2019

INDOOR-WIRELESS LOCATION TECHNIQUES AND ALGORITHMS UTILIZING UHF RFID AND BLE TECHNOLOGIES

Ann M. Whitney
University of Kentucky, ann.whitney@uky.edu
Author ORCID Identifier:
https://orcid.org/0000-0001-8270-5300
Digital Object Identifier: https://doi.org/10.13023/etd.2019.274

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Ann M. Whitney, Student

Dr. Johné Parker, Major Professor

Dr. Alexandre Martin, Director of Graduate Studies

INDOOR-WIRELESS LOCATION TECHNIQUES AND ALGORITHMS UTILIZING UHF RFID AND BLE TECHNOLOGIES

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Engineering
at the University of Kentucky

By
Ann M. Whitney
Lexington, Kentucky
Director: Dr. Johné Parker, Associate Professor of Mechanical Engineering
Lexington, Kentucky
2019

https://orcid.org/0000-0001-8270-5300

ABSTRACT OF DISSERTATION

INDOOR-WIRELESS LOCATION TECHNIQUES AND ALGORITHMS UTILIZING UHF RFID AND BLE TECHNOLOGIES

The work presented herein explores the ability of Ultra High Frequency Radio Frequency (UHF RF) devices, specifically (Radio Frequency Identification) RFID passive tags and Bluetooth Low Energy (BLE) to be used as tools to locate items of interest inside a building. Localization Systems based on these technologies are commercially available, but have failed to be widely adopted due to significant drawbacks in the accuracy and reliability of state of the art systems. It is the goal of this work to address that issue by identifying and potentially improving upon localization algorithms.

The work presented here breaks the process of localization into distance estimations and trilateration algorithms to use those estimations to determine a 2D location. Distance estimations are the largest error source in trilateration. Several methods are proposed to improve speed and accuracy of measurements using additional information from frequency variations and phase angle information. Adding information from the characteristic signature of multipath signals allowed for a significant reduction in distance estimation error for both BLE and RFID which was quantified using neural network optimization techniques. The resulting error reduction algorithm was generalizable to completely new environments with very different multipath behavior and was a significant contribution of this work.

Another significant contribution of this work is the experimental comparison of trilateration algorithms, which tested new and existing methods of trilateration for accuracy in a controlled environment using the same data sets. Several new or improved methods of triangulation are presented as well as traditional methods from the literature in the analysis. The Antenna Pattern Method represents a new way of compensating for the antenna radiation pattern and its potential impact on signal strength, which is also an important contribution of this effort. The performance of each algorithm for multiple types of inputs are compared and the resulting error matrix allows a potential system designer to select the best option given the particular system constraints.

KEYWORDS: Localization, RFID, Bluetooth Low Energy, Trilateration, RSSI-Informed Phase, Antenna Pattern Localization

Ann Michelle Whitney

(Name of Student)

03/25/2019

Date

INDOOR-WIRELESS LOCATION TECHNIQUES AND ALGORITHMS UTILIZING UHF RFID AND BLE TECHNOLOGIES

By
Ann M. Whitney

Johné Parker, Ph.D.
Director of Dissertation

Alexandre Martin, Ph.D., P.E.
Director of Graduate Studies

03/25/2019
Date

DEDICATION

To my beloved husband, Jamie.

ACKNOWLEDGMENTS

The author would like to thank Lexmark International, Incorporated for the use of their semi-anechoic chambers, advanced RF equipment and technical support. The work of John Fessler and Zachary Kratzer were extremely helpful and this work would not have been possible without their technical direction, support and guidance.

The author would also like to thank Dr. Johné Parker for forging the Lexmark relationship, as well as her guidance and input throughout this process. Additionally the time and effort extended by the Doctural Advisory Committee is greatly appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1 Motivation for Enhanced Localization

Localization is the technology of combining sensors with algorithms to determine the location of objects or individuals of importance. This technology is utilized, for example, to identify where critical cargo shipments are anywhere in the world, track critical pieces of medical equipment inside hospitals and take inventories in storage areas. Information about location can be important for financial, and safety reasons to many industries from manufacturing, to health care, to transportation and logistics.

The enabling technology for localization is typically an Internet of Things (IoT), or in an industry setting, an Industrial Internet of Things (IIoT). This erupting technology is becoming wide spread due to a combination of inexpensive networked sensors, and data analytics merging together to accomplish a wide range of tasks. These systems which were once task-specific, like a Wi-Fi system, can now be harnessed to provide information from all types of sensors (e.g., temperature, motion, audio, power, and clocks) to generate new, independent information about an environment (e.g., voice recognition, security surveillance, and energy efficiency). Localization provides valuable feature to a pre-existing technology, leveraging the sensor capabilities and allowing for more complex solutions.

While the economic benefits from the localization of items and individuals both indoors and outdoors are significant, outdoor localization is mostly standardized using the global positioning system (GPS). In contrast, indoor localization has significant room for improvement and uses any number of different methods. Very few existing methods are

being rapidly adopted in large enterprise in a consistent way, due to a variety of limitations such as accuracy, cost, and time.

Both the hardware and, especially, the algorithms required for effective deployment are not developed to the level that customers expect. As such, the initial attempts by corporations to bring products to market have been met with only limited success. The combination of high set-up costs, and poor performance mean customers are hesitant to adopt the technology, even if the function is extremely desirable. The current indoor localization technology networks have generally low precision, and additionally are prone to extreme erroneous readings. The extreme readings and complex hardware deployments, as discussed in greater detail in the next chapter, are all indications of the underdevelopment of this highly valuable technology.

1.2 Problem Statement

The purpose of this research is to discover or improve upon indoor wireless localization methods to achieve a fast, accurate, environmentally robust and flexible algorithm, which can locate a varying number of devices using a varying number of readers/reader antennas, with minimal implementation cost. The fundamental research problems investigated in this research include the effects of motion, the environment, and the equipment on localization accuracy. The primary environmental effect studied in this work is multipath, which is known to have a significant impact on the accuracy of distance measurements.

1.3 Overview and Goals of the Research

For this project, the first step is to identify the best indoor location algorithm/systems currently available and to quantify the limitations of those systems. A major goal of this research is to generate a superior method to determine location, if possible, while also robustly addressing current technology issues to suggest an improved product and solution. One key focus is the reduction of both outliers/extreme results from a localization system, as well as improved general accuracy for all location estimations. Outlier locations have the potential to significantly erode system reliability, and can lead to the perception that the system as a whole is inaccurate, while reducing error in general would improve the value of the system and expand possible applications.

This research focuses on two categorizations of indoor localization: many low-to-moderate cost items, or fewer high-value items. Some examples of the first situation are prescription bottles filled at a pharmacy, small medical equipment or tools in an operating room, pieces or products on the assembly line of a manufacturing facility, or merchandise in a retail store. As discussed more in the next chapter, RFID (Radio Frequency Identification) technology is selected for this application for its smaller variable cost, with individual tags costing only a few cents. The second key application is the identification of more expensive mobile devices which might move around a hospital or industrial plant, and therefore require a larger detection range. These items might include, for instance, infusion pumps at a medical facility, fork lift trucks at a manufacturing site, or even critical individuals such as newborns in a hospital. The investigation into Bluetooth technology is intended to address this type of situation with an imbedded battery that yields a longer read range and a lower implementation cost for tracking only a few objects. The goal of this

research is to utilize the combination of RFID and Bluetooth technologies to address both cases and find commonality in fundamental algorithms that can benefit both.

The project was divided into three phases:

- Indoor Localization Background
- Research into Distance Measurement Accuracy
- Identification of the Best Dynamic Localization Solutions

In the first phase, a review of the current literature was completed and initial testing was performed to create a baseline for the performance of existing hardware systems and algorithms. During phase 2 of the research, the impact of motion and the environment on RF-based distance estimates was explored in detail, and robust methods to mitigate those impacts were proposed. In the third phase of this work, the impact of equipment on localization accuracy, and methods to incorporate the actual antenna pattern, were further explored. A detailed description of the three phases is presented in Appendix D.

1.4 Organization of Document

This document is organized as follows:

Chapter 1: Introduction,
Statement of the problem, motivation, scope and organization of the work

Chapter 2: State of the art in localization
A comparison of technologies and algorithms is presented.

Chapter 3: RF Propagation and Device Signal Strength
RSSI principles and theoretical vs. experimental performance are explored.

An overview of how these interact with Bluetooth and RFID technologies is presented.

Chapter 4: Modeling RFID Propagation

This chapter experimentally and theoretically verifies the propagation of signals for RFID devices, which has been misrepresented in the literature.

Chapter 5: RSSI-Informed Phase Method for distance calculations

Contribution 1: A new method for leveraging RSSI and Phase distance measurements to create an improved hybrid method is presented in [1].

Chapter 6: Reducing RF Distance Error by Characterizing Multipath

Contribution 2: A new method to leverage information from frequency to reduce distance estimation errors with RSSI is presented. This was first reported in [2]; it has been further expanded and presented in [3].

Chapter 7: Representing the RSSI Signature

Further explorations of the RSSI signature introduced in Chapter 6.

Chapter 8: Characterizing the Environment using RSSI Signature.

Exploration of the capacity of the RSSI signature to characterize the system environment.

Chapter 9: Methods of Trilateration

Contribution 3: Several existing methods of trilateration are improved.

Chapter 10: Antenna Pattern Method of Localization

Contribution 4: A new method using a simplified approximation of the sensitivity pattern of the antenna to improve localization accuracy is presented.

Chapter 11: Experimental Comparison of Trilateration

Contribution 5: A comparison of methods using combinations of distance and location algorithms from a single large experimental data set is presented.

Chapter 12: Error Analysis

Chapter 13: Conclusions

Summary of Contributions and Future Work

CHAPTER 2. STATE OF THE ART IN LOCALIZATION METHODS

2.1 General Location Estimation Methods

The challenges associated with locating objects or individuals arises from a variety of technical constraints. The structure and characteristics of localization algorithms impact performance attributes, flexibility, and sensitivities; thus those trade-offs were investigated to determine which technology or combination of properties was most appropriate, and what the key technical considerations are.

- Localization vs. Tracking
- Indoor vs. Outdoor
- Error Sources
- Geometrical vs. Statistical Localization

2.1.1 Localization vs. Tracking

The first consideration is the purpose of the location algorithm and understanding the distinction between localization and tracking.

- **Localization** – A localization algorithm calculates the current location of an object, independently of any of its previous locations.
- **Tracking** – Tracking algorithms use an iterative process which updates the previous location by applying any changes in measurement, in order to calculate the current location of an object.

These two processes are not mutually exclusive. Tracking must begin with an initial localization step, and for accuracy should periodically re-locate the object. Additionally, localization algorithms can be used for tracking if they can be measured or calculated rapidly enough for the specific application. Algorithm scaling with a variable number of tagged objects becomes a significant factor in selection for complex scenarios.

2.1.2 Indoor vs. Outdoor

Indoor vs. outdoor algorithms vary based upon the technology used, and environmental information available for each situation. Typically, indoor vs. outdoor localization algorithms are separated by reader range, rather than the algorithm used. Outdoor systems have the advantage of fewer barriers, which create interference for radio-frequency based signals. Outdoor systems therefore are frequently more accurate than indoor signals in a given limited-range situation. At the same time the accuracy requirements for outdoor systems are generally lower than a comparable indoor installation due to the nature of the items being located or tracked.

2.1.3 Outdoor vs. Indoor Location Methods

The following is an overview of the measurement hardware and algorithm technology, which is currently used for outdoor and indoor location applications:

- **Outdoor**
    - Technology
        - Mobile Phone
        - Global Navigation Satellite System (GNSS)
        - Internal Navigation System (INS)

        - Radar
    - Measurement
        - Time of Arrival (TOA) – Calculates separation distance using the travel time of the signal from transmission to reception.
        - Angle of Arrival (AOA) – Measures phase angle differences using an antenna array to calculate the incident angle of an incoming signal.
        - Received Signal Strength Indication (RSSI) – A measurement of the amplitude of the incoming signal and used to find separation distance.

- **Indoor**
    - Technology
        - Radio [RFID, Bluetooth, WiFi]
        - Imaging [Infrared, Visible]
        - Sound [Audible, Ultrasonic]
    - Measurement
        - Received Signal Strength Indication (RSSI)
        - Phase Angle – A measure of signal location within a sinusoidal cycle. Phase angle, along with changing frequencies can be used to calculate separation distance.

The technologies selected for an indoor or outdoor application reflect the scaled differences between the two problems. Outdoor applications use more global navigation and long-range technologies, such as satellite systems, due to the scale of the environment.

Indoor systems have the advantage of an initial understanding of the location of the sensors, general infrastructure in that environment, and environmental extremes. These ranging technologies however must be effective over ranges the length of a large room.

The measurement algorithm technologies reflect these differences as well. Time of Arrival (TOA) and Angle of Arrival (AOA) methods are extremely accurate but require long distances to generate a change in time or angle large enough to be measurable with existing equipment. Conversely the RF technologies, direct line of sight technologies or audio technologies all have limited range and issues with objects impacting their accuracy. RF technologies do have the advantage though of being easily deployed in building settings with relatively low cost for initial installation. Satellite location is generally blocked within a building, so while it is possible for global positioning to be used and linked to an internal system providing general coordinates, this technology is not likely to be applicable for the type of indoor location needed by customers.

One potential cross-over technology is the mobile phone which contains both links to cellular and GPS networks as well as internal Wi-Fi and Bluetooth networks. These devices are ubiquitous and may well form an important link to localization infrastructure of the near future. It should be noted the RF technologies investigated in this research will lend themselves well to this hybrid infrastructure as it becomes available.

2.1.4 Error Sources (Indoor and Outdoor)

The key sources of error in location algorithms are similar for both indoor and outdoor localization. These include instrument error, such as reader losses, mismatches and noise, as well as environmental errors like ambient noise and multipath. A more in-depth

investigation and discussion of error sources for this research is provided in the Error Analysis, Chapter 12.

- **Multipath** – Multipath error is one of the largest sources of error for both indoor and outdoor localization. It is caused by reflections along the path of the signal, yielding a longer total distance traveled by that signal. This additional distance is understood by the reader as the object physically being further away. Figure 2-1 illustrates how reflections in the environment can yield longer total signal path lengths.
- **Reader Error** – Reader error varies significantly depending on the type of measurement. RSSI measurements, for instance, are easier for the reader to quantify than a phase measurement. Thus, phase has more variability in any particular measurement then RSSI. Measuring time of arrival (TOA) requires extreme precision and accuracy because light travels quickly. Therefore, TOA measurements are only reliable when the transmitter and receiver are separated by a large distance (cell phone to satellite) and the receiver has incredibly accurate, precise, and expensive measurement equipment, as is contained in a satellite. Angle of arrival (AOA) is calculated using an antenna array and differences in phase angle, thus AOA data requires a relatively high level of precision and a larger, more expensive antenna. AOA measurements are typically only practical in outdoor environments.
- **Moving Device** – Locating an object takes time; therefore, depending upon the speed of the object being tracked and how long it takes to locate it, an object in motion can have significant location errors.

- **Antenna Angle** – Errors due to antenna angle (or pointing error) are applicable in both indoor and outdoor localization systems. Antenna designs produce irregular sensitivity patterns which unevenly intensify signal amplitudes. Indoor systems are more sensitive to antenna pointing errors as the amplification effect has larger variations within the smaller field of interest. Cell phone towers have a fairly isotropic antenna radiation pattern for objects on the ground, and satellites are extremely directional, but are precise in their angle towards earth. For indoor systems like RFID, Bluetooth or Wi-Fi, signal direction is frequently unknown, and due to desires for low-cost installation the design goal is to have these systems deliberately respond to signals from any angle. Thus, these antennas are often designed to be as uniform as possible, but a truly isotropic antenna does not exist.

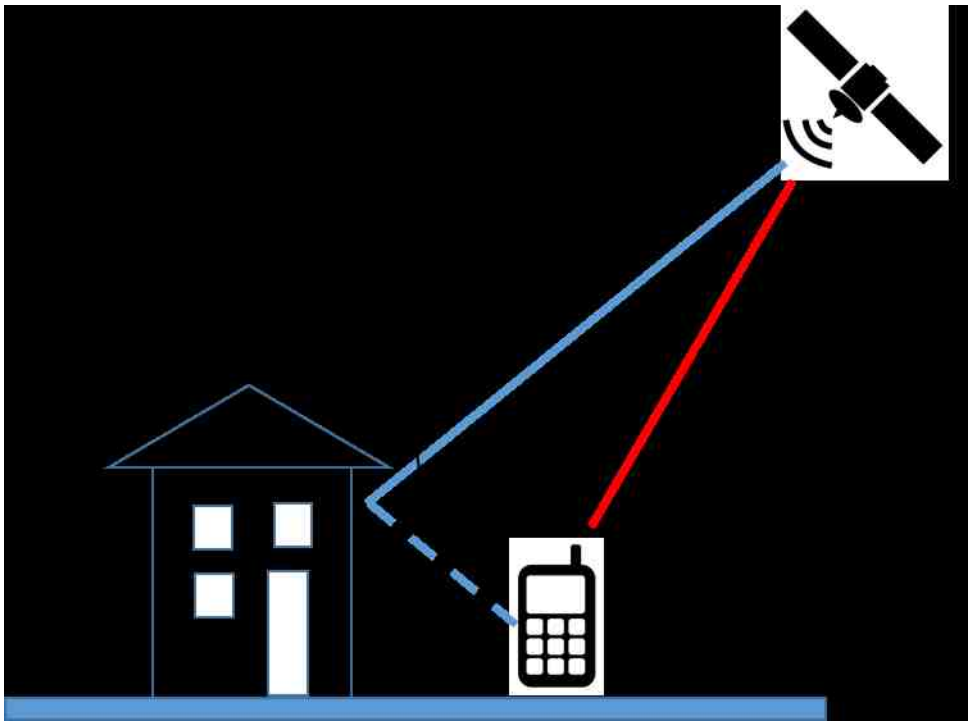


Figure 2-1: Reflections and multipath in an outdoor environment

2.1.5 Geometrical vs. Statistical Location

Another distinction for location algorithms is the employment of geometrical or statistical algorithms. Geometrical algorithms used trigonometry to calculate the location of an object, whereas statistical methods use previous or related data and probability analysis to find the likely location.

- Geometrical
    - Triangulation – Finding the location of an object using known angles.
        - Uses AOA data.
    - Trilateration – Locating an object using known distances.
        - Uses TOA, RSSI, phase, or other data which can be used to calculate distance.
- Statistical
    - Fingerprinting – Comparing incoming signal data to a map of known values, in a constant environment, to determine the likely location of the object.
        - Typically uses RSSI data.
    - Near Neighbor – Using reference devices at known locations, in comparison to the incoming signal, to determine the likely location of the object.
        - All measurement types can potentially be used for near neighbor.
    - Machine Learning– Supervised machine learning algorithms with a significant amount of training data can be used to find a likely location. The result is different from a fingerprint as it can handle more variability.

While geometrical and statistical methods are distinct in their approach to calculating location, often a combination is used to capitalize on the strengths of each. For example, geometrical methods are quite useful in finding an approximate location with minimal computation; however, as is the case with trilateration, any error in the system can mean the geometry will not converge on an actual location. Thus, geometry can be used to find an initial area, and then statistically based optimization methods can be used to narrow the area down to the most likely location.

2.2 Choosing Sensors and a Location Algorithm

2.2.1 Problem Statement

The problem statement which initiated this research was:

Problem Statement

The purpose of this research is to discover or improve upon indoor wireless localization methods to achieve a:

- Fast
- Accurate
- Environmentally robust algorithm
- Locating a varying number of devices
- Using a varying number of readers/reader antennas
- With minimal implementation cost.

The problem statement specifies indoor localization, thus outdoor technologies such as cellular networks, GNSS, INS and radar can be eliminated, along with the measurement methods TOA and AOA. Therefore, the possible measurement methods include: radio, sound, and imaging. The problem statement also dictates fast localization rather than tracking because a single quick localization algorithm can be used for both location and tracking applications. Image analysis is often computationally intensive and is highly specific to the environment in which it's trained. Audio is a slower measurement due to the differences between the speeds of sound vs. light and complex with regards to noise interference. Thus radio sensors were determined to be the technology of choice. The following graphic (Figure 2-2) shows how the different indoor localization technologies compare. Radio based technologies meet the requirements for an indoor localization system with low installation cost and a fast enough speed to potentially allow for device tracking. There are two different technologies that fall into this general category, Radio Frequency Identification (RFID) and Bluetooth technologies as shown in Figure 2-2. Both will be evaluated in this work.

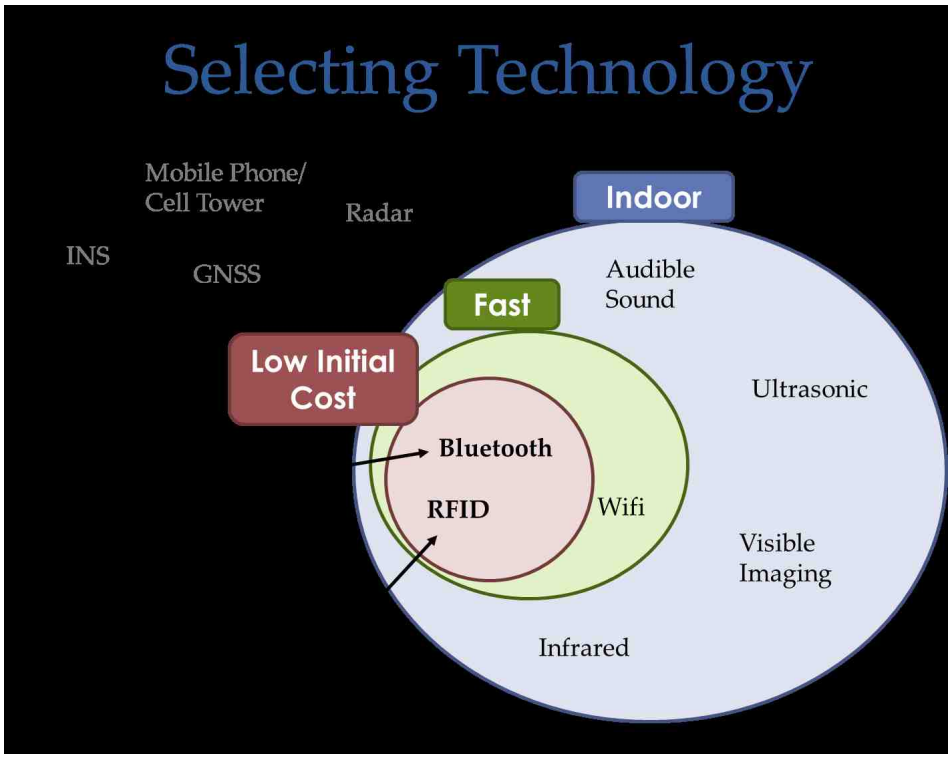


Figure 2-2: Selecting a technology for localization

2.2.2 Radio Devices

Radio sensors are applicable indoors, have the potential for high accuracy, are fast, can easily incorporate a varying number of devices, and depending on the technology have a low implementation cost. Indoor environments are complex and constantly changing, as a result RF devices often have large location error indoors. Thus, a goal of this project is minimizing of the impact of complex and variable indoor environment on location accuracy.

Between various RF technologies the dominant factor for its selection is prevalence in the market, which is linked to cost and ease of purchase. To minimize the implementation cost, while handling a varying number of devices and readers, passive ultra-high frequency (UHF) radio frequency identification (RFID) and Bluetooth Low Energy (BLE)

technologies were selected as the optimal technologies for the two different types of situations discussed in the previous chapter.

- **Passive UHF RFID** – For passive RFID, a reader is used to transmit a signal for the purpose of both supplying information and powering an RFID tag. The RFID tag receives the query, and then uses the power from the reader's signal to transmit a return signal of its own. The return transmission includes any information stored on the tag. This process is discussed in greater detail in Chapters 3 and 4.

  RFID tags were originally designed as an information-dense bar-code replacement, or non-contact/non-line-of sight identifiers. The potential of these devices quickly drove research into their use for additional applications like localization. A typical RFID tag holds 64 bytes [4] of (user available) data and recent tags can hold up to 8KB [4] of data, in comparison UPC bar codes hold 4.25 bytes [5] of data.

  Passive RFID tags have a read range of <10m [6] and commonly cost between $0.05 and $1.00 [7], [8], [9], [10] per tag. The passive (no battery needed) nature of these RFID tags reduces both the cost but also the read range. Passive RFID tags have a long lifespan and are often coated in plastic to make them environmentally robust. There is no battery that needs be replaced, so there is also no periodic upkeep. An RFID reader however, costs between $500 and $2,000 [11]. This makes RFID systems ideal for applications where a large number of inexpensive items will need to be tracked over a limited range. For instance, RFID tags are useful in retail and manufacturing environments to store information about

a multitude of different objects to which they are attached and can be used to locate those objects when needed.

- **BLE** – Unlike RFID systems, BLE devices can be used as either readers or beacons. This, in conjunction with the different networks operating (discussed in the next chapter), make BLE a versatile sensor. BLE transponders are imbedded in a wide range of technologies and utilizing the potential functionality of these other devices could significantly reduce the implementation cost of a localization network. Adoption of BLE technology into mobile devices has driven the prevalence of the devices up and the cost per device down. The price of a BLE device ranges from $10-$50, with a read range of >100m, making it an excellent and cost-effective solution for a handful of devices which need to be tracked over a greater range. They also have the ability to create connected decentralized networks which could enable location technology over a larger area than a single device can monitor.

  BLE devices achieve the higher read range because they rely upon a connected power sources such as a battery or a wall outlet connection. Battery powered BLE beacons require periodic battery charging or replacement. These characteristics make them ideal for scenarios such as tracking medical equipment in a hospital, or large pieces of industrial equipment.

2.2.3 Key Localization Algorithms

- Triangulation, shown in Figure 2-3, uses known angles from AOA to calculate the location of an object, but given the requirements of an indoor localization algorithm, this is not a viable option.

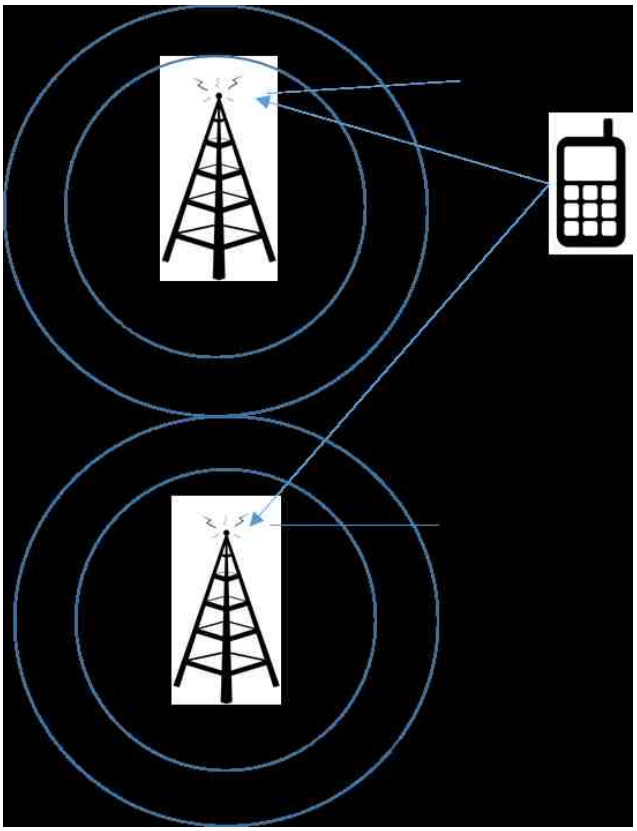


Figure 2-3: Angle of Arrival (AOA) used for triangulation

Thus, the major remaining possibilities for indoor localization algorithms using radio sensors are:

- **Trilateration** -Trilateration uses 3 or more readers at known locations, then finds the location of the object using calculated separation distances. If there is no error in the system, this can be calculated using the Law of Cosines, otherwise the geometry will not converge on a single location. If this is the case, then graphically circles can be overlaid to show the likely locations or areas. If there is some error then space in the center of the overlapping area, or some additional method of optimization is employed to find the most probable location. Figure 2-4 graphically illustrates basic idea of trilateration.

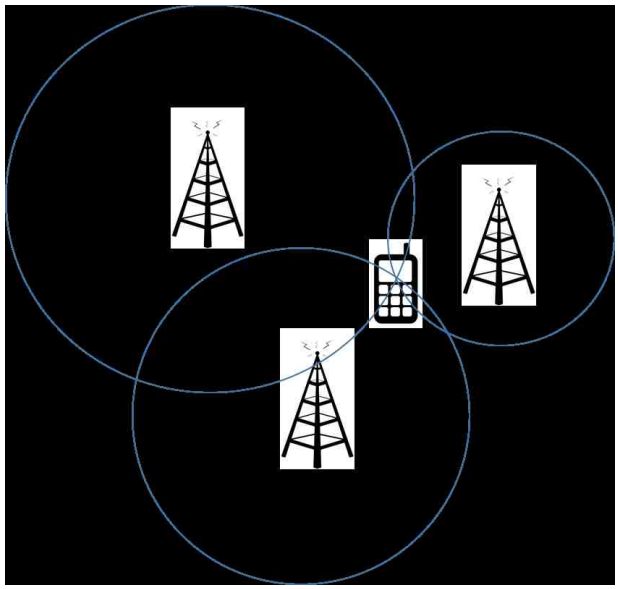


Figure 2-4: Trilateration – Using distances to find an object

- **Fingerprinting** – Fingerprinting is a method that relies on a matrix of known measurement values from a particular environment and typically uses a map of known RSSI values to determine its location. The RSSI from the desired device or beacon is compared to the map of RSSI values [12]. This narrows down the possible locations to places where the same or similar RSSI values have been recorded. Low accuracy applications can use fingerprinting with a single reader or can incorporate multiple readers along with an optimization algorithm to find the most probable location. A single reader would not be able to distinguish between two locations with the same signal strength but with a unique map of values it is still possible. Additional readers would then further narrow down the possible locations apart from errors due to changes in the environment. Fingerprinting methods sometimes suffer from extreme values when predicted location is a significant distance from the actual. This is because more than one area in the RSSI map have the same value. This issue along with environmental inconsistency has

slowed adoption in commercial applications. An example of a fingerprinting map can be seen in Figure 2-5. Experimental set-up information for this figure can be found in Appendix A, section A.4.1.

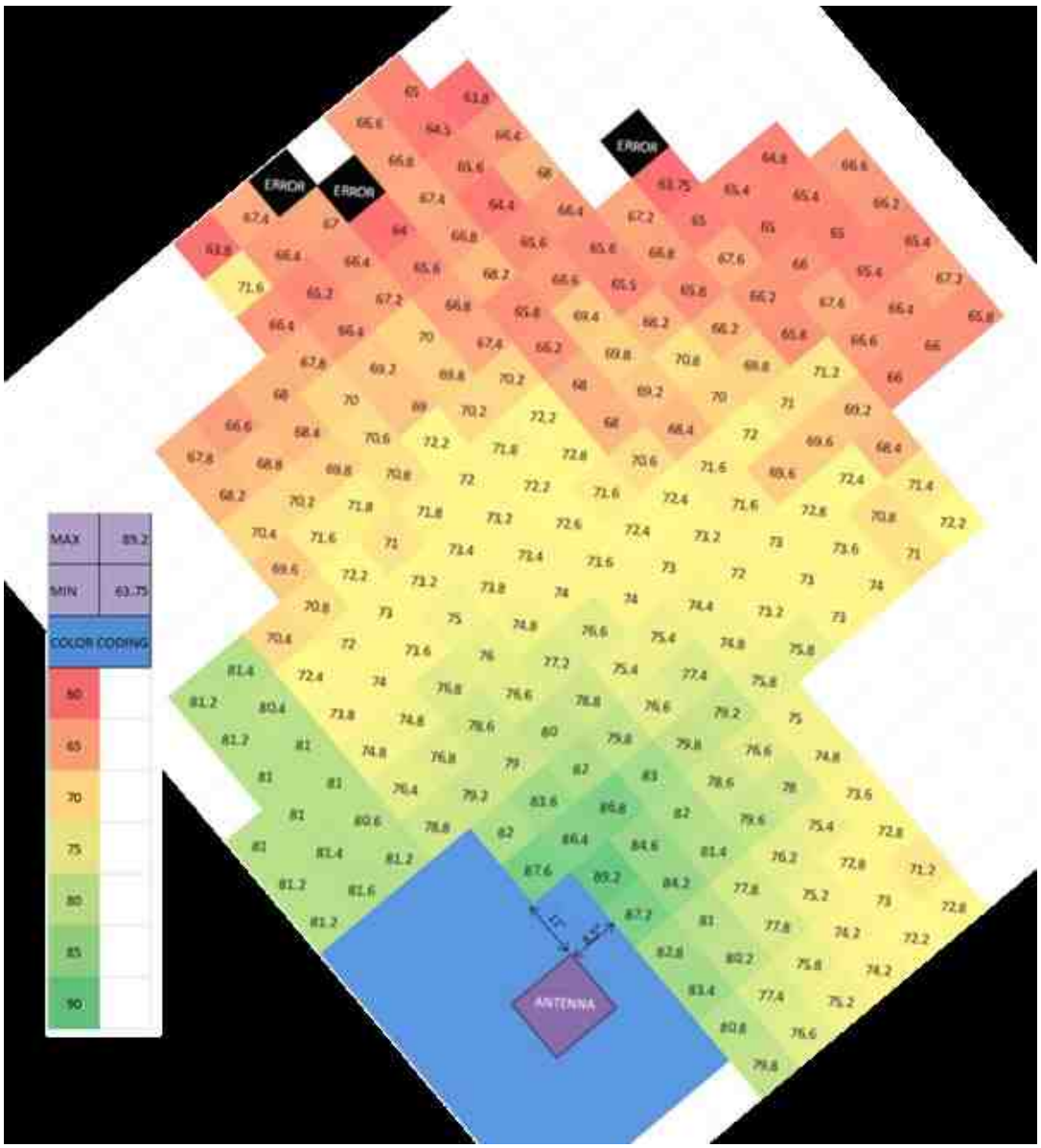


Figure 2-5: Fingerprinting map with relative RSSI values for every ft^2

The main issue with fingerprinting is the need for constantly updating the map of RSSI values. RSSI is heavily impacted by the surrounding environment. Thus, if a fingerprinting map is extremely accurate, it can be very useful for location because multipath effects are already accounted for and it acts as a unique fingerprint of the area of interest. However, even small changes in the environment can yield large changes in the fingerprinting map and result in large localization error. If a fingerprinting method is to be employed, the map must be continuously updated, which is a tedious process.

Finally, changes in antenna angle cannot be accounted for regardless of how often the fingerprint map is updated. The impact on the measured RSSI value will cause the device to be assigned or located incorrectly on the fingerprint map.

- **Near Neighbor** – Near neighbor is another comparative statistical algorithm, which can be used for either localization or tracking. It uses a large number of reference devices in the environment at known locations, then typically the RSSI of the unknown beacon is compared to the RSSI values of the reference devices to find the probable location of the beacon. Figure 2-6 illustrates the near neighbor system setup. The algorithm most commonly used is k Nearest Neighbors (k-NN), which is considered to be a simplistic pattern recognition machine learning algorithm.

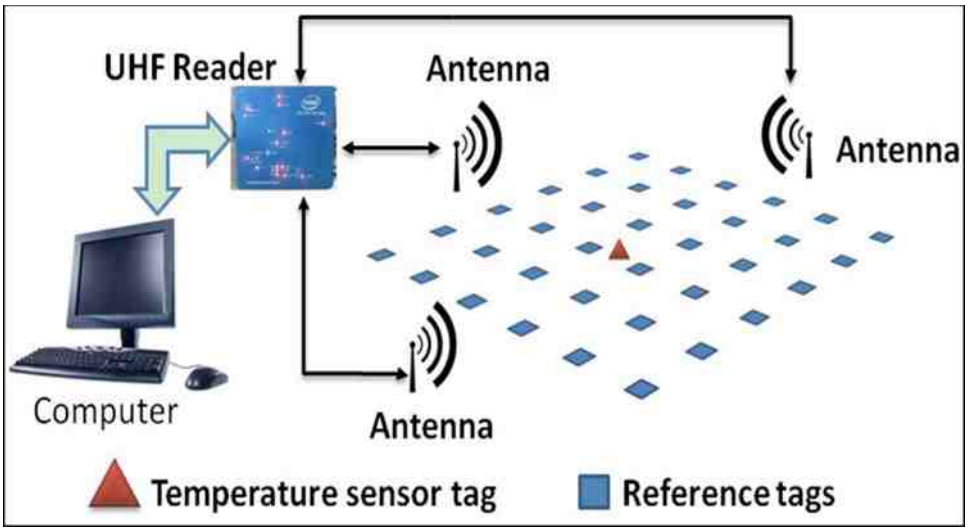


Figure 2-6: Depiction of near neighbor localization

In contrast to fingerprinting, near neighbor does not use a historical map of data for comparison, but live data from reference devices. This makes it environmentally robust when initially deployed like fingerprinting, but also continuously adapting to changes in the environment. The key drawback of near neighbor methods is the requirement for a large number of reference devices to be deployed in any environment at accurately known locations. This requires a large implementation cost both in terms of time and money.

2.3 Choosing a Localization Algorithm

In order to reduce the implementation and maintenance cost of the localization system; fingerprinting and near neighbor methods were eliminated in favor of a trilateration. While fingerprinting and near neighbor methods have been demonstrated in the literature to be more robust than trilateration, the cost and time of implementation and maintenance make them difficult to employ and maintain in real world commercial situations.

Trilateration uses very few reference devices but instead incorporates signal processing knowledge to estimate distance and location, giving it a relatively low cost of deployment and support. The plan for this research is to increase the accuracy of trilateration by finding an optimization algorithm in conjunction with geometrical trilateration to compensate for environmental impacts. Thus, the focus of this research is to find or create the optimal combination of algorithms for indoor RF trilateration.

CHAPTER 3. RF PROPAGATION AND DEVICE SIGNAL STRENGTH

3.1 Overview

The purpose of this chapter is to review the general principles of UHF (Ultra High Frequency) propagation and verify the relationship between signal strength and distance as it applies to indoor RF trilateration. The deterioration of signal amplitude as a function of distance is an underlying principle for the most common form of indoor RF distance estimation [12]. Generating distance measurements in this fashion is the first step in trilateration. Distances are generated from a satellite for devices such as a RFID reader or BLE, to a target beacon, such as an RFID tag or BLE device in an unknown location. Signal strength is often used for calculating distance and primarily impacted by dispersion as it propagates and multipath effects. For RFID systems readers/satellites are sophisticated and relatively expensive, while the tags/beacons are inexpensive and often plentiful within the specified area. Bluetooth (BLE) devices are active (contain a power source such as a battery) and can serve as either satellites or beacons. In comparison to RFID, BLE make for inexpensive satellites, but expensive beacons. This tradeoff works well for deciding what technology is appropriate for a given application.

3.2 Signal Strength vs. Distance

For both BLE and RFID, the distance between satellite and beacon is normally determined by the amplitude of the signal received by the reader, or RSSI. As the RF signal is radiated from the transmitting antenna, the electromagnetic wave density follows the inverse-square law, meaning the power density is proportional to the inverse square of the distance traveled. This phenomenon is called RF wave propagation, the most simplistic

(free-space) form of which is described by the Friis [13] equation (Equation 3-1). Figure 3-1 shows empirical data demonstrating this one-way free-space propagation, by use of a 5 meter fully-anechoic chamber. Information about the experimental set-up for this experiment can be found in Appendix A, section A.4.2. For this graph, the power at the various separation distances is given by the RSSI value as measured by the reader.

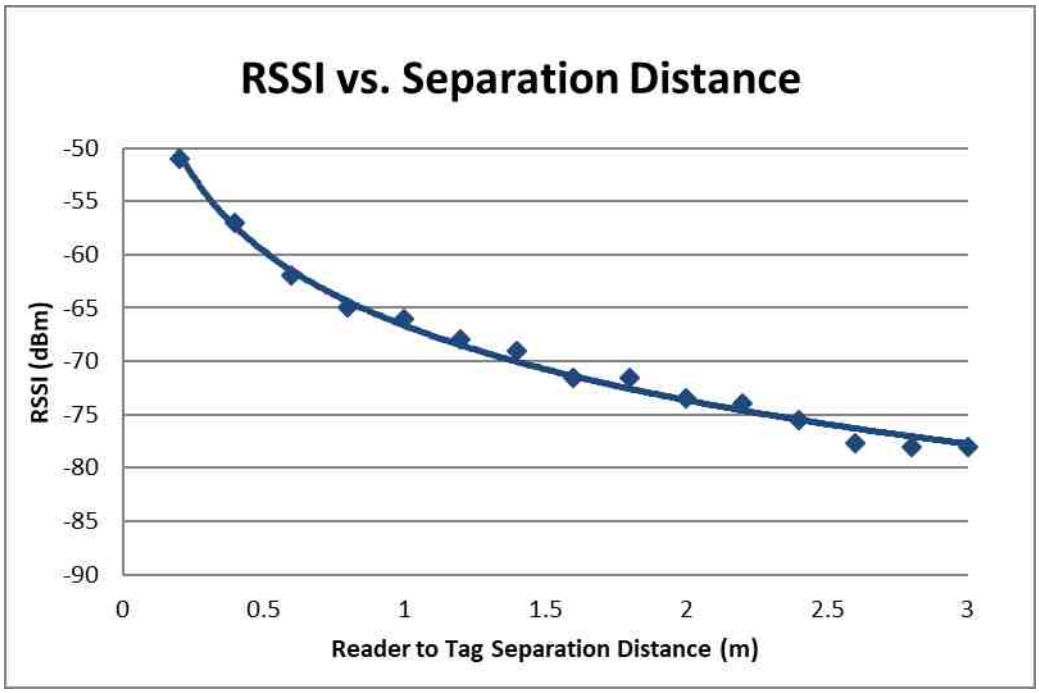


Figure 3-1: RFID data demonstrating how RSSI vs. Distance follows the inverse-square law, meaning the power density is proportional to $1/d^2$ where d is the separation distance between satellite and beacon.

Assuming an idealized free space situation, the Friis equation can be used to estimate the RSSI based on the antenna to tag distance. The Friis equation is given by Equation 3-1 [13]:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi R}\right)^2 \qquad \text{Equation 3-1}$$

Where:

P_r= Power to the receiving antenna (dBm)

P_t=Power from the transmitting antenna (dBm)

G_r=Gain of the receiving antenna (dBi)

G_t=Gain of the transmitting antenna (dBi)

λ = wavelength (m)

R = Distance between antennas (m)

It should be noted the Friis equation, in the form shown in Equation 3-1, does not take into account any number of real-world losses or disturbances, such as impedance mismatches, polarization mismatches, equipment losses or multipath. For passive RFID, some of the power transmitted by the reader is consumed by the RFID tag for its modulated response. This topic will be discussed in greater detail in the next chapter.

One assumption when using the Friis transmission formula (Equation 3-1) is the communication takes place in the far field. There are three key criteria for estimating far field (Equation 3-2) [14]:

1) $r > \frac{2D^2}{\lambda}$

2) $r > 5D$

3) $r > 1.6\lambda$ Equation 3-2

Where:

r = Separation distance (m)

D = Longest dimension of the antenna (m)

λ = wavelength (m)

For the half-wave dipole antenna, which is commonly used in this work for RFID, $D = 0.5\lambda$ and λ=0.33 m. Therefore, to be considered far field, it must satisfy 1) r > 0.165m, 2) r > 0.825m and 3) r > 0.528m for RFID. Similarly, the antenna on the end of the BLE circuit was measured to be 6.35cm long, as seen in Figure 3-2. Since the wavelength for BLE is 0.125m, this means it too is a half-wave dipole antenna. Therefore, for BLE these far-field conditions are as follows 1) r > 0.063m, 2) r > 0.313m and 3) r > 0.200 m. For this work, far-field conditions are more easily violated with RFID than with Bluetooth, as the primary constraint is r > 0.825m for RFID and r > 0.313m for Bluetooth.

Figure 3-2: BLE device used for this work, highlighting the imbedded antenna.

The majority of this work takes place clearly in the far field; however, the data collected in the near field is not excluded. While this means theoretical far field equations such as Equation 3-1 cannot be accurately assumed, the majority of this work relies on these empirical relationships anyway due to several other unknowns within the system, such as the gain of the RFID tag [4] or BLE device [15], the power consumption of the

RFID tag, or the gain of the circular polarized RFID antenna [16]. Thus, general relationships are utilized like the inverse-square rule, rather than the precise theoretical calculated values.

3.3 Multipath

Multipath is the phenomena where signals reflect off one or more surfaces before reaching the receiver. Given a measurement (M) of the reported value of signal strength from an RF beacon as seen by the satellite, the measurement will include environment multipath and the impact of instrumentation. This measurement, M, can be defined as:

$$\mathrm{M} = P_{RD} + \sum_{j=0}^{j=K} (P_{RM} \sin \psi)_j \qquad \text{Equation 3-2}$$

Where:

P_{RD} is the power received from the beacon's direct signal (in Watts),

P_{RM} is the power received from a beacon's multipath signal (in Watts),

ψ is the phase offset between the direct signal and multipath signal (radians) and

K is the number of multipath signals [3].

The summation portion of Equation 3-2 represents the constructive and destructive interference of the multipath signals, which effectively become error in the distance measurement. This discussion of multipath and the impact of frequency will be further explored in Chapter 6.

When the multiple multipath signals travel different paths, each path likely has its own unique distance. With these variations in distance, the phase angle of the signals (ψ) when they finally reach the receiver also vary. When these signals sum together at the receiver, they can either be in sync causing constructive interference and making the signal amplitude greater and the RF device appear closer, or they can be out of sync causing the signals to cancel each other out and making the RF device appear further away. The diagram in Figure 3-3 illustrates how this effect works.

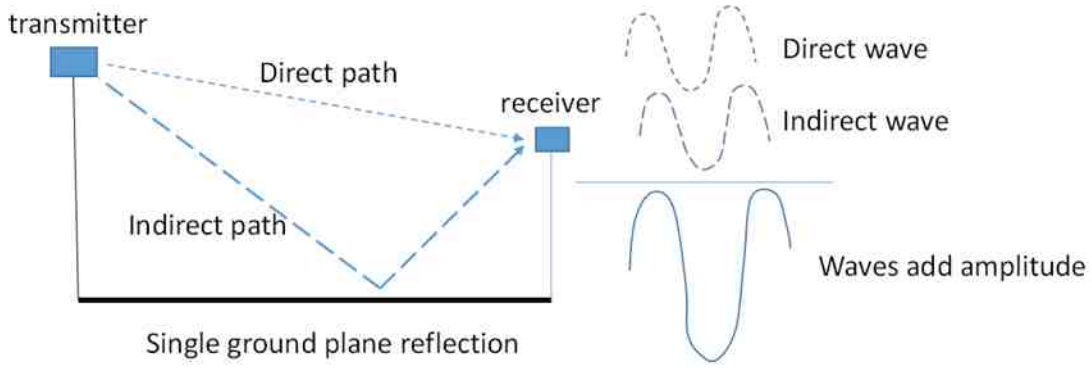


Figure 3-3: Multipath, or in this case a single ground plane refection can cause constructive interference, where the direct and indirect signals sum together yielding a greater total signal amplitude.

Multipath is often the largest component to distance error and incredibly complicated to model in real world situation [17]. This is due to the fact every RF reflective surface would need to be considered in this model, therefore the exact orientation of every object in the environment would need to be known and included. This level of detail is not practical for normal applications and even if the information was collected, the wave propagation model would be massive. Therefore, to verify a simplistic form of multipath, an experiment was performed in a semi-anechoic chamber to compare to the theoretically predicted multipath of a single plane of reflection to the empirical data. The results are

shown in Figure 3-4. The experimental set-up for this figure can be found in Appendix A, section A.4.3.

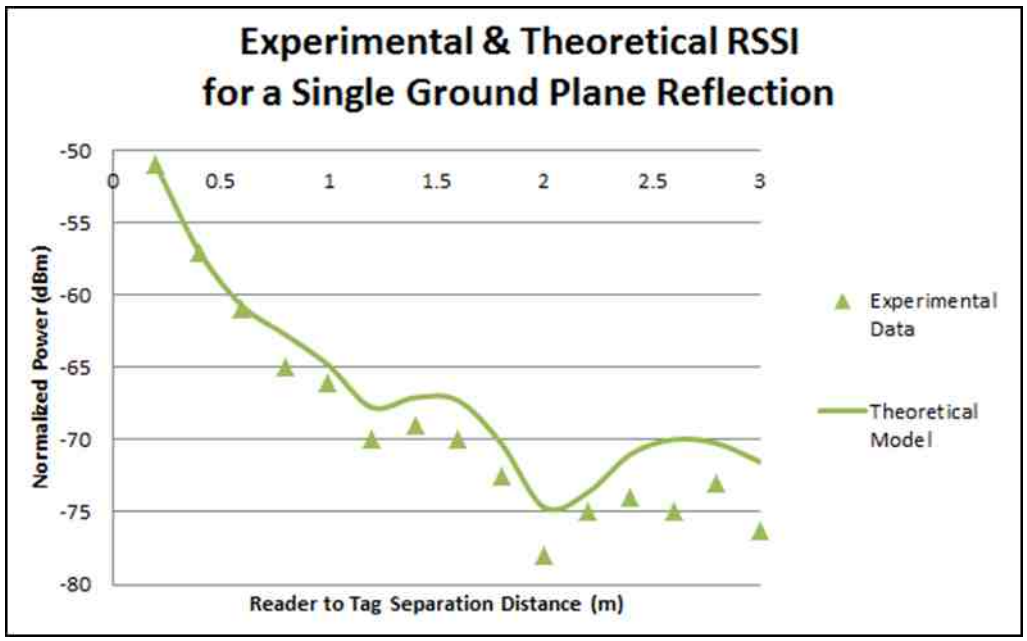


Figure 3-4: A comparison of theoretical vs. experimental multipath in the situation of a single ground plane reflection.

The theoretical ground plane reflection diagrammed in Figure 3-4 is calculated using the electric field strength equation from [18] shown below as Equation 3-3.

$$E_{DV} = \frac{\sqrt{30 P_T G} R^2 \left[{d_2}^6 + {d_1}^6 |\rho_v|^2 + 2 {d_1}^3 {d_2}^3 |\rho_v| \cos(\varphi_v - \beta[d_2 - d_1]) \right]^{1/2}}{{d_1}^3 {d_2}^3}$$

Equation 3-3

Where:

E_{DV} = Ground-wave electric field strength (μV/m)

P_T = Radiated power (pW)

G = Gain (dBi)

R = Horizontal distance between transmitter and receiver (m), seen in Figure 3-5

d_1 = Direct path length (m), seen in Figure 3-5

d_2 = Indirect path length (x +x'), seen in Figure 3-5

ρ_v = vertical reflection coefficient

φ_v = phase angle (radians)

$\beta = 2\pi/\lambda$, free-space wavenumber (phase constant)

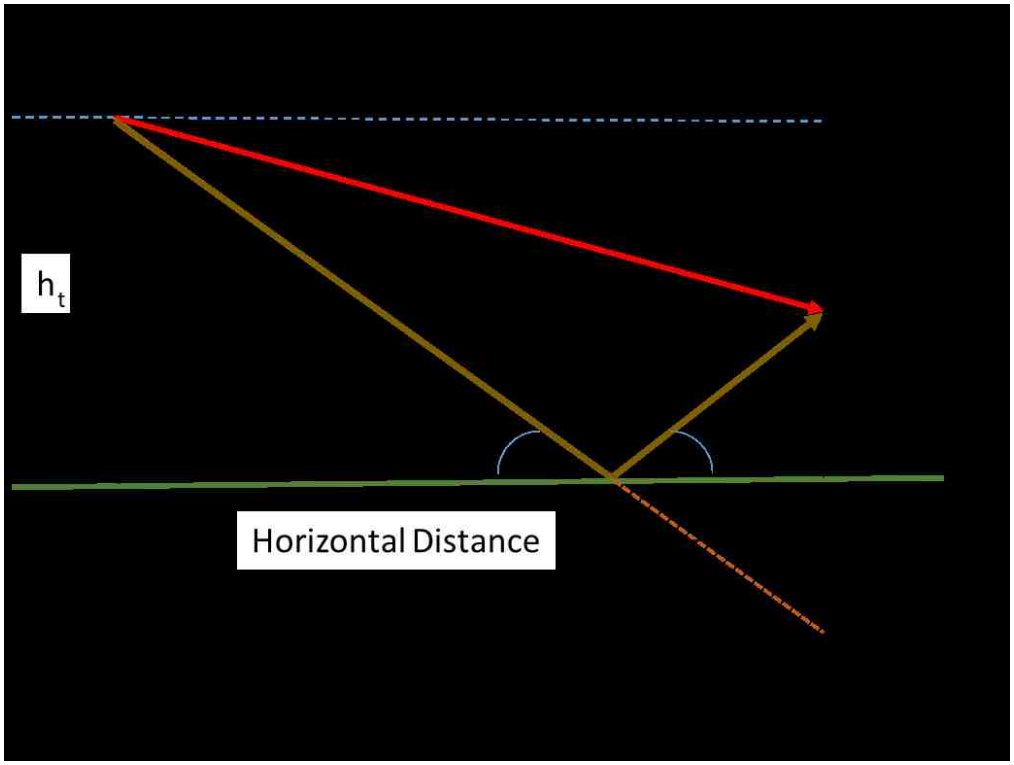


Figure 3-5: A comparison of theoretical vs. experimental multipath in the situation of a single ground plane reflection.

The shape of the curve from the theoretical calculation matches almost identically with the empirical data shown in Figure 3-4, with the exception that the experimental data deteriorates faster than the theoretical line. This would indicate there are additional real-world losses not accounted for. Chapter 12 further explains the possible sources of error and losses within the experimental setup.

3.4 RFID vs. Bluetooth Communication

To establish communication, a satellite broadcasts an encoded signal looking for a response from the beacon. The beacon is activated by that signal and replies with an encoded signal containing the requested information. In the case of a passive RFID tag, which contains no battery, the energy in the satellite's signal is scavenged and modulated to create a response. The communication can either be a point-to-point link or a point-to-multipoint. Either way, passive RFID always takes the form of a centralized network as depicted in Figure 3-6(A).

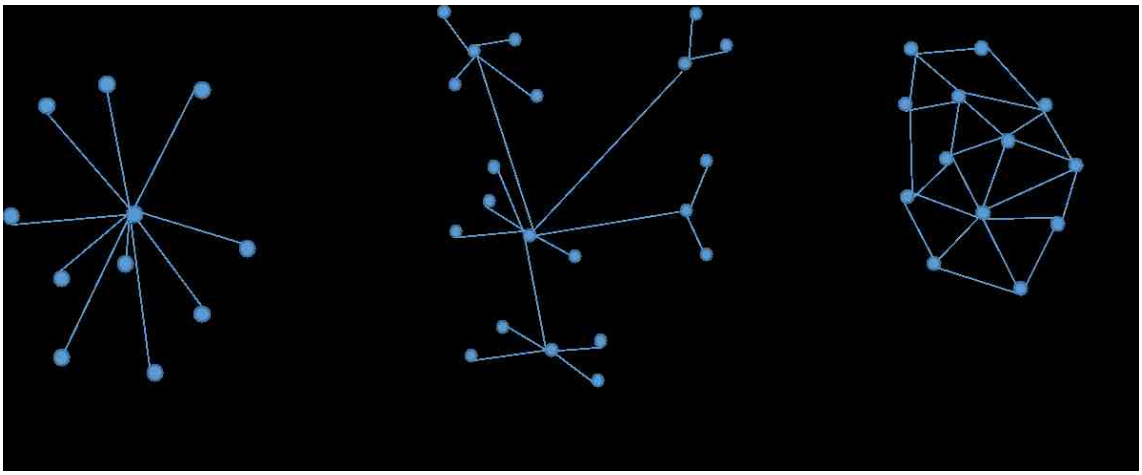


Figure 3-6: Illustration of (A) Centralized, (B) Decentralized, and (C) Distributed Mesh Networks

A BLE network is more complex because any device can serve as a satellite or beacon; or in Bluetooth specific terminology, a master or slave. This decentralized network is illustrated in Figure 3-6(B). Similar to RFID, BLE communication can either come in the form of a point-to-point connection, which for Bluetooth is called synchronous connection oriented (SCO), or point-to-multipoint called asynchronous connection less (ACL) [19]. A cluster of BLE devices is called a piconet (as shown in Figure 3-7), and a

BLE device can be a master in one piconet and at the same time a slave in another. In this way the BLE devices create a decentralized mesh network allowing information to travel great distances through the network of piconets, called a scatternet. Although BLE is often considered a mesh network, a true mesh network would be of the form illustrated in Figure 3-6(C).

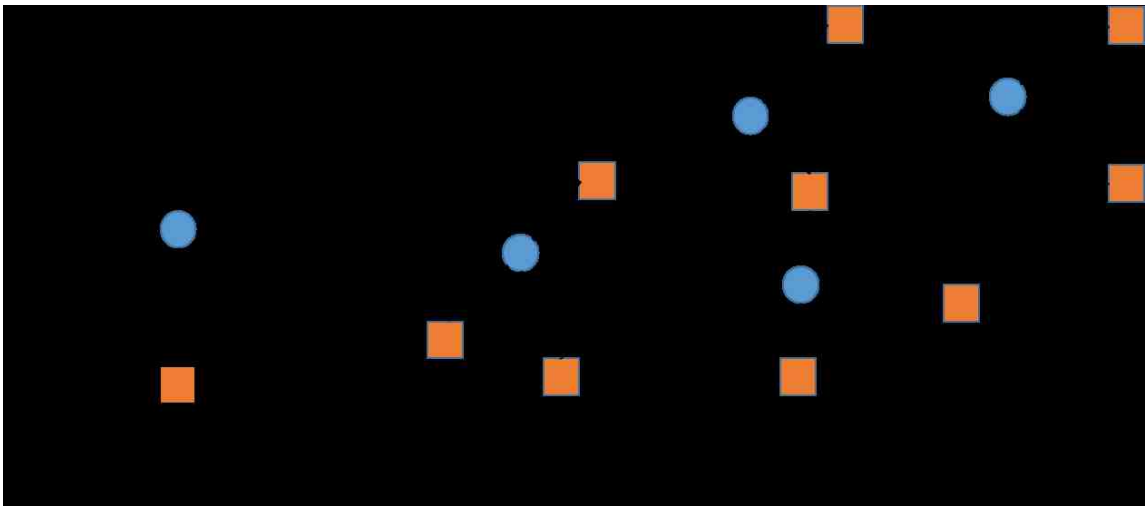


Figure 3-7: Various network formations of Bluetooth communication. (a) Point-to-Point or SCO communication, (b) Point-to-Multipoint or ACL, forming a Piconet, (c) Network of Piconets called a Scatternet

To minimize interference, RFID and Bluetooth rely on pseudo-random frequency hopping. Based on FCC regulations, RFID utilizes 50 hop frequencies within the range of 902-928 MHz while BLE hops between 40 channels ranging from 2.40-2.48 GHz, as can be seen in Figure 3-8.

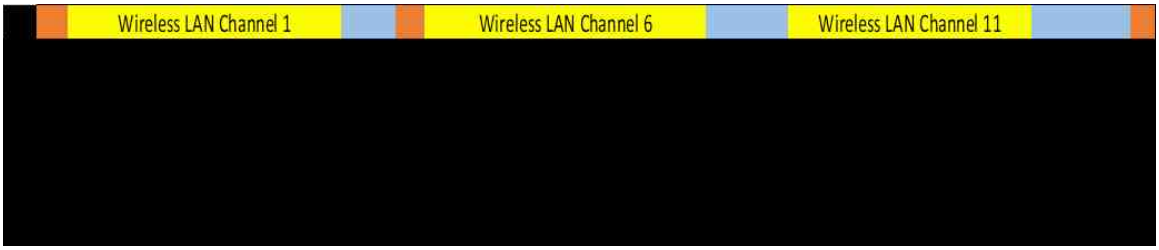


Figure 3-8: Illustration of the Bluetooth hop frequencies overlaid with the Wi-Fi channels.

Both Bluetooth and Wi-Fi operate on the same frequency band. However, these technologies can interfere with each other. This interference is investigated and is described in Chapter 12.

3.5 Conclusion

Signal strength for both Bluetooth and RFID decrease as an inverse-square function of the distance traveled and are significantly impacted by multipath. This relationship was demonstrated for both free-space and single ground plane reflection scenarios using fully anechoic and semi-anechoic chambers.

RFID and Bluetooth have unique methods of communication, with RFID solely relying on a centralized network, while Bluetooth devices can operate in a centralized or decentralized structure. These two networks structures are made possible due to the fact any Bluetooth device can operate as a mater or slave and can simultaneously serve as a master in one piconet and a slave in another. These groupings of piconets are called a scatternet and allow information to travel long distances through the network of devices.

CHAPTER 4. MODELING RFID PROPAGATION

4.1 Background

Chapter 3 explored signal strength and its dissipation as it propagates between transmitter and receiver. When the propagated signal reaches the receiving RFID tag, the tag uses power from this signal to send back a response, which in the literature is referred to as backscatter. In this chapter, the idea of RFID tag backscatter will be explored, and some of the misconceptions in the literature are clarified about how RFID tags function and how they can accurately be modeled.

4.2 RFID Backscatter

One definition of backscatter is, "the scattering of radiation or particles in a direction opposite to that of the incident radiation due to reflection from particles of the medium traversed" [20]. Essentially, backscatter is the reflection of particles in the opposite direction. One example of backscatter would be if an RF wave encountered an antenna and some of the energy was re-radiated towards the RF source. While backscatter is often thought of as a simple reflection, it can be more complex. Based on the definition of backscatter, particles must completely change direction, but as is the case with RFID tags, this redirection can be accomplished through any number of means. Since passive RFID tags have no battery and simply rely on the power of the incident wave to send a reply signal, this response is accurately referred to in the literature as backscatter [21] [22] [23]

[24] [25] [26]. True RFID backscatter is achieved by a collection of the incident wave power, and a retransmission by shifting between two states creating a modulated signal response. Unfortunately, as backscatter is often thought of as purely a reflection, there is a common misconception in the literature of oversimplifying the RFID propagation model. If RFID propagation and response is misunderstood as a basic reflection, than it will be described as a simple two way propagation model, as was incorrectly demonstrated in multiple sources [27] [28] [29] [30] [17]. There are many different forms of backscatter, and this chapter will demonstrate with theoretical circuit models as well as experimental data, why the simple reflection definition for backscatter is not applicable.

4.3 One vs. Two Way Reflection

One of the steps to improving distance estimations is to first understand the properties of the signal strength and signal propagation. The most common use of the term backscatter would suggest that the signal from the radio is reflected off of the RFID tag and back to the radio in a two-way propagation model. The first step of the signal propagating from the reader antenna to the RFID tag can be described by the Friis equation as shown in (Equation 4-1).

One-Way Propagation

$$P_{tag\ Received} = P_{reader\ Transmitted} G_{reader} G_{tag} \left(\frac{\lambda}{4\pi d}\right)^2 \quad \text{Equation 4-1}$$

Where:

$P_{tag\ Received}$ - The power received by the passive RFID tag from the radio (dBm)

$P_{reader\ Transmitted}$ - Power transmitted by the radio (dBm)

G_{reader} - Gain for the antenna of the radio which sent the transmission (dBi)

G_{tag} - Gain for the antenna inside of the tag (dBi)

λ - Wavelength of the transmitted signal (m)

d - Distance between radio and tag (m)

In a two-way propagation model, the $P_{tag\ Received}$ then becomes the starting point for the reflected energy (sometimes with an efficiency loss). Substituting this value into the starting power in (Equation 4-1) the return reflection of power for a two-way propagation is given by:

$$P_{reader\ Received} = P_{tag\ Received} G_{reader} G_{tag} \left(\frac{\lambda}{4\pi d}\right)^2 \qquad \text{Equation 4-2}$$

And therefore:

Two-Way Propagation

$$P_{reader\ Received} = P_{reader\ Transmitted} G_{reader}^2 G_{tag}^2 \left(\frac{\lambda}{4\pi d}\right)^4 \qquad \text{Equation 4-3}$$

Where $P_{reader\ Received}$ is the power received back by the reader from the RFID tag. In a system with a passive tag, energy can come only from the intercepted radio signal, thus on first glance this model appears to be a reasonable assumption of how the RFID tag performs, but that would be a mistake.

4.4 Passive RFID System

The RFID reader initiates communication by broadcasting a signal. This signal has two key components, a modulated signal with information to be decoded by the tag's chip, and a continuous wave which is used to power the tag, as shown in Figure 4-1.

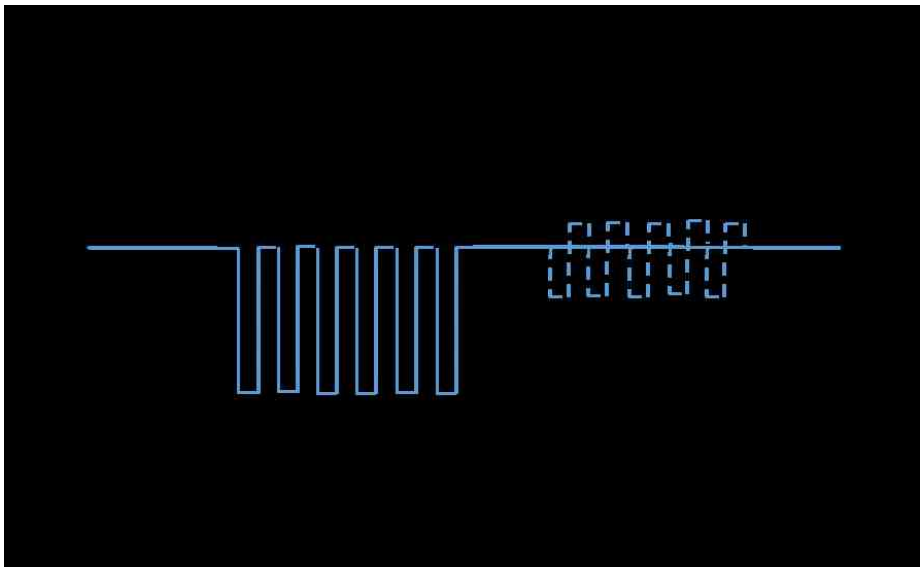


Figure 4-1: Communication link between RFID reader and tag.

When the passive RFID tag responds to the reader, it does so by modulating the continuous wave from the reader to create a signal of its own. The re-radiated signal is created based on switching between two load impedances (as can be seen in Figure 4-2), determined by the logic and information stored on the chip. "The tag sends data back by switching its input impedance between two states and thus modulating the backscattered signal. At each impedance state, the RFID tag presents a certain radar cross section (RCS). One of the impedance states is usually high and another is low to provide a significant difference in the backscattered signal." [31]

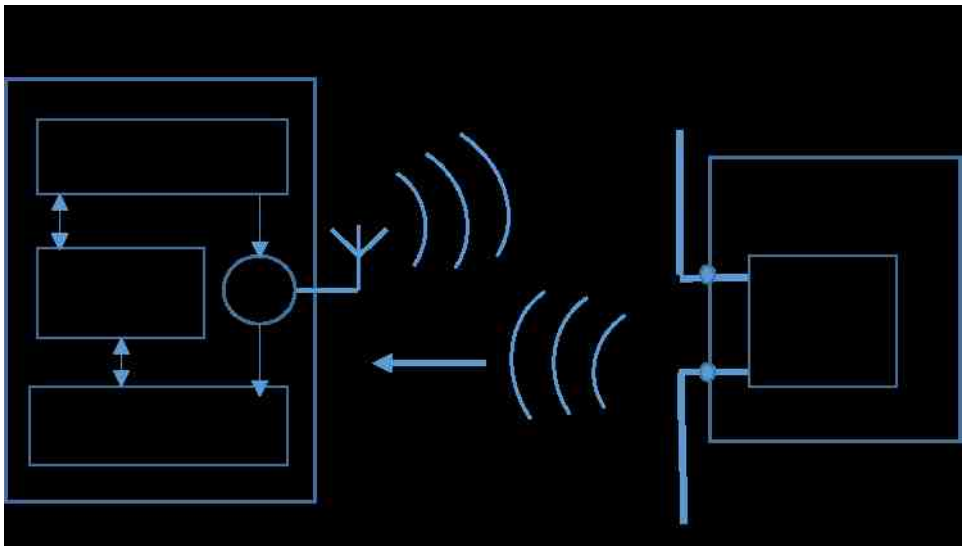


Figure 4-2: System overview of passive RFID operation.

As described in [21], an RFID tag is a transmitter and receiver but with a rectifier added to supply the DC voltage to the tag chip. The RFID chip passes through four stages of operation, from being idle waiting on a signal, to powering up when it receives a continuous RF signal, reading the address from a coded signal from the radio and finally responding if the received address matches the tag address. It is the power level of this last stage, exiting the tag, which is of interest to determining distance from tag to reader.

In order to do the complex change in signal from a received query to a response that contains a different set of data, the chip associated with the RFID tag contains several structures. Those structures are shown in the diagram in Figure 4-3. According to Yao et.al. the voltage multiplier converts a part of the incoming RF signal power to DC to supply power for all active circuits on the chip. A 250 pF blocking capacitor stores supply energy during short gaps in the received signal of up to approximately 100 μs. [22]

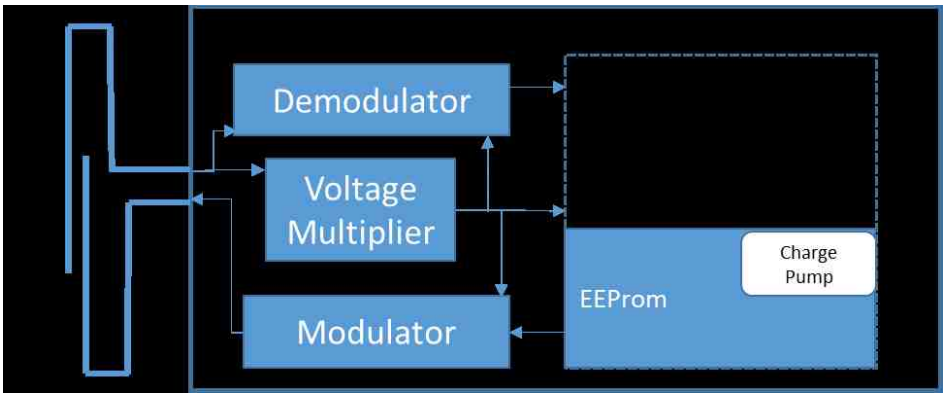


Figure 4-3: A diagram of the logic circuit for an RFID tag.

The voltage multiplier converts RF signal into DC supply voltage. It is also called a "charge pump" because it is the DC power source for the rest of the chip. The modulator circuit changes the input impedance (capacitance) which modulates the electromagnetic wave scattered back to the antenna. A simple demodulation circuit is shown in Figure 4-4. The capacitor, C1, helps to switch the voltage across the antenna between plus and minus V_{DD}.

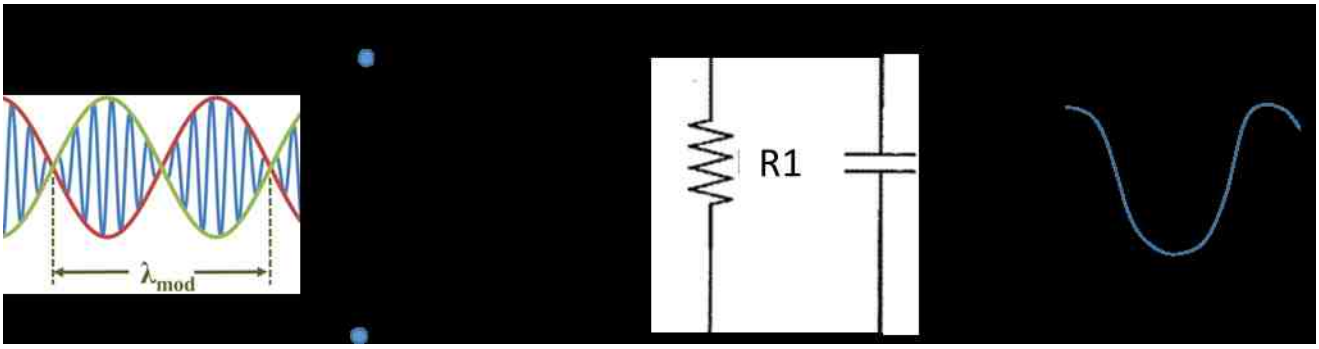


Figure 4-4: A theoretical demodulation circuit used to strip signal from a frequency modulated source. In an RFID circuit, the capacitors are the sources of energy to switch the voltage across the antenna from positive to negative and back again, creating the signal sent back to the reader.

Thus, the power going to the antenna is limited by the charge on the capacitors. These capacitors are charged from the continuous transmission wave. Therefore, there is

a limit to how much power the capacitors can store; and any additional power received will not be usable by the RFID tag for the modulated response. The tag's circuit was optimized to give the greatest read range, meaning it was designed to operate fully using the minimum supplied power. Thus, under most conditions, the response signal can be accurately received by the reader at the largest intended separation distance; but as the tag comes closer to the reader, the additional power seen by the tag is not reflected in the tag response. Instead, for the most part, the tag's response will always be broadcasted at the same power level, based upon the particular tag design.

The operation of RFID tags can be found in the literature, but often how this translates to the greater system model of RFID tag and reader can be a point of confusion. The key issue lies in the difference between simple reflective backscatter and transmission of a modulated backscattered signal.

4.5 Backscatter vs. Transmission

Every moving charge (i.e. current) emits an electromagnetic field around it. The electric field results from the voltage changes occurring in the antenna, while the magnetic field is the result of the changes of the current flow. Therefore, transmission or radiation occurs when a time varying current is passed through an antenna.

Passive UHF RFID tags operate without a battery, and simply rely on the power within the continuous wave sent by the reader. The tag responds by modulating on top of this continuous wave, essentially sending the power in the opposite direction. In the most high-level and literal sense, this is backscatter. Thus, in many instances this process of communication is accurately referred to as backscatter [21] [22] [23] [24] [25] [26].

However, in many cases this terminology has been picked up and misinterpreted as a simple reflection or modulated reflection of the incident wave. This misunderstood model is a not collect and re-transmit model but rather the tag is described as reflecting the energy by changing of the antenna's impedance to create backscatter [32]. Because there is a limit to how much energy the tag can absorb, any additional energy is not captured.

Nikitin and Rao [31], were unique in that they neither simplified to a one-way model, nor did they inaccurately depict the tag response as a simple reflection backscatter with an efficiency loss. Instead, they modified the two-way propagation equation (Equation 4-5) to include a coefficient (σ) to represent the measured loss in the system due to a calculated Thevenin equivalent circuit (Equation 4-6).

$$P_{received} = \frac{P_{ant\,Transmitted} G_{ant}^2 \lambda^2 \sigma}{(4\pi)^3 r^4} \qquad \text{Equation 4-5}$$

Where

$$\sigma = |S|^2 \frac{(4\pi)^3 r^4}{G_{ant}^2 \lambda^2} \qquad \text{Equation 4-6}$$

And:

S is the returned loss based upon the circuit design,

G_{ant} is the antenna gain

P_{ant} is the power from the antenna.

The reflection of the signal is mathematically more like a re-transmission as it starts at a power level much lower than $P_{tag\ Received}$. Thus, the most simplistic and accurate model for a passive RFID system is to depict it as a one-way, rather than a two-way propagation

model. In this way, the power level begins at a constant, and losses only occur on the way back. This is not truly the case, as power is lost in both directions, but if excess power received by the tag is never reflected, the tag always appears to radiate at the same power level.

One analogy is that of a mirror vs. a solar panel as illustrated in Figure 4-5. Backscatter is often thought of more as a light and a mirror. Light is reflected off of the mirror, perhaps changed slightly or absorbed by the imperfections in the mirror; but as a light is brought closer to the mirror (meaning more light reaches the mirror), proportionally more light is returned. In the literature when RFID tag response is referred to as backscatter, this analogous model is often incorrectly assumed. In fact, an RFID tag operates more similarly to a light and a solar panel. When light reaches the solar panel, it is collected, and can be used, in this case to power a light shining in the opposite direction. There are losses in each stage of energy conversion, but perhaps more importantly, the solar panel only can absorb so much light energy. Therefore, once it is at its maximum capacity, shining more light at the solar panel is ineffective and that energy is lost.

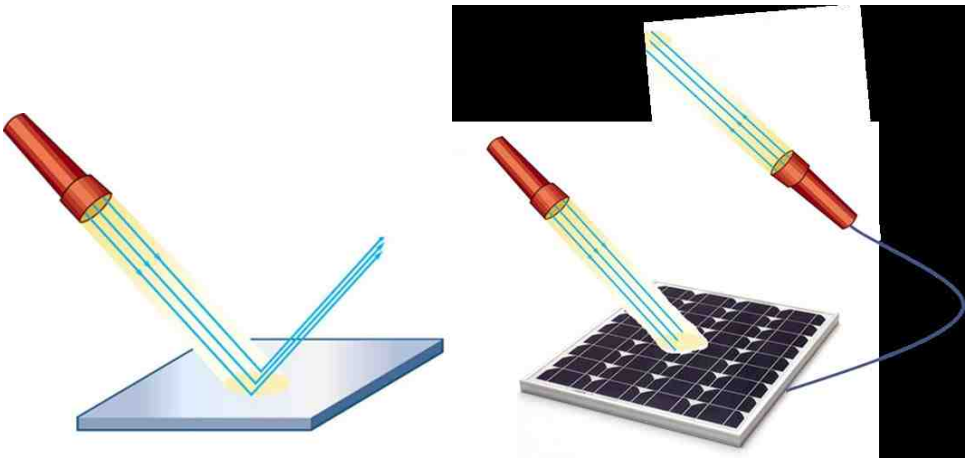


Figure 4-5: Rather than reflecting like a mirror an RFID tag works more like a solar collector powering a flashlight. The energy hitting the collector provides the power for the responding flashlight but also controls how much can be returned. Energy not captured by the collector cannot be re-transmitted back.

There are several reasons why the misconception of simple reflection may persist. It is difficult to obtain the equipment necessary to test the difference between these two propagation models as they apply to RFID tags. The manufacturers of these devices are also not clear about the design of their systems and how they minimize internal losses or make design decisions on internal impedance. Indeed, when the manufacturer was contacted and questioned about this issue the response was that the signal was based on backscatter and no further information was provided. The Friis equation, which often cited as the basis of these theoretical models, was not developed to model the sending and receiving of a given signal, but rather to look at one-way propagation alone. Researchers who begin by examining or developing tag design and structure nearly always correctly identify the mechanisms involved, but typically don't go further to provide a propagation model that can be used. Researchers who instead focus on propagation generally misuse a two-way propagation model without appropriate alteration, as was demonstrated by Nikitin and Rao [31].

4.6 Experimental Validation

The first test to verify RFID system communication was to collect the signals of both the RFID reader and the tag's response in a fully anechoic chamber using an oscilloscope. The results of this experiment can be seen in Figure 4-6. The experimental set-up for this can be found in Appendix A, section A.4.5.

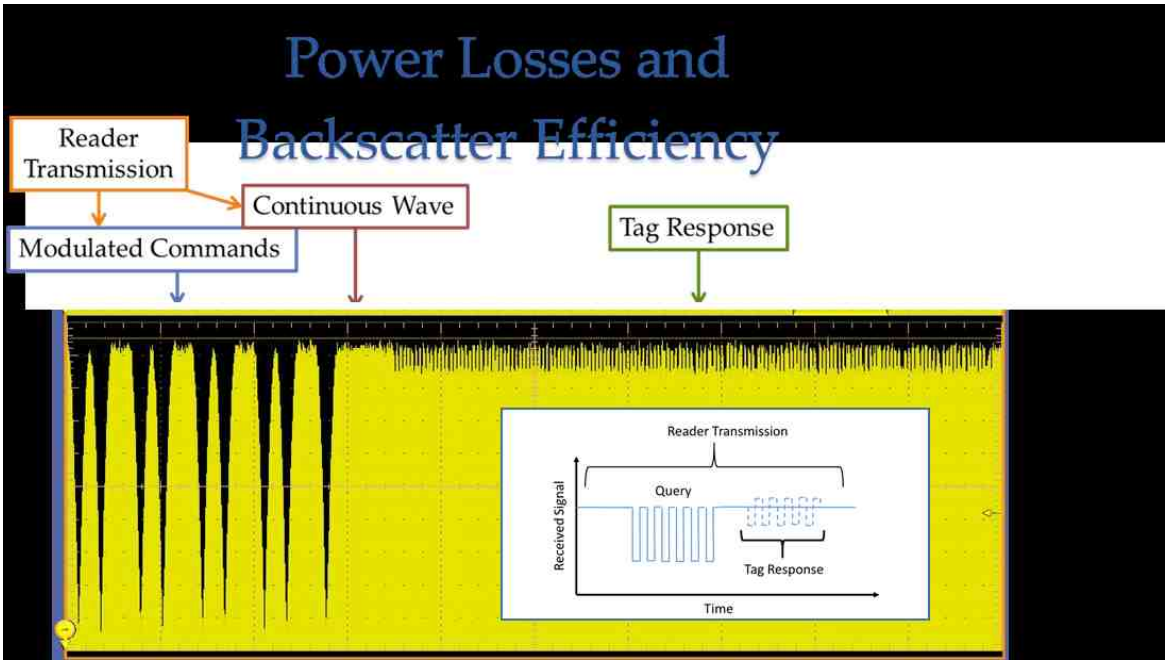


Figure 4-6: Experimentally collected signal from combine RFID reader and tag response, overlaid with diagram from Figure 4-1.

It can easily be seen the tag and reader do behave as described in the literature with a modulated command from the reader, followed by a continuous wave. Then the RFID tag responds by sending a modulated response signal on top of the continuous wave.

The next experiment was to validate the overflow theory of energy collected by an RFID tag. For this experiment, all factors were held constant for an RFID tag read, including the separation distance and carrier frequency, except for the transmit power of

the radio which was incrementally decreased. If the reflected power was a function of the power received, the return signal would decrease proportionally to the decreased reader signal. This was however not the case. The results of the experiment can be seen in Figure 4-7, and a description of the experimental set-up can be found in Appendix A, section A.4.4. The RSSI from the tag did not change proportionally to the transmit power of the reader, but instead remained constant until the power got so low that the system could not fully charge. This would suggest a response from the tag is designed to be at a set power level, making the tag response more realistically modeled as a one-way propagation model.

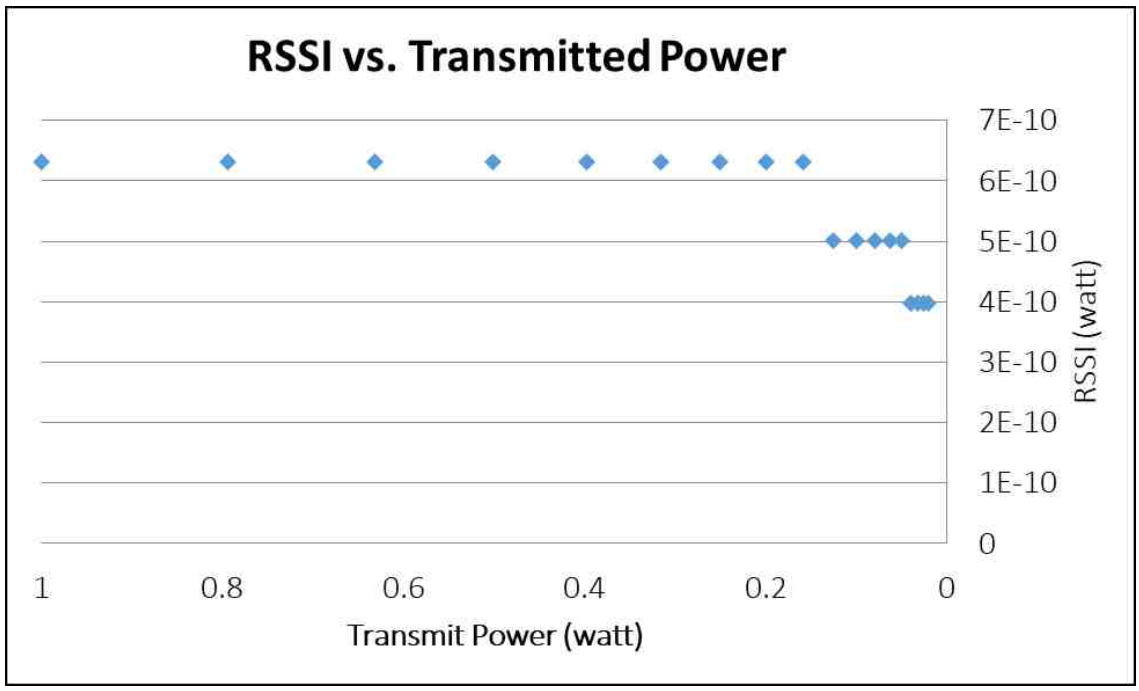


Figure 4-7: The transmit power of the reader was dropped to find what impact that would have on the RSSI response from an RFID tag. The tag maintained a constant RSSI response until the transmit power became too low to fully charge it.

The same issue was noted by Chen et al. in 2013. When the power received by the tag dropped for reasons other than distance, the tag responded with a constant power back. [33]. RSSI remained at a floor level until the transmitting power exceeded 24 dBm. Above

that level the RSSI signal had a near step-response to an RSSI value over 1000. In this paper no real explanation was provided which might explain why the RSSI levels off rather than continue proportionally to the transmitted power of the reader. However, it is clear they experienced the same effect of changing the transmit power level as noted here.

Another method was developed to verify this hypothesis that systems could be accurately modeled as one-way propagation. For this experiment, a set of passive UHF RFID tags was selected from several manufacturers. The experiment was run in a fully anechoic chamber, and individually the tags were measured at increasing separation distances from the RFID reader antenna. The results of this experiment can be seen in Figure 4-8. The experimental set-up can be found in Appendix A, section A.4.6.

Figure 4-8 The theoretical model for one-way and two-way reflection showing the expected decrease in signal strength, compared to normalized data taken for RFID tags tested in an anechoic chamber.

From the results in Figure 4-8, it is easy to see the power loss is not proportional to the distance to the 4th power (d^4) as is the case with a two-way propagation model (Equation 4-3), but instead proportional to distance squared (d^2) like a one-way model (Equation 4-1). This is important, because over the course of the literature review, the majority of literature described RFID tag response as a two-way propagation model with a simple efficiency loss in the tag, as was shown in examples in section 4-2, which is in fact incorrect.

4.7 Conclusion

RFID tags are designed to read at a maximum separation distance, and therefore consume the minimum energy. However, when supplied additional energy, the tag does not proportionally send the extra energy back in the backscattered signal. Instead, a constant power is output from the tag, therefore taking the form of a one-way propagation model. The one-way propagation model has been explained and demonstrated experimentally in this chapter as one appropriate way to represent the reader-tag communication link budget.

Some key topics covered in this chapter were:

- Backscatter is the reflection of the incident wave, and a backscatter modulator works by changing the load impedance and therefore changing the reflection coefficient of the tag.
- Passive tags do not simply reflect the reader's wave, but instead collect and re-transmit a portion of the energy, leaving excess energy unutilized. This makes the RSSI signal vs distance respond following two separate one-way

propagation models, where the starting point for the tag model is normally significantly below the amount of energy the tag was exposed to.

    - One experiment showed the RSSI from the tag was not proportional to the transmit power level of the reader, as would be expected with simple reflective backscatter or two-way propagation.
- The one-way propagation model was used to successfully predict the RSSI in the free-space/fully anechoic experiment.

It is suggested to provide clarification of the actual method of energy return from an RFID tag, a different term be used other than “backscatter”. Backscatter is often mistaken as a simple reflection and has caused significant confusion in the literature as the examples in section 4-2 illustrate.

CHAPTER 5. RSSI-INFORMED PHASE METHOD FOR RFID DISTANCE CALCULATIONS

5.1 Overview

Trilateration, which was the selected method of localization from Chapter 2, relies on distance estimation to locate beacons. Received Signal Strength Indication (RSSI) is the most common means for distance estimation for indoor RF devices [12]. Other methods use phase angle in various ways, such as the difference in phase angle between two receiving antennas, or the difference in phase angle between two different carrier frequencies by the same receiving antenna. (Note: Phase angle difference between carrier frequencies is the primary method of phase-based distance estimation in this work, as it requires the least equipment). These methods however, are more complex than using signal strength, as phase is more difficult to measure than RSSI and less straightforward to convert into a distance. Bluetooth devices do not have the ability to measure phase shift in signals to date, so the method for distance estimation described in this chapter applies only to RFID devices.

Phase based distance measurements have the advantage of being more robust against multipath, therefore often more accurate for distance estimation than signal strength measurements. In comparing the two methods, it was discovered a combination of the two could be used to leverage the benefits of each, since each method provided slightly different information. This new method of distance estimation was called RSSI-Informed Phase, and the work that lead to and verified this discovery is detailed in this chapter.

RSSI-Informed Phase became one of the benchmarks used in comparing localization algorithms at the end of this work. This distance estimation method was found to excel in real-world, complex, and changing environments, which is specifically the type of

environment of greatest interest to this research. The RSSI-Informed Phase method was presented at the IEEE/ASME Advanced Intelligent Mechatronics Conference in 2015 [1].

This chapter will start with an explanation of the current distance estimation methods using either RSSI or phase angle. Then the process of RSSI-Informed Phase distance calculations will be described. Next, several different experiments are described to test the accuracy and robustness of the RSSI-Informed Phase method. The results of the experiments will be discussed, and finally a summary and conclusion on the RSSI-Informed Phase method will be given.

