Dataset)	26
Table 5-3: Average Error for Unique Locations (Near Beacons Dataset)	28
Table 5-4: Average Error for Unique Location (Far Beacons Dataset)	29
Table 5-5: Average Error for Unique RSSI Using K-fold (Near Beacons Dataset)	33
Table 5-6: Average Error for Unique RSSI Using K-fold (Far Beacons Dataset)	34
Table 5-7: Average Error for Unique Locations Using K-fold (Near Beacons Dataset)	35
Table 5-8: Average Error for Unique Locations Using K-Fold (Far Beacons Dataset)	36
Table 5-9: Input data with k=5 independent treatments	40
Table 5-10: Descriptive statistics of treatments	40
Table 5-11 One -way ANOVA of your k=5 independent treatments	41
Table 5-12: Tukey HSD results	41
Table 5-13: Input data with k=5 independent treatments	42
Table 5-14: Descriptive statistics of treatments	42
Table 5-15 One -way ANOVA of your k=5 independent treatments	43
Table 5-16: Tukey HSD results	43
Table 5-17: Input data with k=5 independent treatments	44
Table 5-18: Descriptive statistics of treatments	44
Table 5-19 One -way ANOVA of your k=5 independent treatments	45
Table 5-20: Tukey HSD results	45
Table 5-21: Input data with k=5 independent treatments	46
Table 5-22: Descriptive statistics of treatments	46

Table 5-23 One -way ANOVA of your $k=5$ independent treatments	47
Table 5-24: Tukey HSD results.....	47

List of Figures

Fig. 1-1: Trilateration	5
Fig. 1-2: Triangulation	6
Fig. 1-3: RF fingerprinting	7
Fig. 4-1: Bluetooth Beacons	14
Fig. 4-2: Gimbal Beacon Manager	15
Fig. 4-3: Raspberry Pi	16
Fig. 4-4: Lab setup with beacons in the inside of the lab partition panels	17
Fig. 4-5: Lab setup with beacons in the outside of the lab partition panels	17
Fig. 4-6: (A) (B) (C) (D) - 3D plots of RSSI values from beacon 1,2,3,4	21
Fig. 5-1: Classification vs. Nearest Neighbor for near Bluetooth data	27
Fig. 5-2: Classification vs. Nearest Neighbor for far Bluetooth dataset	27
Fig. 5-3: Regression vs. Classification for near Bluetooth dataset	31
Fig. 5-4: Regression vs. Classification for far Bluetooth dataset	31
Fig. 5-5: Regression vs. Classification for far Bluetooth dataset	37
Fig. 5-6: Regression vs. Classification for far Bluetooth dataset	38

Chapter 1:Introduction

1.1 Overview

Indoor localization has been gaining much popularity among researchers and big software companies like Google [1] and Apple [1]. There are several companies that offer indoor navigation in locations such as airports [2]. There is a big competition for indoor localization currently in the software market between start-ups as well as established companies.

Indoor localization has been a long-standing problem during recent years. This can be applied in many fields of interest like retail market, security, military and hospitals. Losing one's way in hospitals and airports is a very common human error. Indoor positioning system at hand will help issues like these very easily.

With this research we aim to investigate a solution for indoor localization using off-the-shelf communicative equipment. To resolve this problem we propose a novel machine learning approach that uses machine-learning techniques such as regression and classification algorithms using received radio signal strength as a measurement parameter.

Use of radio signals and RSSI has been a well-known method for positioning and localization. This method is easy for implementation and popular for predicting the position even if the location of the access points is unknown. Use of RSSI based devices could prove to show a good performance for accuracy and availability at nominal costs for infrastructures.

GPS signals are inadequate for receivers in indoor environments, like inside a building, because the GPS signals are not easily penetrable through solid material and if there is no clear

line of sight [27]. For developing such a system for indoor localization we chose to use radio signals. They can be translated into a quantifiable parameter in the form of RSSI (Radio Signal Strength Indicator). The choice of RSSI was appropriate because RSSI is very easily adaptable with different systems or computational techniques such as machine-learning techniques, which will be discussed in detail in coming sections of the thesis.

For this study we are investigating a novel machine learning approach for the indoor localization. There are several techniques that make use of RSSI for indoor positioning for example trilateration [3], triangulation [4], fingerprinting [5] and many others.

This research will be conducted in three phases. The steps can be outlined as follows:

1) Collection of data 2) Analysis of data 3) Prediction of location coordinates.

The first stage involves collection of RSSI data and storing it in csv files. We are making use of 2.4 GHz, series-10 proximity Bluetooth beacon. For the datasets from Bluetooth beacons, values are collected as RSSI signal samples over a coordinate grid in the indoor environment. In this case we are creating the test bed in the UCCS VAST lab. In the second stage we will analyze the data collected from both Bluetooth beacons and the Wi-Fi access point. The third and final stage is doing the prediction of coordinates for the Bluetooth signal receiver based on the RSSI datasets. In this study we are mainly making use of the machine learning approach for the following benefits:

1. To avoid the use of a fixed signal to distance model for localization
2. To eliminate the need to know the positions of the access points for the RSSI.
3. To produce a continuous (rather than discrete) location estimate.

This thesis presents the results from an investigational study about an indoor localization system using radio received signal strength using machine-learning techniques. The RSSI is derived from low energy Bluetooth beacons and is received on a Raspberry pi. The datasets are analyzed for their behavior in an indoor environment. Further we discuss the selection of the machine learning algorithms. Finally we explain how the random forest technique outperforms the classic localization approach of nearest neighbor for fingerprinting

1.2 Research Goals and Approach

The main purpose of this study is to create a system for indoor localization that does not require the conversion of radio RSSI into distance. There are several conventional methods used for localization. In this investigation we are trying to analyze if we can perform indoor localization without using RSSI as a function of distance as well as analyze and compare the different machine learning techniques like regression, classification and nearest neighbors algorithms which are most commonly used for the indoor localization by fingerprinting.

The main objectives of this study are as follows:

1. Indoor positioning with RSSI without the knowledge of the access points.
2. Indoor positioning with RSSI without deriving distance from it.
3. Analyzing the use of different machine learning algorithms and suggesting the best out of them for the prediction of the locations based on the training data.
4. Indoor positioning system using low cost off the shelf communicating equipment.

1.3 Theory

1.3.1 Characteristics of Radio Received Signal strengths

Since the introduction of this indoor localization problem, the use of sensors for the signal transmission for indoor positioning systems has also increased. If we observe our surrounding, we can find a variety of wireless device networks around us. And many of them use radio frequency signals as their medium of signal transmission. For example: Wi-Fi, which transmits radio signals using the 2.4-gigahertz (12 cm.) UHF and 5 gigahertz (6 cm.) SHF ISM radio bands [6]. So using the RSSI was a feasible method for an indoor environment. Using RSS is also an appropriate choice as it is easily available with most wireless devices around us, without costing an overhead for infrastructure. The RSSI is a measure of power level of a radio signal in a wireless network. It is usually measured as decibels or a number from the range 0 to 100, this value could be either negative or positive. The values closer to the number zero represent a stronger power level for signals [7].

The following formula could be used for RSSI and since we have the information for RSSI, we can use this to derive distance from the access point [8]:

$$\text{RSSI} = -(10n\log_{10}d) + A$$

Where:

n : Signal propagation constant

d: Distance from the access point

A: Received signal strength at 1 meter distance

1.3.2 Techniques using RSS for indoor localization

There are several techniques that use the RSS signals for indoor localization. Most of these techniques use the known positions of the access points (signal transmitters). Some of the most popular techniques using RSSI signals for indoor positioning are as follows:

A) Trilateration

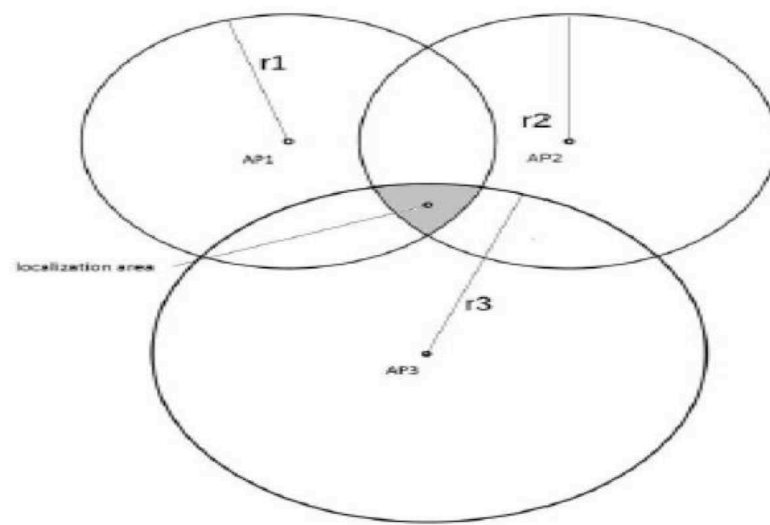


Fig. 1-1: Trilateration

One of the most popular methods used for indoor positioning is trilateration. Trilateration is a method of determining the relative location of a device from two known points of references based on the measurement of distances and using the intersection of the circles around the reference points [9]. The points of references could be signal transmitters such as cellphone towers or Wi-Fi access points. This method is very efficient when the RSS is converted to distance and then the location at an indoor location can be easily determined. The GPS uses trilateration as its method for location services [11]. At any given point the GPS uses three satellites and performs trilateration to locate.