5.2 State of the Art in Combining RSSI and Phase

Range estimation for passive UHF RFID tags typically relies on either RSSI or phase angle metadata [23], [34]. RSSI is known to be susceptible to the multipath of the surrounding environment [30], [35], whereas phase-based measurements are subject to cycle ambiguity [34], [36]. There have been many proposed methods to resolve the cycle ambiguity of phase-based range estimations [37], [34], [36], [38], [24], but nearly at the same time as this RSSI-Informed Phase method was presented at the IEEE/ASME Advanced Intelligent Mechatronics Conference in 2015, Martinelli et. al. [35] similarly proposed the idea of using the combination of RSSI and phase angle for distance estimation in passive UHF RFID tags. For this method, Martinelli uses RSSI for an initial measurement, with a secondary step combining RSSI and phase measurements for a more accurate distance estimation and localization of the tag. For the RSSI and phase measurements to be effectively combined, Martinelli's method requires an initial RSSI distance measurement with accuracy to within one wavelength before the phase portion of

the algorithm can be implemented. It should be noted Martinelli operated in the narrow band of 865-868 MHz, making it difficult to use the phase vs. frequency slope, and therefore a formula he used required the measured phase angle as well as an initial distance estimate to calculate the final phase-based distance estimations. This is the main distinction between the RSSI-Informed Phase method presented here, and the method proposed by Martinelli. This RSSI-Informed Phase method is a single distance calculation, rather than first calculating distance then secondarily honing the accuracy of the initial estimation.

RSSI-Informed Phase method proposed in this work, yields a robust distance estimation for passive RFID tags in the 902-928 MHz band, which is the band utilized by RFID in the United States. This method works by using an initial phase vs. frequency slope calculated from the measured RSSI, then resolves the phase cycle ambiguity by adding or subtracting increments of π to the phase points such that they align as closely as possible with the RSSI slope. This method simultaneously combines information from RSSI and phase for all measurements. In doing so this distance estimation method utilizes the robustness of phase angle measurements along with the speed of an RSSI measurement, and eliminates the need for an initialization step which had been required by Martinelli. The proposed method is faster than a two-step approach and more easily integrates into a localization algorithm.

The contributions are as follows:

1) A new method which combines RSSI and phase angle data for a more robust distance estimation that does not require an initialization stage.
2) An experimental investigation of this new method to demonstrate its robustness with respect to mobile tags, extreme antenna angles, and multipath.

5.3 Background

The RFID reader (more generally referred to as a satellite in localization) to tag (beacon) separation distance can be calculated by using either the RSSI or phase angle from the tag's return signal. RSSI and phase distance estimation methods take advantage of the change in the signal as a function of the distances between the transmitter and the receiver of the signal. This section describes how each of those methods produce a distance estimate, and how they each contain inherent issues that must be addressed by the system designer.

5.3.1 RSSI Distance Estimation

Most methods of localization begin with a distance estimation [12]. Chapter 3 describes how the strength of the received signal is a function of the distance between the reader and the receiver. RSSI is the most common way of calculating separation distance and is in many cases the best option available. Nearly RFID readers are capable of measuring RSSI (though some are not calibrated and merely report a relative value) [39] and using RSSI to predict distance is a quick calculation. The distance is calculated by knowing the propagation loss, or in other words, the relationship between RSSI and

separation distance. This relationship, which most accurately is calculated with an empirical version of the Friis Equation [13], as described in Equation 3-1 and seen in Figure 5-1. Experimental set-up for Figure 5-1 is given in Appendix A, section A.4.2. Chapter 4 discusses this relationship in greater detail. Once the best fit curve of the data is found, the equation can be used to solve for the separation distance using the RSSI value as the input.

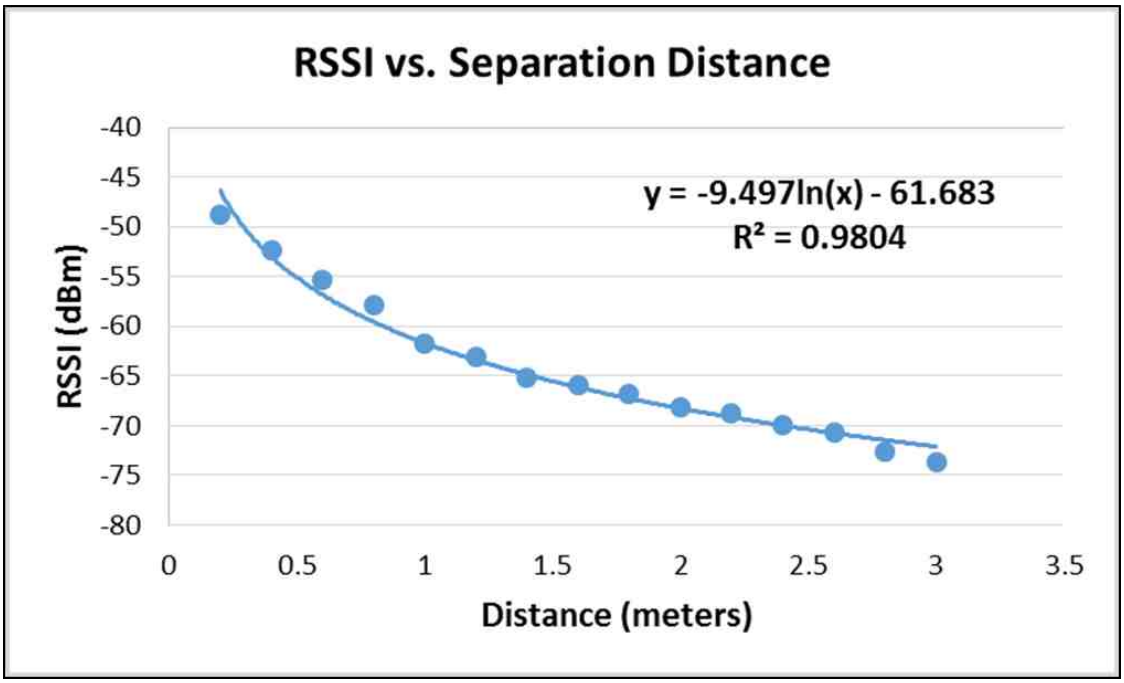


Figure 5-1: RSSI vs. Distance best fit curve

Unfortunately, RSSI is a measure of the power level of the tag's return signal and is easily distorted by environmental factors, such as absorption or multipath. Therefore, in many "real world" environments using the RSSI to predict separation distance can lead to large errors in the estimated distance.

Fingerprinting techniques, as discussed in Chapter 2, attempt to overcome this issue by mapping out the unique absorption and reflection characteristics at a matrix of locations in which a measurement might need to be made [12]. The method of fingerprinting is

therefore sensitive to changes in the physical environment with time, for example the movement of furniture or equipment. The map of the RSSI values, which constitutes the fingerprint of the environment, must therefore be continuously updated to maintain accuracy.

5.3.2 Phase-based Distance Estimation

The phase shift (φ) depends upon the round trip time of a signal, and therefore the distance between a radio and tag can be computed, but there are issues with cycle ambiguity when the phase shift is greater than 2π. For RFID systems, the tag produces a signal that is phase locked to the reader, so an appropriately equipped reader can determine the phase shift between the sent and received signals. The use of phase angle to calculate distance is significantly less common than the use of the measured RSSI value. One possible reason is fewer RFID readers report phase metadata. Another reason may be that using phase angle to calculate separation distance is more complex than RSSI.

A shift in phase angle between the emitted query signal and the returned response from the tag can be used as an indication of distance that is significantly more robust against multipath than the strength of the returned signal, due to the fact that the frequency of a wave is less impacted than an amplitude. Distance is ultimately calculated by measuring the phase shift between two or more different carrier frequencies. Note, this is a different way of calculating distance than the use of an antenna array, which would look at the phase difference between two antennas but on the same carrier frequency. The diagram in Figure 5-2 shows how two carrier frequencies will have slightly different phase angles.

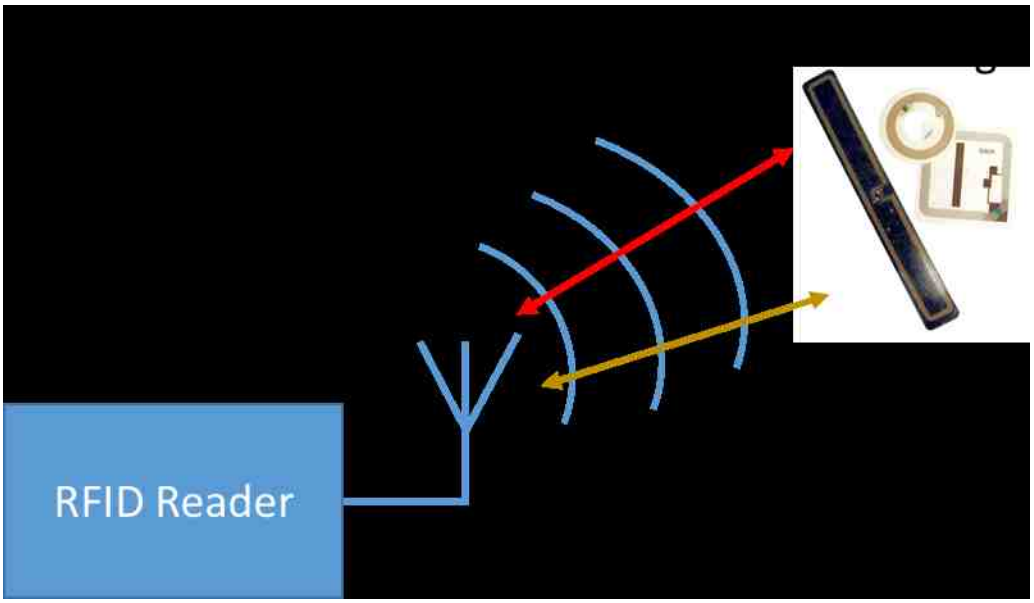


Figure 5-2: Phase-based Spatial Identification of UHF RFID Tags

The first step in using phase angle to calculate separation distance is to measure the phase angle at all or nearly all possible hop frequencies for that particular radio. Measurements at more hop frequencies reduce issues with cycle ambiguity. These hop frequencies are based upon the United States Federal Communication Commission requirements for the 902-928 MHz frequency range, where the radio must "hop" between a minimum of 25 to 50 different frequencies.

Radios capable of calculating phase angle do so by taking the inverse tangent of the quadrature over the in-phase [40], as shown in Equation 5-1. Hence the phase angle is further limited to from 0 to π radians. When the phase angle is measured and graphed with respect to its carrier frequency, the result is a saw-tooth pattern as shown in Figure 5-3.

$$\varphi = \tan^{-1}\left(\frac{Q_{nk}}{I_{nk}}\right) \qquad \text{Equation 5-1}$$

Where:

φ = phase angle

Q_{nk} = quadrature, and

I_{nk} = in-phase

To calculate distance from phase, the following equation is applied [40] [34], which utilizes the change in phase over the change in frequency. Here, β is an empirically found offset, with its values dependent on the radio setup.

$$d = -\frac{c}{4\pi}\frac{\partial\varphi}{\partial f} + \beta \qquad \text{Equation 5-2}$$

Where:

d = predicted reader to tag distance (m)

c = speed of light (m/s)

φ = phase angle (radians)

f = frequency (Hz)

β = experimental offset (m)

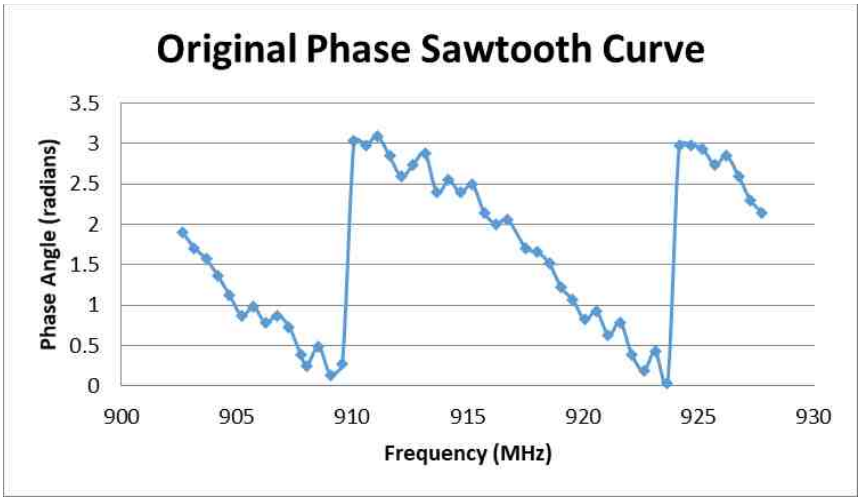


Figure 5-3: Example of a saw-tooth phase curve

In order to apply Equation 5-2, the overall shape must be found. To more easily calculate the slope, increments of π are added or subtracted at each jump in the saw-tooth pattern to create one line. It should be noted since only the slope is needed, it is not important if the data are shifted up or down through this process, as the y-intercept value of the line is not used. The experimental set-up for Figures 5-3 and 5-4 can be found in Appendix A, section A.4.7. The program used to automate the linearization of the data is found in Appendix C.

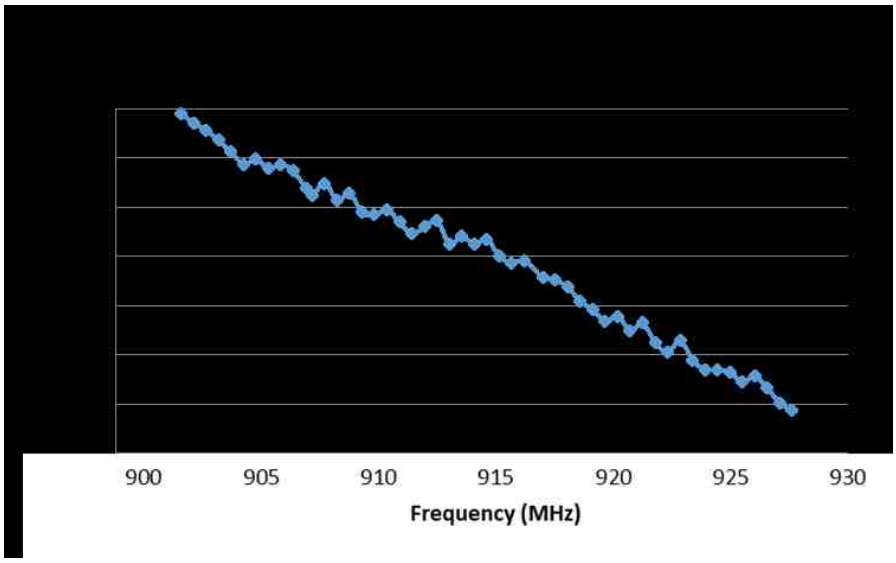


Figure 5-4: An example of a linearized phase graph

The next step is to use linear regression, an example of a linearized phase graph is shown in Figure 5-4, to find the best fit line of the phase vs. frequency data. The slope of this best fit line can then be applied to Equation 5-1, and used to solve for separation distance, as shown in Equation 5-2. The MATLAB code to do this is located in Appendix C, section C.1.

Even though phase angle is more complicated to use than RSSI, the use of phase angle for distance calculations holds a great deal of promise. Phase angle is based upon the frequency of the signal, rather than the amplitude like RSSI, therefore it is not impacted by signal attenuation to the extreme extent RSSI is. This makes phase angle distance estimations more accurate in most "real world" environments.

However, the phase angle distance calculations are highly sensitive to small movements of the tag as they rely on multiple reads of the tag, so this method assumes the

reader and tag are both stationary. If in fact the tag is moving, then this disturbance can create large errors in the separation distance calculation.

Additionally, the time to fully capture the frequency vs. phase angle relationship to the point where the cycle ambiguity is eliminated is significant, especially since federal regulations require the reader to hop pseudo-randomly between frequencies. This means the changes in frequency cannot be intentional or systematic except in a controlled research setting. The ideal method will allow for an understanding of the $\frac{\partial \varphi}{\partial f}$ value with a minimum number of measurements.

5.4 RSSI-Informed Phase Distance Calculation

The proposed method of RSSI-Informed Phase [41], [1] combines the accuracy and environmental robustness of the phase calculations, while utilizing the speed of the RSSI, to make the distance estimation less susceptible to error caused by motion, or multipath. One of the issues with phase-based calculations is confidently knowing where the jumps in the saw-tooth curve are located to eliminate cycle ambiguity, requires a nearly complete array of the measured phase angles. If large gaps are left in the phase angle vs. frequency graph, and the number and/or location of the jumps are not clear, an inaccurate guess can change the slope of the phase vs. frequency line significantly and ultimately result in extreme errors in the distance calculations.

Collecting all the necessary phase angle data takes several seconds. This process of measuring the phase angle at every hop frequency takes even longer when the radio is frequency hopping pseudo-randomly. While the exact amount of time depends upon both

the hardware and software being used, in the following experiments it took an average of 15.3 sec to measure the phase angle at each of the 50 hop frequencies when using pseudo-random frequency hopping. Even at a slow walking speed, many meters can be covered over the course of 15 seconds. This potential change in distance over the course of the phase measurements results in extreme distance estimation error, given the necessary assumption that the tag is remaining stationary.

For the RSSI-Informed Phase method, the RSSI value is incorporated to solve both of these problems simultaneously. The first step in the RSSI-Informed Phase method is to read the RFID tag a few times (four measurements are used for this work) at different hop frequencies. Measuring at only four pseudo-random frequencies took an average of 1 second during the following experiments, which is a significant decrease in sampling time in comparison to the phase method. Additionally, this time could likely be reduced with more straight-forward programming of the radio to take less processing time. While the user is unable to choose which frequency due to the pseudo-random frequency hopping requirements, ideally these frequencies will be spaced throughout the possible frequency band. Therefore, if the points measured do not span a wide range of frequencies within the frequency band, it could be helpful to wait for and collect an additional data point. With each read the RSSI and phase angle of the tag's return signal is recorded.

The next step is to take an average of the RSSI values. The RSSI value will likely be different at the various hop frequencies even when holding all other conditions constant as described in further detail in Chapter 6 [2]. Therefore, an average of the four measured RSSI values is the most reliable. Using this average RSSI value, an initial RSSI distance estimation can be found using the empirical relationship between RSSI and separation

distance, as seen in Figure 5-1. This initial distance is then applied to Equation 5-2 and used to solve backwards for an approximate phase vs. frequency slope. One of the mid-frequency points can then be used as a fixed location with which the estimated slope passes through. This initial RSSI-based line serves as an approximation to which the other phase points can be adjusted.

Increments of π can be added or subtracted from the remaining phase points in order to align them as closely as possible with the slope from the RSSI measurement. Note because slope rather than intercept is used, which point is chosen as the fixed point is unimportant. Next a best fit slope of the altered phase data points is calculated, resulting in the phase angle slope. Finally, in order to encompass the benefits of both RSSI and phase distance estimations and give more robust results, the average of the two slopes (RSSI estimated slope, and phase angle slope) is found. This average slope is applied to Equation 5-2 to find the final RSSI-Informed Phase distance calculation. One example of this process is illustrated in Figure 5-5.

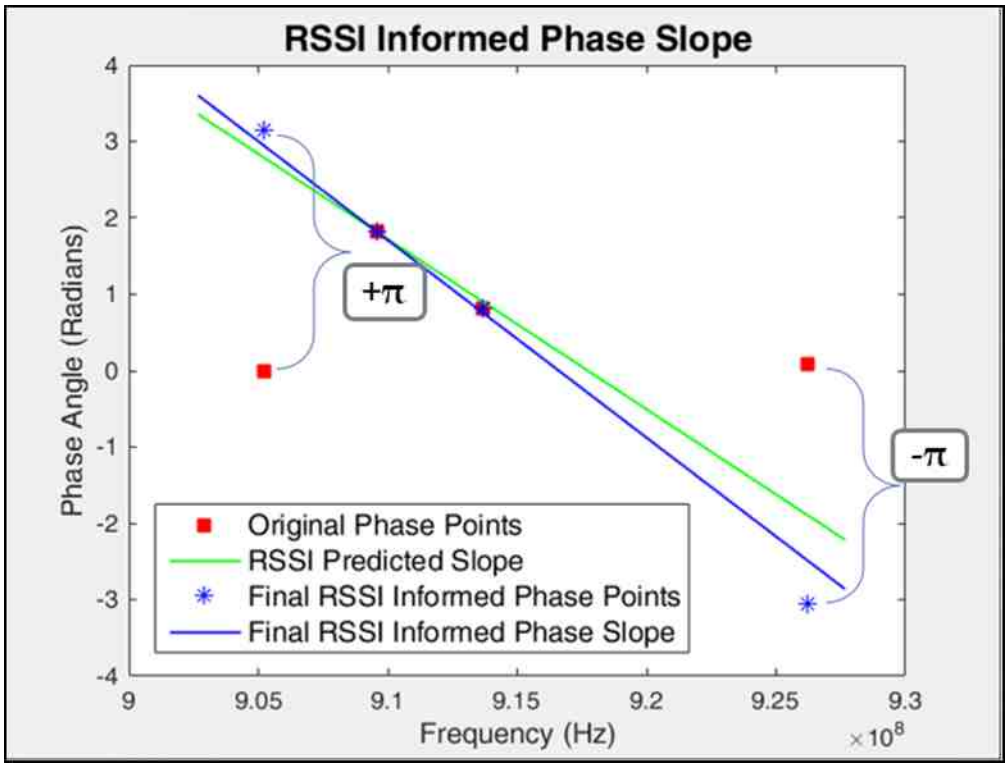


Figure 5-5: Calculation of final phase slope in RSSI-Informed Phase process

The MATLAB code for RSSI-Informed Phase can be found in Appendix C, section C.2.

5.5 Experimental Analysis

5.5.1 Experiment 1: Mobile Tag

The purpose of this initial experiment was to test the hypothesis that a moving tag would create large errors in phase estimated distance in comparison to both RSSI and RSSI-Informed Phase methods, which are believed to be more robust with respect to motion. The experiment was conducted in a 10 m semi-anechoic chamber, using a ThingMagic M6e radio, a circular polarized antenna, and a vertically polarized Alien Squiglette passive UHF RFID tag. The experimental set-up is described in Appendix A, section A.4.8. The MATLAB code can be found in Appendix C, section C.3. Both the

RFID tag and the reader antenna were 1 m above the ground plane. Using a stepper motor and pulley system, the tag began 6 m away from the reader antenna, then was moved at a constant rate towards the reader antenna. RSSI and phase were continuously measured using pseudo-random frequency hopping, and this metadata was used to calculate separation distance by employing the three previously mentioned methods: RSSI, phase, and RSSI-Informed Phase. The results are shown below in Figure 5-6. The true separation distance was assumed to be the distance from the reader antenna to the mid-point between the RFID tag's beginning and ending positions. The experiment was run three times for each speed, and the distance error for each distance estimation method was averaged at each speed. Different speeds were tested, ranging from a control where the tag remained stationary, to 0.1 m/s toward the reader antenna.

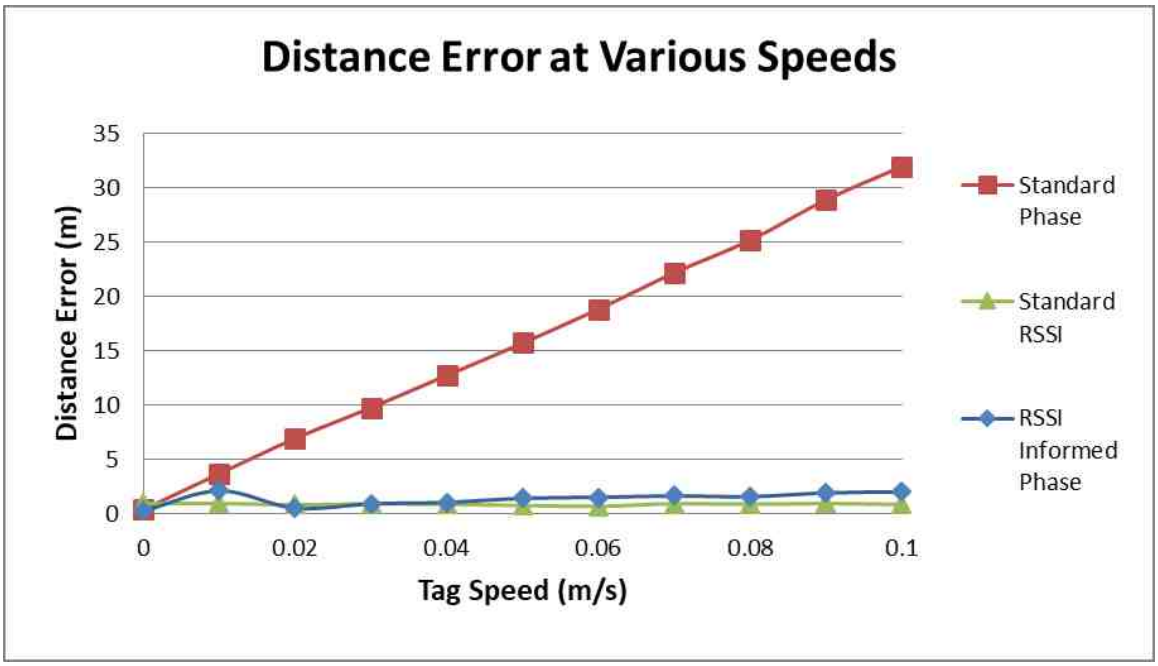


Figure 5-6: Distance Error of the Three Methods of Distance Estimation, Average Moving Error: Phase = 16 m, RSSI = 0.87 m, RSSI-Informed Phase = 1.3 m

After analyzing the results, it was found that when completely stationary, the phase distance estimation method had the least error, at 0.47 m error. However, even when the tag barely begins to move at a speed of 0.03m/s, the phase distance estimation method already has over 10.38 m of error. This is a significant amount of error at such a slow speed. To put it in context the average adult walks at a speed of 1.4m/s [42], which is nearly 50 times faster than this initial tag speed of 0.03m/s. This would indicate a distance estimation method which purely relies on the collection of phase angle and variation in hop frequency is likely to incur large amounts of error with even small amounts of motion. The error from phase distance calculations for mobile tags was found to be statistically significantly larger than both the RSSI and RSSI-Informed Phase methods. When in motion the RSSI-Informed Phase method had statistically significantly less error than even the RSSI method (with a p-value of 0.0021 and $\alpha = 0.05$), making it the best performing method for this mobile experiment.

5.5.2 Experiment 2: Orientation and Polarization Mismatch

One of the leading contributors to error in RFID distance estimation is extreme tag orientation resulting in a polarization mismatch between the reader antenna and the tag [43], [24]. While there is currently no easy solution to this problem, each distance estimation algorithm will behave distinctively. Therefore, to test the impact of polarization mismatch between RFID tag and reader antenna on the same three distance estimation methods, the following experiment was devised.

For this experiment, which took place in a 5 meter fully anechoic chamber, the same ThingMagic M6e RFID radio was used. However, in order to create extreme polarization mismatches a vertically (linearly) polarized half-wave dipole was used as the

reader antenna rather than a circularly polarized antenna as used in the previous experiment. The RFID tag used was the same Alien Squiglette tag which is also linearly polarized. Both the reader antenna and the Alien Squiglette RFID tag were placed 1 m above the ground plane and at a 1.5 m separation distance. This makes the tag relatively close, but still in the far field. The tag was then attached to a rod which was pivoted in 10 degree increments by a stepper motor. In Figure 5-7, 0° and 180° are the angles of complete polarization match, where the RFID tag and half wave dipole antenna are both vertically polarized. In contrast, 90° and 270° are the angles of cross-polarization where the half wave dipole reader antenna is vertically polarized, and the tag is horizontally polarized. At each increment of the stepper motor (turning the tag by 10°), RSSI and phase angle metadata were recorded at 50 hop frequencies. This process of rotating the tag and then measuring the RSSI and phase angle was repeated for 30 full revolutions of the tag, yielding 30 measurements at each angle. Once recorded, the data was analyzed using the RSSI, phase, and RSSI-Informed Phase methods of distance estimation. The distance estimation error from each of the three methods at various angles of polarization mismatch is displayed in Figure 5-7 and the MATLAB code used to generate the graph is given in Appendix C, section C.4.

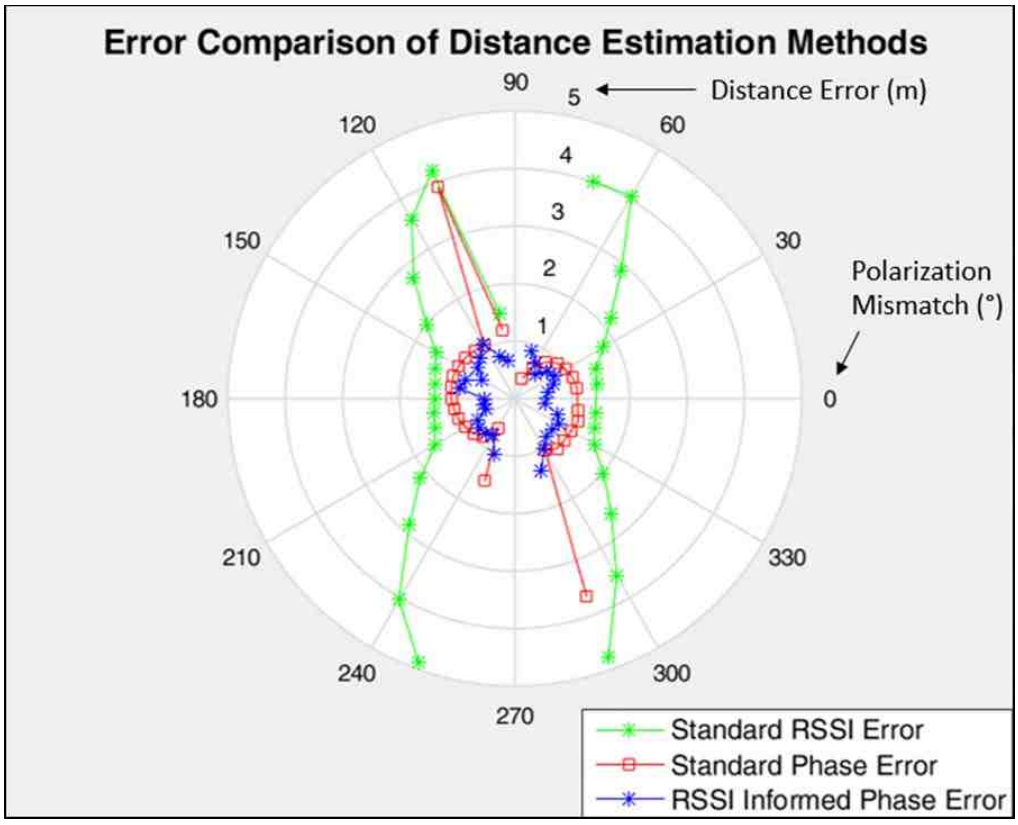


Figure 5-7: Distance error from RSSI, Phase, and RSSI-Informed Phase Methods, with various degrees of polarization mismatch. Average error: Phase = 1.1 m, RSSI = 2.2 m, RSSI-Informed Phase = 0.8 m

While the RFID reader was able to communicate with the tag at most of the polarization mismatch angles, as expected there were a few angles where the tag could not be read. These angles were: 80°, 90°, 260°, 270°, and 280°.

In this experiment it was found RSSI distances calculations resulted in the greatest error; significantly greater than both the phase (2 x) and RSSI-Informed Phase (2.75 x) methods. The phase and RSSI-Informed Phase methods perform similarly and produce fairly accurate results for a wide range of polarization mismatch angles, but RSSI-Informed Phase ultimately was the most accurate.

5.5.3 Experiment 3: Multipath

Multipath is a phenomenon where the transmitted or received signal takes more than one path, as described in greater detail in Chapter 3. By traveling different paths, the radio waves travel different distances, leading to errors in reader to tag distance estimates. Although difficult to account or compensate for, multipath is inherent to non-anechoic environments. RSSI is known to be impacted more significantly than phase angle by multipath, and therefore phase angle measurements are hypothesized be more robust in high multipath environments [34].

This last experiment for RSSI-Informed Phase compares the three methods of distance estimation: RSSI, phase, and RSSI-Informed Phase, in various environments, with varying levels of multipath. For this experiment the ThingMagic M6e radio was used along with the circularly polarized reader antenna and the Alien Squiglette tag. The first set of experiments was conducted in relatively "low multipath" environments, which included semi and fully anechoic chambers. The second set of experiments was conducted in "normal multipath" settings, including open spaces within warehouse or laboratory environments. The final set of data was collected in "high multipath" environments, which were cluttered warehouse and laboratory settings. A description of these environments can be found in Appendix A, section A.3. In each of the various settings the tag was placed at varying separation distances and positioned in a vertically polarized orientation. The absolute distance was based upon the separation distance measured manually, which was found to have a 95% confidence interval of 0.047 cm (see Chapter 12 for error analysis). Finally, the three methods were used to calculate the measured separation distance. The results of these experiments are shown in Table 5-1.

Table 5-1: Average and Standard Deviation of Magnitude of Distance Error from RSSI, Phase, and RSSI-Informed Phase Methods.

Average and Standard Deviation of Magnitude of Distance Error (Meters)			
Average (Standard Deviation)	RSSI	Phase	RSSI Informed Phase
Low Multipath	0.51 (0.54)	0.45 (0.31)	0.26 (0.34)
Normal Multipath	0.58 (0.82)	0.54 (0.69)	0.39 (0.62)
Extreme Multipath	0.74 (0.9)	0.68 (0.73)	0.57 (0.7)

In Table 5-1 the following definitions apply:

- Low Multipath = Anechoic & Semi-Anechoic Environments
- Normal Multipath = Open Warehouse & Laboratory Environments
- Extreme Multipath = Cluttered Warehouse & Laboratory Environments

In each of the situations with varying levels of multipath in the environment, the RSSI-Informed Phase method is the best in terms of the average magnitude of error. Additionally, for each scenario, the standard deviation of error for the RSSI-Informed Phase method is either best or second best. It is only in the case of the lowest multipath environments (the fully anechoic and semi-anechoic chambers) the standard deviation of error for the phase method surpasses that of the RSSI-Informed Phase method.

5.6 Results

Merging the data from the Multipath Experiment together, a histogram of distance estimation error was created which can be seen in Figure 5-8. The MATLAB code which produced this figure can be found in Appendix C, section C.5.

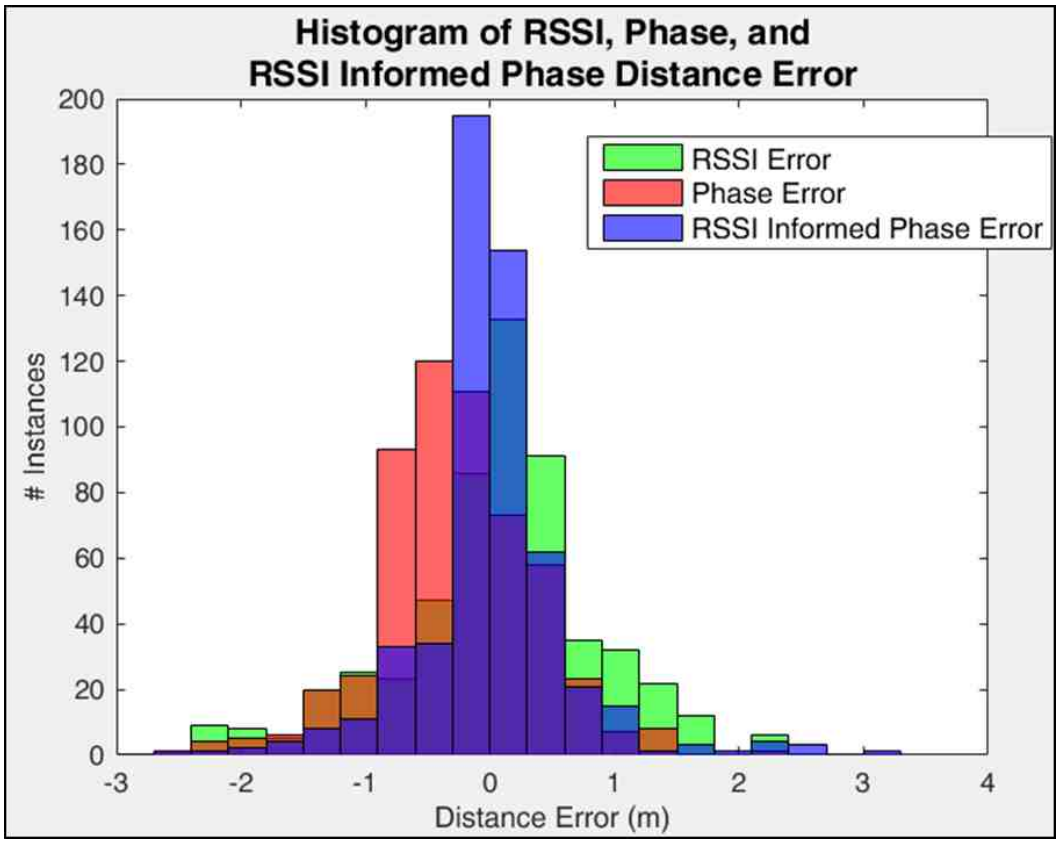


Figure 5-8: Histogram of RSSI, Phase, and RSSI-Informed Phase distance estimation error, various multipath environment data combined average error: Phase = 0.5 m, RSSI = 0.6 m, RSSI-Informed Phase = 0.4 m

The RSSI-Informed Phase method has the most zero-centered error distribution, yet also has the furthest outliers on either side of the bell curve. In analyzing the distribution of error, 63.0% of all RSSI-Informed Phase measurements are within ± 0.3 m of zero (the center two bars), as compared to 39.5% for RSSI and 33.2% for phase. All three methods

contain data which fall between -2.42 and 3.02 m, and those maximum and minimum values are from the RSSI-Informed Phase method. It should be noted even the most extreme errors fall within the RFID tag's read distance. In Figure 5-8 the most extreme errors on either side of the histogram are from the RSSI-Informed Phase method. This is likely due to one of the phase data points being incorrectly moved, which would occur when the RSSI slope was somewhere in the middle ($\frac{\pi}{2}$) of a jump from the original phase point. Nevertheless, this method still maintains the highest level of overall accuracy.

5.7 Discussion

An ideal method of distance estimation is be both accurate and robust. The accuracy would be demonstrated by a low average magnitude of error, and robustness by consistently low error in a wide range of conditions and a low standard deviation.

For the RSSI-Informed Phase method the average magnitude of error was the lowest for a range of situations including: mobile tag, varying degrees of polarization mismatch, and different levels of multipath. This demonstrates not only a general accuracy, but the overall robustness of this method of distance estimation, making it highly suitable for a distance based localization algorithm designed for "real world" environments and applications.

5.8 Conclusion

The RSSI-Informed Phase method for distance estimation was compared to both RSSI and phase methods of distance estimation in a variety of situations: mobile tag, polarization mismatch, and varying levels of environmental multipath. In all situations the

RSSI-Informed Phase had the lowest average magnitude of error. The overall consistency and accuracy of the RSSI-Informed Phase method is extremely compelling. It demonstrates that the RSSI-Informed Phase method is likely the best method for calculating the separation distance of an RFID tag given a variety of unknown conditions and possible sources of error.

CHAPTER 6. REDUCING RF DISTANCE ERROR BY CHARACTERIZING MULTIPATH

6.1 Abstract

This chapter is a slightly modified version of "Reducing RF Distance Error by Characterizing Multipath" published in IEEE Transactions on Instrumentation and Measurement in November 2018 [3] and explores the RSSI versus frequency pattern (the RSSI signature) and its ability to reliably quantify the effect of environmental multipath, specifically on RSSI-derived distance measurements for RFID and BLE systems. Radar technology has demonstrated the use of frequency information for range measurements, given an extremely large bandwidth. In contrast this work shows the applicability of these concepts to the ultra-high frequency Radio Frequency spectrum and the relatively narrow bandwidth. Primarily this chapter will focus on RFID technology, but section 6.8 will also verify the applicability to BLE devices and distance calculations.

First, this chapter presents a theoretical model illustrating the need for frequency information to separate multipath error from RSSI measurements. Practically, a closed-form method to extract the multipath component using data from a complex environment is not feasible; therefore, a neural network is used to emulate theoretical variable separation and extract measurement error from multipath via the RSSI signature. The subsequently predicted distance error not only captures the error magnitude, but also informs the direction of the error, thus making it possible to compensate for this error and significantly improve the original distance prediction, even in a completely new environment.

6.2 Introduction and Related Work

There are many commercial applications for radio frequency identification (RFID), and an increasing number of these applications include localization [44] [17] [28] [29] . RFID localization techniques are typically classified into 5 categories: trilateration/multilateration, triangulation, hybrid direction/range methods, radio map matching methods, and proximity sensing as discussed in greater detail in Chapter 2 [45]. Trilateration/multilateration techniques are based on multiple range estimations from satellite points, while triangulation relies on direction of arrival measurements from multiple points, and hybrid techniques use a combination of these approaches. In radio mapping, such as fingerprinting, an RF environment is charted, and new measurement signals are compared to stored information for a closest match.

Each method balances the trade-offs between accuracy, logistical complexity and flexibility. While some applications employ additional means for enhanced accuracy, such as establishing reference points, for example in a matrix of devices covering an area of interest, or frequent mapping of the environment to mitigate changes, these methods are not appropriate for all applications.

Many applications for localization are more appealing to potential customers if they allow for maximum flexibility of installation and modification, and minimal up-front investment. For example, radio mapping can be quite accurate, but requires a significant amount of time and human effort to maintain. Reference tag systems can always be made more accurate with additional reference locations, but this also means additional time and cost [17].

When investigating logistically simple and inexpensive methods of indoor localization, range estimation input to a trilateration algorithm is one of the best, if not the best alternative as discussed in Chapter 2. However, range estimation measurements therefore become the main source of error for these methods, specifically due to issues with multipath [17].

Currently in the literature, there are three main methods of dealing with multipath error in range estimation. The first and most common is to model multipath error as random noise and mitigated accordingly, for instance using Kalman filters [17], [46], [47]. The second is to create a simple multipath model with a single or double bounce. Research in radar based target location has used Doppler mapping and Time Of Arrival (TOA) to address outdoor moving targets, which allowed for some exploitation of multipath signals; however, when modeling signals, only a single bounce or double bounce for multipath was assumed [48]. For indoor RFID sensing modeling, Subedi et al. [28] created a simulation with included multipath, but limited the situation scenarios to ones where a single bounce would be the only significant multipath factor. While this process was promising for a simulation or a semi-anechoic environment, an actual complex environment could have a multitude of multipath signals with similar intensity, making accurate distance estimation challenging. In some instances, RSSI signals from multipath can exceed its direct measurement in signal strength.

A third and more effective approach was described by Wu et al. for stepped frequency Through-the-Wall Radar Imaging (TWRI) [47]. TWRI is used to identify military targets hidden behind opaque obstacles. Unlike traditional TWRI, which assumes a frequency-independent scattering model, Wu et al used frequency dependent signals to

improve target identification accuracy. In the TWRI approach, transmitted signals span an extremely wide bandwidth, “from Megahertz to a few Gigahertz” and the radar cross section principle (RCS) suggests that the frequency impact on multipath signal is not negligible across this large range. The stepped frequency approach, combined with this algorithm, and constraints designating where targets could be found, improved accuracy in the experiment.

The theory of frequency based multipath information described by Wu et al. was founded upon the assumption that the information could be extracted owing to the extremely wide bandwidth (MHz-GHz). In contrast, commercial use of range measurements requires a relatively narrow bandwidth to operate within regulatory (FCC) requirements. Therefore, the wide bandwidth assumption, which can be used effectively for military purposes, is not available for normal ultra-high frequency (UHF) applications. It would be significant if frequency based multipath, such as that used effectively for military TWRI, also contains enough information for range improvement within a narrow frequency range, like that of UHF RFID or BLE.

In this chapter, a new method is proposed to correct range estimation error for passive UHF RFID and secondarily for BLE in the presence of multipath. The proposed method improves upon similar that described in Wu [47] by examining the frequency response in the North American (NA) 902-928 MHz band (and 2.402-2.480 GHz band) and its ability to distinguish multipath, extracting relevant information about the multipath in the environment in this case by using a neural network. Typical results from experiments in this chapter showed a 20% to 60% reduction in error in similar environments and a 12% reduction in range error for a completely new, untrained, complex environment.

This chapter will next describe a theoretical model which explicitly considers the effect of frequency on the direct and multipath signal components is presented. Addition of the frequency response signal allows for the theoretical separation of multipath error from the primary signal. Next, experimental results are presented to demonstrate the influence of multipath from the surrounding physical environment on RSSI signatures, extending the authors' previous work [2]. Finally, the main contribution of this chapter is to demonstrate that an RSSI signature even within a narrow frequency band can enable separation of direct and multipath signals. A trained neural network is applied to extract this multipath information from the RSSI signature in real environments and utilize it to reduce range estimation error.

6.3 Background

Due to the low cost of an individual passive UHF RFID tag, this technology is frequently deployed to track a large numbers of objects. BLE is also becoming wide spread, and is used for tracking fewer objects over a longer space. Range errors in localization measurements have slowed the adoption of RF localization technology, resulting in a strong push for reducing error in distance measurements [30], [34], [14], [49].

6.3.1 Signal strength response to excitation frequency

There are many factors which are known to impact RSSI values, including impedance mismatches and polarization mismatching. Additionally, multipath is known to have a considerable effect [30], [34]. The impact of the carrier frequency on the RSSI measurements for UHF applications is an area that has not been well-studied. In Wu et al. [50], it is noted that the impact on RSSI from the multipath of the surrounding environment

and the observation that RSSI is not constant with respect to frequency. Conference publications represented by both this chapter and Chapter 5, this phenomenon is further explored, and its potential utility as a unique RSSI vs. frequency signature and its relation to multipath is demonstrated [2], [1]. The RSSI signature is simply the measurement of RSSI values over a spectrum of carrier frequencies. This RSSI signature is further shown to be repeatable under identical multipath conditions. An example RSSI signature is shown in Fig. 6-1. The experimental set-up used to generate this data is given in Appendix A, section A.4.9.

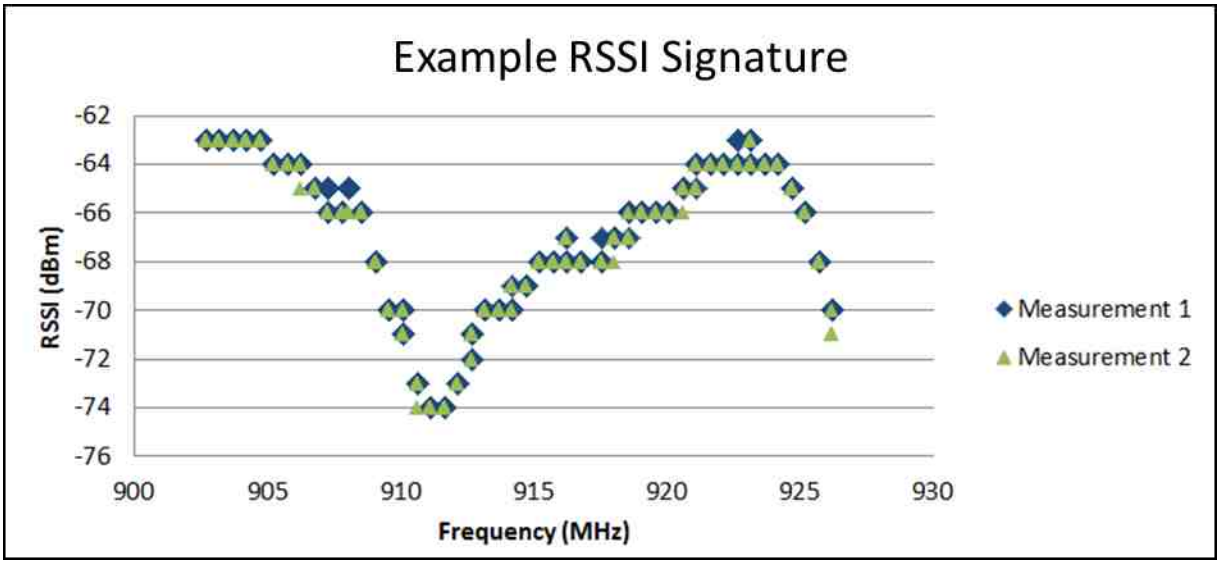


Figure 6-1: Example RSSI signature (RSSI vs. Frequency)

The RSSI signature represented in Figure 6-1 demonstrates not only how significantly RSSI can vary with respect to the carrier frequency, but also the repeatability of this signature given the same environmental surroundings and multipath. The United States Federal Communication Commission rules specify that a radio in the 902-928 MHz frequency band must jump between a minimum of 25-50 different carrier frequencies

within that range over a brief time interval to allow for frequency sharing between multiple devices operating in that bandwidth. The 50 hop frequencies shown in Figure 6-1 are those used by the ThingMagic M6e radio in observance of this requirement. In the following sections, a new approach is presented, in order to demonstrate the novel finding that the RSSI signature can be used to significantly reduce error in RFID distance estimations due to multipath. This work is an extension of the authors' previous investigations [2].

6.4 Theoretical Discussion

The hypothesis of this chapter is the concept that multipath error cannot be sufficiently identified by varying just distance and environment, and the frequencies used to obtain the measurements are critical in accounting for multipath effects, even across a relatively narrow bandwidth. This theoretical section is provided to demonstrate the necessity of utilizing variations with respect to frequency for isolating multipath error, without which the decoupling would not be possible.

Given a measurement (M) of the reported value of signal strength from an RFID tag or passive UHF RF device as seen by the receiving radio, the measurement will include environment multipath and the impact from the instrumentation, the variations of which are assumed to be included but not expressed by analytic equations. As discussed later in this section, the impact of the equipment, like multipath, varies with respect to frequency, and is therefore included. This measurement, M, can be defined as:

$$\mathrm{M} = P_{RD} + \sum_{j=0}^{j=K} (P_{RM} \sin \psi)_j \qquad \text{Equation 6-1}$$

Where:

P_{RD} is the power received from the tag's direct signal (in Watts),

P_{RM} is the power received from a tag's multipath signal (in Watts),

ψ is the phase offset between the direct signal and multipath signal (radians) and

K is the number of multipath signals.

In Equation 6-1, additional error due to factors such as the quantization of the reported RSSI, is assumed to be normally distributed and negligible in this study. Thus, an error term representing the signal's multipath can be obtained from Equation 6-1; this error term is represented as Φ and also given in Watts.

$$\Phi = \sum_{j=0}^{j=K} (P_{RM} \sin \psi)_j \qquad \text{Equation 6-2}$$

All waves traveling indirect paths will travel a greater total distance to the receiver than the direct wave. The time difference in arrival between the direct and various indirect signals at the receiver will result in a phase difference (ψ). This phase difference can yield either constructive or destructive interference. Changes in the carrier frequency cause these constructive and destructive interference effects to shift and result in changes in amplitude with respect to frequency. In a simple environment, one might expect a sinusoidal transition with frequency changes but, with increasing complexity of the multipath of the environment, this amplitude to frequency relationship becomes more intricate and representative of the environment's unique multipath.

The complexity resulting from the compounding multipath signals means that even small changes in the environment can have a drastic impact on the multipath error. Therefore, in a real-world situation, it is feasible to assume that multipath error may change due to even small changes in the environment. This is true even if the separation distance and direct path remain constant, as described by Equation 6-3.

$$\mathrm{M}_1 = \Phi_1 + P_{RD} \neq \mathrm{M}_2 = \Phi_2 + P_{RD} \qquad \text{Equation 6-3}$$

Several traditional approaches to indoor RF distance measurements and localization systems, such as radio mapping (fingerprinting), chart the RSSI response as a function of the location (*x, y*) for all areas of interest in an environment. A measurement M would therefore be recorded for each of the x and y coordinates, specifying the location of the tag or device with respect to the radio. These measurements are then used as a look-up table to convert any future measurement in that environment back to a separation distance, *s*. In this method, a measurement M is understood to be dependent on the separation distance and unique to the particulars x, y location with the multipath environment, as represented in Equation 6-4.

$$\mathrm{M}_{xy} = (\Phi + P_{RD})_{xy} = f(s) \qquad \text{Equation 6-4}$$

The issue with the fingerprinting method is twofold. Firstly, the mapping process is time consuming, costly, and labor intensive. Secondly, Φ_{xy} is not guaranteed to be constant with time, due to typical changes in real-world environments. A model for this type of variation could be given such that:

$$\mathrm{M}_{xy}(t = 1) \neq \mathrm{M}_{xy}(t > 1) + tolerance \qquad \text{Equation 6-5}$$

Using this type of model in a control algorithm for an actual localization implementation, if the difference in measurements is greater than the tolerance, then the re-mapping process must be repeated to update the table of reference measurements to maintain the specified system accuracy.

It is known that the variation in antenna gain with respect to frequency has a non-trivial and measurable effect on RSSI data [14], and similar variations were empirically verified in Chapter 12. Thus, the gain of both the transmit and receive antennas are functions of frequency, as shown in Equation 6-6.

$$G_R, G_T = f(\lambda) \qquad \text{Equation 6-6}$$

In this equation G_R is the gain of receiving antenna (dB), G_T is the gain of transmitting antenna (dB) and λ is the wavelength (meters). Thus, the power received from the tag's direct signal is dependent on the carrier wavelength, as can be seen in Equation 6-7.

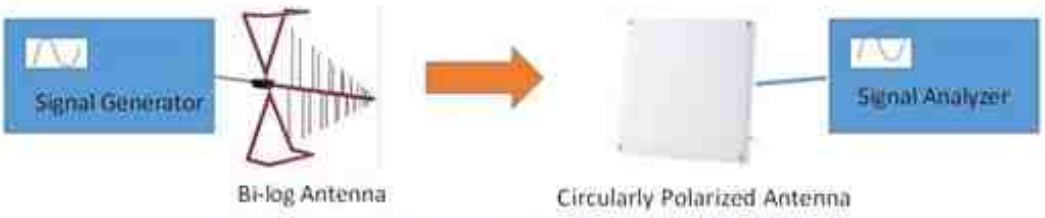
$$P_{RD} = f\big(\lambda, G_R(\lambda), G_T(\lambda)\big) \qquad \text{Equation 6-7}$$

This forms a unique relationship such that the power from the direct signal is not only based upon the separation distance, but also has a dependence on the frequency (or wavelength) as well. Therefore, if the separation distance was held constant, the result is expected to be a distinct pattern of received power with respect to frequency, related to the hardware of that system.

Without this additional understanding of the frequency dependence of the hardware, the only frequency or wavelength dependence in free space transmission would be from the carrier wave frequency. The dependence of a circularly polarized antenna on

frequency was measured in a 5 meter anechoic chamber as described in Chapter 12, and the results illustrate this effect.

The experimental set-up and the resulting measured signal variation due to frequency are shown in Figure 6-2. In this experiment, the received amplitude was compared to the transmitted amplitude at three different frequencies and, taking into account other factors such as the bi-log antenna gain, the cable losses for the 6ft long cable, and the pre-amp, the antenna gain could be determined. The experimental set-up is given in Appendix A, section A.4.10.



Frequency (MHz)	Antenna Gain (dB)
900	5.99
914	5.44
930	4.48

Figure 6-2: Antenna gain measurements as a function of the sent frequency

The complex relationship between antenna gain (and other equipment properties) and frequency means that even a small difference in the carrier frequency yields a new feature in the coupling of the transmit and receive antennas. However, this new feature can potentially serve as additional information used to separate the direct signal power, needed for distance calculations, from the multipath error.

If a set of measurements is considered, each would have some separation distance *s*, and vary by x and y locations. The set of measurement data, X_s, which is a set of measurements containing both multipath error Φ_{f,s_n} and the base direct signal, $P_{RD_{f,s_n}}$, is shown in Equation 6-8.

$$X_S = \begin{bmatrix} \mathrm{M}_{s_1} \\ \mathrm{M}_{s_2} \\ \vdots \\ \mathrm{M}_{s_n} \end{bmatrix} = \begin{bmatrix} \left(\Phi_{f,s_1} + P_{RD_{f,s_1}}\right) \\ \left(\Phi_{f,s_2} + P_{RD_{f,s_2}}\right) \\ \vdots \\ \left(\Phi_{f,s_n} + P_{RD_{f,s_n}}\right) \end{bmatrix} \qquad \text{Equation 6-8}$$

Each set of training data, *X*, varies separation distance *s* between the radio and tag or RF device. Even with many training examples, which vary the environmental noise, it is not possible using this approach to separate the multipath error Φ, from the direct signal P_{RD}, because both are related to the wavelength of the carrier signal.

In free-space conditions [13], $\Phi_{f,s} = 0$, such that each measurement M, is simply given by:

$$\mathrm{M} = P_{RD_{f,s}} \qquad \text{Equation 6-9}$$

This is because in free-space conditions there is no multipath effect, and all measurements are functions of the direct signal. If a series of training measurements was taken in free-space conditions with the same hardware, while varying distance, the result would only be a function of the direct signal and the carrier frequency wavelength. This is shown in the free-space training set in Equation 6-10, where ΔX_S is the difference in measurement data, $\frac{\partial \mathrm{M}}{\partial s_n}$ is the derivative of measurement (M) with respect to separation

distance (s) and $\Delta P_{RD_{f,s_n}}$ is the change in power of the tag's direct signal which is dependent on frequency f, and separation distance s_n.

$$\Delta X_s = \begin{bmatrix} \left(\frac{\partial \mathrm{M}}{\partial s_1} \Delta P_{RD_{f,s_1}}\right) \\ \left(\frac{\partial \mathrm{M}}{\partial s_2} \Delta P_{RD_{f,s_2}}\right) \\ \vdots \\ \left(\frac{\partial \mathrm{M}}{\partial s_n} \Delta P_{RD_{f,s_n}}\right) \end{bmatrix} \qquad \text{Equation 6-10}$$

If instead, a set of training examples was collected in a real-world environment, the additional multipath error term would be present. The variations in measured distance for a real-world data set contain both multipath and direct power terms, as shown in Equation 6-8. In contrast, the real-world training set, represented by Equation 6-11, contains the two signal components, but accounts for variations with respect to frequency. In this equation $\Delta \Phi_{f,s_n}$ is the change in multipath error between measurements, and is also dependent on frequency f, and separation distance s_n.

$$\Delta X_s = \begin{bmatrix} \left(\frac{\partial \mathrm{M}}{\partial s_1} \Delta \Phi_{f,s_1} + \frac{\partial \mathrm{M}}{\partial R_1} \Delta P_{RD_{f,s_1}}\right) \\ \left(\frac{\partial \mathrm{M}}{\partial s_2} \Delta \Phi_{f,s_2} + \frac{\partial \mathrm{M}}{\partial R_2} \Delta P_{RD_{f,s_2}}\right) \\ \vdots \\ \left(\frac{\partial \mathrm{M}}{\partial s_n} \Delta \Phi_{f,s_n} + \frac{\partial \mathrm{M}}{\partial R_n} \Delta P_{RD_{f,s_n}}\right) \end{bmatrix} \qquad \text{Equation 6-11}$$

In most RSSI-based distance or location calculations, only an average frequency for the range is assumed rather than the precise hop frequency for each measurement. These variations in frequency within training sets are generally neglected for distance and location calculations, but are actually significant because two measurements even at the same separation distance if at different carrier frequencies both the power from the direct

signal as well as differences in multipath, as described by the difference in measurements in Equation 6-12.

$$\Delta P_{RD_{f,s_i}} \neq 0$$

$$\Delta \Phi_{f,s_i} \neq 0 \qquad \text{Equation 6-12}$$

Furthermore, without specifically accounting for the carrier frequency, there is simply not enough information in the collected complex environment data to effectively separate multipath error from the power of the direct signal.

The method proposed in this chapter attempts to separate multipath error from the power of the direct signal by considering each measurement to be a function of both separation distance and carrier frequency. Thus, a series of measurements $X_{f,s}$ are obtained, by varying both the frequency and separation distance, as shown in Equation 6-13 where n is the number of separation distances in training set and m is the number of hop frequencies in training set.

$$X_{f,s} = \begin{bmatrix} \mathrm{M}_{f_1 s_1} & \mathrm{M}_{f_2 s_1} & \cdots & \mathrm{M}_{f_m s_1} \\ \mathrm{M}_{f_1 s_2} & \mathrm{M}_{f_2 s_2} & \cdots & \mathrm{M}_{f_m s_2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{M}_{f_1 s_n} & \mathrm{M}_{f_2 s_n} & \cdots & \mathrm{M}_{f_m s_n} \end{bmatrix} \qquad \text{Equation 6-13}$$

Given any separation distance s_i, Equations 6-4 and 6-13 can be combined to yield Equation 6-14:

$$X_{f,s_i} = \begin{bmatrix} \left(\Phi_{f_1,s_i} + P_{RD_{f_1,s_i}}\right) \\ \left(\Phi_{f_2,s_i} + P_{RD_{f_2,s_i}}\right) \\ \vdots \\ \left(\Phi_{f_m,s_i} + P_{RD_{f_m,s_i}}\right) \end{bmatrix} \qquad \text{Equation 6-14}$$

A closed-form solution to extract the multipath component using data from a complex environment is not feasible for real environments; therefore, a neural network is used to emulate variable separation and extract measurement error due to multipath.

Training examples are created specifically to incorporate variations in the surrounding physical environment. These differences in the environment result in variation in the multipath error Φ, but the direct signal will be a consistent function of frequency/wavelength and distance, based upon the hardware. Thus, the difference in training sets is given by:

$$\Delta X_{f,s_i} = \begin{bmatrix} \left(\frac{\partial \mathrm{M}_{f_1,s_i}}{\partial \Phi_{f_1,s_i}} \Delta\Phi_{f_1,s_i} + \frac{\partial \mathrm{M}_{f_1,s_i}}{\partial P_{RD_{f_1,s_i}}} \Delta P_{RD_{f_1,s_i}}\right) \\ \left(\frac{\partial \mathrm{M}_{f_2,s_i}}{\partial \Phi_{f_2,s_i}} \Delta\Phi_{f_2,s_i} + \frac{\partial \mathrm{M}_{f_2,s_i}}{\partial P_{RD_{f_2,s_i}}} \Delta P_{RD_{f_2,s}}\right) \\ \vdots \\ \left(\frac{\partial \mathrm{M}_{fm,s_i}}{\partial \Phi_{fm,s_i}} \Delta\Phi_{f_m,s_i} + \frac{\partial \mathrm{M}_{fm,s_i}}{\partial P_{RD_{fm,s}}} \Delta P_{RD_{f_m,s_i}}\right) \end{bmatrix} \qquad \text{Equation 6-15}$$

Unlike Equation 6-12, when frequency is accounted for, in Equation 6-15 the change in power from the direct signal given a constant frequency and separation distance should be zero.

$$\Delta P_{RD_{f_j,s_i}} = 0 \qquad \text{Equation 6-16}$$

For a given s_i, Equation 6-15 can be simplified to the following form:

$$\Delta X_{f_j} = \frac{\partial \mathrm{M}_{f_j}}{\partial \Phi_{f_j}} \Delta \Phi_{f_j} \qquad \text{Equation 6-17}$$

Equation 6-17 shows that the signal's multipath error has been theoretically separated from the power of the direct signal, which is used to obtain separation distance. This separation of the multipath error makes it possible to account for it and ultimately reduce error in range estimation. Without the additional frequency dimension, this separation would not be possible. For this reason multipath effects cannot be adequately accounted for if the specific carrier frequency is ignored, as is the case in most applications.

In practice, this separation is extremely complex, but the relationship between $P_{RD_{f_j}}$ (the direct power) and $P_{RM_{f_j}}$ (a 2-D pattern of the multipath power) is a correlation machine learning algorithms are well suited to resolve. A series of experiments were performed to validate the hypothesis that a separation between multipath error and the direct signal was feasible in a bandwidth such as UHF, and this separation could potentially be used to improve the distance estimation.

6.5 Experiments

The experiments in this section used the following equipment: a ThingMagic M6e radio with a circular polarized reader antenna [51], and an Alien Squiglette passive UHF RFID tag [4], vertically polarized. The reader antenna and tag are both at a height of 1 meter above the ground. Appendix A, sections A.4.9 – A.4.11 can been seen for additional information on experimental set-up.

6.5.1 Distance Measurements Using RSSI

When the RFID tag is measured in a fully anechoic chamber, simulating free space propagation loss, the RSSI vs. separation distance curve closely matches the Friis transmission formula [13].

Figure 6-3 compares this theoretical free-space Friis transmission formula, Equation 3-1, to the measured fully anechoic data, along with data from semi-anechoic chamber in comparison to a simulated single ground plane reflection [18]. The data sets in Figure 6-3 are normalized with respect to the maximum values of each curve for ease of comparison; the actual offsets between the theoretical free-space and ground plane models and the experimental data presented in Figure 6-3 are likely due to the power lost from backscatter from the passive RFID tag as well as any errors in antenna gain or cable loss [31] , [27] additionally discussed in Chapters 4 and 12. Experimental set-up for Figure 6-3 is given in Appendix A, section A.4.3.

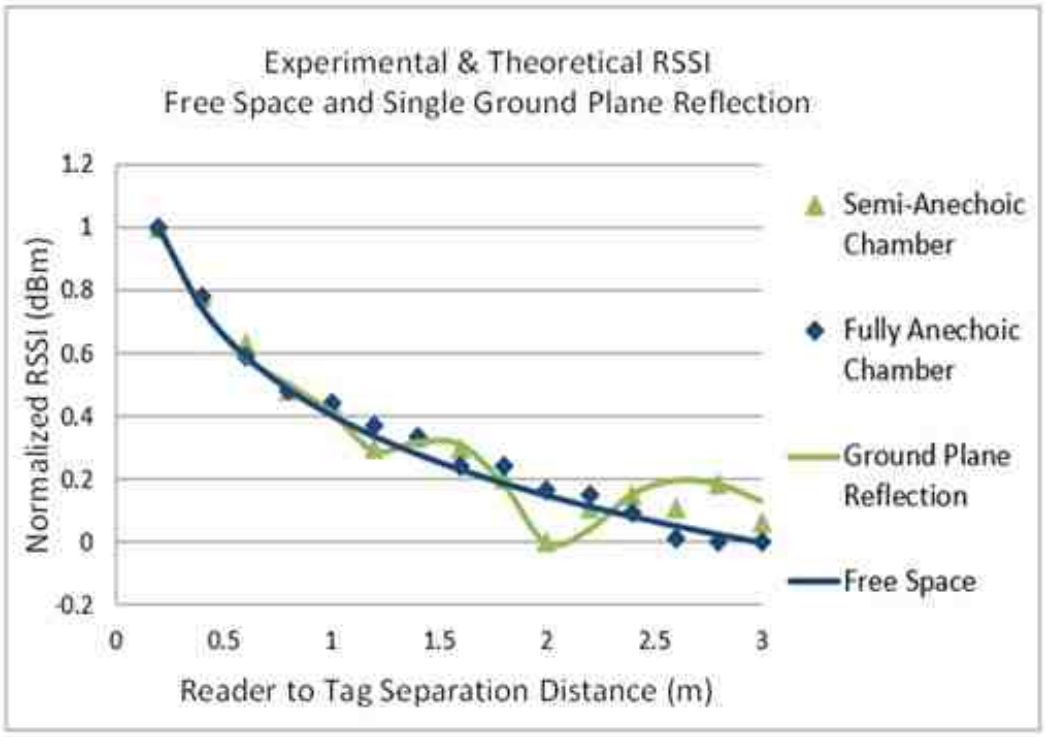


Figure 6-3: RSSI vs. Distance for ideal free space conditions and a single ground plane reflection

As described in Chapter 3 and section 6.2, multipath is the phenomena where the radio signal takes more than one path from the transmitter to receiver due to the presence of one or more nearby reflectors. When the signals collide at the receiver, the possible outcomes range from a complete cancellation of the signal to an increase in signal amplitude. For example, in Figure 6-3 the single ground plane refection both increases and decreases the RSSI in comparison to free-space, due to this constructive and destructive interference effect.

6.5.2 Experiment: Moving Entire Setup by Increments

It was hypothesized earlier that small changes in the environment would yield gradual changes in the RSSI signature due to multipath. To confirm this hypothesis, the following experiment was designed using a room filled with common office objects

including wood and metal furniture, providing a range of multipath sources. See Appendix A, section A.3 for a more detailed description of this environment. The goal was to change the surrounding environment slowly enough that the gradual transformation in the RSSI signature could be observed. Therefore, the RFID reader and tag were kept at a constant separation distance and orientation with respect to each other. Then together the reader and tag were moved by small (approximately 0.1 wavelength increments) through the non-anechoic environment described above. For each increment, the RFID tag was read at every hop frequency and the RSSI signature recorded.

When analyzed, it was found that gradually shifting the RFID setup through the "real world" environment did result in gradual transitions in the RSSI signature, as expected. A smooth change in the signature was observed as the RFID tag and reader together moved through the environment. To better observe this transition, 4^{th} order polynomials fits for the (quantized) signatures were used to approximate the "true signature." Examples of these curve fits are presented in Figure 6-4.

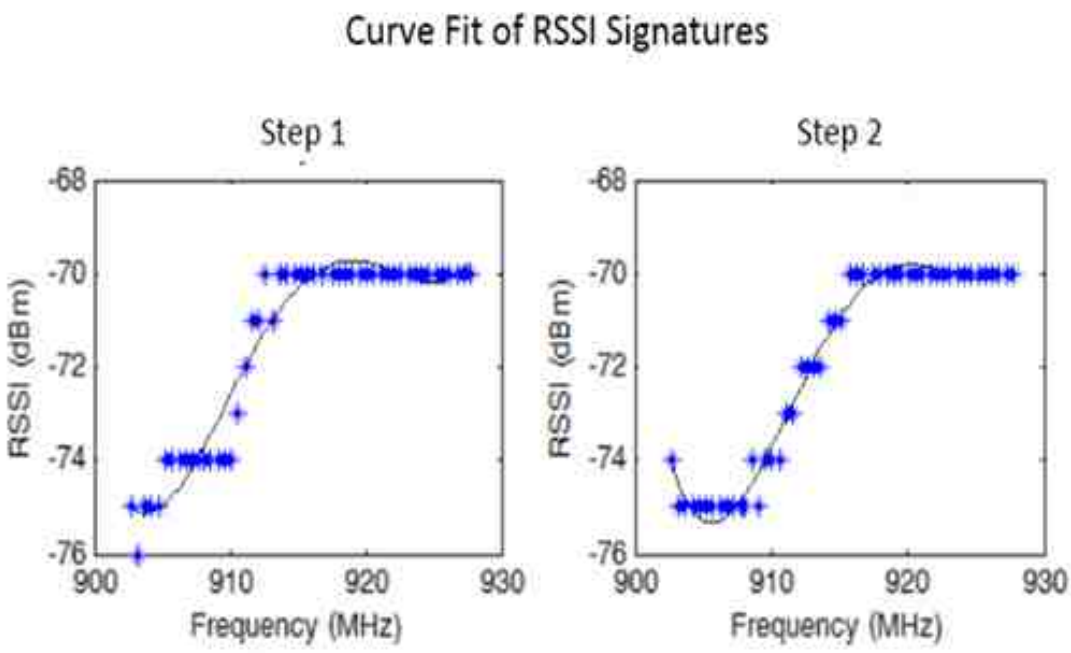


Figure 6-4: 4th order polynomial fit of typical sequential signatures.

Figure 6-5 and 6-6 utilize the 4th order polynomial fits to illustrate this transition of the RSSI signature as the setup moved through the environment using the sequential steps as the third dimension. The shape of the signature gradually changes, revealing a fluid transition from one signature to the next in a wave-like motion. The experimental set-up for Figures 6-5 and 6-6 is given in Appendix A, section A.4.11. The MATLAB code which was created to generate these figures is given in Appendix C, section C.6.

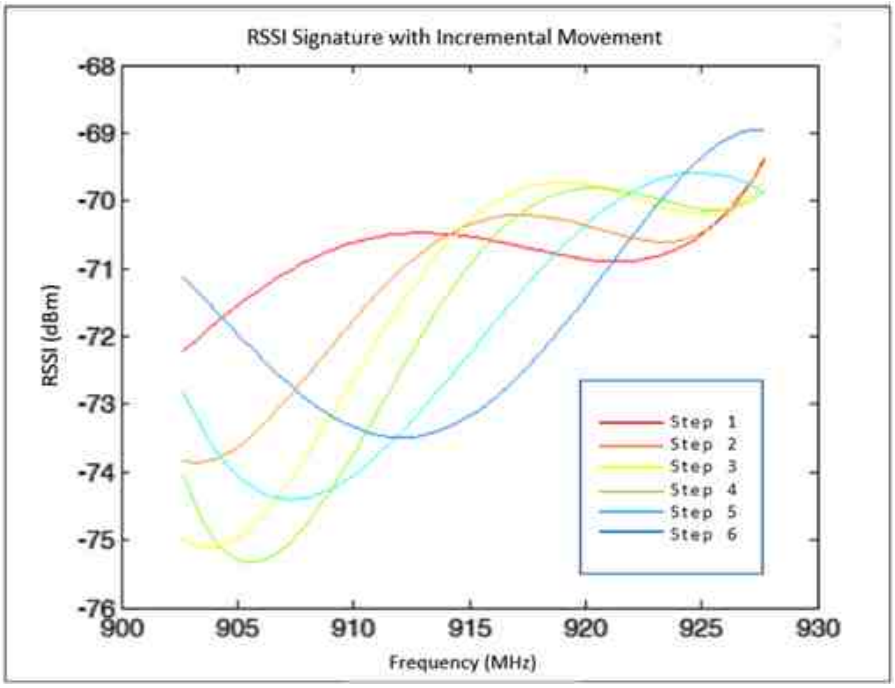


Figure 6-5: Wave effect of signature with sequential shifts in environment in 2D

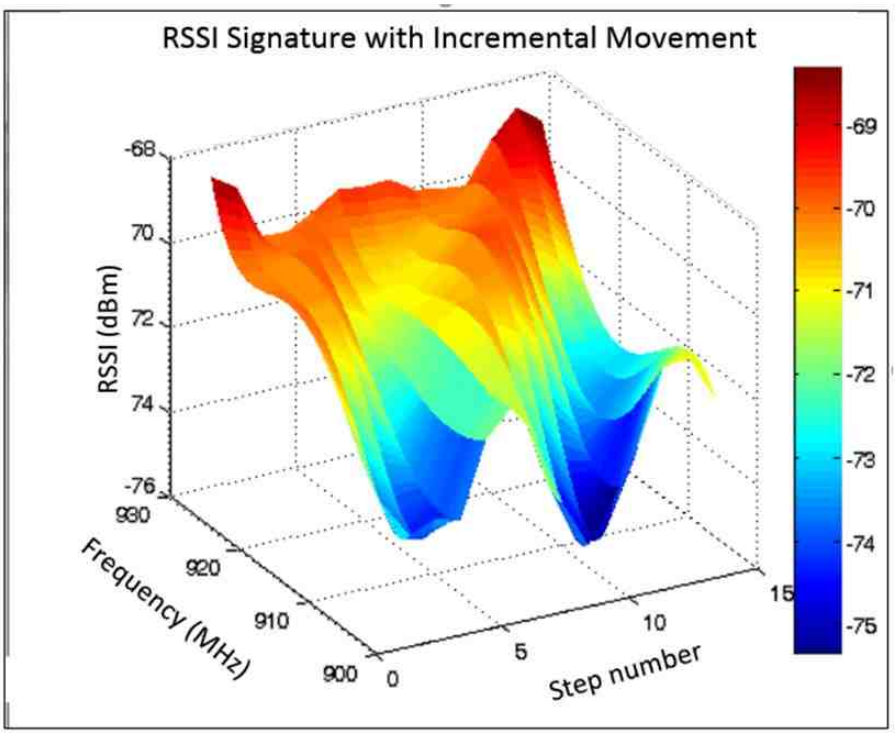


Figure 6-6: Wave effect of signature with sequential shifts in environment in 3D

As predicted and demonstrated in the figures above, gradual wave-like transitions were observed in the measured RSSI signatures. Due to the fact that the reader and tag remained stationary with respect to each other, the changes in the RSSI signature therefore relate directly to changes in the surrounding environment. Thus, these wave-like transitions of the signatures are likely due to gradual multipath changes caused by the shifting position of reflective surfaces surrounding the RFID setup. This experiment served as another validation that the multipath of the surrounding environment has a significant impact on the RSSI signature, resulting in a unique representation of the surrounding multipath, even across a relatively narrow bandwidth.

6.6 Machine Learning Analysis

This initial work indicated both that the RSSI signature was responding strongly to multipath across the UHF bandwidth, and that this response was repeatable for a constant environment, as had been theoretically predicted and hypothesized. The experiment described in Section 6.5 demonstrated how a different location of the RFID setup even within the same setting can result in different RSSI signatures. The process of decoupling the base signal from the multipath was found to be sufficiently complex, so a neural network was selected as an appropriate tool for pattern recognition and separation.

For a neural network, each element of an input dataset is called a feature, and these datasets are called observations. In order to train a neural network one must input a large number of observations in proportion to the number of features; at least 10 times as many. For this analysis, the input features were the RSSI values at each of the 50 hop frequencies, providing the neural network with a complete RSSI signature.

A sample size of greater than 500 signatures was therefore required for a valid analysis. These signatures were collected in six different complex environments, including warehouse-like environments with large open spaces and pallets of product, laboratory settings with a variety of office furniture and laboratory equipment, and a room with an arrangement of typical domestic furniture. Appendix A, section A.3, further describes these environments. The variety of environments was important because an assortment of settings would likely yield diverse RSSI signatures from varying amounts and types of multipath. It was expected that this diversity would yield an algorithm (trained neural network) that would generalize better.

One other key component needed for the neural network analysis was the desired output, otherwise known as the prediction. Based on the theoretical model from section 6.4, the neural network was trained using the RSSI signatures to predict the multipath error in RSSI-based distance estimations. The measured error in RSSI-based distance estimations was found by the difference between the calculated separation distance and the physically measured distance from the reader antenna to the tag, and assumed to be primarily caused by multipath. If this assumption is incorrect, the result would be that the neural network would be unable to significantly predict and reduce distance error.

The neural network selected to model the relationship between the RSSI signature and separation distance was a machine learning library-based MATLAB configurable model. The three-hidden-layer, fully connected, neural network was trained using a random 90% of the data, tested using the remaining 10%. Care was taken not to allow data contamination during training. Incomplete RSSI signatures (those sets with missing hop

frequencies) were not used for this analysis, due to the fact that neural networks cannot handle incomplete data sets [52] [53]. Chapter 7 will address this challenge.

The physically measured distance, which for these purposes was considered to be the true value, had a 95% confidence interval of 0.047cm. The confidence interval is discussed in greater detail in Chapter 12. For the original distance predictions used as the control, RSSI values were measured at each of the 50 hop frequencies. An average of these 50 RSSI values was then compared to an RSSI vs. distance best fit to generate the predicted distance for the control. If fluctuations in frequency were truly random, then this average should mitigate that error, and there would be no improvement from the frequency values applied to the neural network.

6.7 Results

6.7.1 Reduction in Error Using RSSI Predicted Distance

Neural networks use gradient descent for their nonlinear regression algorithm, which finds the local rather than global minimum, thus several neural networks were trained. Then using the remaining withheld data, consisting of approximately 30 datasets, the following results were generated. Initially, only the absolute value of the error was used; however it was quickly evident that the neural network regression results were capable not only of predicting the magnitude of the distance error, but also whether that error was positive or negative.

Due to the complexity of the relationship, and the nature of non-linear optimization, there were several trained neural networks that were rejected for poor performance due to

convergence on a local minimum. However, often the neural network was able to predict RSSI distance estimation error, and typical improvements ranged from 30% to 70% reduction in error, for data from a similar environment. The results from the 30 datasets of withheld test data from one such neural network are shown in Figure 6-7. This trained network achieved an average of 50.2% reduction in distance error.

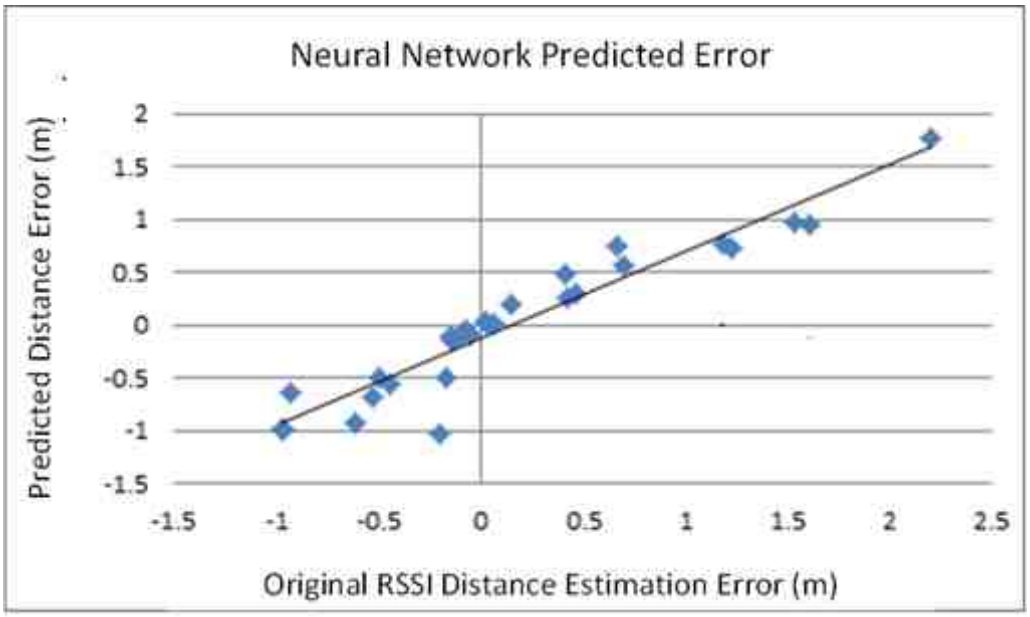


Figure 6-7: Neural network predictions of RFID RSSI distance estimation error. When used to correct error, achieves 50.2% reduction of error.

6.7.2 Applying Signature to Phase Distance Estimations

Another commonly-used method of distance estimation for RFID is phase-based distance estimations [34]. This method relies on the relationship between phase angle and carrier frequency described in detail in Chapter 5 and illustrated in Figure 5-2. If the separation distance between the reader and the tag remains constant, and the carrier frequency is incremented, then the phase angle will also shift. Due to the way in which the phase angle is calculated within an RFID radio, the phase angle will range from 0 to π

radians and appear as a saw tooth curve as shown in in the previous chapter, Figure 5-3. This change in phase angle with respect to the change in frequency yields a line whose slope is proportional to the reader to tag separation distance. This relation is given by Equation 5-2.

Although the use of phase angle for distances estimations is understood to be more robust with respect to multipath than RSSI distance estimations [34], multipath can still significantly impact phase angle measurements and therefore cause errors in distance estimations.

Figure 6-8 shows the experimental set-up to examine the phase vs. frequency response and sensitivity to small changes in environment. The experiment was conducted in a warehouse-like environment (see Appendix A, section A.3). The radio and tag were positioned 1m above the ground plane and secured to non-RF reflecting materials. Figure 6-9 shows a comparison of the phase vs. frequency RFID data taken in the same room at a constant reader to tag separation distance, but with different locations of the metal cart, thus providing a simple modification in the surrounding environment.

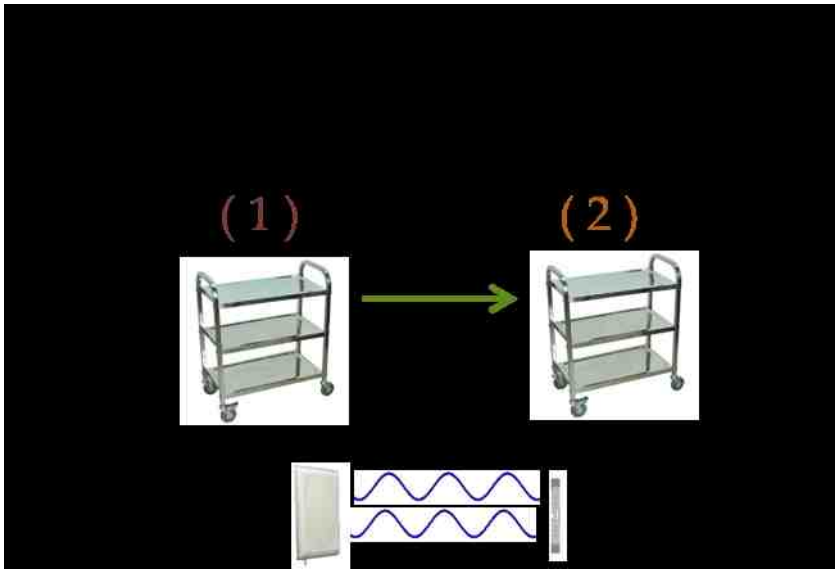


Figure 6-8: Illustration of experiment noting the impact of changes in multipath on phase angle by moving a metal cart from one side of the room to the other.

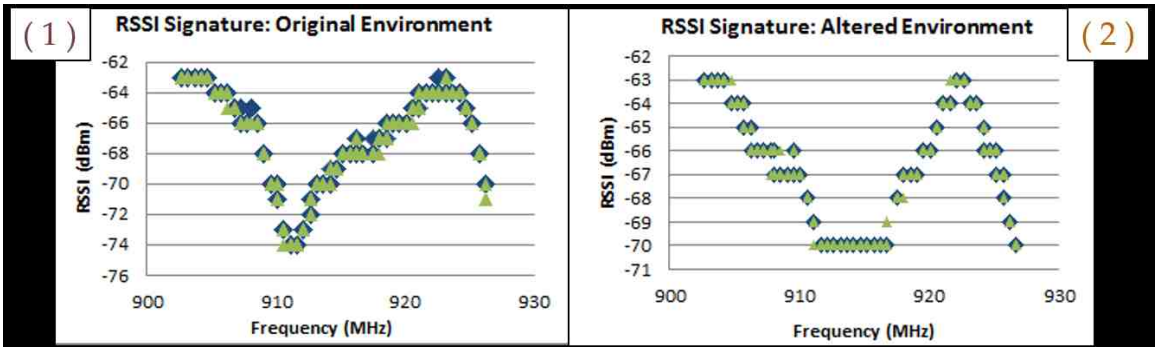


Figure 6-9: Phase vs. Carrier frequency data for cart in Location (1) and (2)

As shown, the change of multipath which occurred when the metal cart was moved had a noticeable impact on the phase angle measurement. If multipath has a significant impact on phase angle, then it could also be expected to be a key contributor to error in phase-based distance estimations. Therefore, it was hypothesized that implementation of the RSSI signature in a neural network might also be able to predict error in phase-based RFID distance estimations.

Similar to the discussion in Section 6.6, a neural network was used to extract multipath error in RFID distance estimation for phase distance estimations. It should be noted that the phase distance error was used as the prediction or output for both training and testing of the neural network, while the standard RSSI signature was used as the input observations.

When using only the 30 sets of withheld test data for performance analysis, it was found that the RSSI signature used in conjunction with a phase-error trained neural network could predict phase distance estimation error and ultimately be used to reduce overall phase-distance error. Typical results ranged from a 20% in environments with low total error due to minimal multipath signals, to 60% reduction in error for environments containing many objects that produced multipath responses. The range of environments was similar to those used in Section 6-5 and discussed in Appendix A, section A.3. Additionally, within an environment, different neural network based algorithms had a variation in performance due to convergence on different local minimums.

An example of neural network predicted error vs. the initial distance error for phase data across multiple types of environments is shown in Figure 6-10. The RSSI signature interpreted by the neural network for phase was similarly able to not only to predict the magnitude of error from the environmental multipath, but also whether the error was positive or negative. This distinction confirms the utility of the neural network as a tool to extract multipath error and improve upon the original phase distance prediction.

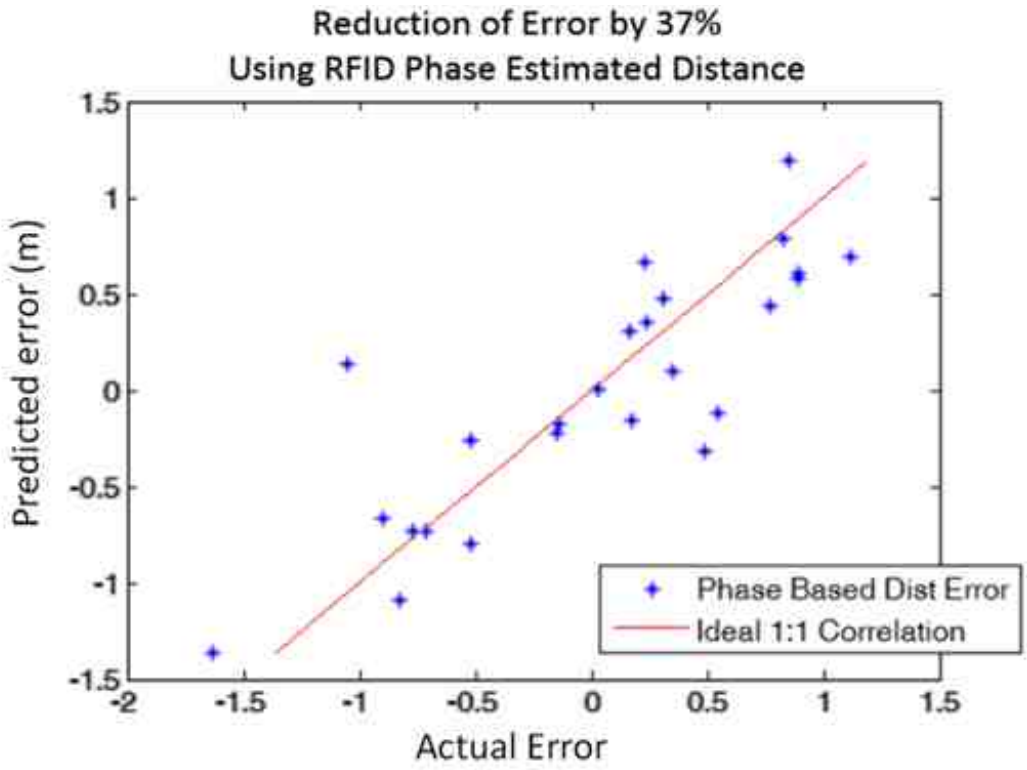


Figure 6-10: Neural network prediction of Phase-Distance estimation error for a range of environments and distances.

6.7.3 Reducing Distance Error in New Environment

To validate the theoretical prediction that the neural network would be able to separate the direct signal measurement from the multipath error, the trained neural network was then tested with a data set from a completely new and significantly different environment. This environment was a room with different domestic furniture, room proportions and layout. If the neural network had merely learned the patterns of the environments it had seen, the resulting distance prediction would show no improvement from the standard RSSI vs. distance best fit. However, if the neural network had succeeded in separating the multipath error from the direct measurement then that information should result in improved distance estimation for the new environment.

The results of this experiment, shown in table 6-1, support the hypothesis that the neural network could, at least to some extent, separate the direct path signal, P_{RD}, from the multipath error, Φ, for the new environment, using information contained in a frequency sweep. With these variables separated, the neural network compensated estimate was more accurate because the quantified multipath error was removed from the distance estimate. The table shows the uncompensated error in both cases. The average percent reduction in error of 11.8% was found for distance estimation in the new environment using a previously trained neural network. Additionally, these results show that the variation in RSSI with respect to frequency could be generalized, otherwise no improvement would have been found. If the neural network had not been able to quantify the multipath error there would have been a negligible decrease or an increase in the average distance error.

Table 6-1: Comparison of control vs. completely new environment for location error.

	Original Distance Error Φ	Neural Network Compensated Error Φ
Mean (m)	0.49	0.35
Standard Deviation (m)	0.65	0.53
Percent Reduction		11.8 %

This experiment represents the results of nearly the minimum number of training sets required for training a neural network, approximately 500 samples from six environments. Further reduction in error would be anticipated with a larger training dataset and additional variation in environments from more significantly varied multipath.

6.8 Using RSSI Signature with Bluetooth Low Energy

It was theorized that the same factors which led to the correlation between the RSSI signature, multipath, and distance error in RFID technologies, might also have a similar effect using Bluetooth. To test this theory, the Bluetooth Low Energy (BLE) Texas Instruments CC2541 Mini Development Kit [15]was used to collect 380 RSSI signatures.

Bluetooth operates on the 2404 – 2480 MHz band and has 40 distinct hopping channels. Channels 37, 38, and 39 at 2402, 2426, and 2480 MHz respectively are used for advertisement, leaving the rest of the channels for data exchange [54]. Therefore in collecting a Bluetooth RSSI signature, 37 different hopping frequencies can be used.

Due to space limitations, all of the data was collected in a 10 meter semi-anechoic chamber. The signatures were recorded at a variety of separation distances ranging from 1 to 9 meters. The resulting RSSI vs. distance curve was used for the initial distance calculation, given by Equations 6-18 and 6-19.

Best Fit of RSSI vs. Distance for BLE

$$RSSI = -5.49\ln(d) - 53.56 \qquad \text{Equation 6-18}$$

Or

$$d = e^{\left(\frac{RSSI+53.56}{-5.49}\right)} \qquad \text{Equation 6-19}$$

Where:

RSSI = measured BLE RSSI value (dBm)

d = reader to tag separation distance (m)

When the neural network was trained and tested using the BLE RSSI signatures it was found that, similar to RFID, the neural network was able to predict the distance estimation error. Figure 6-11 is a graph of the neural network predicted BLE distance error vs. the actual distance error.

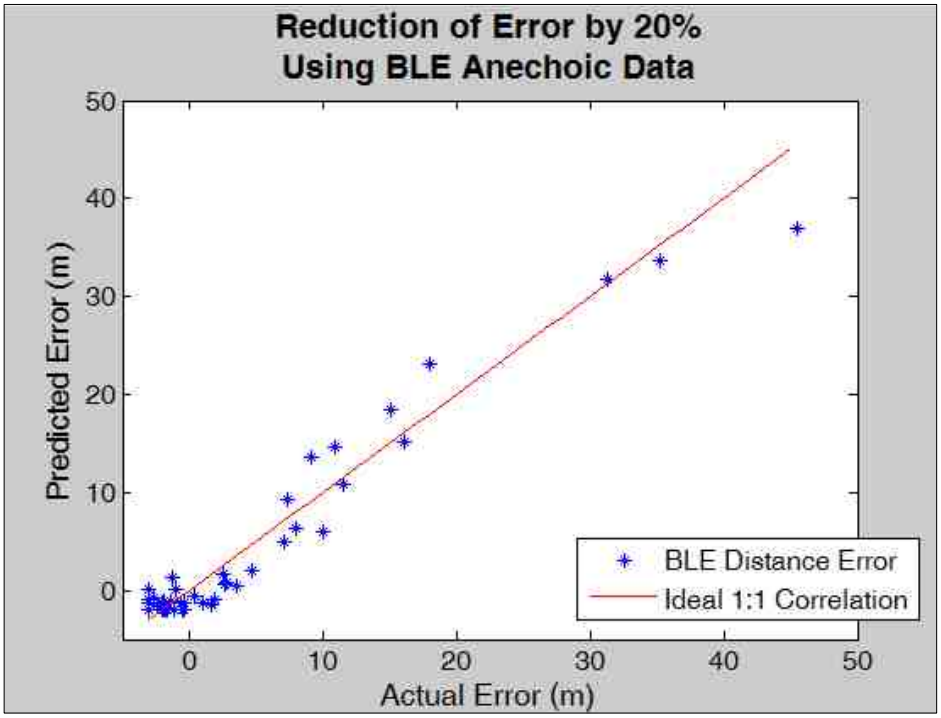


Figure 6-11: Neural network predictions of BLE distance estimation error vs. actual error

For BLE distance estimations, the neural network was able to reduce error by varying amounts, and up to 40%. It is quite possible the reason for the smaller reduction in error was simply due to the location which the data was collected. All of the BLE data was collected in a semi-anechoic chamber, which would have significantly less multipath than in the "real world" environments in which the RFID data was collected. It would be

expected that the BLE data would have less initial error due to multipath, and therefore reducing the remaining error from multipath would be difficult.

6.9 Conclusion

A variable separation model was presented that demonstrated that a frequency-based RSSI signature might be capable of separating the direct signal measurement from multipath error. The model used variations with respect to frequency to predict multipath error. While a simple correlation of the shape of the RSSI versus frequency signature directly to multipath present in the environment was not feasible nor was it a closed for solution, it was found that machine learning via a neural network has the pattern recognition capabilities necessary to analyze the RSSI signature, and extract this error.

Through experimentation, the relationship between the RSSI signature and multipath of the surrounding environment was investigated. Given identical multipath situations, the RSSI signature is repeatable, as expected. However, even small changes in multipath result in noticeable changes in the RSSI signature, as theoretically predicted.

A series of experiments were performed to demonstrate a generalizable method to improve distance measurements using the RSSI signature. After training the neutral network, test data reserved to analyze the predictive capabilities showed the neural network was able to not only extract the magnitude of the multipath error, but also whether the multipath generated positive or negative error.

While the exact reduction in error varied with each neural network trained and the data with which it was trained, the output from the neural network could potentially be

used to reduce the distance estimation error by upwards of 50%. Moreover, it was found that the RSSI signature could be used to reduce error in both RSSI and phase distance estimations methods for RFID and RSSI distance estimates for BLE. A larger reduction in error was achieved when applied to the RSSI distance estimations than phase, likely due to the fact that phase measurements are more robust than RSSI with respect to multipath.

Finally, to validate this generalizable process, data was collected from an entirely new environment for RFID. When a previously trained neural networks were used to predict the distance error in the new environment, the result was an 11.8% reduction in distance error.

The RSSI signature is shown in this chapter to be a unique representation of the multipath of the surrounding environment. In this work, it was demonstrated that changes in the environment yield transformations of the RSSI signature. Secondly, by using the RSSI signature with machine learning, this method was able to significantly reduce distance estimation errors from multipath.

6.10 Resources

The MATLAB code used for each of these methods is given in Appendix C, section 7. The code to train the neural network is given in section C.7.1. The code to generate an RSSI signature from RFID data is given in section C.7.2, to leverage the RSSI Signature to improve phase in C.7.3 and to create an RSSI Signature for a BLE network in C.7.4.

CHAPTER 7. REPRESENTING THE RSSI SIGNATURE

7.1 Overview

In the previous chapter the relationship between an RSSI vs. frequency pattern (RSSI signature) and the multipath of the surrounding environment were explored. The RSSI signature could be used by a trained neural network to predict distance estimation error in both magnitude and direction. This allows the RSSI signature to compensate for multipath error and ultimately reduce error in real world distance estimations by approximately 50% in trained environments. The key issue being that incomplete signatures cannot be input into a neural network, and therefore cannot be used by the methods described in the previous chapter. In this chapter, the practical aspects of implementing an RSSI signature in a manner that addresses both complete and incomplete RSSI signatures for distance error reduction are discussed. The innovation presented in this chapter does not modify the neural network, but rather explores the ways in which the signature can be represented in a feature set and delivered to the neural network. A thorough statistical analysis and discussion of the various methods of representing the RSSI signature is also presented. Through this research it was discovered that a close fit approximation of the RSSI signature yields the best results for error reduction. The MATLAB code used in this chapter can be found in Appendix C, section C.8.

7.2 Introduction

The accuracy of received signal strength indication (RSSI) based trilateration systems for the localization of indoor wireless devices is often poor due to the large error in RSSI distance estimations. While errors may arise from a number of different sources,

RSSI is known to be prone to multipath error from reflection in the surrounding environment. In Chapter 6, the way in which multipath impacts the shape of an RSSI vs. frequency plot, named an RSSI signature, was investigated. That shape was analyzed using a neural network to predict error in initial range estimations for both RFID and Bluetooth systems. Because standard neural networks require a constant number of input features [52] [53], this method of using the RSSI signature to reduce distance estimation error was limited to situations where complete signatures had been collected. In order to create a robust method for RSSI signature use, it's necessary to develop methods using incomplete RSSI signatures to reduce error. The method proposed in this chapter, of training neural networks with incomplete information, is to represent the signature in a compressed feature space, and subsequently use the compressed feature space to train the neural network.

7.3 Background

In the previous chapter, a significant reduction in error in distance measurements was achieved by decoupling the multipath error from the base distance signal using the information from a frequency vs. RSSI "signature". This improvement in distance measurement accuracy ultimately reduces the localization error when determined by multiple distance measurements from multiple readers as will be discussed in Chapter 11. However, it is impractical for a distance measurement to require measuring RSSI at all hop channel frequencies prior to determining a distance result for two reasons. First, requiring all RSSI measurements at all hop frequencies requires considerable processing time, which is not compatible with the need for rapid distance acquisition in most tracking applications. Second, the regulatory requirements of hop patterns prevent users from requesting the order

of frequencies required to make such a determination, or verifying that all frequencies are measured making the process more time consuming and the signature more likely to be incomplete.

Reducing error allows for significant improvements into accuracy, utility and expanded applications of localization, and therefore motivates the investigation in this chapter into methods that circumnavigate complete data constraints, and thus permit effective neural network training in realistic data-limited conditions. The goal of this work is to determine if a representation of the signature, constructed from incomplete data, would be sufficient to train a neural network and still achieve the desired error reduction.

7.3.1 Received Signal Strength Indication

There are several different methods of indoor wireless localization such as: fingerprinting, near neighbor, triangulation, and trilateration, as described in greater detail in Chapter 2. The method selected for this research is trilateration, which relies on distance estimation and primarily RSSI measurements. RSSI is the power level of the returned signal from the queried devices as seen by the reader. Due to the laws of wave propagation the power level of the signal decreases (by a function of the inverse square) the further the signal travels as discussed in Chapter 3. Therefore, RSSI can be used to estimate separation distance from the reader to the unknown device.

Unlike measurement techniques such as Time-of-Arrival or Angle-of-Arrival, RSSI is available in most wireless devices. Thus, RSSI is commonly used for localization despite its susceptibility to multipath error by assuming it as random noise. Some methods which use RSSI attempt to compensate for its inherent vulnerability to multipath error by

assuming it as a random noise. In methods such as fingerprinting or near neighbor, the RSSI of the unknown device is compared with either a map of known RSSI values, or to other known devices within the area. This allows the multipath error to be mitigated by comparing the RSSI values to other sources similarly impacted by multipath. Unfortunately, fingerprinting requires frequent updates of the RSSI map, and near neighbor requires a large number of known devices in the field. These requirements are prohibitive in real world applications which may have variable numbers of devices, and exist in dynamic or unmapped environments. For this reason, RSSI distance estimation based trilateration was selected in Chapter 2, and ways in which to make it more robust with respect to multipath are therefore further explored in this chapter.

7.3.2 RSSI Signature

In Chapter 6, a unique investigation of the pattern of RSSI values vs. frequency, which was called an RSSI signature is described. In the previous chapter the characteristics of the RSSI signature from an RFID system are investigated, such as its repeatability in identical conditions, and the influence of the surrounding environment on its particular shape. It was hypothesized that the variable paths of reflection and refraction of waves traveling to and from the reader, otherwise known as multipath, were causing the distinct patterns. Figure 7-1 is an example of an RSSI signature; demonstrating both how RSSI changes with respect to frequency, and how under identical environmental conditions the signature is repeatable. The experimental set-up used to create this figure is given in Appendix A, section A.4.9.

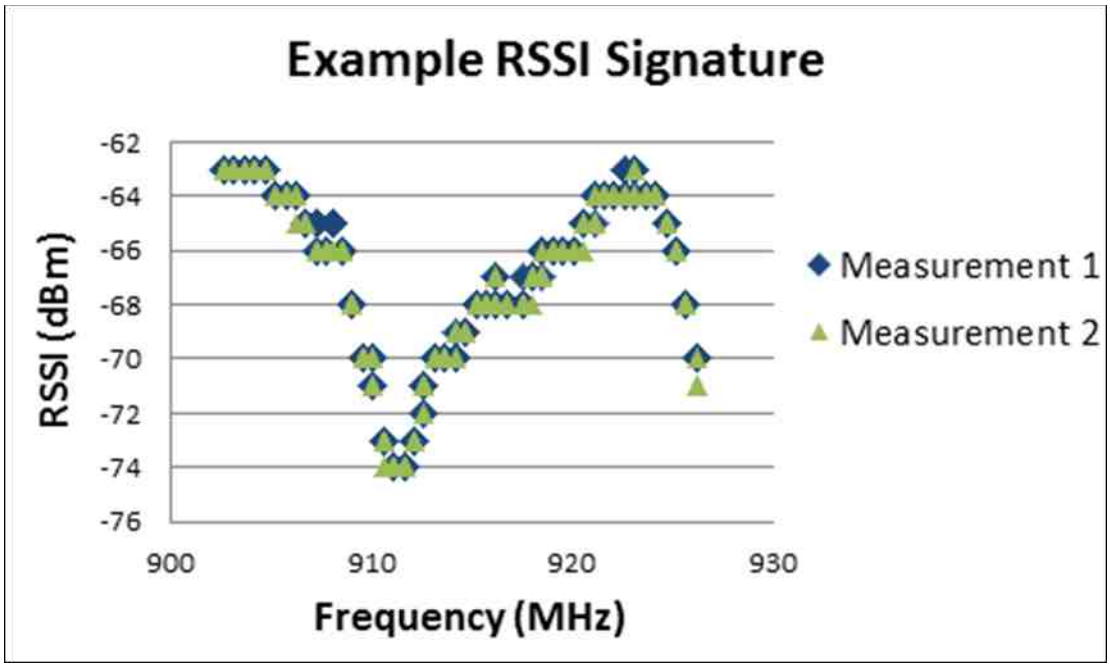


Figure 7-1: Example RSSI signature (RSSI vs. Frequency)

The RSSI in Figure 7-1, is measured at 50 distinct frequencies for RFID. These frequencies are based upon the 50 hop frequencies used by the ThingMagic M6e radio in compliance with the Federal Communication Commission rules on radios operating within the 902-928 MHz frequency band, and their requirement to jump between a minimum of 25 to 50 different carrier frequencies.

7.3.3 Using RSSI Signature to Reduce Error

In Chapter 6 the potential of the RSSI Signature method to reduce error range estimation in both RFID and Bluetooth systems was demonstrated. This was accomplished by first collecting over 500 signatures from a variety of different environments and separation distances. For each signature the separation distance was measured manually. Then the empirical RSSI vs. distance curve, as shown in Figure 5-1, was used to calculate an initial separation distance.

The estimated separation distance was compared to the absolute distance (the error of which is discussed in Chapter 12) to find the distance error. A neural network was then trained using the RSSI signature data as the input, each of the RSSI values of the signature were used as features, and each signature as an observation. For RFID systems this meant 50 features, and 37 for Bluetooth. In this training process the error in the initial distance estimation was used as the prediction; thus the neural network was trained to use the signature in order to predict the error in the initial distance estimation.

Through this process it was found that not only could the signature be use to predict the magnitude of distance error, but also determine if the initial estimation was an over or under prediction. Thus, a trained neural network could use the RSSI signature to compensate for, and ultimately reduce, errors in distance estimations by approximately 50%. The signature was found to be effective for both RFID and Bluetooth systems, as well as RSSI and phase angle based distance estimations, although the percent reduction of error varied among the different methods.

To validate that the result was real and not a result of the bleed of information during the training phase, a new set of data collected in a completely new environment was used as a validation set. This set of data also saw a reduction in error of about 12%.

7.3.4 Incomplete RSSI Signatures

For the data sets used in Chapter 6, only complete signatures were used as inputs for both training and testing the neural networks. This is due to the fact that, as discussed in the previous chapter, neural networks are unable to handle a varying number of feature inputs. Neural Networks function by mapping a set of input features into a structure of

nodes and hidden layers [55]. This machine learning algorithm finds patterns in the features by multiplying them by a series of nodes called hidden layers. It adjusts these node values using an optimization algorithm until the nodes become tuned to patterns of features which correlate with the desired classification, resulting in a statistically optimized prediction based on a learned model. Unfortunately, in real world applications there are a number of different reasons why it would be impractical or impossible to collect a complete signature. In situations of extreme multipath, the reader could be unable to read at certain frequencies due to destructive interference, often from its own signal. When a device is nearly out of range, there may be frequencies at which the reply signal is near the noise floor, and the reader is incapable of measuring the response. Alternatively, when using pseudorandom frequency hopping in RFID systems, it takes an average of 15 seconds to measure at every one of the 50 hop frequencies (as discussed in Chapter 5), which is time prohibitive in many distance estimation or localization applications. Therefore, it is of significant interest to find methods for improving accuracy from RSSI signatures when complete signatures are not available.

One possible approach is to use machine learning algorithms capable of incorporating variable numbers of features, such as those resulting from incomplete datasets. There is currently a major field of investigation and development surrounding machine learning algorithms which can use incomplete datasets [55]. However, due to their complexity and variability, it was determined that a more practical approach would be to represent the signal intensity at variable frequencies as a curve, which could be found by sampling at a smaller number of frequencies and then fit to the measured data. This

reconstructed intensity/frequency relationship could then be used as an input into a neural network to estimate and compensate for error.

It was hypothesized that a mathematical curve fit the RSSI signature might be able to preserve the multipath information, and provide a consistent number inputs for the neural network, while maintaining approximately the same level of error reduction. While there may be a nearly unlimited number of ways to represent the signature, the purpose of this investigation is to demonstrate that there are effective ways to represent signatures (both complete and incomplete), which do not significantly impact the neural network's ability to predict distance estimation error.

In this chapter the error reduction method described in Chapter 6 is expanded to investigate ways in which RSSI signatures can be represented using a variable number of measured inputs.

7.4 Experimentation

The collected RSSI signatures used for the analysis in this paper are the same datasets used in Chapter 6, but without excluding incomplete sets. For the collected data, the following equipment was used: a ThingMagic M6e radio, a circular polarized reader antenna, and an Alien Squiglette passive UHF RFID tag vertically polarized. The reader antenna and tag are both at a height of 1m above the ground plane. Given that the RSSI Signature method was found to be applicable to both RFID and Bluetooth in the previous chapter, here just RFID data will be used with the assumption that the findings on incomplete datasets will translate to Bluetooth technology as well. The RSSI signatures were collected in a variety of environments including: fully-anechoic, semi-anechoic,

laboratory, and warehouse-like. The signatures collected, therefore, represent a wide range of multipath environments. Additionally, for each environment the RSSI signatures were recorded at a range of separation distances. A total of 550 signatures were collected, and in contrast to the method described in Chapter 6, incomplete signatures were included.

7.5 Representing the RSSI Signature

For each of the following methods to represent the signature, neural networks were trained and tested using 5, 25, and 50 features. The purpose of training and testing the different neural networks with a varying number of features is to investigate not only preferred RSSI signature approximations, but also the advantages and disadvantages of using more or less input features. The RFID ThingMagic M6e systems yields 50 distinct hop frequencies, and thus providing the neural network with the same 50 features would be preserving the entirety of the collected RSSI signature information. In this way, using more points would be expected to produce greater reduction in error because it would mean additional information. However, given that there are only a set number of training observations (because one cannot collect an infinite amount of data), reducing the number of input features would increase the feature-to-observation radio, essentially leveraging the number training observations. Reducing the number of features therefore could result in greater error reduction by reducing overfitting. Thus in this chapter some of the analysis will cover the performance of a neural network as a function of the number of input features, to determine the optimal feature set when using the RSSI signature for error reduction purposes.

7.5.1 Sine Wave Curve Fit

One method used to represent the RSSI signature is a custom systematic sine wave curve fit. This line form can be represented using the following general equation:

General Form of Sine Wave

$$RSSI = A\ sin(Bf - C) + D$$

Equation 7-1

Where:

RSSI = vertical axis (dBm)

A = amplitude from center (dBm)

f = horizontal axis (Hz)

B = angular frequency (radians/Hz)

C = phase shift (radians)

D = vertical offset (dBm)

When optimally fitting a sine wave to a signature, the initial step is to determine approximate values for each of the unknowns (A, B, C, and D). Initially, the MATLAB Curve Fitting Toolbox was used to find a best fit sine representation of the RSSI signature. Unfortunately, these results were not ideal, and thus a sine curve fitting algorithm was specifically developed for this purpose. The MATLAB code to do the sine wave fit is given in Appendix C, section C.8.2. The values were optimized using Gradient Descent;

a non-linear algorithm which uses the derivative of a cost function to iterate towards a final result. The nature of nonlinear algorithms is such that they find local but not global minimums, therefore an accurate initial estimate of the constants is crucial.

$$A = \frac{Y_{max} - Y_{min}}{2}$$

Equation 7-2

An initial approximation of the amplitude (A) is found by taking the highest RSSI value (Y_{max}) minus the lowest RSSI value (Y_{min}) and dividing by two (as given by Equation 7-2). The angular frequency (B) is found by observing the number of changes in direction of the RSSI signature. The length of a period is approximated by the distance between the first peak and value of the signature. Then the number of periods within the signature's range is found by looking at the number of changes in direction (divided by two). Finally, B is found by Equation 7-3.

$$B = \frac{2\pi * Num\ Periods}{Length\ Periods}$$

Equation 7-3

Where:

Num Periods = Estimated Number of Periods (#)

Length Periods = Estimated Period Length (Hz)

The phase shift (C) is found by locating the center point between the high and low points within one period, then locating the middle, and using this as the offset. The vertical

offset (D) is simply the middle vertical height between the highest and lowest points. Figure 7-2 shows an initial fit of RSSI signature points using these initial measurements.

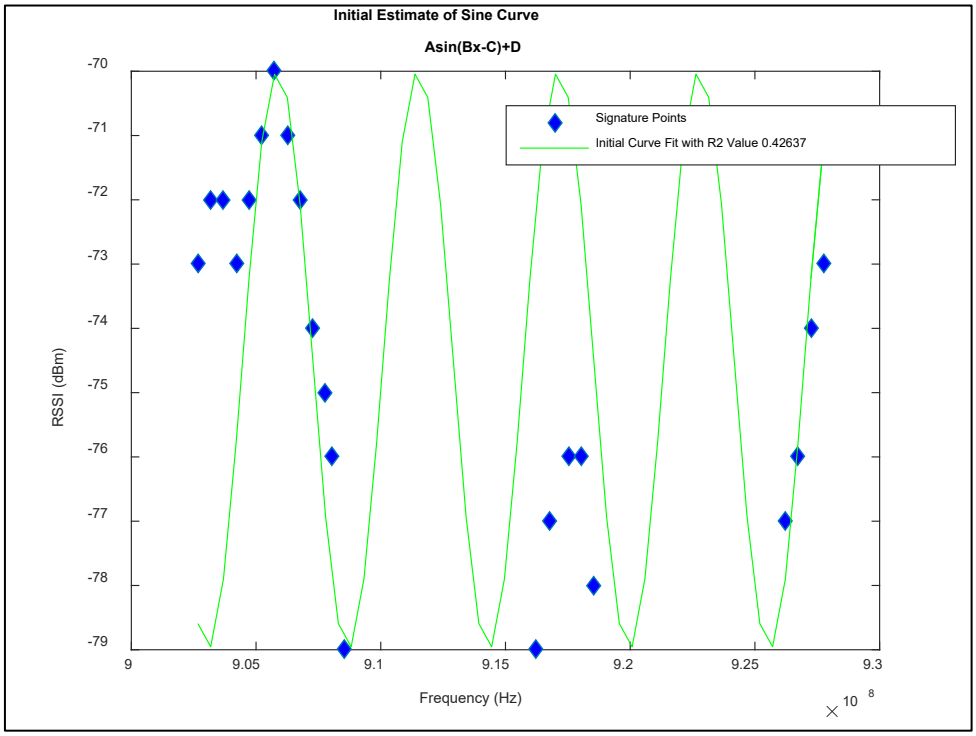


Figure 7-2: Example initial sine wave curve fit of an incomplete RSSI signature.

The cost function used to optimize the constants was an R^2 function, which finds the distance between the true value and the value predicted by the function. Gradient descent worked well to find the optimal values of A, C, and D, but B was more complex. Due to the irregular contour of the objective function, several initial values were tested to find the optimal (or near optimal) curve fit of the RSSI signature. Figure 7-3 shows the complexity of the contour of the objective function, and Figure 7-4 shows the final solution of an example sine wave curve fit of an RSSI signature.

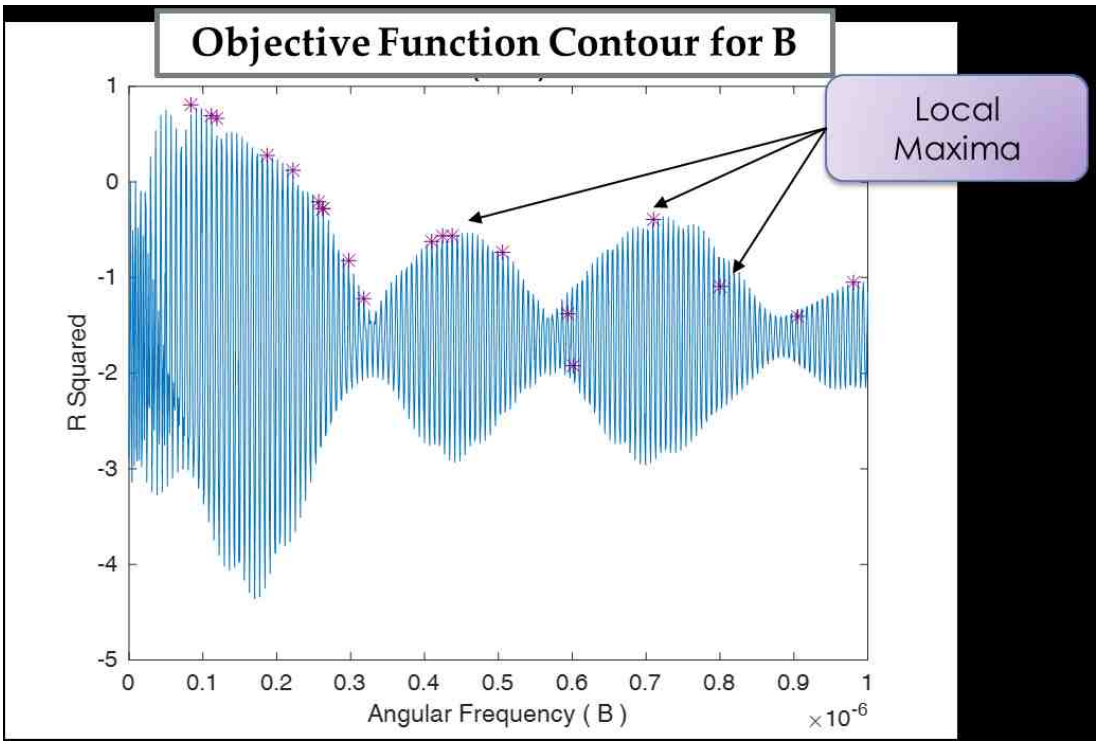


Figure 7-3: The complex objective solution space for the variable B.

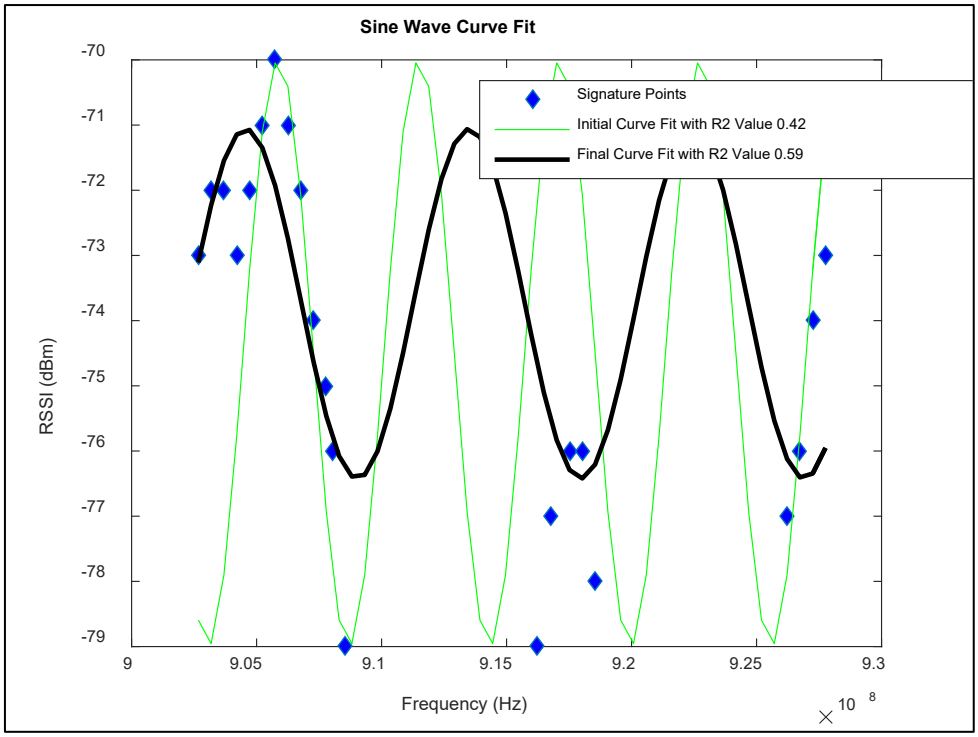


Figure 7-4: Final sine wave curve fit of an example RSSI signature.

Whether using 5, 25, or 50 features for the neural network, these points were selected by using the RSSI value taken from the sine wave curve fit at evenly spaced increments, rather than the signature itself. This approach also allowed the sine wave curve fit to represent the RSSI signature as the constants (A, B, C, and D) of the best fit sine wave. Thus, another variation of this method was to use these constants as the inputs to the neural network, reducing the number of input features from 50 to just 4.

7.5.2 Polynomial Curve Fit

A polynomial fit of the RSSI signature was investigated due to the irregularity of the signature, and the ability of polynomials to fit irregular waveforms. Several different order polynomials were used from 2nd order to 6th order polynomials of the form:

General Form of Polynomial Curve

$$RSSI = A_0 + A_1 f + A_2 f^2 + \cdots + A_n f^n$$

Equation 7-4

Where:

RSSI = vertical axis (dBm)

A_i = polynomial coefficients (dBm/Hz^i)

f = horizontal axis (Hz)

The polynomial curve fit of the RSSI signature was optimized using the "polyfit" function in MATLAB. The code used to generate this signature approximation can be

found in Appendix C, section C.8.3. In this function, MATLAB calculates a Vandermonde matrix, which is then used to calculate best fit polynomial coefficients, as seen in Figure 7-5.

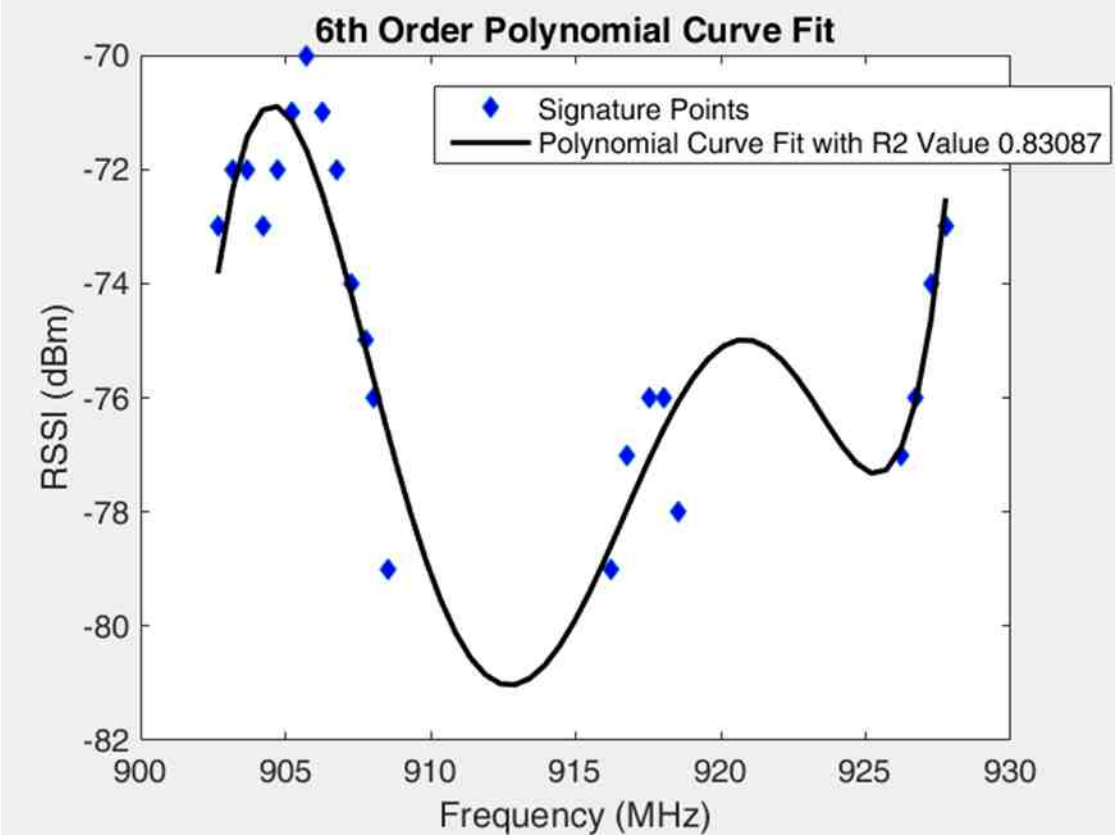


Figure 7-5: Polynomial curve fit of 6th order of an example RSSI signature.

When selecting points for the input features of the neural network, for each deviation, 5, 25, or 50 evenly spaced points were selected along the polynomial curve fit and used as neural network inputs. This method also allowed the coefficients of the best fit polynomial (A_0, A_1, … A_n) to represent the RSSI signature as input features for the neural network.

7.5.3 Cubic Spline Interpolation

A cubic spline is a piecewise equation fit of a set of points using 3rd order polynomial equations. For this paper the "spline" function from MATLAB was used to fit the RSSI signature. The code to create the cubic spine can be found in Appendix C, section C.8.4. A cubic spine is a spline constructed through a piece-wise third order polynomial which passes through some number of control points that define it. The definition of a cubic spline is given as follows: [56]

Let $K=\{x_o, \dots, x_m\}$ be a set of given knots with $a=x_o<x_1<\dots<x_m=b$

A function $s \epsilon C^2[a,b]$ is called a cubic spline on [a,b], if s is a cubic polynomial s_i in each interval $[x_i, x_{i+1}]$.

It is common to set the second derivative of each polynomial equal to zero at its endpoints to provide boundary conditions to solve the set of equations created by the spline. This is done automatically in MATLAB in the spline function.

For each situation of 5, 25, or 50 features, MATLAB was used to calculate the polynomial. A vector of the desired number of evenly spaced frequency points are given to the "spline" function, which in turn provides the corresponding RSSI nodes for that particular cubic spline interpolation. Figure 7-6 shows a cubic spline curve fit of an example RSSI signature using 5, 25 and 50 points.

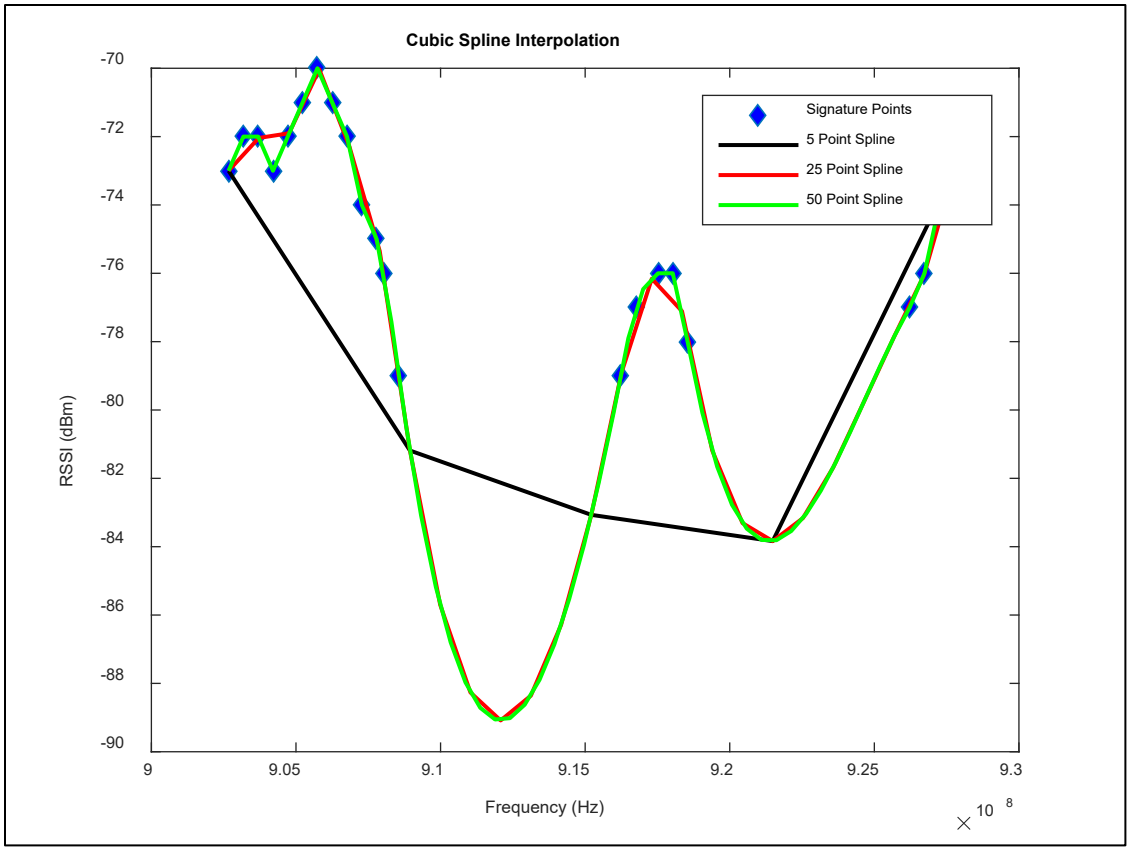


Figure 7-6: Cubic spline curve fit using 5, 25, and 50 points.

7.5.4 Rolling Mean and Linear Interpolation

The RSSI values measured using the ThingMagic M6e radio are solely integer values, which leads to discretization of the data, and a possible loss of precision. It is likely that the true RSSI values in the signature would yield a smoother curve than is depicted by the measured data. Therefore, another method to represent the RSSI signature is to smooth the data by using a rolling mean. In this analysis, each point in the rolling mean was calculated as the average of the original neighboring five points. This was accomplished by using the "smooth" function in MATLAB, as can be seen in Figure 7-7.

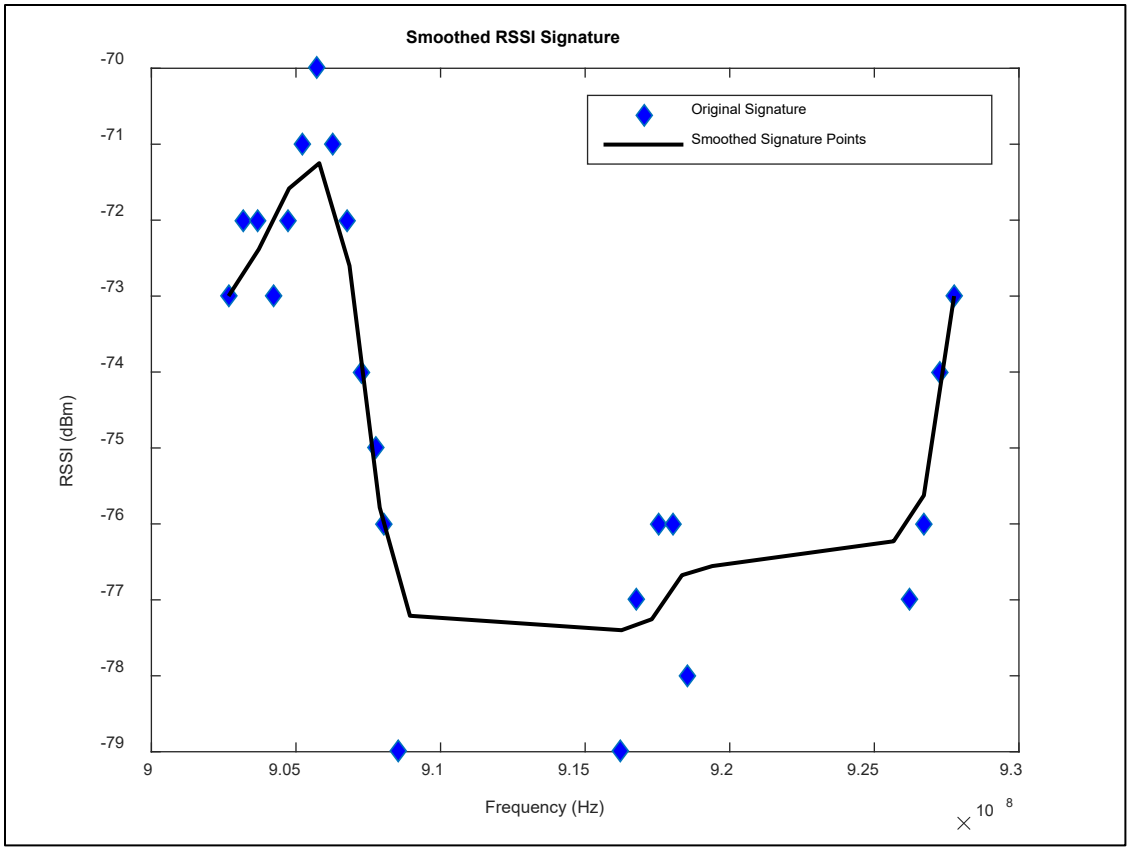


Figure 7-7: Running average smoothing of and example RSSI signature.

After the data was smoothed to make a less discretized signature, any missing points were replaced using linear interpolation of the surrounding two points. Missing points from at the beginning or end of a signature were replaced with the nearest measured value. When selecting 5, 25 or 50 input features, evenly spaced RSSI values were selected. If there were not an RSSI value at the desired location, then this was also calculated using the linear interpolation of the surrounding points.

7.5.5 Constants for Incomplete Data Sets

The final method for representing the RSSI signature and handling incomplete signatures is to simply use a constant value in place of missing data points. Often when a

tag or device is nearly out of range, or multipath creates destructive interference, points are lost due to the return signal being too close to the noise floor. Thus filling in all lost points with a constant also near the noise floor, such as -80 dBm, could preserve the multipath information, while maintaining a constant number of input for the neural network.

For the choice of neural network features, the desired number of evenly spaced RSSI values would be used. If an RSSI value did not exist at the desired frequency, then that point would be assumed as -80dBm.

7.6 Testing:

The purpose of this investigation is to characterize the utility of several possible methods of representing the RSSI signature, typically by fitting it to a y=f(x) function, where x is the frequency and y is the RSSI signature at that frequency. This allows for data at missing or unobtainable frequencies to be predicted using a smaller set of obtainable data, and satisfies the neural network's need for a constant number of input features, without significantly reducing the neural network's ability to reduce error in distance estimation. Thus, each of the methods discussed in the previous section were used to train and test several neural networks, and were evaluated based on neural network performance.

The 550 collected, complete and incomplete, RSSI signatures were first approximated by one of the previously discussed methods. The next step was to use 5, 25, or 50 input features from the signature approximation method as input features for the neural network. For the sine wave and polynomial models, the constants from their curve fits were used as another variation for representing the signature. The neural networks were trained using error in distance estimation from the RSSI vs. distance prediction as the

desired output. For this analysis, 80% of the data was used for training, 10% for validation, and 10% for testing. When comparing the different RSSI signature representation methods, only the test data sets were used for the analysis. Each neural network was trained with four hidden layers.

Neural networks typically use a nonlinear optimization method called gradient descent, thus local optimums rather than global optimums are found. Therefore, each trained neural network is likely to be different from the next, even if it is based upon the same training data. To compensate for this variability, for each of the methods to represent the RSSI signature 500 neural networks were trained and tested. With this larger sample set, along with an analysis of variance (ANOVA), it's possible to determine that is the optimal model despite the fluctuations in results.

7.7 Results

While the median reduction in error from the 500 trained and tested neural networks was often around 20% as shown in Figure 7-8, the neural network with the best performance would be selected for use in an application. However, given the inherent variability of the situation simply determining the best method based upon the single maximum reduction could be misleading. Therefore, both the maximum and median percent reduction in error are given in Table 7-1.

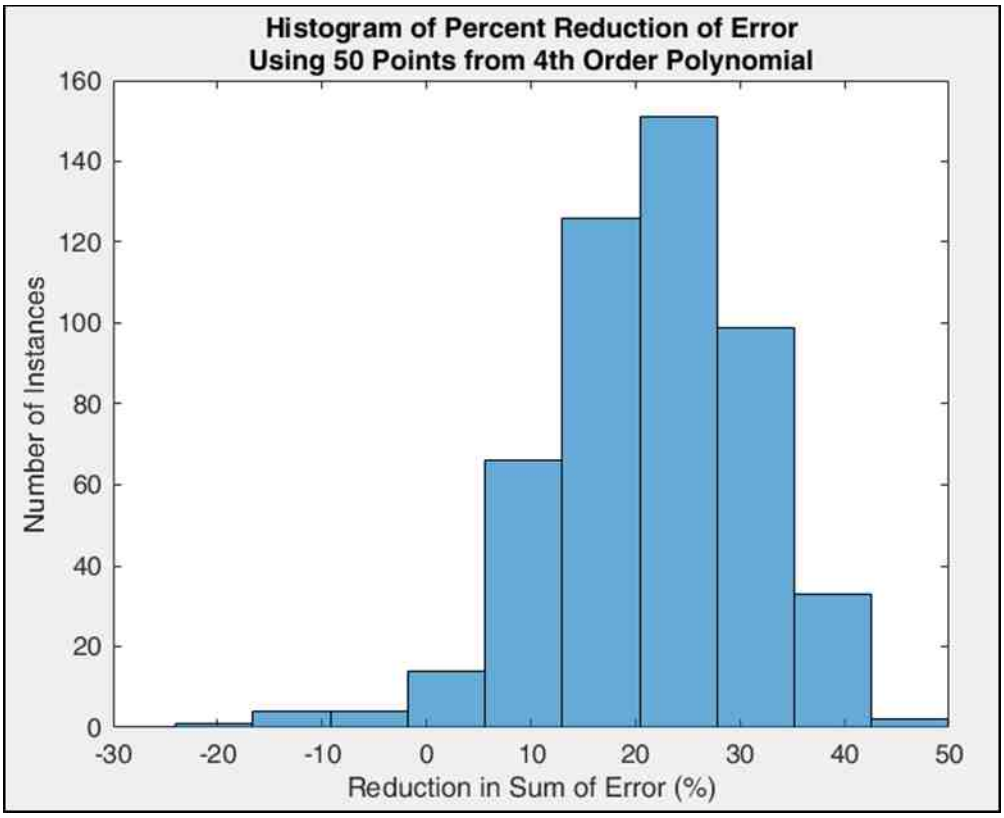


Figure 7-8: Distribution of percent reduction in sum error

Table 7-1: Summary of results for percent reduction of error

% Reduction in Error (Max | Med)				
	5 Points	25 Points	50 Points	Best Fit Consts
Sine Wave	47.1 | 21.8	49.6 | 23.1	44.2 | 22.7	30.0 | 10.6
2nd Order Poly	47.6 | 19.2	44.8 | 19.9	44.4 | 20.8	24.0 | 7.1
3rd Order Poly	47.5 | 20.5	46.2 | 20.0	45.7 | 19.8	20.13 | 1.6
4th Order Poly	46.4 | 21.1	46.9 | 21.1	49.6 | 22.4	17.7 | 1.3
5th Order Poly	45.0 | 22.0	47.0 | 23.9	51.3 | 23.9	16.6 | 1.6
6th Order Poly	44.5 | 21.5	51.3 | 25.1	55.3 | 24.1	17.0 | 2.6
Cubic Spline	46.6 | 21.6	44.6 | 22.2	48.2 | 22.4	
Rolling Mean	47.8 | 22.5	55.1 | 25.0	47.8 | 23.1	
-80 in Holes	48.0 | 22.0	53.2 | 22.4	48.8 | 21.7	

7.8 Analysis

Upon initial inspection of the results there are a few notable trends which appear. The most noticeable difference is how much greater error reduction is achieved when using

points along one of the curve fits, rather than using the coefficients or constants from the best fit equation. The minimal reduction in error from the use of best fit constants could potentially be due to the ambiguity of how these input features relate to the actual RSSI Signature. In this scenario, the neural network would be relied upon to essentially determine what equation these constants are in reference to. Therefore, using the more straightforward methods of simply providing the RSSI data points appears to be significantly more effective.

Another observation is which methods of representing the signature achieve a percent reduction of error greater than 50%; these are the 5^{th} and 6^{th} order polynomials, the rolling mean, and using -80 dBm in place of missing data points. Each of these methods are close fits of the original signature. The 5^{th} and 6^{th} order polynomials are relatively high order polynomial fits, such that they appear as a continuous and smoothed version of the original RSSI signature. It is quite likely that even higher order polynomial fits would also be successful representations of the RSSI signature. The rolling mean in many ways is similar to a high order polynomial fit. It appears as a smoothed version of the signature, and therefore is quite a close approximation. Use of the constant -80 dBm in the holes of incomplete datasets is maintaining the original signature as closely as possible. It seems that the main characteristic all of these methods have in common is that these approximations are very close to the original signature. As discussed in [3] and Chapter 6, it appears that the RSSI Signature contains information about the surrounding multipath which can be used to reduce error. Thus the best representations of the signature appear to be those that closely fit the original data.

To further explore the statistical significance of the various representation methods, an ANOVA analysis was employed. In Figure 7-9 the methods of: sine wave, 6^{th} order polynomial, cubic spline, rolling mean, and -80 dBm for holes in incomplete datasets, are compared. The corresponding ANOVA analysis yielded a p-value of 4.7e-8. Given a standard $\alpha = 0.05$, this analysis concludes that the deviation among methods is statistically significant.

Figure 7-10 is a comparison of the various order polynomial fits of the RSSI signature, and from comparison of the box and whisker plot it can be seen that the overall reduction in error increases with higher order polynomials. This shows the variation among the different order polynomial fits in predicting error is even more distinct than between the different types of curve fits, with a p-value of 1.5e-33. For this reason, the results of the 6^{th} order polynomial fit, which had the highest reduction in error, are used in the analysis shown in Figure 7-9 as part of the overall comparison of the methods of representing the signature.

For the analysis shown in Figures 7-9 and 7-10 the results from 5, 25, and 50 points were combined for each method, resulting in 1,500 points for comparison.

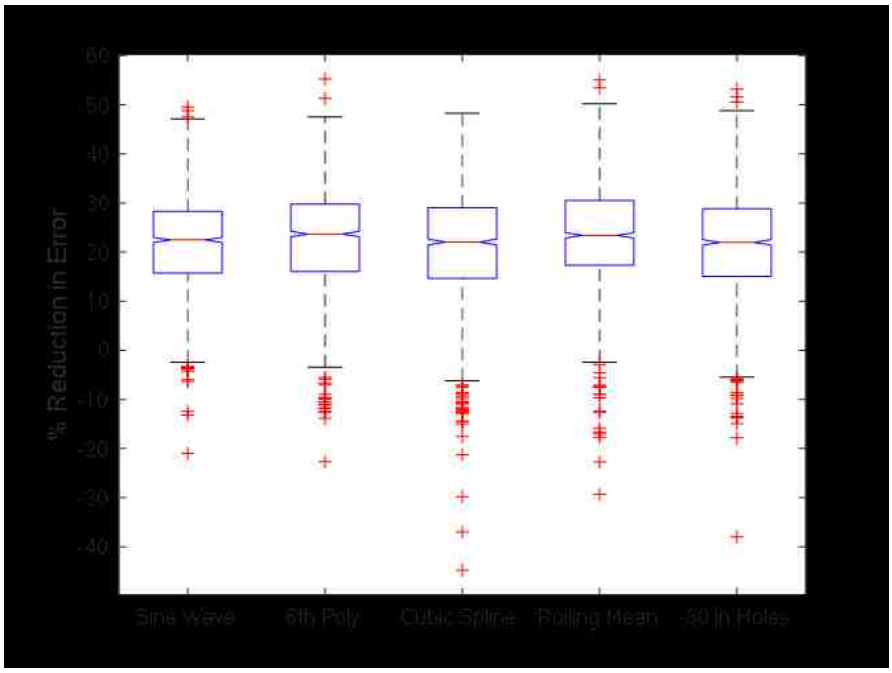


Figure 7-9: Boxplot of percent reduction in sum of error for sine wave, 6th order polynomial, cubic spline, rolling mean, and -80 dBm in place of missing data points

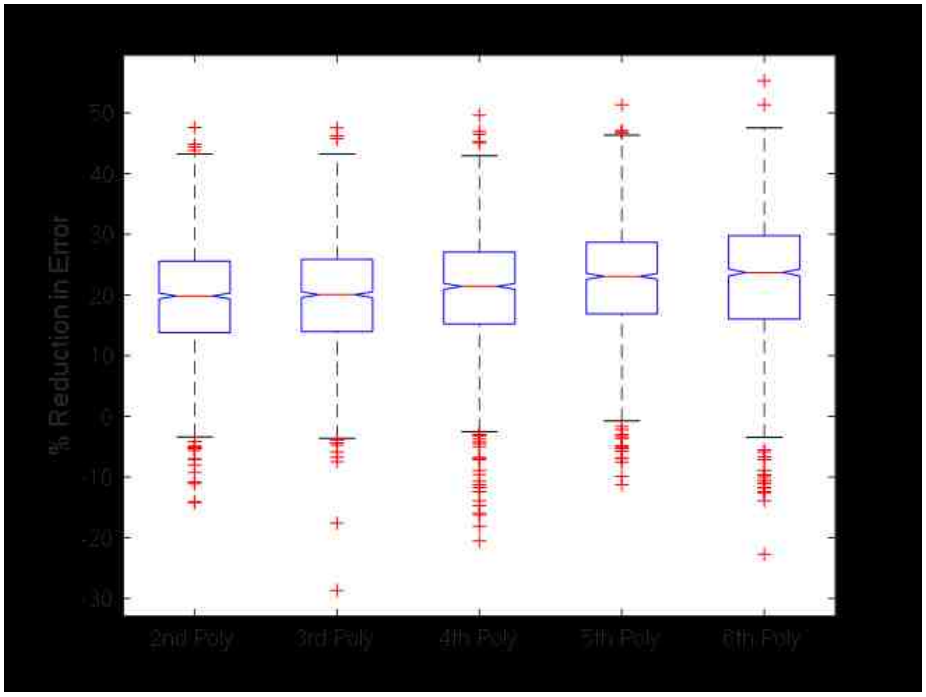


Figure 7-10: Boxplot of percent reduction in sum of error for 2nd, 3rd, 4th, 5th, and 6th order polynomial fits.

The statistical analysis of the methods in Figure 7-9 concludes that they are statistically significantly different; therefore, the next step it to determine which have the potential for better overall performance in real world situations. Each time a neural network is trained it will likely produce a different result, but in application only the one neural network, which gives the greatest reduction in error, will be used. Thus, comparing the methods of representing the signature based upon the lower quantiles, whiskers, or outliers is ineffectual. Rather, a contrast of the upper quantiles, or whiskers would provide more meaningful insight. While an argument could be made to compare the different representation approaches based upon the maximum reduction in error achieved, the analysis will not be made off of the statistical outliers. The location of the medians, upper quantiles, and upper whiskers from Figure 7-9 are listed in Table 7-2.

Table 7-2: Comparison of Box and Whisker Plot Results

% Reduction in Error			
	Upper Whisker	Upper Quantile	Median
Sine Wave	47.1	28.3	22.5
6th Order Poly	47.5	29.8	23.7
Cubic Spline	48.2	29.1	22.1
Rolling Mean	50.2	30.5	23.4
-80 in Holes	48.8	28.8	22.0

Based upon the upper whisker and upper quantile it appears that the rolling mean approach yields the largest percent reduction in error when neglecting statistical outliers. The 6th order polynomial fit, cubic spline, and -80 dBm for holes in datasets, are all quite comparable. Of those three, using -80 dBm in holes has the highest reduction in error of the upper whisker; and the 6th order polynomial has the highest reduction for the upper quantile and median. Use of a best fit sine wave appears to attain the least reduction in error, producing the lowest results for both the upper whisker and upper quantile. This is

likely due to the fact that a sine wave doesn't appear to be an effective or close approximation for the RSSI signature.

The next comparison of methods of representing the RSSI signature is that of using points from the actual signature or curve fit vs. using constants or coefficients of the best fit equation. For this analysis only five points from the sine wave and polynomial curve fits are used, such that there is a similar number of input features to the neural networks as the best fit equation constants. The ANOVA analysis produced statistically significant results, and the boxplot of the data shown in Figure 7-11 makes the comparison evident. Using constants from the best fit equation yields a median reduction in error only slightly greater than zero. It is clear that using points directly from the curve fit of the signature or from the signature itself obtains significantly larger reductions in error. It is quite possible that the neural network has difficulty determining what fundamental equation the constants are defining, and therefore the correlation between the features and the signature curve is ambiguous.

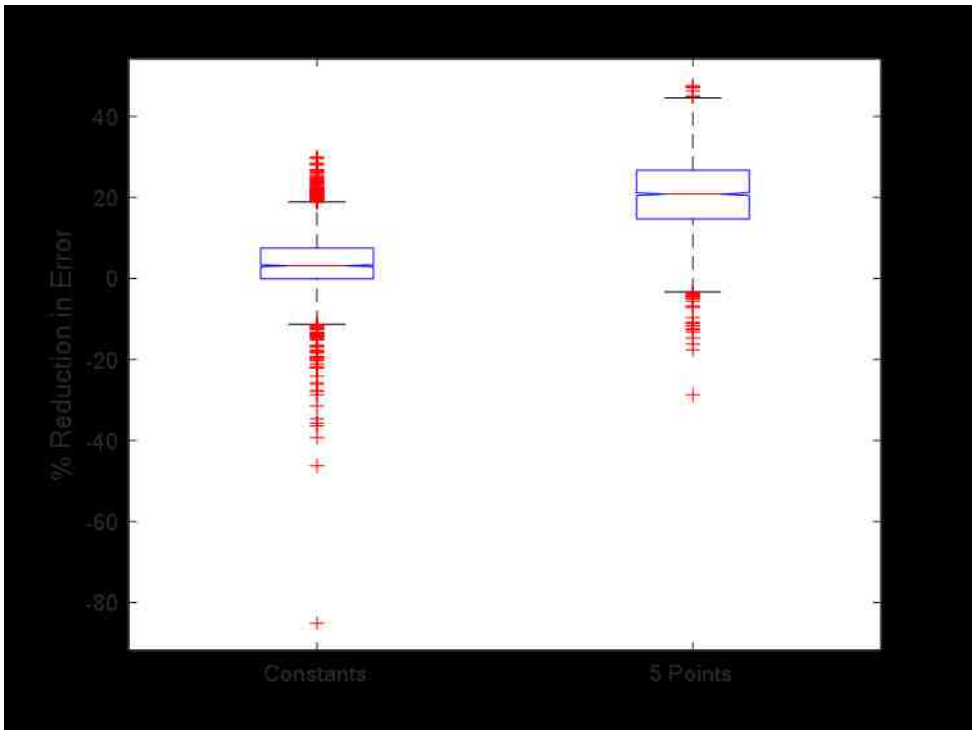


Figure 7-11: Boxplot showing differences in percent reduction in error from the use of constants from best-fit equations vs. using 5 RSSI values from the curve fits as input features to the neural network.

The comparison of the number of input features used for the neural network is also a critical facet of representing the signature. In order to significantly reduce distance estimation error, it would be ideal to provide the neural network with the greatest amount of information in the most succinct way possible. Thus, the comparison of the number of input features could shed some light on how this tradeoff impacts the reduction in error results.

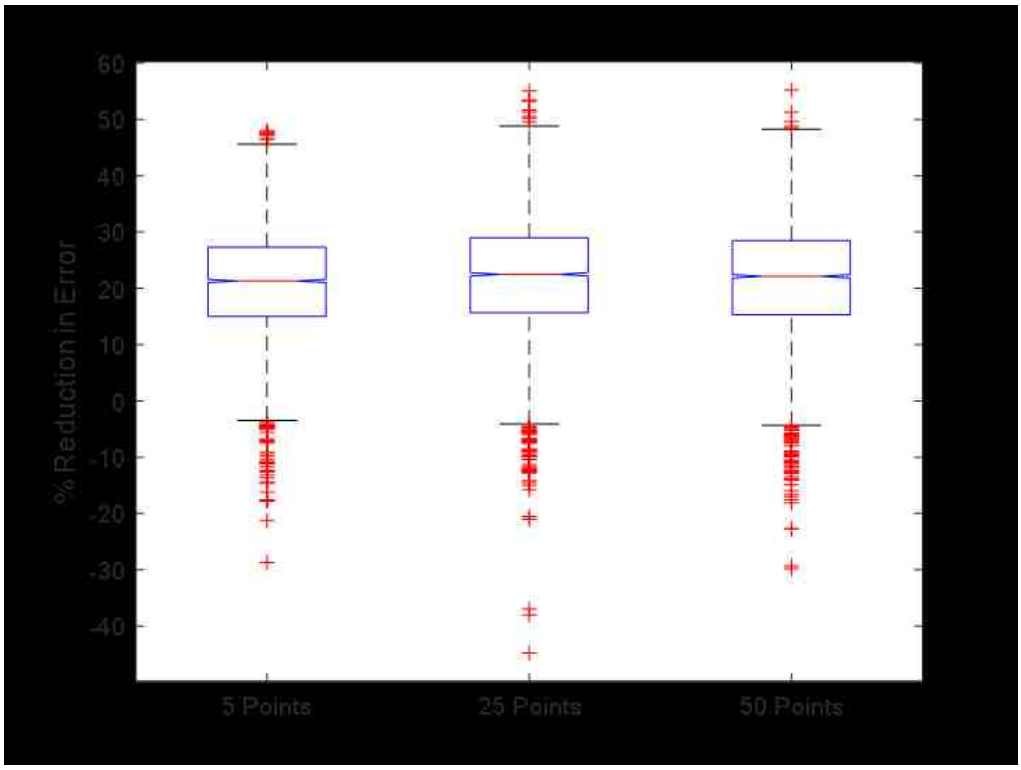


Figure 7-12: Boxplot of percent reduction in error from the use of 5, 25, and 50 points from the representation of the signature at input features to the neural network.

Table 7-3: Box and Whisker plot results from 5, 25 and 50 features

	% Reduction in Error		
	Upper Whisker	Upper Quantile	Median
5 Points	45.8	27.3	21.3
25 Points	48.9	29.0	22.5
50 Points	48.3	28.5	22.1

From the comparison of the results from using different number of features, as shown in Figure 7-12 and Table 7-3, use of 25 input features obtains the greatest reduction in error for each the upper whisker, upper quantile, and median. These results would indicate that there exists a balance between the number of input features and the information provided to the neural network. Simply providing every possible point may

not attain the best outcome even though theoretically it is supplying the neural network with the greatest amount of information. In contrast using only 5 input features appears to be an oversimplification, significantly reducing the multipath information and negatively impacting the percent reduction of error. It is likely that the ideal number of input features is not exactly 25, and further investigations should be made to identify the ideal number of input features for the representation of the RSSI signature.

7.9 Discussion

It is clear that there are a nearly infinite number of ways to represent the RSSI signature. The purpose of this particular investigation is to demonstrate the potential for representing the RSSI signature, such that incomplete signatures can be effectively utilized.

When examining the various methods proposed in this paper, it appears that a close fit approximation of the RSSI signatures achieves the highest reduction in distance error. The rolling mean technique achieved the highest reduction in error for both the upper quantile and upper whisker values when comparing the distribution error reduction. Other methods such as cubic spline, 6^{th} order polynomial fits, and the replacement of missing data with constant (-80 dBm) values also achieved large reductions in error. Each of these methods is a close approximation of the initial RSSI signatures, indicating the importance of preserving the multipath information contained within the original RSSI Signature. This is also demonstrated by the varying order polynomial fits. In this analysis, 2^{nd} through 6^{th} order polynomials were used to approximate the RSSI signature, and it was evident that increasing the order of the polynomial fit also had the effect of increasing the potential for reduction in error. While not fully investigated in the scope of this paper, it is likely that

increasing the order fit of the polynomial would continue to increase the percent reduction in error until this effect eventually asymptotically leveled out.

Another interesting observation is that the rolling mean attains a slightly higher reduction in error than using -80 dBm in the holes of the incomplete data sets, which is closer to the original signature. The rolling mean is a very close smoothed fit of the original RSSI Signature, but the -80 dBm method contains a larger number of points from the original signature. One hypothesis for this observation involves the measurement error inherent in the RFID radio due to discretization. Each RSSI value measured by the RFID radio is only reported to the nearest integer, while the true RSSI is unlikely to be such discretized values. Therefore, the true RSSI values are likely to lie somewhere along a close fitting smooth curve of the measured RSSI signature. If this was the case, then one would expect a close fit approximation of the RSSI signature to yield the best results. This is an interesting outcome because it indicates that using a smoothing or curve fitting method to represent the signature may produce the optimal results regardless of whether the RSSI Signature is complete or not.

A typical reduction in error for the RFID systems used in Chapter 6 ranged from 30% to 70%. In this investigation the maximum reduction in error achieved was 55%. One reason for this difference might lie in the difference of datasets used for the analysis. In Chapter 6 only complete datasets were used in the analysis, whereas this investigation used those datasets as well as a large number of incomplete datasets. By using the incomplete datasets there was less information available to the neural network, and thus it may not be possible to achieve the same percent reduction in error.

Another aspect tested was varying the number of input features to the neural network. It was anticipated that there would be a tradeoff between increasing the number of features to increase the amount of information, vs. decreasing the number of features to improve the observation to feature ratio. In the statistical analysis, it became evident that using 25 features yielded better results than either 5 or 50 input features. While 25 points may not be the ideal number of features, this is an indication that the optimal number lies somewhere between 5 and 50 features. The use of the best fit constants or coefficients as input features was also investigated as a method of representing a complex curve using very few points, and thus leveraging the observation to feature ratio. However, based upon these results, it appears that the neural network was unable to effectively decipher what equation the constants and coefficients were modeling. Thus the percent reduction in error from using the constants from the best fit equations had little or no effect.

While each of the proposed methods was compared based upon the median, upper quantile, and upper whisker values, in a real world application the neural network which consistently achieves the largest reduction in error would be employed. Thus based upon this investigation using a close-fit approximation of the RSSI signature, whether that be a rolling mean, high order polynomial, cubic spline, or another method, can effectively be used to represent complete and incomplete RSSI signatures such that a neural network can predict and reduce multipath error in a trained environment by approximately 50%.

CHAPTER 8. CHARACTERIZING THE ENVIRONMENT USING RSSI SIGNATURE

8.1 Introduction

Chapter 6 investigated the relationship between the RSSI signature and the multipath present in the surrounding environment. This chapter builds upon analysis conducted in Chapter 6 to investigate the utility of the RSSI signature in characterizing the surrounding environment. The idea is to understand how the signature can be used to identify attributes of an environment, and to what extent this is feasible. First in this investigation the goal was to distinguish between fully anechoic (*idealized*) and example room (*real world*) environments. The hypothesis was that an RSSI signature can be used to characterize the environment and therefore distinguish between environments of distinctly different levels of multipath (such as real world vs. anechoic chambers). This hypothesis was tested by using a neural network trained using data from these two categories, and then tested to determine how effective the trained neural network can be at distinguishing between these environments. Using the results, an initial observational comparison was made to identify possible characteristics of RSSI signatures as a result of the environments in which they were collected.

The next experiment in this analysis compared the RSSI signatures using zero, one, and two reflectors in the fully anechoic chamber. For this experiment, the neural network was trained and applied to determine the number of objects in the environment. Observational correlations are made between the RSSI signatures of this experiment as well as those from the previous anechoic (no-multipath) and real world (complex multipath) experiments. It should be noted that while in Chapter 6 the RSSI signature was

demonstrated to be an effective tool for mitigating the multipath effects in distance estimations for both RFID and Bluetooth technologies, in this chapter, only passive RFID technology is evaluated. It is hypothesized that the general principles will be applicable to Bluetooth but not specific features.

This chapter will be organized as follows: first a brief background will be given as a summary of the relevant RSSI signature findings, followed by a classification analysis for different environments using the RSSI Signature method. The next section will explore classifying environmental settings based upon objects within an environment, and finally a conclusion will discuss the potential to use the RSSI signature to characterize environments.

8.2 Background

In Chapter 6, the complexity of the RSSI Signature was explored in an experiment where the reader and tag maintained a constant distance and orientation with respect to each other and were moved together through the environment. The resulting RSSI signature patterns (which can be seen in Figure 6-5 or 6-7) were a wave-like transition of the magnitude vs. frequency plot, essentially yielding a range of signatures even with without any change in the environment. This experiment demonstrated the intricate and elaborate relationship between the RSSI signature, objects in the environment, and the difficulty involved in correlating the two. The environment itself was unchanged, but simply by varying the experimental setup location the RSSI signature varied drastically. Nevertheless, there is a desire to at least begin to investigate and understand more generally

how the RSSI signature is related to the environment, and how it can be used to derive information about the surrounding environment.

8.3 Fully Anechoic vs. Real World

The first step in characterizing the signature, or using the signature to characterize the environment, is to simply determine if any such characterization is even possible. For this reason, a neural network is used to determine if this basic information about the environment is contained within the RSSI signature. Secondarily, the RSSI signatures from each test environment situation can be observationally evaluated to begin to hypothesize about the shape of the RSSI signature and how it may be correlated to its current setting.

The RSSI signature data, originally used in Chapter 6 to examine distance error, was recorded in several different environments of varying levels of multipath, including a fully anechoic chamber, and several warehouse and laboratory type environments, as described in Appendix A, section A.3.

The categories for classification for the following experiment were:

1. Fully Anechoic (no additional objects)
2. Real World (non-anechoic, with few or many objects)

In each instance, the ThingMagic M6e radio was used, along with the circularly polarized reader antenna, and an Alien Squiglette tag vertically polarized described in more

detail in Appendix A. The same passive RFID tag was used for all experiments for the sake of consistency. Both the tag and reader antenna were maintained at a distance 1m above the ground plane. The method of smoothing, as described in Chapter 7, was used for the analysis of the RSSI signatures to reduce the number of features to 25 points and handle any incomplete signatures.

A subset of these signatures was used for training a classification neural network (90%), and the remaining (10%) were used solely for testing the efficacy of the neural network. The neural network was trained using four hidden layers and all analysis of the neural network's predictive accuracy was done using the results from the test datasets. In previous analysis, it was found that more than four hidden layers would likely result in overfitting of the data. A "smoothed" or rolling mean version of each RSSI signature was used as the input, as was discussed in the previous chapter to handle incomplete datasets. Exactly 100 neural networks were trained and tested to find the typical level of classification accuracy. The results of this experiment can be seen in Figure 8-1.

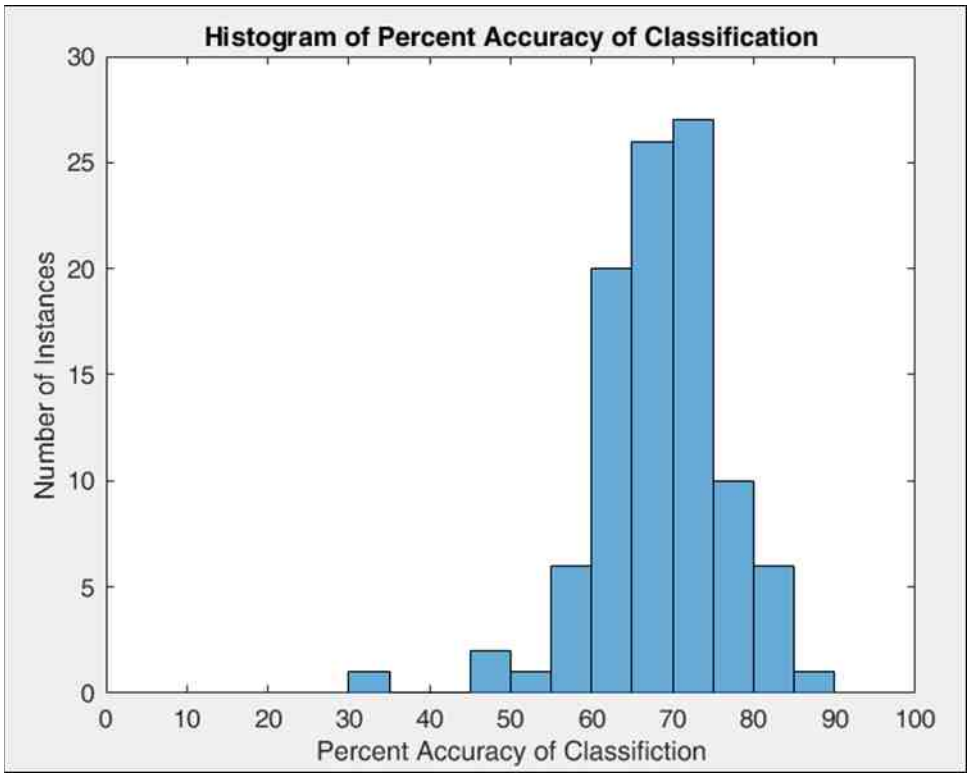


Figure 8-1: Histogram of accuracy from neural network classification of the environment (into fully anechoic vs. real word) based on RSSI signature.

These results (seen in Figure 8-1) illustrate that a RSSI signature trained neural network can in fact classify if the environment in which the signature was recorded was fully anechoic or real world. Given that there were two categories, a completely random chance would result in a histogram centered about 50%. While the neural network does not yield high classification accuracy, it is statistically significantly greater than random chance, with a median classification accuracy of 69% (and a p-value of $3*10^{-41}$).

As a first step in understanding the relationship between the shape of the RSSI Signature and the type of environment it was recorded in, the mean and standard deviation were calculated at every frequency to give an average RSSI signature shape for both the anechoic and non-anechoic settings. The vertical offsets between the two were eliminated

for a simpler comparison of the overall shape (allowing them to overlap). The resulting average RSSI signature shapes for the two environments can be seen in Figure 8-2.

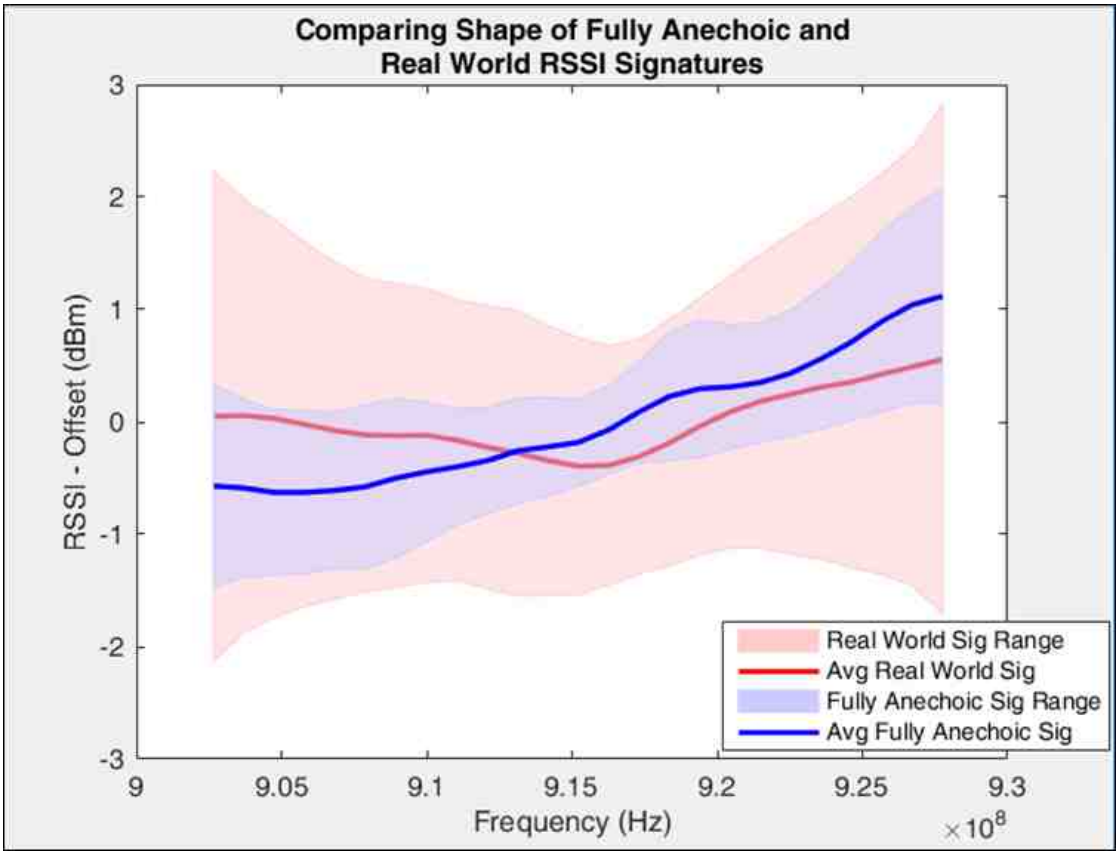


Figure 8-2: Shape of RSSI signature for fully anechoic and real-world environments and the standard deviation range at each frequency.

When observing the different overall shapes of the anechoic and real world (non-anechoic) signatures, there are a few distinctions between the two. Primarily, the standard deviation range of the anechoic signature is much narrower than that of the real-world environments. This makes sense given that a fully anechoic chamber should essentially be simulating free-space, thus any positioning of the RFID tag and reader should theoretically yield the same results. Perhaps the deviation that does exist in the fully anechoic data is indicative of the slight imperfection of the anechoic chamber. Even fully anechoic

chambers have a few areas of the floor not covered by absorbers, for researchers to move about and set up experiments these and other imperfections could be the source of these variations.

A second key difference in Figure 8-2 is in the shape of the average signature. The fully anechoic signature has a gradual positive slope, while the real world signature is more horizontal with a dip in the middle of the signature. Interestingly, if these two average signatures are compared to a theoretical free space signature calculated with the Friis equation, as can be seen in Figure 8-3, the theoretical signature instead has a negative slope.

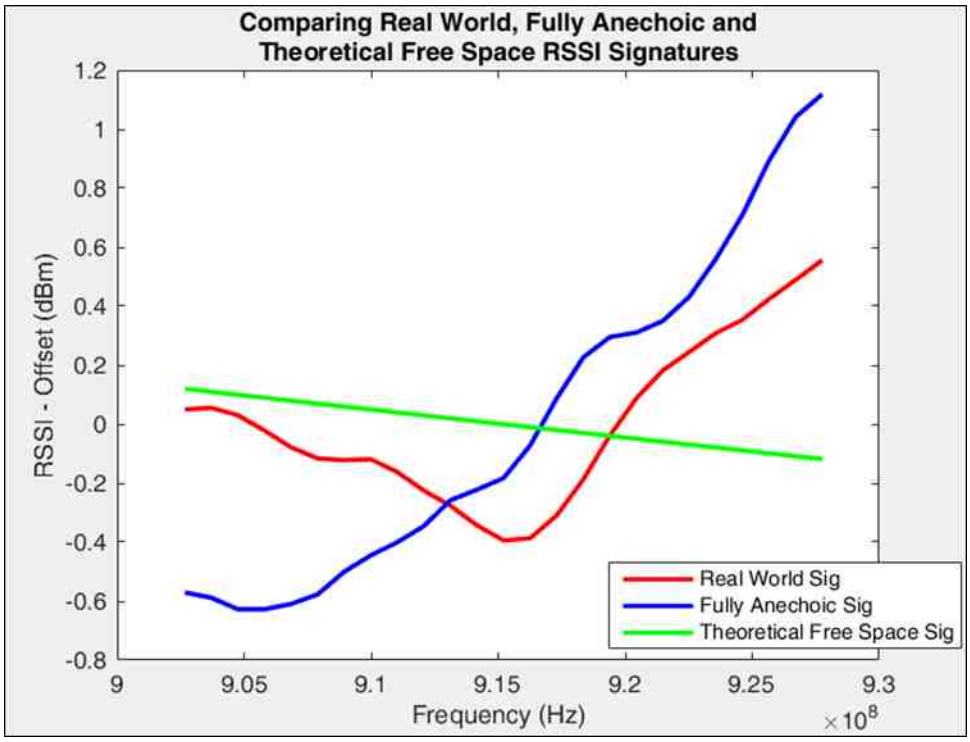


Figure 8-3: Average RSSI signature for real world, fully anechoic, and theoretical free space

Theoretically the free space Friis equation signature, and the measured fully anechoic signature should be equivalent; however, the fully anechoic chamber is not

perfectly anechoic and Friis equation was not intended to be used in this manner. Typically, the Friis equation is used when calculating the power loss over a distance range, rather than the power loss with a set distance over a frequency range. Therefore, it is likely that the Friis equation isn't a suitable model for RSSI signatures.

When observing the difference between the real world and fully-anechoic signatures in Figures 8-2 and 8-3, there is a discernable difference, yet no good reference or model which could describe what an RSSI signature should look like in a given environment. The next step would therefore be a series of experiments which introduce a small number of RF reflective objects to the idealized environment. It is likely that the complexity of multipath yields a multitude of possible signatures rather than a general signature which can be theoretically modeled.

8.4 Grouping Similar RSSI Signatures

For the next experiments, the idea of characterizing or classifying the environment was approached from a slightly different perspective. Given a generally constant environment, could the signature be used to identify changes in an environment, or perhaps the number of additional objects?

8.4.1 Aluminum Cube

An experiment was devised in which a single $(43\ cm)^3$ aluminum covered cube was placed in a variety of locations within a fully anechoic chamber. An RFID tag and reader remained unmoved at opposite ends of the chamber at a distance of 3.35m apart (as shown in Figure 8-4). For each position of the cube, the RFID tag was read and the RSSI signature

measured. This process was repeated twice for each location of the aluminum cube, including a control where the cube was removed entirely from the anechoic chamber. For all cases the orientation of the cube remained constant, with the faces of the cube aligned with the grid shown in Figure 8-4; where each increment on the grid in represents 0.3 m.

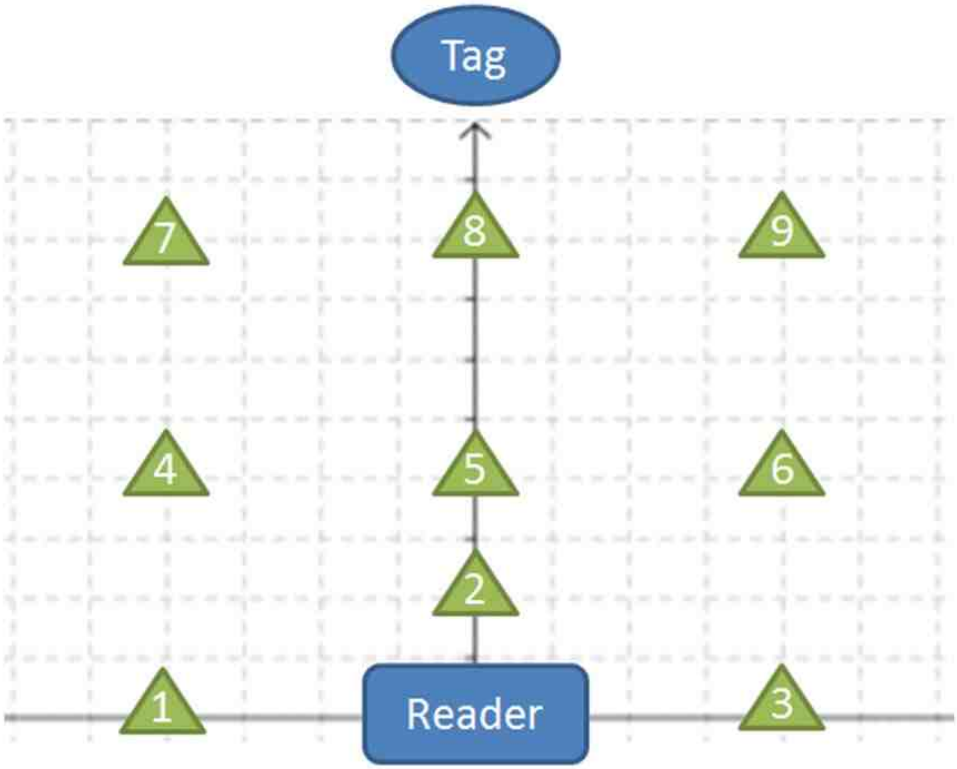


Figure 8-4: Diagram of locations of aluminum cube in anechoic chamber

The aluminum-covered cube was used to induce multipath, since radio waves would be reflected off of its surface. If the signal traveling between the reader and the tag hit the aluminum cube, this would serve as a small amount of introduced multipath. Due to the fact that multipath is believed to be a key contributor to the patterns of the RSSI signature, the hypothesis for this experiment was that the various positions of the aluminum cube would result in related changes in the RSSI signatures.

Once the experiment was complete, and the RSSI signatures were analyzed, there were a few noticeable observations. While all of these signatures are quite similar to each other, further inspection revealed groups of nearly identical signatures. The top graph of Figure 8-5 shows the two nearly identical signatures both measured at position 2 (from Figure 8-4), demonstrating measurement repeatability for similar environments. In contrast, the bottom graph of Figure 8-5 is of visibly distinctive signatures measured when the cube was at two different positions, (1 and 7, respectively). This illustrates how unchanged multipath environments yield repeatable signatures, but variations in the environment create varied RSSI signatures.

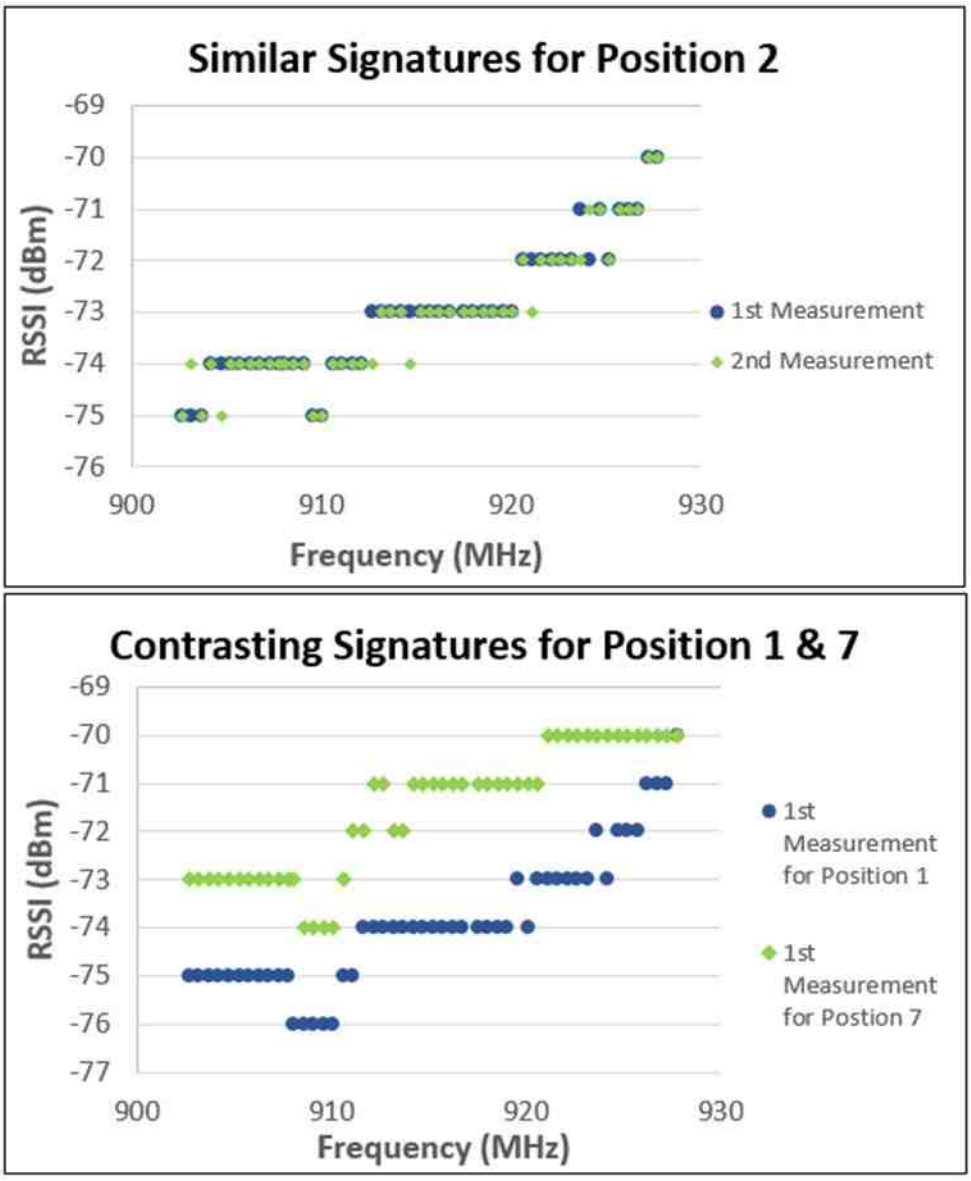


Figure 8-5: Top: Similar RSSI signatures recorded at cube position 2. Bottom: Contrasting RSSI signatures recorded at cube positions 1 & 7 as shown in diagram 8-6.