B) Triangulation

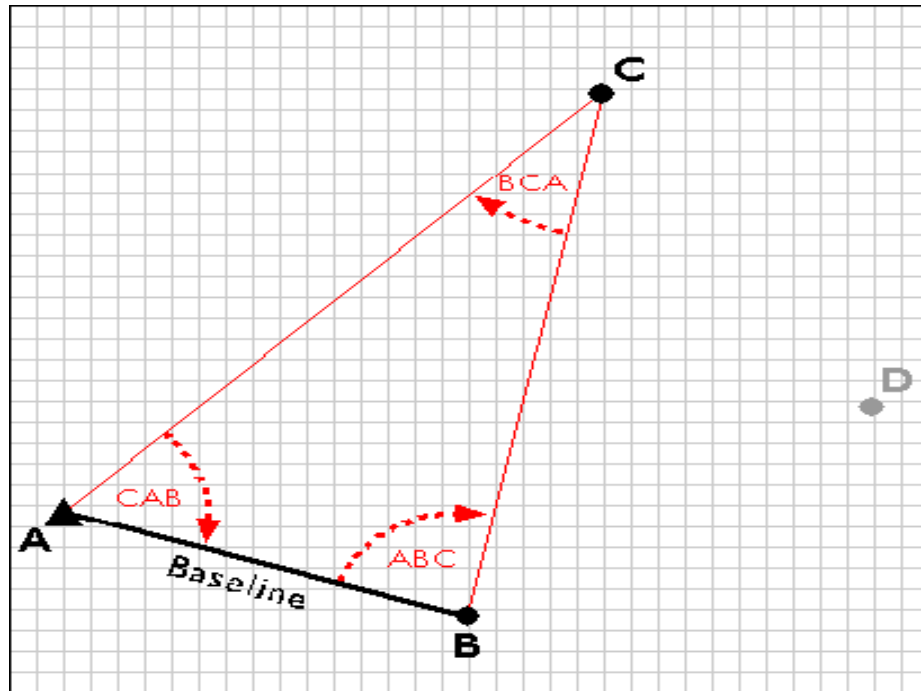


Fig. 1-2: Triangulation

Triangulation refers to the process of localization of a point or a device by measuring the angles to it from known access points at either end of a fixed baseline, which could be a known distance between two known reference points or access points. This method however does not involve the calculation of distances.

If we observe, both the above-mentioned methods, they require the known positions of the access points or the reference points. Angulation locates an object with angles with respect to the multiple reference points. The most common methods for positioning using triangulation is by using time of arrival (TOA) [26] or by using the time difference of arrival (TDOA) instead of converting the RSS directly into distance [26].

C) Fingerprinting

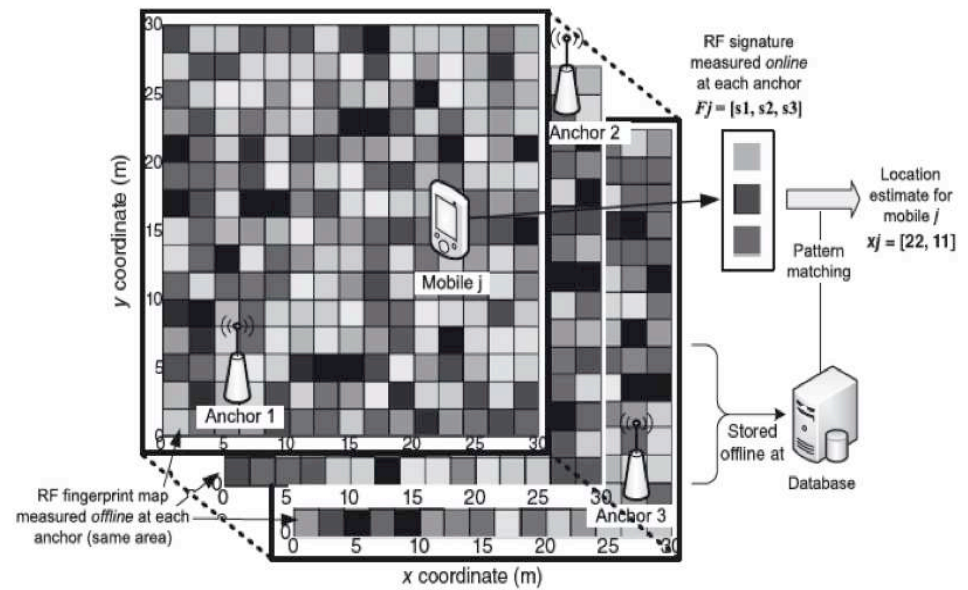


Fig. 1-3: RF fingerprinting

The radio frequency fingerprinting is a technique is useful when the radio signals vary over the area in question as well as the different locations in that area. The above figure [12] shows the processes appropriately. The whole process is divided onto two phases:

a) Offline Training Phase

In this phase, the signal information is recorded at each reference point and stored in database of radio maps [12].

b) Online Localization Phase

In this phase the signals from the device for localization is compared to the radio maps using several patterns matching techniques like K nearest neighbors and other. The location is selected based on the best match or by interpolating between the best matches. [12]

Chapter 2:Literature survey

There are several techniques that can be utilized for creating an indoor positioning system. Indoor positioning can be very useful in several retail scenarios and emergency situations that require context awareness and location aware information for either navigation or only the localization information.

There are several research approaches for achieving the indoor localization. With the prevalence of the wireless networks around us, there is a great opportunity for indoor positioning. The Wi-Fi infrastructure is one of the most popular and readily available technologies as a solution for this problem. Since it transmits radio signals, creating a signal-to-distance model from the RSSI is a well-known approach. Redpin [13] create such an indoor localization system, which is similar to fingerprinting. The main objective of their research is to eliminate the offline training phase [13] before the online fingerprint-matching phase. However since we are planning to use machine-learning approach for our research, we require the creation of a training dataset so that the algorithms can learn for predicting the test sets correctly. Another important method for fingerprinting is using the k nearest neighbor algorithm. Kodipalli et.al. [20] have given a very good methodology of integrating the fingerprinting and Trilateration. Their method does not require the known locations of the access points. They are using the area around the access points as the range of area between the access points and the target device/location. However they do require calculating the distance for their modified trilateration method.

Wi-Fi triangulation and trilateration method is another popular technique used for indoor localization with or without fingerprinting. Fingerprinting is a technique where we measure radio signals and create radio maps. Navarro et al. [21] have used RSSI fingerprinting and triangulation for localization. However using triangulation is easy for implementation but could be harder during real time prediction of location as it uses RSSI as a function of distance, as also fingerprints can have a drift with time. In the research indoor localization method based on Wi-Fi trilateration technique by Schchekotov [24] describes the use of a Wi-Fi for trilateration for the localization in an indoor environment. He is using the Wi-Fi access points for developing a system of equations to get the location using trilateration. He is using the RSS measurements for the distance measurements.

Since we are using BLE technology for this research, we have discussed the research that is using Bluetooth. Bluetooth Indoor Positioning by Korona et al. [22] had based their theory on the advertising capability of the Bluetooth beacons. They are using a signal to distance-based model for multi-lateration for localization. Similarly for the localization of autonomous robots in an indoor environment, Raghavan et al. [23] in their research, Accurate mobile robot localization in indoor environments, are using RSSI and the distance from the Bluetooth access points. They are using USB Bluetooth dongles as their hardware and they are using Bluetooth inquiry protocol for their distance estimates. For the localization they are using trilateration method. However in both the above-discussed research they are using the known locations of the Bluetooth devices. Since they are using the signal to distance model for their localization algorithm there will be an issue with the multipath signal propagation of Bluetooth devices.

There are several research papers that survey the various techniques and systems used and built for indoor localization. Liu et.al. [26] , in their analysis called the survey of wireless indoor positioning techniques and systems have given analysis about indoor positioning using a plethora of wireless system networks and techniques used for localization. But in their paper they have stressed more on the fingerprinting method, which is the most widely used method for localization used in many localization systems. According to the authors, fingerprinting is mostly used as a conventional method for indoor positioning in an open area and indoor positioning using the RFID tags is the best so far an indoor localizations. [26] They also suggest that the more density of the radio signals signal transmitters will increase the rate of accuracy in a localization system.

Chapter 3: Machine learning algorithms

3.1 Selection of algorithms

For our experiments we have considered using regression, classification and nearest neighbors. We are using the tool kit Scikit learn [25] for implementing the machine learning techniques. After observing the recorded data we can discern that the data is not fit for a linear curve, nor do we see a Gaussian pattern from the 3D plots in Fig. 3-8. Hence our hypothesis is that tree based regression will outperform the linear regression curve. The candidates we chose for testing our results are as follows:

3.1.1 Random Forest Regression

3.1.2 Random Forest Classifier

3.1.3 Extra Tree Regression

3.1.4 Extra Tree Classifier

3.1.5 Nearest Neighbor

Regression and classification can both be used for the prediction of variables where regression is used, which have a continuous value dataset, and classification is used with values that could be categorized into separate classes. With the regression algorithms we can test the prediction of locations based on interpolation between the location values. We want to explore this with the help of two trials whether regression and classification would perform better than the conventional method of nearest neighbors for fingerprinting.

The selection of the algorithms can be validated using fair experiments. Based on the analysis of the data and speculation over the selection of algorithms, we have arrived on two hypotheses.

1. Random forest classification will outperform nearest neighbor in determining location from an RSSI vector recorded at a 'previously seen' location.
2. Random forest regression will outperform random forest classification in determining location from an RSSI vector recorded at a 'previously unseen' location.

Chapter 4: Experiments

4.1 Dataset Creation

Since we are making use of the RSS from the given access points, we wanted to make use of technology that can be easily available and not very expensive for establishing a new infrastructure altogether. Hence we chose to make use of the low energy, low cost Bluetooth beacons by Qualcomm [14] as our transmitters for the initial indoor experiments. To further increase the scalability of this theory and to use a wider area, we chose the RSS obtained from the Wi-Fi access points around the campus as an appropriate choice.

4.1.1 Hardware used

A) Bluetooth Beacons



Fig. 4-1: Bluetooth Beacons [14]

The Fig 4-1 shows the Gimbal Qualcomm Bluetooth beacons, which run on the BLE technology. The transmission power of this Bluetooth beacon has a range from -23dBm to 0dBm, where 0dBm represents the full power. The transmission range of the beacon is about 50 meters with a clear line-of-sight condition [15]. They run on a small replaceable CR2032 battery. Hence these little beacons are very low cost and ideal for a smaller area. They can be easily configured using the Gimbal beacon manager as shown in Fig 4-3. They operate using the iBeacon technology [16].