Based upon these observed similarities and differences, the signatures were visually categorized into 5 different groups based upon their similar shapes. Once these groups were created, the classification of each was then related back to the position of the aluminum cube, revealing the following pattern shown in Figure 8-6.

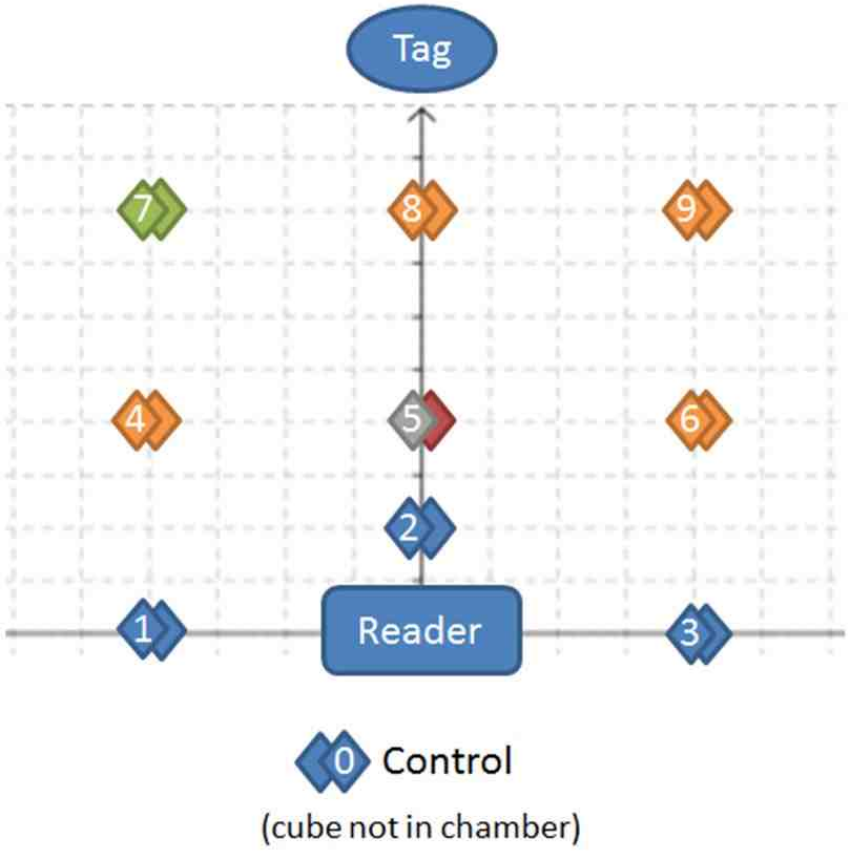


Figure 8-6: Groupings of RSSI signatures in relation to the location of the aluminum cube when the RSSI signatures were recorded.

In Figure 8-6:

- Each "♦" represents a recorded RSSI signature when the cube was at that location.
- Each color "♦ ♦ ♦ ♦ ♦" represents a group of visually similar signatures.

It is interesting to note that, except for position 5, the RSSI signatures at any individual location were visually grouped together as would be expected. At position 5, these two signatures each make up groups of their own. One possible reason is that position 5 is almost half way between the reader and tag and would therefore be expected to have

the largest influence on the multipath. This strong effect on multipath could cause even small unintended changes in the location or orientation of the cube to significantly impact the RSSI signature. Also of interest is the fact that the RSSI signatures of positions 1, 2, and 3 look like that of the control, where the cube was located outside the chamber completely. It's conceivable that for these situations that the cube had negligible impact on the RSSI signature. For positions 1 and 3, the box could be outside of the radiation pattern of the circular polarized reader antenna, resulting in negligible impact from multipath. Given that the cube was only 43cm tall and the reader antenna was 1 m above the ground for position 2, the beam of the reader antenna may have passed completely over the cube, without generating any multipath reflections to or from the tag.

Although the groupings of the RSSI signatures do not exactly parallel the location of the cube, there is a definite association. This observed relationship is further indication of the intricate correlation between the multipath of the surrounding environment and the RSSI signature.

8.4.2 Multiple Reflectors Experiment

For the next experiment, a combination of two reflective surfaces were used to determine if it was possible to identify the number of electromagnetically significant objects in the room. Three scenarios were tested and compared.

The situations tested in this experiment were:

1. Fully Anechoic (no additional objects)
2. Fully Anechoic with One Reflector
3. Fully Anechoic with Two Reflectors

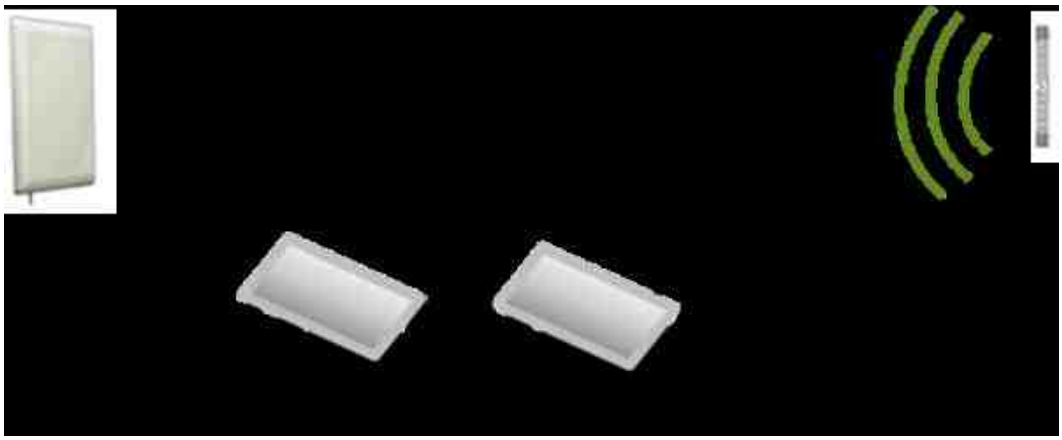


Figure 8-7: Representation of experiment with two reflectors

The data for this experiment was collected in a 5-meter fully-anechoic chamber with the ThingMagic M6e radio, circularly polarized antenna, and Alien Squiglette RFID tag vertically polarized. The reflectors consisted of 0.5 x 0.5 m^2 aluminum foil-covered cardboard as shown in the diagram (Figure 8-7). The reader and tag were maintained at the same location, orientation and separation distance for all of the recorded samples. Thus, the only changes to the environment were the addition of one or two reflective objects. For each of the three conditions, 100 RSSI signatures were collected, such that a neural network could be trained solely with the data from these datasets. As in previous chapter and experiments a random 90% of the data was used for training the neural network, and the remaining 10% of test data was used for the analysis, as those shown in Figure 8-8. Several neural networks were trained and tested, and all of them had nearly identical results as in Figure 8-8.

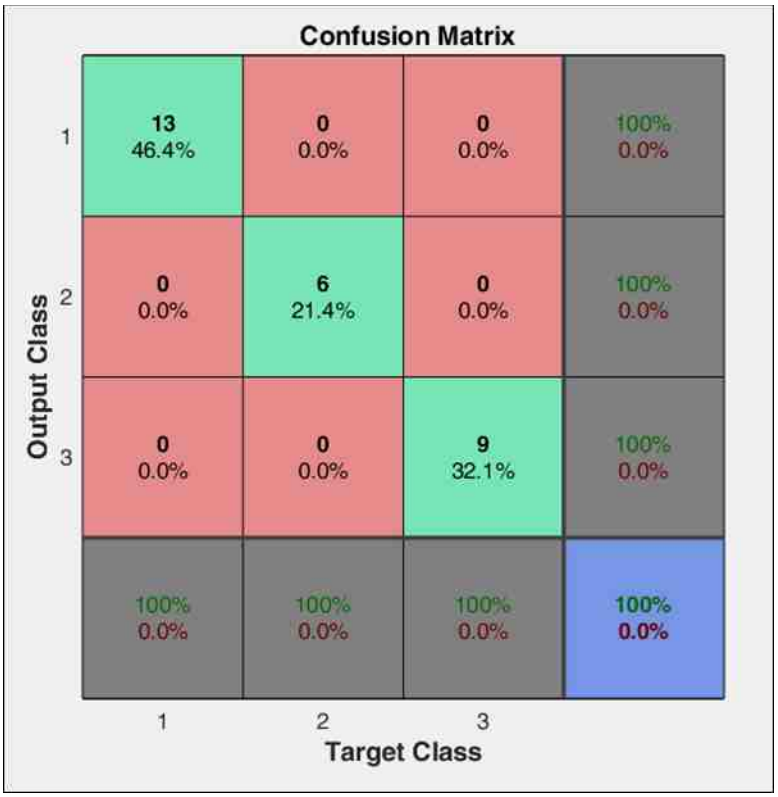


Figure 8-8: Confusion matrix of classification of signatures/environments with situations: 1) empty anechoic chamber 2) one reflector and 3) two reflectors

The confusion matrix shown in Figure 8-8 represents the actual conditions of each signature on the bottom, called the target class, and the neural network's guess based on the RSSI signature on the left, called the output class. For each classification the neural network had a 100% positive identification rate.

This is an interesting result in comparison to the results of Fully Anechoic/Real World experiment. For different environments, the classification rate for only two classes was around 69%, whereas the classification rate for the three classes of quantity of objects was 100%. One likely reason for this is the higher level of consistency involved in this experimental set-up given that all of the data was recorded in one session. For this the

reader and tag were kept in the same location for all of the measurements, thus the RSSI signatures would be more consistent. In contrast, the RSSI signatures from fully-anechoic/real-world experiments were from a variety of different environments, separation distances, and experiment sessions. Thus it was likely easier for the neural network to distinguish between the signatures the different number of objects in the same environment, than to make generalizations about the environment as in the first experiment. To further investigate how the RSSI signatures relate to the environments in which they were collected, the same analysis used in Figure 8-2 was applied to this experimental data set. The average signatures have been collected for each of the three conditions of the number of objects experiment and overlaid to observe the differences. The results are shown in Figure 8-9.

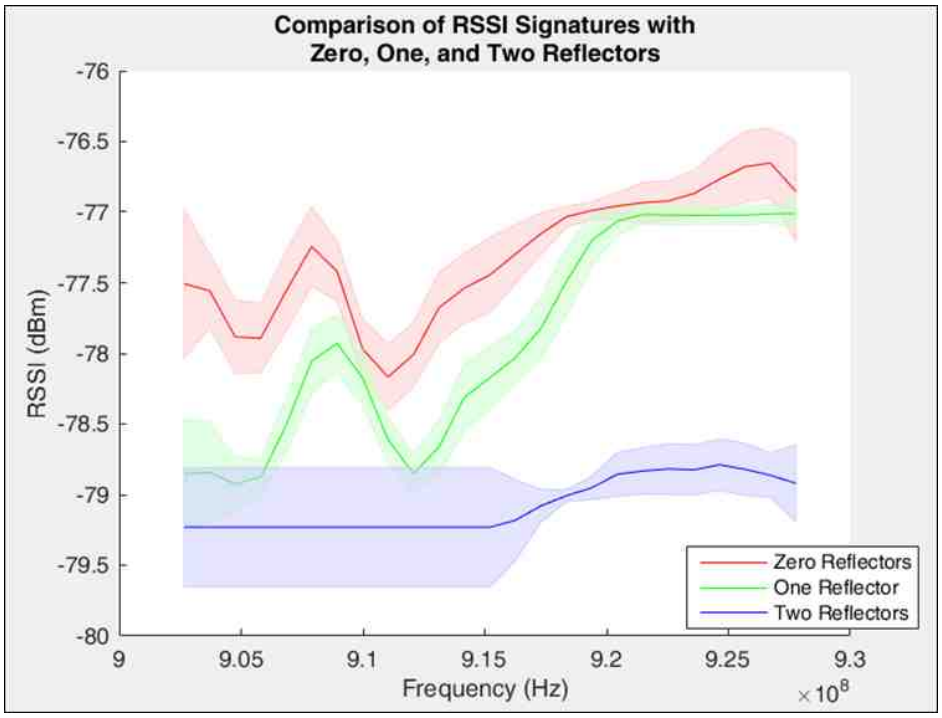


Figure 8-9: Comparison of RSSI signatures from situations: 1) empty anechoic chamber 2) one reflector and 3) two reflectors

In Figure 8-9, the average RSSI signatures are graphed along with the standard deviation at each frequency for each of the situations 1) fully anechoic chamber 2) one reflector in the fully anechoic chamber and 3) two reflectors in the fully anechoic chamber. For the case of two reflectors, shown in blue, below 915 MHz the RSSI signature is horizontal and uniform. This is due to destructive interference from the multipath of the RF signal, yielding amplitudes below the noise floor of the radio meaning the tag couldn't be real. While the radio simply did not read these frequencies, the smoothing algorithm developed in the previous chapter which was intended to handle incomplete RSSI Signatures filled in these gaps with the RSSI value at the nearest successfully measured frequency.

Looking at Figures 8-9, it is easy to tell how the neural network was able to confidently classify these three situations. Interestingly, the cases of the empty fully-anechoic chamber and the fully-anechoic chamber with one reflector, shown in red and green respectively in Figures 8-9, are the most visually similar. The RSSI signature from the second reflector, shown in blue, has the greatest distinction. Due to angles of reflection, the greatest from the multipath of a reflector would be in the middle of the path between the reader and the tag. It is possible that while both of the reflectors were placed between the reader and tag, the second reflector may have been slightly closer to the true mid-point, thus resulting in a more drastic change in the RSSI signature.

Regardless of the reason why the second reflector seems to have had a larger impact on the RSSI signature, each of the RSSI signatures from the three cases are unique and easily distinguishable, even simply by observation. Therefore, it could be hypothesized

that in other cases, given a baseline or reference RSSI signature, additional objects or changes in the environment could be detected by measuring and comparing the RSSI signatures.

8.5 Conclusion

Based on the experiments in this and previous chapters, it is clear that the RSSI signature is tightly correlated with the surrounding physical environment. In this chapter, an initial investigation was made into the possibility of characterizing the RSSI signature and relating it to the environment in which it was recorded. The fully-anechoic vs. real-world comparison used RSSI signatures measured in a variety of real-world environments, and in a fully-anechoic chamber, then compared the signatures using a neural network classification algorithm. When several neural networks were trained and tested, it was found that the neural networks achieved an accuracy of approximately 69% in correctly identifying the type of environment in which the RSSI signature was recorded. While 69% is statistically significantly greater than a random chance, it is not generally considered a high level of accuracy. This implies that while there may be some slight distinction between the RSSI signatures for the two environments, there seems to be a great deal of overlap and ambiguity as well.

In contrast, the single environment with different quantities of objects consistently achieved the high level of accuracy of 100%. For this experiment, many RSSI signatures were collected for the situations of 1) an empty fully anechoic chamber 2) one reflector in a fully anechoic chamber and 3) two reflectors in a fully anechoic chamber. When the

RSSI signatures from these three conditions were compared, they were obviously and visually distinct.

These experiments helped investigate the cause and effect relationship between the surrounding physical environment and the measured RSSI signature. They indicated that while classifying general types of environments using the RSSI signature may not yield a high level of accuracy, the RSSI signature can effectively be used to distinguish between two different environments or changes in an environment, when given a baseline or reference RSSI signature for comparison.

MATLAB code for this work is given in Appendix C, section C.9.1 for code that classifies the environment and C.9.2 for code that characterizes the environment.

CHAPTER 9. METHODS OF TRILATERATION

9.1 Overview

Moving from distance estimation to localization (specifically trilateration as discussed in Chapter 2) involves comparing a variety of algorithms that use distance information and output the coordinates of the item in question. For indoor localization, typically only a 2D location is necessary and the third dimension adds significantly more complexity to the problem and requires more satellites. Therefore, trilateration in this chapter will focus on 2D location estimates. The goal of this chapter is to identify and mathematically reproduce each of the key trilateration geometry methods used in localization. Note that each method presented here is represented by MATLAB code which can be found in Appendix C, section C.10.

9.2 Background

Trilateration uses distance estimations from reference readers, more generally called satellites, in order to locate a device, known as a beacon. The terminology of satellites and beacons is due to trilateration algorithms often being used for GPS type applications but have expanded to become general terms for technology agnostic localization. For purely geometrical methods it is assumed that the circle's radii will be absolutely accurate, and therefore trilateration will produce a single location (represented in Figure 9-1), in real world applications however, it is recognized that most likely the circles will not converge on a single location. Thus, the various methods use combinations of trilateration and statistical algorithms to compensate for some error in the distance calculations\ in order to converge on a single likely location.

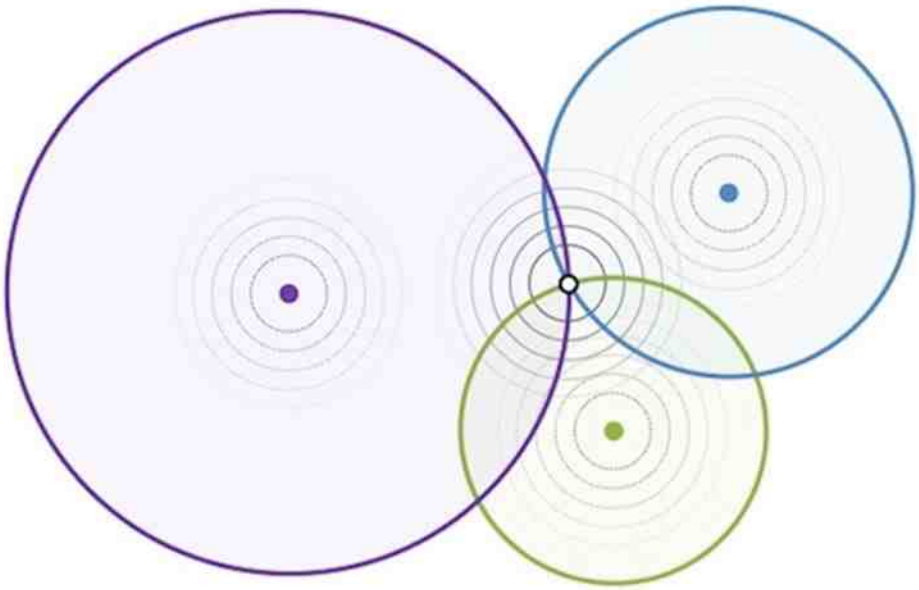


Figure 9-1: Zero error in distance estimation leading to a single location through trilateration

Most trilateration methods begin by mapping the separation distance from the satellites to the beacon as a circle, with the satellite in the center of each circle. Theoretically for any one circle that means that the beacon could be located at any point along that circle. When two circles overlap, they narrow down the possible beacon locations to the two points where the circles overlap as shown in Figure 9-2. Thus three satellites are needed such that a single point is found, which gives the location of the beacon.

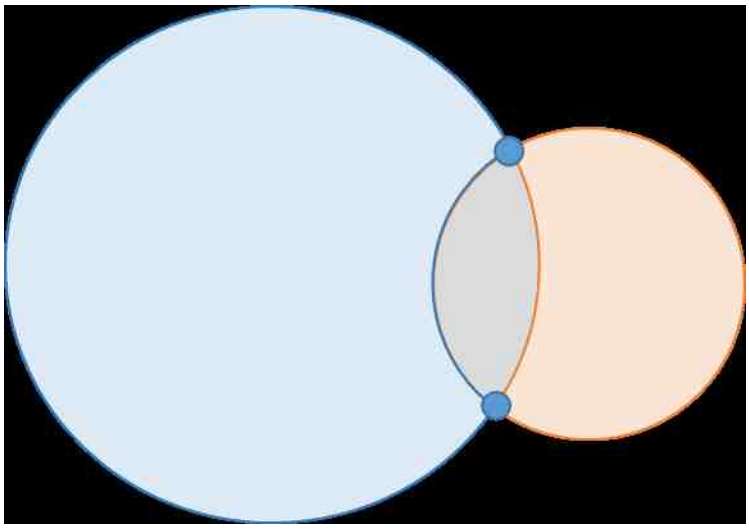


Figure 9-2: Overlapping satellite radii

This chapter reviews nine distinctive trilateration algorithms from the literature search conducted for this research. Based on the requirements of this research, these algorithms will be discussed for 2-D localization, requiring a minimum of three satellites.

9.3 Linear Least Squares

The method of linear least squares is the most common method of trilateration, since distance formulas inherently take the shape of a linear least squares equation. One issue however, is that this method does not consider the nonlinearity of the fundamental problem of trilateration. Linear least squares method essentially uses an assumption that measurements have equal variance and are completely uncorrelated. Linear least squares is calculated as follows [57]:

$$X = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \qquad \text{Equation 9-1}$$

$$A = \begin{bmatrix} 2(x_1 - x_0) & 2(y_1 - y_0) \\ 2(x_2 - x_0) & 2(y_2 - y_0) \\ \vdots & \vdots \\ 2(x_n - x_0) & 2(y_n - x_0) \end{bmatrix}$$ Equation 9-2

$$b = \begin{bmatrix} (x_1 - x_0)^2 + (y_1 - y_0)^2 - {d_1}^2 \\ (x_2 - x_0)^2 + (y_2 - y_0)^2 - {d_2}^2 \\ \vdots \\ (x_n - x_0)^2 + (y_n - y_0)^2 - {d_n}^2 \end{bmatrix}$$ Equation 9-3

Where

x_0 = x-coordinate of initial guess for location of beacon

y_0 = y-coordinate of initial guess for location of beacon

x_i = x-coordinate of satellite i

y_i = y-coordinate of satellite i

d_i = RF measured distance from satellite i to the beacon

n = number of satellites

$$e = A * X - b$$ Equation 9-4

$$X = (A^T A)^{-1} A^T b$$ Equation 9-5

Unlike the weighted or iterative least squares methods, one major benefit is that linear least squares method is a closed form solution for trilateration. The resulting coordinates of the location of the beacon are given by the vector X.

9.4 Linear Locus of Position

Linear locus of position is a purely geometrical method of trilateration which uses the intersection of lines to find the location of the beacon in the solution space. This method starts with the overlap of any two satellite circles. With these overlapping circles, a line is fit through the two points where the circles overlap. If this is repeated using a third or different pair of satellites, two lines are created, the intersection of these lines marks the location of the beacon. Figure 9-3 shows two different scenarios using linear locus of position to find the location of a beacon.

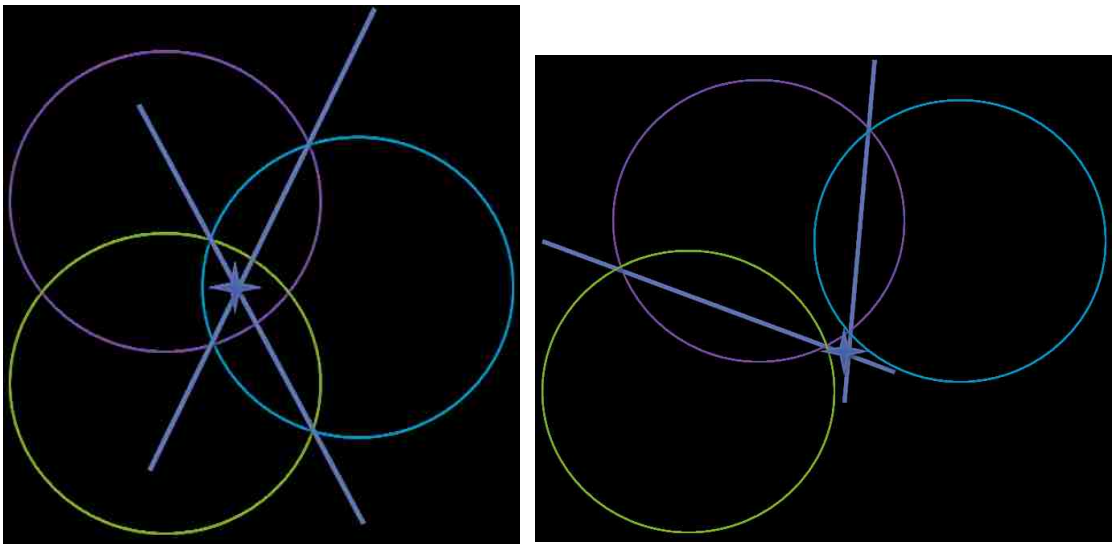


Figure 9-3: Two different linear locus of position scenarios

Linear locus of position is an extremely quick and easy way to narrow down the solution space to a single location. It even provides accurate results when both satellites have equal error, as can be seen in Figure 9-4. This can often be the case in GPS systems, where a beacon may have a constant timing/clock offset, and yielding equal distance error to all of the satellites.

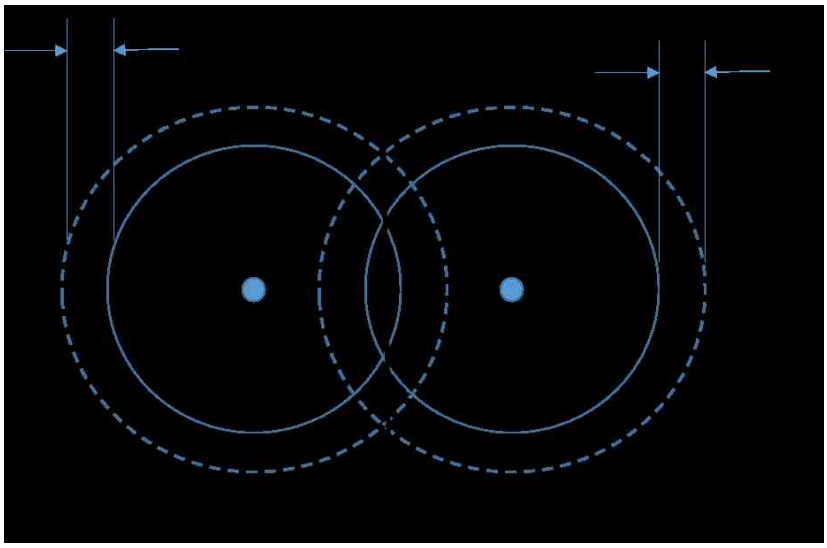


Figure 9-4: Linear locus of position in the presence of equal distance error

For indoor environments, equal distance error is seldom a reasonable assumption, and thus the linear locus of position method can yield large location errors in less predictable indoor conditions. Another potential issue of the linear locus of position method is that sometimes the satellites significantly underestimate the separation distance, meaning the circles don't overlap at all. For this research analysis, when this is the case the distance half-way between the satellites is used as a fixed point for the line, with the slope of the line perpendicular to an imaginary line between the two satellites. Finally, the linear locus of position method cannot incorporate more than three satellites into its formulation, because that would yield multiple intersection points. Thus when using this method for comparison in this research, the three satellites with the highest RSSI values or confidence levels are used in the algorithm.

9.5 Centroid

Like the linear locus of position method, the centroid method uses the locations of where the satellite circles overlap. The x and y coordinates of each point are compiled using a method similar to finding the center of mass of an object with a uniform density [58], [59], as can be seen in Figure 9-5.

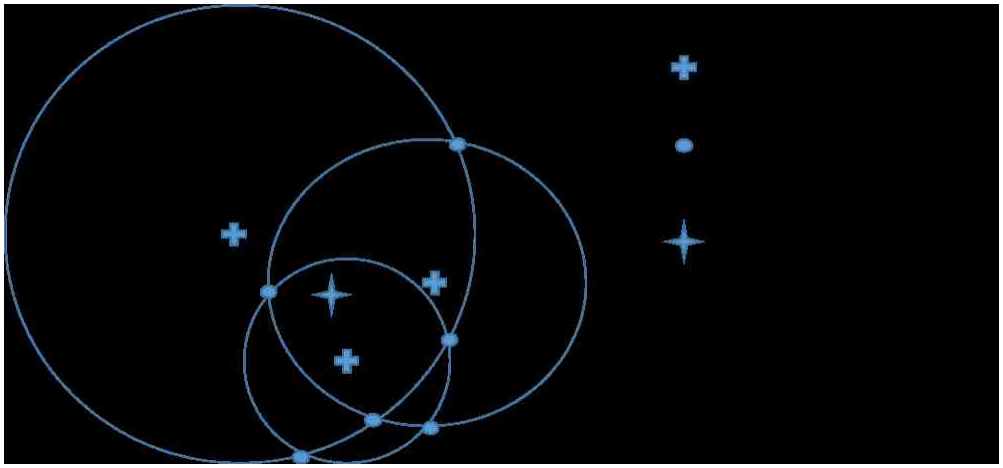
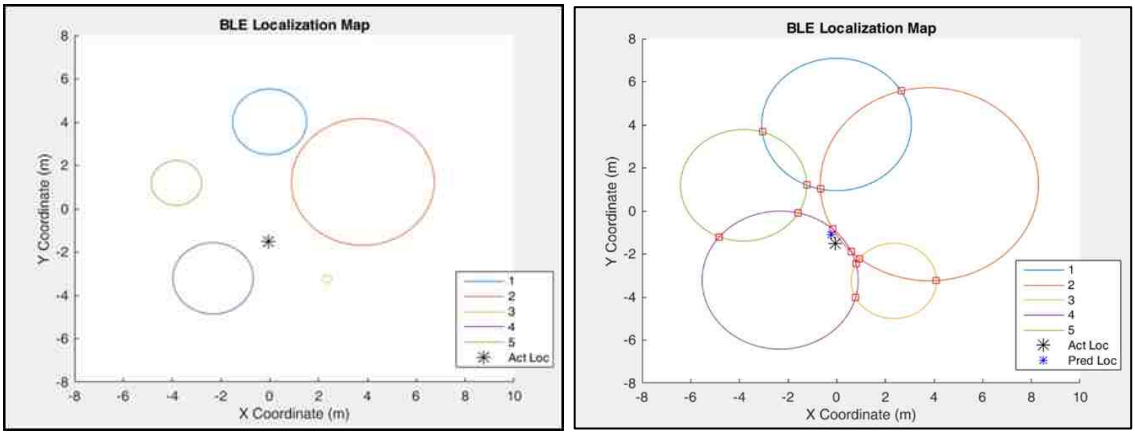


Figure 9-5: Beacon location found with Centroid Method.

When there is little error in the system this method works well at finding the center of the overlapping area, essentially the beacon location. It is also highly adaptable to a varying number of satellites. However, as with linear locus of position, when the radii of the satellites don't overlap at all, this creates a problem. Thus, in order to apply this method to all of the experimental data from this research, the following solution was implemented.

The idea is to expand all of the circles by the same amount such that each has some overlap with another circle. Once this is completed, the standard centroid method can be applied to find the beacon location. Figure 9-6 shows the original satellite radii, and the

expanded radii used to employ the centroid method. The data in Figure 9-6 was taken in the 10 meter fully anechoic chamber using BLE devices.



(a) (b)

Figure 9-6: (a) Original satellite radii and (b) expanded satellite radii to employ Centroid Method

The other issue with the centroid method is that it simply finds the average coordinate (an average of the x and y coordinates) from the overlapping points. Ideally there should be some weighting method which put more confidence in smaller radii circles and less confidence in larger circles. This is because smaller radii circles are the result of larger RSSI values. The relationship between RSSI and distance, as shown in Figure 3-1 ad reproduced here as Figure 9-7, illustrates how there is less distance resolution as the RSSI values decrease. Therefore, theoretically, higher RSSI values should be weighted in some manner with a larger confidence. The experimental set-up used to create Figure 9-7 is described in Appendix A, section A.4.12.

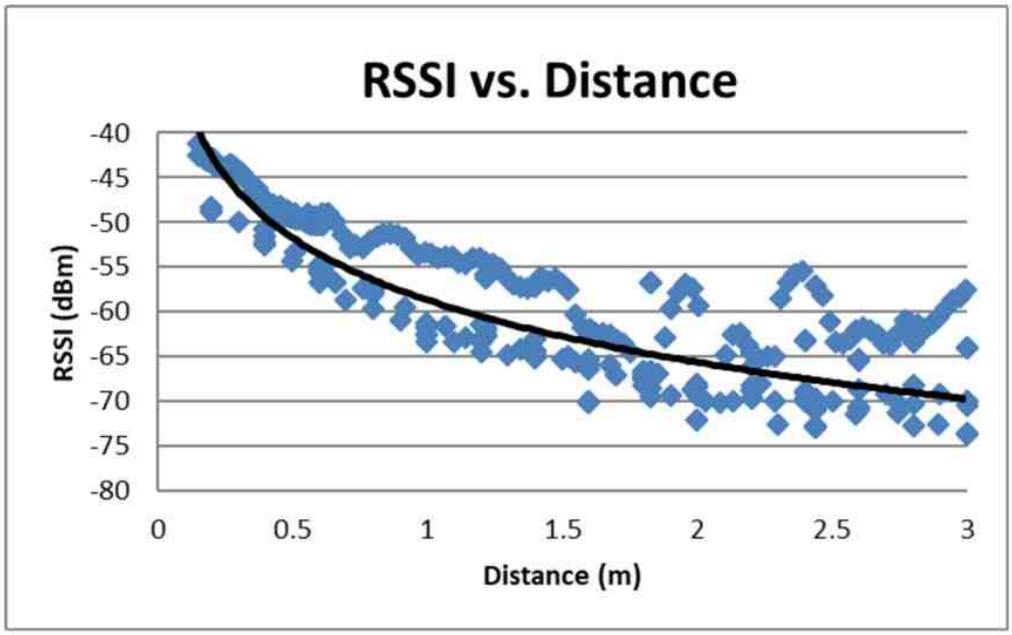


Figure 9-7: The relationship between RSSI and the distance separating the satellite and beacon

If weighting is to be implemented, alternative problems arise. For example, the experimental BLE data collected in the 10-meter anechoic chamber, which can be seen in Figure 9-6(a), has a very small radius for the bottom right hand satellite, shown in yellow. When such a large underestimation occurs, it can be detrimental to the overall trilateration accuracy because of the large weight put on one inaccurate point. To find a balance between these two issues, of no-weighting vs. overweighting, the solution previously mentioned was employed to address this problem as well.

When the satellite circles are expanded, to make the centroid method possible, all of the radii are increased by the same amount. Therefore, because the overall radii expanded equally, the ratio of the radii between satellites actually decrease. Then a weight to each of the circle intersection points is applied using the inverse of the new radii for both of the

circles. As radius increases, the weight will decrease, and an intersection point from two small circles will have a larger weight then on large and one small circle. Additionally, circles which are so large that they encompass all the others, will not have an intersection point and won't be used in the calculations. This is another advantage because satellites with a vast radius, are often least reliable references. To find the final value, the x and y coordinates, with their respective weights are averaged to find the likely location the beacon as shown in Equation 9-6 [58].

$$(X_G, Y_G) = \left(\frac{\sum_{i=1}^{n} m_i X_i}{\sum_{i=1}^{n} m_i}, \frac{\sum_{i=1}^{n} m_i Y_i}{\sum_{i=1}^{n} m_i}\right) \quad \text{Equation 9-6}$$

9.6 Linear Approximation

The linear approximation method uses some geometrical assumptions which mathematically move the satellites to convenient locations at the origin and along the y-axis to turn the trilateration system of equations into a single linear solution. The basic trilateration equation is the Pythagorean Theorem or the equation of a circle.

$$r_{1,2,3}{}^2 = \left(x - x_{1,2,3}\right)^2 + \left(y - y_{1,2,3}\right)^2 \quad \text{Equation 9-7}$$

Where

r = read distance from satellite to beacon [m]

x = x-coordinate of beacon location [m]

y = y-coordinate of beacon location [m]

$x_{1,2,3}$ = x-coordinate of satellite 1, 2, & 3 locations [m]

$y_{1,2,3}$ = y-coordinate of satellite 1, 2, & 3 locations [m]

Theoretically, one would simply apply this equation three times for each of the satellites, and solve the equations for the x and y coordinates of the beacon. However, these equations cannot simply be solved using standard mathematical methods, which is one of the key reasons why there are so many trilateration algorithms.

In [60] the system of equations was simplified using assumptions about the center location of each of the satellites, see Figure 9-8.

Locations of Satellites:

Satellite 1 = (0, 0)

Satellite 2 = (x_2, 0)

Satellite 3 = (x_3, y_3)

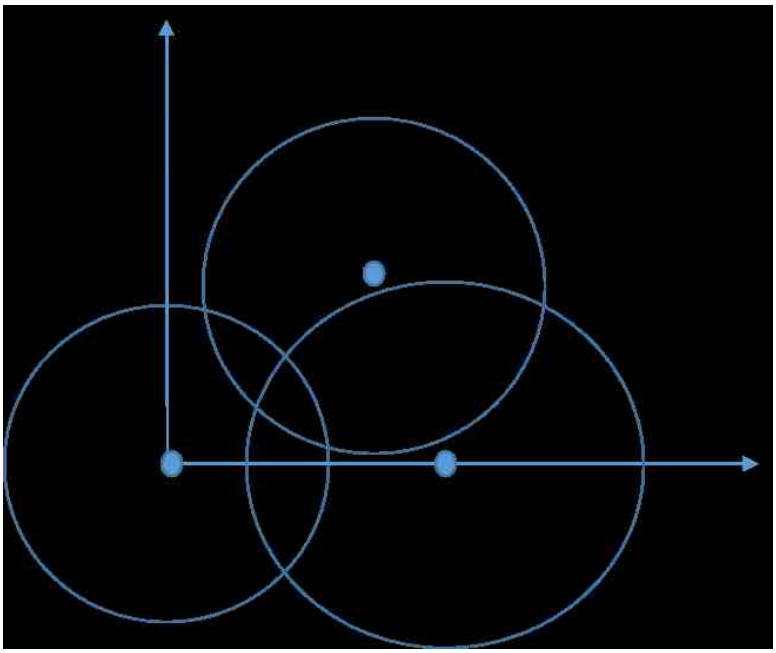


Figure 9-8: Linear approximation using three satellites

Uses these satellite centers, the equations of the three circles become:

$$r_1{}^2 = x^2 + y^2 \qquad \text{Equation 9-8}$$

$$r_2{}^2 = (x - x_2)^2 + y^2 \qquad \text{Equation 9-9}$$

$$r_3{}^2 = (x - x_3)^2 + (y - y_3)^2 \qquad \text{Equation 9-10}$$

These equations can then be simplified and solved for the x and y coordinates of the beacon.

$$x = \frac{r_1{}^2 - r_2{}^2 + x_2{}^2}{2x_2} \qquad \text{Equation 9-11}$$

$$y = \frac{r_1{}^2 - r_3{}^2 + x_3{}^2 + y_3{}^2 - (2x_3 x)}{2y_3} \qquad \text{Equation 9-12}$$

Once the location of the beacon is solved for in this theoretical reference frame, it must be translated back into the original reference frame of the environment.

This method of a linear approximation is an extremely efficient method of calculating the location of the beacon; however, it is a simplification of the problem which could potentially cause it to be less accurate in application. One additional issue is the lack of flexibility. These equations only work for exactly three satellites, which may not be the best solution, given that the problem statement requires a "varying number of readers." To deal with this issue when testing the algorithm, only the satellites with the highest RSSI values or greatest level of confidence were used.

9.7 Hyperbolic Locus of Position

The hyperbolic locus of position method works very similarly to the linear locus of position method, except that the nonlinearity of the hyperbolas add a weighting factor to

the algorithm by bending towards the satellite with the shortest read distance. Hyperbolic based localization was described in [61], however an error in their description prevented that method from being directly applied here. It is noted that the method is also generally described in [62], however not in detail that can be used to actually implement an algorithm. One of the contributions of this work, the algorithm described below was developed based on these sources, but derived to be a functional localization algorithm that was successfully tested with experimental data.

A shorter read distance will often relate to a more confident measurement, thus by using a hyperbola instead of a line, it's possible to weight the confidence of the two measurements. Figure 9-9 shows a comparison of the linear vs. hyperbolic locus of position methods.

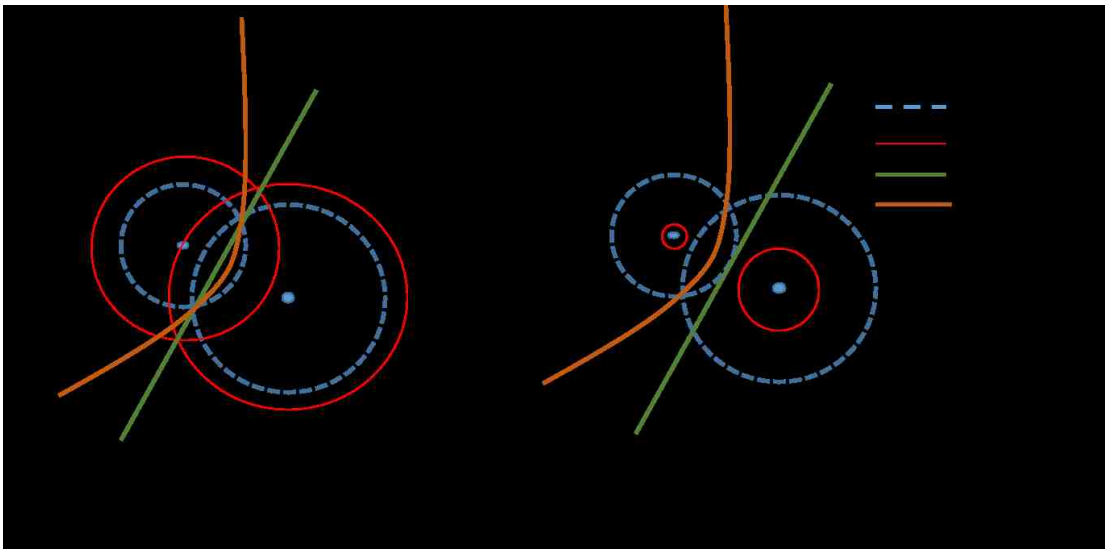


Figure 9-9: Using linear vs. hyperbolic locus of position methods for (a) overestimation and (b) underestimation.

One of the major challenges with the hyperbolic locus of position was that it could only be found discussed in generalities in the literature. Therefore the equation for this method had to be derived and created.

As with the linear locus of position method, if the estimated distances from the satellites do not overlap, the midpoint between the two satellites is used as a fixed point, and in the case of the hyperbola, it is used as the vertex for the hyperbola (as can be seen in Figure 9-9(a)).

Implementing this hyperbolic locus of position method begins with the basic equations for hyperbolas, given in Equations 9-13 and 9-14.

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad or \quad \frac{y^2}{b^2} - \frac{x^2}{a^2} = 1 \qquad \text{Equations 9-13 \& 14}$$

Where:

x, y = x and y Cartesian coordinates

a, b = constants which relate to the eccentricity of the hyperbola.

Equation 9-13 gives an East-West opening hyperbola, as shown in Figure 9-10(a), and Equation 9-14 models a North-South opening hyperbola as illustrated in Figure 9-10(b).

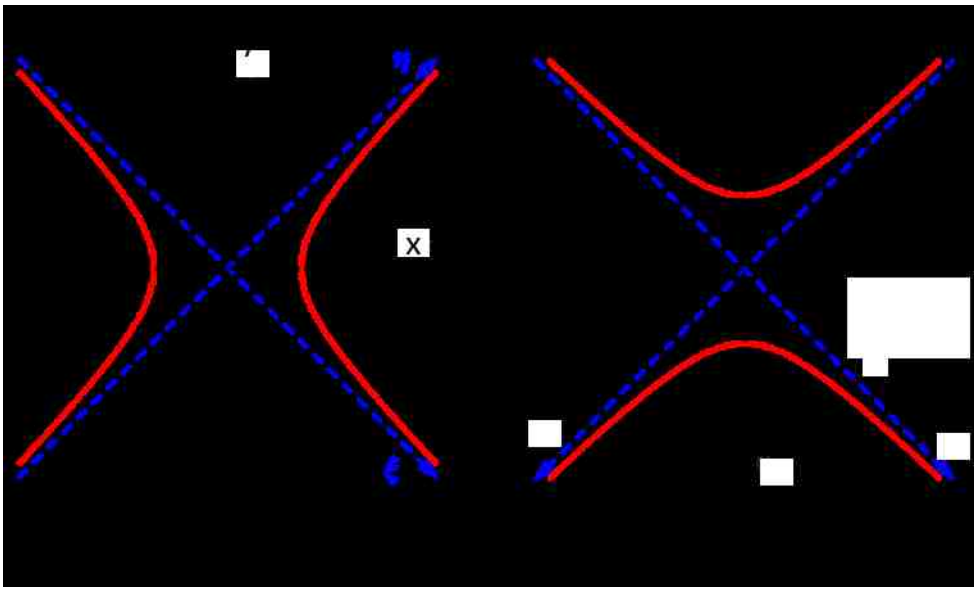


Figure 9-10: Example hyperbolas (a) East-West Opening hyperbola and (b) North-South Opening hyperbola

The North-South or East-West opening hyperbola equations make it easy to calculate the eccentricity, by using Equations 9-15 and 9-16.

$$c^2 = a^2 + b^2 \qquad \text{Equation 9-15}$$

$$e = \frac{c}{a} \qquad \text{Equation 9-16}$$

Where

c = Distance from the center to the focus

e = Eccentricity

Eccentricity is a geometrical characteristic which describes the curved nature of the hyperbola. To put it in perspective:

- The eccentricity of a circle = 0

- The eccentricity of an ellipse > 0 & < 1
- The eccentricity of a parabola = 1
- The eccentricity of a hyperbola > 1
- The eccentricity a line = ∞

When using the hyperbolic locus of position method, the algorithm begins by applying the read distance from the satellite to find the eccentricity of the hyperbola as given by Equation 9-17 [61].

$$e = \frac{D}{|\Delta|} \qquad \text{Equation 9-17}$$

Where:

D = Distance between satellites (m)

Δ = Difference in the predicted read distances from each of the satellites (m)

The center is found by the midpoint between satellites, as shown in Equation 9-18.

$$c = \frac{D}{2} \qquad \text{Equation 9-18}$$

Using the center distance "c", Equations 9-15 and 9-16 can be applied to solve for the constants "a" and "b", which define the pair of hyperbolas. When applying the hyperbolic locus of position method, only the hyperbola facing the satellite with the smaller radius is used.

While the hyperbolic locus of position has distinct benefits over some of the other methods in terms of accuracy, there are four major issues with this method. The first is

using the equations of a hyperbola. Hyperbolas are typically given by two standard equation forms (Equations 9-15 and 9-16). However, these equations only describe hyperbolas in exactly vertical or horizontal positions with their centers on the origin. When applying hyperbolas for this method, they could have any center location or any orientation. One option to translate the basic hyperbola equation, by using Equation 9-19.

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \qquad \text{Equation 9-19}$$

This version of the hyperbola equation gives the versatility of graphing a hyperbola in any orientation, but it can be difficult to relate the parameters to the shape of the hyperbola, as with the standard equations. Thus, for this purpose the hyperbolas are calculated with the standard equations centered on the origin and aligned with the x axis, then translated and rotated in order to properly position with the associated satellites. An example of the translated and rotated pair of hyperbolas can be seen in Figure 9-11.

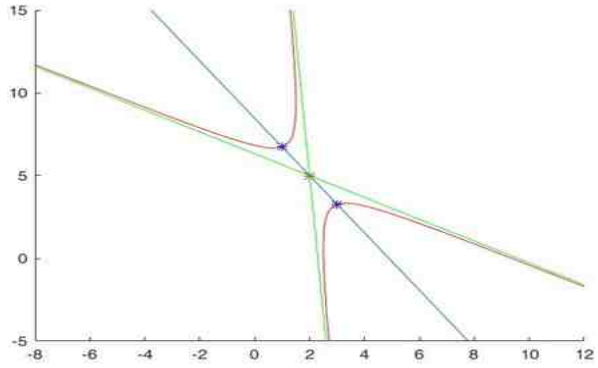


Figure 9-11: MATLAB graph of hyperbola after translation and rotation

The second major problem has to do with the nonlinearity of the hyperbola equations. Given that the hyperbola equations are nonlinear, an intersection of two

hyperbolas cannot be easily solved algebraically. The solution to this problem was to solve for the intersection numerically, which was fairly simplistic using MATLAB. If this algorithm was to be implemented as a real-time location algorithm, the method used to find the intersection of the hyperbolas could likely be improved to increase the computation speed.

The third issue with the hyperbolic locus of position method is that there are times when the two hyperbolas don't intersect at all. This is a likely outcome if the eccentricity, using the satellite locations and read distances, this can yield an elliptical rather than hyperbolic shape. To use this method, even when the hyperbolas and or ellipses don't overlap, the intersection point was chosen to be the point in space where the two are closest to each other.

The last problem is a common issue for trilateration algorithms, which is that only three satellites can be utilized. Thus, just as with the other methods, only three satellites were selected to be used in this formula.

9.8 Taylor Series Approximation

A Taylor Series approximation of location theoretically should be a computationally reasonable trilateration method suitable for cost-effective, real time location tracking. Taylor Series is a linear iterative method that alters the linearized equations as it approaches the best solution. The method selected for this research is described in [63]. In this method a first order Taylor Series is used to approximate the range measurement distances and optimize those to a location for a beacon based on distance radii from multiple satellites

and has been shown to be effective as long as the optimization starts with an initial guess that is fairly close to the final values.

The first order distance equations can be written as a series of simultaneous linear equations [N_r N_a] in matrix form. N_r and N_a correspond to the number of anchor nodes (satellites) and number of agents (beacons), respectively.

$$\Gamma\delta \approx z - e \quad \text{Equation 9-20}$$

Where:

$$\Gamma = diag\{\Gamma_1, \Gamma_2, \dots, \Gamma_{N_a}\} \quad \text{Equation 9-21}$$

$$e = \left[e_{11}, e_{21}, \dots, e_{N_r N_a}\right]^T \quad \text{Equation 9-22}$$

$$\delta = \left[\delta_{\theta_1}, \delta_{\varphi_1}, \delta_{\theta_2}, \delta_{\varphi_2}, \dots, \delta_{\theta_{N_a}}, \delta_{\varphi_{N_a}}\right]^T \quad \text{Equation 9-23}$$

$$\Gamma_i = \begin{bmatrix} a_{1i}{}^{(\theta_i)} & a_{1i}{}^{(\varphi_i)} \\ a_{2i}{}^{(\theta_i)} & a_{2i}{}^{(\varphi_i)} \\ \vdots & \vdots \\ a_{N_r i}{}^{(\theta_i)} & a_{N_r i}{}^{(\varphi_i)} \end{bmatrix} \quad \text{Equation 9-24}$$

$$z = \begin{bmatrix} \tilde{d}_{11} - d'_{11} \\ \tilde{d}_{21} - d'_{21} \\ \vdots \\ \tilde{d}_{N_r N_a} - d'_{N_r N_a} \end{bmatrix} \quad \text{Equation 9-25}$$

The weighted least squares solution can now be written as:

$$\hat{\delta} = (\Gamma^T \Sigma^{-1} \Gamma)^{-1} \Gamma^T \Sigma^{-1} z \quad \text{Equation 9-26}$$

Where Σ is the covariance matrix of the distance measurements.

$$\Sigma = diag\{\sigma_{11}^2, \ldots \sigma_{N_r N_a}^2\}$$ Equation 9-27

Given this matrix of linear equations, the solution method is as follows. First an initial guess of the beacon location is made, this is the trial origin point (θ',φ') given as azimuth and polar angle. This location is then used in the first order Taylor Series expansion to determine the matrix in Equation 9-20. The matrix Z can now be calculated using Equation 9-25 and the error "cost" term can be calculated using Equation 9-26. If the solution has not sufficiently converged the estimate of the true location is updated using Equation 9-28.

$$\theta_i' \leftarrow \theta_i' + \widehat{\delta_{\theta_i}}$$

$$\varphi_i' \leftarrow \varphi_i' + \widehat{\delta_{\varphi_i}}$$ Equation 9-28

The system then iterates until a sufficiently accurate estimate of the location is obtained.

9.9 Weighted Least Squares

Unlike a linear least squares algorithm which assumes that all variables have equal variance, the weighted least squares method makes no such assumption. For this for algorithm, a specific weight is calculated for each factor on every iteration, to arrive at the solution more quickly. There are many versions of weighted least squares algorithms, as there are many ways in which to weight the factors. The algorithm chosen for this research is given as follows [64].

$$\boldsymbol{X} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad \text{Equation 9-29}$$

$$\boldsymbol{A} = \begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) \\ 2(x_3 - x_2) & 2(y_3 - y_2) \\ \vdots & \vdots \\ 2(x_n - x_{n-1}) & 2(y_n - x_{n-1}) \end{bmatrix} \quad \text{Equation 9-30}$$

$$\boldsymbol{b} = \begin{bmatrix} ({x_2}^2 - {x_1}^2) + ({y_2}^2 - {y_1}^2) + \left({d_1}^2 - {d_2}^2\right) \\ ({x_3}^2 - {x_2}^2) + ({y_3}^2 - {y_2}^2) + \left({d_2}^2 - {d_3}^2\right) \\ \vdots \\ ({x_n}^2 - {x_{n-1}}^2) + ({y_n}^2 - {y_{n-1}}^2) + \left({d_{n-1}}^2 - {d_n}^2\right) \end{bmatrix} \quad \text{Equation 9-31}$$

Where

x_0 = x-coordinate of initial guess for location of beacon

y_0 = y-coordinate of initial guess for location of beacon

x_i = x-coordinate of satellite i

y_i = y-coordinate of satellite i

d_i = RF measured distance from satellite i to the beacon

n = number of satellites

The error for this system of equations is given by:

$$\boldsymbol{e} = \boldsymbol{A} * \boldsymbol{X} - \boldsymbol{b} \quad \text{Equation 9-32}$$

The weighting matrix adds a value to each of the elements of the error vector.

$$\boldsymbol{W} = diag(\boldsymbol{e}) \quad \text{Equation 9-33}$$

So that the resulting location vector for the weighted least squares will be determined by:

$$\boldsymbol{X} = (\boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{W}^{-1}\boldsymbol{b} \qquad \text{Equation 9-34}$$

For this process, arrays **A** and **b** can be calculated once, then arrays **e**, **W**, and **X** are found in sequence iteratively until the value **e** becomes quite small. At this point **X** will contain the computed location coordinates of the beacon.

9.10 Iterative Re-Weighted Least Squares

The iterative least squares method is from [57] also uses a reweighting process, but the weighting algorithm is different.

The process begins by initializing the weight Σ, with an identity matrix, and creating a matrix of satellite coordinates, B.

$$\Sigma = I^{n*n} \qquad \text{and} \qquad B = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \qquad \text{Equation 9-35 \& 9-36}$$

Where

x_i = x-coordinate of satellite i

y_i = y-coordinate of satellite i

n = number of satellites

Then, the following steps are iterated until σ is appropriately small.

$$\sigma = \sqrt{diag(\Sigma)} \qquad \text{Equation 9-37}$$

$$X_i = \frac{\Sigma_{j=1}^{n}\left({}^{B_j}/_{\sigma_j}\right)}{\Sigma_{j=1}^{n}\left({}^{1}/_{\sigma_j}\right)} \qquad \text{Equation 9-38}$$

$$A = \begin{bmatrix} 1 & 2(x_1 - x_0) & 2(y_1 - y_0) \\ 1 & 2(x_2 - x_0) & 2(y_2 - y_0) \\ \vdots & \vdots & \vdots \\ 1 & 2(x_n - x_0) & 2(y_n - y_0) \end{bmatrix}$$ Equation 9-39

$$b = \begin{bmatrix} (x_1 - x_0)^2 + (y_1 - y_0)^2 - {d_1}^2 \\ (x_2 - x_0)^2 + (y_2 - y_0)^2 - {d_2}^2 \\ \vdots \\ (x_n - x_0)^2 + (y_n - y_0)^2 - {d_n}^2 \end{bmatrix}$$ Equation 9-40

$$\beta = (A^T \Sigma^{-1} A\alpha)^{-1} A^T \Sigma^{-1} b$$ Equation 9-41

$$X_{i+1} = \beta_{n,2:3} + X_i$$ Equation 9-42

$$\sigma^2 = \left| \frac{1}{n} \sum_{i=1}^{n} b_i \right|$$ Equation 9-43

$${d_r}^2 = \begin{bmatrix} (x_1 - x_0)^2 + (y_1 - y_0)^2 \\ (x_2 - x_0)^2 + (y_2 - y_0)^2 \\ \vdots \\ (x_n - x_0)^2 + (y_n - y_0)^2 \end{bmatrix}$$ Equation 9-44

$$\Sigma = diag\left(4{d_r}^2 \sigma^2\right)$$ Equation 9-45

Once the algorithm converges, the coordinates of the beacon will be contained in the variable X.

9.11 Non-Linear Least Squares

Nonlinear Least Squares is a form of the least squares method that fits a nonlinear model of order “n” to a series of data of quantity “m” where m>n. The algorithm works

by first fitting a linear model and then refining that model for greater accuracy with higher parameters on successive iterations.

The Euclidian distance between the actual location and the predicted one can be given by:

$$d_{r_i} = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} \qquad \text{Equation 9-45}$$

$$F = d_{r_i} - d_i \qquad \text{Equation 9-46}$$

In a nonlinear system the derivatives are function of both the independent variables and the parameters.

$$J = \begin{bmatrix} \frac{\partial d_1}{\partial x} & \frac{\partial d_1}{\partial y} \\ \frac{\partial d_2}{\partial x} & \frac{\partial d_2}{\partial y} \\ \vdots & \vdots \\ \frac{\partial d_n}{\partial x} & \frac{\partial d_n}{\partial y} \end{bmatrix} \qquad \text{Equation 9- 47}$$

The gradient equations do not have a closed solution, and the parameters are refined iteratively. The minimum values occurs when the sums are zero. The iteration is carried out until a minimal error is found.

$$J^T J = \begin{bmatrix} \sum_{i=1}^{n} \frac{(x_0 - x_i)^2}{{d_{r_i}}^2} & \sum_{i=1}^{n} \frac{(x_0 - x_i)(y_0 - y_i)}{{d_{r_i}}^2} \\ \sum_{i=1}^{n} \frac{(x_0 - x_i)(y_0 - y_i)}{{d_{r_i}}^2} & \sum_{i=1}^{n} \frac{(y_0 - y_i)^2}{{d_{r_i}}^2} \end{bmatrix} \qquad \text{Equation 9-48}$$

9.12 Conclusion

These algorithms together cover a wide range of trilateration approaches. The MATLAB code for each can be found in Appendix C. The next chapter will discuss an

entirely new and developed trilateration algorithm, then all of the algorithms will be compared in Chapter 11.

CHAPTER 10. ANTENNA PATTERN METHOD OF LOCALIZATION

10.1 Overview

The goal of this work is to reduce system and signal error in localization methods. Multipath error is addressed with respect to individual measurements using machine learning to separate multipath information from the base signal. The other key source of error is the variation in signal strength caused by antenna radiation patterns known as pointing error. This error cannot be predicted from single distance measurements, as it depends on the relative orientations of the transmitting and receiving antennas. The goal of this section of the work was to create a new method of localization, which rather than ignoring error from non-isotropic antenna patterns, would leverages this information to assist with localization and ultimately reduce error.

In this chapter pointing error is addressed by fitting a simplified antenna pattern to the measured distance data to determine the beacon location. The method for deriving both the pattern and the way in which it was incorporated into a deep optimization algorithm is described. The iterative process described here addresses the significant issue of non-isotropic antenna radiation patterns, which is a challenge for the vast majority of trilateration methods [65] [66] [67] [17] [44] [29]. Similar to the RSSI Signature method described in Chapter 6 [3], the Antenna Pattern method utilizes additional information not incorporated into traditional methods, to reduce overall error.

10.2 Background

As discussed in the previous chapter, most trilateration algorithms essentially work by finding the location at the center of overlapping circles [66], [23], [46], [12], [68], [69]. In those algorithms, the circle surrounding the anchor is produced by using the location of the anchor as the center of the circle, and the distance from the anchor to the device is used as the radius of the circle, as shown in Figure 10-1.

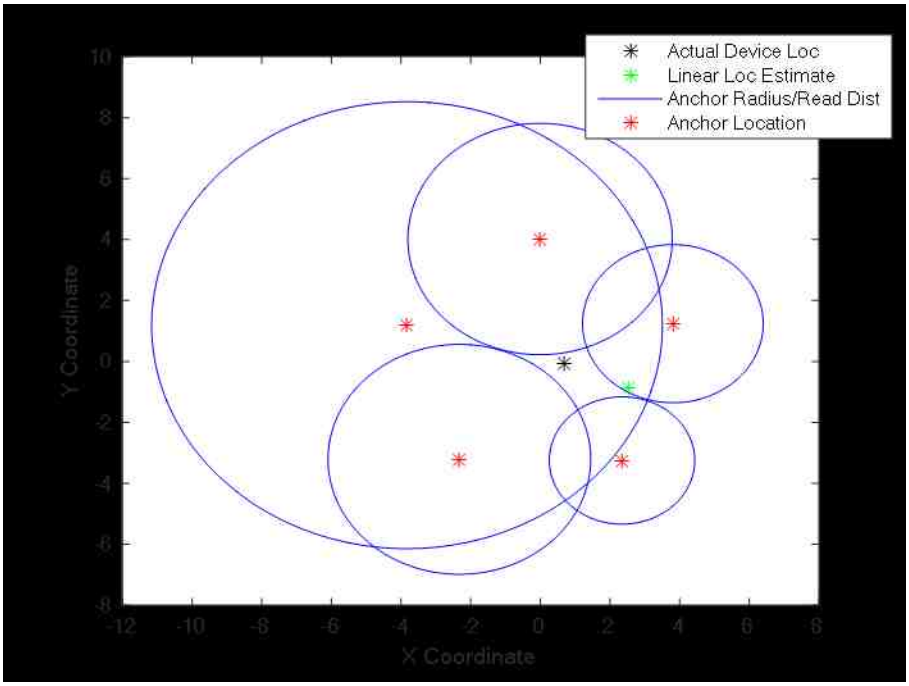


Figure 10-1: A graphic representation of a trilateration situation in which the readout distances given as blue lines surround the satellite (anchor) location. Standard methods will yield the green * as the location of a device, actually located at the black *.

Using the measured distances as the circle radii assumes all signals are returned at the same strength all the way around, and ignores the variation in returned signal strength due to the variation in antenna sensitivity at different relative positions [14]. Antenna to antenna relative alignment is usually unknown or difficult to determine, thus this antenna angle mismatch is stated as a known source of error in descriptions of most localization

methods [66] [70] [71] [72]. Antennas are sensitive in different directions based on their geometry. When an antenna is aligned with an incoming signal the antenna has a high efficiency when receiving that signal. When aligned (approximately) with a null in the antenna pattern the signal received is highly inefficient or not read at all. In other words, antennas radiate energy in a non-uniform pattern, creating areas of high signal intensity and nulls or zero signal intensity in different locations even at the same radius away.

The orientation of the antenna can make a large impact on the distance estimation, and therefore the localization error. Figures 10-2 and 10-3 below demonstrate how signal strength can be significantly impacted by antenna alignment, and in extreme cases the error from misalignment can exceed the true distance or even the entire read range.

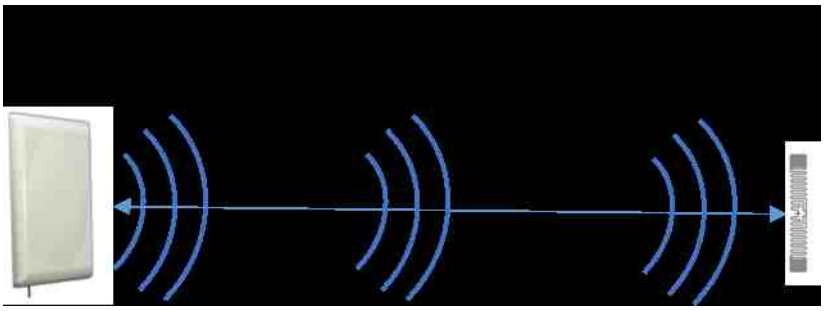


Figure 10-2: It is quite possible that an anchor and device have a large separation distance, but if they are directly facing each other, the pointing error will be low, and thus the calculated separation distance smaller.

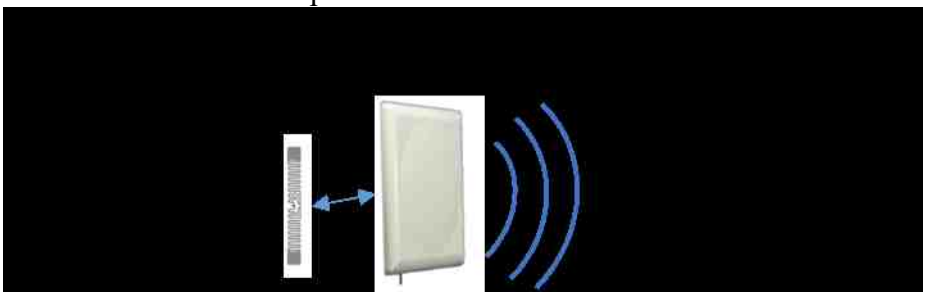


Figure 10-3: In contrast to the arrangement in Figure 10-2, the device and anchor could be right next to each other, but if one of them happens to be in the null of the other's antenna pattern, the calculated separation distance would be extremely large.

Antenna patterns are complex but are characterized by areas of high and low sensitivity. The research hypothesis is that a simplified model of an antenna pattern could be used to inform the localization algorithm and further reduce location error. For this hypothesis to be validated it would be necessary to leverage information from a set of measurements to inform the shape and orientation of the modeled pattern. Implementing this proposed solution would result in a new method of trilateration, which incorporates a generalized antenna pattern into the location optimization algorithm.

10.3 Generalized Approach

The proposed new trilateration method is an extension of methods currently in use for RF localization. The generalized approach to the new trilateration algorithm would however insert a key additional optimization step into standard trilateration. Measurements would be taken from multiple anchors and an approximate area of interest would be identified. These measurements could utilize the improved measurements described in Chapters 5 or 6, with error reduced by machine learning with the RSSI Signature, the RSSI-Informed Phase method, or any other distance estimation available. At this point an antenna sensitivity optimization algorithm based on an idealized antenna pattern would be used to further reduce error. As the distance measurements for the anchors will inherently be impacted by the antenna orientation of the tag or beacon being located, all measurements will contain information about that pattern. An optimization algorithm then rotates or scales an idealized antenna model until the model best fits the measurement data. The beacon location would be at the center of that antenna model, as the beacon antenna is creating the radiation pattern. By fitting an antenna pattern this algorithm allows for

measurement signals to be weighted uniquely as they may be near a null or the center of a lobe. This step should further reduce pointing error which would still remain after multipath error had been minimized. This method was proposed for both Bluetooth and RFID systems, having a representative antenna with various orientations representing the location where the tag or beacon is to be found.

10.3.1 Algorithm Process

1. Just as in a normal trilateration algorithm, circles are placed around each of the anchor locations to indicate the distance from the anchor to the device.
2. An initial estimate for the location of the device is found using a linear method. (Such as: Linear Least Squares, Centroid, or even simply using the average x and y coordinates of the anchors)
3. The basic antenna pattern is overlaid at the initial device location.
4. The distance between each of the circles, and the nearest point on the antenna pattern is found.
5. The gradients of each of these distances is calculated, looking at how the distance would change with variations in the antenna pattern orientation, size and location.
6. Gradient descent is used to minimize the sum of square errors, with distance between the antenna pattern and the nearest point on the anchor's circles serving as the error. The iterative process of gradient descent is used to find the most likely location of device. This minimum is the location where the antenna pattern best fits into the cluster of circles, and therefore the most likely location of the device.

10.3.2 Algorithm Process Explained

The center of the antenna pattern as shown in Figure 10-4 is used as the location of the device. Thus the x and y coordinates of the center of the antenna pattern are values to be optimized in the gradient descent algorithm, as well as the size and angle of the antenna pattern.

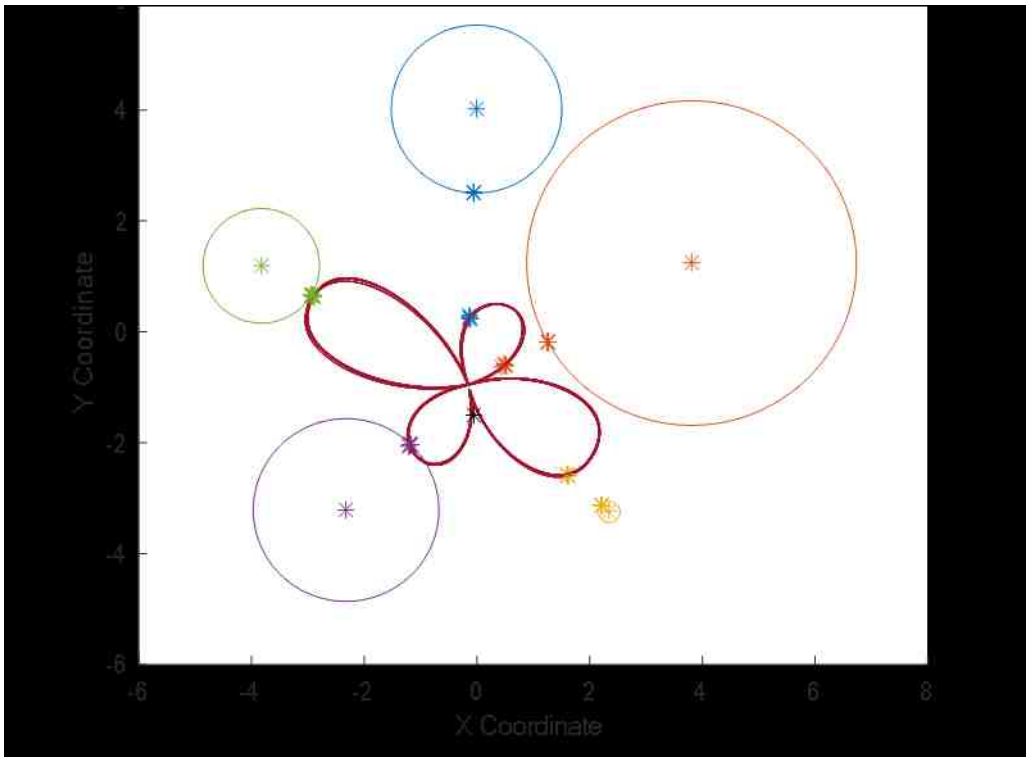


Figure 10-4: The dark red 4 lobed design represents the simplified antenna pattern which can be rotated or scaled to best fit the data being provided by the real world measurement.

The antenna pattern selected for this model takes into account antenna sensitivity due to its non-uniformity. Using large and small lobes in the model it is possible to emulate an antenna pattern which can then be rotated, expanded or contracted, and optimized to match the distance measurements. The antenna pattern is used in the localization/optimization algorithm (seen in Figure 10-4) is greatly simplified in comparison to the actual antenna pattern of the device. Most antenna patterns have many small lobes and nulls. These features in an optimization algorithm would result in a significant number of local minima, making it hard to find the most likely global minimum. The antenna pattern created and used for this algorithm was a reduced version of the original BLE antenna pattern and was characterized by a combination of a large front lobe

and a slightly smaller back lobe, with two smaller side lobes, as described in greater detail in this chapter.

One final addition to the algorithm was to have the initial orientation of the antenna pattern in the optimization algorithm be with the smaller of the main two lobes (back lobe) facing the circle with the smallest radius. A circle with smaller radii means a shorter separation distance, and shorter separation distances are frequently more accurate. Thus by having the slightly shorter lobe in the direction of the smaller circle, this forces the sum of square errors to bring the antenna pattern closer to the smaller circle in order to compensate for the shortened lobe. This is essentially a way of weighting the smaller circle, which has a higher confidence. This effect can be seen in Figure 10-4.

10.3.3 Selection of Parameters

The Antenna Pattern method is effective because it's a trilateration method that accounts for the non-uniformity of the beacon antenna. This is accomplished by using the measured distance from the anchors to the beacon, not to resolve to a single point, but rather to arrange the lobbed antenna pattern such that its edges correlate with these RF measured distances. Given the arduous nature of nonlinear point trilateration, the antenna trilateration algorithm begins with the same information as other algorithms, and adds the optimization of a multi degree of freedom beacon antenna representation. The antenna trilateration uses three key features. The first is the additional optimization parameters. In addition to an x and y coordinate, the Antenna Pattern method includes scaling and orientation of the antenna shape. The second component is the increased difficulty in

calculating distance. For point trilateration, distance is found between two points; the position of the anchor and the supposed position of the beacon. In contrast, the Antenna Pattern method must find the two points which are closest to each other on a circle and a complex geometrical shape, and then use those to calculate distance. Finally, the derivative or change in this distance must be calculated for any change in each of the four optimization parameters. These gradients serve as a key component of the optimization process.

10.3.4 Gradient Descent

This method is unique because of its associated antenna pattern optimization algorithm. Most trilateration algorithms use a form of least squares to find the most likely location. However, this is generally a linear optimization, and due the fact that the distance equation or Pythagorean Theorem is a nonlinear equation, gradient descent (which is a nonlinear optimization algorithm) was selected. Additionally, gradient descent is an extremely versatile algorithm which can be used for any differentiable equation. The key issue of using gradient descent, like nearly all nonlinear optimization, is it won't always find the global optimum, but rather the local optimum. To mitigate this issue, Linear Least Squares is used to find an initial location, then the gradient descent algorithm is applied to find the nonlinear optimum.

10.4 Creation and Validation of the Antenna Model

The algorithm was created in two phases. In the first phase, a test algorithm based on a numerical model was created. After using this model to establish how the optimization algorithm would need to interact with an antenna model, a second final model was created to meet the computational speed requirement.

10.4.1 Numerical vs. Equation Based

Initially the Antenna Pattern method was completed numerically, using a set of points for the antenna pattern, calculating the distance between the anchor circle and beacon antenna pattern with interpolation methods, and a numerical gradient. This work was used to verify the concept of an antenna model for improved error reduction. The numerical model was more quickly implemented in code and allowed for a period of experimentation to determine general effectiveness and highlight areas for algorithm improvement. However, these numerical methods, took approximately 2.5 seconds per case. Given that this algorithm would be used for both localization and tracking, the computation time must be minimized, and 2.5 seconds would not meet the goal of rapid position determination.

To increase the processing speed, the algorithm was re-formulated using a theoretical equation based method. For this technique, the antenna pattern had to be described in a single continuous equation.

10.4.2 Selection of Antenna Pattern

When using an antenna pattern as part of the optimization algorithm, the goal is not to use an exact representation of the actual antenna pattern, but rather a reasonable alternative that would allow for a few peaks and nulls without creating too many local minima for the optimization algorithm. Nevertheless, the process of identifying an appropriate representation began by measuring the radiation pattern itself. The radiation pattern of the Bluetooth beacon was captured in a fully anechoic chamber and is shown below. To create the pattern shown in Figure 10-5, the beacon broadcasted from the center

of the chamber, while slowly being rotated on a revolving platform. The signal was received on the other side of the chamber with a stationary antenna.

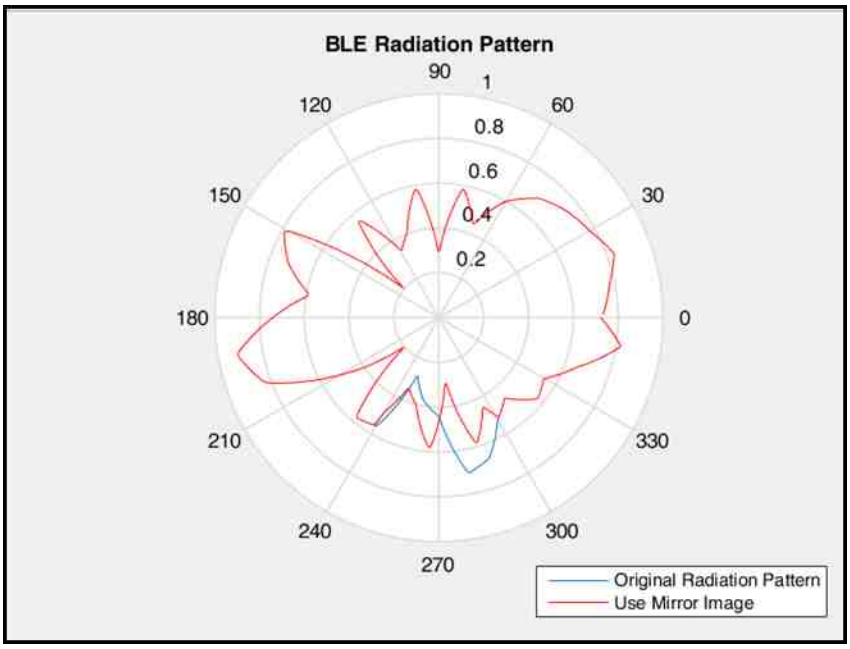


Figure 10-5: Bluetooth Antenna Pattern – this pattern was found experimentally by measuring the transmitted signal in a semi-anechoic chamber.

In Figure 10-5 there are both blue and red lines. These are because the BLE beacon had a wire attached to one side (approximately at 270° on in Figure 10-5). Wires impact the measured radiation pattern, therefore the mirror image from 240° to 300° was used to create a more likely true antenna pattern. It is important to note that, measuring the exact pattern is unnecessary however because only a very rough approximation will be used in the final analysis.

A measured antenna pattern is not effective as a tool for optimization because of the complexity of the radiance, which is both difficult to differentiate and susceptible to having an optimization algorithm stop at one of many local minima. Although the measured antenna pattern did not directly provide an effective model, it did inform a basis

for a more general model. Several variations of this pattern were examined to find the best option.

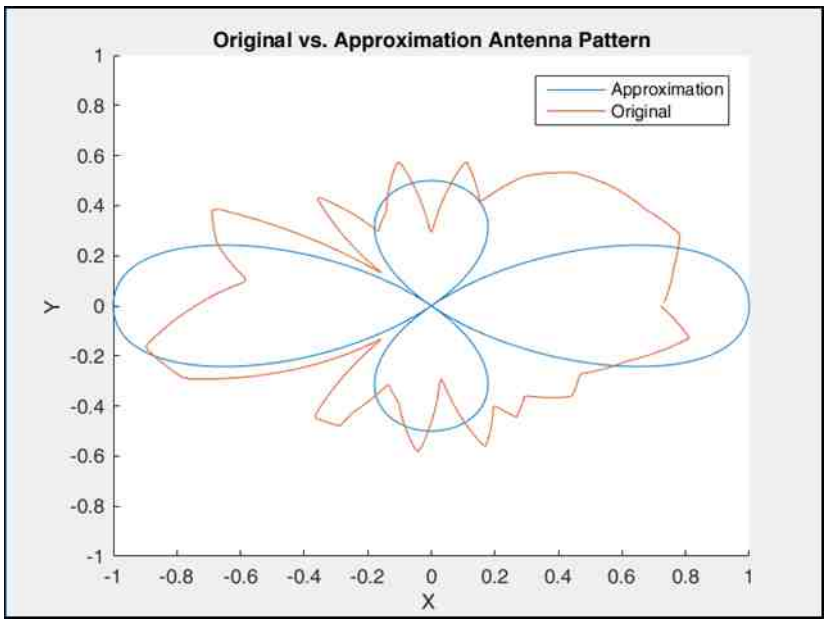


Figure 10-6: Actual vs. Approximate Antenna Pattern. The approximate pattern utilizes two large and two small lobes to simulate the general structure of the measured antenna pattern.

The approximate antenna pattern decided upon, shown in Figure 10-6, was chosen for several reasons. The first reason is it has distinct front and back lobes, which relate to the main lobes of the actual beacon. These lobes are smooth, and therefore easily differentiable. Two smaller side lobes create some sensitivity in the model for the multiple side lobe areas, but creating fewer local minima for the optimization. The selected pattern is a variation of a radiation pattern from two isotropic point sources [14]. While the antenna pattern method was tested against both BLE and RFID data, there was not an expectation that RFID data would show an improvement with this method. All RFID data taken for the analysis was taken with the tag vertically (linearly) polarized and therefore on the horizontal plane was not directional but appeared more isentropic.

10.4.3 Converting the antenna pattern into a differentiable equation

Creating an equation for the representative antenna pattern began by using the equation for two isotropic point sources one wavelength apart, given by Equation 10-1 and shown in Figure 10-7:

$$r = \cos\left(\pi cos(\theta)\right) \qquad \text{Equation 10-1}$$

This equation by itself created much too wide of a pattern, therefore it was reduced in the vertical direction using a multiplication factor of 0.5 in the y direction. This was accomplished by first expressing in Cartesian coordinates, resulting in Equations 10-2 and 10-3. The impact on the shape of the antenna pattern can also be seen in Figure 10-7.

$$x = r\cos(\theta) \qquad \text{Equation 10-2}$$

$$y = 0.5r\sin(\theta) \qquad \text{Equation 10-3}$$

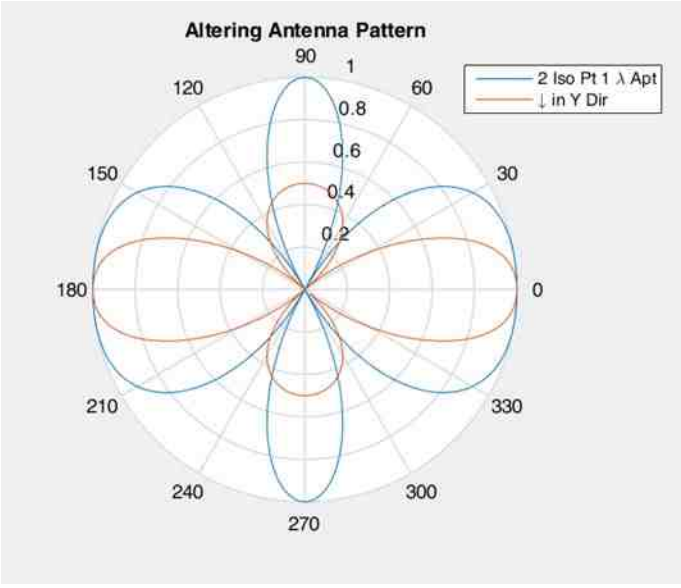


Figure 10-7: Reducing the antenna pattern in the vertical direction

Then reverting back and expressing the equation in polar coordinates results in Equations 10-4 and 10-5.

$$r(\theta) = \sqrt{\left(cos\theta cos(\pi cos\theta)\right)^2 + \left(0.5 sin\theta cos(\pi cos\theta)\right)^2}(\theta) \qquad \text{Equation 10-4}$$

$$\varphi(\theta) = \tan^{-1}\left(\frac{0.5 sin\theta}{cos\theta}\right) \qquad \text{Equation 10-5}$$

The necessary form for this model is r as a function of φ, instead of r and φ both as functions of θ. To reduce and combine terms, Equation 10-5 was transformed into Equation 10-6, then 10-7.

$$\tan\varphi = 0.5\tan\theta \qquad \text{Equation 10-6}$$

$$\theta = \tan^{-1}(2\tan\varphi) \qquad \text{Equation 10-7}$$

With θ in terms of φ, a single equation can be written with r as a function of φ.

$$\mathrm{r} = (\cos(\tan^{-1}(2\tan\varphi)) * \cos(\pi\cos(\tan^{-1}(2\tan\varphi)))^2 + (0.5\sin(\tan^{-1}(2\tan\varphi)) * \cos(\pi\cos(\tan^{-1}(2\tan\varphi))))^2)^{0.5}$$

Equation 10-8

The approximate antenna pattern shown in Figures 10-6 and 10-7, have equal length front and back lobes. The ability to separate the size of the font and back lobes, and have

one of the main lobes slightly smaller was considered superior as it relates to the true antenna pattern having asymmetrical front and back lobes. The challenge with shortening one lobe, was again to create a single differentiable (continuous) equation.

Based on a recommendation from Dr. Brandon Kemp, Arkansas State University, a Fourier Series was chosen as a means to reduce one of the main lobes. The lobe chosen was between -45° and 45°, therefore a Fourier Series was constructed which had a value of one for all values except those between -45° and 45° (or -π/4 and π/4), as illustrated in Figure 10-8 and given in Equation 10-9.

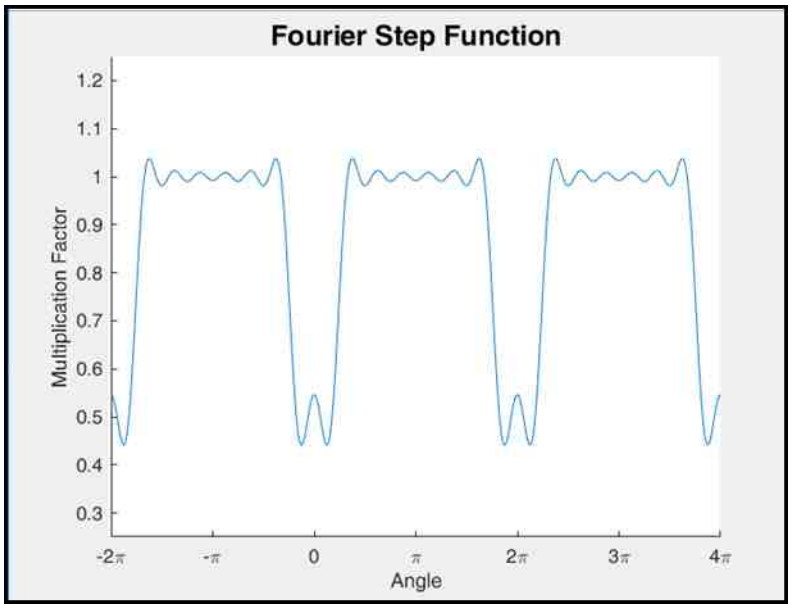


Figure 10-8: Fourier Step function for transforming antenna pattern

$$\text{Fourier Step} = \left(\frac{3+h}{4}\right) + \left(\frac{2h-2}{\pi\sqrt{2}}\right)\cos\varphi + \left(\frac{2h-2}{\pi}\right)\cos 2\varphi + \left(\frac{2h-2}{3\pi\sqrt{2}}\right)\cos 3\varphi + \left(\frac{-2h+2}{5\pi\sqrt{2}}\right)\cos 5\varphi + \left(\frac{-2h+2}{6\pi}\right)\cos 6\varphi + \left(\frac{-2h+2}{7\pi\sqrt{2}}\right)\cos 7\varphi \qquad \text{Equation 10-9}$$

Where:

h = Reduction factor of back lobe

By multiplying Equations 10-8 and 10-9, the back lobe of the antenna pattern can be reduced, as shown in Figure 10-9, and given by Equation 10-10.

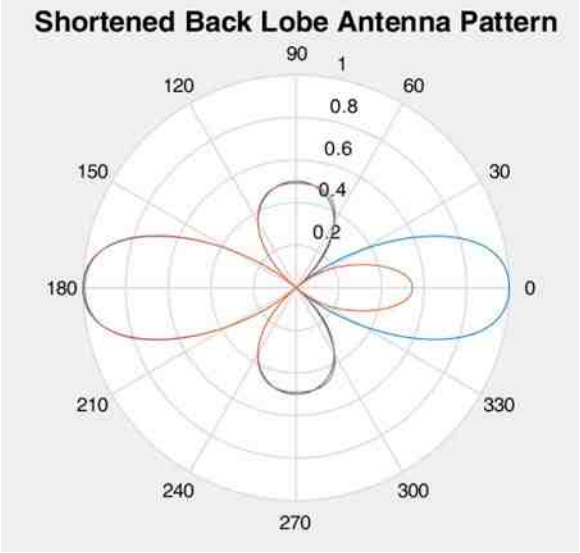


Figure 10-9: Shortened Back Lobe of Antenna Pattern.

$$\mathrm{r}(\varphi) = \left(\begin{pmatrix} cos(tan^{-1}(2\,tan\,\varphi)) * \\ cos\left(\pi cos(tan^{-1}(2\,tan\,\varphi))\right) \end{pmatrix}^2 + \begin{pmatrix} 0.5sin(tan^{-1}(2\,tan\,\varphi)) * \\ cos\left(\pi cos(tan^{-1}(2\,tan\,\varphi))\right) \end{pmatrix}^2 \right)^{0.5} * \left[\left(\frac{3+h}{4}\right) + \left(\frac{2h-2}{\pi\sqrt{2}}\right) cos\,\varphi + \left(\frac{2h-2}{\pi}\right) cos\,2\varphi + \left(\frac{2h-2}{3\pi\sqrt{2}}\right) cos\,3\varphi + \left(\frac{-2h+2}{5\pi\sqrt{2}}\right) cos\,5\varphi + \left(\frac{-2h+2}{6\pi}\right) cos\,6\varphi + \left(\frac{-2h+2}{7\pi\sqrt{2}}\right) cos\,7\varphi \right]$$

Equation 10-10

This equation for the approximate antenna pattern was then ready to be applied to the optimization algorithm to increase the accuracy by accounting for the non-isotropic pattern of the actual beacon antenna.