My Beacons											
Search				Show 10 entries							
Icon	Name	Hardware	Factory ID	Firmware	Battery Level	Visibility	Type	Assigned Configuration	Applied Configuration	Tags	Actions
	Beacon 3		5HKU-4YPEB	1.8.0	Low	Public	iBeacon	iBeacon 1.3	iBeacon 1.3		Edit Delete
	Beacon 4		X61F-4HE15	1.8.0	Low	Public	iBeacon	iBeacon 1.4	iBeacon 1.4		Edit Delete
	Beacon 2		KGQ5-Z5V83	1.8.0	Low	Public	iBeacon	iBeacon 1.2	iBeacon 1.2		Edit Delete
	Beacon 1		Q1SN-28NP5	1.8.0	Low	Public	iBeacon	Beacon 1.1	Beacon 1.1		Edit Delete
	Beacon 6		FVJQ-YN82	1.8.0	Low	Private	iBeacon	iBeacon 1.6	iBeacon 1.6		Edit Delete
	Beacon 5		F5K7-V2KHN	1.8.0	Med-LOW	Public	iBeacon	iBeacon 1.5	iBeacon 1.5		Edit Delete
Showing 1 to 6 of 6 entries										First Previous 1 Next Last	

Fig. 4-2: Gimbal Beacon Manager

B) Raspberry Pi



Fig. 4-3: Raspberry Pi

The first and the foremost quality of the Pi is it is affordable, and easily available. Its compact design makes it another good choice. For our experiments we are making use of the Pi as our receiving device, when we attach a USB Bluetooth dongle. The SD card storage in the Pi enables us to easily record the received RSSI that can also be easily retrieved. The Pi can run easily on a power supply at 5V. So for conducting our experiments we used a 7V rechargeable battery pack with a 5V DC-DC regulator. This made the data-collection relatively quicker and easier. We are making use of a USB Bluetooth adapter with Bluetooth 4.0 technology [18]. This device is very useful for a simple connective just to serve the purpose of collecting the RSS values for the study.

4.1.2 Lab setup

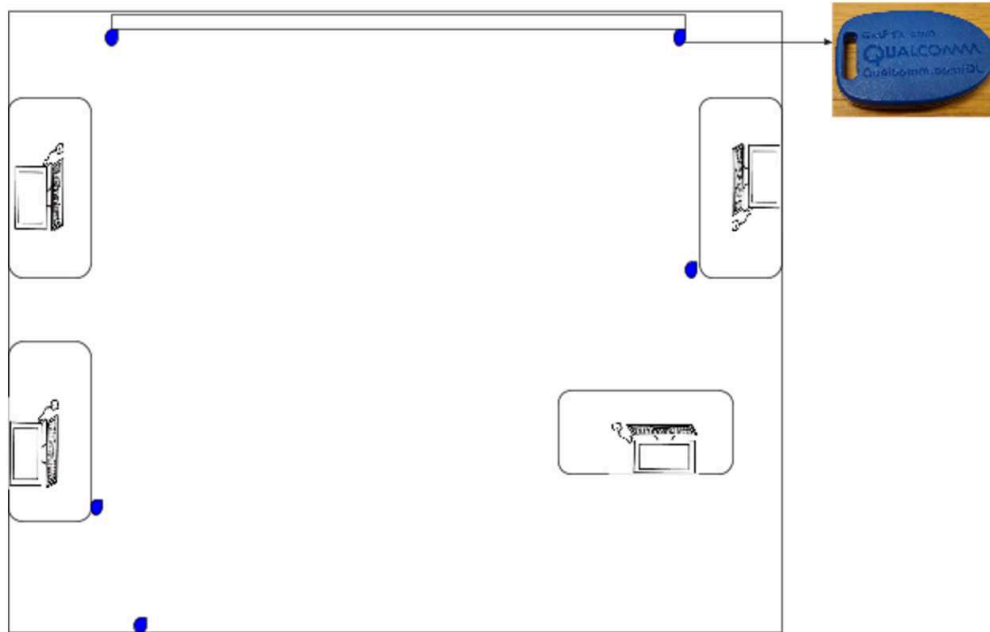


Fig. 4-4: Lab setup with beacons in the inside of the lab partition panels

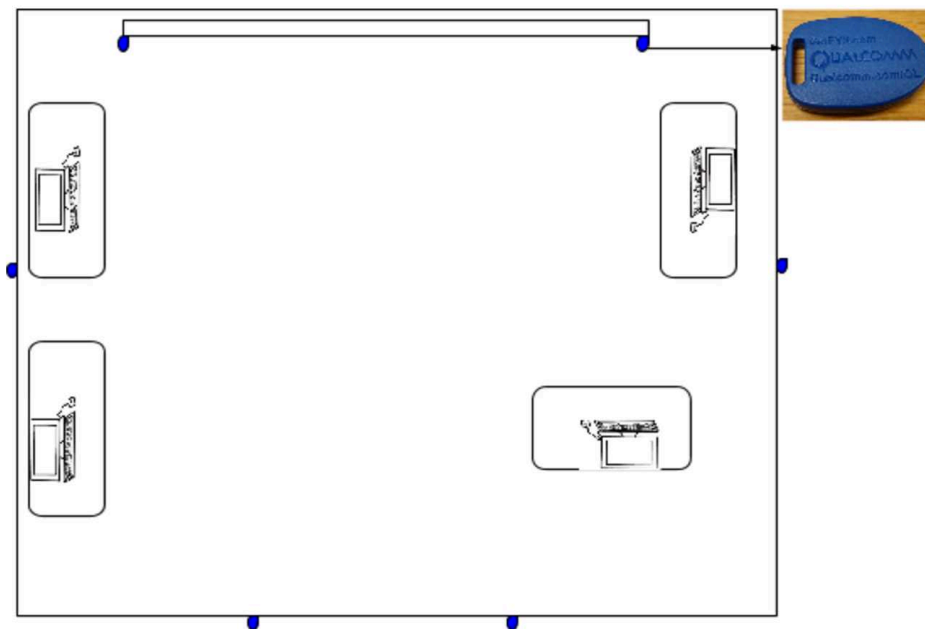


Fig. 4-5: Lab setup with beacons in the outside of the lab partition panels

Fig. 4-4 shows that the Bluetooth beacons are placed on the inside of the lab partition panels and the Fig. 4-5 shows that they are placed on the outside of the partition panels of the lab. These two scenarios give us different behavioral patterns of the RSS with respect to the indoor environment. The area used as a test bed is approximately 15x15 Sq. Feet.

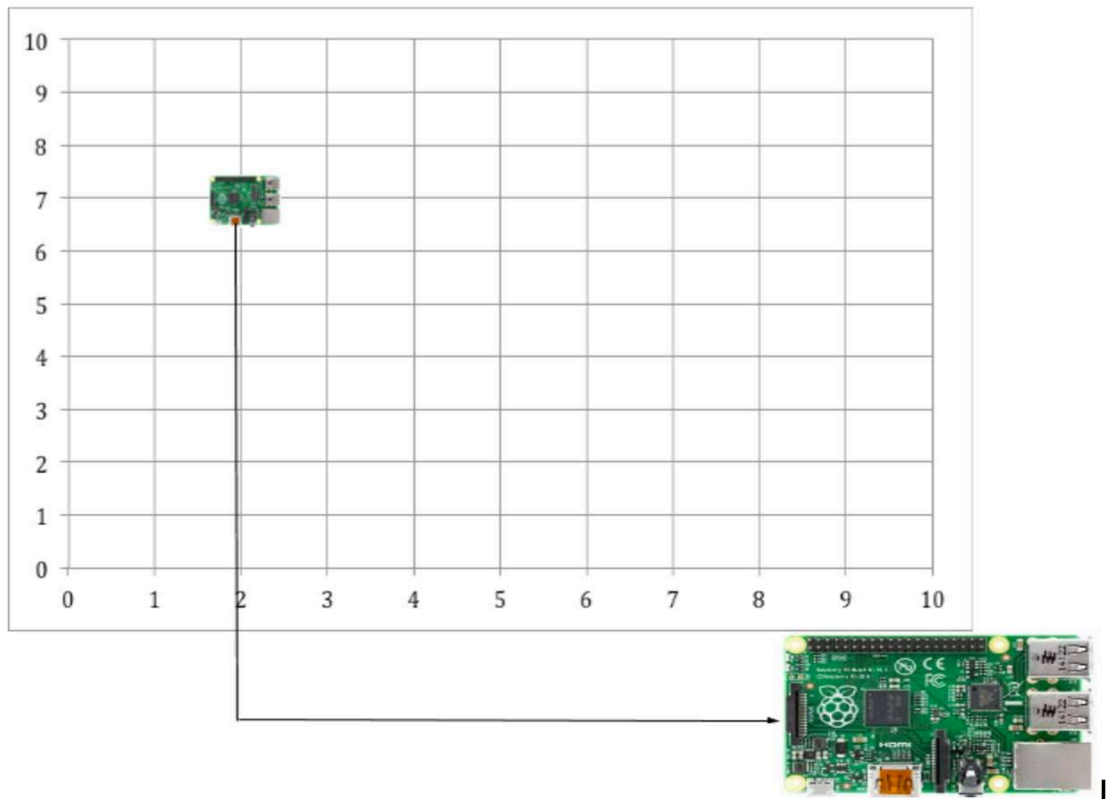


Fig. 4-6: Grid setup with Raspberry Pi (10x10 Sq. Feet)

We created an imaginary grid using the floor tiles in the lab, which are about one square foot each. The Fig 4-7 shows this in a schematic fashion. For calculating the RSSI, we set up an imaginary grid across the lab. We make this grid as a 2D coordinate plane. Since we are using the Pi as our receiver of the radio signals, to record the RSSI values, we are placing the Pi at each of the points on the coordinate plane.

4.1.3 Bluetooth beacons Dataset recordings

We will record two different datasets for training and testing for the machine-learning algorithm. We have two separate datasets for Bluetooth beacons. The first one is where the beacons are placed on the inner panels of the lab and the second one is when the beacons were placed on the outer panels of the lab.

During the first dataset recording, Pi was placed on the coordinate point and a hundred scans were performed each time. During each scan about twelve measurements were recorded. For the second dataset, we recorded the data by placing the Pi on the point and picking it up ten times. During each positioning, ten scans were performed and about twelve measurements were recorded.

The recorded values will be created in the form of a tuple (0,0,1,-70,2,-69,...5,-70) which is in the form ((x,y) coordinates, minor number of beacon (b1), RSSI beacon (b1) , minor number of beacon (b2), RSSI beacon (b2) minor number of beacon (bn), RSSI beacon (bn)) and is stored in csv files.