10.5 Formulation of the Algorithm

With the antenna pattern created, the location optimization begins with formulating the gradient descent algorithm for this scenario. The equations that need to be optimized represent the distance from the radii of each anchor to the scalable and rotatable theoretical sensitivity model, which is a nonlinear set of equations. Again, gradient descent was selected for its robustness and versatility to handle this type of equation set. The first step for gradient descent is to create a vector containing the optimization parameters, in this case they are: size of antenna pattern, angle of antenna pattern, x-coordinate, y-coordinate. The size of the antenna pattern allowed the pattern to be scaled to fit the gap occurring between the different measurement circles. The angle allowed for rotation to accommodate orientation variations, and the x and y coordinate moved the pattern and therefore located the beacon or tag.

$$X = \begin{bmatrix} S \\ A \\ x_{Tag} \\ y_{Tag} \end{bmatrix} \qquad \text{Equation 10-11}$$

Where:

X = Vector of optimization parameters

S = (Size) Multiplication factor for the radius of the antenna pattern

A = (Angle) Additive angle to the antenna pattern

x_{Tag} = X-coordinate of tag/beacon center

y_{Tag} = Y-coordinate of tag/beacon center

The vector G, is created to contain the functions to be optimized. In this case these are the distances from the antenna pattern to the radius of each anchor.

$$G = \begin{bmatrix} d_1(S, A, x_{Tag}, y_{Tag}) \\ d_2(S, A, x_{Tag}, y_{Tag}) \\ \vdots \\ d_n(S, A, x_{Tag}, y_{Tag}) \end{bmatrix} \quad \text{Equation 10-12}$$

Where:

d = Distance from anchor radius to the antenna pattern

n = Number of anchors

G = Vector of functions

The distance between the antenna pattern and the radius of the anchor is essentially the error, and therefore must be minimized, as illustrated in Figures 10-10 and 10-11.

The cost function is then given by [73]:

$$F(X) = \frac{1}{n-1} G^T G \quad \text{Equation 10-13}$$

Where:

F(X) = Cost function

The gradient of the cost function is found by using the Jacobian of vector G.

$$\nabla F(X^{(0)}) = J_G\left(X^{(0)}\right)^T G\left(X^{(0)}\right) \qquad \text{Equation 10-14}$$

$$J_G = \begin{bmatrix} \frac{\partial d_1}{\partial S} & \frac{\partial d_1}{\partial A} & \frac{\partial d_1}{\partial x_{Tag}} & \frac{\partial d_1}{\partial y_{Tag}} \\ \frac{\partial d_2}{\partial S} & \frac{\partial d_2}{\partial A} & \frac{\partial d_2}{\partial x_{Tag}} & \frac{\partial d_2}{\partial y_{Tag}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial d_n}{\partial S} & \frac{\partial d_n}{\partial A} & \frac{\partial d_n}{\partial x_{Tag}} & \frac{\partial d_2}{\partial y_{Tag}} \end{bmatrix} \qquad \text{Equation 10-15}$$

Where:

J_G = Jacobian of set of equations G with respect to X(S, A, x_{Tag}, y_{Tag})

Thus, with gradient descent the Equation 16 can be iterated to converge on the best parameter values, including the beacon location.

$$X^{(1)} = X^{(0)} - \gamma \nabla F\left(X^{(0)}\right) \qquad \text{Equation 10-16}$$

Where:

γ = Regularization parameter

The resulting gap between the antenna pattern and the anchor is a measure of how good the fit is. This distance is minimized to create an optimum antenna placement.

$$d_n = \sqrt{\left(x_{Ant_n} - T_x\right)^2 + \left(y_{Ant_n} - T_y\right)^2} - R_n \qquad \text{Equation 10-17}$$

Where:

R_n = Radius of anchor n

d_n = Resulting gap between antenna pattern and anchor n

x_{Ant_n} = X-coordinate of anchor n (center of anchor circle)

y_{Ant_n} = Y-coordinate of anchor n (center of anchor circle)

T_x = X-coordinate of the antenna pattern

T_y = Y-coordinate of the antenna pattern

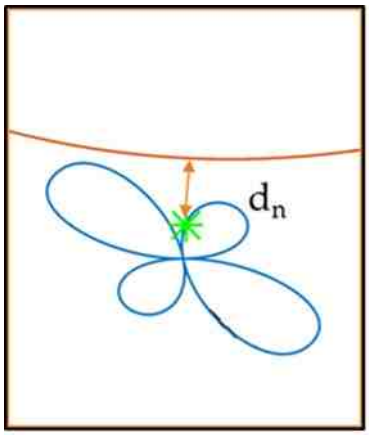


Figure 10-10: The distance d_n is given by the distance between the antenna pattern and circle surrounding the anchor.

The circle around the anchor, shown as an orange curve in Figure 10-10, is the representation of the anchor to beacon distance estimation, calculated using one of several

methods discussed in previous chapters. The variables T_x and T_y are a means of representing a point on the antenna pattern, as can be seen in Figure 10-11.

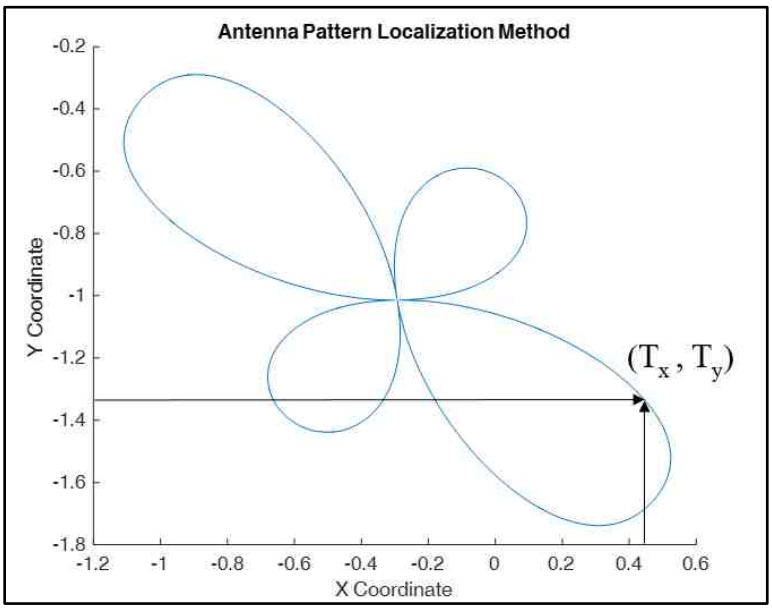


Figure 10-11: The variables T_x and T_y are the x and y coordinates on the antenna pattern

To calculate the Jacobian, the derivative of distance d_n must be found with respect to each parameter (S, A, x_{Tag}, y_{Tag}).

$$\frac{\partial d_n}{\partial S} = \frac{(x_{Ant_n} - T_x)\frac{-\partial T_x}{\partial S} + (y_{Ant_n} - T_y)\frac{-\partial T_y}{\partial S}}{\sqrt{(x_{Ant_n} - T_x)^2 + (y_{Ant_n} - T_y)^2}} \quad \text{Equation 10-18}$$

$$\frac{\partial d_n}{\partial A} = \frac{(x_{Ant_n} - T_x)\frac{-\partial T_x}{\partial A} + (y_{Ant_n} - T_y)\frac{-\partial T_y}{\partial A}}{\sqrt{(x_{Ant_n} - T_x)^2 + (y_{Ant_n} - T_y)^2}} \quad \text{Equation 10-19}$$

$$\frac{\partial d_n}{\partial x_{Tag}} = \frac{(x_{Ant_n} - T_x)\frac{-\partial T_x}{\partial x_{Tag}} + (y_{Ant_n} - T_y)\frac{-\partial T_y}{\partial x_{Tag}}}{\sqrt{(x_{Ant_n} - T_x)^2 + (y_{Ant_n} - T_y)^2}} \quad \text{Equation 10-20}$$

$$\frac{\partial d_n}{\partial y_{Tag}} = \frac{(x_{Ant_n} - T_x)\frac{-\partial T_x}{\partial y_{Tag}} + (y_{Ant_n} - T_y)\frac{-\partial T_y}{\partial y_{Tag}}}{\sqrt{(x_{Ant_n} - T_x)^2 + (y_{Ant_n} - T_y)^2}}$$ Equation 10-21

The next step is to represent T_x and T_y in terms of the optimization parameters (S, A, x_{Tag}, y_{Tag}).

$$T_x = S * r(\varphi) * \cos(\varphi + A) + x_{Tag}$$ Equation 10-22

$$T_y = S * r(\varphi) * \sin(\varphi + A) + y_{Tag}$$ Equation 10-23

Where:

φ = Orientation angle of beacon

The variables T_x and T_y must also be differentiated with respect to each of the optimization parameters to feed into the Jacobian, J.

$$\frac{\partial T_x}{\partial S} = r\,cos(\varphi + A)$$ Equation 10-24

$$\frac{\partial T_x}{\partial A} = S\,cos(\varphi + A)\frac{\partial r}{\partial \varphi(A)} - Sr\,sin(\varphi + A)\left(\frac{\partial \varphi}{\partial A} + 1\right)$$ Equation 10-25

$$\frac{\partial T_x}{\partial x_{Tag}} = S\,cos(\varphi + A)\frac{\partial r}{\partial \varphi(x_{Tag})} - Sr\,sin(\varphi + A)\frac{\partial \varphi}{\partial x_{Tag}} + 1$$ Equation 10-26

$$\frac{\partial T_x}{\partial y_{Tag}} = S\,cos(\varphi + A)\frac{\partial r}{\partial \varphi(y_{Tag})} - Sr\,sin(\varphi + A)\frac{\partial \varphi}{\partial y_{Tag}}$$ Equation 10-27

$$\frac{\partial T_y}{\partial S} = r\,sin(\varphi + A)$$ Equation 10-28

$$\frac{\partial T_y}{\partial A} = S\,sin(\varphi + A)\frac{\partial r}{\partial \varphi(A)} + Sr\,cos(\varphi + A)\left(\frac{\partial \varphi}{\partial A} + 1\right) \qquad \text{Equation 10-29}$$

$$\frac{\partial T_y}{\partial x_{Tag}} = S\,sin(\varphi + A)\frac{\partial r}{\partial \varphi(x_{Tag})} + Sr\,cos(\varphi + A)\frac{\partial \varphi}{\partial x_{Tag}} \qquad \text{Equation 10-30}$$

$$\frac{\partial T_y}{\partial y_{Tag}} = S\,sin(\varphi + A)\frac{\partial r}{\partial \varphi(y_{Tag})} + Sr\,cos(\varphi + A)\frac{\partial \varphi}{\partial y_{Tag}} + 1 \qquad \text{Equation 10-31}$$

The next step is to find the angle on the antenna pattern φ, and how it relates to the overall orientation of the antenna pattern. This relationship between the overall orientation of the antenna pattern, and the angle on the antenna pattern based on the relative position of the anchor, as can be visualized in Figure 10-12.

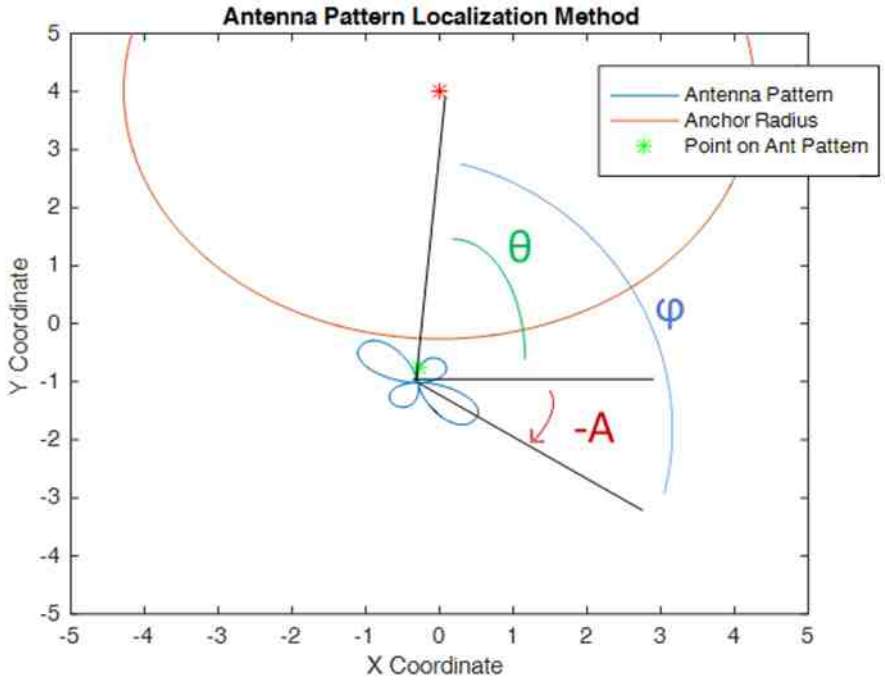


Figure 10-12: Angle φ is from the center of the main lobe to the line between the center of the antenna pattern to the center of the anchor.

In Figure 10-12, Angle θ is the angle from the horizontal to the line between the center of the antenna pattern and the center of the anchor. "A" is the angle between the horizontal and the main lobe of the antenna pattern. In Figure 10-12 it can be seen that the angle A controls the orientation of the antenna pattern, as it is the angle between the horizontal axis and the main lobe of the antenna pattern. The angle φ is also dependent upon the orientation of the antenna pattern, as it is the angle between the main lobe of the antenna pattern and the line between the center of the antenna pattern and the center of the anchor. Angle θ relates the other two angles, as it is the difference between φ and A. Using these relationships, the equation of φ can be formulated as follows:

$$\varphi = \theta - A = \tan^{-1}\left(\frac{y_{Ant}-y_{Tag}}{x_{Ant}-x_{Tag}}\right) - A \qquad \text{Equation 10-32}$$

Where:

θ = Angle between the horizontal and the line between the center of the antenna pattern and the center of the beacon.

With an equation for φ in terms of the optimization parameters, the next step is to differentiate in terms of each optimization parameter.

$$\frac{\partial \varphi}{\partial S} = 0 \qquad \text{Equation 10-33}$$

$$\frac{\partial \varphi}{\partial A} = -1 \qquad \text{Equation 10-34}$$

$$\frac{\partial \varphi}{\partial x_{Tag}} = \frac{y_{Ant}-y_{Tag}}{(y_{Ant}-y_{Tag})^2+(x_{Ant}-x_{Tag})^2} \qquad \text{Equation 10-35}$$

$$\frac{\partial \varphi}{\partial y_{Tag}} = \frac{x_{Tag} - x_{Ant}}{(y_{Ant} - y_{Tag})^2 + (x_{Ant} - x_{Tag})^2}$$ Equation 10-36

The next step is to incorporate the antenna pattern from Equation 10-10. Equations 10-37 through 10- 41 represent Equation 10-10 in a simpler form for differentiation.

$$r = \sqrt{(r_1 * r_2)^2 + (r_1 * r_3)^2} * r_4$$ Equation 10-37

Where:

$$r_1 = \cos(\pi \cos(\tan^{-1}(2 \tan \varphi)))$$ Equation 10-38

$$r_2 = \cos(\tan^{-1}(2 \tan \varphi))$$ Equation 10-39

$$r_3 = 0.5 \sin(\tan^{-1}(2 \tan \varphi))$$ Equation 10-40

$$r_4 = \text{Equation 9 (Fourier Step Function)}$$ Equation 10-41

The antenna pattern is then differentiated with respect to each of the optimization parameters, the equations of which can be viewed in the MATLAB code in Appendix C, section C.11. The resulting equations are combined to give the derivative of the distance d_n with respect to S, A, x_{Tag}, and y_{Tag}.

10.6 Conclusion

One of the most novel discoveries to come from the research was the use of the fitting of a simplified antenna pattern to the distance data in order to determine a location. This multistep process addressed a significant issue with trilateration methods; antenna directionality. Antenna directionality cannot be predicted in the field when the measurements are being generated. However, like the RSSI Signature, the data that exists in the distance measurements can be used as additional information about the actual

environment at the time of measurement, and this in turn can be used to reduce error overall.

The Antenna Pattern method of optimization is based upon the hypothesis that localization can be improved if antennas aren't assumed to be perfectly isotropic, as is the case in most algorithms. This method uses an approximation of the beacon antenna pattern, by incorporating a large front lobe, a slightly smaller back lobe, and two small side lobes. By using this pattern, the optimization algorithm can allow for the fact that a beacon may be close to an anchor, yet still not receive a strong signal, as it may be in the null of the beacon antenna. Similarly, a beacon could be a great distance away from an anchor, but have a strong received signal strength, because the main lobe could be directly facing the anchor.

In the next chapter, this Antenna Pattern method will be compared to several others to determine its effectiveness for localization and the overall best combination of technology and localization algorithm.

CHAPTER 11. EXPERIMENTAL COMPARISON OF TRILATERATION

11.1 Overview

Localization of an unknown tag or beacon using trilateration requires both a series of distance measurement addressed in Chapters 3-7, and the algorithms used to move from distance measurements to actual location information described in Chapters 2, 9 & 10. The goal of this work was to make contributions in both of these areas. Distance measurement error has been experimentally reduced using 1) signal combinations (RSSI-Informed Phase Chapter 5) and 2) information from the frequency response of the system referred to as the RSSI Signature in Chapter 6. Known trilateration methods have been enhanced with improved weighting models in the case of the Centroid method, the derived method of the Hyperbolic LOP described in Chapter 9 and the addition of an Antenna Pattern method in Chapter 10. The goal of this chapter was to theoretically compare the performance of all possible combinations of these distance measurements and localization algorithms to understand their strengths and weaknesses.

While it is difficult to get a good comparison of actual error in meters that should be expected from trilateration of BLE device networks, a few experiments described in the literature, compare a proposed unique solution to "standard" trilateration methods using Bluetooth devices, and the location error reported ranged from 3 – 3.75 meters [74] [75]. Papers where localization are systems set-up with unlimited anchor nodes usually have a location accuracy of about 0.7 meters [60] [76] [77], but that higher accuracy comes with a large infrastructure and maintenance overhead as was specifically to be avoided in the problem statement of this work. The methods explored in this analysis all require minimum

infrastructure and maintenance, and as such the distance errors found in this work of 1.5 to 2 meters on average, as will be shown later in this chapter, compare favorably.

In Chapter 9 a survey of current trilateration methods was presented. These current methods use a variety of mathematical techniques to deal with issues such as error in distance measurements, but negate the possibility of a closed form solution. The benefit of a closed form solution is the fast computation speeds, whereas algorithms which account for the nonlinearity of the problem will be iterative. Chapter 10 proposed a new method of trilateration, which accounted for the non-isotropic antenna pattern of the beacon, but is also iterative. This new method uses a simplified model of an antenna radiation pattern to compensate for antenna error in the distance measurements.

In this chapter, a thorough examination the different localization algorithms methods surveyed in Chapter 9 and 10 are presented. This analysis includes the impact of improvements to distance estimates and therefore input data, including improvements to distance estimates due to reduced multipath error (the RSSI Signature in Chapter 6), as well as the new Antenna Pattern method proposed in Chapter 10. The resulting localization accuracy for all combinations of methods for data taken in a controlled environment are simulated using empirical data. This represents 525 different combinations of technology and algorithms. At each level of technology/algorithm chose, the results are presented to show how each impacts the final trilateration accuracy. This chapter then explores the results and discusses which technologies or algorithms are the best choice for different applications or requirements.

11.2 Comparison of Localization Algorithms

Information available describing localization algorithms are reported for different environments and different constraints. There is currently no published information which effectively compares methodologies in comparable situations with similar metrics of accuracy or computational time. It was one of the goals of this work to create such a data set and resulting analysis, so that given the same settings each algorithm could be compared against the others do understand the tradeoffs of potential application design decisions on system performance.

In order to compare the algorithms using empirical data, a large amount of data was taken for both RFID and BLE systems where there was one beacon or tag and many readers surrounding it. Localization systems will function by getting multiple distance measurements from multiple readers at the same time from the same beacon. This reinforces the issue of antenna radiance patterns as the beacon will, by the constraints of the application, be facing different readers differently when a measurement is taken. For the RFID system, four circular polarized antennas were connected to a single ThingMagic M6e radio, and positioned one meter above the ground plane. Figure 11-1 is a diagram of the experimental setup. A vertically polarized Squiglette tag was placed one meter above the ground plane and at numerous locations within the area, and each location was measured in terms of a constant Cartesian coordinate system. It was moved incrementally using a stepper motor/pulley system and location was recorded and calculated using an encoder attached to the motor. Both RSSI and phase values were collected from each antenna for all 50 hop frequencies (or all frequencies which were readable). As mentioned in earlier chapters, interference or low signal response can cause a null reading at a

particular frequency. This data was then collected and organized for use in each of the possible localization algorithms.

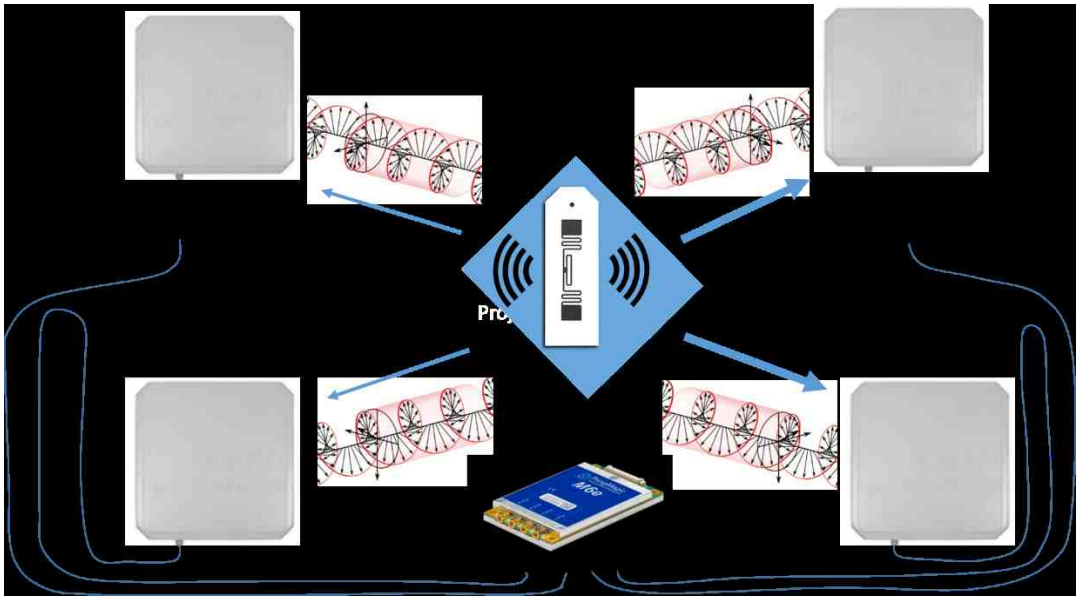


Figure 11-1: RFID Experimental setup showing 4 circularly polarized antennas reading one Squiglette RFID tag.

The BLE setup had 5 transponders surrounding a center beacon. All of the BLE device locations were accounted for using a common coordinate system. The beacon was moved incrementally using a stepper motor and pulley system, and the position was communicated using an encoder. Due to their physical geometry the BLE devices were placed horizontally polarized and similarly 1m above the ground plane. RSSI values were collected from the transmission of the center beacon to each of the surrounding transponders, at each of the hop frequencies. The diagram in Figure 11-2 shows how the system slowly pulled the target center beacon through the room allowing for a gradual change in orientation and distance for all transponders simultaneously.

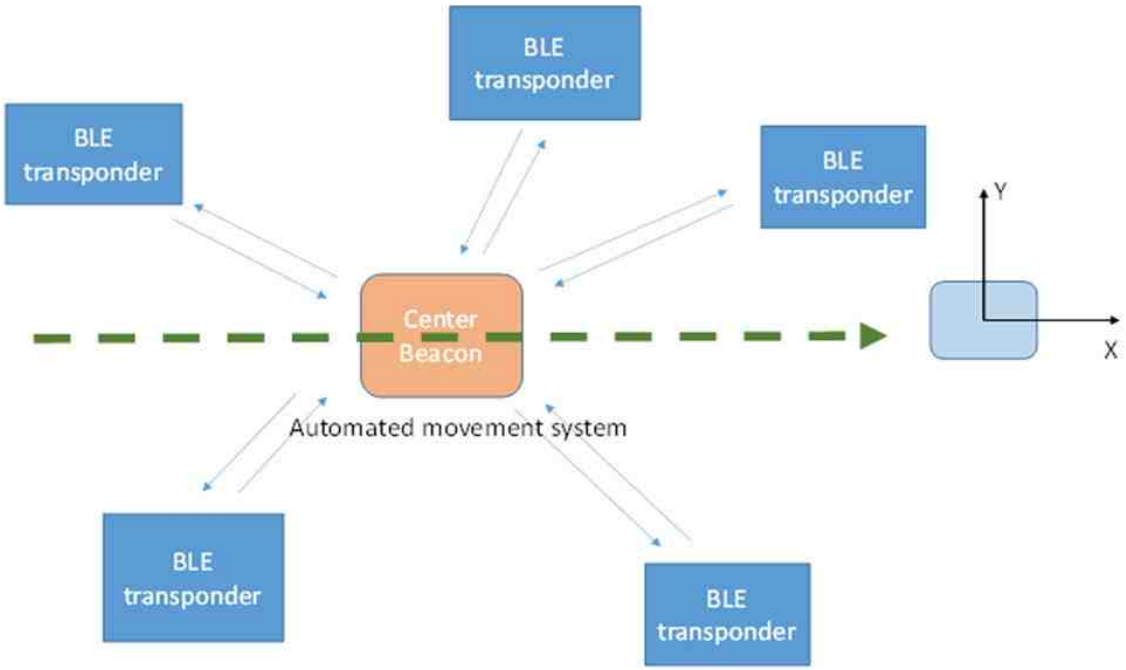


Figure 11-2: A diagram of the interaction between Bluetooth devices in a network.

This experiment required the development of an automated pulley system which had the dual benefit of allowing the data to be collected over many hours' time, overnight when the semi-anechoic chamber was available, and also allowed it to be collected with no people in the chamber which improved data quality.

Figure 11-3 is a photograph of the actual automated test set-up of the BLE localization in the 10 meter semi-anechoic chamber at Lexmark. Material selection for the equipment needed to be RF transparent, light weight and inexpensive. The data acquisition program was developed to handle the multiple beacon inputs and the computational time delays. The program is running on the laptop in the foreground of the picture.

Figure 11-3: BLE localization data collection in 10 meter anechoic chamber.

Once the data was collected, the trilateration algorithms and distance estimation methods were compared for both the RFID and BLE systems. The data was applied to each algorithm and combination of algorithms using MATLAB and the results were compared to the "known" location in the coordinate system. It should be noted that there is some error associated with the "known" position as described in Chapter 12, however this error was the same for all systems and should therefore not significantly impact the analysis.

The distance estimation methods which fed the different algorithms are shown in Table 11-1 below.

Table 11-1: The different distance measurement methods used in this analysis and a brief description of each.

Name	Description	Chapter
RSSI	Distance is calculated from an empirical RSSI vs. distance relationship	2-4
RSSI Signature	Distance is calculated initially using RSSI then improved with the error estimation of the RSSI signature.	5-8
Phase	Distance is calculated by observing changes in phase angle with changes in the carrier frequency.	5
Phase Signature	As with RSSI Signature, phase distance was modified by using information from the RSSI Signature to reduce the error.	6
RIP (RSSI-Informed Phase)	This hybrid approach developed over the course of this work uses both RSSI information and Phase information for more robust readings.	5

Five different distance estimation methods are described in Table 11-1. RSSI Signature, Phase Signature and RSSI-Informed Phase were developed as a result of this work. RSSI and phase are commercially available methods of distance measurement with RF systems.

As described earlier, these different distance measurements where then each fed into a matrix of localization algorithms. This yielded a large matrix in which each combination of distance measurement input with each method of localization method could be compared for accuracy and time.

The different localization algorithms used in this analysis are described in Chapters 9 and 10. Since all measurements contain errors there is no closed form solution to move from distance estimations to a tag or beacon location. Localization therefore relies not only on distance measurements but on the algorithms to move from those measurements to a reported tag/beacon location. These methods are shown in table 11-2 below.

Table 11-2: Localization Algorithms used in the following analysis and comparison.

Number	Name	Description
1	Linear Locus of Position (Linear LOP)	A purely geometrical method which uses the intersection of line produced from the distance estimation radii.
2	Lear Least Squares (Linear LS)	A commonly used linearized method that assumes that all measurements have equal variance and are uncorrelated.
3	Centroid	A method that looks at the center point of all distance radii overlap. Since not all distances will have this overall all distance radii are expanded equally until the overlap is created in the centroid of the overlap can be found.
4	Linear Approximation	A translation of the beacon/tag location to a mathematically more convenient location along a coordinate axis based on simplifying assumptions, and then translated back to their global position.
5	Hyperbolic Locus of Position (Hyperbolic LOP)	The non-linearity of the hyperbolas adds a weighting factor to the Linear LOP algorithm.
6	Taylor Series	A linear iterative method that alters the linearized equations as it approaches the best solution.
7	Weighted Least Squares (Weighted LS)	This differs from Linear LS in that it is iterative and a specific weight is calculated for every iteration making converging on a solution faster.
8	Iterative Least Squares (Iterative LS)	Similar to the Weighted LS but with a different weighted algorithm.
9	Non-Linear Least Square (Nonlinear LS)	A version of the Least Squares algorithm used to fit a non-linear model.
10	Antenna Pattern	A weighting on the distance measurements based on the optimized simplified antenna pattern oriented to best match the distances measured.

Of the 10 methods shown in Table 11-2 it should be noted that Centroid and Hyperbolic Locus of Position were significantly improved upon during this work. Additionally, the Antenna Pattern method is an entirely new method developed during this work.

11.3 Levels of Trilateration

When employing any trilateration algorithm, there are a variety of options (e.g. technology or algorithms) to consider when choosing the optimal combination. For this work, these will be described as "levels" of the trilateration algorithm. Figure 11-4 is an illustration of the various levels for trilateration explored in this chapter.

- Level 1 – Technology
    - This research investigates both RFID and Bluetooth transponders.
- Level 2 – Distance Estimation Method
    - The next step in trilateration is to find a range estimation, as discussed in Chapters 5, 6 and 9. These differ between RFID and Bluetooth, because Bluetooth does not provide phase information.
- Level 3 – Selection of Distance Measurements
    - Trilateration requires a minimum of three distance estimates; while four or more satellites were used in this research in order to explore the use of additional satellites, some algorithms can only incorporate three. Thus, for Level 3 different strategies were compared using all or particular distance estimates from the satellites.
- Level 4 – Non-Iterative Trilateration (or Initial Location Estimation)
    - Most non-iterative trilateration algorithms are linear simplifications of a complex solution space. These can be used as a final localization result, or can be used as an initial value for the iterative trilateration algorithms.

- Level 5 – Iterative Trilateration
    - These algorithms initialize using the non-iterative solutions, and leverage optimization tools to resolve higher order models.

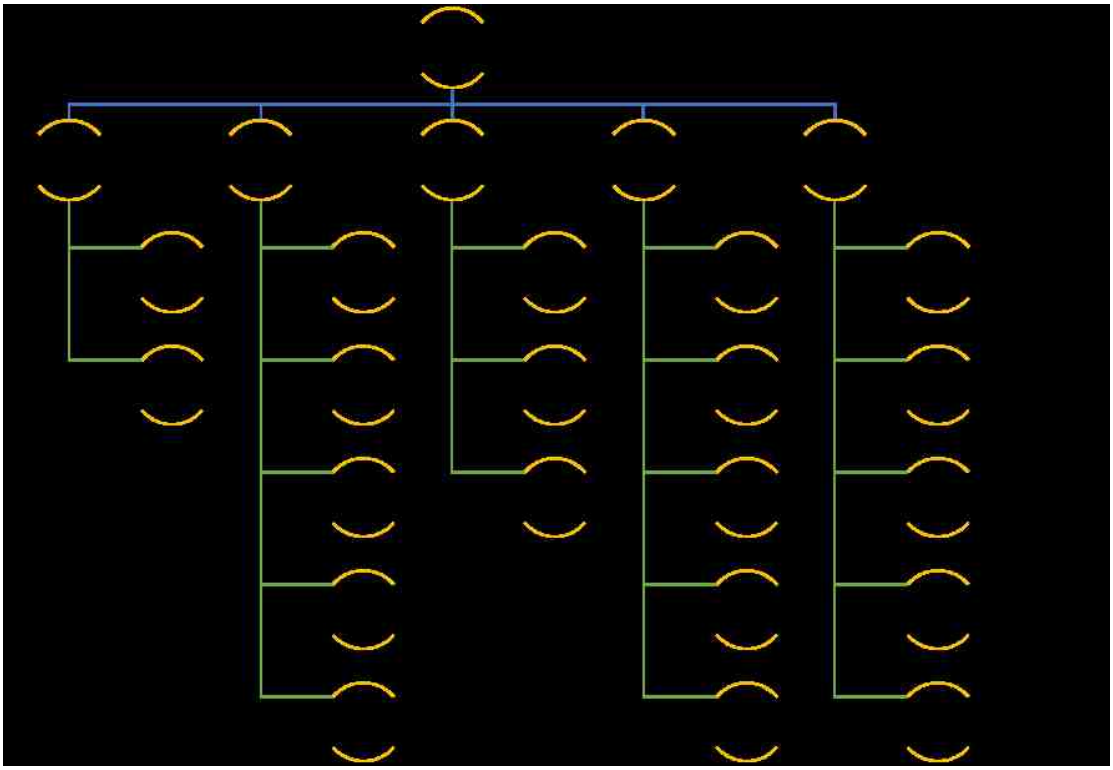


Figure 11-4: Levels of options investigated for trilateration

Figure 11-4 graphically illustrates the matrix of options discussed in this analysis. Level 1 selects the appropriate technology for the particular application. In this research both RFID and Bluetooth were used as options, and empirical data was collected for both, such that any algorithm (or combination of algorithms) could be tested using the data. The second level is the type of data/distance estimation method. RFID is able to use phase angle, thus the distance estimation methods of: phase, RSSI-Informed phase, and phase signature are all available, as well as RSSI and RSSI Signature. Bluetooth is only able to measure RSSI, thus RSSI and RSSI Signature are the only two methods available for Level

2. The third level is a choice of either using the data from every anchor, or choosing the best three, by either the highest three RSSI values, or those of the highest confidence. Confidence is chosen by finding the approximate error in the distance measurement by employing the RSSI Signature. The three distances with the least error as estimated by the trained RSSI Signature neural network, would then be utilized in the remaining levels of the trilateration process. The fourth and fifth levels are both localization algorithms. As discussed above, the difference between the two is that level 4 contains closed form solutions, and thus become the starting point for the fifth level, which are the iterative methods. The algorithms chosen represent a variety of key trilateration methods discussed in the literature, therefore by comparing all combinations using the same set of empirical data, it is possible to truly evaluate the of the merits of each technique or combination of techniques.

11.4 Overall Results

A high level comparison of the different distance techniques for RFID is shown in Figure 11-5 below. This graph and the following table summarize the overall mean error for all the data by the Level 2 distance measurement techniques.

RFID distance measurements are limited to a shorter range than BLE due to the non-powered nature of the device, but they have the advantage of being able to use phase-based distance estimations. The phase method of distance estimation is found to be the best for low error, and is superior with the variations of that measurement, including the Phase Signature and the RSSI-Informed Phase methods. This is possibly due to the data being collected while stationary in a fully anechoic chamber, thus the RSSI-Informed Phase and

RSSI Signature methods were not as necessary for compensating for multipath, motion, or other large sources of error. Phase-based measurements require more data per measurement (more frequency information) than the RSSI-Informed Phase method and may not be suitable for fast processing. Additionally the equipment required to do phase measurements is more expensive than just RSSI measurements and for that reason phase-based techniques will be limited to situations where the combined value of the items being track warrants the higher capital outlay.

RSSI measurements for trilateration are the most common method for the reasons just mentioned. For these measurements the RSSI signature was superior to the RSSI traditional methods.

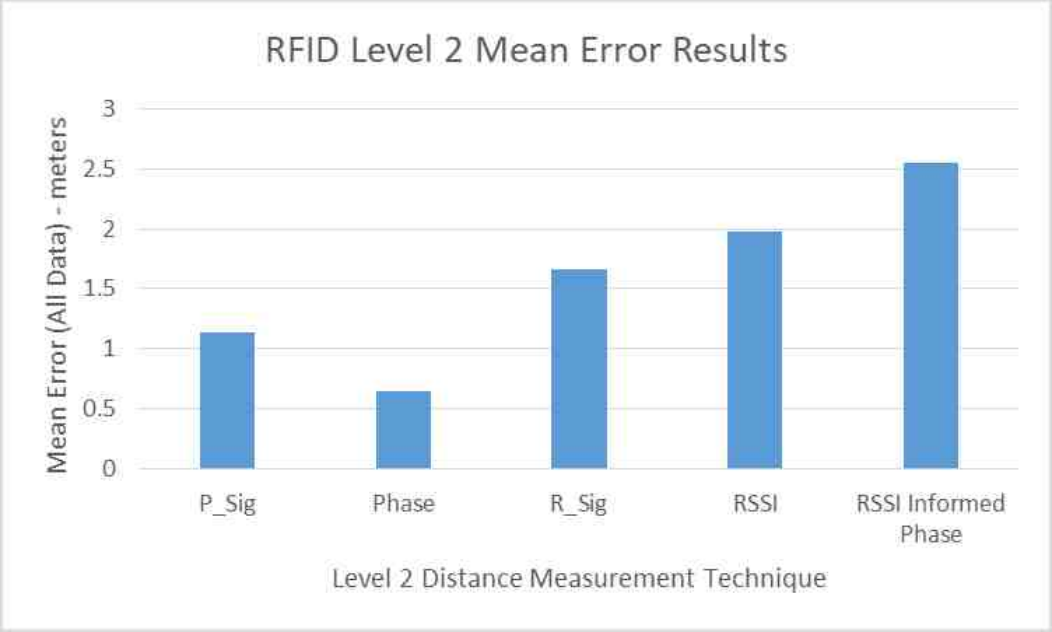


Figure 11-5: The error (in meters) from each distance estimation method for RFID system. Resulting error is the mean of all data for all trilateration methods.

Table 11-3 Level 2 Results

RFID Level 2 Results	
Row Labels	**Average of 2.050406904**
P_Sig	1.13543794
Phase	0.653476406
R_Sig	1.668292554
RSSI	1.98153388
RSSI Informed Phase	2.546218189
Grand Total	**1.603218376**

Figure 11-6 shows the results including Level 3 and 4 algorithms evenly distributed among the data types for each RFID distance estimation method. For all the iterative methods which require an initialization point the results from the hyperbolic locus of position algorithm were used as the starting point for iterations to begin. This algorithm was chosen since it often achieved high accuracy in comparison to the other closed form solutions.

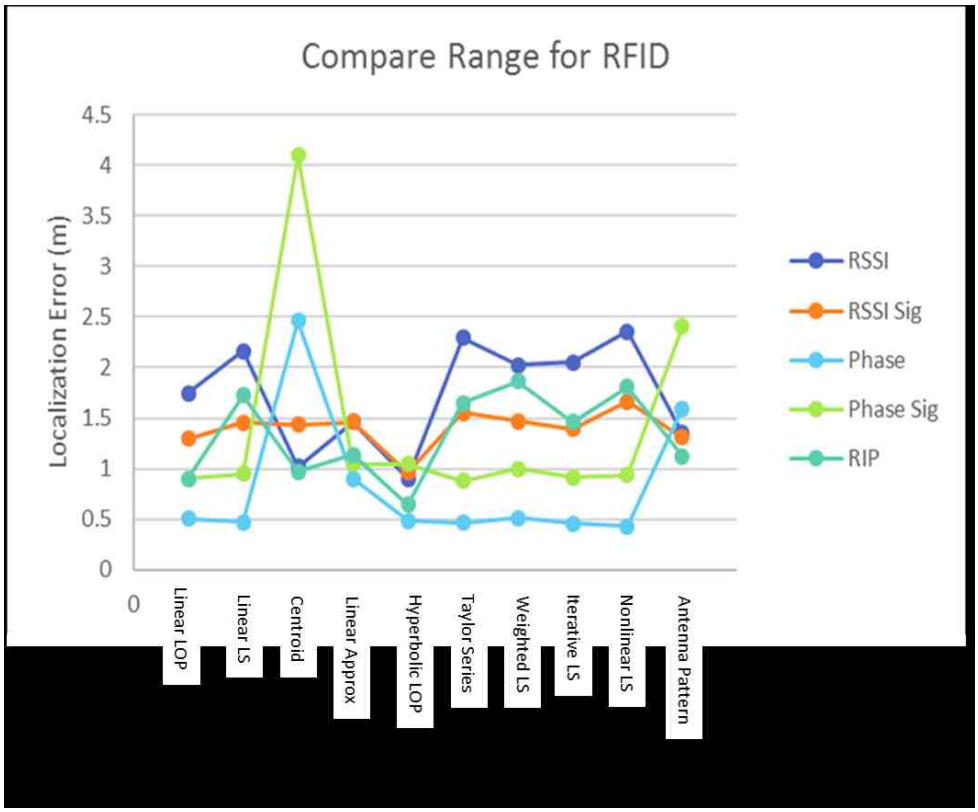


Figure 11-6: Each distance estimation method for RFID System with respect to different trilateration algorithms (numbers are listed in Table 11-2).

Figure 11-6 illustrates how not all trilateration techniques behave the same with various distance methods. Different algorithms work better with different distance measurements. The Centroid method, item 3 in Table 11-2, is an excellent example of this as it had the highest error for the two overall most accurate distance techniques, phase and Phase Signature. For this reason the selection of a trilateration method is also a critical part of the final result as will be discussed later in this chapter.

11.4.1 RSSI Signature

While employing phase is typically more accurate for range estimation, it requires more expensive hardware to implement. Therefore, for more cost sensitive applications,

an RSSI-based technique will likely be selected. When comparing the RSSI and RSSI Signature techniques, the results demonstrate that the RSSI Signature method does in fact increase accuracy significantly for nearly all the localization algorithms. Given that all the trilateration measurements were collected in a fully anechoic chamber, there was less RSSI multipath error to be compensated for. Thus, the potential for increased accuracy in complex environments is significant.

11.4.2 Phase Signature

Based on the results displayed in table 11-3, it can be seen that for most of the trilateration algorithms, the phase-based distance estimation has the lowest total error. Utilizing the Phase Signature, while also fairly accurate in comparison to all of the methods, actually lowers the accuracy from simply using phase angle. With the data having been collected in an anechoic chamber with minimal multipath, and with phase measurements being more robust with respect to multipath it's possible that by applying the signature to reduce multipath the total error increased because there was minimal multipath error to compensate for.

11.4.3 RSSI-Informed Phase

In these experiments RSSI-Informed Phase is shown to be the least accurate of the Level 2 options; however, RSSI Signature, phase, and Phase Signature are all movement sensitive, and RSSI is sensitive to multipath and antenna angle and can yield extreme errors. RSSI-Informed Phase was intended to be used in situations of extreme multipath, antenna angle, and motion; therefore it makes sense that it doesn't perform well on this

particular experiment which has none of those factors. The RSSI-Informed Phase is the most generally robust technique, and therefore for some applications this method would still be the appropriate choice. For instance, when using the Centroid method, which has high levels of error for both phase and phase signature, the RSSI-Informed Phase method significantly decreased the error by incorporating RSSI distance estimation. The experimental conditions for the trilateration comparison, were completely stationary tags in a fully anechoic chamber. Thus, the advantages of the RSSI-Informed Phase method are not expected to be evident.

There are different ways to observe the results from Table 11-3. Either from the perspective of a combination of levels which achieves the highest accuracy could be selected, for this analysis that combination would be would be phase, hyperbolic locus of position, and nonlinear least squares. In contrast, another way to interpret the data is to look at the general trends as to what methods typically perform well. It should also be noted which methods are fast (RSSI and RSSI-Informed Phase) and which are slow (RSSI Signature, phase, and Phase Signature). As discussed in Chapter 7, the RSSI Signature can represented in such a way that decreases the number of data points needed, and thus speeds up the distance estimation time but this time is still larger than methods considered “fast”. If a fast method is needed and high levels of accuracy are unnecessary, but occasional reads with extreme error cannot be tolerated, then the RSSI-Informed Phase method is likely the best choice. If instead high precision is required and more time can be taken, then the RSSI Signature or phase distance estimation methods are the best options.

11.5 Localization Comparison – Level 2

11.5.1 Bluetooth Low Energy

A similar process was performed for BLE systems, however there are only two possible distance estimation methods for BLE, given that there is no phase measurement. The overall comparison shows that the mean of the distance error is less for RSSI Signature than for traditional RSSI measurements. These results are shown in the graph and table in Figure 11-7.

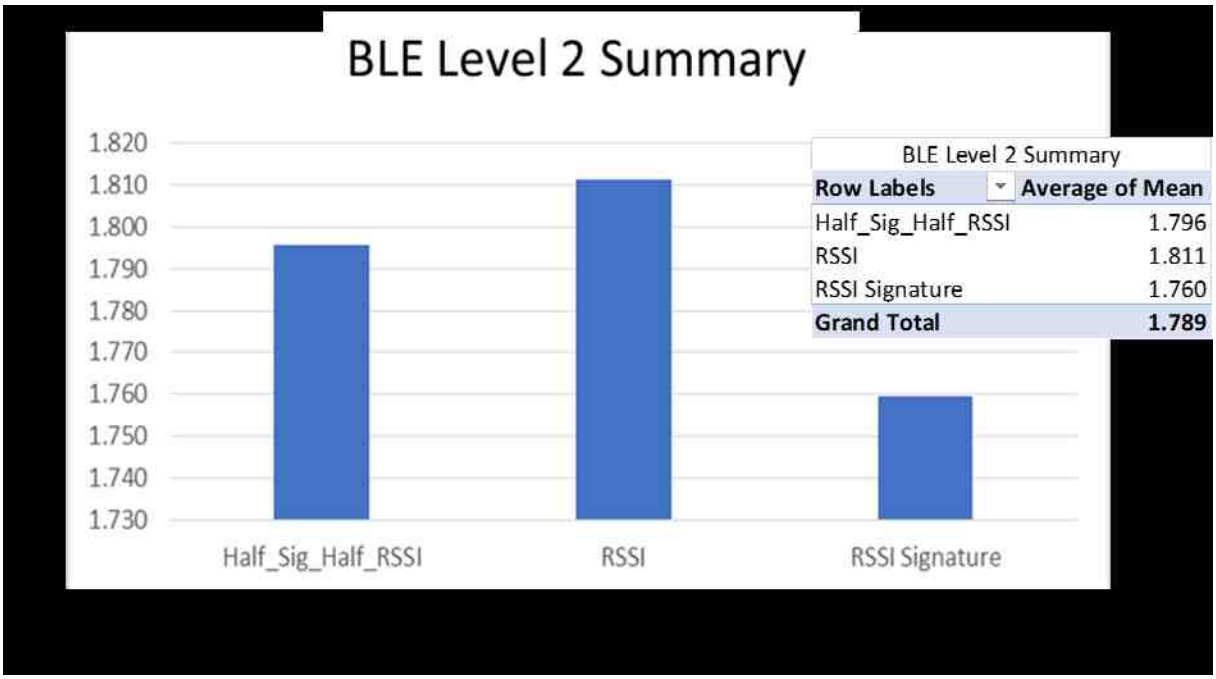


Figure 11-7: Comparing the two distance estimation methods, and a combination for the BLE System. Data for the graphic is shown in the pivot table. The lowest mean error for all data was using RSSI signature.

In the histogram shown in Figure 11-8, it can be seen that not only does that RSSI Signature method shift the bell curve to be more centered about zero, but it also reduces

the tails (especially to the right). This is a strong indication that the RSSI Signature method is very effective for both reducing outlier measurements and improving overall measurement accuracy.

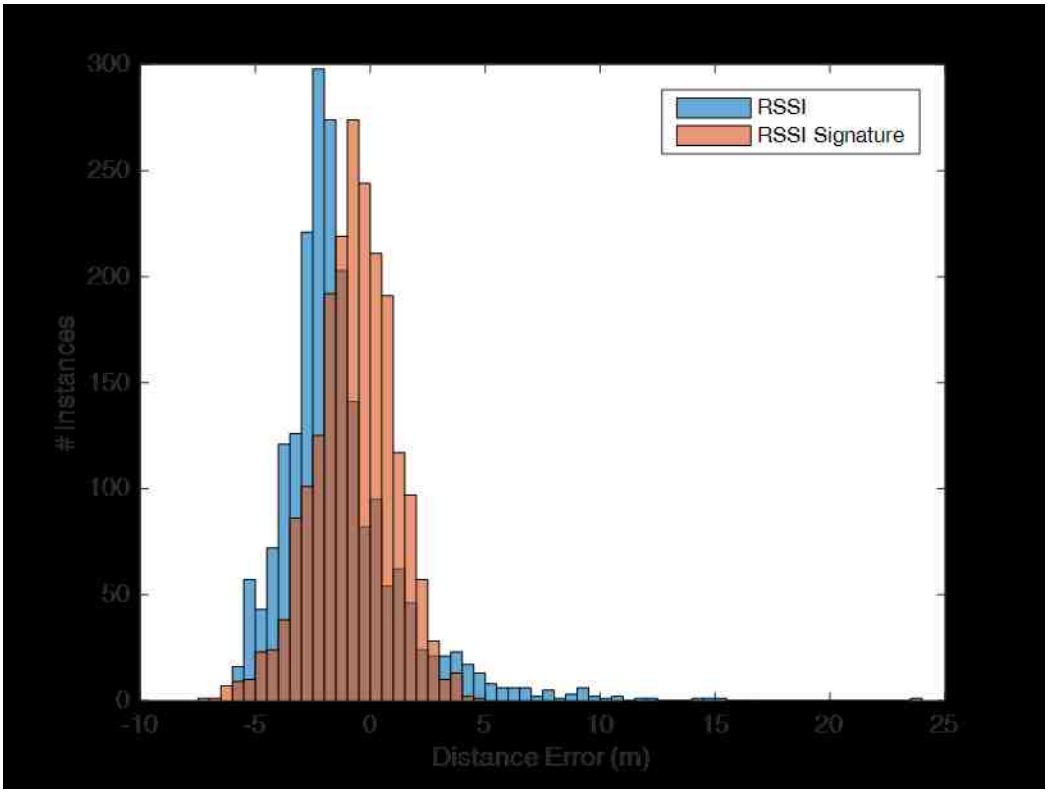


Figure 11-8: Histogram of RSSI and RSSI Signature distance estimation error.

The results of the comparison of each of the trilateration algorithms with respect to the two distance estimation methods is shown in Figure 11-9. This further slicing of the data, as was demonstrated for the RFID analysis, shows that the individual trilateration method has different sensitivities to the types of error generated by the different methods.

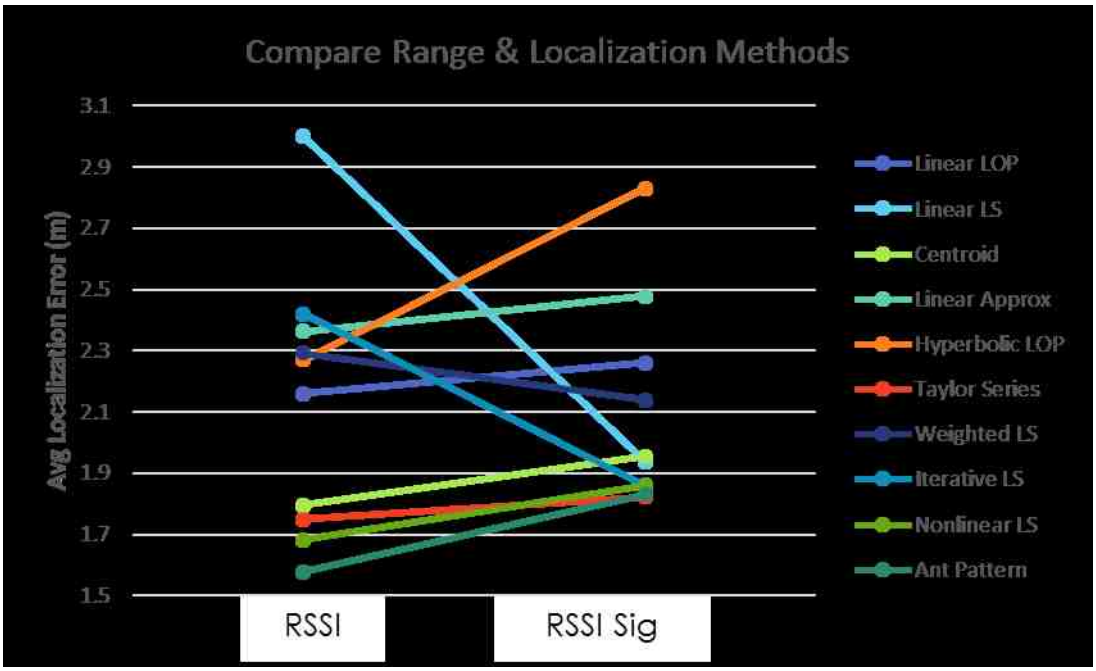


Figure 11-9: Comparison of the trilateration algorithms for BLE system

It is interesting to note that when moving between simple RSSI and RSSI Signature distance estimation methods, some of the trilateration algorithms improve, while others digress. This data was collected in a semi-anechoic chamber, and therefore the RSSI Signature did not have as much multipath error to compensate for, and yet on average it still out-performed the non-compensated values. In tested situations where complex environments existed the RSSI Signature was able to reduce error consistently over the RSSI algorithm, and therefore if this analysis had been expanded into more complex environments it would be anticipated that the RSSI Signature would have shown a greater advantage over RSSI alone.

The difference in part, as indicated by the histogram in Figure 11-8, is the RSSI Signature's ability to suppress extreme outlier values. These values occur occasionally even in very controlled environments as is shown in the graph in Figure 11-10 below. This

compares the error from the original distance measurement to the error after it was reduced by the RSSI Signature. The horizontal axis is the difference between the maximum RSSI value for that measurement and the RSSI Signature value, yielding what should be a reduced error. Note that positive numbers means that the RSSI had an error value larger than the RSSI Signature value. If RSSI resulted in larger inaccuracies the points would be high on the y-axis. If then the RSSI Signature was adept at reducing error this would mean the points would be more to the left (centered at zero). Figure 11-10 shows that most points have a larger y value than x value. This means that the RSSI Signature method is effective for reducing error.

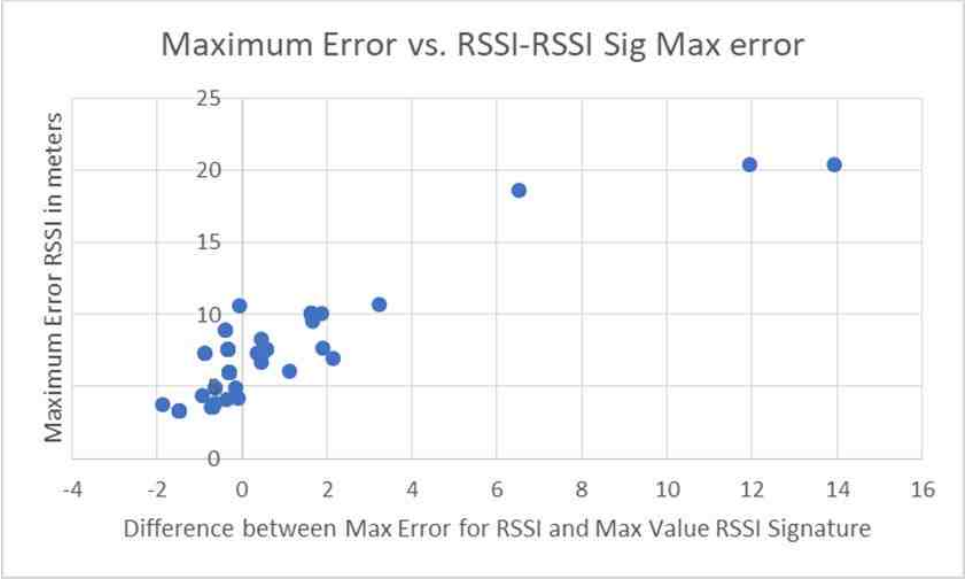


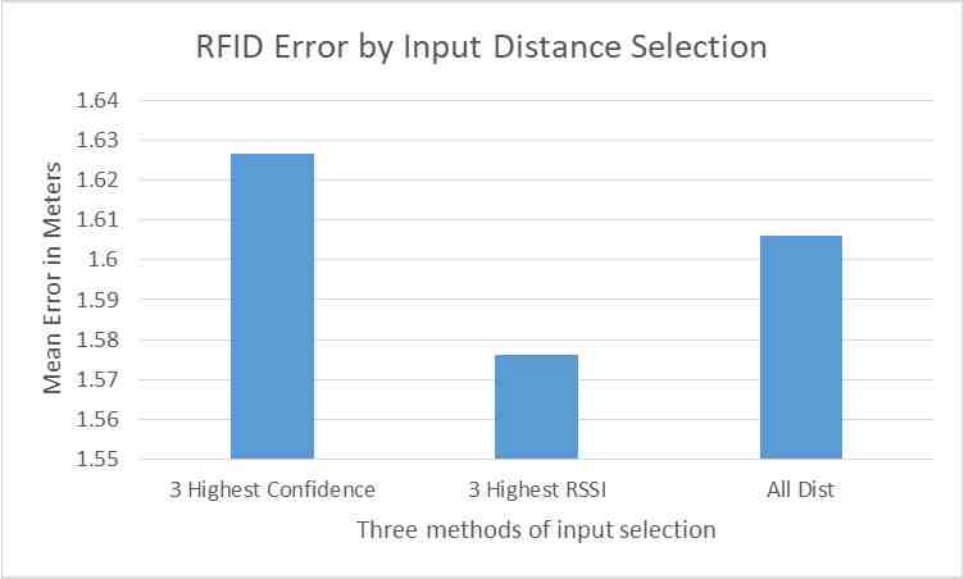
Figure 11-10: A comparison of the likelihood of outlier data points given the same RSSI signal input for the RSSI and RSSI Signature method in the semi-anechoic chamber localization experiment. Positive numbers mean that the RSSI distance had the larger error.

11.6 Level 3 – Which Data to Use

While three distance measurements are required for a 2D location calculation, five satellites or beacons were actually taking measurements at each location spot for BLE and 4 for RFID. For those methods which can use more than three data points, is the additional data useful or not? This issue has been minimally discussed in the literature and it is mainly suggested to eliminate down to 3 points without any justification and there was no clear hypothesis that this choice could or would make a difference in the final result [12] [57] [62].

The three options tested were selected to test different theories. The first theory is that the more data the better, and each additional data point brings additional information to the result. The second theory was that the highest RSSI value was likely the most accurate and so just using those values with the highest RSSI value might prevent erroneous readings from detracting from the correct value. The third theory was that using the signals with the highest confidence level as determined by the RSSI Signature of the distance measurement, would eliminate the high risk values and lead to a better result. This analysis compares those choices to see if they had an impact and at what significance.

The results of the selection of input data is shown in Figure 11-11. This indicates that selecting those measurements with the highest RSSI values yielded the best result.



Row Labels	Average Error
Grand Total	1.603218376

Figure 11-11: Overview of effect of input data selection on trilateration error.

The RFID data had more types of distance measurements due to its use of phase angle and some had significantly lower error than others. However, in every case, using the measurements from the satellites with the strongest signal from the tag resulted in the most accurate estimate of tag location. This includes using the strongest RSSI signal on a measurement to best select which phase measurement to use. Those results are shown in Figure 11-12 below. While the difference was not large compared to the method chosen for the distance estimate, it still repeated with all the measurement types.

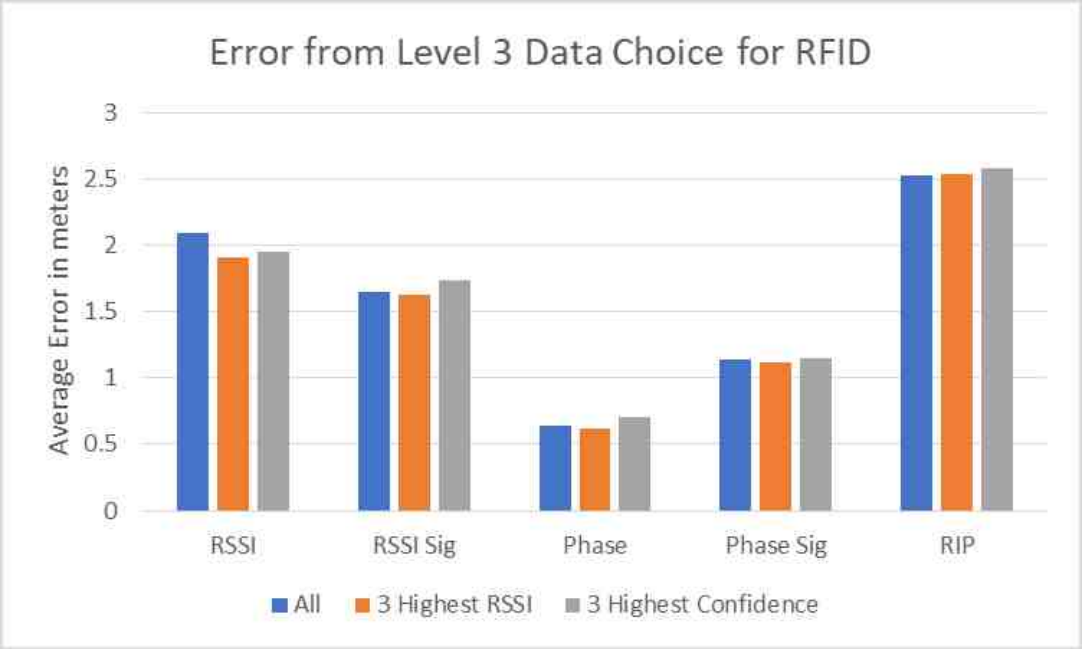
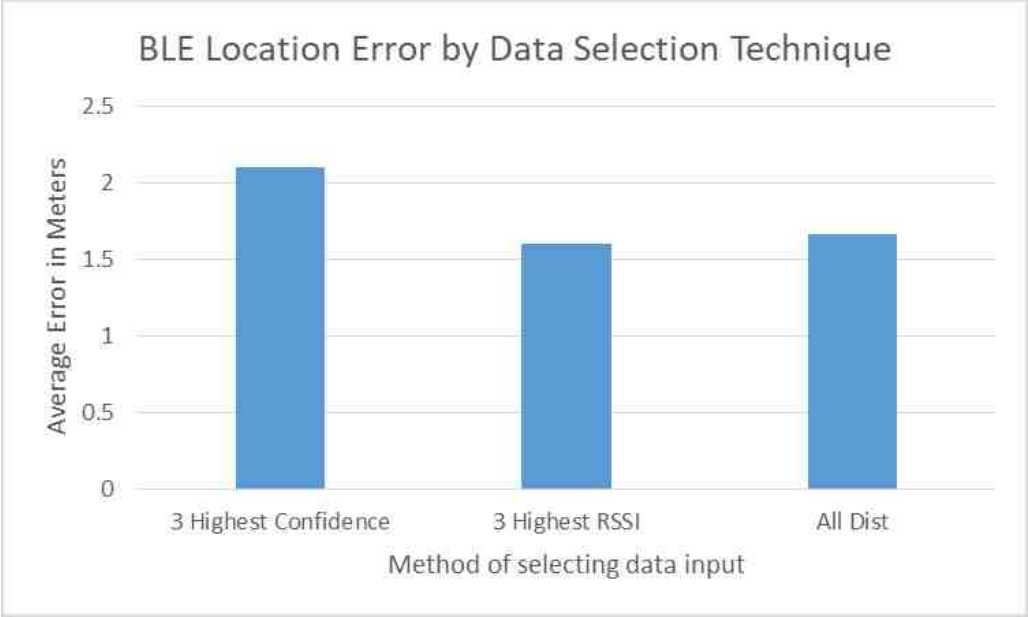


Figure 11-12: The average error for the data for RFID measurements grouped by data selection. Using the three distance measurements with the highest RSSI consistently gave the lowest localization error across all distance measurement methods.

This suggests that for RFID trilateration methods, even if data from more than one satellite exists the accuracy is improved by selecting the three strongest signals rather than trying to use a more consensus view from the entire data set. This would make algorithm application easier as the system could filter a varying number of satellites and select the three highest for the rest of the computation. This therefore would scale better than a method that uses all the data available or one that required more computation before selecting the final three.

Bluetooth technology has fewer distance measurement choices as phase was not possible. The result for the BLE hardware was very much the same as for the RFID hardware, and can be seen in the graph and table in Figure 11-13.



Row Labels	Average error
Grand Total	1.788801003

Figure 11-13: BLE location error as a function of the method of selecting input measurements to use in a trilateration algorithm.

Independent of the method of distance estimate (RSSI or RSSI Signature) using the three highest RSSI signals resulted in the best outcome. Figure 11-14 shows how this method was superior for all of the trilateration methods.

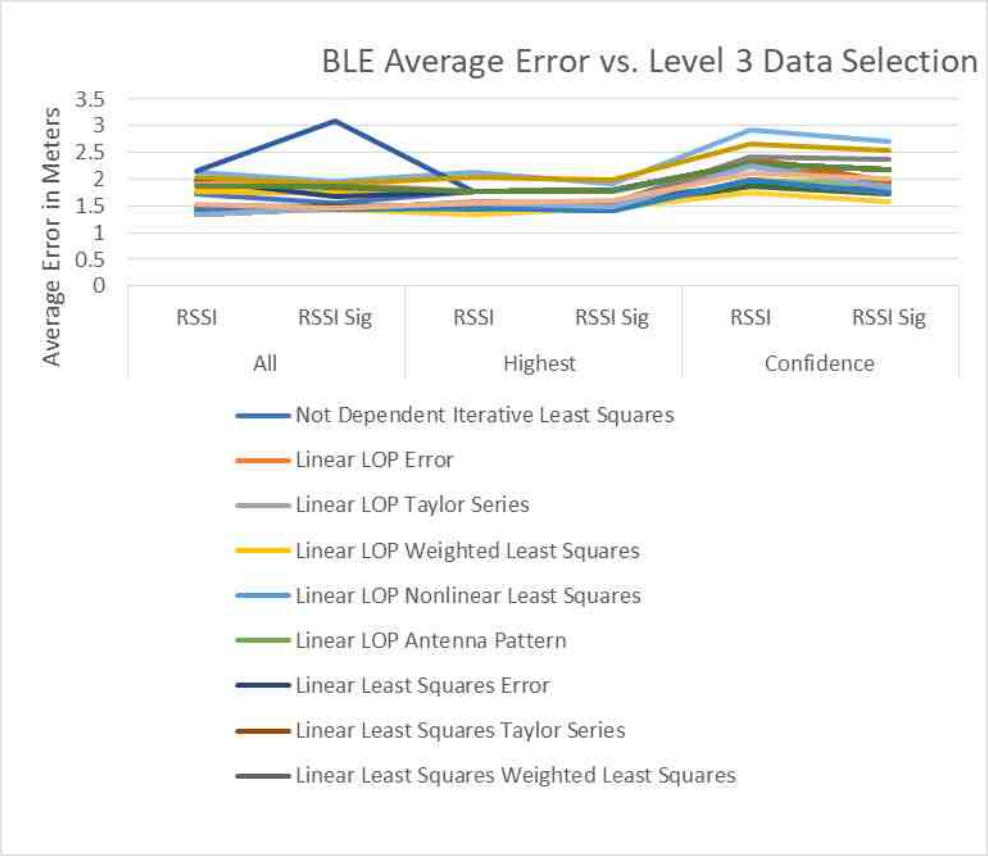


Figure 11-14: Bluetooth average localization error as a function of the type of data input into the different localization algorithms. In every case using the 3 highest RSSI values as inputs gave the lowest error result.

In addition to the average error, the maximum localization error was also examined to see if the choice made in the Level 3 (all distances, 3 highest RSSI, or 3 highest confidence) also impacted the outlier values. Table 11-3 below shows that using the 3 highest RSSI values reduced the number of outlier values while still having many low values for error. Outliers are shown in dark red in this table and at a glance it is easy to see that using the highest RSSI value almost removed all outliers from the analysis for this study, and combining that technique with RSSI Signature to determine the distance was even more effective.

Table 11-4: Comparison of the maximum error for each type of data selection option for Bluetooth technology.

Outlier Check for BLE Measurements

All		Highest		Confidence	
RSSI	RSSI Sig	RSSI	RSSI Sig	RSSI	RSSI Sig
12.27542	14.45772	5.999893	6.28861	10.04575	8.169646
6.002618	5.538338	6.002618	6.293417	10.05611	8.43099
4.406218	5.345154	3.55076	4.259383	7.573617	7.912672
6.002618	5.538338	6.002618	6.293417	10.05611	8.43099
3.736001	5.618716	3.333524	4.808111	7.357459	8.2283
4.036575	5.167062	4.90062	5.527408	8.921752	9.318229
15.86442	18.44037	6.002618	6.293417	10.05611	8.43099
4.40952	5.353037	3.55076	4.259383	7.573617	7.912672
10.60421	8.660768	6.002618	6.293417	10.05611	8.43099
3.736005	5.618418	3.333524	4.808111	7.357459	8.2283
7.242481	8.982151	4.90062	5.527408	8.921752	9.318229
4.294791	4.355755	4.176317	4.279495	4.923596	5.08395
4.4062	5.345519	3.550759	4.259394	7.573534	7.009073
11.38401	10.86289	6.002618	6.293417	10.05611	8.43099
3.735999	5.618495	3.333524	4.807836	7.357461	7.00489
4.441083	4.829423	3.850771	4.491264	6.702625	6.25625
6.075053	4.941785	6.075053	4.94529	10.7102	7.48147
4.406784	5.345773	3.550765	4.260677	7.573578	7.905052
26.13094	33.67344	6.002618	6.293417	10.05611	8.43099
3.735996	5.618073	3.333524	4.807849	9.567589	7.917931
5.249575	5.16385	4.069613	4.418456	8.263195	7.810304
20.47427	6.538127	20.47427	8.514765	18.61382	12.10394
4.406088	5.345221	3.550743	4.259452	7.573533	7.009074
10.57912	10.65273	6.002618	6.293417	10.05611	8.43099
3.735994	5.620695	3.333524	4.808592	7.357468	7.00489
6.953959	4.79897	7.670422	5.775822	7.337746	6.845935

11.7 Non-Iterative Location Algorithms

The next step in the localization process is to choose a non-iterative method to find a likely location for the beacon/tag in question or possibly the final location. If the application requires that the final algorithm be iterative, one of the non-iterative methods will most likely serve as a starting point.

The non-iterative location methods examined here were: Linear Locus of Position (Linear LOP), Linear Least Squares (Linear LS), Centroid, Linear Approximation, and Hyperbolic Locus of Position. As mentioned in Chapter 9 one of the products of this research was an improved weighting for the Centroid method and taking the Hyperbolic Locus of Position to a theoretical concept to a working algorithm.

The importance of this level of comparison is that the measurements are already done and which algorithm is applied is a zero cost difference choice to any potential customer. Knowing how the different algorithms behave should allow for any systems developer to select the best performing algorithm for their particular application.

For the RFID data the Hyperbolic LOP had the lowest average error of all the non-iterative types, as shown in Figure 11-15. Recall that the Hyperbolic LOP is like the Linear LOP except that the non-linear weighting factor of a hyperbola causes the algorithm to bend the location vector towards the satellite with the shortest read distance. Essentially this incorporates the nonlinear weighting directly into the trilateration rather than shifting a linear result. As the higher RSSI readings are apparently more accurate the algorithm that appropriately weights this is also more accurate. It is interesting that the Linear LOP is not the next most effective algorithm, suggesting that the non-linear hyperbola is a significant improvement. In fact the single best result from these experiments was with the Hyperbolic LOP with no further localization improvement form an iterative method.

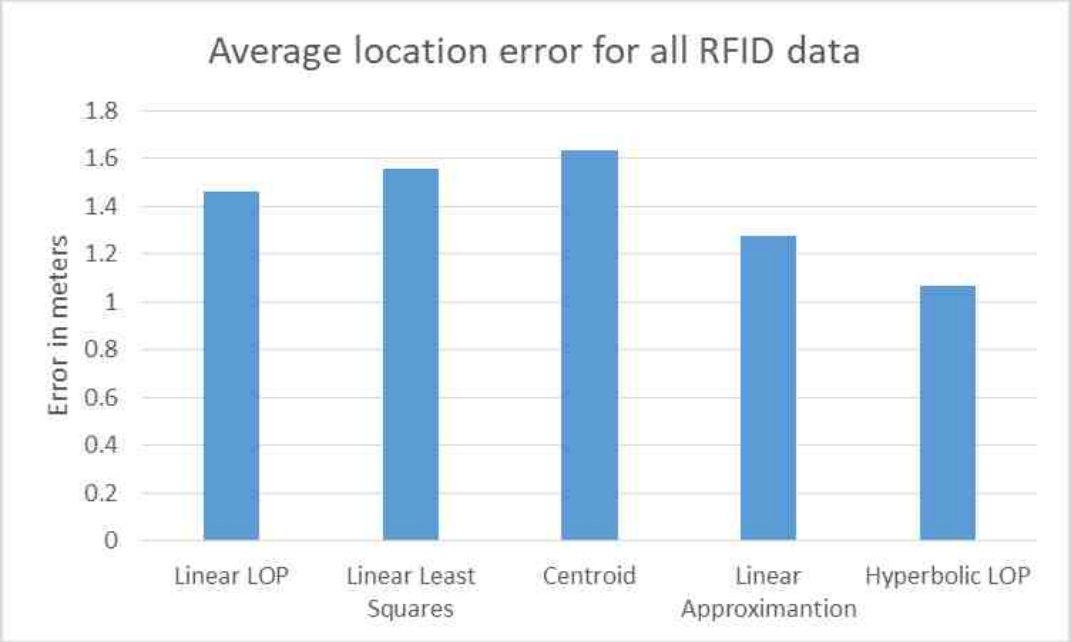


Figure 11-15: Average location error for RFID by non-iterative location algorithm. This is an average of all data taken using all the Level 2 and Level 3 options.

On the other side of the performance spectrum it is interesting to note that the Centroid method had the worst performance. Even with the improved weighting the Centroid proved most susceptible to outlier distance measurements. The Bluetooth data results however are very different as seen in Figure 11-16 below. For the Bluetooth arrangement the Centroid method turned out to be the most accurate and the Hyperbolic LOP one of the worst.

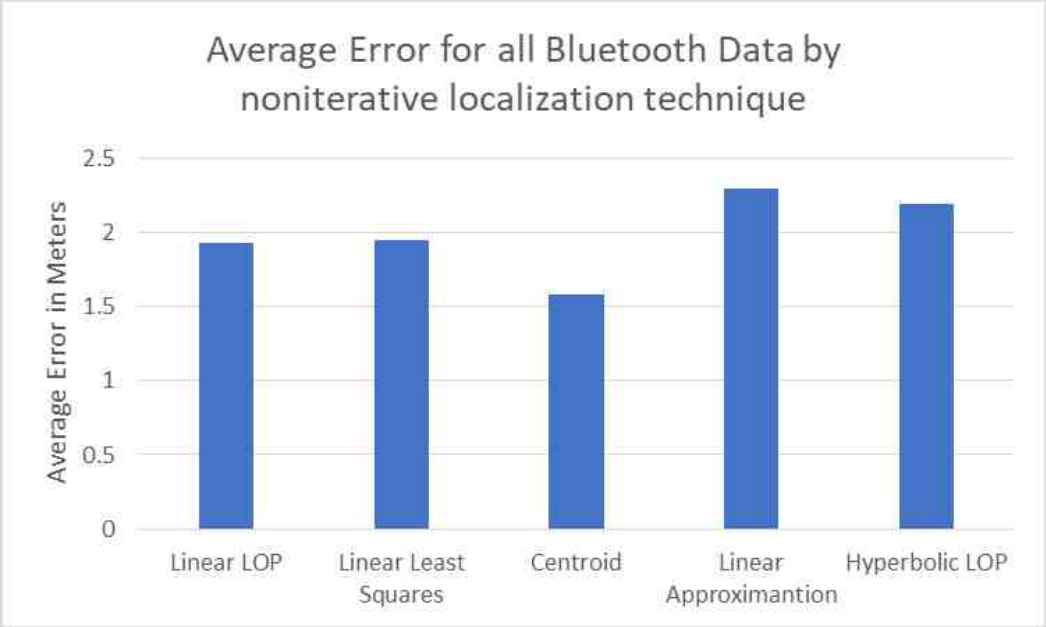


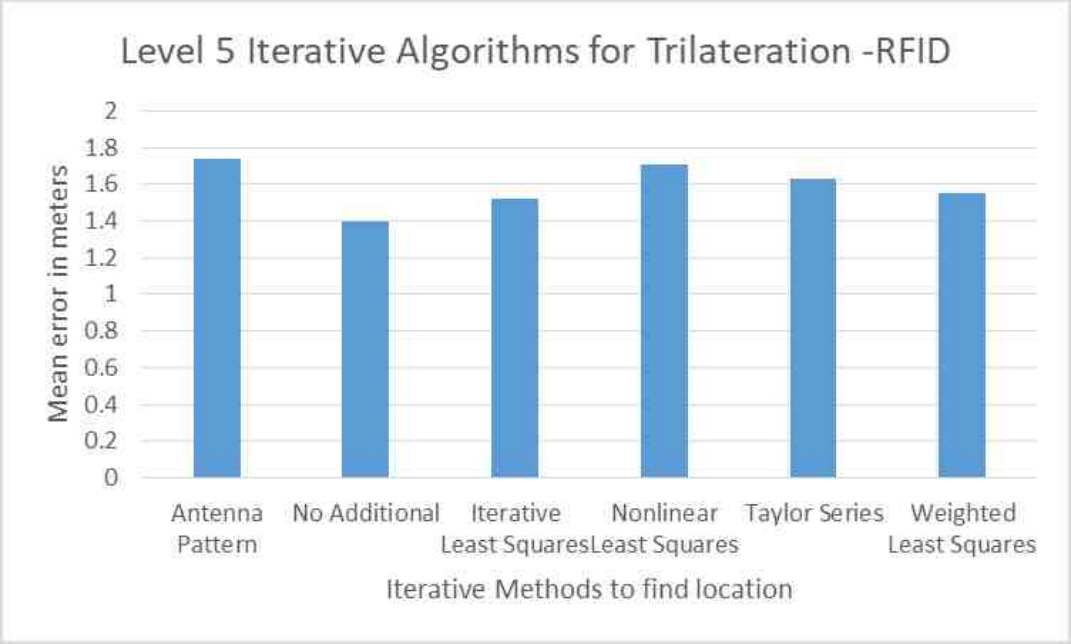
Figure 11-16: A comparison of the average error for all the Bluetooth data by Level 4 method of non-iterative localization. Unlike the RFID data, the Centroid method appears to work very well with Bluetooth devices

So why is the Centroid so much more accurate for a Bluetooth system than for an RFID system? The answer might be in the non-passive (powered) nature of the Bluetooth devices. The system was not as susceptible during this test to low power reads. Even though the experiment was conducted in a 10 meter semi-anechoic chamber, these distances were easily within range for the Bluetooth system. In contrast these measurements were frequently much closer to being out of range for the RFID devices. It is possible that if this experiment were repeated in a larger arena such that the Bluetooth device could be closer to their real limit, that the graph in Figure 11-16 might start to look more like to the graph in Figure 11-15.

11.8 Iterative Localization Methods

The final level in the localization hierarchy is the iterative methods of localization. These build on a starting point from one of the non-iterative methods and try and move the initial location estimation to a more accurate final position. Since these algorithms need a starting point, the Hyperbolic LOP was used as the starting point for all but the Iterative Least Squares which is not dependent on a starting point from some other algorithm. Ideally the added work from an iterative solution should be able to improve somewhat on the initial result, although that did not necessarily turn out to be the case. As mentioned above for the RFID data the non-iterative methods actually achieved a better performance with less error.

The table and graph in Figure 11-17 compares the iterative methods of localization for all data for RFID tags.

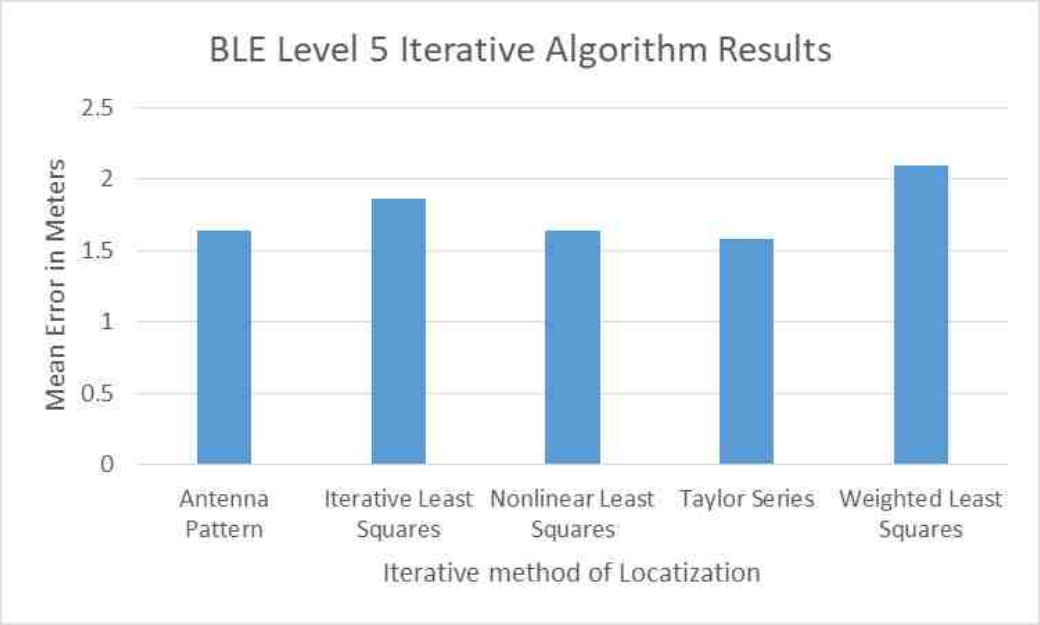


Row Labels	Mean Error
Grand Total	1.652118924

Figure 11-17: RFID based error results for location based on different iterative techniques to attempt to improve performance.

Appendix B of this document contains a comparison for iterative output results as a function of all the different non-iterative inputs and data selection techniques. As expected the better the input from the non-iterative method, the better the final result from the iterative one. The best iterative method in this set of experiments for RFID was the Iterative Least Squares and the Weighted Least Squares methods.

The Bluetooth data however had different results when testing the Level 5 iterative algorithms. The results of that comparison are shown in the table and graph of Figure 11-18.



Row Labels	Average of Error
Grand Total	1.743991955

Figure 11-18: Iterative algorithm techniques for Bluetooth device data

As the Centroid method had performed the best in the non-iterative location, when used as the starting point, it gave each of the iterative methods their best result. The Antenna Pattern method apparently was best able to take advantage of this good starting point and yielded the overall lowest error on the table with that combination. This is shown in Table 11-4 below. This could be an indication that the Antenna Pattern method is

effective at incorporating additional information about the beacon antenna pattern, but is susceptible to local minima. Thus a more accurate starting location would give the Antenna Pattern method a chance at narrowing in on the true and global minima.

Table 11-5: A pivot table showing the relative average error for the different iterative methods given the different initial non-iterative starting point for the Bluetooth data.

Average of error	Column Labels					
Row Labels	Antenna Pattern	Iterative Least Squares	Nonlinear Least Squares	Taylor Series	Weighted Least Squares	Average of error
	1.48961242					
Average of error	1.651308382	1.885004985	1.665397783	1.59223311	2.009707197	1.737058921

In the experimental results shown in Table 11-4 the Taylor Series was the most accurate method followed by the Antenna Pattern method if using any starting point. However, the best overall value is the Centroid and Antenna Pattern combination.

11.9 Accuracy is not the only criteria

While the analysis in this chapter to this point has focused on the accuracy contributions of the different levels of the localization algorithm process, it should be noted that accuracy is not the only design feature that must be considered. Table 12-5 below reviews the different iterative and non-iterative location algorithm and compares them by computational speed, the ability to handle varying numbers of satellites (flexibility) and demonstrated robustness against highly erroneous results.

Table 11-6: A comparison of the different location algorithms using the RSSI Signature method for Bluetooth localization.

Summary of Algorithm Attributes

Algorithm	Computational Speed	Satellite # Flexibility	Outlier Robustness
Linear LOP	fast	no	average
Linear LS	fast	yes	average
Centroid	fast	yes	poor
Linear Approximation	fast	no	good
Hyperbolic LOP	Fast, more complex	no	good
Taylor Series	iterative	yes	average
Weighted LS	iterative	yes	average
Iterative LS	iterative	yes	average
Nonlinear LS	iterative	yes	average
Antenna Pattern	iterative	yes	good

In Table 11-5 speed refers to whether the algorithm would be capable of tracking applications in indoor environments, flexibility has to do with whether it can take a variety of reader/beacon inputs, and robustness was the relative sensitivity of the method to outlier data.

Depending on the system requirements, a flexible system with good robustness to outliers may be an important criteria. Conversely the speed of computation may be more important in other applications. The same type of considerations feed into the selection of a distance estimation algorithm as shown in Table 11-6.

Table 11-7: A comparison of the different distance measurement techniques, accuracy is compared for average of best 3 algorithms.

Measurement Method Comparison

Distance Method	Accuracy	Outlier Performance	Relative Hardware Cost	Tracking Applicable	Available for BLE	Complex Environment
RSSI	good	Movement tolerant	moderate	fast	Yes	poor
RSSI Signature	good	Movement sensitive	moderate	slower	Yes	Best across most methods
Phase	best	Movement sensitive	high	slower	No	Best for simple conditions
Phase Signature	good	Movement sensitive	high	slower	No	good
RSSI-Informed Phase	good	Movement tolerant	high	fast	No	Better for extreme conditions

In Table 11-6 accuracy refers to distance error, while outlier performance was the likelihood of the method to produce high error values. Relative hardware cost refers to the cost of a reader system. Tracking applicable means the speed of the method fast enough for tracking applications. Complex environments column shows how well different distance measurements perform in different conditions, or environments.

While defining the best method of distance estimation or trilateration depends upon the environment being used, for these experiments the phase-based distance estimation method with a nonlinear least squares algorithm, based in the 3 highest RSSI values (or RSSI Signature if phase is not available) was the best solution for RFID systems. In

contrast a RSSI Signature distance estimation in conjunction with the Antenna Pattern method using the Centroid method as a starting point achieved the highest accuracy for the BLE system.

11.10 Conclusion

The purpose of this chapter was to compare the different combinations of distance estimation methods and localization algorithms for both the RFID system and the Bluetooth system. While the right combination of design choices will vary based on the particular needs of the application, there were some noticeable winners in this test.

For the RFID system the phase distance estimation had the lowest error, while more expensive it gives the most accurate results. If cost is an issue the RSSI Signature is very effective and might have performed even better if this analysis was performed in a more complex environment. For the Bluetooth the RSSI Signature was the best performer, though again a more complex environment is expected to favor the RSSI Signature method even more.

For the data to feed into the trilateration algorithms, both the RFID and the BLE systems benefited from using just the three strongest RSSI signals. This was the only level of this research for which both technologies responded the same, which is significant though not originally a focus of this localization research.

For non-iterative trilateration algorithms the Hyperbolic LOP was the best performer for the RFID system. This method was developed into a functional algorithm for the purposes of this work. The Hyperbolic LOP performed well enough to be the best overall

method for the RFID system. For the Bluetooth system, the Centroid method, for which a better weighting method was developed for this work, gave the best results. Additionally for the Bluetooth data the best overall result was the combination of the Centroid and the Antenna Pattern method developed by this investigation.

Finally it should be noted that the best hardware and algorithm for any application will ultimately depend on the application, and thus the pros and cons of the technology's combination with each level of trilateration have been addressed. It is hoped that the system design engineer will be able to use this information to more easily select the technology and algorithm most suited to their customer's needs.

CHAPTER 12. ERROR ANALYSIS

12.1 Introduction

Every measurement system contains some amount of error, but confidence in the findings comes from understanding and quantifying the error as much as possible. This research involved measurements that generated error from electronic instrumentation which can be modeled using traditional signal response techniques. Additionally, error was generated in attempting to identify the true location for comparison. The error in this work is the difference between the "true" position and calculated position for a tag using different localization techniques. For localization systems, determining the range from tag to reader is the largest source of position error (as discussed in previous chapters), and so items contributing to those errors are the source of closest scrutiny.

12.2 Overview

This chapter explores the most likely sources of error involved in this research. These include:

Types of Error

- Actual Location Determination
  - "True" Distance
- Instrument Error
  - Reader Error
  - Cable Losses
  - Antenna Error
  - Polarization Mismatch
  - Balance Error
  - Impedance Mismatch to Reader
  - Power and Frequency Output of Radio
  - Frequency Response of RFID Tag and BLE Device
- Environment Errors
  - Ambient RF Noise
  - Directivity and Pointing Error

- Environment and Multipath

The errors listed above can be grouped into three general categories. The first category addresses the measurement of "true" distance; against which RF localization and distance measurements were compared. Most of the remaining error falls into the second category of instrumentation errors associated with making the RF distance measurements. These errors include factors listed above from reader error to the frequency response of the RFID tag or BLE device. Lastly, there are errors which arise from potential application environments: ambient RF noise, antenna directionality, and multipath. These errors will vary from situation to situation; and in this chapter it has been attempted to decouple and quantify each type of error.

12.3 Actual Location Determination

12.3.1 "True" Distance

When preforming experiments using RF to find either distance or location e.g., in Chapters 5 and 6, "true" or "absolute" distance was used to find the error from the RF measurement. This value was found by using a tape measure, and was selected for cost and availability, and provided results well within measurement system requirements. As is the case with any type of measurement method, there is inherently some error in this approach as well. Therefore, the magnitude of error was quantified by repeated measurements of the same distance, to give a confidence interval for this method of measurement. The bias error from a potentially incorrectly calibrated measuring tape was deemed to be insignificant upon inspection, and the main source of error would be accuracy and repeatability in reading the measurement from the measuring tape itself.

To quantify the error, many measurements were made for a single distance. Each measurement was recorded such that the deviation between measurements could be observed and calculated. Figure 12-1 is a histogram of 100 recorded measurements of the same distance. This type of distance measurement is a classic example for the statistical Central Limit Theorem [78] and thus this law can be used both to find the most likely true distance value as well as the confidence interval from using this distance measurement technique.

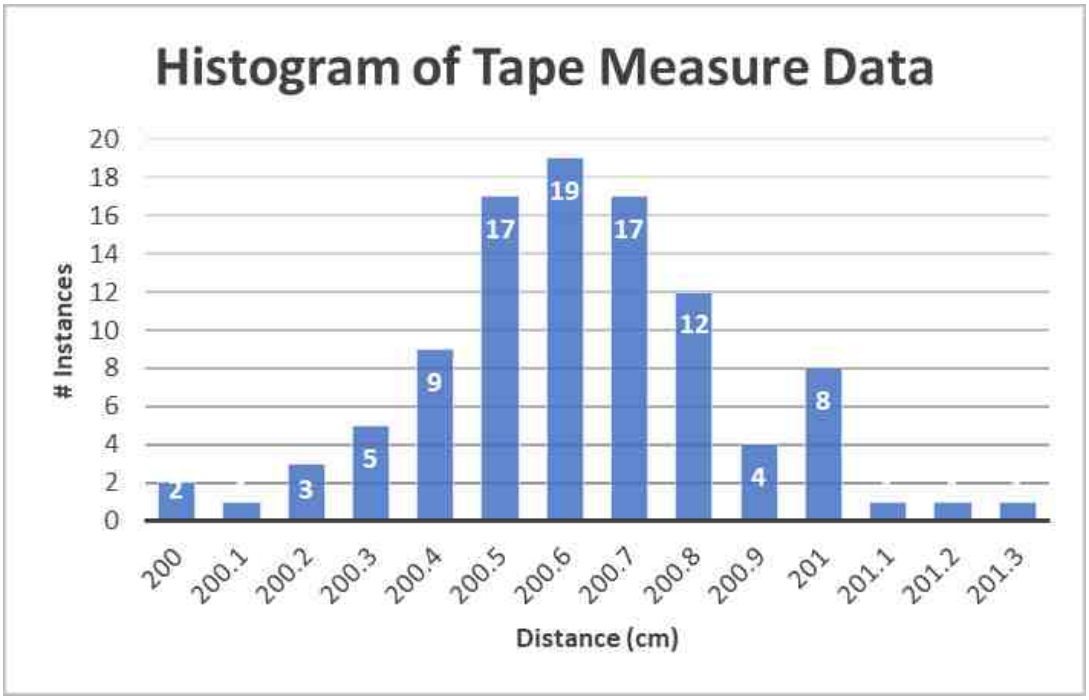


Figure 12-1: Histogram of tape measure distance measurement for calculation of confidence interval

This experiment had 100 samples with a mean measurement of 200.624 cm and a standard deviation of 0.241259 cm. The 95% confidence interval for this data set is 0.047 cm meaning that there is a high probability that the mean lies within the range of values taken.

12.4 Instrument Error

12.4.1 Reader Error

One potential source of error is the RFID radio or Bluetooth beacon error. This would be any error incurred from within the RFID or Bluetooth receiver when quantifying the RSSI or phase values of the incoming signal. While reader error would be incredibly difficult to quantify, there are some likely characteristics of this error. The first and most important for the purposes of localization is the reader error would remain constant between reads. The equipment was used well within the specified interval for additional calibration, and calibration had been performed to appropriate standards prior to use. No instances of impact, extreme temperature excursions or other incidents were noted during the course of the experiments that might call the assumption of reasonably repeatable measurement to measurement error into question.

Another expected attribute would be that the RSSI and phase values would be more accurate when the signal strength is greater. When a signal gets close to the noise floor, the radio has a difficult time distinguishing that particular signal from the ambient noise. Thus weaker signals will probably have greater error in the measurement, but would be consistent between similar measurements. This could also be partly why the trilateration comparison found greatest accuracy when simply relying on the measurements with the greatest signal strength.

For the purposes of this research it was assumed that reader error is a constant offset for all measurements, with the exception of values obtained near the noise floor, which would be those obtained at the edge of the read range for the device being measured.

Constant offsets would not impact the analysis given, regardless of size. For measurements near the noise floor, these are expected to have the same constant offset with the quantization offset superimposed. In this case this +/- 0.5dBm error for these edge of read measurements is the estimated amount attributed to both quantization and reader error near the noise floor.

12.4.2 Cable Losses

When an antenna captures an RF signal it relays the signal to the reader which then determines signal magnitude and phase shift. Therefore, any energy lost within the cable between the antenna and reader will yield a lower measured RSSI value. For a functional cable, this energy loss is directly proportional to the length of the cable, thus will be constant between measurements as long as the same cable is used. There are also some slight differences of cable losses when the transmitted frequency changes. Ultimately, given the same cable and the same frequency, cable losses should remain constant between measurements.

The cable energy loss was measured for the 6 ft. cable between the antenna and RFID reader using a signal analyzer. For each measurement the signal analyzer was calibrated using ideal open and closed conditions for comparison. The cable loss was measured at 900, 914, and 930 MHz yielding the results in Table 12-1.

Table 12-1Cable losses and antenna gain with respect to frequency of RFID system

Cable Loss and Antenna Gain

Frequency (MHz)	Cable Loss (dB)	Antenna Gain (dB)
900	-7.59	5.99
914	-7.64	5.44
930	-7.34	4.48

In addition to a drop in RSSI with cable length, the phase angle measurement will also change proportionally to cable length. This is due to the fact that the signal doesn't stop once it reaches the antenna, but continues to travel through the cable to the reader. Thus, when using phase angle to calculate distance there will be an offset which must be accounted for based upon the length of the cable between the reader and antenna. This was estimated by several different cable lengths and using phase to calculate the tag to reader separation distance in a fully anechoic chamber, then solving back for the phase offset due to the different cables. The relationship between cable length and phase distance offset was confirmed to be a linear relationship, yielding the following correlation.

$$Phase\ Distance\ Correction = 1.925 * Cable\ Length + 0.334\ [meters]$$
$$With\ Confidence\ of\ \pm 0.37\ [meters]$$

The confidence interval calculation yielded a high level of uncertainty, but this is mostly due to the small sample size from the limited supply of cables with different lengths.

For the Bluetooth devices the antenna and reader were contained within the same device, and so it wasn't possible to measure the loss between the cable and reader. However, due to the significantly reduced distance between antenna and reader, it is likely that the error is negligible in those devices. Even if the loss between the BLE antenna and reader is significant in and absolute sense, it would still have the characteristics of being constant between reads, which is the most important criteria for this study.

12.4.3 Antenna Error

Manufacturers typically provide information and specifications about antennas they sell, and one of the most important is the antenna gain [16]. The gain of an antenna is related to the directivity of the antenna and is used in equations such as path loss. It should be noted that this gain is separate from the error produced by the orientation of the antenna to the tag or BLE device, as addressed in section 12.5.2.

Some error in the published antenna gain is always expected due to manufacturing tolerances. To quantify this error, the circular polarized antenna typically used in the RFID reader system was connected to a signal analyzer. Then to calculate the gain a signal was transmitted using a signal generator and a bi-log antenna (all within a fully-anechoic chamber) as seen Figure 12-2.

Figure 12-2: Test set up to check the error in the circularly polarizing antenna

The received amplitude was compared to the transmitted amplitude at three different frequencies, and taking into account other factors such as the bi-log antenna gain, the cable losses for the 6ft long cable described in section 12-6 above, and the preamp, the antenna gain of the circular polarized antenna was calculated. The resulting circular polarized antenna gain values, with respect to frequency, are given in Table 12-1.

As was the case with cable loss, antenna gain of BLE devices is difficult to quantify because the antenna is contained within the device. However, it is assumed in this study that the antenna of the BLE device has the same error characteristics as other antennas of similar design and manufacture.

Just like many of the other sources of error, the error in antenna gain changes slightly with frequency, but is constant between reads. Given that all of the calculation in this study with calculated distance or location use experimental values rather than absolutes, slight deviations of the antenna gain from the manufacture's published information doesn't significantly impact the results of this work. Additionally, slight changes with respect to frequency play a role in analyzing the RSSI signature as discussed in Chapter 6.

12.4.4 Polarization Mismatch and Balance Error

One of the basic characteristics of any RF wave it its polarization. Electromagnetic waves can have linear, elliptical, or circular polarizations. A polarization mismatch occurs when the transmitting and receiving antennas have different polarizations. A diagram of antenna polarization is shown in Figure 12-3.

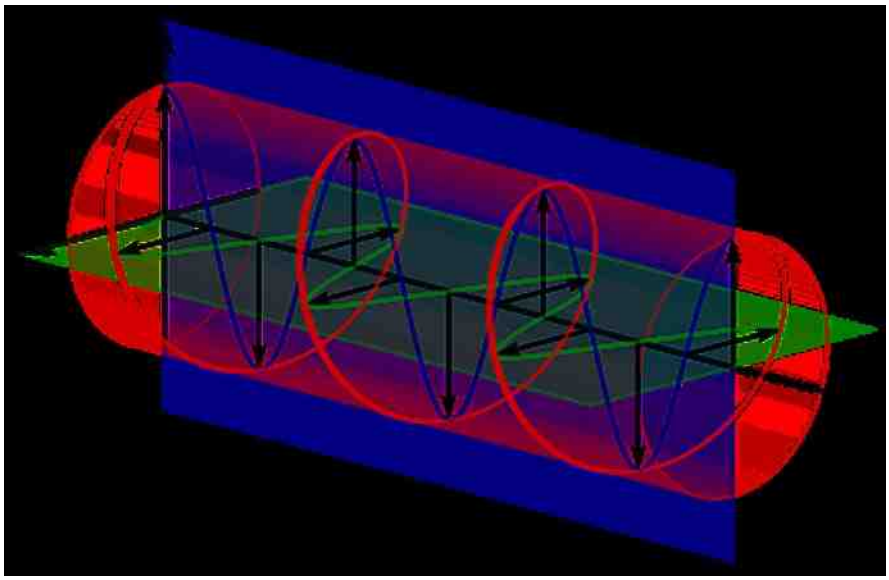


Figure 12-3: Illustration of electromagnetic circular, vertical, and horizontal polarization [79]

In the RFID system used, the reader antenna is circularly polarized while the Alien Squiglette tag is linearly polarized. Thus it is expected that there will be a polarization mismatch between the two. However, one of the benefits of using a circular polarized reader antenna is that the polarization mismatch will be constant with changes in the vertical or horizontal orientation of the RFID tag. For example, if the RFID tag is oriented vertically, giving it a vertical polarization, this should yield exactly the same polarization mismatch as if the RFID tag is oriented horizontally (or horizontally polarized). Thus, the source of inconstant error with respect to a polarization mismatch, is based upon how close the reader antenna actually is to being truly circularly polarized. If the reader antenna is perfectly circularly polarized, then there will be no changes in the polarization mismatch as the RFID tag rotates; however, if it isn't truly circularly polarized then this will introduce some error into the measurement.

Antenna balance is a characteristic of how symmetrical an antenna is. If an antenna is perfectly balanced, then it will yield the same results if turned by 180°. The imbalance is another possible source of error associated with relative antenna orientation.

While it can be quite difficult to separate the source of orientation error (true circular polarization vs. balance) from the reader antenna or tag, it is possible to quantify these combined sources of error. To do so, the typical Alien Squiglette RFID tag and circularly polarized reader antenna were used in a fully anechoic chamber. The RFID tag was read (at every hop frequency), and then rotated by 10° increments, measuring the RSSI value at each orientation. By doing so it was possible to see any error in the measurement due to a combination of polarization mismatch and antenna imbalance. The results can be seen in Figure 12-4.

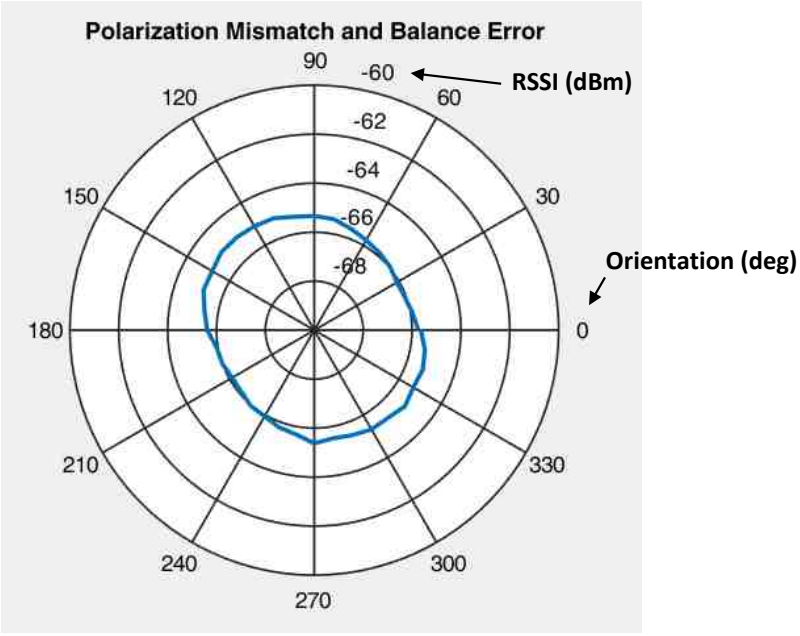


Figure 12-4: Combined polarization mismatch and balance error

Each value was calculated by taking the average RSSI value over the 50 hop frequencies. It should be noted that in Figure 12-7 the RSSI value at 0° is -65.74 dBm, while the RSSI at 360° (which should be the same) is -65.64. These measurements were repeated such that at any particular orientation the standard error was found to be 0.01 dB.

In Figure 12-4, an orientation of 0° or 180°corresponds to a vertical polarization of the RFID tag, and an orientation of 90° or 270° corresponds to a horizontal polarization. Thus, the balance of the antennas can be analyzed by comparing the RSSI at 0°/360° to 180°, and the RSSI at 90° to 270°, or any measurements 180° apart. When each RSSI value was compared to the value 180° away, the standard error from the difference between the two measurements was found to be 0.03 dB. This indicates that there is in fact a small amount of error from antenna imbalance; with a standard error magnitude of approximately 0.03 dB.

The polarization error can be observed by the non-uniform, and more elliptical shape of the plot in Figure 12-4. If the other possible variables are isolated as intended, then this non-circular shape should be the error associated with the reader antenna not being perfectly circularly polarized. The standard error due to this non-uniformity was found to be 0.06dB, from the difference between the widest and narrowest point; then by subtracting out the general measurement error of 0.01 dB found above. The result is a standard polarization mismatch error of 0.05 dB.

For the BLE system there will absolutely be polarization mismatches with changes in orientation, due to the fact that both transmitting and receiving antennas are horizontally (linearly) polarized.

12.4.5 Impedance Mismatch to Reader

In order for a reader to transmit or receive the maximum amount of power to or from the antenna, the impedance of the reader and reader antenna must be the same. If this is not the case, there will be some power loss due to the impedance mismatch between the antenna and the reader.

Both the M6e ThingMagic RFID reader [49] and the MT-262013/TRH/AK circular polarized antenna [16] are listed as having impedances of 50 ohms, therefore theoretically they should have zero impedance mismatch. While it would be extremely difficult to measure the reader's impedance, the circular polarized antenna was measured at 46.6 Ω at 918.4 MHz (approximately the center of the frequency range). The frequency dependence of the antenna impedance can be seen in Figure 12-5.

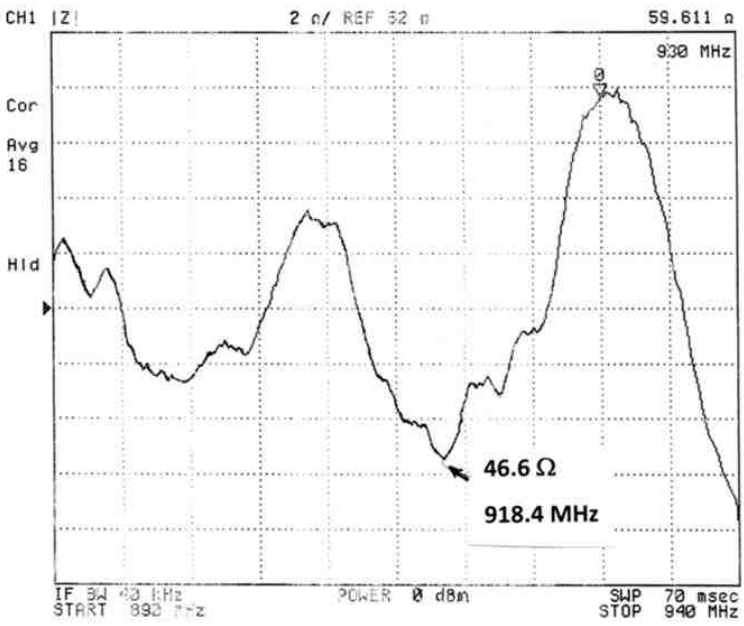


Figure 12-5: Impedance measurement of the circular polarizing antenna as a function of frequency.

In Figure 12-5 the 918.4MHz reading represents the approximate center of the frequency band for RFID in the United States. At that frequency the measured impedance was slightly under the rated specification of 50 Ω. This however would also be a source of error consistent between readings, and therefore should not impact these studies.

12.4.6 Power and Frequency Output Radio

One possible source of error would be the reader transmitting at inconsistent amplitudes. The peak of power for each transmission is very close to the 20 dBm mark (top of the scale of the graph in Figure 12-6). This is expected as the measurement system had a 10 dBm attenuator which lowered the reading from the 30 dBm actual output power. The power output is very uniform across frequencies.

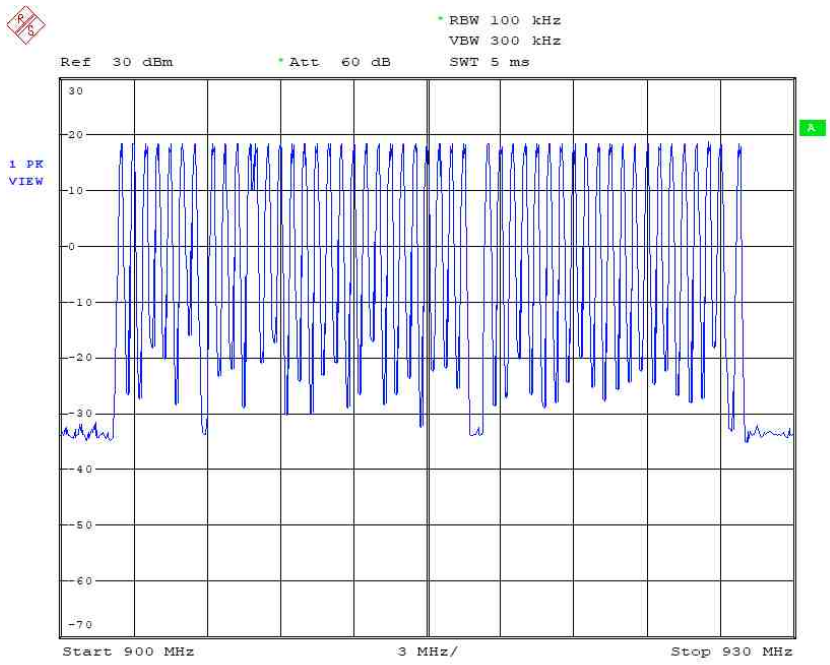


Figure 12-6: Spectrum analyzer measurement of RFID radio output as a function of frequency.

The goal of this test was to confirm power output of the radio over the frequency spectrum. A spectrum analyzer was used to measure power output from the system. Measurements taken of the system suggest that the radio is capable of producing the desired power output independent of the hop frequency.

12.4.7 Frequency Response of Tag of BLE Device

Another source of error is from the frequency response of the RFID tag or Bluetooth device. When excited by the radio at a given frequency, the device responds with an efficiency which is impacted by small mismatches and losses within each device. Unlike the RFID reader, where the antenna can be analyzed separately from the radio, for the RFID tags and BLE devices the IC chip or radio is imbedded into the circuit with the antenna. Thus the radio and antenna must be analyzed as a whole.

The frequency response of an RFID tag or BLE device is an interesting problem because, as was demonstrated in Chapters 6, the RSSI vs. frequency response of the BLE device or RFID tag relies mostly on the surrounding physical environment. Even in a fully anechoic chamber, changes in the separation distance or positioning of the transmitter and receiver can change the RSSI signature.

The following experiment was performed to investigate frequency response as shown in Figure 12-7 below. The fully-anechoic chamber is a 5 meter long chamber with acoustic deadening on all 6 walls. The antenna was placed on a raised stand and the target tag was placed at the same elevation of 1 meter above the ground. In Figure 12-7 the circularly polarizing antenna is seen as the square device on top of a stand. Data was taken at a several distances and angles between the tag and the reader antenna. The maximum

standard deviation in this data set was found to be +/- 0.4dB, occurring at the lowest measured frequency.

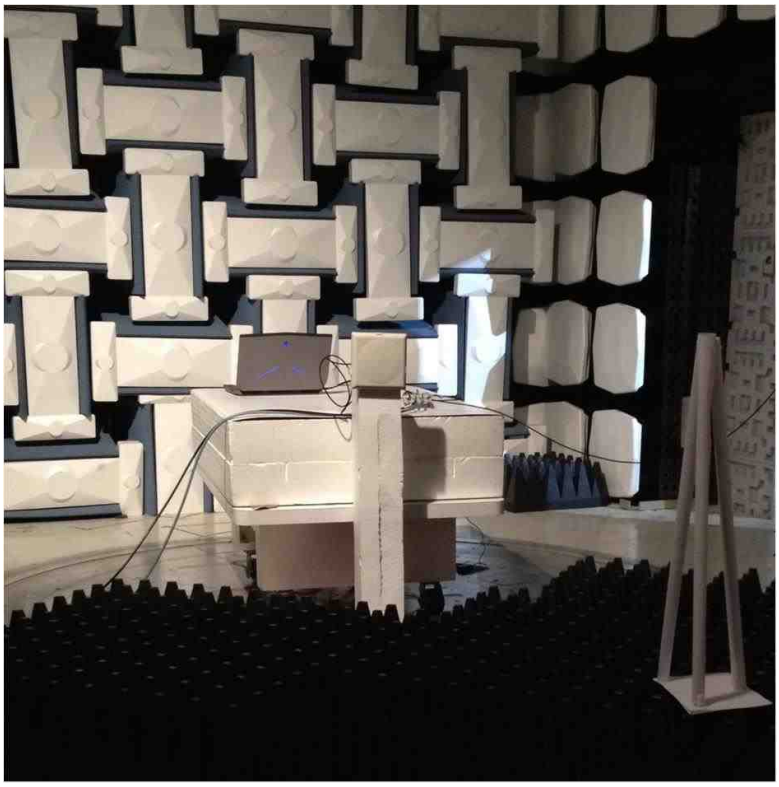


Figure 12-7: RFID experimental set-up

If all RSSI signatures measured in the fully-anechoic chamber are analyzed together, the possible error due to the frequency response of the tag (or BLE device) becomes a little clearer. Figure 12-8 shows the average RFID RSSI Signature along with the range at each frequency; similarly Figure 12-9 illustrates the same for BLE devices.

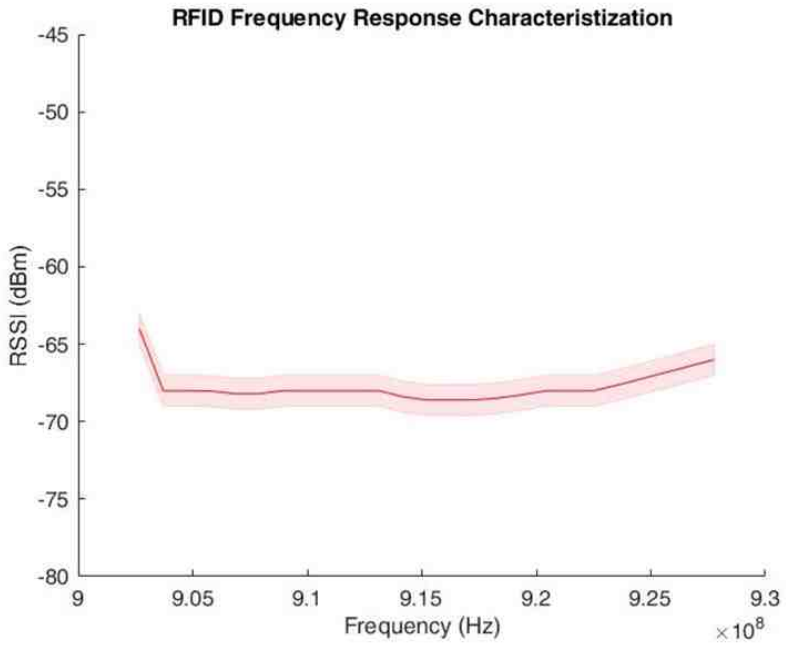


Figure 12-8: RFID RSSI signature range for measurements in a fully anechoic chamber

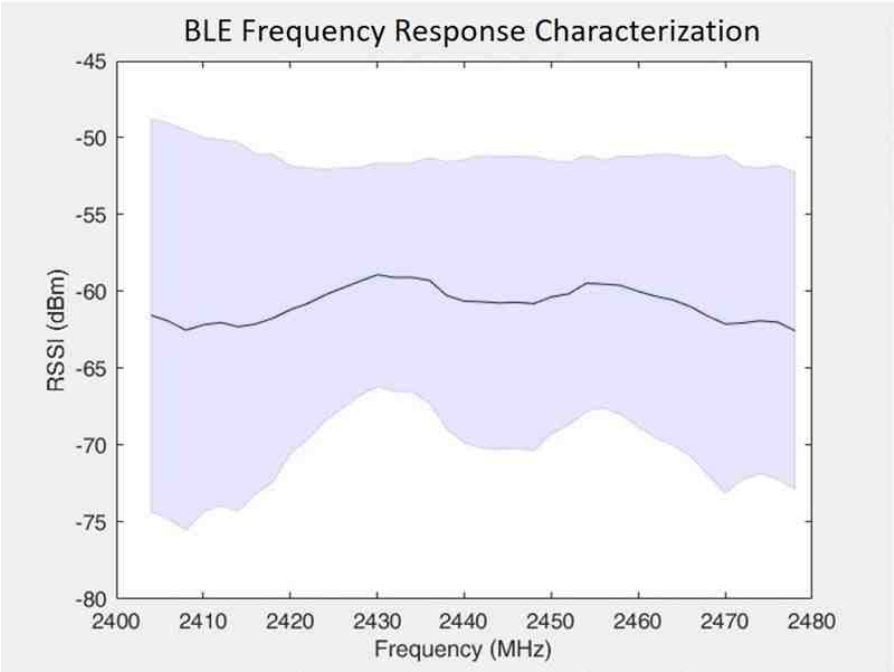


Figure 12-9: BLE RSSI signature variation in a fully anechoic chamber

The data collected represented many data points that include extreme antenna angles and distances. The fact that the signal varied shows the importance of finding a good localization algorithm. The frequency response issue is reflected by the variation of the mean. Ideally the dark line in the center of Figures 12-8 and 12-9 would be flat, straight lines. The up and down variation of that line represents the frequency response error of the device.

In both the BLE and RFID systems it is difficult to distinguish between the true frequency response of the tag or BLE device and the impact of the surrounding environment. Although these measurements were taken within a fully-anechoic chamber, this research has shown that even relatively small changes in the surrounding environment can noticeably impact the resulting RSSI signature. In both cases however, the general response of the system to changes in frequency is minimal for the devices themselves.

There is a noticeable difference between the two systems in the range of signals that were recorded. RFID systems have a more limited range of readings due to their passive nature. The highest signal they can return is limited by the power from the radio. The low end of the range is also restricted because at low signal strengths the RFID response drops out to no reading. This results in a narrow range across all of the frequencies. At the extremes of the designed frequency range the frequency sensitivity becomes more pronounced for this system. The powered nature of the BLE does not limit the range of response strengths across the experiments to the same extent, however the frequency response characteristic is still relatively flat.

12.5 Environmental Errors

12.5.1 Ambient RF Noise Error

Background RF radiation could potentially add error to measurements taken in real (non-anechoic chamber) environments. To measure the impact of ambient RF signals on RSSI values, two experiments were performed. For the first experiment a spectrum analyzer was used to measure variations in ambient RF noises, and simultaneously the RSSI signature was recorded (measuring RSSI at each hop frequency). It was found that even with significant fluctuations in the surrounding RF signals at the same frequency, the RSSI signature remained nearly unchanged, as can be seen in Figure 12-2. The small variation could be due to ambient RF noise but also likely impacted by discritization, since even the small changes in RSSI values could appear larger as they switch between two integer values. Essentially, if an RSSI value was truly half way between to integer values, then the measured discretized RSSI would be unstable and recorded at either of the nearest whole dBm values, yielding an apparent change in reading.

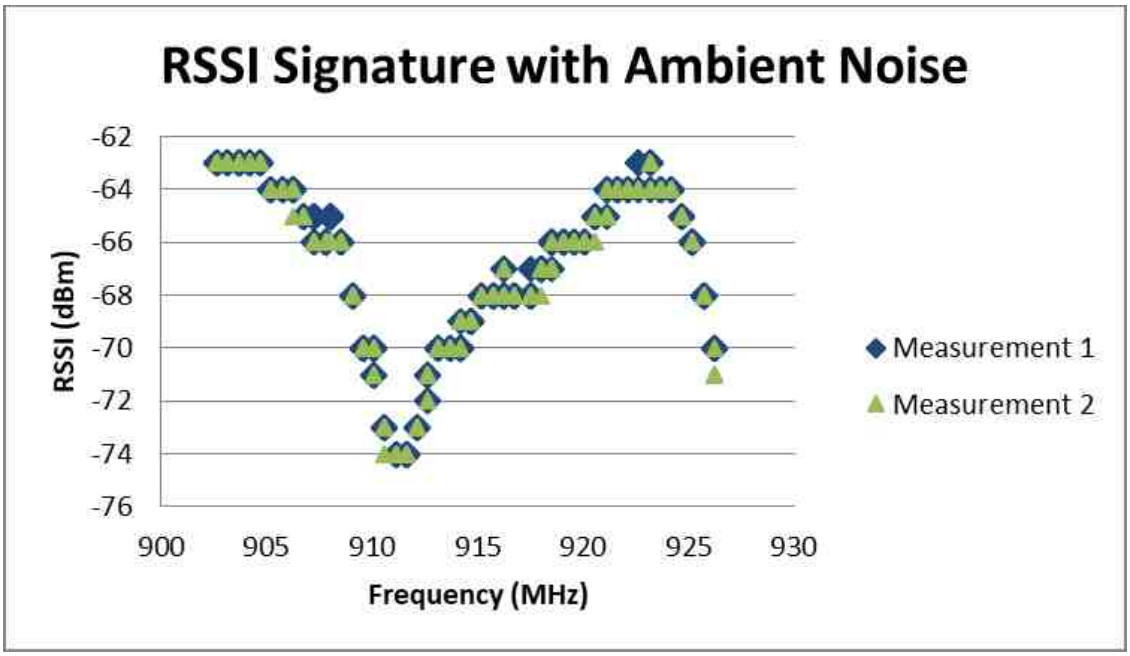


Figure 12-10: RSSI signatures in varying ambient RF noise.

In Figure 12-2 the signal varies in power level as a function of frequency, but random background RF signals which were verified by a spectrum analyzer did not seem to impact reading repeatability. While most of the data points are duplicates, discretization could account for the few which are not.

As part of the work exploring RSSI Signatures for Chapter 6, Bluetooth device output was measured as a function of frequency and ambient RF noise was investigated. For this situation it was easier to control the ambient RF signal by purposefully producing a Wi-Fi signal which occupies the same frequency band as Bluetooth. Figure 12-11 is a diagram showing the overlap between the Wi-Fi channels and the Bluetooth hopping frequencies, while Figure 12-12 shows the Wi-Fi band.

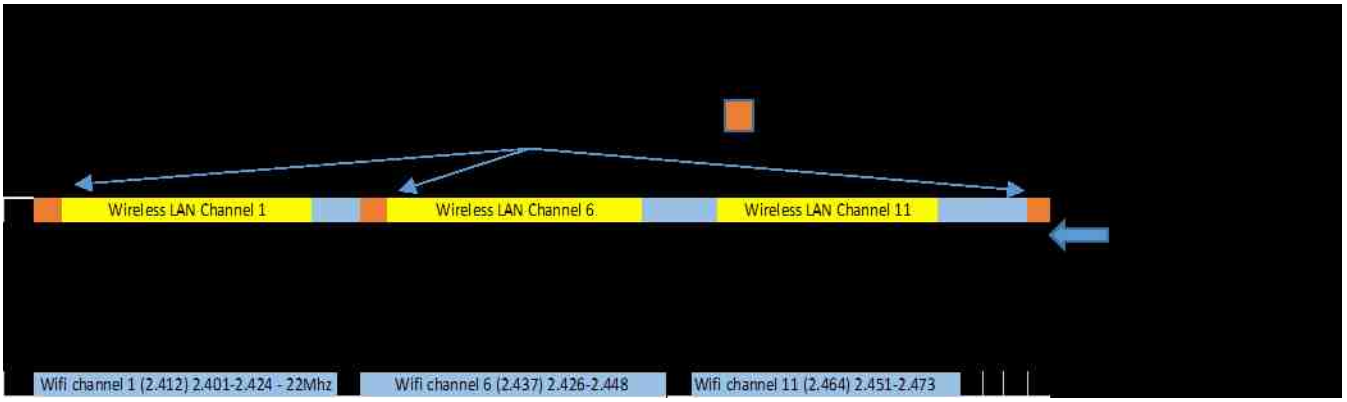


Figure 12-11: Bluetooth and Wi-Fi band overlap

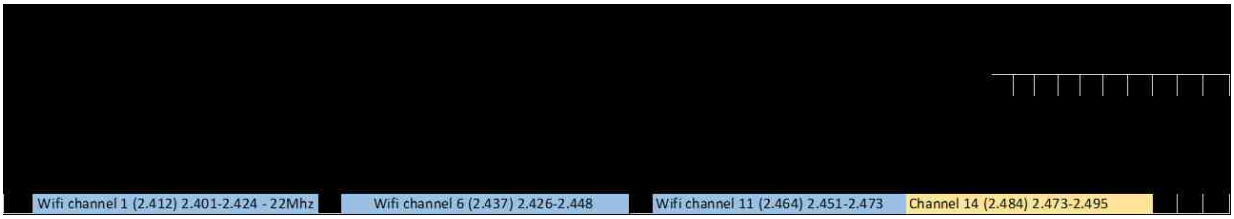


Figure 12-12: WiFi Band for United States usage.

A continuous Wi-Fi data exchange was initiated at each of the channels 2, 4, 6, 8 and 10 as shown in Figure 12-11, while at the same time the Bluetooth RSSI signature was recorded, measuring every hop frequency. This was also repeated without Wi-Fi broadcasting for a control measurement. In each instance the Bluetooth RSSI signature was almost exactly the same as the control, with the exception of the portion where it overlapped with the Wi-Fi channel. In the frequency band where the two transmissions overlapped, instead of significantly changing the measured RSSI value of the Bluetooth device, the Bluetooth simply did not read at several of those frequencies. The results of the Bluetooth RSSI signature when measured simultaneously with Wi-Fi channel 8 broadcasting can be seen in Figure 12-13.

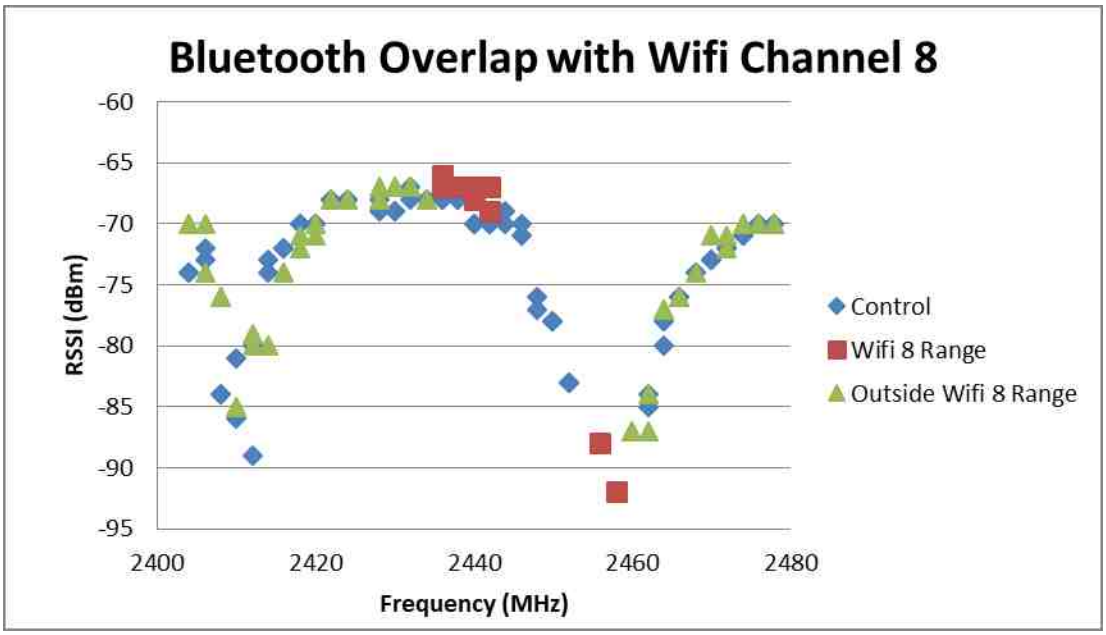


Figure 12-13: Bluetooth RSSI signature measured simultaneously with Wi-Fi Channel 8 broadcast. The blue symbols represent data taken in an anechoic chamber with no background RF.

It can be observed that the Bluetooth RSSI signature remains very close to the control measurement even within the Wi-Fi channel 8 range. RSSI readings vary in intensity as a function of frequency, but background RF does not impact the measurement value other than where Wi-Fi noise prevents a reading. There are, however, several BLE hop frequencies within that range which simply drop out. This is an important finding because it means that when there is significant impact from ambient RF, the receiver will not read at that frequency, rather than reporting a skewed value. This means measured values are more reliable. For the two experiments performed, RF noise error was found to be negligible for the measurement values reported by the readers.

12.5.2 Directivity and Pointing Error

Directivity is a characteristic of an antenna, which describes how uniform the radiation pattern is. In effect it is the power density of the peak in relation to the mean power density at any set distance from the antenna. While directivity itself is not an error but a characteristic of an antenna, error from changes in antenna orientation or angle will be related to the directivity of the antenna being moved.

Pointing error is the angular offset from the main lobe of an antenna. Zero pointing error would mean that the antenna's main lobe is directly pointing at the target location. For non-isotropic antennas, as the pointing error increases, the main lobe begins to point away from the target, and thus the theoretical gain of the antenna is no longer applicable. Instead, the target will see a drop in power related to the directivity of the transmitting antenna. Figure 12-14 graphically demonstrates how pointing error can impact the signal strength of a measurement.

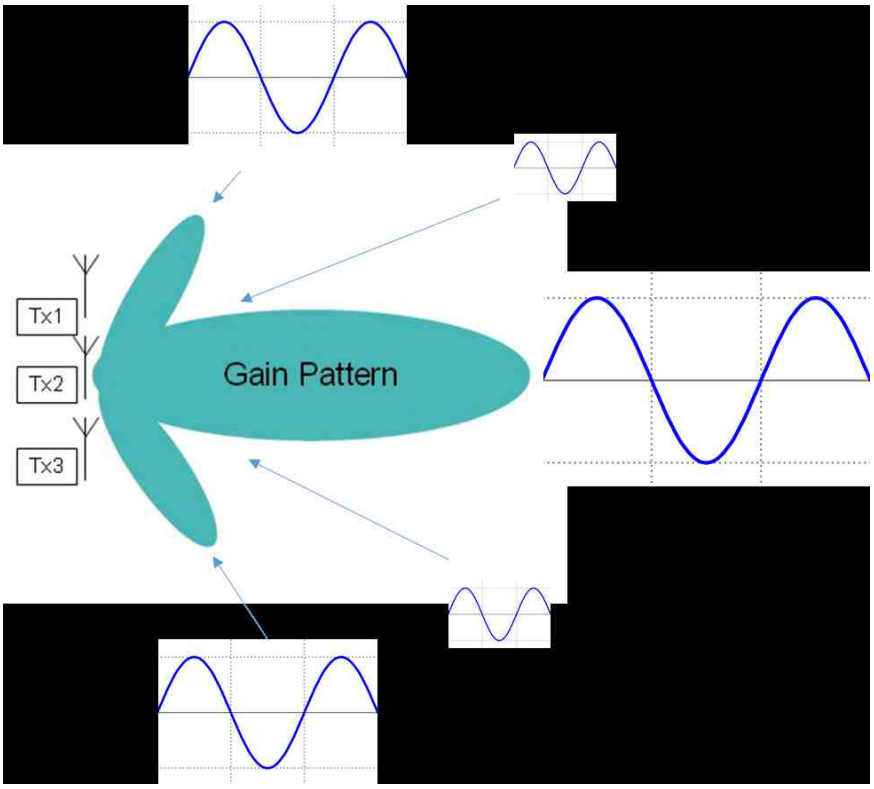


Figure 12-14: Illustration of pointing error

In order to calculate the error due to antenna orientation, the first step is to find the standard pointing error. Using the pointing error and the directivity of the antenna, it's possible to calculate the standard error from changes in antenna orientation.

Since the pointing error cannot be mitigated in real-world situations, methods (specifically the Antenna Pattern method) in this research allowed for pointing error to be compensated for in some way. One of the goals of this work was to find ways to compensate for variations in alignment between the target and the antenna, in the developed algorithms.

12.5.3 Environment and Multipath

Much of this research discussed error due to multipath of the surrounding physical environment and ways to compensate for the error, or overcome it through optimization. The work described demonstrated how environment and multipath error can be characterized using only the RSSI signature to reduce the error in localization algorithms.

Information contained in signal variations was found to contain repeatable patterns that could be analyzed by machine learning algorithms to result in corrections in distance measurements. Based on the assumption that the RSSI signature can be used in combination with a trained neural network to extract error due to multipath, it was found that the standard error for multipath in the BLE system was 0.113 meters, and 0.033 meters for RFID.

CHAPTER 13. CONCLUSION

13.1 Overview

The work presented in this document covers an exploration into localization technologies based on UHF RF devices. The goal of the work was to identify and potentially improve upon the state-of-the-art technologies for practical RF based localization for applications to indoor environments. From this effort has come innovations in the areas of RF signal processing, multipath error reduction, and localization algorithms for UHF devices.

This chapter will review the contributions made to localization through this work.

13.2 Improvements in RF signal processing

RSSI is the standard method for determining a distance based on a tag/reader pair. This method uses the drop in signal strength as a function of distance between tag and reader to calculate separation distance. Multipath signals are a significant source of error in real-world settings, resulting from any non-RF transparent object such as furniture, walls, people and devices. These are incredibly prevalent in modern offices, warehouses, hospitals, etc., and serve as barriers and reflective surfaces for RF waves. Phase distance provides more precise information about distance as it is more robust with respect to multipath; however all phase measurements require additional information (often requiring more time) to resolve the issue of cycle ambiguity.

The work described extends the methods of RSSI and phase by combining them in a leveraged approach that allows for a fast and robust measurement to calculate distance. The new method leverages the macro-distance benefits of the RSSI distance measurement technique with the enhanced accuracy of the phase technique. This method was found to be the most robust among RSSI and phase measurements in terms of mobile tags, high multipath, and extreme antenna angle, and was presented in [1].

13.3 Improvements in Error Estimation

Multipath error is most often the largest source of error in "real-world" distance estimation. Through studying RF measurements in this research, a relationship was found between the multipath environment and the pattern of RSSI measurements with respect to the hop frequency. This was first presented at the IEEE/ASME Advanced Intelligent Mechatronics conference in 2014 [2]. The work was expanded with a theoretical analysis of the ability to separate multipath error from a base measurement signal using the extra information provided by the RSSI Signature.

A neural network was used to demonstrate that multipath information could be separated from empirical data and used to predict the magnitude and direction of multipath error of the distance measurement. Likely application of this method would be to use big data techniques to train neural networks on a wide range of potential environments. The concept was proven on a more limited set of environments, and using the trained neural network on new test data was able to reduce the distance error estimation by approximately 50%. To verify that the neural network had truly been able to pull from the relatively small data set the fundamentals of the patterns associated with the base signal vs. the multipath,

an additional data set from a completely new environment was used to test the resulting algorithm. The algorithm was able to reduce the error for this untrained environment as well. This was reported in the journal article from IEEE Transactions on Instrumentation & Measurement "Reducing RF Distance Error by Characterizing Multipath," [3].

To improve the RSSI Signature method further, the next step was to make it practical by approximating the signature in such a way that the neural network could use the incomplete data sets. These various representations of the RSSI Signature were used for error predictions then the results were statistically analyzed to determine which curve fitting methods were viable for representing incomplete RSSI signatures. The best results were found to be very close fit approximations of the RSSI signature, but it was in fact demonstrated that the abstracted representations of the signature could be used effectively to reduce error in a distance measurement, even with incomplete data.

13.4 Improved RF Based Trilateration

The goal of the project was to determine the best localization algorithm for use in an indoor environment for locating objects such as: critical equipment, sensitive documents, tools, or even individuals. A literature search was performed and a large number of algorithms for trilateration were found; however nearly all could be categorized into approximately 10 types of methods. Interestingly two of these methods described in the literature contained significant errors, which had to be resolved in this work before the method could actually be tested. One of the contributions of this work was deriving of the Hyperbolic Locus of Position method, which was described with erroneous calculations in the article [61]. Additionally, the Centroid algorithm had to be altered to use smaller

distance estimations, which should not be neglected as they are likely the most accurate distance estimations; while at the same time adding appropriate weightings.

As the trilateration methods were compared, the way in which the methods failed was analyzed and the issue of non-uniform antenna radiation patterns was identified as key contributor. It was hypothesized that instead of overlooking this error and simply assuming an isotropic antenna, the information from the antenna radiation pattern could be incorporated into the trilateration to improve localization accuracy. For this method an approximate antenna pattern was arranged and optimized among the distance measurements from the satellites to find where it best fit, thus providing the beacon location. To make this method practical for tracking as well as localization the original numerical approach was replaced with a theoretical model, which took significantly less time to process and made it feasible for tracking. The Antenna Pattern method will be soon submitted as a journal article.

13.5 Comparison of Trilateration Accuracy

The final contribution of this work is a survey of the accuracy of all major current methods of distance estimation and trilateration for RFID and BLE technologies. The same data sets were used against all trilateration methods so a direct comparison of algorithms could be made. The various technologies and methods had different strengths, and the best localization method was different for each. The following tables (13-1 and 13-2) summarize the trilateration comparison work.

Table 13-1: Summary of results of trilateration study comparing the location accuracy of all combinations of methods. Green highlights are methods developed in this research. Yellow highlights are methods significantly improved through this work.

Results of Trilateration Study

	Technology	Best Method
Distance Estimation	RFID	Expensive - Phase
		Inexpensive- RSSI signature
	Bluetooth	RSSI signature
Data used in algorithm	RFID	3 Highest RSSI values
	Bluetooth	3 Highest RSSI values
Non-iterative trilateration algorithm	RFID	Hyperbolic Locus of Position
	Bluetooth	Centroid
Iterative trilateration algorithm	RFID	Iterative Least Squares
		Weighted Least Squares
	Bluetooth	Antenna Pattern based on Centriod

The accuracy of a given RF localization system will be a function of the choices made at each level of the design, and the environment in which the system must operate. Typical location accuracy using RFID tags is between 1.5 to 2 meters. The maximum read range for this type of device in an ideal environment of an anechoic is about 8 meters. In comparison a powered device working in the same frequency band, the Bluetooth system has an almost identical localization accuracy but can reach up to 70 meters. The testing in this study was all done indoors and was limited to 10 meters for both device types.

The absolute lowest error found in this study for RFID was using Phase distance estimation, the three highest RSSI signals and the Hyperbolic Locus of Position non-iterative method. For this experiment that combination produced an average error of 0.42 meters. Similarly, for BLE the experiment the absolute lowest localization error used the distances with the three highest RSSI values. However, all of the other levels were

different, with the lowest error for BLE resulting from a combination of the RSSI Signature for distance, the Centroid algorithm for an initial location estimation, and the Antenna Pattern method for a more enhanced final estimate. This yielded an average accuracy of 1.32 meters, which is significantly improved over the literature for a similar infrastructure. This information will be submitted soon for a journal article.

It should also be noted that the experiments were performed in a semi-anechoic chamber, and it is believed that the methods more robust to multipath (such as the RSSI-Informed Phase method) would likely perform better in comparison if similar experiments were repeated across a set of more RF complex environments.

13.6 Summary of Contributions

Through the process of working with RF signals, distance estimation methods, and trilateration algorithms, multiple new methods were developed and others were improved upon. A summary of the contributions of this work to the field of RF localization are shown in Table 13-2.

Table 13-2: Author contributions to Localization using RFID and Bluetooth technologies. More detail about each can be found in the locations listed in this chart.

Author Contributions to Localization		
Name	**Contribution**	**Covered in**
RSSI Informed Phase	Created distance estimation method robust with respect to multipath, mobile tags, and antenna angle.	Chapter 5 - RSSI Informed Phase method for RFID Distance Calculations "RSSI Informed Phase Method for Distance Calculations," in IEEE/ASME International Conference on Advanced Intelligent Mechatronics, (2015).
RSSI Signature	Created distance estimation method to significantly reduce multipath error using RSSI vs. frequency pattern and a trained Neural Network	Chapter 6 - "Reducing RF Distance Error by Characterizing Multipath," in IEEE Transactions on Instrumentation & Measurement DOI 10.1109/TIM.2018.2875899 (2018) & "Received Signal Strength Indicaiton Signature for Passive UHF Tags" (2014)
Phase Signature	Created distance estimation method, reducing in phase measurements using the RSSI Signature and a trained Neural Network	Chapter 9 - Using RSSI Signature to reduce phase error
Antenna Pattern	Created a trilateration method using antenna radiation pattern and a optimization algorithm to increase location accuracy	Chapter 11- Antenna Pattern Method of Localization (Manuscript to be submitted)
Hyperbolic Locus of Position	Reduced described model to practice, generating the mathematical rotation of a hyperbola for non-linear weighting	Chapter 10 - Methods of Trilateration
Centroid	Allowed for the incorporation of small distances and applied weighting algorithm for this method to reduce loalization error for RF systems	Chapter 10 - Methods of Trilateration
Comparision of trilateration Methods	Experimentally compared new and existing methods of trilateration and distance estimation for accuracy in a controlled environment using the same data sets. Found variations among the methods which can be used for system design.	Chapter 12 - Experimental Comparison of Trilateration (Manuscript to be submitted)

With the rapid adoption of the Internet of Things, emphasis has been placed on leveraging existing systems of devices to generate additional information such as localization of devices. In this work RFID and BLE devices have been used to leverage information in the signal to locate items in a complex environment. The main hindrances for implementation of localization are cost, time, and accuracy, all of which have been

addressed in this work. First by choosing low cost technologies, with fast response time, then selecting localization methods which require minimal time and infrastructure, and then by improving distance and trilateration algorithms to make the system more accurate and robust with respect to “real world” conditions.

APPENDICES

APPENDIX A: EXPERIMENTAL SET-UP AND DATA

This Appendix contains the data and experimental set-ups used to generate graphics and figures in the dissertation. Graphics are reproduced for reference.

A.1: RFID Experimental Set-Up

The majority of RFID data was collected in a 5-meter anechoic chamber as seen in Figure A-1, but some of the data was also collected in each of the environments described in A.3. A ThingMagic M6e radio [49] was used primarily with circular polarized antennas [16]. When specifically addressing polarization mismatch, a half-wave dipole antenna was used in place of the circular polarized antenna. An Alien Squiglette RFID tag [4] was used in a vertically polarized position, unless otherwise stated. Both RFID tag and reader antenna were consistently held 1 meter above the ground plane. RF transparent materials of paper and Styrofoam were used to support both the RFID tags and antenna.

For localization, 4 antennas were used to surround the RFID tag. And for mobile tag experiments a stepper motor pulley system was used to move the RFID tag at a constant speed through the environment.

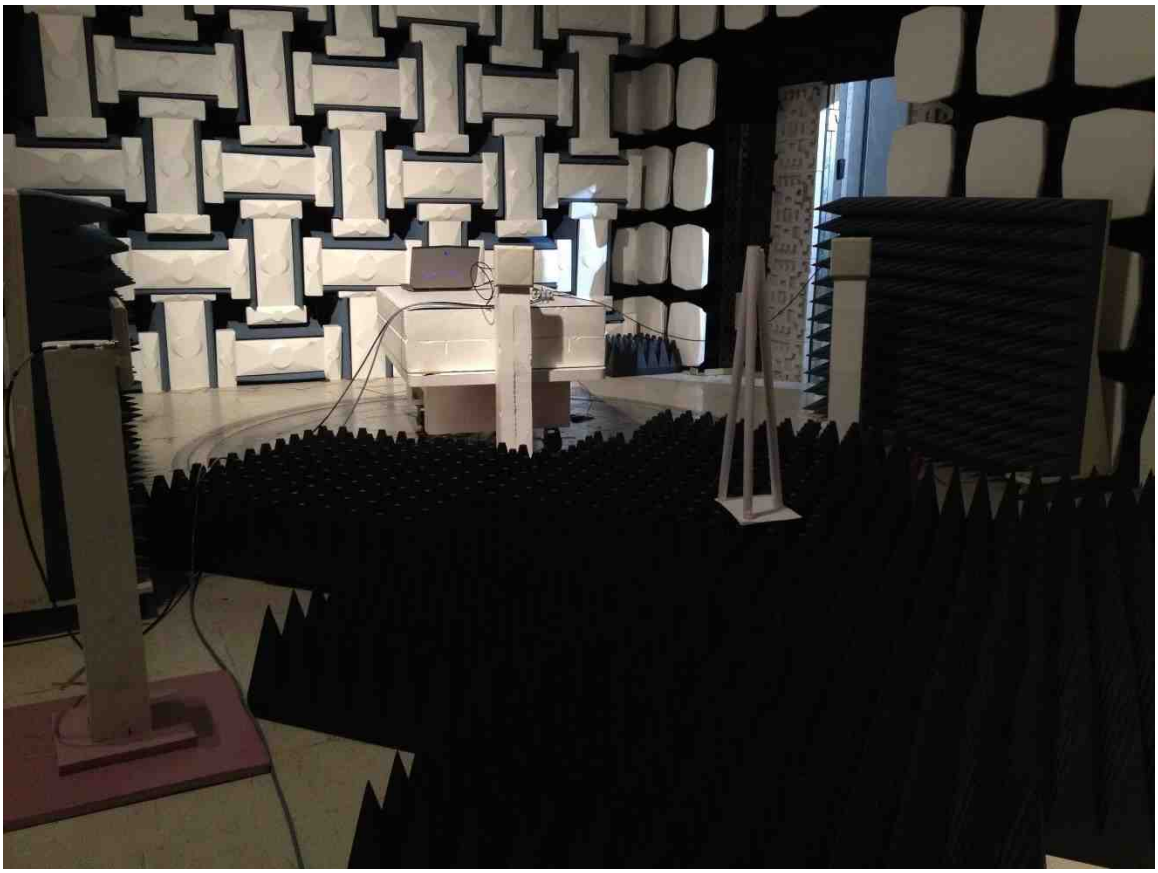


Figure A-1: Photo of RFID localization experimental setup.

A.2: Bluetooth Experimental Set-Up

Bluetooth data was primarily collected in a 10 meter anechoic chamber using Texas Instruments BLE CC Debugger Wireless Development Kit [15], as can be seen in Figure A-2. Due to the geometry of the BLE device, the data was collected in a horizontally polarized orientation. Satellites and beacons were all the same type of device for BLE data, and all of the data was collected with the instrumentation 1 meter above the ground plane on Styrofoam pillars which are RF transparent. A stepper motor pulley system was used both to collect stationary data autonomously as well as data for a mobile device. An

Arduino was programmed to control the BLE devices, record the data, control the stepper motor, and read the encoder.

Figure A-2: Photo of BLE localization experimental setup.

A.3: Experiment Environments

Table A-1: Various environments for data collection

Environments used for Data Generation

Real World Environments	RF Controlled Chamber
A. Domestic - This environment was composed of a single room with a varying assortment of domestic furniture such as upholstered chairs and wooden tables.	**D. 5 Meter Fully Anechoic Chamber** - Lexmark International
B. Office - This environment was similar to the domestic environment except the floor was left as solid concrete (no carpet) and furniture was more metal and wood tables and chairs.	**E. 5 Meter Semi-Anechoic chamber**. This was the 5 Meter Fully Anechoic Chamber with the floor absorbers removed.
C. Warehouse - This was a large commercial room with stored electronic equipment in boxes.	**F. 10 Meter Semi-Anechoic Chamber** - Lexmark International

Data for distance measurements was acquired in a variety of environments to improve the generalizability of experimental results. These environments are broken into two categories: Real World and Chamber.

The following table shows a general description of these environments. This code will be referred to in this Appendix to identify the environment in which the data was taken.

A.4: Specific Experiment Data

A.4.1- Fingerprinting Data Chapter 2

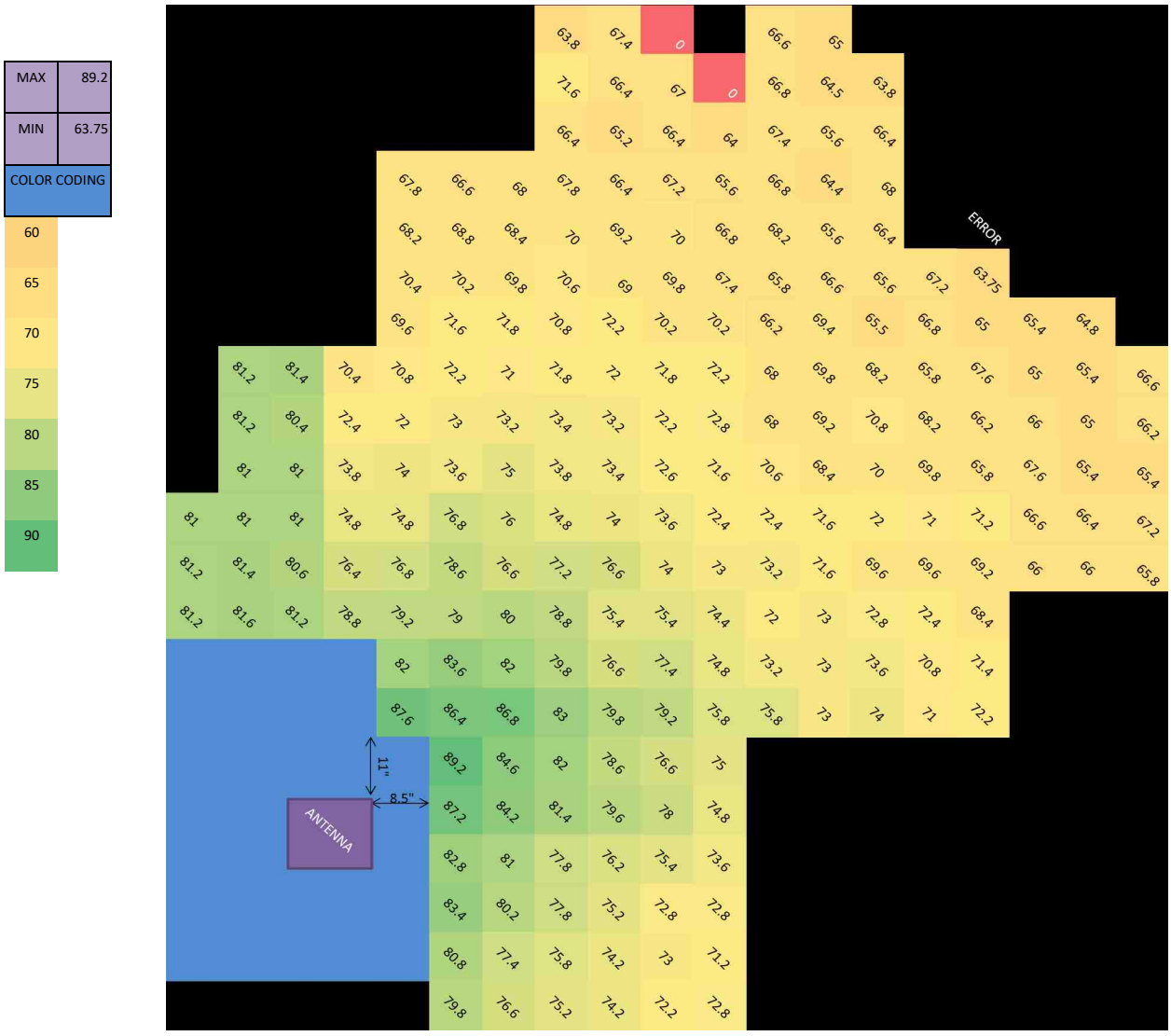


Figure A-3: The data in each square represents the data taken for beacons in every point on the grid. (Reproduction of Figure 2-5)

Table A-2: Additional experimental information used to generate Figure 2-5

Experiment Information
2/25/2013
Chamber Environment D
Half Wave Dipole Antenna
M5e ThingMagic Radio
H4 Squiggle White Wet RFID Tag
Reading RSSI
Each increment of RSSI ≈ 1.2dB
Vertical Polarization
Transmit Antenna 39.75” above ground plane

A.4.2- RSSI vs. Separation Distance

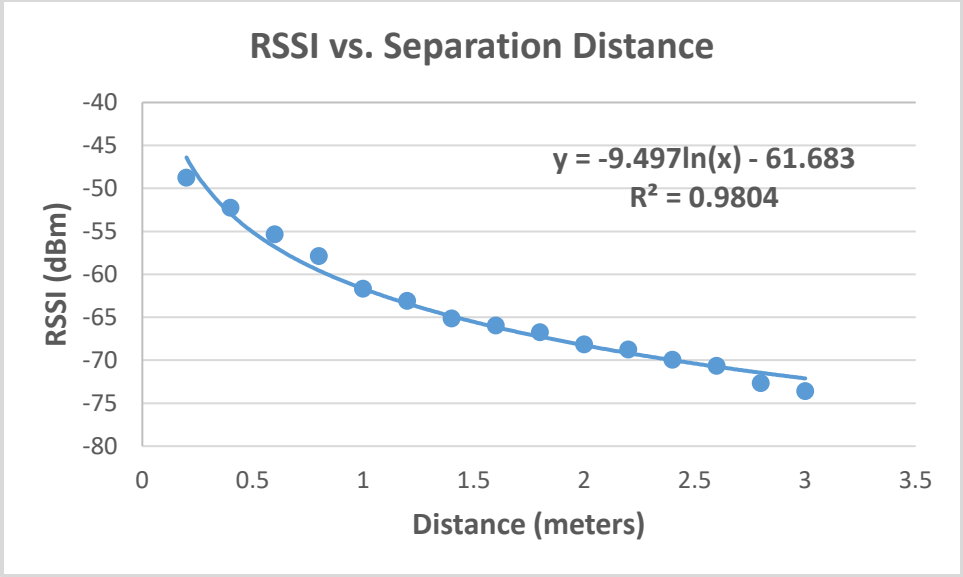


Figure A-4: Empirical RSSI vs. Separation Distance (Figure 3-1, 5-1 reproduced)

Table A-3: Experimental Set-up used to produce Figure 3-1, 5-1

Experiment Information
6/8/16
Chamber Environment D
Circularly Polarized Antenna – 1 Meter Above Ground Plane
Alien Squiglette – 1 Meter Above Ground Plane
Thing Magic M6e Radio
Measuring RSSI with Changing Separation Distance

Table A-4: Experimental Data used to produce Figure 3-1, 5-1

Dist (m)	RSSI (dBm)	Dist (m)	RSSI (dBm)
0.4	-52.29	**2**	-68.17
0.8	-57.91	**2.4**	-69.96
1.2	-63.12	**2.8**	-72.68
1.6	-65.97		

A.4.3- RFID Free Space and Single Ground Plane Reflection

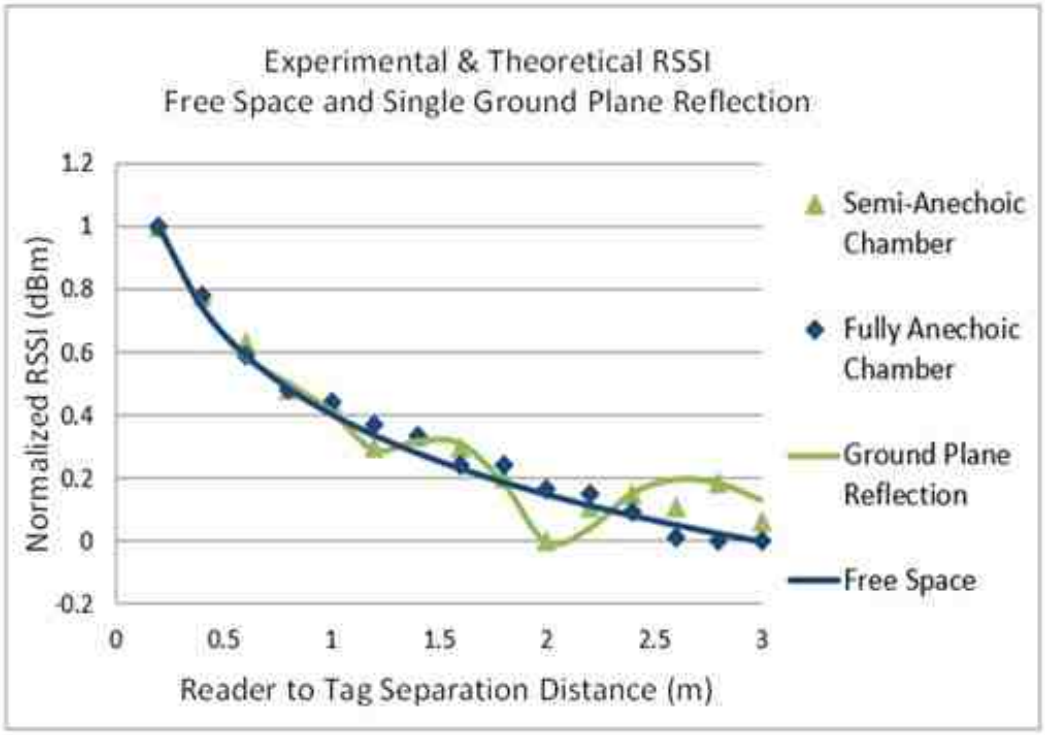


Figure A-5: Empirical RSSI vs. Separation Distance (Figure 3-4, 6-3 reproduced)

Table A-5 Experimental Set-up for Figures 3-4 and 6-3

Experiment Information
2/10/2014
File Name: Ground Plane Comparison with Half Wave Dipole.xlsx
RF Controlled Chambers D & E
Half Wave Dipole Antenna – 1 Meter Above Ground Plane
Alien Squiglette – 1 Meter Above Ground Plane
Thing Magic M6e Radio
Measuring RSSI with Changing Distance
With and Without Absorbers on Anechoic Chamber Floor
Phase linearization Chapter 5, Figures 5-3 and 5-4

A.4.4 – RSSI Returned vs. Transmitted Power from Radio

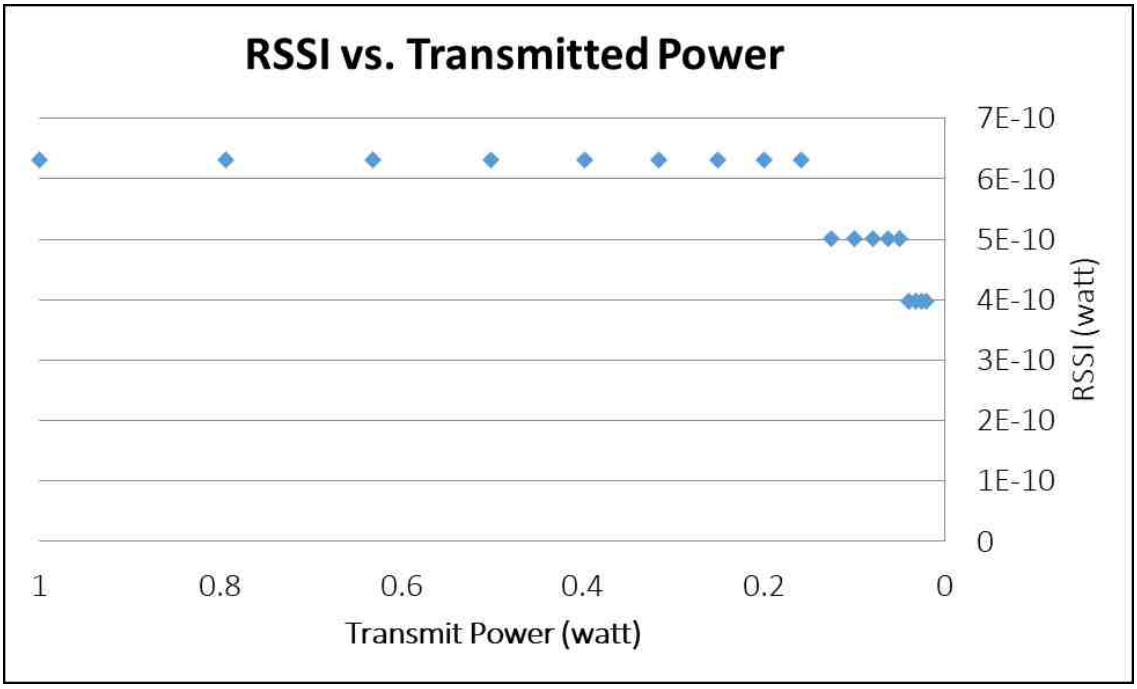


Figure A-6: RSSI remains constant for a given distance of tag to radio, even as transmitted power from the radio is reduced. (Figure 4-7 reproduced)

Table A-6: Experimental set-up for RSSI vs. Transmitted Power (Figure 4-7)

Experiment Information
7/1/2013
File Name: Many Locations 5 M Chamber.xlsx
RF Controlled Environment D
35.5" Tag to Ground Plane
35.5" Circular Polarized Antenna to Ground Plane
14" Circular Polarized Antenna to Tag
Alien Squiglette Tag Vertically Polarized
ThingMagic M6e Radio

A.4.5- Measuring RFID Tag Signal

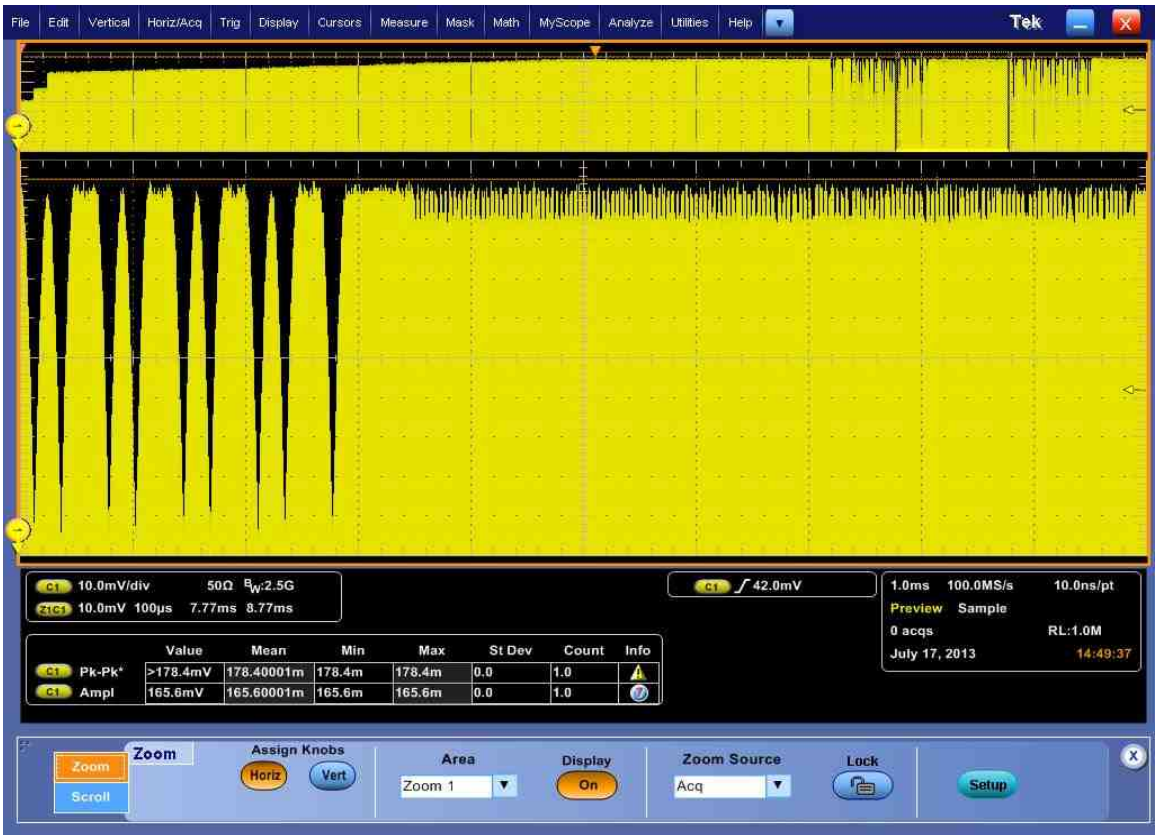


Figure A-7: Oscilloscope trace of radio to tag communication (Figure 4-6 reproduced)

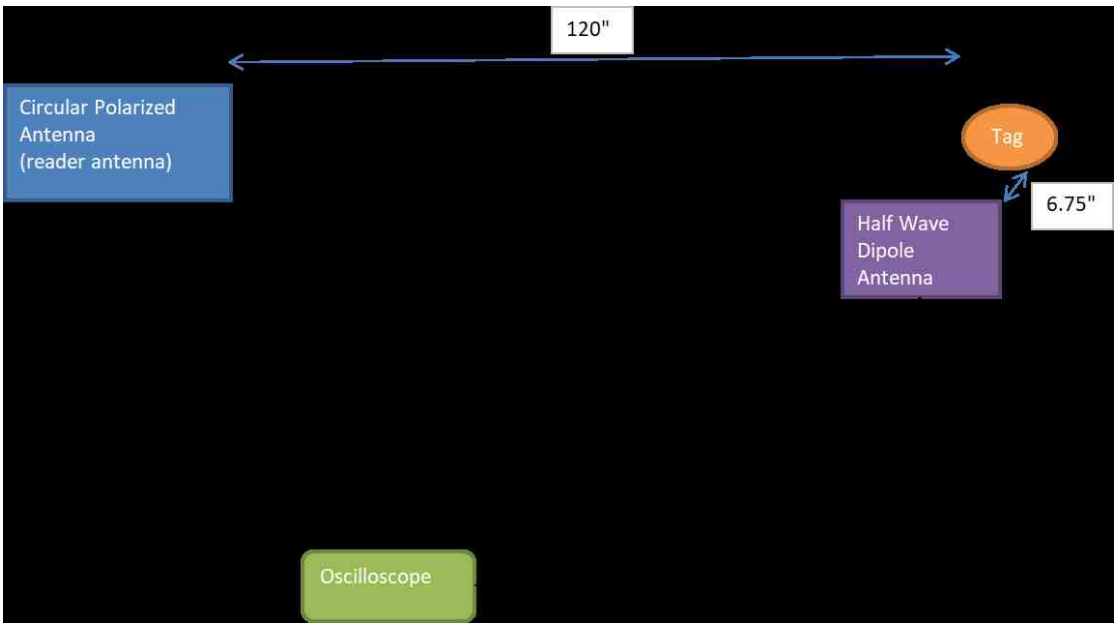


Figure A-8: Diagram of experimental set-up

Table A-7: Experimental set-up for RFID tag signal measurement (Figure 4-6)

Experiment Information
7/17/2013
File Name: Finding Tag's Signal Using Oscilloscope.xlsx
RF Controlled Environment D
35.5" ground to circular polarized antenna for ThingMagic M6e radio
35" ground to Alien Squiglette Tag
6.75" tag to half wave dipole antenna for oscilloscope
120" tag to circular polarized antenna (reader)

A.4.6- RSSI vs. Distance for Many RFID Tags

Figure A-9: Data collected on RSSI as a function of distance follows a one-way propagation model. (Figure 4-9 reproduced)

Table A-8: Experimental set up for one way vs. two way propagation (Figure 4-9)

Experiment Information
4/30/2014
File Name: Different tag.xlsx
RF Controlled Chambers D (5 Meter Fully Anechoic)
Half Wave Dipole Antenna – 1 Meter Above Ground Plane
Thing Magic M6e Radio
RFID Tags – 1 Meter Above Ground Plane
Measuring RSSI with Changing Distance
Tags Measured: Squiglette, Short White, Hammer, and Rafsec

A.4.7- Collected Phase Angle Data

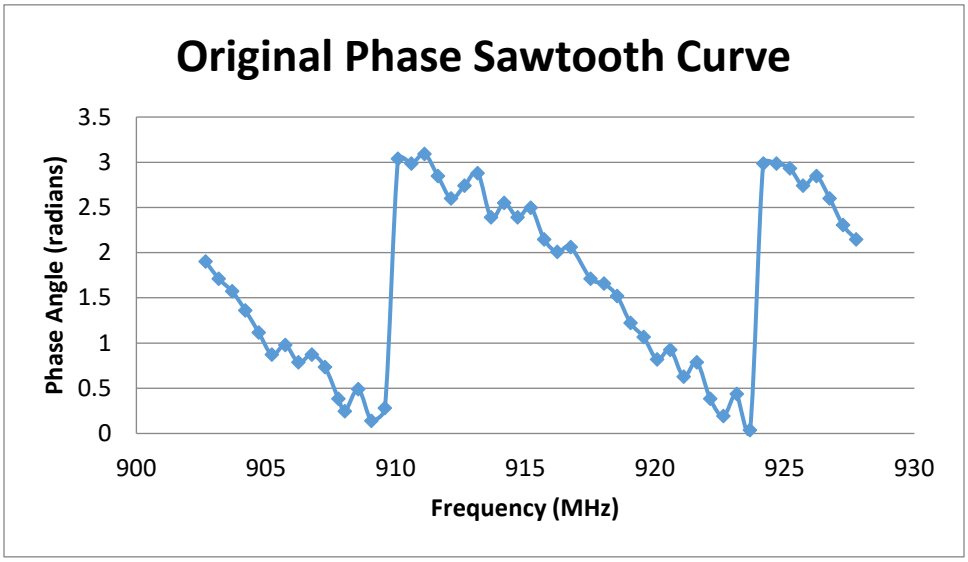


Figure A-10: Phase Angle Sawtooth Curve (Figure 5-3 reproduced)

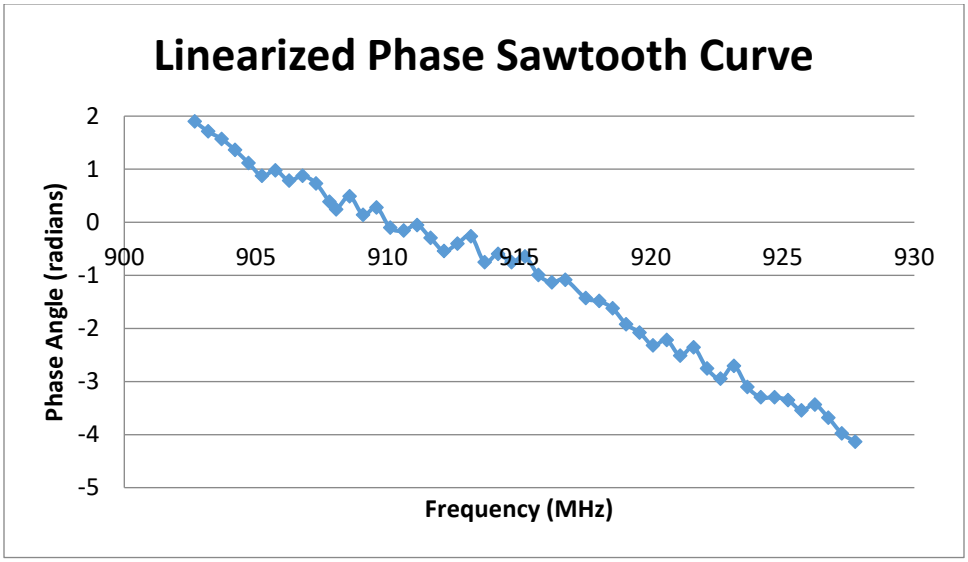


Figure A-11: Phase Angle Linearized Curve (Figure 5-4 reproduced)

Table A-9: Experimental Set-up for Figures 5-3 and 5-4

Experiment Information
7/9/13
Real World Environment A
Circularly Polarized Antenna – 1 Meter Above Ground Plane
Alien Squiglette – 1 Meter Above Ground Plane
Thing Magic M6e Radio
Measuring RSSI with Changing Separation Distance

Table A-10 Experimental data used to create Figures 5-3 and 5-4

Experimental Data		
Frequency (MHz)	Phase (radians)	Linearized Phase (radians)
902.669	1.902409	1.902409
903.181	1.710423	1.710423
903.693	1.570796	1.570796
904.205	1.361357	1.361357
904.717	1.117011	1.117011
905.229	0.872665	0.872665
905.741	0.977384	0.977384
906.253	0.785398	0.785398
906.765	0.872665	0.872665
907.277	0.733038	0.733038
907.789	0.383972	0.383972
908.045	0.244346	0.244346
908.557	0.488692	0.488692
909.069	0.139626	0.139626
909.581	0.279253	0.279253
910.093	3.036873	-0.10472
910.605	2.984513	-0.15708
911.117	3.089233	-0.05236
911.629	2.844887	-0.29671
912.141	2.600541	-0.54105
912.653	2.740167	-0.40143
913.165	2.879793	-0.2618
913.677	2.391101	-0.75049
914.189	2.548181	-0.59341
914.701	2.391101	-0.75049
915.213	2.495821	-0.64577
915.725	2.146755	-0.99484
916.237	2.007129	-1.13446
916.749	2.059489	-1.0821
917.518	1.710423	-1.43117
918.03	1.658063	-1.48353
918.542	1.518436	-1.62316
919.054	1.22173	-1.91986
919.566	1.064651	-2.07694

920.078	0.820305	-2.32129
920.59	0.925025	-2.21657
921.102	0.628319	-2.51327
921.614	0.785398	-2.35619
922.126	0.383972	-2.75762
922.638	0.191986	-2.94961
923.15	0.436332	-2.70526
923.662	0.034907	-3.10669
924.174	2.984513	-3.29867
924.686	2.984513	-3.29867
925.198	2.932153	-3.35103
925.71	2.740167	-3.54302
926.222	2.844887	-3.4383
926.734	2.600541	-3.68264
927.246	2.303835	-3.97935
927.758	2.146755	-4.13643

A.4.8- RSSI-Informed Phase with Mobile Tag

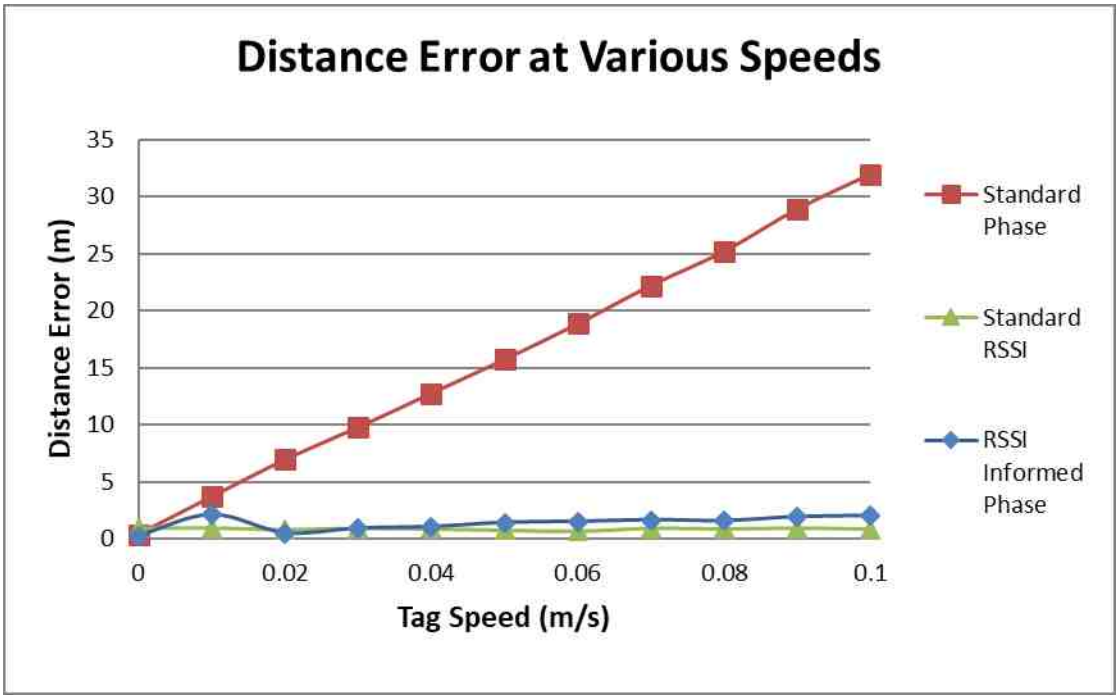


Figure A-12: A comparison of method distance error as a function of tag movement (Figure 5-6 reproduced)

Table A-11: Experimental setup for Figure 5-6

Experiment Information
5/12/2015
File Name: RSSI Informed Phase Comparison for AIM 2015.xlsx
RF Controlled Environment E
Circularly Polarized Antenna – 1 Meter Above Ground Plane
Alien Squiglette – 1 Meter Above Ground Plane
Thing Magic M6e Radio
NEMA-17 stepper motor for moving the tag

A.4.9- RSSI Signature Data

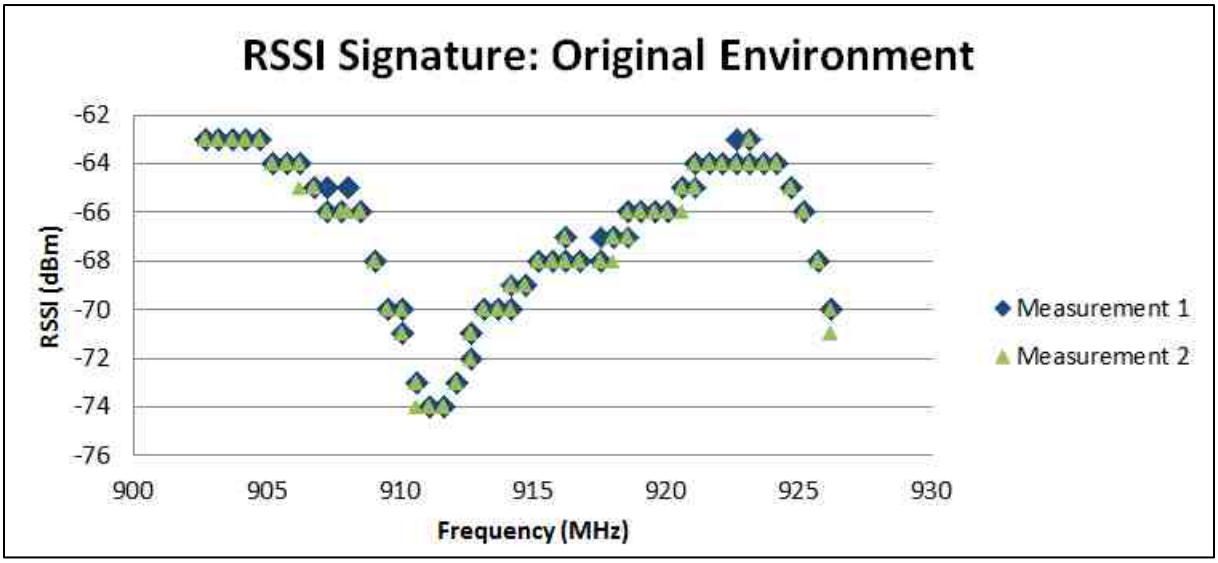
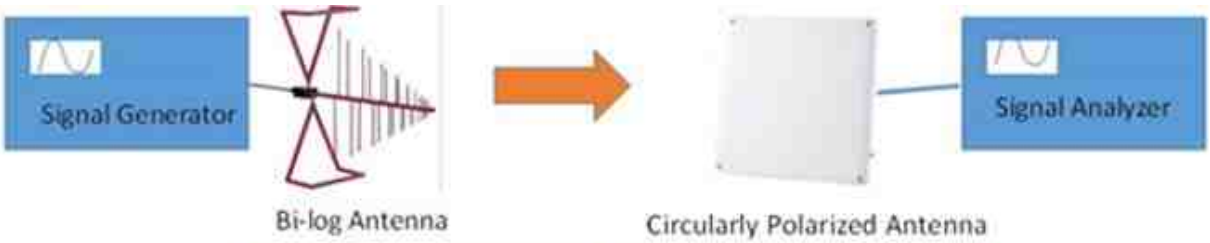


Figure A-13: Measured RSSI Signature (Figure 6-1 and 7-1 reproduced)

Table A-13: Experimental setup for Figure 6-1, 7-1

Experiment Information
9/30/13
File Name: Spectrum Analyzer.xlsx
Real World Environment C
Circularly Polarized Antenna – 1 Meter Above Ground Plane
Alien Squiglette – 1 Meter Above Ground Plane
Thing Magic M6e Radio
Measuring RSSI with Varying Hop Frequency

A.4.10- Antenna Gain Calculations



Frequency (MHz)	Antenna Gain (dB)
900	5.99
914	5.44
930	4.48

Figure A-14: Antenna Gain Calculations (Figure 6-2 reproduced)

Table A-14: Experimental setup for Figure 6-2

EXPERIMENT INFORMATION
7/15/16
File Name: CalculatingAntennaGain.xlsx
RF Controlled Chamber D
1.2 Meters Above Ground Plane
Signal Generator + Bi-Log Antenna
Signal Analyzer + Circular Polarized Antenna
Measuring Power at 3 Frequencies and using it to calculate Gain

A.4.11- Incremental Changes in RSSI Signature

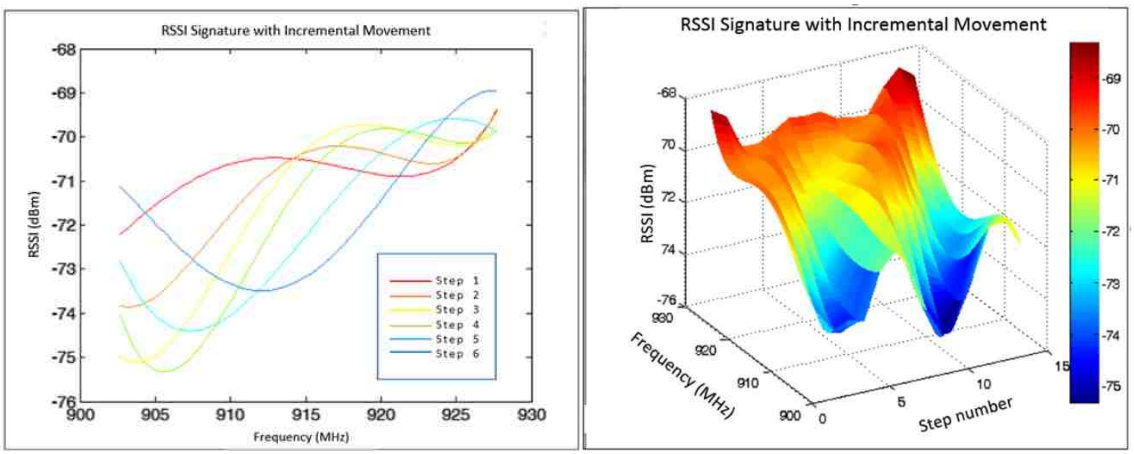


Figure A-15 – The change in RSSI with frequency as the tag and reader are moved together through a complex environment. (Figures 6-5 and 6-6 reproduced)

Table A-15: Experimental set-up for Figure 6-5 and 6-6

EXPERIMENT INFORMATION
1/23/2014
File Name: RSSI Signature inch increments.xlsx
Real World Environment A
ThingMagic M6e Radio
Circular Polarized Antenna 1 Meter Above Ground Plane
Alien Squiglette Tag Vertically Polarized 1 Meter Above Ground Plane
RSSI vs. Frequency recorded at 1-inch increments
Maintaining Distance and Orientation between Reader Antenna and Tag

A.4.12- RSSI vs. Distance in All Environments

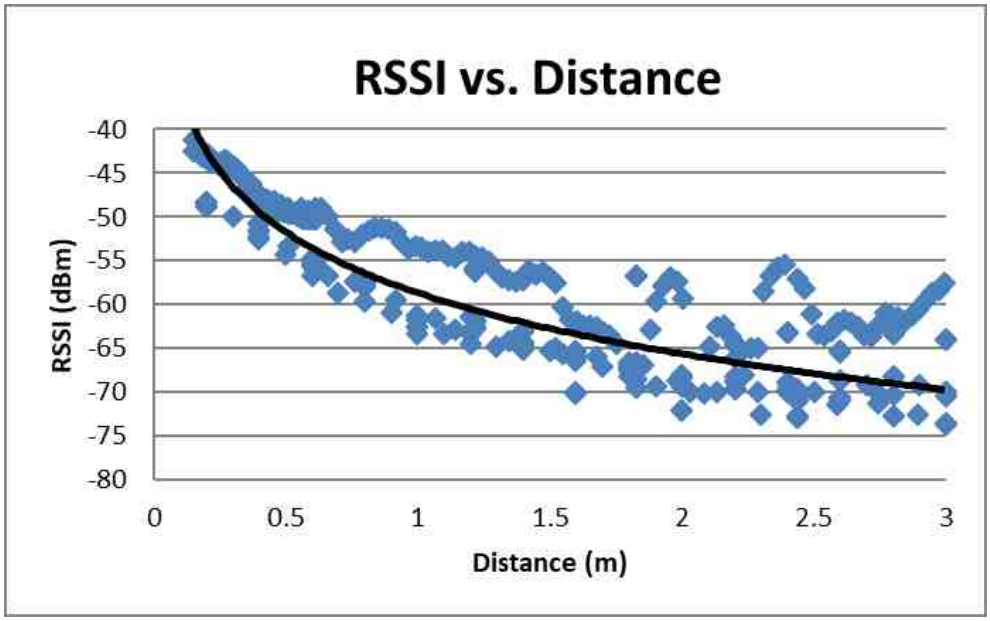


Figure A-16: RSSI measurement vs. separation distance (Figure 9-7 reproduced)

Table A-16: Experimental setup for Figure 9-7

EXPERIMENT INFORMATION
RFID RSSI Measurements
Date: Many Experiments
File Name: M6e Data Combined.xlsx
Environments A, B, C, D, E, F
1 Meters Above Ground Plane
Alien Squiglette Tag Vertically Polarized
Circular Polarized Antenna and M6e Radio

APPENDIX B: TRILATERATION RESULTS TABLES BY LEVEL

The following pivot tables were created based on all data taken. The average values represent the average error in meters for all localization estimate combinations.