In Table 4-1, the first two columns represent the (x,y) coordinates from the grid. The columns 3 to 7 show the collected RSSI values from beacons 1 to 5. The first two columns along with the other columns represent the tuples for the data recorded from the Bluetooth beacons.

x	y	Rssi1	Rssi2	Rssi3	Rssi4	Rssi5
0	0	-82	-76	-86	-87	-96
0	0	-82	-76	-86	-87	-96
0	0	-82	-78	-86	-87	-96
0	0	-82	-78	-86	-87	-96
0	0	-82	-78	-86	-87	-96
0	0	-82	-75	-86	-87	-96
0	0	-82	-75	-86	-87	-96
0	0	-82	-75	-87	-87	-96
0	0	-82	-75	-87	-87	-89
0	0	-82	-75	-87	-87	-89
0	0	-82	-78	-87	-87	-89
0	0	-82	-78	-87	-87	-89
0	0	-82	-78	-100	-87	-89
0	0	-82	-78	-100	-87	-89
0	0	-82	-78	-100	-87	-96
0	0	-82	-75	-100	-87	-96

Table 4-1: Sample of data recorded from Bluetooth beacons.

4.2 Data Analysis

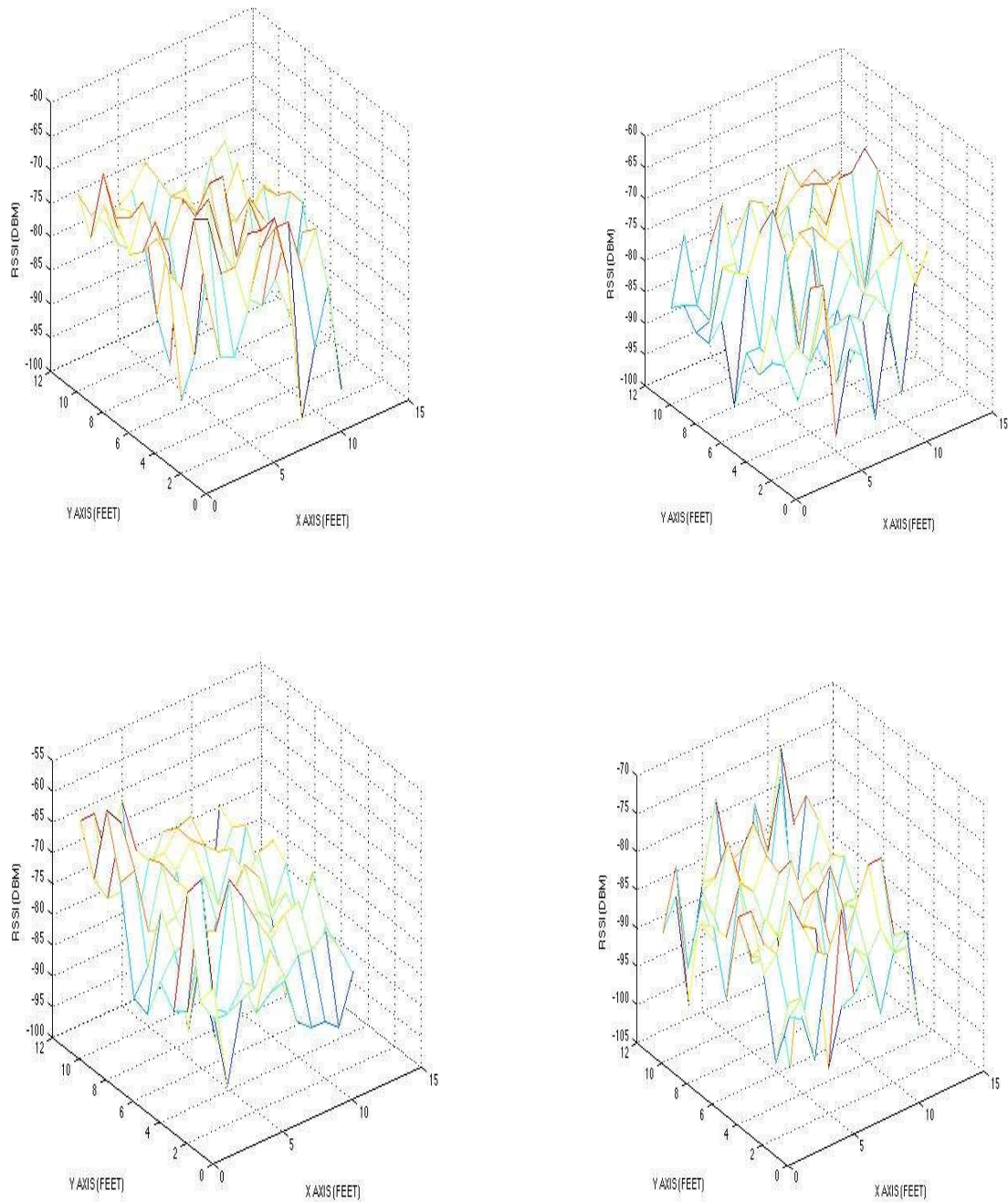


Fig. 4-6: (A) (B) (C) (D) - 3D plots of RSSI values from beacon 1,2,3,4

The figures Fig. 4-6 A, B, C, D show the RSSI values in the test bed. The X-axis and the Y-axis represent the 10 feet by 10 feet test bed area from the lab. The Z-axis shows the RSSI values. For ample coverage we had kept the Bluetooth beacons on the four panel walls of the lab. Therefore we can see that the RSSI obtained by the receiver are not concentrated in any one particular direction or area. We can see that they do not in a certain pattern. Since we are taking the RSSI recordings in a real environment, there is a lot of noise in the environment. The range of RSSI that we recorded was -66 to -100.

Chapter 5:Results

After analyzing the signals and their pattern, we came to reason whether the signals would fit with a linear regression. The reason for speculating this with regression is because it is possible to interpolate or extrapolate the signals (RSSI) or the location values based on the training set. Therefore we formulated the tables shown in figures from Fig 5-1 to Fig 5-4. They show the values of the average error of the location prediction based on the different algorithms for our analysis. We have formulated the following tables, which show the entries for variable test sizes from 90 percent to 10 percent. The tables also show the entries for the standard deviation.

There are two separate datasets for near beacons and far beacons. They are divided as unique locations as well as unique RSSI value tuples.

5.1 Unique RSSI and Unique Locations

The values computed using the “Unique RSSI” means that the values from the dataset in this table do not contain any duplicate RSSI vector. This helps with the Classification and Nearest Neighbor algorithms to accurately predict the locations without having repeated RSSI vectors.

The values that are computed using the “Unique Locations ” means that the values from the dataset in this table do not have duplicate locations. This means that when we split the dataset using this dataset we will not have overlapping locations in the test and train splits. These unique locations will be useful in experiments used for interpolation or extrapolation.

The Table 5-1 shows the values from the dataset with the beacons placed on the inside of the lab partition panels. The table has average error values, obtained by varying the test dataset split size from 90 percent to 10 percent. The Table 5-2 shows the same data as mentioned above but computed with the dataset, which has, the beacons placed on the outside of the panels.

The Table 5-3 and 5-4 have the values that show the average errors computed with the datasets having unique locations and hence the test and train sets do not contain the values of location coordinates overlapping. The Table 5-3 has the average errors calculated from the RSSI with the beacons placed inside the partition panels from the lab space and the Table 5-4 has the average errors of the dataset with beacons placed on the outside of the partition panels in the test space.

Table 5-1: Average Error for Unique RSSI (Near Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	1.66	1.52	0.9	1.01	2.1
0.8	1.35	1.41	0.8	0.7	1.83
0.7	1.18	1.36	0.7	0.59	1.71
0.6	1.06	1.31	0.6	0.49	1.59
0.5	0.97	1.27	0.5	0.45	1.51
0.4	0.9	1.23	0.4	0.4	1.43
0.3	0.84	1.2	0.3	0.36	1.36
0.2	0.8	1.18	0.2	0.33	1.3
0.1	0.74	1.14	0.1	0.3	1.24

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	1.54	1.44	0.9	0.89	2	0.9	1.64	2.59
0.8	1.24	1.32	0.8	0.6	1.71	0.8	1.25	2.37
0.7	1.07	1.26	0.7	0.49	1.57	0.7	1.05	2.23
0.6	0.96	1.2	0.6	0.41	1.45	0.6	0.9	2.09
0.5	0.87	1.15	0.5	0.37	1.38	0.5	0.8	1.99
0.4	0.81	1.11	0.4	0.33	1.3	0.4	0.7	1.91
0.3	0.75	1.08	0.3	0.29	1.22	0.3	0.66	1.82
0.2	0.71	1.05	0.2	0.26	1.16	0.2	0.61	1.75
0.1	0.65	1.01	0.1	0.24	1.13	0.1	0.58	1.72

Table 5-2: Average Error for Unique RSSI (Far Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	2.64	1.61	0.9	2.31	2.72
0.8	2.4	1.6	0.8	1.8	2.59
0.7	2.21	1.56	0.7	1.49	2.47
0.6	2.07	1.54	0.6	1.27	2.34
0.5	1.94	1.52	0.5	1.1	2.22
0.4	1.84	1.49	0.4	0.95	2.1
0.3	1.75	1.47	0.3	0.83	1.99
0.2	1.68	1.43	0.2	0.75	1.89
0.1	1.6	1.39	0.1	0.67	1.83

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	2.53	1.64	0.9	2.05	2.67	0.9	3.03	2.95
0.8	2.23	1.6	0.8	1.48	2.45	0.8	2.57	2.94
0.7	2.01	1.56	0.7	1.13	2.24	0.7	2.26	2.89
0.6	1.84	1.52	0.6	0.92	2.07	0.6	2.02	2.8
0.5	1.7	1.47	0.5	0.73	1.9	0.5	1.84	2.74
0.4	1.58	1.44	0.4	0.58	1.71	0.4	1.71	2.7
0.3	1.48	1.39	0.3	0.48	1.58	0.3	1.59	2.64
0.2	1.39	1.32	0.2	0.41	1.47	0.2	1.47	2.59
0.1	1.29	1.25	0.1	0.32	1.3	0.1	1.39	2.55