B.1 RFID Results by level

Table B.1.1 – RFID Error results in meters by Level 2 (type of distance measurement)

RFID Level 2 Results

Row Labels	Mean of dist error (m)
P_Sig	1.13543794
Phase	0.653476406
R_Sig	1.668292554
RSSI	1.98153388
RSSI-Informed Phase	2.546218189
Grand Total	**1.603218376**

Table B.1.2 – RFID Error results in meters by Level 3 (which distance measurements were used in the analysis)

RFID Result by Level 3 Input Selection

Row Labels	Average Error
3 Highest Confidence	1.626600305
3 Highest RSSI	1.576110394
All Dist	1.606103871
Grand Total	**1.603218376**

Table B.1.3 – RFID Error results in meters by Level 4&5 (Type of trilateration algorithm tested)

RFID Iterative vs. Non-iterative

Row Labels	Mean Error
Non Iterative Location	1.40
Iterative Least Squares	1.52
Weighted Least Squares	1.56
Taylor Series	1.63
Nonlinear Least Squares	1.71
Antenna Pattern	1.74

Table B.1.4 – RFID Error results in meters by Level 5 (iterative trilateration method)

RFID Level 5 Iterative Algorithm Results

Row Labels	Mean Error
Antenna Pattern	1.741353981
No Additional	1.399025548
Iterative Least Squares	1.522418357
Nonlinear Least Squares	1.705379824
Taylor Series	1.629420832
Weighted Least Squares	1.555106292
Grand Total	**1.603218376**

Table B.1.5 – RFID Error results in meters by Level 3,4 & 5 Showing the interaction between the algorithm choices for RFID localization.

RFID Detailed Level 3&4 Summary						
Average of average error	Column Labels					
Row Labels	Antenna Pattern	Iterative Least Squares	Nonlinear Least Squares	Taylor Series	Weighted Least Squares	Grand Total
Centroid	**1.765860974**		**1.608508456**	**1.624949095**	**1.544061758**	**1.635845071**
3 Highest Confidence	1.72302669		1.674153077	1.695947988	1.546041146	1.659792225
3 Highest RSSI	1.644853608		1.565943629	1.576851995	1.49634782	1.570999263
All Dist	1.929702625		1.585428662	1.602047304	1.589796307	1.676743724
Hyperbolic LOP	**1.561287726**		**1.59822361**	**1.550066377**	**1.547863821**	**1.564360384**
3 Highest Confidence	1.646599861		1.662437889	1.574050951	1.546041146	1.607282462
3 Highest RSSI	1.545319952		1.548896453	1.502022185	1.49634782	1.523146602
All Dist	1.491943366		1.58333649	1.574125994	1.601202495	1.562652086
Linear Approximantion	**1.874504565**		**2.026390394**	**1.685359105**	**1.616310405**	**1.800641117**
3 Highest Confidence	1.893766897		2.137681387	1.721498349	1.546041146	1.824746945
3 Highest RSSI	1.857281444		2.087694972	1.682273655	1.49634782	1.780899473
All Dist	1.872465354		1.853794822	1.652305312	1.806542249	1.796276934
Linear Least Squares	**1.775082887**		**1.630612483**	**1.624153992**	**1.552102218**	**1.645487895**
3 Highest Confidence	1.799934812		1.68951099	1.651422623	1.546041146	1.671727393
3 Highest RSSI	1.708425342		1.601725698	1.587428641	1.49634782	1.598481875
All Dist	1.816888506		1.600600761	1.633610713	1.613917687	1.666254417
Linear LOP	**1.70662153**		**1.614881755**	**1.615345106**	**1.481864941**	**1.604678333**
3 Highest Confidence	1.799934812		1.68951098	1.651422623	1.546041146	1.67172739
3 Highest RSSI	1.708425342		1.601725688	1.587428641	1.49634782	1.598481873
All Dist	1.611504436		1.553408598	1.607184055	1.403205856	1.543825736
Not Dependent		**1.530584636**				**1.530584636**
3 Highest Confidence		1.545418152				1.545418152
3 Highest RSSI		1.4958483				1.4958483
All Dist		1.550487454				1.550487454
Grand Total	1.736671537	1.530584636	1.69572334	1.619974735	1.548440628	1.644506468

B.2 Bluetooth Low Energy Summary in Pivot tables by each level

Table B.2.1 – BLE Error results in meters by Level 2 (Type of distance measurement method used)

BLE Level 2 Summary

Row Labels	Average of Mean
Half_Sig_Half_RSSI	1.796
RSSI	1.811
RSSI Signature	1.760
Grand Total	**1.789**

Table B.2.2 – BLE Error results in meters by Level 3 (Type of distance measurement data selected to be included)

BLE Error by Level 3 Data Selection

Row Labels	Average error
3 Highest Confidence	2.099078833
3 Highest RSSI	1.599569873
All Dist	1.667754302
Grand Total	**1.788801003**

Table B.2.3 – BLE Error results in meters by Level 4 (Type of non-iterative trilateration method used)

BLE Level 4 Summary

Row Labels	Average of Mean
Centroid	1.645232336
Hyperbolic LOP	1.808021189
Linear Approximation	1.892432802
Linear Least Squares	1.853972017
Linear LOP	1.728918094
Not Dependent	1.753934638
Grand Total	**1.785024404**

Table B.2.4 – BLE Error results in meters by Level 5 (Type of iterative trilateration method used)

BLE Level 5 Iterative Algorithm Results

Row Labels	Average of Error
Antenna Pattern	1.638142402
Iterative Least Squares	1.865943875
Nonlinear Least Squares	1.634888901
Taylor Series	1.579915244
Weighted Least Squares	2.09863089
Grand Total	**1.743991955**

Table B.2.5 – BLE Error results in meters by Level 5 (A more in depth look at the impact of the type of non-iterative method that fed the iterative method)

BLE by Iterative Method - Level 5						
Average of error	**Column Labels**					
Row Labels	**Antenna Pattern**	**Iterative Least Squares**	**Nonlinear Least Squares**	**Taylor Series**	**Weighted Least Squares**	**Grand Total**
Centroid	1.48961242		1.651070551	1.582151012	1.979607385	1.675610342
Hyperbolic LOP	1.713593953		1.650550499	1.583613924	1.962591271	1.727587412
Linear Approx	1.809883629		1.694581303	1.605993498	2.213567518	1.831006487
Linear LOP	1.608666589		1.665483335	1.594713905	1.928227595	1.699272856
Linear LS	1.634785321		1.665303227	1.594693213	1.964542214	1.714830994
Not Dependent		1.885004985				1.885004985
Grand Total	**1.651308382**	**1.885004985**	**1.665397783**	**1.59223311**	**2.009707197**	**1.737058921**

Table B.2.6 – BLE Error results in meters showing detailed results by full breakdown of input options.

BLE Trilateration Data Summary

Row Labels	Average of Mean
Antenna Pattern	**1.638142402**
3 Highest Confidence	**1.979390355**
Centroid	1.659797748
Hyperbolic LOP	2.028937342
Linear Approximation	2.377296069
Linear Least Squares	1.915460307
Linear LOP	1.915460307
3 Highest RSSI	**1.495491345**
Centroid	1.379312318
Hyperbolic LOP	1.564910049
Linear Approximation	1.561073015
Linear Least Squares	1.486080672
Linear LOP	1.486080672
All Dist	**1.439545507**
Centroid	1.382924961
Hyperbolic LOP	1.475282465
Linear Approximation	1.468407373
Linear Least Squares	1.478573682
Linear LOP	1.392539056
Error	**1.976999001**
3 Highest Confidence	**2.319949242**
Centroid	1.77980494
Hyperbolic LOP	2.550371504
Linear Approximation	2.807734089
Linear Least Squares	2.230917838

Linear LOP	2.230917838
3 Highest RSSI	**1.803346787**
Centroid	1.467043158
Hyperbolic LOP	2.015793904
Linear Approximation	2.017447717
Linear Least Squares	1.758224578
Linear LOP	1.758224578
All Dist	**1.807700974**
Centroid	1.468420064
Hyperbolic LOP	1.98249162
Linear Approximation	2.04228721
Linear Least Squares	1.794840739
Linear LOP	1.750465239
Iterative Least Squares	**1.865943875**
3 Highest Confidence	**2.222086444**
Not Dependent	2.192283427
(blank)	2.236987952
3 Highest RSSI	**1.75764111**
Not Dependent	1.723077554
(blank)	1.774922889
All Dist	**1.61810407**
Not Dependent	1.61810407
Nonlinear Least Squares	**1.634888901**
3 Highest Confidence	**2.049849173**
Centroid	2.011898145
Hyperbolic LOP	2.013178367
Linear Approximation	2.112728167
Linear Least Squares	2.055720593
Linear LOP	2.055720593
3 Highest RSSI	**1.483721333**
Centroid	1.484454558
Hyperbolic LOP	1.482632623
Linear Approximation	1.482033586
Linear Least Squares	1.484742949
Linear LOP	1.484742949
All Dist	**1.371096198**
Centroid	1.371593401
Hyperbolic LOP	1.371870442
Linear Approximation	1.370708095
Linear Least Squares	1.370598813
Linear LOP	1.370710239
Taylor Series	**1.579915244**
3 Highest Confidence	**1.890686035**
Centroid	1.862888826

Hyperbolic LOP	1.866907412
Linear Approximation	1.923229412
Linear Least Squares	1.900202264
Linear LOP	1.900202264
3 Highest RSSI	**1.425407048**
Centroid	1.425752817
Hyperbolic LOP	1.42545128
Linear Approximation	1.425175116
Linear Least Squares	1.425328014
Linear LOP	1.425328014
All Dist	**1.423652648**
Centroid	1.423269344
Hyperbolic LOP	1.4237732
Linear Approximation	1.423334159
Linear Least Squares	1.424114531
Linear LOP	1.423772005
Weighted Least Squares	**2.09863089**
3 Highest Confidence	**2.230917838**
Centroid	2.230917838
Hyperbolic LOP	2.230917838
Linear Approximation	2.230917838
Linear Least Squares	2.230917838
Linear LOP	2.230917838
3 Highest RSSI	**1.758268604**
Centroid	1.758297955
Hyperbolic LOP	1.758297955
Linear Approximation	1.758297955
Linear Least Squares	1.758224578
Linear LOP	1.758224578
All Dist	**2.306706227**
Centroid	1.972108968
Hyperbolic LOP	1.929501836
Linear Approximation	2.385822229
Linear Least Squares	3.495632862
Linear LOP	1.750465239
Grand Total	**1.788801003**

APPENDIX C: MATLAB CODE USED FOR ANALYSIS

C.1 MATLAB Code used in Chapter 5

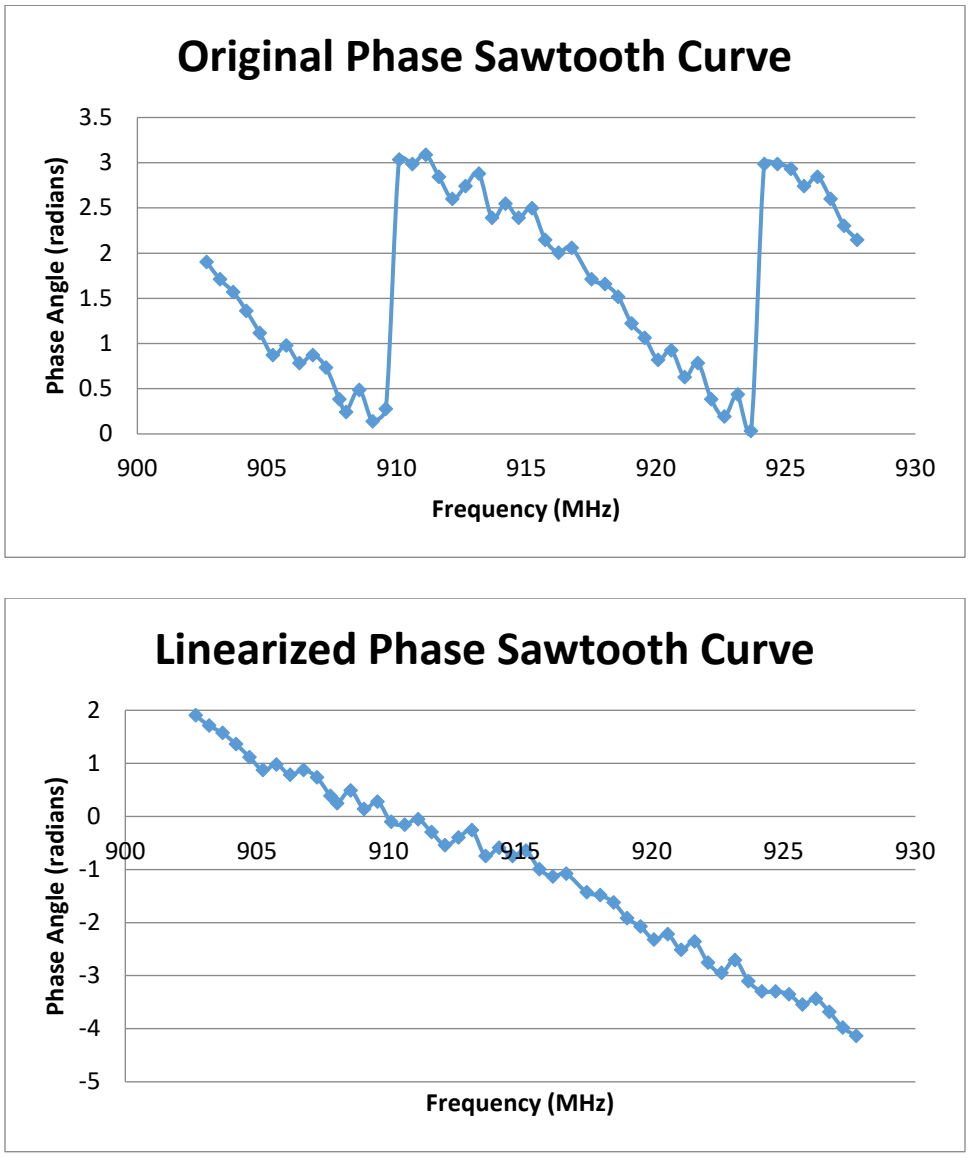


Figure C-1: Figure 5-3 and Figure 5-4 reproduced

C.1.1- linearize_phase.m

```
%%Ann Whitney%%
%%linearize_phase.m%%
%%Last Edited: 1/23/14%%
```

```
%Inputs: column vector of frequencies (50) in Hz, a matrix of rssi
values
%in dBm,and a matrix of phase angles in radians.(each column is a
different
%measurement)
%Output: a vector of the approximated antenna to tag distances for each
%measurement in meters

function [dist_app,phi,slope] =
linearize_phase(freq,rssi,phi,cableLength)
ln_coeff = -9.809;
offset = -60.907;
c = 2.99e8;

cableLength=2.16;%(m) = 7ft long cable
beta = -((1.925*cableLength)+0.334); %Experimental Phase Offset
%%due to cable length from radio to antenna

[numfreq,numpts]=size(rssi);
    for x=1:numpts
%         %%If Graphing:
%         close all
%         plot(freq,phi(:,x),'-*','MarkerSize',7);
%         axis([900000000 930000000 0 3.5])
%         xlabel('Frequency (Hz)','FontSize',12);
%         ylabel('Phase Angle (Radians)','FontSize',12);
%         title('Original Phase Sawtooth Curve','FontSize',15);
        first = 0;
        if isnan(rssi(1,x))==1
            for k = 2:size(rssi,1)
                if (isnan(rssi(k,x))==0)&&(first==0)
                    phi(1,x)=phi(k,x);
                    RSSI_dist = exp((rssi(k,x)-offset)./ln_coeff);
                    first = 1;
                end
            end
        end
        for i = 2:numfreq
            if isnan(rssi(i,x))==0
%                 %%If Graphing:
%                 close all
%                 plot(freq(1:i-1),phi(1:i-1),'*');
%                 axis([900000000 930000000 -10 10])
                dif = phi(i-1,x)-phi(i,x);
                inc = round(dif/pi);
                if i>5
                    if min(J(:,x))<inc
                        if abs((dif/pi)-inc)>0.3
                            inc = min(J(:,x));
                        end
                    end
                end
                J(i-1,x)=inc;
                phi(i,x)= phi(i,x)+(inc*pi);
            else
                if first ==0
```

```
                    RSSI_dist = exp((rssi(i-1,x)-offset)./ln_coeff);
                    first = 1;
                end
                RSSISlope = (-beta+RSSI_dist).*-((4*pi)/c);
                phi(i,x)=phi(i-1,x)+((freq(i)-freq(i-1))*RSSISlope);
            end
        end
%         %%If Graphing:
%         close all
%         plot(freq,phi(:,x),'-*','MarkerSize',7);
%         xlabel('Frequency (Hz)','FontSize',12);
%         ylabel('Phase Angle (Radians)','FontSize',12);
%         title('Linearized Phase Sawtooth Curve','FontSize',15);

        slope(x)=(mean(freq.*phi(:,x))-
(mean(freq)*mean(phi(:,x))))/((mean(freq.^2))-(mean(freq)^2));
        dist_app(x)=((-c/(4*pi))*slope(x))+beta;
        d_ft(x)=dist_app(x)/.3048;
    end %end of loop through measurements

%     %%If Graphing:
%     plot(freq,phi(:,x),'*');   %use this to plot the linearized phase
line
%     axis([9e8 9.3e8 -10 3])
%     xlabel('Frequency (Hz)')
%     ylabel('Phase Angle (radians)')
%     title('Linearized Phase')
end
```

C.2: RSSI-Informed Phase Function

C.2.1- RIP.m

```
function [newDist,RIPpts,fixedpt,slope,b,newSlope,newInt] = RIP(
freq,phase,RSSI,cableLength )
%%Ann Whitney%%
%%RIP.m%
%%Last Edited 12/13/16%%
%%This function analyzes using the new RSSI-Informed Phase method.
%%Using the lowest frequency phase point as the anchor
%%It returns the estimated distance.

%%Inputs:
%% frequency - 2 or more by 1 vector (Hz)
%% Phase - 2 or more by 1 vector (rad)
%% RSSI - 2 or more by 1 vector (dBm)
%% ActDist - actual distance (m)
%% graph - if graph ==1 then graph the data

%%Output:
%% distance - horizontal vector (m)
%% RSSI distance  - horizontal vector (m)

%%Constants%%
ln_coeff = -10.04;
offset = -58.657;
c = 2.99e8;
% beta = -((1.925*cableLength)+0.334); %Experimental Phase Offset
beta = -4.5;

%%Initial Calculations%%
rssiDist = exp((mean(RSSI)-offset)/ln_coeff);
slope = (rssiDist-beta)*((4*pi)/-c);
[freq,idx] = sort(freq);
phase = phase(idx);

%%Spacing%%
diff = freq(2:length(freq))-freq(1:length(freq)-1);
for j = 1:length(freq)-2
    FreqGap(j) = mean([diff(j),diff(j+1)]);
end
% FreqGap
if exist('FreqGap','var')
    fixedpt = find(FreqGap==max(FreqGap))+1;
    fixedpt = fixedpt(1);
else
    fixedpt = (1);
end
% fixedpt = myRand(2,length(freq)-1,1);
b = phase(fixedpt)-slope*freq(fixedpt);
RIPpts(fixedpt)=phase(fixedpt);

%%RSSI-Informed Phase%%
```

```
for i =[1:fixedpt-1 fixedpt+1:length(freq)]
    Rgap(i) = phase(i)-(freq(i)*slope+b);
    Rgap(i) = Rgap(i)/pi;
    inc(i) = round(Rgap(i));
    newpt = phase(i)-inc(i)*pi;
    RIPpts(i)=newpt;
end

newSlope = (mean(freq'.*RIPpts)-
(mean(freq)*mean(RIPpts)))/(mean(freq.^2)-mean(freq)^2);
newSlope = mean([newSlope,slope]);
newInt = phase(fixedpt)-newSlope*freq(fixedpt);
newDist = ((-c/(4*pi))*newSlope)+beta;
% error = ActDist-newDist;

%     %Graphing%%
%     allFreq = [902669000:0510000:927758000];
%     Figure;
%     h4 = plot(freq,phase,'sr','MarkerFaceColor','r','MarkerSize',7);
%     hold on
%     h3 = plot(allFreq,(slope*allFreq+b),'g','LineWidth',1);
%     h1 = plot(freq,RIPpts,'*b','MarkerSize',7);
%     h2 = plot(allFreq,(newSlope*allFreq+newInt),'b','LineWidth',1);
```

C.3: Moving RSSI-Informed Phase vs. RSSI and Phase

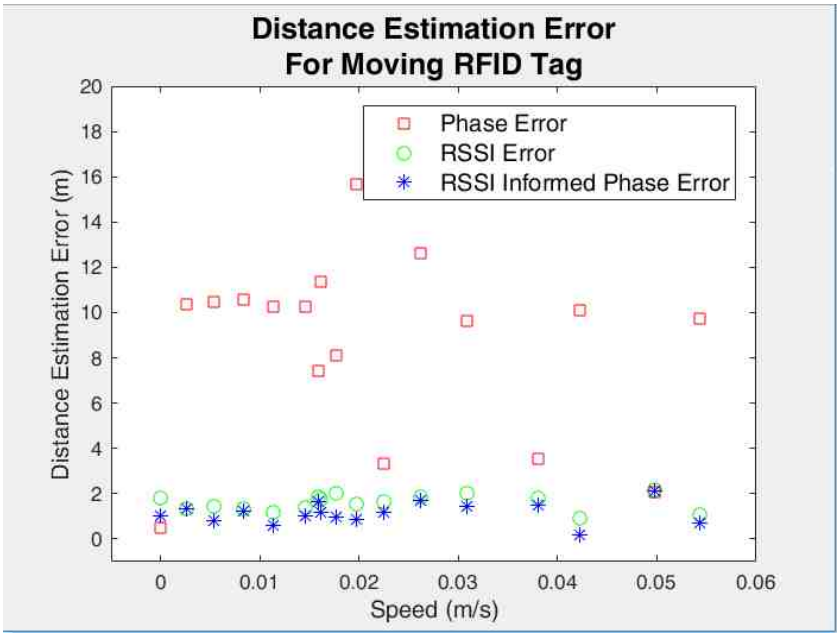


Figure C-2: Comparison of distance estimations from moving tag (produced using the code below)

Note: MovingRIP.m (C.3.1) uses linearize_phase.m (C.1.1) and stationary_linearize_phase.m (C.3.2)

C.3.1- MovingRIP.m

```
%%MovingRIP.m%%
%%Ann Whitney%%
%%Originally Written%%
%%5/17/16%%
%%Last Edited%%
%%5/17/16%%

%%The purpose of this program is to use the MovingRIPData.mat file to
%%Figure compare distance estimation methods (rssi, phase, rip)

close all
clear all
clc

load('D:\My Stuff\School\RFID\MATLAB\Data\Moving
RIP\MovingRIPData.mat')
numFiles = size(Data,2);
for i = 1:numFiles
    close all

    %%Inputs%%
    rssi = Data(i).rssi;
    phase = Data(i).phase;
```

```
time = Data(i).time;
freq = Data(i).freq;
encoder = Data(i).encoder;
if length(encoder)>length(rssi)
    encoder(length(rssi)+1:end)=[];
end
which = Data(i).which;
finalDist = Data(i).dist/100;
initialDist = 6;
whichNaN = isnan(rssi)+isnan(phase)+isnan(freq)+isnan(encoder);
where = find(whichNaN~=0);
whichNaN(where)=1;
rssi = rssi(~whichNaN);
phase = phase(~whichNaN);
time = time(~whichNaN,:);
freq = freq(~whichNaN);
encoder = encoder(~whichNaN);


%%Time%%
if length(unique(encoder))>1
    beginLoc = find(encoder~=encoder(1),1,'first');
    if length(encoder)>length(time)
        endLoc = length(time);
    else
        endLoc = find(encoder~=encoder(end),1,'last');
    end
else
    beginLoc = 1;
    endLoc = length(encoder);
end
beginTime = time(beginLoc,:);
beginseconds = num2str(round(str2double(beginTime(7:end))));
if strcmp(beginseconds,'60')
    beginminute = num2str(str2double(beginTime(4:5))+1);
    beginTime(4:8)=strcat(beginminute,':00');
elseif length(beginseconds)<2
    beginseconds = strcat('0',beginseconds);
    beginTime(7:8)=beginseconds;
else
    beginTime(7:8)=beginseconds;
end
beginTime(9:end)=[];
endTime = time(endLoc,:);
endseconds = num2str(round(str2double(endTime(7:end))));
if strcmp(endseconds,'60')
    endminute = num2str(str2double(endTime(4:5))+1);
    endTime(4:8)=strcat(endminute,':00');
elseif length(endseconds)<2
    endseconds = strcat('0',endseconds);
    endTime(7:8)=endseconds;
else
    endTime(7:8)=endseconds;
end
endTime(9:end)=[];
elapsedTime = datetime(endTime)-datetime(beginTime);
```

```
    elapsedTime = seconds(elapsedTime);
    if elapsedTime<0
        hourBack = num2str(str2num(beginTime(1:2))-1);
        PretendbeginTime=beginTime;
        PretendbeginTime(1:2)=hourBack;
        PretendendTime = endTime;
        PretendendTime(1:2)='23';
        elapsedTime = datetime(PretendendTime)-
datetime(PretendbeginTime);
        elapsedTime = seconds(elapsedTime);
    end
    speed(i) = ((initialDist-finalDist)/elapsedTime);



    %%RSSI Dist%%
    R_dist = exp((rssi(beginLoc)+58.365)./-10.64);
    ActRdist = initialDist;
    R_Error(i) = ActRdist-R_dist;

    %%Phase Dist%%
    for j = 1:length(freq)
        l = length(unique(freq(1:j)));
        if l ==50
            break
        end
    end
    [Phasefreq,idx]=sort(freq(1:j));
    Phasephase = phase(idx);
    Phaserssi = rssi(idx);
    if (i==1)||(i==2)||(i==3)
        [P_dist,phi,slope] =
stationary_linearize_phase(Phasefreq,Phaserssi,Phasephase);
    else
        [P_dist,phi,slope] =
linearize_phase(Phasefreq,Phaserssi,Phasephase);
    end
    endPhaseTime = time(j,:);
    endPhaseseconds = num2str(round(str2double(endPhaseTime(7:end))));
    if strcmp(endPhaseseconds,'60')
        endPhaseminute = num2str(str2double(endPhaseTime(4:5))+1);
        endPhaseTime(4:8)=strcat(endPhaseminute,':00');
    elseif length(endPhaseseconds)<2
        endPhaseseconds = strcat('0',endPhaseseconds);
        endPhaseTime(7:8)=endPhaseseconds;
    else
        endPhaseTime(7:8)=endPhaseseconds;
    end
    endPhaseTime(9:end)=[];
    elapsedPhaseTime = datetime(endPhaseTime)-datetime(beginTime);
    elapsedPhaseTime = seconds(elapsedPhaseTime);
    stopPhasePosition = abs(encoder(j));
    ActPdist = (((stopPhasePosition-encoder(1))/(encoder(end)-
encoder(1)))*(finalDist-initialDist))+initialDist;
    if isnan(ActPdist)
        ActPdist = initialDist;
```

```
    end
    P_Error(i) = ActPdist-P_dist;
    TotalPhaseTime(i)=elapsedPhaseTime;



    %%RIP%%
    num=3;
    RIPfreq = freq(beginLoc:beginLoc+num);
    if length(unique(RIPfreq))<4
        num = num+1;
        RIPfreq = freq(beginLoc:beginLoc+num);
    end
    RIPrssi = rssi(beginLoc:beginLoc+num);
    RIPphase = phase(beginLoc:beginLoc+num);
    endRIPTime = time(beginLoc+num,:);
    endRIPseconds = num2str(round(str2double(endRIPTime(7:end))));
    if strcmp(endRIPseconds,'60')
        endRIPminute = num2str(str2double(endRIPTime(4:5))+1);
        endRIPTime(4:8)=strcat(endRIPminute,':00');
    elseif length(endRIPseconds)<2
        endRIPseconds = strcat('0',endRIPseconds);
        endRIPTime(7:8)=endRIPseconds;
    else
        endRIPTime(7:8)=endRIPseconds;
    end
    endRIPTime(9:end)=[];
    elapsedRIPTime = datetime(endRIPTime)-datetime(beginTime);
    elapsedRIPTime = seconds(elapsedRIPTime);
    stopRIPPosition = abs(encoder(beginLoc+num));
    ActRIPdist = (((stopRIPPosition-encoder(1))/(encoder(end)-
encoder(1)))*(finalDist-initialDist))+initialDist;
    if isnan(ActRIPdist)
        ActRIPdist = initialDist;
    end
    [RIP_dist,error,RIPpts,fixedpt,newSlope,newInt] =
RIP(RIPfreq,RIPphase,RIPrssi,ActRIPdist);
    RIP_Error(i) = ActRIPdist - RIP_dist;
end
clearvars -except R_Error P_Error RIP_Error i speed TotalPhaseTime

count = 0;
for j = 1:3:length(speed)-2
    count = count+1;
    Avgspeed(count) = mean(speed(j:j+2));
    AvgP_Error(count) = mean(abs(P_Error(j:j+2)));
    AvgR_Error(count) = mean(abs(R_Error(j:j+2)));
    AvgRIP_Error(count) = mean(abs(RIP_Error(j:j+2)));
end
clearvars -except AvgR_Error AvgP_Error AvgRIP_Error i Avgspeed
TotalPhaseTime
P_Error = AvgP_Error;
R_Error = AvgR_Error;
RIP_Error = AvgRIP_Error;
speed = Avgspeed;
```

```
close all
h1 = plot(speed,abs(P_Error),'sr','MarkerSize',7);
hold on
h2 = plot(speed,abs(R_Error),'og','MarkerSize',7);
h3 = plot(speed,abs(RIP_Error),'b*','MarkerSize',7);
axis([-0.005 0.06 -1 20])
xlabel('Speed (m/s)','FontSize',12);
ylabel('Distance Estimation Error (m)','FontSize',12);
str={['Distance Estimation Error'];['For Moving RFID Tag']};
title(str,'FontSize',15)
leg = legend('Phase Error','RSSI Error','RSSI-Informed Phase Error');
set(leg,'FontSize',12);
```

C.3.2- stationary_linearized_phase.m

```
%%Ann Whitney%%
%%stationary_linearized_phase.m%%
%%Last Edited: 6/7/16%%
%Inputs: column vector of frequencies (50) in Hz, a matrix of rssi
values
%in dBm,and a matrix of phase angles in radians.(each column is a
different
%measurement)
%Output: a vector of the approximated antenna to tag distances for each
%measurement in meters

function [dist_app,phi,slope] =
stationary_linearize_phase(freq,rssi,phi)
ln_coeff = -9.809;
offset = -60.907;
c = 2.99e8;
beta = -4.5; %Experimental Phase Offset

[numfreq,numpts]=size(rssi);
    for x=1:numpts
        first = 0;
        if isnan(rssi(1,x))==1
            for k = 2:size(rssi,1)
                if (isnan(rssi(k,x))==0)&&(first==0)
                    phi(1,x)=phi(k,x);
                    RSSI_dist = exp((rssi(k,x)-offset)./ln_coeff);
                    first = 1;
                end
            end
        end
        for i = 4:numfreq
            m = (mean(freq(1:i-1).*phi(1:i-1,x))-(mean(freq(1:i-
1))*mean(phi(1:i-1,x))))/((mean(freq(1:i-1).^2))-(mean(freq(1:i-
1))^2));
            b = mean(phi(1:i-1,x))-m*mean(freq(1:i-1));
            if isnan(rssi(i,x))==0
%                 close all
%                 plot(freq(1:i-1),phi(1:i-1),'*');
%                 hold on
%                 plot(freq,m*freq+b)
```

```
%                 axis([900000000 930000000 -15 10])
                dif = phi(i-3,x)-phi(i,x);
                inc = round(dif/pi);
                if i>4
                    if J(end,x)<inc
                        inc = J(end,x);
                    end
                end
                phi(i,x)= phi(i,x)+(inc*pi);
                if phi(i,x)-(m*freq(i)+b)>2.25
                    phi(i,x)= phi(i,x)+(-1*pi);
                    inc = inc-1;
                elseif phi(i,x)-(m*freq(i)+b)<-2.25
                    phi(i,x)= phi(i,x)+(pi);
                    inc = inc+1;
                end
                J(i-1,x)=inc;
            else
                if first ==0
                    RSSI_dist = exp((rssi(i-1,x)-offset)./ln_coeff);
                    first = 1;
                end
                RSSISlope = (-beta+RSSI_dist).*-((4*pi)/c);
                phi(i,x)=phi(i-1,x)+((freq(i)-freq(i-1))*RSSISlope);
            end

        end
%         close all
%         plot(freq,phi(:,x),'-*','MarkerSize',7);
%         xlabel('Frequency (Hz)','FontSize',12);
%         ylabel('Phase Angle (Radians)','FontSize',12);
%         title('Linearized Phase Sawtooth Curve','FontSize',15);
        slope(x)=(mean(freq.*phi(:,x))-
(mean(freq)*mean(phi(:,x))))/((mean(freq.^2))-(mean(freq)^2));
        dist_app(x)=((-c/(4*pi))*slope(x))-4.5;
        d_ft(x)=dist_app(x)/.3048;
        if (sum(unique(J))>0)&&(length(J)>1)
            jump1(x) = find(J(1,x)~=J(:,x), 1 );
        else
            jump1(x)=0;
        end
    end %end of loop through measurements


%     Figure
%     plot(freq,phi(:,x),'*')   %use this to plot the linearized phase
line
end %end of function
```

C.4: Polarization Mismatch with RSSI, Phase, and RSSI-Informed Phase

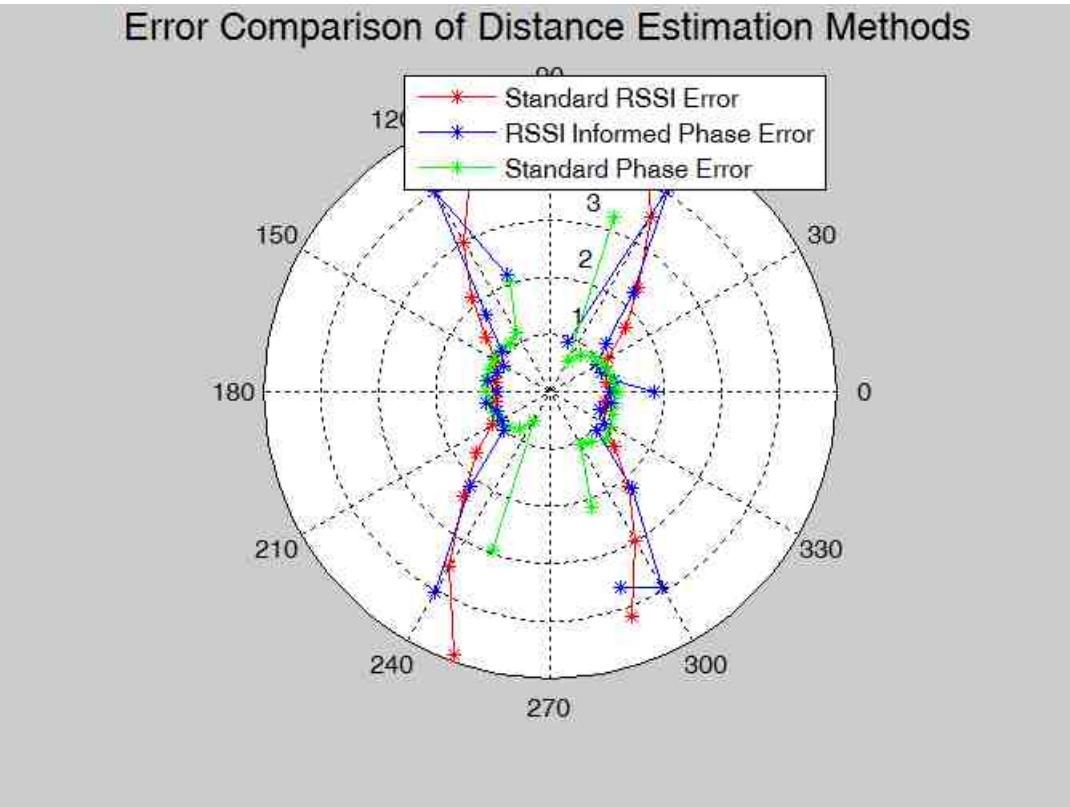


Figure C-3: Comparison of distance estimations from various tag positions (produced using the code below, and reproduction of Figure 5-7)

C.4.1- Orientation_Comparison.m

```
%%Orientation_Comparison.m%%
%%Ann Whitney%%
%%Last Edited: 6/11/2015%%
%%The purpose of this program is to compare Standard Phase data to RSSI
%%Informed Phase data with changes in polarization angle

clear all
close all
clc


%%RSSI Informed Phase Error with Respect to Orientation%%
load('RSSIInformedPhase.mat');%data comprising of two vectors Angle &
Error
Error = abs(Error);
j = 1;
k = 0;
start = 1;
deg = [0:10:360];
for i = 1:length(Angle)
    if deg(j)==Angle(i)
```

```
        k = k +1;
    elseif sum(Angle==deg(j))==0
        deg(j)=[];
        k=1;
    else
        stop = i-1;
        finalError(j) = mean(Error(start:stop));
        start = stop+1;
        j = j+1;
        k = 1;
    end
end
stop = i;
finalError(j) = mean(Error(start:stop));
RP_Error = finalError;
degRP = deg;
clearvars -except RP_Error degRP

%%Phase and RSSI Error with Respect to Orientation%%
load('StandardPhase.mat'); %data comprising of two vectors Angle &
Error
Error = abs(Error);
RSSI_Error = abs(RSSI_Error);
j = 1;
k = 0;
start = 1;
deg = [0:10:360];
for i = 1:length(Angle)
    if deg(j)==Angle(i)
        k = k +1;
    elseif sum(Angle==deg(j))==0
        deg(j)=[];
        k=1;
    else
        stop = i-1;
        finalSPError(j) = mean(Error(start:stop));
        finalRError(j) = mean(RSSI_Error(start:stop));
        start = stop+1;
        j = j+1;
        k = 1;
    end
end
stop = i;
finalSPError(j) = mean(Error(start:stop));
finalRError(j) = mean(Error(start:stop));
SP_Error = finalSPError;
R_Error = finalRError;
degSP = deg;
clearvars -except RP_Error SP_Error R_Error degSP degRP

%%Conversion%%
degSP = degSP*(pi/180);
degRP = degRP*(pi/180);


%%Graphing%%
```

```
polar(degSP(1:8),R_Error(1:8),'-r*')
hold on
polar(degRP(1:8),RP_Error(1:8),'-b*')
polar(degSP(1:8),SP_Error(1:8),'-g*')

polar(degSP(9:23),R_Error(9:23),'-r*')
polar(degSP(9:23),SP_Error(9:23),'-g*')
polar(degRP(9:22),RP_Error(9:22),'-b*')

polar(degSP(24:31),R_Error(24:31),'-r*')
polar(degSP(24:31),SP_Error(24:31),'-g*')
polar(degRP(23:30),RP_Error(23:30),'-b*')

Legend = legend('Standard RSSI Error','RSSI Informed Phase
Error','Standard Phase Error');
set(Legend,'FontSize',10)
title('Error Comparison of Distance Estimation Methods','FontSize',15)
```

C.5: Comparing Error for RSSI, Phase, and RSSI-Informed Phase

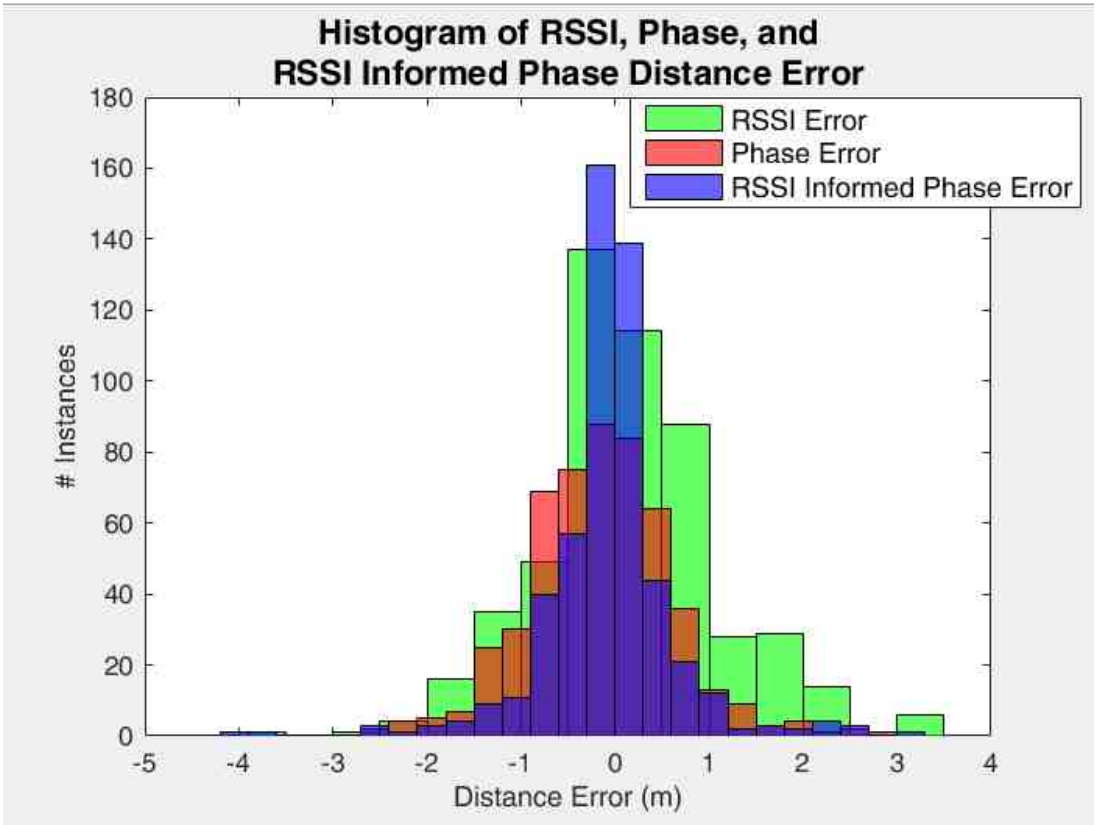


Figure C-4: Comparison of distance estimations from various multipath environments (produced using the code below, and reproduction of Figure 5-8)

C.5.1- Compare_Distance_Estimation_Methods.m

```
%%Compare Distance Estimation Methods%%
%%Ann Whitney%%
%%Last Edited%%
%%5/2/16%%

%%The purpose of this program is to compare the accuracy of the
different
%%methods of distance estimation

clear all
close all
clc

%%Load Data%%
load('D:\My Stuff\School\RFID\MATLAB\Data\CompleteRealWorld.mat')


dist = dist';
% phase=phase*(pi/180);
```

```
obs = length(dist); %number of observations
WhatObs = [1:obs];

%%Phase Distance Estimation%%
[P_dist_app,phi,slope] = stationary_linearize_phase(freq,rssi',phase');
P_Error = P_dist_app-dist;
Avg_P_Error = mean(abs(P_Error(WhatObs)));
clearvars -except dist freq phase rssi P_Error WhatObs


%%RSSI Distance Estimation%%
AvgRSSI = mean(rssi,2);
% AvgRSSI = avgRSSI;
%R_dist_app = exp((AvgRSSI+58.365)/-10.64);
R_dist_app = exp((AvgRSSI+57.95)/-9.977);
R_Error = R_dist_app-dist';
% Avg_R_Error = mean(abs(R_Error(WhatObs)));
Avg_R_Error = mean(abs(R_Error));
clearvars -except dist freq phase rssi P_Error R_Error R_dist_app YY

%%RSSI Informed Phase%%
cableLength = 6*0.3048;
for i = 1:length(dist)
    [newDist,RIPpts,fixedpt,slope,b,newSlope,newInt] = RIP(
freq,phase(i,:)',rssi(i,:)',cableLength );
    RIP_Error(i) = newDist-dist(i);
    clearvars -except dist freq phase rssi P_Error R_Error RIP_Error i
cableLength
end


%%Graphing%%
figure
histogram(R_Error,'FaceColor','g');
hold on
histogram(P_Error','FaceColor','r');
histogram(RIP_Error,'FaceColor','b');
l=legend('RSSI Error','Phase Error','RSSI Informed Phase Error');
set(l,'FontSize',11);
xlabel('Distance Error (m)')
ylabel('# Instances')
t = {'Histogram of RSSI, Phase, and';'RSSI Informed Phase Distance
Error'};
title(t,'FontSize',13)
```

C.6: RSSI Signature Gradual Change in Environment

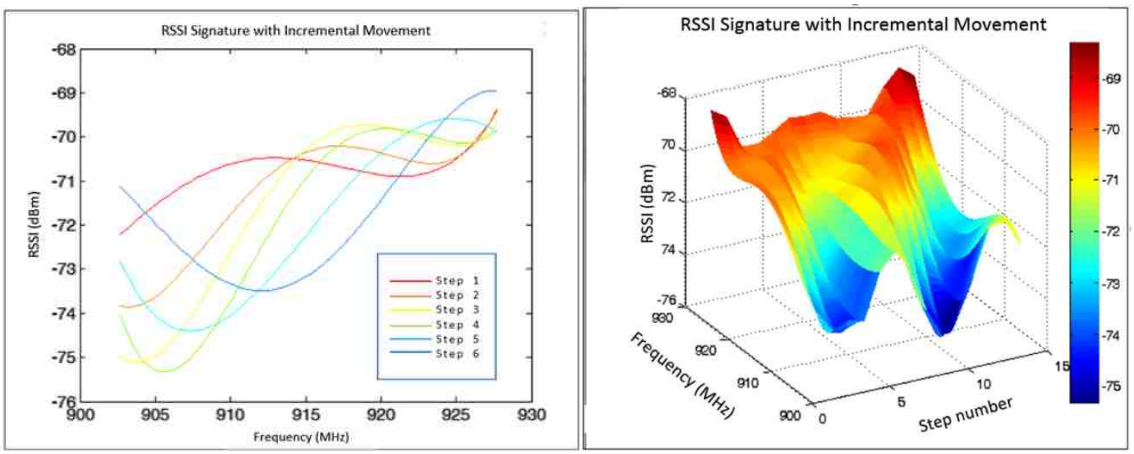


Figure C-5: Looking at the smooth transition of the RSSI Signature with small changes in the surrounding environment. (produced using the code below, and reproduction of Figure 6-5 and 6-6)

C.6.1- Compare_Distance_Estimation_Methods.m

```
%Sig_Small_Increments%
%1/23/14%
%last edited%
%2/24/16%

clear all
close all
clc

name='..\Excel Data\RSSI Signature inch increments.xlsx';
sheet='Data';
range = 'C4:EV53';


[freq,rssi,phi,num_measure]=Load_Data(name,sheet,range);
%rows are different frequencies
%columns are different measurements
colors = [1 0 0;1 0.5 0;1 1 0;0.5 1 0;0 1 1;0 0.5 1;0 0 1;...
    0.5 0 1;1 0 1;1 0 0.5];
count = 0;
start = 20;
y = freq/1e6;
x = [19:2:45];%[31:2:41]
% surf(x,y,rssi(:,x));
subplottitle = ['Step 1';'Step 2';'Step 3';'Step 4';'Step 5';'Step 6'];

for i = x
    count = count+1;
    [p,S] = polyfit(freq/1e6,rssi(:,i),4);
    clc
```

```
%     subplot(3,2,count+2)
%     plot(freq/1e6,polyval(p,freq/1e6),'k')
%     hold on;
%     plot(freq/1e6,rssi(:,i),'*')

%     plot(freq/1e6,polyval(p,freq/1e6),'Color',colors(count,:))
    Z(:,count) = polyval(p,y);
%     hold on;

%     xlabel('Frequency (MHz)','FontSize',10);
%     ylabel('RSSI (dBm)','FontSize',10);
%     title(subplottitle(count,:),'FontSize',14);
%     subtitle('Curve Fit of RSSI Signatures')

%     title('RSSI Signature with Incremental Movement','FontSize',18)
%     Leg =legend('Step 1','Step 2','Step 3','Step 4','Step 5','Step
6');
%     set(Leg,'FontSize',12)
%     set(gca,'FontSize',10)
end
h =surf([1:length(x)],y,Z,'EdgeColor','none')

% colormap(gray)
colorbar
xlabel('Step #','FontSize',14)
ylabel('Frequency (MHz)','FontSize',14)
zlabel('RSSI (dBm)','FontSize',14)
title('RSSI Signature with Incremental Movement','FontSize',18)

%%Making a Movie%%
% [freq,rssi,phi,num_measure]=Load_Data(name,sheet,range);
% mov = VideoWriter('Signature 1 inch all.avi');
% mov.FrameRate=6;
% open(mov)
% for i = 1:2:num_measure
%     figure;
%     plot(freq/1e6,mean([rssi(:,i),rssi(:,i+1)],2),'+')
%     axis([900 930 -75 -66])
%     xlabel('Frequency (MHz)');
%     ylabel('RSSI (dBm)');
%     title('RSSI Signature')
%     frame=getframe(gcf);
%     writeVideo(mov,frame)
% end
% close(mov);
```

C.7: Training and Testing a Neural Network with the RSSI Signature

Note: Neural_Network.m (C.7.1) is used in RFID_Sig.m (C.7.2), RSSI_Sig_for_Phase.m (C.7.3), and BLE_Signature.m (C.7.4)

C.7.1- Neural_Network.m

```
function [net, Actual, Guess,Reduction] = Neural_Network(
input_data,desired_output,hidden_layers)
%%input_data = n x m matrix of data, where n is the number of features
and
%%m is the number of observations
%%desired_output = 1 x m vector of data, where m is the number of
%%observations
%%hidden_layers = number of desired hidden layers

%%Cross Validation%%
k=10;
cvFolds = crossvalind('Kfold', desired_output, k);   % get indices of
10-fold CV for each fold
i=1;
testIdx = (cvFolds == i);                % get indices of test instances
trainIdx = ~testIdx;                     % get indices training
instances

%%Training Network%%
net = fitnet(hidden_layers);
net.divideParam.trainRatio = .8;
net.divideParam.valRatio = .1;
net.divideParam.testRatio = .1;
[net,tr] = train(net,input_data(:,trainIdx),desired_output(trainIdx));
Actual = desired_output(testIdx);
Guess = net(input_data(:,testIdx));
After = mean(abs(Actual-Guess));
Before = mean(abs(Actual));
Reduction = (1-(abs(After)/abs(Before)))*100;
end
```

C.7.2- RFID_Sig.m

```
%%RFID_Sig.m%%
%%Ann Whitney%%
%%Originally Written%%
%%5/2/16%%
%%Last Edited%%
%%3/11/19%%

%%The purpose of this program is to train and test a neural network to
%%analyze the RFID RSSI Signature using RSSI distance

clear all
close all
clc
```

```
%%Load Data%%
load('D:\My Stuff\School\RFID\MATLAB\Data\CompleteRealWorld.mat')

dist = dist';
% phase=phase*(pi/180);
obs = length(dist); %number of observations
WhatObs = [1:obs];

%%RSSI Distance Estimation%%
AvgRSSI = mean(rssi,2);
% AvgRSSI = avgRSSI;
%R_dist_app = exp((AvgRSSI+58.365)/-10.64);
R_dist_app = exp((AvgRSSI+57.95)/-9.977);
R_Error = R_dist_app-dist';
% Avg_R_Error = mean(abs(R_Error(WhatObs)));
Avg_R_Error = mean(abs(R_Error));
clearvars -except dist freq phase rssi P_Error R_Error R_dist_app YY

% %%RSSI Signature Using RSSI%%
% %%Training R Signature%%
%%No Smoothing%%
%%Only Complete Signatures%%
addpath('D:\My Stuff\School\RFID\MATLAB\Programs')
count = 0;
for k = 1
    for i = 1:10
        [R_net, Actual, Guess, Reduction] = Neural_Network(
rssi',R_Error',k);
        count = count+1;
        info(count,:)=[k Reduction];
        if count==1
            best_Rnet = R_net;
            best_Red = Reduction;
        else
            if Reduction>best_Red
                best_Rnet = R_net;
                best_Red = Reduction;
            end
        end
    end
end
h1 = histogram(info(:,2),10);  %differnt % reduction of error from
different neural networks
best_Red
clearvars -except best_Rnet
best = best_Rnet;
clear best_Rnet
%save('D:\My Stuff\School\RFID\MATLAB\Programs\best_Rnet.mat');
%%TEST ON 201 Environment%%
load('D:\My Stuff\School\RFID\MATLAB\Data\201Environment.mat')
AvgRSSI = mean(rssi,2);
R_dist_app = exp((AvgRSSI+57.95)/-9.977);
Original_Error = R_dist_app-dist;
Computed_Error = best(rssi')';
Computed_Dist = R_dist_app-Computed_Error;
Final_Error = Computed_Dist-dist;
```

```
Before = mean(abs(Original_Error))
After = mean(abs(Final_Error))
Reduction = (1-(abs(After)/abs(Before)))*100
figure;
plot(Original_Error,Computed_Error,'*')
xlabel('Original_Error')
rand_rssi=std2(rssi)*randn(size(rssi))+mean(mean(rssi));
Rand_Computed_Error=best(rand_rssi')';
Rand_Computed_Dist=R_dist_app-Rand_Computed_Error;
Rand_Final_Error = Rand_Computed_Dist-dist;
Rand_After = mean(abs(Rand_Final_Error))
Y=[Final_Error Rand_Final_Error];
[p,~,~]=anova1(Y)


% %%RSSI Signature Using RSSI%%
% %%Using Smoothing%%
% % for j=1:size(rssi,1)
% %     [xx,yy]=Smooth(freq,rssi',25);
% %     YY(j,:)=yy;
% % end
% %%%OR
% YY = rssi;
% addpath('D:\My Stuff\School\RFID\MATLAB\Programs')
% count = 0;
% for k = 2
%     for i = 1:1000
%         [R_net, Actual, Guess, Reduction] = Neural_Network(
YY',R_Error',k);
%         count = count+1;
%         info(count,:)=[k Reduction];
%         if count==1
%             best_Rnet = R_net;
%             best_Red = Reduction;
%         else
%             if Reduction>best_Red
%                 best_Rnet = R_net;
%                 best_Red = Reduction;
%             end
%         end
%     end
% end
% h1 = histogram(info(:,2),10);
% best_Red
% clearvars -except best_Rnet
% best = best_Rnet;
% clear best_Rnet
% % save('D:\My Stuff\School\RFID\MATLAB\Programs\best_Rnet.mat');


% %%Testing R Signature%%
% load('D:\My Stuff\School\RFID\MATLAB\Programs\best_Rnet.mat');
% GuessError = best(rssi');
% GuessDist = R_dist_app+GuessError' ;
% After = GuessDist'-dist;
% plot(dist,dist)
% hold on
```

```
% plot(dist,R_dist_app,'*')
% plot(dist,GuessDist,'*')
% legend('Ideal','Before','After')
%
% After = mean(abs(After));
% Before = mean(abs(R_Error));
% Reduction = (1-(abs(After)/abs(Before)))*100



% %%Application of RSSI Sig to RSSI Method%%
% load('best_Rnet.mat')
% R_Guess = best_Rnet(rssi');
% plot(R_Guess,R_Error,'r*')
% set(gca,'FontSize',15)
% xlabel('Neural Network Predicted Error (m)')
% ylabel('Original Error (m)')
% title('Original vs. Predicted Error')
% R_dist_app = R_dist_app'-R_Guess;
% R_Sig_Error = R_dist_app-dist;
% Avg_R_Sig_Error = mean(abs(R_Sig_Error))
```

C.7.3- RSSI_Sig_for_Phase.m

```
%%RSSI_Sig_for_Phase.m%%
%%Ann Whitney%%
%%Originally Written%%
%%5/2/16%%
%%Last Edited%%
%%3/11/19%%

%%The purpose of this program is to train and test a neural network to
%%analyze the RFID RSSI Signature using phase distance estimation

clear all
close all
clc

%%Load Data%%
load('D:\My Stuff\School\RFID\MATLAB\Data\CompleteRealWorld.mat')

dist = dist';
% phase=phase*(pi/180);
obs = length(dist); %number of observations
WhatObs = [1:obs];

%%Phase Distance Estimation%%
[P_dist_app,phi,slope] = stationary_linearize_phase(freq,rssi',phase');
P_Error = P_dist_app-dist;
Avg_P_Error = mean(abs(P_Error(WhatObs)));
clearvars -except dist freq phase rssi P_Error WhatObs

%%RSSI Signature Using Phase%%
count = 0;
for k = 4:8
```

```
    for i = 1:100
        [P_net, Actual, Guess, Reduction] = Neural_Network(
rssi',P_Error,k);
        count = count+1;
        info(count,:)=[k Reduction];
        if count==1
            best_Pnet = P_net;
            best_Ped = Reduction;
        else
            if Reduction>best_Ped
                best_Pnet = P_net;
                best_Ped = Reduction;
            end
        end
    end
end
save('best_Pnet.mat','best_Pnet');
best_Ped
plot(info(:,1),info(:,2),'*');
axis([1 10 0 40])

%%Application of RSSI Sig to Phase Method%%
load('best_Pnet.mat')
P_Guess = best_Pnet(rssi');
figure;
plot(P_Guess,P_Error,'g*')
P_dist_app = P_dist_app-P_Guess;
P_Sig_Error = P_dist_app-dist;
Avg_P_Sig_Error = mean(abs(P_Sig_Error))
```

C.7.4- BLE_Signature.m

```
%%BLE_Signature.m%%
%%Ann Whitney%%
%%9/28/16%%

%%The purpose of this program is to find the best neural network for
the BLE RSSI Signature

close all
clear all
clc

load('D:\My Stuff\School\Bluetooth\Data\BLE_Dist_Data.mat')
num = size(RSSI,1);

%%RSSI Distance Estimation%%
    coeff = -5.786;
    offset = -55.154;
    freq = [0:36];
for i  = 1:num %number of datasets
    rssi = RSSI(i,:);
    AvgRSSI(i) = mean(rssi(rssi~=0));
    DistPred(i) = exp((AvgRSSI(i)-offset)/(coeff));
end
```

```
R_Error = DistPred'-(dist./100);
Avg_R_Error = mean(abs(R_Error));

%%Average/Smooth%%
numpts = round(size(RSSI,2)/2);
top = strcat('Using ',{' '},num2str(numpts),' Points from Smoothed
Signature');
top = [{'Histogram of Percent Reduction of Error'};top];
for i = 1:num
    x = freq;
    y = RSSI(i,:);
    incomplete = find(y==0);
    if ~isempty(incomplete)
        y(incomplete)=[];
        x(incomplete)=[];
    end
    [ xx,yy ] = Smooth( x,y,numpts,'BLE' );
    input(:,i)=yy';
end

%%Apply to Neural Network%%
 count = 0;
for k = 4 %hidden layers
    for i = 1:500
        [net, Actual, Guess, Reduction] = Neural_Network( input
,R_Error',k);
        count = count+1;
        info(count,:)=[k Reduction];
        if count==1
            best = net;
            best_Red = Reduction;
        else
            if Reduction>best_Red
                best = net;
                best_Red = Reduction;
            end
        end
    end
end
h1 = histogram(info(:,2),10);
xlabel('Percent Reduction in Error')
ylabel('Number of Instances')
title(top)
max(info(:,2))
clearvars -except best
save('D:\My Stuff\School\Bluetooth\Data\best_Rnet.mat')
```

C.8: Representing the RSSI Signature

Note:

- *RepresentingSig.m (C.8.1) uses the other curve fitting functions below.*
- *The method for replacing null values with a constant is completely contained in this code and does not need an additional function*
- *All of these functions rely on myR2 (8.6.1) and myRand (8.6.2)*
- *This code is used by uncommenting the section of code associated with the desired curve fit.*

C.8.1- RepresentingSig.m

```
%%RepresentingSig.m%%
%%Ann Whitney%%
%%7/14/16%%
%%Last Edited%%
%%9/28/16%%
%%RepresentingSig%%

%%The purpose of this program is to analyze different ways of
representing
%%the signature, and use the different curve fits

close all
clear all
clc

% load('D:\My
Stuff\School\RFID\MATLAB\Data\M6e_Data_AllIncludingIncomplete.mat')
load('D:\My Stuff\School\RFID\MATLAB\Data\RealWorld.mat')
num = size(rssi,1);

% %%RSSI Distance Estimation%%
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     x(col)=[];
%     y(col)=[];
%     AvgRSSI(i) = mean(y);
% end
% R_dist_app = exp((AvgRSSI+58.365)/-10.64);
% R_Error = R_dist_app'-dist;
% Avg_R_Error = mean(abs(R_Error));


% %%Sine Approximation%%
% numpts = 5;
% top = [{'Sine Wave Approximation'};{'Using 5 Points'}];
%
% numpts = 25;
% top = [{'Sine Wave Approximation'};{'Using 25 Points'}];
%
% numpts = 50;
```

```
% top = [{'Sine Wave Approximation'};{'Using 50 Points'}];
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     x(col)=[];
%     y(col)=[];
%     [ xx,yy ] = SinFit(x,y,numpts);
%     input(:,i)= yy;
% end
%
% numpts = 5;
% top = [{'Sine Wave Approximation'};{'Using Coefficients'}];
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     x(col)=[];
%     y(col)=[];
%     [ A,B,C,D ] = SinFit_Constants(x,y,numpts);
%     input(:,i)= [A,B,C,D]';
% end



% % %%Polynomial Approximation%%
% numpts = 50;
% order = 8;
% top = [{'Histogram of Percent Reduction of Error'};{'Using
Coefficients from 4th Order Polynomial'}];
%
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     x(col)=[];
%     y(col)=[];
%     [p,xx,yy] = PolynomialFit(x,y,order,numpts);
%     input(:,i)=yy';
% end


% %% Cubic Spline %%
% numpts = 5;
% % top = [{'Histogram of Percent Reduction of Error'};{'Using 50
Points from Cubic Spline'}];
% addpath('D:\My Stuff\School\RFID\MATLAB\Programs')
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     x(col)=[];
%     y(col)=[];
%     [ xx,yy ] = CubicSpline(x,y,numpts);
%     input(:,i)=yy';
% end
```

```
% %%-80 dBm%%
% numpts = 50;
% top = [{'Histogram of Percent Reduction of Error'};{'Using 50 Points
Directly from Signature'}];
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     col = find(isnan(y));
%     y(col)=-80;
%     xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
%     for j = 1:numpts
%         where(j)=find((abs(x-xx(j)))==min(abs(x-xx(j))));
%     end
%     yy = y(where);
%     input(:,i)=yy';
% end

% %%Average/Smooth%%
% %%RSSI%%
% numpts = 25;
% WhichDevice='RFID';
% top = [{'Histogram of Percent Reduction of Error'};{'Using 25 Points
from Smoothed Signature'}];
% for i = 1:num
%     x = freq;
%     y = rssi(i,:)';
%     [ xx,yy ] = Smooth( x,y,numpts,WhichDevice );
%     input(:,i)=yy';
% end

% %%Average/Smooth%%
% %%Phase%%
% numpts = 25;
% top = [{'Histogram of Percent Reduction of Error'};{'Using 25 Points
from Smoothed Signature'}];
% for i = 1:num
%     RSSI = rssi(i,:);
%     Phi = phase(i,:)*(pi/180);
%     Freq = freq;
%     col = find(isnan(Phi));
%     Phi(col)=[];
%     Freq(col)=[];
%     RSSI(col)=[];
%     [dist_app,phi,slope] = linearize_phase(Freq,RSSI',Phi');
%     P_dist(i)=dist_app;
%     P_Error(i) = P_dist(i)-dist(i);
%     x = Freq;
%     y = phi';
%     [ xx,yy ] = Smooth( x,y,numpts );
%     input(:,i)=yy';
% end
%
%
%
% %%Apply to Neural Network%%
```

```
%  count = 0;
% for k = 6 %hidden layers
%     for i = 1:500
%         [net, Actual, Guess, Reduction] = Neural_Network( input
,P_Error,k);
%         count = count+1;
%         info(count,:)=[k Reduction];
%         if count==1
%             best = net;
%             best_Red = Reduction;
%         else
%             if Reduction>best_Red
%                 best = net;
%                 best_Red = Reduction;
%             end
%         end
%     end
% end
% h1 = histogram(info(:,2),10);
% xlabel('Percent Reduction in Error')
% ylabel('Number of Instances')
% title(top)
% max(info(:,2))
% clearvars -except best
% save('D:\My Stuff\School\RFID\MATLAB\Data\best_Pnet.mat')

% %%ANALYSIS%%
% data = xlsread('C:\Users\Ann\Desktop\Rep Sig\Compare
Methods.xlsx','Sheet1','D7:AR506');
% Med = median(data,1);
% Max = max(data);
% Pts5 = [1 5 8 11 15 19 23 27 39];
% Pts25 = [2 6 9 12 16 20 24 28 40];
% Pts50 = [3 7 10 13 17 21 25 29 41];
% Poly2 = [11 12 13];
% Poly3 = [15 16 17];
% Poly4 = [19 20 21];
% Poly5 = [23 24 25];
% Poly6 = [27 28 29];
% Poly = [Poly2 Poly3 Poly4 Poly5 Poly6];
% Coeff = [4 14 18 22 26 30];
% Full = [1 11 15 19 23 27];
% Sin = [1 2 3];
% Cubic = [5 6 7];
% Replace80 = [8 9 10];
% RollingMean = [39 40 41];
%
% % %%Compare Basic Methods%%
% % Poly2Data=[];
% % Poly3Data=[];
% % Poly4Data=[];
% % Poly5Data=[];
% % Poly6Data=[];
% % SinData=[];
% % CubicData=[];
% % Replace80Data=[];
% % RollingMeanData = [];
```

```
% % for i = 1:length(Poly2)
% %     Poly2Data = [Poly2Data;data(:,Poly2(i))];
% %     Poly3Data = [Poly3Data;data(:,Poly3(i))];
% %     Poly4Data = [Poly4Data;data(:,Poly4(i))];
% %     Poly5Data = [Poly5Data;data(:,Poly5(i))];
% %     Poly6Data = [Poly6Data;data(:,Poly6(i))];
% %     SinData = [SinData;data(:,Sin(i))];
% %     CubicData = [CubicData;data(:,Cubic(i))];
% %     Replace80Data = [Replace80Data;data(:,Replace80(i))];
% %     RollingMeanData = [RollingMeanData;data(:,RollingMean(i))];
% % end
% %
% %
% % clear data;
% % % x=[{'Sin' '2nd' '3rd' '4th' '5th' '6th' 'Cubic' 'Roll' '-80'}];
% % % y = [SinData Poly2Data Poly3Data Poly4Data Poly5Data Poly6Data
CubicData RollingMeanData Replace80Data];
% % x=[{'Sine Wave' '6th Poly' 'Cubic Spline' 'Rolling Mean' '-80 in
Holes'}];
% % y = [SinData Poly6Data CubicData RollingMeanData Replace80Data];
% % % x=[{'2nd Poly' '3rd Poly' '4th Poly' '5th Poly' '6th Poly'}];
% % % y = [Poly2Data Poly3Data Poly4Data Poly5Data Poly6Data];
% % p = anova1(y,x);
% % median(y,1);
% % max(y);
% % % xl = xlabel('Methods of Representing the Signature')
% % yl = ylabel('% Reduction in Error');
% % % s = title('Boxplot of Percent Reduction of Error');
% % % set(s,'FontSize',15)
% % % set(xl,'FontSize',12)
% % set(yl,'FontSize',12)
% % %%%%%%%%%%%%%%%%%%%%%%%%
%
% %%Calculate Whiskers Etc.%%%%%
% % %%Info matrix stores
% % %% row (1) upper whisker
% % %% row (2) upper quartile
% % %% row (3) median
% % %% row (4) lower quartile
% % %% row (5) lower whisker
% % data = sort(y,1);
% % for i = 1:size(data,2)
% %     Info(2:4,i)=fliplr(quantile(data(:,i),[0.25 0.5 0.75]));
% %     UpWhisker = Info(2,i)+1.5*(Info(2,i)-Info(4,i));
% %     LowWhisker = Info(4,i)-1.5*(Info(2,i)-Info(4,i));
% %     Info(1,i)=UpWhisker;
% %     Info(5,i)=LowWhisker;
% % %     where = max(find(data(:,i)<=UpWhisker))
% % %     Info(1,i)=data(where,i)
% % %     where = min(find(data(:,i)>=LowWhisker))
% % %     Info(5,i)=data(where,i)
% % end
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% % %% Coeff vs Points%%
% % CoeffData=[];
```

```
% % FullData=[];
% % for i = 1:length(Coeff)
% %     FullData = [FullData;data(:,Full(i))];
% %     CoeffData = [CoeffData;data(:,Coeff(i))];
% % end
% % x = {'Constants' '5 Points'}
% % y = [CoeffData FullData];
% % p=anova1(y,x)
% % median(y,1);
% % max(y);
% % yl = ylabel('% Reduction in Error');
% % set(yl,'FontSize',12)
% % %%%%%%%%%%%%%%%%%%%%
%
% %% Number of points%%
% Pts5Data=[];
% Pts25Data=[];
% Pts50Data=[];
% for i = 1:length(Pts5)
%     Pts5Data=[Pts5Data;data(:,Pts5(i))];
%     Pts25Data=[Pts25Data;data(:,Pts25(i))];
%     Pts50Data=[Pts50Data;data(:,Pts50(i))];
% end
% x = {'5 Points' '25 Points' '50 Points'}
% y = [Pts5Data Pts25Data Pts50Data];
% p=anova1(y,x)
% median(y,1);
% max(y);
% yl = ylabel('% Reduction in Error');
% set(yl,'FontSize',12)
% %%%%%%%%%%%%%%%%%
%
% %Calculate Whiskers Etc.%%%%%
% %%Info matrix stores
% %% row (1) upper whisker
% %% row (2) upper quartile
% %% row (3) median
% %% row (4) lower quartile
% %% row (5) lower whisker
% data = sort(y,1);
% for i = 1:size(data,2)
%     Info(2:4,i)=fliplr(quantile(data(:,i),[0.25 0.5 0.75]));
%     UpWhisker = Info(2,i)+1.5*(Info(2,i)-Info(4,i));
%     LowWhisker = Info(4,i)-1.5*(Info(2,i)-Info(4,i));
%     Info(1,i)=UpWhisker;
%     Info(5,i)=LowWhisker;
% %     where = max(find(data(:,i)<=UpWhisker))
% %     Info(1,i)=data(where,i)
% %     where = min(find(data(:,i)>=LowWhisker))
% %     Info(5,i)=data(where,i)
% end
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

C.8.2- Sine Wave Curve Fit of RSSI Signature

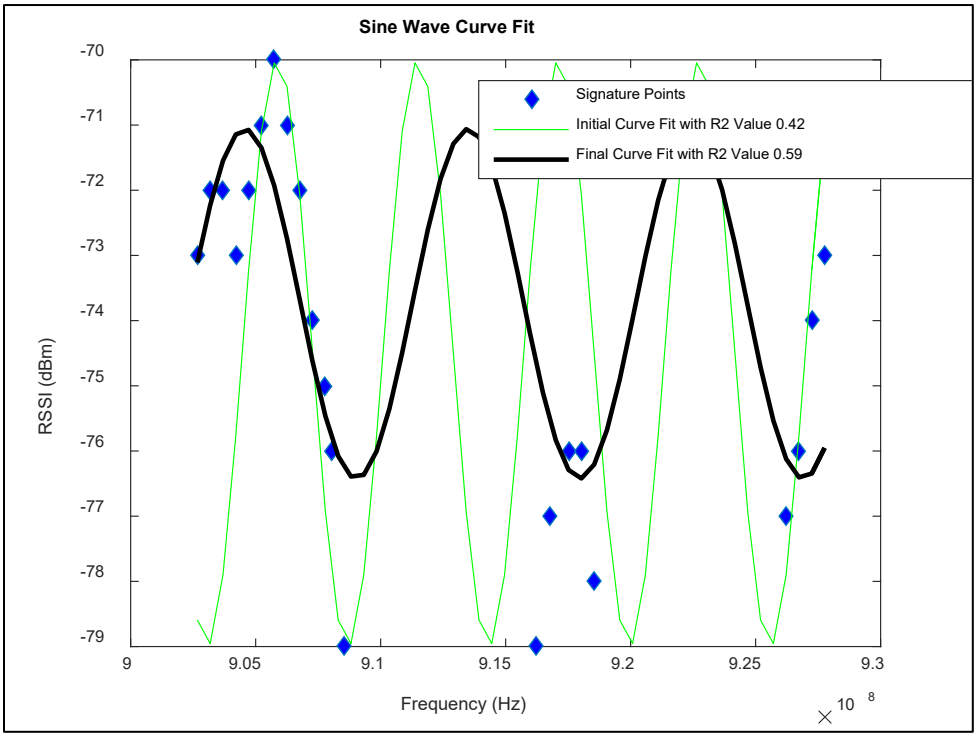


Figure C-6: Using a sine wave to fit approximate the RSSI Signature (produced using the code below, and reproduction of Figure 7-4)

Note: SinFit.m (C.8.2.1) fits a sine wave to the signature and then outputs a variable number of points.

C.8.2.1- SinFit.m

```
function [xx,yy] = SinFit( x,y,numpts )
%Fitting RSSI Signature to sin function
%%Finding Best Sine Wave Fit%%
%%of the form
%% A*sin(Bx-C)+D

%% Example of How To Use %%
% data = xlsread('C:\Users\Ann\Desktop\Rep Sig\Example
Sig.xlsx','C4:D53');
% x = data(:,1);
% y = data(:,2);
% numpts = 50;
% [ xx,yy ] = SinFit( x,y,numpts);

%% Initial Calculations %%
top = max(y);
bottom = min(y);
```

```
mid = (top+bottom)/2;
range = top-bottom;
A = range/2;
D = mid;
fudge = range*0.05; %to find points close to top or bottom

%% Simplification of Changes in Curve %%
for i = 1:length(x); %simplification of changes in curve
    if y(i)>(top-fudge)
        direction(i) = 1;
    elseif y(i)<(bottom+fudge)
        direction(i) = -1;
    end
end

%% Counting Changes in Curve %%
change = 1;
up=-1;
if exist('direction','var')
    for i = 2:length(direction)
        if direction(i)>direction(i-1) %%Going Up
            if up==0;
                change = change+1;
            end
            up = 1;
        elseif direction(i)<direction(i-1) %%Going Down
            if up==1;
                change = change+1;
            end
            up=0;
        end
    end
    low = find(direction==-1,1,'first');
    high = find(direction==1,1,'first');
else
    low = 1;
    high = length(x);
end
lengthPeriods = abs(x(high)-x(low))*2;
numPeriods = round(change/2);
B = (2*pi*numPeriods)/lengthPeriods;

midX = round(abs(high-low)/2)+min(low,high);
phi = x(midX);
if low>high
    phi = phi+lengthPeriods/2;
end
phi = 2*pi*(phi/lengthPeriods);
C = phi;

xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
yy = A*sin(B*xx-C)+D;

% %%Plot%%
% h1 = plot(x/(10^6),y,'d','MarkerFaceColor','b');
% hold on;
% %%%%%%%
```

```
% h2 = plot(xx,yy,'g');


% b = [0:1e-9:1e-6];
% for i = 1:length(b)
%     f = (A*sin(b(i)*x-C))+D;
%     saveRsqB(i) = myR2(x,y,f);
% end
% h5 = plot(b,saveRsqB);
% hold on;
% xlabel('Angular Frequency ( B )')
% ylabel('R Squared')
% title([{'Optimization Space of B'};{'Asin(Bx-C)+D'}])
% % axis([0 10 -5 1])

%%Optimize%%
%%Gradient Descent%%
N = length(x);
nA = 0.5;
nB = 1e-19;
nC = 0.3;
nD = 0.3;
for i = 1:60
    B = myRand(0,1e13,1)*nB;
    for j = 1:10
        f = A*sin(B*x-C)+D;
        df_dA = -sin(C-B.*x);
        dA = (2/N)*sum((f-y).*df_dA);
        A = A - nA*dA;
        df_dB = A.*x.*cos(C-B.*x);
        dB = (2/N)*sum((f-y).*df_dB);
        B = B - nB*dB;
        df_dC = -A*cos(C-B.*x);
        dC = (2/N)*sum((f-y).*df_dC);
        C = C - nC*dC;
        df_dD = 1;
        dD = (2/N)*sum((f-y).*df_dD);
        D = D - nD*dD;
    end
    saveRsq(i) = myR2(x,y,f);
    Save(1,i)=A;
    Save(2,i) = B;
    Save(3,i) = C;
    Save(4,i) = D;
%     h6 = plot(xx,A*sin(B*xx-C)+D,'-','Color',[0.5 0 0.5]);
%     pause(1)
%     h3 = plot(Save(2,end),saveRsq(end),'-*','Color',[0.5 0 0.5]);
%     pause(1)
end

which = find(saveRsq==max(saveRsq));
values = Save(:,which);
A = values(1);
B = values(2);
C = values(3);
D = values(4);
yy = A*sin(B*xx-C)+D;
```

```
%%Plot%%
% h4 = plot(xx/(10^6),A*sin(B*xx-C)+D,'-k','LineWidth',2);
% legend('RSSI Signature','Sine Curve Fit')
% xlabel('Frequency (MHz)')
% ylabel('RSSI (dBm)')
% set(gca,'FontSize',12)
% title('Sine Wave Curve Fit')
%%%%%%%%

xx= xx';
yy=yy';

end
```

Note: SinFit_Constants.m (C.8.2.2) fits a sine wave to the signature and then outputs the coefficients of the equation

C.8.2.2- SinFit_Constants.m

```
function [A,B,C,D] = SinFit_Constants( x,y,numpts )
%Fitting RSSI Signature to sin function
%%Finding Best Sine Wave Fit%%
%%of the form
%% A*sin(Bx-C)+D

%% Example of How To Use %%
% data = xlsread('C:\Users\Ann\Desktop\Rep Sig\Example
Sig.xlsx','C4:D53');
% x = data(:,1);
% y = data(:,2);
% numpts = 50;
% [ xx,yy ] = SinFit( x,y,numpts);

%% Initial Calculations %%
top = max(y);
bottom = min(y);
mid = (top+bottom)/2;
range = top-bottom;
A = range/2;
D = mid;
fudge = range*0.05; %to find points close to top or bottom

%% Simplification of Changes in Curve %%
for i = 1:length(x); %simplification of changes in curve
    if y(i)>(top-fudge)
        direction(i) = 1;
    elseif y(i)<(bottom+fudge)
        direction(i) = -1;
    end
end

%% Counting Changes in Curve %%
change = 1;
up=-1;
```

```
if exist('direction','var')
    for i = 2:length(direction)
        if direction(i)>direction(i-1) %%Going Up
            if up==0;
                change = change+1;
            end
            up = 1;
        elseif direction(i)<direction(i-1) %%Going Down
            if up==1;
                change = change+1;
            end
            up=0;
        end
    end
    low = find(direction==-1,1,'first');
    high = find(direction==1,1,'first');
else
    low = 1;
    high = length(x);
end
lengthPeriods = abs(x(high)-x(low))*2;
numPeriods = round(change/2);
B = (2*pi*numPeriods)/lengthPeriods;

midX = round(abs(high-low)/2)+min(low,high);
phi = x(midX);
if low>high
    phi = phi+lengthPeriods/2;
end
phi = 2*pi*(phi/lengthPeriods);
C = phi;

xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
yy = A*sin(B*xx-C)+D;
%
%Plot%%
%%graphs initial guess
h1 = plot(x,y,'d','MarkerFaceColor','b');
hold on;
h2 = plot(xx,yy,'g');
f = (A*sin(B*x-C))+D;
RsqB = myR2(x,y,f);
% title([{'Initial Estimate of Sine Curve'};{'Asin(Bx-C)+D'}]);
% leg1=strcat('Initial Curve Fit with R2 Value ',{' '},num2str(RsqB));
% leg1=['Signature Points',leg1];
% legend('Signature',leg1);
xlabel('Frequency (Hz)');
ylabel('RSSI (dBm)');

% %%plots objective function space%%
% figure;
% b = [0:1e-9:1e-6];
% for i = 1:length(b)
%     f = (A*sin(b(i)*x-C))+D;
%     saveRsqB(i) = myR2(x,y,f);
% end
% h5 = plot(b,saveRsqB);
```

```
% hold on;
% xlabel('Angular Frequency ( B )')
% ylabel('R Squared')
% title([{'Optimization Space of B'};{'Asin(Bx-C)+D'}])

%%Optimize%%
%%Gradient Descent%%
N = length(x);
nA = 0.5;
nB = 1e-19;
nC = 0.3;
nD = 0.3;
for i = 1:60
    B = myRand(0,1e13,1)*nB;
    for j = 1:10
        f = A*sin(B*x-C)+D;
        df_dA = -sin(C-B.*x);
        dA = (2/N)*sum((f-y).*df_dA);
        A = A - nA*dA;
        df_dB = A.*x.*cos(C-B.*x);
        dB = (2/N)*sum((f-y).*df_dB);
        B = B - nB*dB;
        df_dC = -A*cos(C-B.*x);
        dC = (2/N)*sum((f-y).*df_dC);
        C = C - nC*dC;
        df_dD = 1;
        dD = (2/N)*sum((f-y).*df_dD);
        D = D - nD*dD;
    end
    saveRsq(i) = myR2(x,y,f);
    Save(1,i)=A;
    Save(2,i) = B;
    Save(3,i) = C;
    Save(4,i) = D;

%     %%Plot%%
%     h6 = plot(xx,A*sin(B*xx-C)+D,'-','Color',[0.5 0 0.5]);
%     pause(1)
%     h3 = plot(Save(2,end),saveRsq(end),'-*','Color',[0.5 0 0.5]);
%     pause(1)
end

which = find(saveRsq==max(saveRsq));
values = Save(:,which);
A = values(1);
B = values(2);
C = values(3);
D = values(4);
yy = A*sin(B*xx-C)+D;

%Plot%%
h4 = plot(xx,A*sin(B*xx-C)+D,'-k','LineWidth',2);
leg1=strcat('Initial Curve Fit with R2 Value ',{' '},num2str(RsqB));
leg2 = strcat('Final Curve Fit with R2 Value ',{'
'},num2str(saveRsq(which)));
leg=['Signature Points',leg1,leg2];
legend(leg)
```

```
xlabel('Frequency (Hz)')
ylabel('RSSI (dBm)')
set(gca,'FontSize',12)
title('Sine Wave Curve Fit')
%%%%%%%

xx= xx';
yy=yy';

end
```

C.8.3- Polynomial Curve Fit

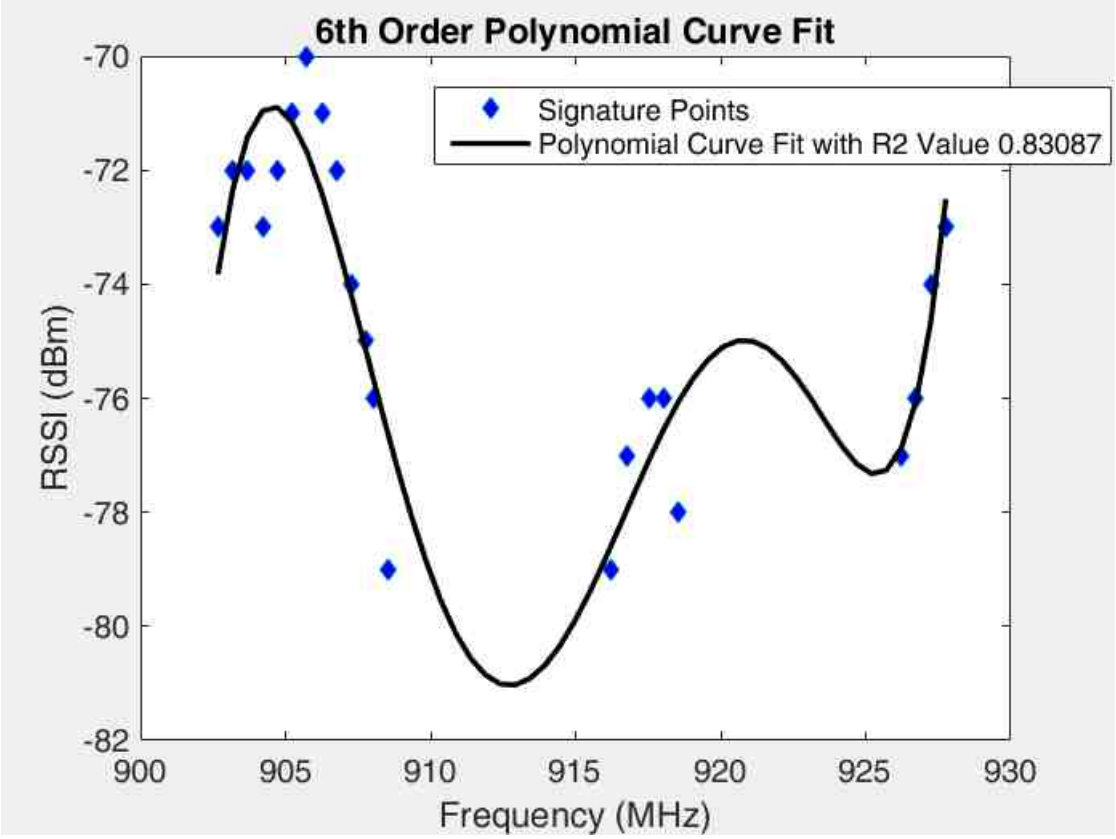


Figure C-7: Using a 6th order polynomial to fit approximate the RSSI Signature (produced using the code below, and reproduction of Figure 7-5)

C.8.3.1- PolynomialFit.m