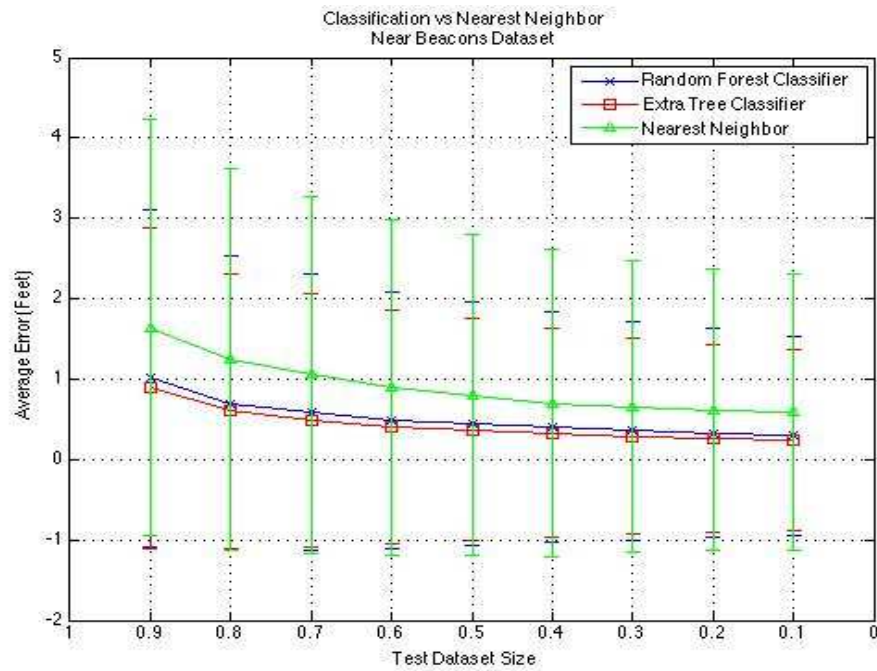


Fig. 5-1: Classification vs. Nearest Neighbor for near Bluetooth data

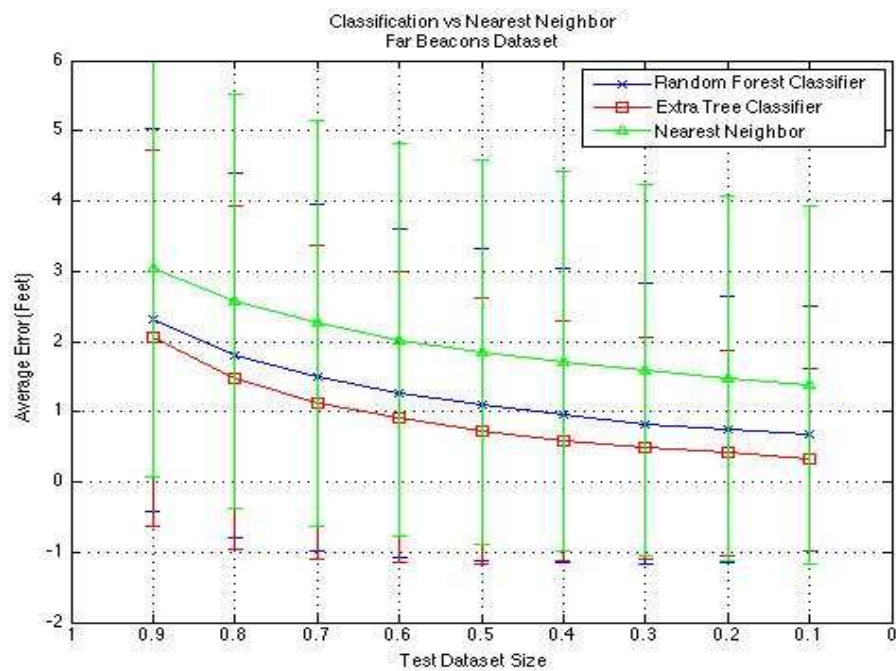


Fig. 5-2: Classification vs. Nearest Neighbor for far Bluetooth dataset

Table 5-3: Average Error for Unique Locations (Near Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	4.19	2	0.9	4.56	2.49
0.8	3.76	1.89	0.8	4.48	2.5
0.7	3.82	1.92	0.7	4.51	2.52
0.6	3.89	1.99	0.6	4.52	2.54
0.5	3.8	2.06	0.5	4.62	2.57
0.4	3.52	1.95	0.4	4.63	2.55
0.3	3.49	1.95	0.3	4.57	2.56
0.2	3.34	1.88	0.2	4.39	2.61
0.1	3.22	1.86	0.1	4.4	2.5

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	3.84	1.92	0.9	4.59	2.5	0.9	4.72	2.5
0.8	3.83	1.9	0.8	4.17	2.3	0.8	4.42	2.42
0.7	3.59	1.75	0.7	4.12	2.27	0.7	4.37	2.39
0.6	3.59	1.84	0.6	4.1	2.34	0.6	4.41	2.43
0.5	3.51	1.8	0.5	4.05	2.35	0.5	4.39	2.43
0.4	3.32	1.81	0.4	3.83	2.36	0.4	4.21	2.42
0.3	3.34	1.72	0.3	3.8	2.31	0.3	4.17	2.42
0.2	3.16	1.78	0.2	3.71	2.3	0.2	4.04	2.39
0.1	3.08	1.88	0.1	3.67	2.24	0.1	3.97	2.3

Table 5-4: Average Error for Unique Location (Far Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	3.38	1.97	0.9	4.21	2.24
0.8	3.93	1.97	0.8	4.32	2.21
0.7	3.74	1.83	0.7	4.32	2.21
0.6	3.72	1.84	0.6	4.38	2.28
0.5	3.77	1.84	0.5	4.33	2.26
0.4	3.65	1.8	0.4	4.31	2.34
0.3	3.69	1.86	0.3	4.31	2.42
0.2	3.61	1.81	0.2	4.33	2.41
0.1	4.17	1.76	0.1	4.92	2.38

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error	Test Size	Average Error	Std. Dev. Error
0.9	3.71	1.85	0.9	4.21	2.24	0.9	4.38	2.31
0.8	3.81	1.86	0.8	4.32	2.21	0.8	4.5	2.28
0.7	3.67	1.75	0.7	4.32	2.2	0.7	4.45	2.25
0.6	3.66	1.81	0.6	4.38	2.28	0.6	4.57	2.33
0.5	3.68	1.79	0.5	4.33	2.26	0.5	4.53	2.33
0.4	3.62	1.78	0.4	4.31	2.34	0.4	4.54	2.4
0.3	3.64	1.83	0.3	4.31	2.42	0.3	4.55	2.47
0.2	3.56	1.77	0.2	4.33	2.41	0.2	4.55	2.43
0.1	4.12	1.71	0.1	4.92	2.38	0.1	5.02	2.47

To validate the mentioned hypotheses in the section 3.2, we conducted the following experiments. The first experiment is to test the performance of classification against nearest neighbor algorithms. We do this using the datasets with unique RSSI values and letting the algorithms learn from all the data, which consists of all the possible location vectors. The charts shown in figures Fig 5-1 and 5-2 show that this hypothesis is validated and we can observe that the average error for classification is much less compared to the nearest neighbor algorithms.

The second experiment is to test the performance of random forest regression against the random forest classification. We conduct this experiment using the RSSI datasets with unique locations table. In this way the dataset used for testing will not contain the location information already used in training. However if we observe the charts from figures Fig 5-3 and Fig 5-4, we can see that this experiment does not support the hypothesis two. The average error observed with the regression algorithms is much greater as compared to the classification algorithms.

Therefore we can deduce that classification algorithms are useful for predicting the locations with least error based on the results of our experiments.

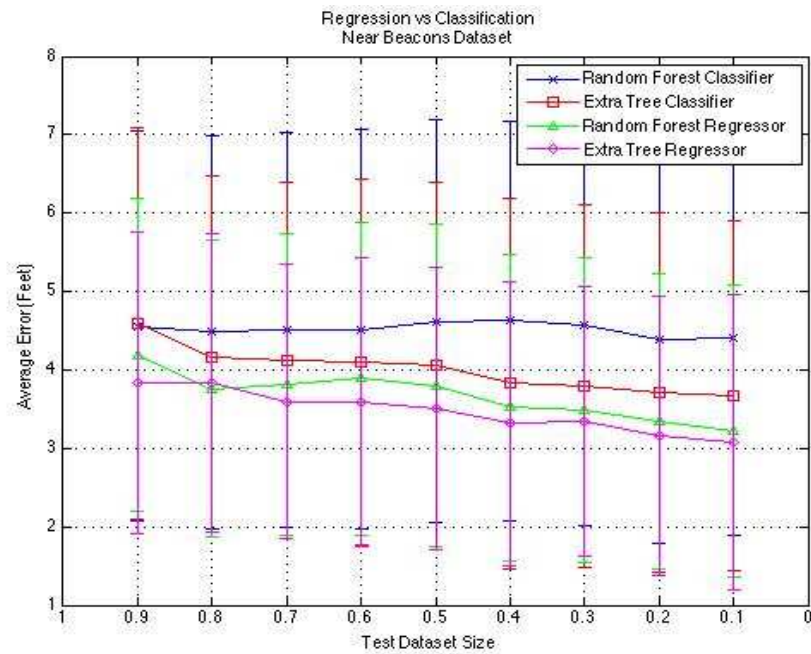


Fig. 5-3: Regression vs. Classification for near Bluetooth dataset

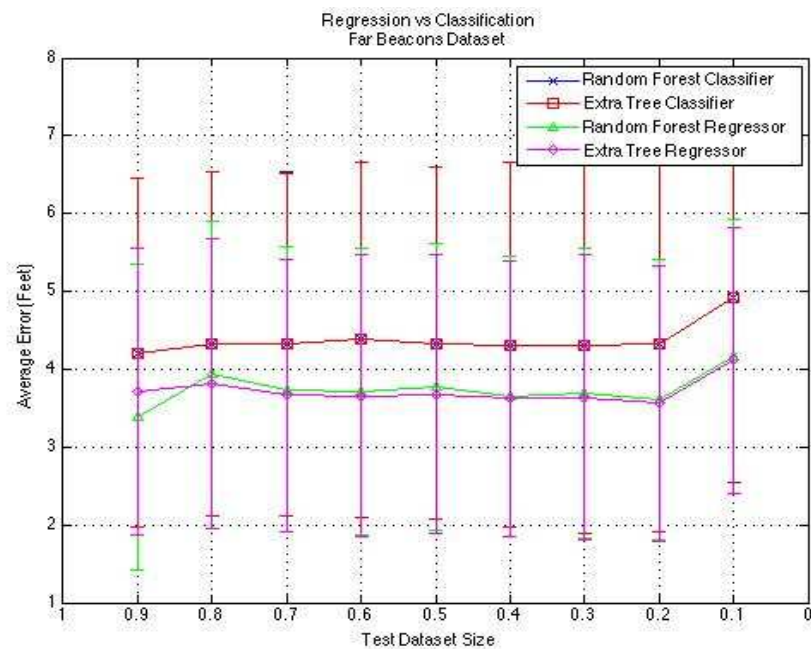


Fig. 5-4: Regression vs. Classification for far Bluetooth dataset

5.2 Cross validation and ANOVA tests

To verify the validity of the hypotheses, we conducted cross validation for our recorded data. To ensure that there is a generalized performance of the recorded data, we used k-fold with $k = 10$ for creating new partitions of the available datasets. We performed the one-way ANOVA (Analysis of Variance) test to check whether there is a comparable and significant difference between the values of the considered machine learning algorithms. Table 5-5 and Table 5-6 show the data distribution using the k-fold algorithm. The use of k-fold datasets will ensure that we are conducting the tests on independent datasets and not on overlapping samples of the collected data.

Table 5-5: Average Error for Unique RSSI Using K-fold (Near Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	1.62	1.66	0	1.46	2.33
1	1.5	1.66	1	0.99	2.2
2	1.28	1.51	2	0.95	1.91
3	1.48	1.63	3	1.13	2.22
4	1.07	1.33	4	0.69	1.76
5	1.66	1.77	5	1.36	2.52
6	1.48	1.64	6	1.04	2.18
7	1.26	1.57	7	0.95	1.99
8	1.87	1.89	8	1.24	2.39
9	1.24	1.53	9	0.85	1.97

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	1.61	1.31	0	0.97	1.95	0	1.33	2.28
1	1.51	1.39	1	0.74	1.91	1	1.16	2.31
2	1.28	1.28	2	0.66	1.65	2	0.96	2.09
3	1.55	1.39	3	0.87	1.97	3	1.31	2.4
4	1.14	1.17	4	0.52	1.54	4	0.78	1.89
5	1.69	1.58	5	1.04	2.24	5	1.5	2.64
6	1.45	1.47	6	0.84	2	6	1.28	2.41
7	1.34	1.3	7	0.66	1.73	7	0.92	2.05
8	1.77	1.57	8	0.88	2.04	8	1.45	2.56
9	1.28	1.39	9	0.73	1.87	9	1.08	2.25

Table 5-6: Average Error for Unique RSSI Using K-fold (Far Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	2.76	1.69	0	2.6	2.75
1	2.42	1.73	1	1.91	2.68
2	2.43	1.7	2	2.03	2.72
3	2.38	1.66	3	1.82	2.64
4	2.3	1.65	4	1.77	2.52
5	2.73	1.81	5	2.47	2.83
6	2.46	1.73	6	1.99	2.68
7	2.21	1.59	7	1.75	2.48
8	2.47	1.62	8	2	2.67
9	2.51	1.76	9	2.11	2.79

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	2.58	1.47	0	1.97	2.59	0	2.68	2.84
1	2.41	1.66	1	1.68	2.59	1	2.53	2.95
2	2.41	1.63	2	1.72	2.62	2	2.44	2.89
3	2.35	1.61	3	1.49	2.45	3	2.39	2.87
4	2.27	1.59	4	1.44	2.41	4	2.37	2.85
5	2.69	1.7	5	2.04	2.76	5	2.82	3.04
6	2.44	1.65	6	1.66	2.55	6	2.68	2.98
7	2.11	1.51	7	1.34	2.3	7	2.16	2.76
8	2.52	1.53	8	1.68	2.51	8	2.54	2.87
9	2.44	1.68	9	1.8	2.7	9	2.65	3.01

Table 5-7: Average Error for Unique Locations Using K-fold (Near Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	3.72	1.87	0	3.86	2.3
1	3.67	2.01	1	4.36	2.43
2	3.46	1.84	2	4.16	2.19
3	3.42	1.63	3	3.92	1.94
4	3.72	1.82	4	4.36	2.18
5	3.55	1.77	5	4.2	2.15
6	3.42	1.87	6	3.93	2.25
7	3.57	2.25	7	3.94	2.54
8	4.19	1.9	8	4.3	2.3
9	3.79	1.95	9	4.13	2.39

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	3.81	1.66	0	3.76	2.32	0	4.18	2.53
1	3.41	2.02	1	4.29	2.39	1	4.36	2.39
2	3.22	1.76	2	4.18	2.22	2	4.31	2.23
3	3.19	1.51	3	4.03	1.9	3	4.13	2.03
4	3.38	1.52	4	4.48	2.23	4	4.45	2.31
5	3.32	1.7	5	4.17	2.16	5	4.24	2.2
6	3.23	1.77	6	3.99	2.31	6	4.12	2.35
7	3.23	2.14	7	3.73	2.54	7	3.99	2.54
8	4.14	1.79	8	4.34	2.34	8	4.79	2.54
9	3.36	1.78	9	4.12	2.45	9	4.45	2.47

Table 5-8: Average Error for Unique Locations Using K-Fold (Far Beacons Dataset)

Random Forest Regressor			Random Forest Classifier		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	4.68	4.68	0	4.67	2.74
1	3.79	1.89	1	4.74	2.39
2	3.52	1.5	2	4.56	1.98
3	2.9	1.35	3	2.9	1.36
4	3.07	1.46	4	4.5	1.81
5	3.53	1.5	5	4.65	1.99
6	3.53	1.79	6	4.49	2.3
7	3.82	1.81	7	4.37	2.36
8	4.88	1.95	8	5.1	2.76
9	3.92	1.83	9	4.14	2.48

Extra Tree Regressor			Extra Tree Classifier			Nearest Neighbor		
Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error	Fold	Average Error	Std. Dev. Error
0	4.68	1.85	0	4.69	2.78	0	5.23	2.86
1	3.64	1.82	1	4.73	2.36	1	4.84	2.39
2	3.47	1.46	2	4.56	2.01	2	4.64	2.08
3	4.48	1.7	3	2.82	1.32	3	4.47	1.74
4	3	1.4	4	4.52	1.78	4	4.53	1.84
5	3.5	1.44	5	4.68	1.97	5	4.76	2.03
6	3.39	1.73	6	4.51	2.31	6	4.58	2.33
7	3.79	1.79	7	4.42	2.37	7	4.72	2.58
8	4.86	1.86	8	5.08	2.78	8	5.31	2.91
9	3.85	1.79	9	4.06	2.48	9	4.59	2.65

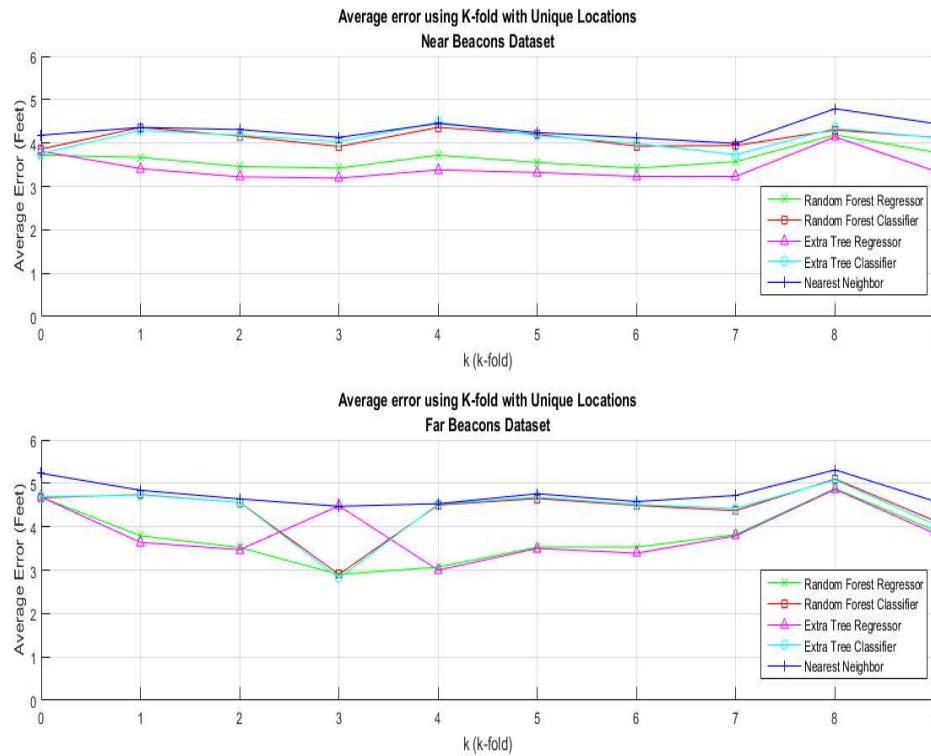


Fig. 5-5: Average error using k-fold with Unique Locations for Near and Far datasets

In this figure, Fig 5-5, we have plotted the average error computed from the sub datasets obtained after performing the k-fold splits. We have $k = 10$ folds for the. We can observe that all the algorithms are performing better as compared to the nearest neighbor algorithms, even when we generalize the partitioning data with the k-fold with non-overlapping train and test dataset pairs. The above graphs are representing the k-fold split data with unique locations using the near and beacons datasets.

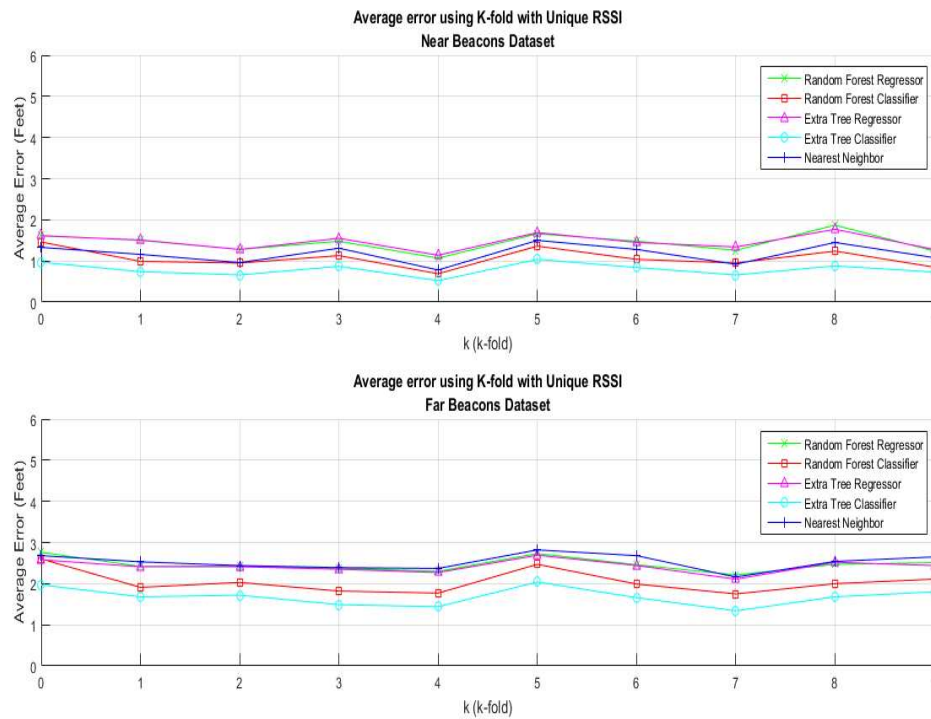


Fig. 5-6: Average error using k-fold with Unique RSSI for Near and Far datasets

In this figure, Fig 5-6, we have plotted the average error computed from the sub datasets obtained after performing the k-fold splits. We have $k=10$ folds for the. The above graphs are representing the k-fold split data with unique RSSI using the near and beacons datasets We can observe that for the far beacons the error with nearest neighbor algorithms is still more than that of the classifiers. In the graph below we can see that the classifier algorithms have outperformed the nearest neighbor in the average error.

The following tables have the results from One-Way ANOVA tests for testing if our hypothesis has indeed worked or not or they can be disproved. The tests were conducted considering all the five machine-learning algorithms.

One-way ANOVA tested with post-hoc Tukey HSD (Honest Significant Difference) Calculator. Tukey HSD uses Tukey-Kramer formula when treatments (sample groups) have unequal observations [28].

The treatments or sample groups considered in the following ANOVA tests are the average errors from the machine learning algorithms obtained after the k-fold splits. The A,B,C,D and E are as follows:

A: Random Forest Regressor

B: Random Forest classifier

C: Extra Tree Regressor

D: Extra Tree Classifier

E: Nearest Neighbor

In this test, the one-way ANOVA is performed initially and if it shows a statistical significance, the pairwise post-hoc Tukey HSD and several other tests are performed on the given samples. The results of these tests are shown in detail in the result tables followed by this description.

5.2.1 One-way ANOVA with post hoc for Unique Locations (near beacons datasets).

Table 5-9: Input data with k=5 independent treatments

Treatment →	A	B	C	D	E
Input Data →	3.72	3.86	3.81	3.76	4.18
	3.67	4.36	3.41	4.29	4.36
	3.46	4.16	3.22	4.18	4.31
	3.42	3.92	3.19	4.03	4.13
	3.72	4.36	3.38	4.48	4.45
	3.55	4.2	3.32	4.17	4.24
	3.42	3.93	3.23	3.99	4.12
	3.57	3.94	3.23	3.73	3.99
	4.19	4.3	4.14	4.34	4.79
	3.79	4.13	3.36	4.12	4.45

Table 5-10: Descriptive statistics of treatments

Treatment →	A	B	C	D	E	Pooled Total
observations N	10	10	10	10	10	50
sum $\sum X_i$	36.5100	41.1600	34.2900	41.0900	43.0200	196.0700
mean \bar{X}	3.6510	4.1160	3.4290	4.1090	4.3020	3.9214
sum of squares $\sum x_i^2$	133.7777	169.7462	118.4305	169.3573	185.5362	776.8479
sample variance S^2	0.0533	0.0368	0.0945	0.0576	0.0516	0.1628
sample std. dev. S	0.2309	0.1920	0.3073	0.2400	0.2271	0.4035
std. dev. of mean $SE_{\bar{X}}$	0.0730	0.0607	0.0972	0.0759	0.0718	0.0571

Table 5-11 One -way ANOVA of your k=5 independent treatments

Source	Sum of squares SS	Degrees of freedom v	mean square MS	F statistic	p-value
treatment	5.3349	4	1.3337	22.6991	2.5859e10
error	2.6441	45	0.0588		
total	7.979	49			

Table 5-12: Tukey HSD results

treatments pair	Tukey HSD Q statistic	Tukey HSD p-value	Tukey HSD inference
A vs B	6.0663	0.0010053	** p<0.01
A vs C	2.8962	0.2606080	insignificant
A vs D	5.9750	0.0010436	** p<0.01
A vs E	8.4928	0.0010053	** p<0.01
B vs C	8.9624	0.0010053	** p<0.01
B vs D	0.0913	0.8999947	insignificant
B vs E	2.4265	0.4365456	insignificant
C vs D	8.8711	0.0010053	** p<0.01
C vs E	11.3890	0.0010053	** p<0.01
D vs E	2.5178	0.3987612	insignificant

5.2.2 One-way ANOVA with post hoc for Unique Locations (far beacons dataset)

Table 5-13: Input data with k=5 independent treatments

Treatment →	A	B	C	D	E
Input Data →	4.68	4.67	4.68	4.69	5.23
	3.79	4.74	3.64	4.73	4.84
	3.52	4.56	3.47	4.56	4.64
	2.9	2.9	4.48	2.82	4.47
	3.07	4.5	3.0	4.52	4.53
	3.53	4.65	3.5	4.68	4.76
	3.53	4.49	3.39	4.51	4.58
	3.82	4.37	3.79	4.42	4.72
	4.88	5.1	4.86	5.08	5.31
	3.92	4.14	3.85	4.06	4.59

Table 5-14: Descriptive statistics of treatments

Treatment →	A	B	C	D	E	Pooled Total
observations N	10	10	10	10	10	50
sum $\sum X_i$	37.6400	44.1200	38.6600	44.0700	47.6700	212.1600
mean \bar{X}	3.7640	4.4120	3.8660	4.4070	4.7670	4.2432
sum of squares $\sum X_i^2$	145.1868	197.7592	152.8116	197.6143	227.9865	921.3584
sample variance S^2	0.3900	0.3446	0.3724	0.3775	0.0826	0.4310
sample std. dev. S	0.6245	0.5871	0.6103	0.6144	0.2874	0.6565
std. dev. of mean $SE_{\bar{X}}$	0.1975	0.1856	0.1930	0.1943	0.0909	0.0928

Table 5-15 One -way ANOVA of your $k=5$ independent treatments

Source	Sum of squares SS	Degrees of freedom v	mean square MS	F statistic	p-value
treatment	7.0160	4	1.7540	5.5959	0.0010
error	14.1051	45	0.3134		
total	21.1211	49			

Table 5-16: Tukey HSD results

treatments pair	Tukey HSD Q statistic	Tukey HSD p-value	Tukey HSD inference
A vs B	3.6601	0.0896600	insignificant
A vs C	0.5761	0.8999947	insignificant
A vs D	3.6319	0.0937332	insignificant
A vs E	5.6653	0.0020404	** $p < 0.01$
B vs C	3.0840	0.2056444	insignificant
B vs D	0.0282	0.8999947	insignificant
B vs E	2.0052	0.6053213	insignificant
C vs D	3.0557	0.2133704	insignificant
C vs E	5.0891	0.0067636	** $p < 0.01$
D vs E	2.0334	0.5942526	insignificant

5.2.3 One-way ANOVA with post hoc for Unique RSSI (near beacons dataset)

Table 5-17: Input data with k=5 independent treatments

Treatment →	A	B	C	D	E
Input Data →	1.62	1.46	1.61	0.97	1.33
	1.5	0.99	1.51	0.74	1.16
	1.28	0.95	1.28	0.66	0.96
	1.48	1.13	1.55	0.87	1.31
	1.07	0.69	1.14	0.52	0.78
	1.66	1.36	1.69	1.04	1.5
	1.48	1.04	1.45	0.84	1.28
	1.26	0.95	1.34	0.66	0.92
	1.87	1.24	1.77	0.88	1.45
	1.24	0.85	1.28	0.73	1.08

Table 5-18: Descriptive statistics of treatments

Treatment →	A	B	C	D	E	Pooled Total
observations N	10	10	10	10	10	50
sum $\sum x_i$	14.4600	10.6600	14.6200	7.9100	11.7700	59.4200
mean \bar{x}	1.4460	1.0660	1.4620	0.7910	1.1770	1.1884
sum of squares $\sum x_i^2$	21.4162	11.8610	21.7382	6.4815	14.3643	75.8612
sample variance s^2	0.0563	0.0553	0.0404	0.0250	0.0568	0.1071
sample std. dev. S	0.2374	0.2351	0.2010	0.1580	0.2383	0.3272
std. dev. of mean $SE_{\bar{x}}$	0.0751	0.0743	0.0636	0.0500	0.0754	0.0463

Table 5-19 One -way ANOVA of your $k=5$ independent treatments

Source	Sum of squares SS	Degrees of freedom v	mean square MS	F statistic	p-value
treatment	3.1425	4	0.7856	16.8035	1.7057e-08
error	2.1039	45	0.0468		
total	5.2465	49			

Table 5-20: Tukey HSD results

treatments pair	Tukey HSD Q statistic	Tukey HSD p-value	Tukey HSD inference
A vs B	5.5574	0.0025675	** p<0.01
A vs C	0.2340	0.8999947	insignificant
A vs D	9.5792	0.0010053	** p<0.01
A vs E	3.9341	0.0576937	insignificant
B vs C	5.7914	0.0015551	** p<0.01
B vs D	4.0218	0.0497311	* p<0.05
B vs E	1.6234	0.7549717	insignificant
C vs D	9.8132	0.0010053	** p<0.01
C vs E	4.1681	0.0386578	* p<0.05
D vs E	5.6452	0.0021307	** p<0.01

5.2.4 One-way ANOVA with post hoc for Unique RSSI (far beacons dataset)

Table 5-21: Input data with k=5 independent treatments

Treatment →	A	B	C	D	E
Input Data →	2.76	2.6	2.58	1.97	2.68
	2.42	1.91	2.41	1.68	2.53
	2.43	2.03	2.41	1.72	2.44
	2.38	1.82	2.35	1.49	2.39
	2.3	1.77	2.27	1.44	2.37
	2.73	2.47	2.69	2.04	2.82
	2.46	1.99	2.44	1.66	2.68
	2.21	1.75	2.11	1.34	2.16
	2.47	2.0	2.52	1.68	2.54
	2.51	2.11	2.44	1.8	2.65

Table 5-22: Descriptive statistics of treatments

Treatment →	A	B	C	D	E	Pooled Total
observations N	10	10	10	10	10	50
sum $\sum x_i$	24.6700	20.4500	24.2200	16.8200	25.2600	111.4200
mean \bar{x}	2.4670	2.0450	2.4220	1.6820	2.5260	2.2284
sum of squares $\sum x_i^2$	61.1229	42.5499	58.8938	28.7306	64.1404	255.4376
sample variance s^2	0.0291	0.0811	0.0259	0.0488	0.0371	0.1459
sample std. dev. S	0.1706	0.2847	0.1609	0.2209	0.1925	0.3820
std. dev. of mean $SE_{\bar{x}}$	0.0540	0.0900	0.0509	0.0699	0.0609	0.0540

Table 5-23 One -way ANOVA of your $k=5$ independent treatments

Source	Sum of squares SS	Degrees of freedom v	mean square MS	F statistic	p-value
treatment	5.1517	4	1.2879	29.0126	5.9773e-12
error	1.9976	45	0.0444		
total	7.1493	49			

Table 5-24: Tukey HSD results

treatments pair	Tukey HSD Q statistic	Tukey HSD p-value	Tukey HSD inference
A vs B	6.3338	0.0010053	** p<0.01
A vs C	0.6754	0.8999947	insignificant
A vs D	11.7820	0.0010053	** p<0.01
A vs E	0.8855	0.8999947	insignificant
B vs C	5.6584	0.0020698	** p<0.01
B vs D	5.4482	0.0032301	** p<0.01
B vs E	7.2193	0.0010053	** p<0.01
C vs D	11.1066	0.0010053	** p<0.01
C vs E	1.5609	0.7794382	insignificant
D vs E	12.6675	0.0010053	** p<0.01

After the cross validation and results from ANOVA and the paired tests, we have enough evidence to conclude that the regression and classification indeed perform better than the nearest neighbor algorithms. The tables, Table 5-9, 5-13, 5-17, 5-21, have the sample data selected for the ANOVA and Tukey HSD test. The p-value corresponding to the statistic analysis of the one-way ANOVA is lower than 0.05. This suggests that there is a significant difference between several samples. Therefore a post-hoc test is required to identify which of the considered pairs are different from each other. Therefore we presented the results of the Tukey HSD test to verify which algorithms are different from one another. The tables, Table 5-11, 5-15, 5-19, 5-23, present the ANOVA analysis for five sample sets (In this case the different algorithms.) for the separate datasets. The tables, Table 5-12, 5-16, 5-20, 5-24, describe the results of the Tukey calculations for the datasets.

From Table 5-12, which has data for unique locations with near beacons dataset, we can see that the regressors have a significant difference between them and the nearest neighbors. The classification algorithms also have a better performance than the nearest neighbor algorithm. We can see that the p-value is less than 0.01. Table 5-16, has information for the dataset having unique locations for far beacons dataset. We can observe here regressors and classifiers are performing better than the nearest neighbor algorithms. Table 5-20 has the data from the unique RSSI for near beacons dataset, we can observe the difference in algorithms and we can see that the regressors and classifiers performing better than the nearest neighbor. Table 5-24, has the data from the unique RSSI values for the far beacons and shows the difference between the error values for the different algorithms. Here also we can observe that the regressors and classifier have a significantly low p-value.

Chapter 6: Conclusion

6.1 Research Conclusion

For this research we successfully made use of simple off-the-shelf equipment, the Bluetooth beacons for solving the problem of indoor localization. We were able to create useful datasets using a simple custom receiver using a Raspberry Pi and a USB Bluetooth dongle.

Based on the hypotheses, the experiment supports the hypothesis 1. The random forest classification has outperformed the nearest neighbor algorithm. Therefore the predictions with discrete values have shown great results. However the experiments for the hypothesis 2 can be improved further. The interpolations of the regression prove to be less accurate as compared to the random forest classification.

From our experiments we were able to create a system for indoor localization using just the radio received signal strengths. Although the standard deviation or error may be improved further, we were able to acquire the localization information without requiring converting the signal to distance.

After analyzing and evaluating the results based on the machine learning algorithms, we were able to accomplish localization of the test dataset up to as low as 0.84 feet using the random forest and extra tree classifier algorithms. This was possible even without requiring the knowledge of the access points.

6.2 Future Work

Having a greater training dataset, which enables the learning algorithms to do more accurate predictions, can further aid in continuing this research. We can observe this based on the tables in section four. Also increasing the density of the beacons or the access points may help in understanding the distribution of the signal strengths in the indoor location. Also this may improve the results of the predictions from machine learning algorithms.

The first area in which we can work in future is to improve the results of the regression algorithms for the Bluetooth datasets. Our intuition is that the random forest regressor needs to be made more robust to handle the highly noisy RSSi data.

The second area in which we could work on is getting this system to work with the RSSI data collected from the Wi-Fi access points. The challenge we faced while conducting those tests was to create a position coordinate system for larger areas. Another challenge we faced was how to handle missing data resulting from access points that are out of range.

References

- [1] R. Empson, “Apple Acquires Indoor GPS Startup WiFiSlam For \$20M,” *TechCrunch*. .
- [2] “Wifarer • Indoor Positioning | Indoor GPS | Technology.” [Online]. Available: <http://www.wifarer.com/technology>. [Accessed: 07-Apr-2016].
- [3] A. Awad, T. Frunzke, and F. Dressler, “Adaptive Distance Estimation and Localization in WSN using RSSI Measures,” in *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools, 2007. DSD 2007*, 2007, pp. 471–478.
- [4] Y. Wang, X. Yang, Y. Zhao, Y. Liu, and L. Cuthbert, “Bluetooth positioning using RSSI and triangulation methods,” in *2013 IEEE Consumer Communications and Networking Conference (CCNC)*, 2013, pp. 837–842.
- [5] V. Honkavirta, T. Perala, S. Ali-Loytty, and R. Piche, “A comparative survey of WLAN location fingerprinting methods,” in *6th Workshop on Positioning, Navigation and Communication, 2009. WPNC 2009*, 2009, pp. 243–251.
- [6] “Wi-Fi,” *Wikipedia, the free encyclopedia*. 03-Apr-2016.
- [7] “How does RSSI (dBm) relate to signal quality (percent)?,” *SpeedGuide*. [Online]. Available: <http://www.speedguide.net/faq/how-does-rssi-dbm-relate-to-signal-quality-percent-439>. [Accessed: 07-Apr-2016].
- [8] E.-E.-L. Lau and W.-Y. Chung, “Enhanced RSSI-Based Real-Time User Location Tracking System for Indoor and Outdoor Environments,” in *Proceedings of the 2007*

International Conference on Convergence Information Technology, Washington, DC, USA, 2007, pp. 1213–1218.

- [9] “Trilateration,” *Wikipedia, the free encyclopedia*. 01-Feb-2016.
- [10] M. Shchekotov, “Indoor Localization Method Based on Wi-Fi Trilateration Technique.”
- [11] R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, “GPS: location-tracking technology,” *Computer*, vol. 35, no. 4, pp. 92–94, Apr. 2002.
- [12] Zekavat, S. A., & Buehrer, R. M. (2012; 2012). *Handbook of position location*. Piscataway, NJ; 4: IEEE Press.
- [13] P. Bolliger, “Redpin - Adaptive, Zero-configuration Indoor Localization Through User Collaboration,” in *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, New York, NY, USA, 2008, pp. 55–60.
- [14] “The Gimbal Store,” *The Gimbal Store*. [Online]. Available: <http://store.gimbal.com/>. [Accessed: 08-Apr-2016].
- [15] “Gimbal Proximity Beacon Series 10,” *The Gimbal Store*. [Online]. Available: <http://store.gimbal.com/products/s10>. [Accessed: 08-Apr-2016].
- [16] “iBeacon for Developers - Apple Developer.” [Online]. Available: <https://developer.apple.com/ibeacon/>. [Accessed: 08-Apr-2016].
- [17] “Raspberry Pi,” *Wikipedia, the free encyclopedia*. 08-Apr-2016.
- [18] “Insignia™ - Bluetooth 4.0 USB Adapter - Black,” *Insignia Products*. [Online].

Available: <http://www.insigniaproducts.com/products/computer-speakers-accessories/NS-PCY5BMA.html>. [Accessed: 08-Apr-2016].

- [19] “Ultimate GPS Module - 66 channel w/10 Hz updates [MTK3339 chipset] ID: 790 - \$29.95 : Adafruit Industries, Unique & fun DIY electronics and kits.” [Online]. Available: <https://www.adafruit.com/products/790?gclid=Cj0KEQjwrZ24BRC098fr-OqnuMkBEiQAKQ9lgGh3r8kzo8o3MP7jzsGOti7o2B3gu48G8byxU38e-joaApf38P8HAQ>. [Accessed: 08-Apr-2016].
- [20] N. S. Kodippili and D. Dias, “Integration of fingerprinting and trilateration techniques for improved indoor localization,” in *Wireless And Optical Communications Networks (WOCN), 2010 Seventh International Conference On*, 2010, pp. 1–6.
- [21] M. Quan, E. Navarro, and B. Peuker, “Wi-Fi Localization Using RSSI Fingerprinting,” *Computer Engineering*, Jan. 2010.
- [22] M. Korona, M. Mandić, V. Pejić, and D. Silađić, “Bluetooth Indoor Positioning,” 2015.
- [23] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan, and B. Ravindran, “Accurate mobile robot localization in indoor environments using bluetooth,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 4391–4396.
- [24] M. Shchekotov, “Indoor localization methods based on Wi-Fi lateration and signal strength data collection,” in *2015 17TH Conference of Open Innovations Association (FRUCT)*, 2015, pp. 186–191.
- [25] “scikit-learn: machine learning in Python — scikit-learn 0.17.1 documentation.” [Online]. Available: <http://scikit-learn.org/stable/>. [Accessed: 09-Apr-2016].

- [26] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of Wireless Indoor Positioning Techniques and Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, Nov. 2007.
- [27] "Why Doesn't GPS Work Inside a Building?" [Online]. Available: <http://science.opposingviews.com/doesnt-gps-work-inside-building-18659.html>. [Accessed: 09-May-2016].
- [28] "ANOVA with post-hoc Tukey HSD Test Calculator with Scheffé, Bonferroni and Holm multiple comparison - input k, the number of treatments." [Online]. Available: http://statistica.moood.com/OneWay_Anova_with_TukeyHSD. [Accessed: 11-May-2016].