```
function [ p,xx,yy ] = PolynomialFit(x,y,order,numpts)
%Using Polynomial to fit Signature

% %%Example of How To Use%%
% data = xlsread('C:\Users\Ann\Desktop\Rep Sig\Example
Sig.xlsx','C4:D53');
```

```
% x = data(:,1);
% y = data(:,2);
% order = 4;
% numpts = 5;
% coeff=PolynomialFit(x,y,order,numpts);


xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)]';
p = polyfit(x,y,order);
clc
yy = polyval(p,xx);
f = polyval(p,x);
RsqB = myR2(x,y,f);

%Plot%%%%%
plot(x/(10^6),y,'d','MarkerFaceColor','b')
hold on;
plot(xx/(10^6),yy,'-k','LineWidth',2)
xlabel('Frequency (MHz)');
ylabel('RSSI (dBm)');
title('6th Order Polynomial Curve Fit');
leg1=strcat('Polynomial Curve Fit with R2 Value ',{' '},num2str(RsqB));
leg1=['Signature Points',leg1];
legend('Signature',leg1);
set(gca,'FontSize',12)
%%%%%%%%%
hold off
end
```

C.8.4- Cubic Spline Curve Fit

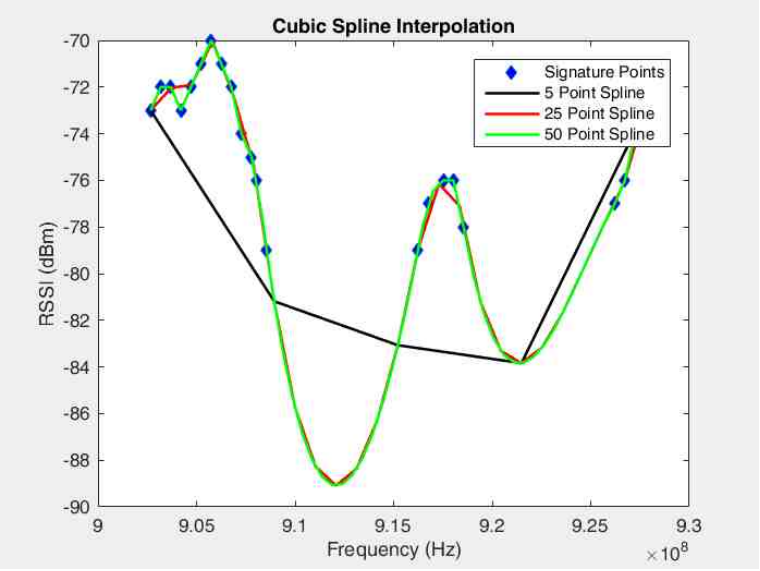


Figure C-8: Using a cubic spline with 5, 25, and 50 points to approximate the RSSI Signature (produced using the code below, and reproduction of Figure 7-6)

C.8.4.1- CubicSpline.m

```
function [ xx,yy ] = CubicSpline(x,y,numpts)
%The purpose of this function is to convert a sig to cubic spline

%% Example of How To Use %%
% data = xlsread('C:\Users\Ann\Desktop\Rep Sig\Example
Sig.xlsx','C4:D53');
% x = data(:,1);
% y = data(:,2);
% numpts = 5;
% [xx,yy] = CubicSpline(x,y,numpts);

xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
yy = spline(x,y,xx);

plot(x,y,'d','MarkerFaceColor','b')
hold on
plot(xx,yy,'k','LineWidth',1.5)
xlabel('Frequency (Hz)')
ylabel('RSSI (dBm)')
title('Cubic Spline Interpolation')

numpts=25;
xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
yy = spline(x,y,xx);
plot(xx,yy,'r','LineWidth',1.5)

numpts=50;
xx = [x(1):(x(end)-x(1))/(numpts-1):x(end)];
yy = spline(x,y,xx);
plot(xx,yy,'g','LineWidth',1.5)

legend('Signature Points','5 Point Spline','25 Point Spline','50 Point
Spline')
hold off
end
```

C.8.5- Rolling Mean Curve Fit

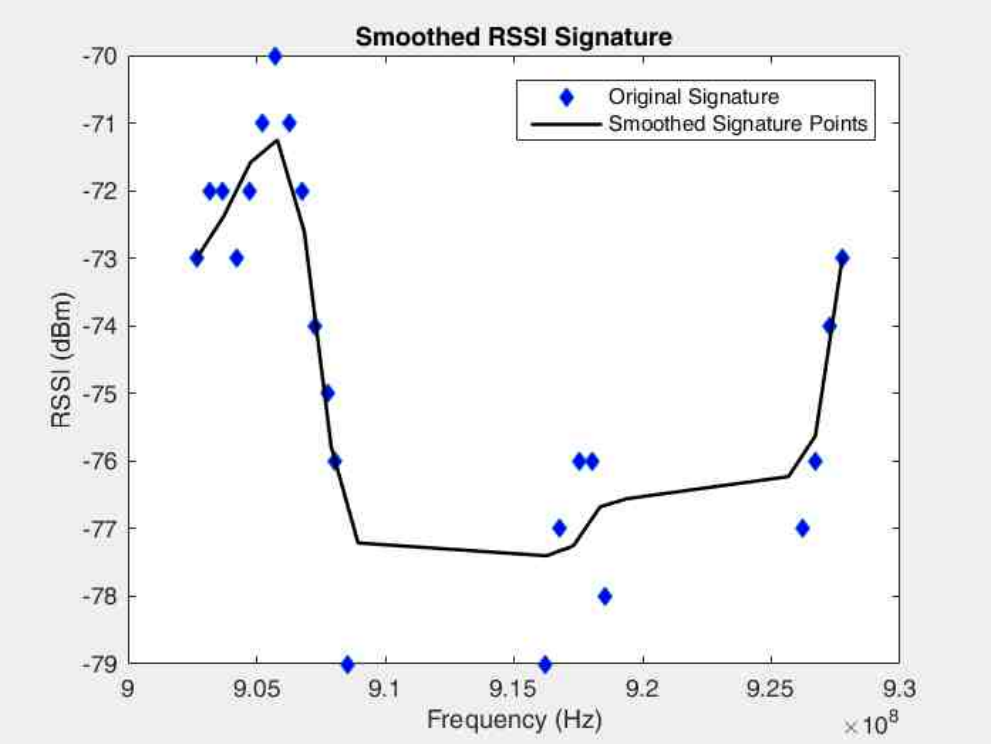


Figure C-9: Using a rolling average to approximate the RSSI Signature (produced using the code below, and reproduction of Figure 7-7)

C.8.5.1- Smooth.m

```
function [ xx,yy ] = Smooth( x,y,numpts,WhichDevice )

%% The purpose of this function is to smooth out an RSSI Signature

    %% Eliminate NaN
    col = find(isnan(y));
    x(col)=[];
    y(col)=[];

    %% Smooth Curve
    [Y]=smooth(y);

    %% Set # Of Points
    if strcmp(WhichDevice,'RFID')
        xmin = 902669000;
        xmax = 927758000;
    else
        xmin=0;
        xmax=36;
    end
    xx = [xmin:(xmax-xmin)/(numpts-1):xmax];
```

```
    for k = 1:numpts
        where = find(xx(k)>x);
        where = max(where);
        if isempty(where)
            yy(k)=Y(1);
        else
            if where==length(x)
                where = where-1;
            end
            yy(k)=((Y(where+1)-Y(where))*(xx(k)-x(where)))/(x(where+1)-
x(where))+Y(where);
        end
    end
    yy(yy<-80)=-80;
    %%%% Plot
    close all
    plot(x,y,'d','MarkerFaceColor','b')
    hold on
%     plot(xx,yy,'k-o')
    plot(xx,yy,'k','LineWidth',1.5)
    xlabel('Frequency (Hz)')
    ylabel('RSSI (dBm)')
    title('Smoothed RSSI Signature');
    legend('Original Signature','Smoothed Signature Points');
    pause;
  %%%%%%%%%

end
```

C.8.6- Supplemental Functions

Note: myR2.m (C.8.6.1) is a function to quickly calculate goodness of fit

C.8.6.1- myR2.m

```
function [ Rsq ] = myR2( x,y,f )
%% The purpose of this function is to quickly calculate the R^2 value

% %%Linear Regression%%
% m1 = mean(x.*y);
% m2 = mean(x)*mean(y);
% m3 = mean(x.^2);
% m4 = mean(x)^2;
% m = (m1-m2)/(m3-m4);
% b = mean(y)-m*mean(x);
%
% %%Coefficient of Determination%%
% f = m.*x+b;
SSres = sum((y-f).^2);
SSreg = sum((f-mean(y)).^2);
SStot = SSres+SSreg;
Rsq = 1-(SSres/SStot);

if isequal(y,f)
    Rsq=1;
end

end
```

Note: myRand.m (C.8.6.2) is a function which calculates a random number between two given numbers

C.8.6.2- myRand.m

```
function [ numVector ] = myRand( bottom,top,howMany )
%%Ann Whitney%%
%%3/17/16%%

%%The purpose of this program it to create a random number generator
that
%%gives a random number between the two given.

range = top-bottom+1;
for i = 1:howMany
    num = rand;
```

```
    num=num*range;
    num = ceil(num);
    numVector(i) = num+bottom-1;
    clear num
end
end
```

C.9: Characterizing the Environment using the RSSI Signature

Note: ClassifyEnvironment.m (C.9.1) attempts to classify RSSI Signature into the type of environment it was collected in

C.9.1- ClassifyEnvironment.m

```
%%ClassifyEnvironment%%
%%Ann Whitney%%
%%Originally Written%%
%%3/2/16%%
%%Last Edited%%
%%3/2/16%%

%%The purpose of this program is to find if and to what degree a neural
%%network can be applied in order to classify the environment based
%%upon information within the RSSI Signature

clear all
close all
clc



% %%Fully Anechoic Data%%
% load('F:\PhD Research\RFID\RFID Data\MATLAB\M6e_Data_FullyAnechoic');
% samples = size(rssi,1);
% channel = [1:length(freq)];
% fullAnRSSI = rssi;
% for i = 1:samples
%     p1(:,i) = polyfit(channel,rssi(i,:),2);
%     clc
% end
%
% %%Semi Anechoic Data%%
% clearvars -except fullAnRSSI p1 channel
% load('F:\PhD Research\RFID\RFID Data\MATLAB\M6e_Data_SemiAnechoic');
% samples = size(rssi,1);
% semiAnRSSI = rssi;
% for i = 1:samples
%     p2(:,i) = polyfit(channel,rssi(i,:),2);
%     clc
% end

%%Anechoic Data Combined%%
load('F:\PhD Research\RFID\RFID Data\MATLAB\M6e_Data_Chamber');
samples = size(rssi,1);
channel = [1:length(freq)];
AnRSSI = rssi;
for i = 1:samples
    p1(:,i) = polyfit(channel,rssi(i,:),2);
    clc
end
```

```
%%Real World Data%%
clearvars -except p1 channel
load('F:\PhD Research\RFID\RFID Data\MATLAB\M6e_Data_RealWorld');
samples = size(rssi,1);
realRSSI = rssi;
for i = 1:samples
    p2(:,i) = polyfit(channel,rssi(i,:),2);
    clc
end
p3 = [];

%%Machine Learning Analysis%%
clearvars -except p1 p2 p3
%2 Categories for SVM
p1 = [p1;ones(1,size(p1,2))];
p2 = [p2;2*ones(1,size(p2,2))];
%2 Categories for Neural Network
% p1 = [p1;ones(1,size(p1,2));zeros(1,size(p1,2))];
% p2 = [p2;zeros(1,size(p2,2));ones(1,size(p2,2))];
%3 Categories for Neural Network
% p1 = [p1;ones(1,size(p1,2));zeros(2,size(p1,2))];
% p2 = [p2;zeros(1,size(p2,2));ones(1,size(p2,2));zeros(1,size(p2,2))];
% p3 = [p3;zeros(2,size(p3,2));ones(1,size(p3,2))];
totalObs = size(p1,2)+size(p2,2)+size(p3,2);
combinedData = [p1 p2 p3];
r = randperm(totalObs);
combinedData = combinedData(:,r);
in = combinedData(1:3,:);
out = combinedData(4:end,:);

%Support Vector Machine%
k=10;
cvFolds = crossvalind('Kfold', (1:size(out,2)), k);   % get indices of
10-fold CV for each fold
i=1;
testIdx = (cvFolds == i);                % get indices of test instances
trainIdx = ~testIdx;
cl=svmtrain(in(:,trainIdx),out(:,trainIdx)');
Actual = out(:,testIdx);
Guess = svmclassify(cl,in(:,testIdx)');
for i = 1:length(Actual)
    target(Actual(i),i)=1;
    output(Guess(i),i)=1;
end
plotconfusion(target,output)


% %Neural Network Classification%
% clearvars -except in out
% hidden_layers=3;
% [net, Actual, Guess] =
Neural_Network_Classification(in,out,hidden_layers);
% plotconfusion(Actual,Guess)
```

Note: CharacterizeEnvironment.m (C.9.2) looks at the RSSI Signature from different environments to analyze the various characteristics

C.9.2- CharacterizeEnvironment.m

```
%%CharacterizeEnvironment%%
%%Ann Whitney%%
%%Originally Written%%
%%1/19/16%%
%%Last Edited%%
%%1/25/17%%

%%The purpose of this program is to find if and to what degree a neural
%%network can be applied in order to characterize the environment based
%%upon information within the RSSI Signature

clear all
close all
clc

%%EXPERIMENT D%%
addpath('D:\My Stuff\School\RFID\MATLAB\Programs');
cd('D:\My Stuff\School\RFID\Data\Characterize Environment\1-29-16\Empty
Fully Anechic Chamber');
file = dir;
file(1:2)=[];
numpts = 25;
for i = 1:size(file,1)
    data = dlmread(file(i).name);
    freq = data(:,1)*1000;
    rssi = data(:,2);
    [xx,yy]=Smooth(freq,rssi,numpts,'RFID');
    YY(i,:)=yy;
    clearvars data rssi freq yy
end
p0 = [ones(size(YY,1),1) zeros(size(YY,1),2) YY];
clearvars -except p0

cd('D:\My Stuff\School\RFID\Data\Characterize Environment\1-29-
16\Reflector A');
file = dir;
file(1:2)=[];
numpts = 25;
for i = 1:size(file,1)
    data = dlmread(file(i).name);
    freq = data(:,1)*1000;
    rssi = data(:,2);
    [xx,yy]=Smooth(freq,rssi,numpts,'RFID');
    YY(i,:)=yy;
    clearvars data rssi freq yy
end
p1 = [zeros(size(YY,1),1) ones(size(YY,1),1) zeros(size(YY,1),1) YY];
clearvars -except p0 p1
```

```
cd('D:\My Stuff\School\RFID\Data\Characterize Environment\1-29-
16\Reflector A & B');
file = dir;
file(1:2)=[];
numpts = 25;
for i = 1:size(file,1)
    data = dlmread(file(i).name);
    freq = data(:,1)*1000;
    rssi = data(:,2);
    [xx,yy]=Smooth(freq,rssi,numpts,'RFID');
    YY(i,:)=yy;
    clearvars data rssi freq yy
end
p2 = [zeros(size(YY,1),2) ones(size(YY,1),1) YY];
XX = xx;
clearvars -except p0 p1 p2 XX

%%Graphing%%%%%%%%%%
figure;
hold on;
for i=0:2
    switch i
        case 0
            YY=p0(:,4:end);
            shade = [1 0.8 0.8];
            solid = [1 0 0];
        case 1
            YY=p1(:,4:end);
            shade = [0.8 1 0.8];
            solid = [0 1 0];
        case 2
            YY=p2(:,4:end);
            shade = [0.8 0.8 1];
            solid = [0 0 1];
    end
    offset = mean(YY,2);
    avgSig = mean(YY,1);
    stdSig = std(YY,1);
    top = avgSig+stdSig;
    bottom = avgSig-stdSig;
    vertical1 = linspace(top(1),bottom(1),25);
    vertical2 = linspace(bottom(end),top(end),25);
    xx = [XX(1)*ones(1,25) XX XX(end)*ones(1,25) fliplr(XX)];
    yy = [vertical1 bottom vertical2 fliplr(top)];
    h(i+1)=fill(xx,yy,shade);
    set(h(i+1),'edgecolor',shade);
    set(h(i+1),'FaceAlpha',0.5);
    h(i+4) = plot(XX,avgSig,'Color',solid);
    clearvars -except p0 p1 p2 i XX h
end
xlabel('Frequency (Hz)')
ylabel('RSSI (dBm)')
t={'Comparison of RSSI Signatures with';'Zero, One, and Two
Reflectors'};
title(t);
legend([h(4) h(5) h(6)],'Zero Reflectors','One Reflector','Two
Reflectors');
```

```
%%%%%%%%%%%%%%%%%%%%

% YY = [p0;p1;p2];
% totalObs = size(YY,1);
% order  = randperm(totalObs);
% YY = YY(order,:);
% in =YY(:,4:end)';
% out = YY(:,1:3)';
%
% %Neural Network Classification%
% for i =1%:100
%     which = rand(1,size(in,2));
%     test = which<0.1;
%     train=~test;
%     hidden_layers=3;
%     [net, Actual, Guess] =
Neural_Network_Classification(in(:,train),...
%         out(:,train),hidden_layers);
%     plotconfusion(Actual,Guess);
%     GuessAgain = net(in(:,test));
% %     figure;
% %     plotconfusion(out(:,test),GuessAgain)
%     accuracy = (out(:,test)==round(GuessAgain));
%     Acc(i) = (sum(accuracy(1,:))/size(accuracy,2))*100;
%     clearvars -except in out Acc
% end


%%EXPERIMENT C%%
% %%Fully Anechoic Data%%
% load('D:\My
Stuff\School\RFID\MATLAB\Data\FullyAnechoicEmpty_Smoothed.mat')
% freq = XX;
% p0 = YY;
% meanRSSI = avgRSSI';
% clear avgRSSI dist XX YY
% %147 Data Sets
% load('D:\My
Stuff\School\RFID\MATLAB\Data\FullyAnechoicLocalization_Smoothed.mat')
% p1 = YY;
% p0=[p0;p1];
% meanRSSI = [meanRSSI;avgRSSI'];
% clear avgRSSI dist XX YY p1
% load('D:\My Stuff\School\RFID\MATLAB\Data\RealWorld_Smoothed.mat')
% p2 = YY;
% clear avgRSSI dist YY
%
% %%remove avg%
% for i =1:size(p0,1)
%     avgRSSI = mean(p0(i,:));
%     YY(i,:) = p0(i,:)-avgRSSI;
% end
% p0=YY;
% clear YY
%
% for i =1:size(p2,1)
%     avgRSSI = mean(p2(i,:));
```

```
%     YY(i,:) = p2(i,:)-avgRSSI;
% end
% p2=YY;
% clear YY
%
% %%Real World%%
% for i = 1:size(p2,2)
%     YY = p2(:,i);
%     middle(i)=mean(YY);
%     standD = std(YY);
%     top(i) = middle(i) + standD;
%     bottom(i) = middle(i) - standD;
%     clear YY;
% end
% vertical1 = linspace(top(1),bottom(1),25);
% vertical2 = linspace(bottom(end),top(end),25);
% xx = [XX(1)*ones(1,25) XX XX(end)*ones(1,25) fliplr(XX)];
% yy = [vertical1 bottom vertical2 fliplr(top)];
% h1=fill(xx,yy,[1 0.8 0.8]);
% hold on
% set(h1,'edgecolor',[1 0.8 0.8]);
% set(h1,'FaceAlpha',0.5);
% h2=plot(XX,middle,'r');
% set(h2,'LineWidth',2);
%
%
% %%Anechoic Chamber%%
% clearvars -except freq meanRSSI p0 p2 XX h1 h2
% for i = 1:size(p0,2)
%     YY = p0(:,i);
%     middle(i)=mean(YY);
%     standD = std(YY);
%     top(i) = middle(i) + standD;
%     bottom(i) = middle(i) - standD;
%     clear YY;
% end
% vertical1 = linspace(top(1),bottom(1),25);
% vertical2 = linspace(bottom(end),top(end),25);
% xx = [XX(1)*ones(1,25) XX XX(end)*ones(1,25) fliplr(XX)];
% yy = [vertical1 bottom vertical2 fliplr(top)];
% h3=fill(xx,yy,[0.8 0.8 1]);
% set(h3,'edgecolor',[0.8 0.8 1]);
% set(h3,'FaceAlpha',0.5);
% hold on
% h4=plot(XX,middle,'b');
% set(h4,'LineWidth',2);
% h5=legend('Real World Sig Range','Avg Real World Sig','Fully Anechoic
Sig Range','Avg Fully Anechoic Sig');
% xlabel('Frequency (Hz)')
% ylabel('RSSI - Offset (dBm)')
% t={'Comparing Shape of Fully Anechoic and';'Real World RSSI
Signatures'};
% title(t);
%
% set(h1,'Visible','off')
% set(h3,'Visible','off')
%
```

```
% %%Friis%%
% clearvars -except freq meanRSSI p0 p2 XX h1 h2 h3 h4 h5
% delete(h5)
% % Pt = 30; %30 dBm Transmit Power
% Pt = 1; %1Watt Transmit Power
% Gt = 1;%tag gain of 1 dB
% R = 2.2727;%mean distance (in m) from data
% % R = R/1000; %mean distance (in km) from data
% % f = XX/1e6; %frequency in MHz
% lamda = 2.99e8./XX; %wavelength (m)
% Gr = 5.44; %circular polarized antenna gain at 914MHz dB
% % Gr = -0.0504*f+51.441; %circular polarized antenna gain
% %%and how it changes with frequency based on Zach's measurements
% % Pr=Pt+Gt+Gr-(20*log10(R))-(20*log10(f))-32.44; %recieved power;
% Pr = (Pt*Gt*Gr.*(lamda.^2))/((4*pi*R)^2);
% Pr = 10*log10(Pr)+30;
% Pr = Pr-mean(Pr);
% h6 = plot(XX,Pr,'g');
% set(h6,'LineWidth',2);
% h5 = legend([h2 h4 h6],'Real World Sig','Fully Anechoic
Sig','Theoretical Free Space Sig');
% t={'Comparing Real World, Fully Anechoic and';'Theoretical Free Space
RSSI Signatures'};
% title(t)



% %%Machine Learning Analysis%%
% clearvars -except p0 p2
%
% p0 = [ones(size(p0,1),1) zeros(size(p0,1),1) p0];
% p2 = [zeros(size(p2,1),1) ones(size(p2,1),1) p2];
%
% totalObs = size(p0,1)+size(p2,1);
% combinedData = [p0;p2];
% r = randperm(totalObs);
% combinedData = combinedData(r,:);
% in = combinedData(:,3:end)';
% out = combinedData(:,1:2)';
% % clearvars -except in out
%
% %Neural Network Classification%
% for i =1:100
%     which = rand(1,size(in,2));
%     test = which<0.1;
%     train=~test;
%     hidden_layers=3;
%     [net, Actual, Guess] =
Neural_Network_Classification(in(:,train),...
%         out(:,train),hidden_layers);
% %     plotconfusion(Actual,Guess);
%     GuessAgain = net(in(:,test));
%     GuessAgain = GuessAgain>0.5;
% %     figure;
% %     plotconfusion(out(:,test),GuessAgain)
%     accuracy = (out(:,test)==GuessAgain);
%     Acc(i) = (sum(accuracy(1,:))/size(accuracy,2))*100;
```

```
%     clearvars -except in out Acc
% end
% histogram(Acc)
% axis([0 100 0 30])
```

C.10: Methods of Trilateration

Note:

- *RFID_BLE_Localization.m (C.10.1) uses the trilateration functions below.*
- *This code also uses the previous functions linearize_phase.m (C.1.1), RIP.m (C.2.1), and Smooth.m (C.8.5.1)*
- *This code is used by varying "Type" 1 through 4 to select the different*
    - *Type of distance measurement used*
        - *RSSI*
        - *R_Sig (RSSI Signature for error reduction in RSSI calculated distance)*
        - *Half_Sig_Half_RSSI (using a combination of RSSI Signature compensated distance and the original RSSI calculated distance)*
        - *Phase*
        - *P_Sig (using the RSSI Signature to reduce error in phase distance estimation)*
        - *RIP (RSSI Informed Phase)*
    - *Number of distance measurements used in localization*
        - *All Dist*
        - *Highest RSSI*
        - *Highest Confidence (using RSSI Signature*
    - *Type of non-iterative trilateration method*
        - *Lin LOP (Linear Locus of Position – LinearLOP.m)*
        - *LLS (Linear Least Squares – LinearLeastSquares.m)*
        - *Cent (Centroid – Centroid.m)*
        - *Lin App (Linear Approximation – LinearApprox.m)*
        - *Hyp LOP (Hyperbolic Locus of Position – Hyperbolic.m)*
    - *Type of iterative trilateration method*
        - *None (not using any additional iterative trilateration method)*
        - *ILS (Iterative Least Squares – IterativeLeastSquares.m)*
        - *Taylor (Taylor Series – TaylorSeries.m)*
        - *WLS (Weighted Least Squares – WeightedLeastSquares.m)*
        - *NonLinLS (Non-Linear Least Squares – NonlinearLeastSquares.m)*
        - *AntPat (Antenna Pattern Method – AntGradientDescent.m)*
- *BLE vs. RFID is selected by commenting or uncommenting the sections "Load BLE Data" or "Load RFID Data"*

C.10.1- RFID_BLE_Localization.m

```
%%RFID_BLE_Localization.m%%
%%Ann Whitney%%
%%Originally Written%%
%%8/25/2016
%%last edited%%
%%11/10/2017
```

```
%%The purpose of this program is to use the RFID and BLE data for
%%localization simulation

clear all
close all
% hold on
clc

Type{1}='RSSI'; %%RSSI %R_Sig %Half_Sig_Half_RSSI %Phase %P_Sig %RIP
Type{2}='All Dist'; %%All Dist %Highest RSSI %Highest Confidence
Type{3}='Cent'; %%Lin LOP %LLS %%Cent %%Lin App %%Hyp LOP
Type{4}= 'AntPat'; %%ILS  %%None %%Taylor %%WLS %%NonLinLS %%AntPat
CellNum = '41';

%% %Load BLE Data%%
load('D:\My Stuff\School\General Research\Experimental
Localization\BLE_Loc_Data.mat')
% load('D:\My Stuff\School\General Research\Experimental
Localization\InsideData_5_13_17.mat')
WhichDevice = 'BLE';
% %
% % %%Load RFID Data%%
% load('D:\My Stuff\School\General Research\Experimental
Localization\RFID_Loc_Data.mat')
% WhichDevice = 'RFID';
% cableLength=[6*0.3048 3.75 6 6*0.3048];


%%%%%%%%% STEP 1 %%%%%%%%%%
%%%% Estimate Distance %%%%
%% General
for i  = 1:size(Data,2)
    for j =1:size(Data(i).RSSI,2)
        DistAct(i,j) = Data(i).DistAct(j);
        if DistAct(i,j)~=0
%             if strcmp(WhichDevice,'BLE')
%                 Step1(i).AntName(j)=Data(i).AntName(j);
%             end
            Step1(i).AntNumber(j)=Data(i).AntNumber(j);
            Step1(i).ActDist(j) = Data(i).DistAct(j);
        end
    end
    if strcmp(WhichDevice,'RFID')
        Data(i).cableLength = cableLength;
    end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice

%% RSSI vs. Distance
if strcmp(WhichDevice,'BLE')
    %% BLE_RSSI = -5.489ln(dist)-53.563
    coeff = -5.786;
    offset = -55.154;
else
    %% RFID_RSSI = -10.04ln(dist)-58.657
    coeff = -10.04;
    offset = -58.657;
```

```
end
count = 0;
for i = 1:size(Data,2)
    for j = 1:size(Data(i).RSSI,2)
        count= count+1;
        DistAct(i,j) = Data(i).DistAct(j);
        if DistAct(i,j)~=0
            rssi = Data(i).RSSI;
            rssi = rssi(:,j);
            AvgRSSI(i,j) = mean(rssi(rssi~=0));
            DistPred(i,j) = exp((AvgRSSI(i,j)-offset)/(coeff));
            Step1(i).RSSI_Dist(j)=DistPred(i,j);
            Data(i).AvgRSSI(j) = AvgRSSI(i,j);
            RSSIerror(count)=DistPred(i,j)-Data(i).DistAct(j);
            RSSIDist(count) = DistPred(i,j);
        end
    end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice
RSSIerror

%% Phase
count = 0;
if strcmp(WhichDevice,'RFID')
    for i = 1:size(Data,2)
        for j = 1:size(Data(i).Phase,2)
            count = count+1;
            DistAct(i,j) = Data(i).DistAct(j);
            if DistAct(i,j)~=0
                freq = Data(i).Freq(:,j);
                rssi = Data(i).RSSI(:,j);
                phi = Data(i).Phase(:,j);
                cableLength = Data(i).cableLength(j);
                incomplete = find(freq==0);
                if ~isempty(incomplete)
                    freq(incomplete)=[];
                    rssi(incomplete)=[];
                    phi(incomplete)=[];
                end
                addpath 'D:\My Stuff\School\RFID\MATLAB\Programs'
                [dist_app,phi,slope] =
linearize_phase(freq,rssi,phi,cableLength);
                Step1(i).Phase_Dist(j) = dist_app;
                Phase_Dist(count)=Step1(i).Phase_Dist(j);
                ActDist(count)=Data(i).DistAct(j);
                Phaseerror(count) =dist_app-Data(i).DistAct(j);
            end
        end
    end
end
% plot(ActDist,Phase_Dist,'*')
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice
RSSIerror Phaseerror

%% RIP
cableLength=[6*0.3048 3.75 6 6*0.3048];
if strcmp(WhichDevice,'RFID')
```

```
    count = 0;
    for i = 1:size(Data,2)
        for j = 1:size(Data(i).Phase,2)
            DistAct(i,j) = Data(i).DistAct(j);
            if DistAct(i,j)~=0
                count = count +1;
                freq = Data(i).Freq(:,j);
                rssi = Data(i).RSSI(:,j);
                phi = Data(i).Phase(:,j);
                incomplete = find(freq==0);
                if ~isempty(incomplete)
                    freq(incomplete)=[];
                    rssi(incomplete)=[];
                    phi(incomplete)=[];
                end
                addpath 'D:\My Stuff\School\RFID\MATLAB\Programs'
                [newDist,RIPpts,fixedpt,slope,b,newSlope,newInt] =
RIP(freq,phi,rssi,cableLength(j) );
                Step1(i).RIP_Dist(j) = newDist;
                RIPerror(count) =newDist-Data(i).DistAct(j);
            end
        end
    end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice
RSSIerror Phaseerror RIPerror
%% Sig - RSSI
if strcmp(WhichDevice,'RFID')
    load('D:\My Stuff\School\RFID\MATLAB\Data\best_Rnet.mat')
%     load('D:\My Stuff\School\RFID\MATLAB\Programs\best_Rnet.mat')
%     threshhold = 1;
else
    load('D:\My Stuff\School\Bluetooth\Data\best_Rnet.mat')
%     threshhold = 0;
end
addpath('D:\My Stuff\School\RFID\MATLAB\Programs')
count =0;
for i = 1:size(Data,2)
    for j = 1:size(Data(i).RSSI,2)
        DistAct(i,j) = Data(i).DistAct(j);
        if DistAct(i,j)~=0
            count = count+1;
            freq = Data(i).Freq(:,j);
            numpts = round(length(freq)/2);
            rssi = Data(i).RSSI(:,j);
            incomplete=[];
            if strcmp(WhichDevice,'RFID')
                if length(nonzeros(freq))<50
                    incomplete = find(freq==0);
                end
            else
                if length(nonzeros(freq))<36
                    incomplete = find(freq(2:end)==0);
                end
            end
            if ~isempty(incomplete)
                freq(incomplete)=[];
```

```
                rssi(incomplete)=[];
            end
            [xx,yy]=Smooth(freq,rssi,numpts,WhichDevice);
            SigError = best(yy');
            SigDist = Step1(i).RSSI_Dist(j)-SigError;
            Step1(i).R_Sig_Dist(j) = SigDist;
            Step1(i).R_Sig_Error(j) = SigError;
            RSSISigerror(count) = SigDist-Data(i).DistAct(j);
        end
    end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice
RSSIerror Phaseerror RIPerror RSSISigerror

%% Sig - Phase
if strcmp(WhichDevice,'RFID')
    load('D:\My Stuff\School\RFID\MATLAB\Data\best_Pnet.mat')
    cableLength=[6*0.3048 3.75 6 6*0.3048];
    count = 0;
    for i = 1:size(Data,2)
        for j = 1:size(Data(i).RSSI,2)
            DistAct(i,j) = Data(i).DistAct(j);
            if DistAct(i,j)~=0
                count = count+1;
                freq = Data(i).Freq(:,j);
                numpts = round(length(freq)/2);
                phi= Data(i).Phase(:,j);
                rssi=Data(i).RSSI(:,j);
                incomplete = find(freq==0);
                if ~isempty(incomplete)
                    freq(incomplete)=[];
                    phi(incomplete)=[];
                    rssi(incomplete)=[];
                end
                [dist_app,phi,slope] =
linearize_phase(freq,rssi,phi,cableLength(j));
                [xx,yy]=Smooth(freq,phi,numpts,WhichDevice);
                SigError = best(yy');
                SigDist = Step1(i).Phase_Dist(j)-SigError;
                Step1(i).P_Sig_Dist(j) = SigDist;
                PhaseSigerror(count) = SigDist-Data(i).DistAct(j);
            end
        end
    end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice
RSSIerror Phaseerror RIPerror RSSISigerror PhaseSigerror

count = 0;
for i = 1:size(Step1,2)
    for j=1:size(Step1(i).ActDist,2)
        count = count+1;
        DistAct(count) = Step1(i).ActDist(j);
        R_Dist(count) = Step1(i).RSSI_Dist(j);
        RSig_Dist(count)=Step1(i).R_Sig_Dist(j);
        if strcmp(WhichDevice,'RFID')
            PSig_Dist(count)=Step1(i).P_Sig_Dist(j);
```

```
            P_Dist(count) = Step1(i).Phase_Dist(j);
            RIP_Dist(count) = Step1(i).RIP_Dist(j);
        end
    end
end

% plot(DistAct,DistAct)
% hold on
% plot(DistAct,R_Dist,'b*')
% plot(DistAct,RSig_Dist,'g*')
% % plot(DistAct,P_Dist,'*')
% % plot(DistAct,RIP_Dist,'*')
% legend({'Ideal';'RSSI Dis';'RSSI Sig Dist'})
% xlabel('Actual Distance (m)')
% ylabel('Predicted Distance (m)')
% title('RSSI Signature Predicted Dist')
% mean(abs(DistAct-R_Dist))
% mean(abs(DistAct - RSig_Dist))
% % mean(abs(DistAct-P_Dist))
% % mean(abs(DistAct-PSig_Dist))
% % mean(abs(DistAct-RIP_Dist))

%%%%%%%%% STEP 2 %%%%%%%%%%
%% Information Passed On %%
switch Type{2}
    case 'All Dist'
        %%All Dist
        for i = 1:size(Step1,2)
            switch Type{1}
                case 'RSSI'
                    Step2(i).Dist = Step1(i).RSSI_Dist; %Select the
type of distance estimation you want to use
                case 'R_Sig'
                    Step2(i).Dist=Step1(i).R_Sig_Dist;
                case 'Half_Sig_Half_RSSI'

Step2(i).Dist=0.5*(Step1(i).R_Sig_Dist+Step1(i).RSSI_Dist);
                case 'Phase'
                    Step2(i).Dist=Step1(i).Phase_Dist;
                case 'P_Sig'
                    Step2(i).Dist=Step1(i).P_Sig_Dist;
                case 'RIP'
                    Step2(i).Dist=Step1(i).RIP_Dist;
            end
            Step2(i).ActDist = Data(i).DistAct;
            Step2(i).AvgRSSI = Data(i).AvgRSSI;
            Step2(i).AntLoc = Data(i).AntLoc;
        end
    case 'Highest RSSI'
        %%3 Highest RSSI
        num = 3;
        for i=1:size(Step1,2)
            rssi = Data(i).AvgRSSI;
            [sorted,idx]=sort(rssi,'descend');
            maxRSSIs = sorted(1:num);
            maxIdxs = idx(1:num);
            switch Type{1}
```

```
                case 'RSSI'
                    Step2(i).Dist = Step1(i).RSSI_Dist; %Select the
type of distance estimation you want to use
                case 'R_Sig'
                    Step2(i).Dist=Step1(i).R_Sig_Dist;
                case 'Half_Sig_Half_RSSI'

Step2(i).Dist=0.5*(Step1(i).R_Sig_Dist+Step1(i).RSSI_Dist);
                case 'Phase'
                    Step2(i).Dist=Step1(i).Phase_Dist;
                case 'P_Sig'
                    Step2(i).Dist=Step1(i).P_Sig_Dist;
                case 'RIP'
                    Step2(i).Dist=Step1(i).RIP_Dist;
            end
            Step2(i).Dist = Step2(i).Dist(maxIdxs);
            Step2(i).ActDist = Data(i).DistAct(maxIdxs);
            Step2(i).AvgRSSI = rssi(maxIdxs);
            Step2(i).AntLoc = Data(i).AntLoc(:,maxIdxs);
        end
    case 'Highest Confidence'
        %3 Highest Confidence
        num = 3;
        for i=1:size(Step1,2)
            confidence = Step1(i).R_Sig_Error;
            [~,idx]=sort(abs(confidence));
            maxIdxs = idx(1:num);
            rssi = Data(i).AvgRSSI;
            switch Type{1}
                case 'RSSI'
                    Step2(i).Dist = Step1(i).RSSI_Dist; %Select the
type of distance estimation you want to use
                case 'R_Sig'
                    Step2(i).Dist=Step1(i).R_Sig_Dist;
                case 'Half_Sig_Half_RSSI'

Step2(i).Dist=0.5*(Step1(i).R_Sig_Dist+Step1(i).RSSI_Dist);
                case 'Phase'
                    Step2(i).Dist=Step1(i).Phase_Dist;
                case 'P_Sig'
                    Step2(i).Dist=Step1(i).P_Sig_Dist;
                case 'RIP'
                    Step2(i).Dist=Step1(i).RIP_Dist;
            end
            Step2(i).Dist = Step2(i).Dist(maxIdxs);
            Step2(i).ActDist = Data(i).DistAct(maxIdxs);
            Step2(i).AvgRSSI = rssi(maxIdxs);
            Step2(i).AntLoc = Data(i).AntLoc(:,maxIdxs);
        end
end
clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type WhichDevice

%%%%%%%%% STEP 3 %%%%%%%%%%
%%% Closed Form Localization %%%

switch Type{3}
    case 'Lin LOP'
```

```
        %% Linear Locus of Position
        for i = 1:size(Step2,2)
            [Step3(i).PredTagLoc,Step3(i).Error]=LinearLOP(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist );
            Error(i) = Step3(i).Error;
        end
        error = mean(Error);
        stDev = std(Error);
    case 'LLS'
        %% Linear Least Squares
        for i = 1:size(Step2,2)
            [Step3(i).PredTagLoc,Step3(i).Error]=LinearLeastSquares(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist );
            Error(i) = Step3(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'Cent'
        %%Centroid
        for i = 1:size(Step2,2)
            [Step3(i).PredTagLoc,Step3(i).Error] = Centroid(
Step2(i).Dist,Step2(i).AntLoc,Data(i).TagLoc );
            Error(i) = Step3(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'Lin App'
        %% Linear Approx
        for i = 1:size(Step2,2)
            [Step3(i).PredTagLoc,Step3(i).Error]=LinearApprox(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist );
            Error(i) = Step3(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'Hyp LOP'
        %% Hyperbolic% LOP
        for i = 1:size(Step2,2)
            [Step3(i).PredTagLoc,Step3(i).Error]=Hyperbolic(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist );
            Error(i) =Step3(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
end
switch Type{4}
    case 'None'
        Where=strcat('G',CellNum,':DB',CellNum);
        xlswrite('D:\My Stuff\School\General
Research\Data\LocalizationError.xlsx',Error,WhichDevice,Where)
    otherwise
        clearvars -except Data Step1 Step2 Step3 Step4 CellNum Type
WhichDevice
end


%%%%%%%%% STEP 4 %%%%%%%%%%
```

```
%% Iterative Localization %

switch Type{4}
    case 'Taylor'
        %% Taylor Series Approximation
        for i = 1:size(Step2,2)
            %     Guess = mean(Step2(i).AntLoc,2)';
            Guess = Step3(i).PredTagLoc;
            [Step4(i).PredTagLoc,Step4(i).Error]=TaylorSeries(
Step2(i).AntLoc', Data(i).TagLoc,Step2(i).Dist,Guess);
            Error(i) = Step4(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'WLS'
        %% Weighted Least Squares
        for i = 1:size(Step2,2)
            %     Guess = mean(Step2(i).AntLoc,2)';
            Guess = Step3(i).PredTagLoc;
            [Step4(i).PredTagLoc,Step4(i).Error]=WeightedLeastSquares(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist,Guess);
            Error(i) = Step4(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'ILS'
        %% Iterative Least Squares
        for i = 1:size(Step2,2)
            [Step4(i).PredTagLoc,Step4(i).Error]=IterativeLeastSquares(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist);
            Error(i) = Step4(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)
    case 'NonLinLS'
        %% Nonlinear Least Squares
        for i = 1:size(Step2,2)
            [Step4(i).PredTagLoc,Step4(i).Error]=NonlinearLeastSquares(
Step2(i).AntLoc', Data(i).TagLoc, Step2(i).Dist, Step3(i).PredTagLoc);
            Error(i) = Step4(i).Error;
        end
        % error = mean(Error)
        % stDev = std(Error)

    case 'AntPat'
        % %% Incorporate Antenna Pattern SLOW METHOD
        % tic
        % for i =1:size(Step2,2)
        %     [Step4(i).PredTagLoc,Step4(i).Error]=IncorpAntPatrn(
Step2(i).AntLoc',...
        %         Data(i).TagLoc,Step2(i).Dist,Step3(i).PredTagLoc);
        %     Error(i) = Step4(i).Error;
        % end
        % error = mean(Error)
        % stDev = std(Error)
        % toc
```

```
        %% Incorporate Antenna Pattern FAST METHOD
        % tic
        for i =1:size(Step2,2);
            [Step4(i).PredTagLoc,Step4(i).Error]=AntGradientDescent(
Step2(i).AntLoc',...

Data(i).TagLoc,Step2(i).Dist,Step3(i).PredTagLoc,Step1(i).R_Sig_Error);
            %     [~,mostConfident] = min(abs(Step1(i).R_Sig_Error));
            %     confidenceGap(i) = sqrt(sum((Data(i).TagLoc-
Step2(i).AntLoc(:,mostConfident)').^2))...
            %         -Step2(i).Dist(mostConfident);
            Error(i) = Step4(i).Error;
        end
        % Error(end)
        % error = mean(Error)
        % stDev = std(Error)
        % toc
end
% Where=strcat('G',CellNum,':DB',CellNum);
% xlswrite('D:\My Stuff\School\General
Research\Data\LocalizationError.xlsx',Error,WhichDevice,Where)
% clearvars -except Data Step1 Step2 Step3 Step4 Type CellNum
WhichDevice Error
mean(Error)
```

C.10.2- LinearLOP.m

```
function [ PredTagLoc,Error ] = LinearLOP( AntLoc, TagLoc, Dist )

%%Ann Whitney%%
%%11/15/16%%
%%Linear LOP%%
%%The purpose of this program is to calculate the most likely location
of
%%the tag using the Linear LOP method from "Beyond Trilateration: GPS
%%Positioning Geometry and Analytical Accuracy"

% plot(TagLoc(1),TagLoc(2),'k*')
% hold on
% for i = 1:length(Dist)
%     PlotCircle(AntLoc(i,1),AntLoc(i,2),Dist(i))
%     plot(AntLoc(i,1),AntLoc(i,2),'r*')
% end



[d,idx]=sort(Dist);
x1 = AntLoc(idx(1),1);
x2 = AntLoc(idx(2),1);
x3 = AntLoc(idx(3),1);
y1 = AntLoc(idx(1),2);
y2 = AntLoc(idx(2),2);
y3 = AntLoc(idx(3),2);
r1 = d(1);
r2 = d(2);
r3 = d(3);

X = [x2-x1;x3-x2;x3-x1];
Y = [y2-y1;y3-y2;y3-y1];
E = -[r1-r2;r2-r3;r3-r1];
Equals = 0.5*[x2^2+y2^2-x1^2-y1^2+r1^2-r2^2;x3^2+y3^2-x2^2-y2^2+r2^2-
r3^2;...
    x3^2+y3^2-x1^2-y1^2+r1^2-r3^2];
M = [X Y E Equals];
M = rref(M);
PredTagLoc = M(1:2,4)';
Error = sqrt(sum((PredTagLoc-TagLoc).^2));

%     plot(PredTagLoc(1),PredTagLoc(2),'b*')

end
```

C.10.3- LinearLeastSquares.m

```
function [ PredTagLoc,Error ] = LinearLeastSquares( AntLoc, TagLoc,
DistPred )
%%Ann Whitney%%
%%Originally Written%%
%%2/16/16%%
%%Last Edited%%
%%11/15/16%%
```

```
%%The purpose of this program is to find the most likely location of
the
%%unknown object using linear least squares method

Thetabar= mean(AntLoc,1)';
xr = Thetabar(1);
yr = Thetabar(2);
Xstar = [2*(AntLoc(:,1)-xr) 2*(AntLoc(:,2)-yr)];
Y = [(AntLoc(:,1)-xr).^2+(AntLoc(:,2)-yr).^2-DistPred.^2'];
Ystar = Y+Xstar*Thetabar;
Theta = inv(Xstar'*Xstar)*Xstar'*Ystar;
Error = sqrt(sum([Theta-TagLoc'].^2));
Theta = Theta';
PredTagLoc = Theta;

end
```

C.10.4- Centroid.m

```
function [ final_loc,error] = Centroid( Dist,AntLoc,TagLoc)
%This function calculates the most likely location of the unknown
deivce
%using the centroid method similar to that in "Automatic virtual
%calibration of range-based indoor localization systems"

%     %1st Plot
%     figure(1);
%     hold on;
%     xlabel('X Coordinate (m)');
%     ylabel('Y Coordinate (m)');
%     title('BLE Localization Map');
%     for j = 1:size(Dist,2)
%         PlotCircle(AntLoc(1,j),AntLoc(2,j),Dist(j));
%     end
%     plot(TagLoc(1),TagLoc(2),'k*','MarkerSize',10);
%     legend({'1','2','3','4','Act Loc'});
%     axis([-15 15 -10 20])

    AntLoc=AntLoc(:,~isnan(Dist));
    Dist=Dist(~isnan(Dist));

    %Calculate
    smallest = find(Dist==min(Dist));
    loc_smallest = AntLoc(:,smallest);
    radius_smallest = Dist(smallest);
    for j = 1:size(Dist,2)
        dist_to_others(j) = sqrt(sum((AntLoc(:,j)-loc_smallest).^2));
    end
    [~,idx]=sort(dist_to_others);
    close = idx(2:3);
    radii_close = Dist(close);
    how_far_apart = dist_to_others(idx(2:3))-
(radii_close+radius_smallest);
    [biggest_diff,~] = max(how_far_apart);
    if biggest_diff>0
```

```
        growth = (biggest_diff/2)+(0.1*(biggest_diff/2));
        all_radii = Dist+growth;
    else
        all_radii = Dist;
    end
    weight = 1./all_radii;
    Intersect = [];
    weighted_intersect = [];

%     %2nd Plot
%     figure(2);
%     hold on;
%     xlabel('X Coordinate (m)');
%     ylabel('Y Coordinate (m)');
%     title('BLE Localization Map');
    for j = 1:size(Dist,2)
%         PlotCircle(AntLoc(1,j),AntLoc(2,j),all_radii(j));
        for k = 1+j:size(Dist,2)
            x1 = AntLoc(1,j);
            y1 = AntLoc(2,j);
            r1 = all_radii(j);
            x2 = AntLoc(1,k);
            y2 = AntLoc(2,k);
            r2 = all_radii(k);
            [x,y] = IntersectingCircles(x1,y1,r1,x2,y2,r2);
            Intersect = [Intersect;x',y'];
            weighted_intersect =
[weighted_intersect;x'*weight(j)*weight(k),y'*weight(j)*weight(k)];
        end
    end
    final_loc= sum(weighted_intersect,1);
    if isempty(final_loc)
        weight=weight/norm(weight);
        final_loc=[sum(weight.*AntLoc(1,:)) sum(weight.*AntLoc(2,:))];
    elseif isnan(final_loc(1))||isnan(final_loc(2))
        weight=weight/norm(weight);
        final_loc=[sum(weight.*AntLoc(1,:)) sum(weight.*AntLoc(2,:))];
    end
    error = sqrt(sum((final_loc-TagLoc).^2));
%     plot(TagLoc(1),TagLoc(2),'k*','MarkerSize',10);
%     plot(final_loc(1),final_loc(2),'b*');
%     plot(Intersect(:,1),Intersect(:,2),'rs');
%     legend({'1','2','3','4','Act Loc','Pred Loc','Intersect Pts'});
%     axis([-15 15 -10 20])

end
```

C.10.5- LinearApprox.m

```
function [ PredTagLoc,Error ] = LinearApprox( AntLoc, TagLoc, DistPred
)
%%Ann Whitney%%
%%Originally Written%%
%%11/16/16%%
```

```
%%The purpose of this program is to find the most likely location of
the
%%unknown object using the close form solution in "Indoor Robot
Positioning
%%using an Enhanced Trilateration Algorithm" and ignoring the z
component;



    [DistPred,idx]=sort(DistPred);
    DistPred = DistPred(1:3);
    AntLoc = AntLoc(idx(1:3),:);

%     %%Plot%%
%     plot(TagLoc(1),TagLoc(2),'k*')
%     hold on
%     for j =1:length(DistPred)
%         PlotCircle(AntLoc(j,1),AntLoc(j,2),DistPred(j));
%     end
%     %%%%%%%



    r1 = DistPred(1);
    r2 = DistPred(2);
    r3 = DistPred(3);
    x1 = AntLoc(1,1);
    x2 = AntLoc(2,1);
    x3 = AntLoc(3,1);
    y1 = AntLoc(1,2);
    y2 = AntLoc(2,2);
    y3 = AntLoc(3,2);

    horiz = x1;
    x1=0;
    x2 = x2-horiz;
    x3 = x3-horiz;
    vert = y1;
    y1=0;
    y2 = y2-vert;
    y3 = y3-vert;

    theta = atan2(y2,x2);
    d2 = sqrt(x2^2+y2^2);
    x2 = d2*cos(0);
    y2 = d2*sin(0);
    phi = atan2(y3,x3);
    d3 = sqrt(x3^2+y3^2);
    x3 = d3*cos(phi-theta);
    y3 = d3*sin(phi-theta);

    x=(r1^2-r2^2+x2^2)/(2*x2);
    y = (r1^2-r3^2+x3^2+y3^2-(2*x3*x))/(2*y3);

    d = sqrt(x^2+y^2);
    x = d*cos(theta)+horiz;
```

```
    y = d*sin(theta)+vert;
    PredTagLoc = [x y];
    Error = sqrt(sum((PredTagLoc-TagLoc).^2));
%     %%Plot%%
%     plot(x,y,'g*')
%     %%%%%%%%%

End
```

C.10.6- Hyperbolic Locus of Position

Note: Hyperbolic.m (C.10.6.1) uses IntersectingHyperbolas (10.6.2)

C.10.6.1- Hyperbolic.m

```
function [ PredTagLoc,Error ] = Hyperbolic( AntLoc, TagLoc, DistPred )

%%Ann Whitney%%
%%Originally Written%%
%%2/16/16%%
%%Last Edited%%
%%11/15/16

%%The purpose of this program is to find the most likely location of
the
%%unknown object using Hyperbolic LOP.

%%Rearrange%%
[DistPred,idx]=sort(DistPred);
AntLoc = AntLoc(idx,:);

% %%Plot%%
% hold on
% for i=1:length(DistPred)
% PlotCircle(AntLoc(i,1),AntLoc(i,2),DistPred(i))
% end
% plot(TagLoc(1),TagLoc(2),'g*')
% %%%%%%%%

for i = 1:2%length(DistPred)-1

    i1 = i;
    i2 = i+1;
    D = sqrt(sum((AntLoc(i1,:)-AntLoc(i2,:)).^2));
    c = D/2;
    delta = abs(DistPred(i1)-DistPred(i2));
    if delta==0
        delta = 0.001;
    end
```

```
    e(i) = D/delta;
    a(i) = c/e(i);
    b(i) = a(i)*sqrt(abs(e(i)^2-1));
    phi(i) = atan2(AntLoc(i1,2)-AntLoc(i2,2),AntLoc(i1,1)-
AntLoc(i2,1));
    cent(i,:) = [mean([AntLoc(i1,1) AntLoc(i2,1)]) mean([AntLoc(i1,2)
AntLoc(i2,2)])];
    focus(i,:) = AntLoc(i1,:);
    if i>1
        [x(i-1),y(i-1)]=IntersectingHyperbolas(a(i-1),b(i-1),cent(i-
1,:),e(i-1),phi(i-1),focus(i-1,:),...
            a(i),b(i),cent(i,:),e(i),phi(i),focus(i,:));
    end
end

x = mean(x);
y = mean(y);
PredTagLoc = [x y];
Error = sqrt(sum((PredTagLoc-TagLoc).^2));

%
%     %%Plot%%
%     plot(PredTagLoc(1),PredTagLoc(2),'k*')
%     %%%%%%%%


end
```

C.10.6.2- IntersectingHyperbolas.m

```
function [ x,y ] = IntersectingHyperbolas(
a1,b1,cent1,e1,phi1,focus1,a2,b2,cent2,e2,phi2,focus2 )
%%The purpose of this function is to calculate where two hyperbolas
%%intersect.

howBig=10;

%%Hyperbola 1
c1 = e1*a1;
if abs(focus1-[(cent1(1)+c1*cos(phi1)) (cent1(2)+c1*sin(phi1))])<0.1
%selcting focus
    if e1>=1
        hyp1Range = a1+howBig:-0.1:a1; %hyperbola 1 range
    else
        hyp1Range = -a1:0.01:a1;
    end
elseif abs(focus1-[cent1(1)-c1*cos(phi1) cent1(2)-c1*sin(phi1)])<0.1
%seclecting focus
    if e1>=1
        hyp1Range = -a1-howBig:0.1:-a1; %hyperbola 1 range
    else
        hyp1Range = -a1:0.01:a1;
    end
else
    fprintf('Not one of Foci')
```

```
end
hyp1(1,:) = hyp1Range; % x vector
hyp1(2,:) = b1*sqrt(abs((((hyp1(1,:)).^2)./a1^2)-1)); % y vector on top
hyp1(3,:) = -b1*sqrt(abs((((hyp1(1,:)).^2)./a1^2)-1)); %y vector on
bottom
hyp1= [hyp1 [fliplr(hyp1(1,:));fliplr(hyp1(3,:));fliplr(hyp1(3,:))]];
%combine vectors
hyp1(3,:) = []; % deleting extra row

r1 = sqrt(sum(hyp1.^2,1)); %radius of points for rotation
alpha1 = atan2(hyp1(2,:),hyp1(1,:)); %angle of points for rotation
hyp1(1,:) = r1.*cos(phi1+alpha1); %rotating x of hyperbola
hyp1(2,:) = r1.*sin(phi1+alpha1); %rotating y of hyperbola
hyp1(1,:) = hyp1(1,:)+cent1(1);%translating x of hyperbola
hyp1(2,:) = hyp1(2,:)+cent1(2);%translating y of hyperbola
%%%%%%%%%%%%%%%

%%Hyperbola 2
c2 = e2*a2;
if abs(focus2-[(cent2(1)+c2*cos(phi2)) (cent2(2)+c2*sin(phi2))])<0.1
%selcting focus
    if e2>=1
        hyp2Range = a2+howBig:-0.1:a2; %hyperbola 1 range
    else
        hyp2Range = -a2:0.1:a2;
    end
elseif abs(focus2-[(cent2(1)-c2*cos(phi2)) (cent2(2)-
c2*sin(phi2))])<0.1 %seclecting focus
    if e2>=1
        hyp2Range = -a2-howBig:0.1:-a2; %hyperbola 1 range
    else
        hyp2Range = -a2:0.1:a2;
    end
else
    fprintf('Not one of Foci')
end
hyp2(1,:) = hyp2Range; % x vector
hyp2(2,:) = b2*sqrt(abs((((hyp2(1,:)).^2)./a2^2)-1)); % y vector on top
hyp2(3,:) = -b2*sqrt(abs((((hyp2(1,:)).^2)./a2^2)-1)); %y vector on
bottom
hyp2= [hyp2 [fliplr(hyp2(1,:));fliplr(hyp2(3,:));fliplr(hyp2(3,:))]];
%combine vectors
hyp2(3,:) = []; % deleting extra row

r2 = sqrt(sum(hyp2.^2,1)); %radius of points for rotation
alpha2 = atan2(hyp2(2,:),hyp2(1,:)); %angle of points for rotation
hyp2(1,:) = r2.*cos(phi2+alpha2); %rotating x of hyperbola
hyp2(2,:) = r2.*sin(phi2+alpha2); %rotating y of hyperbola
hyp2(1,:) = hyp2(1,:)+cent2(1);%translating x of hyperbola
hyp2(2,:) = hyp2(2,:)+cent2(2);%translating y of hyperbola
%%%%%%%%%%%%%%%%%%%



%%Calculate Intersection%%
[x,y]=intersections(hyp1(1,:),hyp1(2,:),hyp2(1,:),hyp2(2,:),0);
```

```
%%If Hyperbolas Don't Intersect%%
if isempty(x)
    m = focus1-focus2;
    m = m(2)/m(1);
    b = focus1(2)-m*focus1(1);
    lineRange=[min([focus1(1) focus2(1)]) max([focus1(1) focus2(1)])];
    lineRange = [lineRange(1)-1 lineRange(2)+1];
    lineX = lineRange(1):(lineRange(2)-lineRange(1))/20:lineRange(2);
    lineY = m*lineX+b;
    [x1,y1]=intersections(hyp1(1,:),hyp1(2,:),lineX,lineY,0);
    [x2,y2]=intersections(hyp2(1,:),hyp2(2,:),lineX,lineY,0);
    x = [x1;x2];
    y = [y1;y2];
end
%%If The Hyperbolas Cross Twice%%
if length(x)>1
    d1 = sqrt((focus1(1)-x(1))^2+(focus1(2)-y(1))^2);
    d2 = sqrt((focus1(1)-x(2))^2+(focus1(2)-y(2))^2);
    if d1<d2
        x=x(1);
        y=y(1);
    else
        x=x(2);
        y=y(2);
    end
end

%
% %%PLOT%%
% plot(hyp1(1,:),hyp1(2,:))
% hold on
% plot(hyp2(1,:),hyp2(2,:))
% plot(x,y,'b*')
% axis([-15 15 -15 15])
% %%%%%%%


end
```

Note: PlotHyperbola.m (C.10.6.3) is used to graph the hyperbolas in a non-traditional orientation

C.10.6.3- PlotHyperbola.m

```
function  PlotHyperbola( a,b,cent,phi,plot_focus )
%originally written
%10/13/16
%last edited
%10/17/16
```

```
%The purpose of this program is to plot hyperbolas based on a, b, c,
and
%slope inputs.

%Calculations
howBig = 10; %how big an area to graph;
% xrange = [cent(1)-howBig cent(1)+howBig]; %range of x data points
% yrange = [cent(2)-howBig cent(2)+howBig]; %range of y data points
% x = xrange(1):xrange(2); %vector of x data points
%
% slope = tan(phi); %slope of center line
% intercept = cent(2)-slope*cent(1); %intercept of center line
% theta = asec(sqrt(a^2+b^2)/a); %angle of asymptote
% theta_vect = -theta:0.0001:theta; %vector of angles for hyperbola
vertex=[cent(1)+a*cos(phi) cent(2)+a*sin(phi);...
    cent(1)-a*cos(phi) cent(2)-a*sin(phi)]; %vertex(1,:) = vertex1,
vertex(2,:) = vertex 2
% asym1 = (x-cent(1)).*tan(phi+theta)+cent(2); %asymptote 1
% asym2 = (x-cent(1)).*tan(phi-theta)+cent(2); %asymptote 2
c = sqrt(a^2+b^2);
focus = [cent(1)+c*cos(phi) cent(2)+c*sin(phi);...
    cent(1)-c*cos(phi) cent(2)-c*sin(phi)]; %focus(1,:) = focus1,
focus(2,:) = focus 2

%HYP #1
hyp1Range = a+howBig:-0.1:a; %hyperbola #1 range
hyp1(1,:) = hyp1Range; % x vector
hyp1(2,:) = b*sqrt((((hyp1(1,:)).^2)./a^2)-1); % y vector on top
hyp1(3,:) = -b*sqrt((((hyp1(1,:)).^2)./a^2)-1); %y vector on bottom
hyp1= [hyp1 [fliplr(hyp1(1,:));fliplr(hyp1(3,:));fliplr(hyp1(3,:))]];
%combine vectors
hyp1(3,:) = []; % deleting extra row

r1 = sqrt(sum(hyp1.^2,1)); %radius of points for rotation
alpha1 = atan2(hyp1(2,:),hyp1(1,:)); %angle of points for rotation
hyp1(1,:) = r1.*cos(phi+alpha1); %rotating x of hyperbola
hyp1(2,:) = r1.*sin(phi+alpha1); %rotating y of hyperbola
hyp1(1,:) = hyp1(1,:)+cent(1);%translating x of hyperbola
hyp1(2,:) = hyp1(2,:)+cent(2);%translating y of hyperbola

%Hyp#2
hyp2Range = -a-howBig:0.1:-a; %hyperbola #2 range
hyp2(1,:) = hyp2Range; % x vector
hyp2(2,:) = b*sqrt((((hyp2(1,:)).^2)./a^2)-1); % y vector on top
hyp2(3,:) = -b*sqrt((((hyp2(1,:)).^2)./a^2)-1); %y vector on bottom
hyp2= [hyp2 [fliplr(hyp2(1,:));fliplr(hyp2(3,:));fliplr(hyp2(3,:))]];
%combine vectors
hyp2(3,:) = []; % deleting extra row

r2 = sqrt(sum(hyp2.^2,1)); %radius of points for rotation
alpha2 = atan2(hyp2(2,:),hyp2(1,:)); %angle of points for rotation
hyp2(1,:) = r2.*cos(phi+alpha2); %rotating x of hyperbola
hyp2(2,:) = r2.*sin(phi+alpha2); %rotating y of hyperbola
hyp2(1,:) = hyp2(1,:)+cent(1);%translating x of hyperbola
hyp2(2,:) = hyp2(2,:)+cent(2);%translating y of hyperbola
```

```
%%Plot
hold on
% axis([xrange yrange])
% plot(x,slope*x+intercept)
% plot(cent(1),cent(2),'r*')
% plot(vertex(1,1),vertex(1,2),'*b')
% plot(vertex(2,1),vertex(2,2),'*b')
% plot(x,asym1,'g')
% plot(x,asym2,'g')
if abs(focus(1,:)-plot_focus)<0.01
    plot(hyp1(1,:),hyp1(2,:))
elseif abs(focus(2,:)-plot_focus)<0.01
    plot(hyp2(1,:),hyp2(2,:))
else
    fprintf('\nNot one of Foci\n')
end
end
```

C.10.7- TaylorSeries.m

```
function [ PredTagLoc,Error ] = TaylorSeries( AntLoc, TagLoc,
DistPred,Guess )
%%the purpose of this program is to practic taylor series method for
%%localization
%%Initial guess
xo = Guess(1);
yo = Guess(2);
% plot(xo,yo,'b*')
% hold on
for j = 1:10
    Gamma = [];
    for i = 1:length(DistPred)
        g = AntLoc(i,1); %x
        h = AntLoc(i,2); %y
        r = DistPred(i);
%         if j==1
%             PlotCircle(g,h,r);
%         end
        f(i) = sqrt((xo-g)^2+(yo-h)^2);
        fx(i) = (2*(xo-g))/f(i);
        fy(i) = (2*(yo-h))/f(i);
        Gamma(i,:) = [fx(i) fy(i)];
        z(i,1) = f(i)-r;
    end
    sigma2 = var(DistPred);
    delta = inv(Gamma'*inv(sigma2)*Gamma)*Gamma'*inv(sigma2)*z;
    xo = xo-delta(1);
    yo = yo-delta(2);
%     plot(xo,yo,'b*')
end
PredTagLoc = [xo yo];
Error = sqrt(sum((PredTagLoc-TagLoc).^2));
end
```

C.10.8- WeightedLeastSquares.m

```
function [ PredTagLoc,Error ] = WeightedLeastSquares( AntLoc, TagLoc,
DistPred, Guess )
%%the purpose of this program is to preform weighted least squares
%%localization


[DistPred,idx]=sort(DistPred);
AntLoc=AntLoc(idx,:);
count =0;
for i = 1:length(DistPred)
    if DistPred(i)<10
        count = count+1;
        dist(count)=DistPred(i);
        antLoc(count,:)=AntLoc(i,:);
    end
end
DistPred = dist;
AntLoc = antLoc;
```

```
clear dist antLoc
n=length(DistPred);
x = Guess';

% %%Plot%%
% plot(TagLoc(1),TagLoc(2),'k*')
% hold on
% for l=1:n
%     PlotCircle(AntLoc(l,1),AntLoc(l,2),DistPred(l))
% end
% %%%

for i = 1:n-1;
    A(i,:) = [2*(AntLoc(i+1,1)-AntLoc(i,1)) 2*(AntLoc(i+1,2)-
AntLoc(i,2))];
    b(i,1) = [(AntLoc(i+1,1)^2-AntLoc(i,1)^2)+((AntLoc(i+1,2)^2-
AntLoc(i,2)^2))+...
        (DistPred(i)^2-DistPred(i+1)^2)];
end

for j = 1:10
    e = A*x-b;

    if sum(abs(e))<1e-13
        break;
    elseif sum(abs(e)<1e-13)>=1
        break;
    end

%     % %%Plot%%
%     plot(x(1),x(2),'b*')
%     % %%%%%%%%

    W = diag(e);

    x = inv(A'*inv(W)*A)*A'*inv(W)*b; %new coordinate
end


PredTagLoc=x';
Error = sqrt(sum((PredTagLoc-TagLoc).^2));

end
```

C.10.9- IterativeLeastSquares.m

```
function [ PredTagLoc,Error ] = IterativeLeastSquares( AntLoc, TagLoc,
DistPred )
%%Ann Whitney%%
%%Originally Written%%
%%3/1/16%%
%%Last Edited%%
%%8/25/16%%

%%The purpose of this program is to find the most likely location of
the
```

```
%%unknown object using iterative least squares


%%Inputt%%
stepSize = 1e-5;


%%Solving%%
n = size(AntLoc,1);
Sigma = eye(n);
B = AntLoc;
%thetar = [mean(AntLoc(:,1)) mean(AntLoc(:,2))];
for i = 1:10
    sigma = sqrt(diag(Sigma));
    thetar = sum(B./[sigma sigma],1)./sum(1./[sigma sigma],1);
    xr = thetar(1);
    yr = thetar(2);
    X = [ones(n,1) 2*(AntLoc(:,1)-xr) 2*(AntLoc(:,2)-yr)];
    Y = [(AntLoc(:,1)-xr).^2+(AntLoc(:,2)-yr).^2-DistPred.^2'];
    vInv = stepSize*eye(3);
    beta = inv(X'*inv(Sigma)*X+vInv)*X'*inv(Sigma)*Y;
    theta = beta(2:end)+thetar';
    x(i) = theta(1);
    y(i) = theta(2);
    di2 = [((x(i)-AntLoc(:,1)).^2+(y(i)-AntLoc(:,2)).^2)];
    sigma2 = abs((1/n)*sum((DistPred.^2')-di2));
    Sigma = diag(4*di2.*sigma2);
%     Error = sqrt((x(end)-TagLoc(1))^2+(y(end)-TagLoc(2))^2)
end
PredTagLoc = theta;
Error = sqrt((x(end)-TagLoc(1))^2+(y(end)-TagLoc(2))^2);

end
```

C.10.10- NonlinearLeastSquares.m

```
function [ PredTagLoc,Error ] = NonlinearLeastSquares( AntLoc, TagLoc,
DistPred, Guess )
%%Ann Whitney%%
%%Originally Written%%
%%3/21/16%%
%%Last Edited%%
%%11/14/17

%%The purpose of this program is to find the most likely location of
the
%%unknown object using nonlinear iterative least squares
%% From Statistical methods in surveying by trilateration
% Guess = [0.0021 -0.0010];
% DistPred = [3.2558 5.3757 4.7103 1.7095];
% TagLoc=[-0.6909 2.2130];
% AntLoc = [0 0;2.0116 1.7738;0.3020 3.9090;-1.4946 0.6832];
%%
Theta=Guess';

%% Theta1 Linear Least Squares Approximation %%
```

```
% Thetabar = mean(AntLoc,1)';
% xr = Thetabar(1);
% yr = Thetabar(2);
% Xstar = [2*(AntLoc(:,1)-xr) 2*(AntLoc(:,2)-yr)];
% Y = [(AntLoc(:,1)-xr).^2+(AntLoc(:,2)-yr).^2-DistPred.^2'];
% Ystar = Y+Xstar*Thetabar;
% Theta = inv(Xstar'*Xstar)*Xstar'*Ystar;
x = Theta(1);
y = Theta(2);

AntLoc(abs(y-AntLoc(:,2))<0.005,2)=AntLoc(abs(y-
AntLoc(:,2))<0.005,2)+0.5;
AntLoc(abs(x-AntLoc(:,1))<0.005,1)=AntLoc(abs(x-
AntLoc(:,1))<0.005,1)+0.5;

%% Newton's Method Applied to Nonlinear Least Squares %%
for i = 1:50
    di2 = ((x-AntLoc(:,1)).^2+(y-AntLoc(:,2)).^2);
    f = sqrt(di2)-DistPred';
    JacobianX = sum(((x-AntLoc(:,1)).^2)./di2);
    JacobianY = sum(((y-AntLoc(:,2)).^2)./di2);
    JacobianXY = sum(((x-AntLoc(:,1)).*(y-AntLoc(:,2)))./di2);
    JtJ = [JacobianX JacobianXY;JacobianXY JacobianY];
    Jf = [sum(((x-AntLoc(:,1)).*f)./di2);sum(((y-
AntLoc(:,2)).*f)./di2)];
    Theta = Theta-inv(JtJ)*Jf;
    x = Theta(1);
    y = Theta(2);
end
PredTagLoc = Theta';
Error = sqrt(sum((PredTagLoc-TagLoc).^2));

End
```

C.10.11- Additional Functions

Note: PlotCircle.m (C.10.11.1) is used to graph circles to represent the distance measurement from the satellite antenna

C.10.11.1- PlotCircle.m

```
function PlotCircle( x,y,r,color )
%x and y are the coordinates of the center of the circle
%r is the radius of the circle
%0.01 is the angle step, bigger values will draw the circle faster but
%you might notice imperfections (not very smooth)
ang=0:0.01:2*pi;
xp=r*cos(ang);
```

```
yp=r*sin(ang);
plot(x+xp,y+yp,color);
end
```

C.10.11.2- mySlope.m

```
function [ slope,b ] = mySlope( x,y )
%% The purpose of this function is to quickly calculate the slope and
intercept

%%Linear Regression%%
m1 = mean(x.*y);
m2 = mean(x)*mean(y);
m3 = mean(x.^2);
m4 = mean(x)^2;
slope = (m1-m2)/(m3-m4);
b = mean(y)-slope*mean(x);

end
```

C.10.11.3- atan3.m

```
function [theta] = atan3(y,x)
%The purpose of this function is to calculate the arctan function
%accounting for the quadrents of x and y, and giving the output ranging
%from 0 to 2pi
theta = atan(y./x)+(pi/2)*(2-sign(y)-sign(y./x));
end
```

C.11: Antenna Pattern Method

Note: AntGradientDescent.m (C.11.1) performs the trilateration method of the Antenna Method

C.11.1- AntGradientDescent.m

```
function [ TagLoc,Error ] = AntGradientDescent( AntLoc, ActTagLoc,
Dist,InitTagLoc,SigError)

addpath('D:\My Stuff\School\Bluetooth\MATLAB')
x = AntLoc(:,1);
x=x(~isnan(Dist));
y = AntLoc(:,2);
y=y(~isnan(Dist));
Dist=Dist(~isnan(Dist));
S = 1;
[~,small]=min(Dist);
Cx = InitTagLoc(1);
Cy = InitTagLoc(2);
A = atan2(y(small)-Cy,x(small)-Cx);
X = [S;A;Cx;Cy];
n = length(x);
n=length(Dist);
J = zeros(n,4);

%%USER INPUTS%%
graph =0;
deepLearning = logical([1 1 1 1]);
gamma = [0.1;0.01;0.1;0.1];
alpha= 0.1;
h=0.8;
iterations = 10;
%%%%%%%%%%%%%%%



gamma = gamma(deepLearning);
for k = 1:iterations;
    S = X(1);
    A = X(2);
    Cx = X(3);
    Cy = X(4);
    null = zeros(1,n);
    for i =1:n
        %Initial Values%
        phi = atan2(y(i)-Cy,x(i)-Cx)-A;
        if (abs(phi-(pi/4))<alpha)||(abs(phi-(3*pi/4))<alpha)||...
                (abs(phi+(pi/4))<alpha)||(abs(phi+(3*pi/4))<alpha)
            null(i)=1;
        end
        r1 = cos(pi.*cos(atan(2*tan(phi))));
        r2 =cos(atan(2*tan(phi)));
```

```
        r3 = 0.5*sin(atan(2*tan(phi)));
        r4 = ((3+h)/4)+(((2*h-2)/(pi*sqrt(2))).*cos(phi))+...
            (((2*h-2)/(2*pi)).*cos(2*phi))+(((2*h-
2)/(3*pi*sqrt(2))).*cos(3*phi))+...
            (((-2*h+2)/(5*pi*sqrt(2))).*cos(5*phi))+(((-
2*h+2)/(6*pi)).*cos(6*phi))+...
            (((-2*h+2)/(7*pi*sqrt(2))).*cos(7*phi));
        r5 = sqrt((r1*r2)^2+(r1*r3)^2);
        r = r5*r4;
        Tx = (r*S)*cos(phi+A)+Cx;
        Ty = (r*S)*sin(phi+A)+Cy;
        G(i,1)=sqrt((Tx-x(i))^2+(Ty-y(i))^2)-Dist(i);

        %Gradient%
        d_phi(1) = 0;
        d_phi(2) = -1;
        d_phi(3) = (y(i)-Cy)/((y(i)-Cy)^2+(x(i)-Cx)^2);
        d_phi(4) = (Cx-x(i))/((y(i)-Cy)^2+(x(i)-Cx)^2);

        d_r1=sin(pi*cos(atan(2*tan(phi))))*pi*sin(atan(2*tan(phi)))*...
            (-4/(3*cos(2*phi)-5));
        d_r2=-sin(atan(2*tan(phi)))*(-4/(3*cos(2*phi)-5));
        d_r3 = 0.5*cos(atan(2*tan(phi)))*(-4/(3*cos(2*phi)-5));
        d_r4 = (((-(2*h-2)/(pi*sqrt(2))).*sin(phi))+...
            ((-2*(2*h-2)/(2*pi)).*sin(2*phi))+...
            ((-3*(2*h-2)/(3*pi*sqrt(2))).*sin(3*phi))+...
            ((-5*(-2*h+2)/(5*pi*sqrt(2))).*sin(5*phi))+...
            ((-6*(-2*h+2)/(6*pi)).*sin(6*phi))+...
            ((-7*(-2*h+2)/(7*pi*sqrt(2))).*sin(7*phi)))*d_phi;
        d_r5 = ((r1*r2)*(r2*d_r1*d_phi+r1*d_r2*d_phi)+...

(r1*r3)*(r3*d_r1*d_phi+r1*d_r3*d_phi))/sqrt((r1*r2)^2+(r1*r3)^2);

        d_r = (r4*d_r5)+(r5*d_r4);

        d_Tx(1) = r*cos(phi+A);
        d_Tx(2) = S*cos(phi+A)*d_r(2)-S*r*sin(phi+A)*(d_phi(2)+1);
        d_Tx(3) = S*cos(phi+A)*d_r(3)-S*r*sin(phi+A)*d_phi(3)+1;
        d_Tx(4) = S*cos(phi+A)*d_r(4)-S*r*sin(phi+A)*d_phi(4);

        d_Ty(1) = r*sin(phi+A);
        d_Ty(2) = S*sin(phi+A)*d_r(2)+S*r*cos(phi+A)*(d_phi(2)+1);
        d_Ty(3) = S*sin(phi+A)*d_r(3)+S*r*cos(phi+A)*d_phi(3);
        d_Ty(4) = S*sin(phi+A)*d_r(4)+S*r*cos(phi+A)*d_phi(4)+1;

        J(i,:)=(((x(i)-Tx)*-d_Tx)+((y(i)-Ty)*-d_Ty))...
            /sqrt((x(i)-Tx)^2+(y(i)-Ty)^2);
    end
    if graph==1
        GraphAntLoc( AntLoc, ActTagLoc, Dist,Cx,Cy,S,A,SigError,h,null)
    end

    null = ~logical(null');
    J = J(null,deepLearning);
    G = G(null);

    grad_F = J'*G;
```

```
    X(deepLearning) = X(deepLearning)-gamma.*grad_F;
    F = (1/(n-1))*G'*G;
    clear J G
end
TagLoc = X(3:4)';
Error = sqrt(sum((ActTagLoc-TagLoc).^2));
end
```

Note: GraphAntLoc.m (C.11.2) is used by AntGradientDescent (C.11.1) to graph the results

C.11.2- GraphAntLoc.m

```
function  GraphAntLoc( AntLoc, ActTagLoc,
Dist,Cx,Cy,S,A,SigError,h,null )
%%The purpose of this function is to graph the current antenna
localization
%%algorithm
hold off

colors = [0 0.447 0.741;0.85 0.325 0.098;0.929 0.694 0.125;...
    0.494 0.184 0.556;0.466 0.674 0.188;0.301 0.745 0.933;0.635 0.078
0.184];

plot(ActTagLoc(1),ActTagLoc(2),'k*')
hold on
phi = 0:0.01:2*pi;
r1 = cos(pi.*cos(atan(2*tan(phi))));
r2 =cos(atan(2*tan(phi)));
r3 = 0.5*sin(atan(2*tan(phi)));
r4 = ((3+h)/4)+(((2*h-2)/(pi*sqrt(2))).*cos(phi))+...
    (((2*h-2)/(2*pi)).*cos(2*phi))+(((2*h-
2)/(3*pi*sqrt(2))).*cos(3*phi))+...
    (((-2*h+2)/(5*pi*sqrt(2))).*cos(5*phi))+(((-
2*h+2)/(6*pi)).*cos(6*phi))+...
    (((-2*h+2)/(7*pi*sqrt(2))).*cos(7*phi));
r = sqrt((r1.*r2).^2+(r1.*r3).^2).*r4;
Tx = (r*S).*cos(phi+A)+Cx;
Ty = (r*S).*sin(phi+A)+Cy;
plot(Cx,Cy,'g*')
plot(Tx,Ty)
clear phi r Tx Ty
for i = 1:length(Dist)
    PlotCircle(AntLoc(i,1),AntLoc(i,2),Dist(i),'b')
%     PlotCircle(AntLoc(i,1),AntLoc(i,2),Dist(i)-SigError(i),'r')
    phi = atan2(AntLoc(i,2)-Cy,AntLoc(i,1)-Cx)-A;
    r1 = cos(pi.*cos(atan(2*tan(phi))));
    r2 =cos(atan(2*tan(phi)));
    r3 = 0.5*sin(atan(2*tan(phi)));
    r4 = ((3+h)/4)+(((2*h-2)/(pi*sqrt(2))).*cos(phi))+...
    (((2*h-2)/(2*pi)).*cos(2*phi))+(((2*h-
2)/(3*pi*sqrt(2))).*cos(3*phi))+...
    (((-2*h+2)/(5*pi*sqrt(2))).*cos(5*phi))+(((-
2*h+2)/(6*pi)).*cos(6*phi))+...
    (((-2*h+2)/(7*pi*sqrt(2))).*cos(7*phi));
```

```
    r = sqrt((r1.*r2).^2+(r1.*r3).^2).*r4;
    Tx = (r*S).*cos(phi+A)+Cx;
    Ty = (r*S).*sin(phi+A)+Cy;
    if null(i)==1
        plot(Tx,Ty,'r*');
    else
        plot(Tx,Ty,'g*')
    end
    plot(AntLoc(i,1),AntLoc(i,2),'r*')
end
legend('Actual Tag Loc','Antenna Pattern','Anchor Radius','Point on Ant
Patrn')
% legend('Actual Tag Loc','Antenna Pattern','Original Radius','RSSI Sig
Radius','Point on Ant Patrn')
% legend('Actual Device Loc','Linear Loc Estimate','Anchor Radius/Read
Dist','Anchor Location')
title('Trilateration')
xlabel('X Coordinate')
ylabel('Y Coordinate')
hold off
% plot(Cx,Cy,'r*')

End
```

APPENDIX D : OVERVIEW OF THE PROJECT

D.1.1 Phase 1

Phase 1Part 1: Review of the Literature.

A literature review was completed to identify state-of-the-art methods for localization based on the criteria of the problem statement.

Phase 1Part 2: Initial testing of distance measurement techniques.

RSSI (Received Signal Strength Indication) and phase-based distance measurements with RFID were tested as a viable option for accurate, low cost, indoor localization; as well as RSSI measurements for Bluetooth Low Energy (BLE) sensors for longer range applications. Testing was also done using algorithms such as near neighbor and fingerprinting to identify the tradeoffs associated with these commonly used localization methods, as well as their potential to be used in conjunction with other algorithms. Line of sight readings in 5 and 10 meter fully and semi-anechoic chambers were used to accurately model and understand the characteristics of the wireless technologies. Additionally, testing was performed in a variety of "real world" environments observe the impact of common disturbances and how they can be mitigated to find effective solutions.

Research into Distance Measurement Accuracy

With the technology selection, both a literature search and experimental investigation were performed to test the accuracy of various distance estimation methods. RSSI measurements are possible for both RFID and BLE for distance estimation but are commonly understood to be significantly impacted by the surrounding physical environment, as will be discussed in detail in the coming chapters. Phase angle, which in

this case measures phase change between different carrier frequencies, is possible with RFID but not BLE. Phase measurements are believed to be more robust with regards to "real world" environments but are not suitable for all applications.

D.1.2 Phase II

Phase 2 Part 1: Localization accuracy using phase-based distance techniques.

As expected phase-based distance techniques provided improved accuracy over standard RSSI techniques. A new and unique method, called RSSI-Informed phase, was developed using a combination of RSSI and Phase was developed through this work, to be robust with regards to "real world" environments, motion, and extreme tag angles. This algorithm was presented in "RSSI-Informed Phase Method for Distance Calculations," at the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, in 2015.

Phase 2 Part 2: The role of walls and obstructions.

Methods such as fingerprinting and near neighbor are often used to mitigate the impact of a "real world" environment, such as obstacles and reflections. These methods are time intensive to implement and maintain, therefore an alternative method to mitigate the impact was explored. The unique pattern of RSSI vs. frequency was found to relate directly to the multipath of the surrounding environment, and therefore utilized for the mitigation of multipath effects. This work lead to an understanding of multipath error, and a subsequent reduction of the error in distance measurements by approximately 50%. This work was published as a conference paper entitled "Received Signal Strength Indication Signature for Passive UHF Tags," [2].

Phase 2 Part 3: Moving toward a working prototype

The next step in the research process was to move from improved distance estimations to an exploration of trilateration algorithms.

D.1.4 Research into Distance Measurement Accuracy

Using empirical data, this research was able to apply a variety of localization algorithms to determine which methods yielded the highest accuracy and could handle a varying number of devices. Additionally, a method was developed in this research which enhanced accuracy by eliminating the assumption of isotropic antennas, and instead incorporating information about the antenna pattern to improve performance.

Phase 3 Part 1: Evaluate Localization Algorithms

Based on distance measurement techniques determined in phase 1, this research explored algorithms for location of Bluetooth and RFID tags while addressing the criteria of the problem statement that the number of tags (beacons) will vary. Location data was collected for both technologies and used to compare a variety of trilateration algorithms from the literature. Ten different localization algorithms using all applicable combinations of distance techniques were compared. The results for Bluetooth devices have resulted in a manuscript "Comparing Trilateration Algorithms for BLE Devices" which will soon submitted to IEEE for review.

Phase 3 Part 2: Improved Localization

A new method of trilateration was designed to address error from extreme antenna angle by fitting a simplified antenna pattern to a set of distance measurements and allowing that model to find the most likely location using an optimization algorithm. This method has been incorporated into a journal article "Antenna Pattern Method for Improved

Localization" which will soon be submitted to IEEE for review. In the final analysis, this method proved to be the most accurate for Bluetooth systems.

BIBLIOGRAPHY

[1] A. Whitney, J. Parker, M. Engelhard, Z. Kratzer and J. Fessler, "RSSI Informed Phase Method for Distance Calculations," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2015.

[2] A. Whitney, J. Fessler, J. Parker and N. Jacobs, "Received Signal Strength Indication Signature for Passive UHF Tags," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2014.

[3] A. M. Whitney, J. M. Parker, Z. C. N. Kratzer, J. T. Fessler and J. G. Whitney, "Reducing RF Distance Error by Characterizing Multipath," *IEEE Transations on Instrumentation and Measurement,* no. DOI 10.1109/TIM.2018.2875899, 2018.

[4] Alien, "ALN-9630 Squiglette Inlay," 16 December 2015. [Online]. Available: http://www.alientechnology.com/products/tags/squiglette/.

[5] RFID Journal, "RFID Frequently Asked Questions: How much do RFID readers cost today?," 2017. [Online]. Available: https://www.rfidjournal.com/faq/show?86.

[6] K. Vo and T. Dauwalder, "UPC Bar Codes," [Online]. Available: https://courses.cs.washington.edu/courses/cse370/01au/minirproject/TeamUPC/UPC.html.

[7] M. Roberti, "What IS the Read Range of a Passive RFID Tag," 14 August 2013. [Online]. Available: http://www.rfidjournal.com/blogs/experts/entry?10684.

[8] K. Ashton, "Whither the Five-Cent Tag?," 21 February 2011. [Online].

[9] A. Ozdemir and M. A. Bayrak, "Assessment of RFID Investment in the Military Logistics Systems through the Life Cycle Cost (LCC) Model," *Journal of Military and Information Science,* vol. 3, pp. 88-102, 2015.

[10] J. Säilä, "The Magic 5 Cents Per Tag," RFID Arena, 2012.

[11] T. Watson, "Simple Cost Analysis for RFID Options," AMI, 2013.

[12] S. Gezici, "A Survey on Wireless Position Estimation," *Wireless Personal Communications,* pp. 263-282, 2008.

[13] H. T. Friis, "A Note on a Simple Transmission Formula," *Proceedings of the I.R.E and Waves and Electrons,* pp. 254-256, 1946.

[14] W. L. Stutzman and G. A. Thiele, Antenna Theory and Design, 3rd ed., Hoboken, NJ: Wiley, 2013.

[15] Texas Instruments, "2.4-GHz Bluetooth® low energy System-on-Chip," June 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2540.pdf.

[16] MTi Wireless Edge , "SPECIFICATION FOR MT – 242025/NRH RHCP READER ANTENNA," MTI WIRELESS EDGE LTD., [Online]. Available: http://www.mtiwe.com/_Uploads/dbsAttachedFiles/MT-242025-NRH_EngSpec.pdf. [Accessed 2018].

[17] S. Subedi, E. Pauls and Y. D. Zhang, "Accurate Localization and Tracking of a Passive RFID Reader Based on RSSI Measurement," *IEEE Journal of Radio Frequency Identification,* vol. 1, no. 2, pp. 144-154 doi:10.1109/JRFID.2017.2765618, June 2017.

[18] A. A. Smith, R. F. German and J. B. Pate, "Calculations of Site Anttenuation From Antenna Factors," *IEEE Transactions on Electromagnetic Compatibility,* vol. 24 no. 3, no. doi: 10.1109/TEMC.1982.304041, pp. 301-316 , 1982.

[19] N. C. Gupta, Inside Bluetooth Low Energy, Boston, 2013.

[20] Merriam-Webster, "Definition of Backscatter," Merriam-Webster.com, [Online]. Available: https://www.merriam-webster.com/dictionary/backscatter. [Accessed 23 February 2019].

[21] A. Ashry, K. Sharaf and M. Ibrahim, "A compact low-power UHF RFID tag," *Microelectronics Journal,* vol. 40, pp. 1504-1523 doi:10.1016/j.mejo.2009.01.013, 2009.

[22] Y. Yao, J. Wu, Y. Shi and F. Foster Dai, "A Fully Integrated 900 M-Hz Passive RFID Transponder Front End with Novel Zero-Threshold RF-DC Rectifier," *IEEE Transactions on Industrial Electronics,* vol. 56, no. 7 doi: 10.1109/TIE.2008.2010180, pp. 2317-2325, 2009.

[23] M. Bouet and A. L. dos Santos, "RFID Tags Positioning Principles and Localization Techniques," in *1st IFIP Wireless Days doi: 10.1109/WD.2008.4812905*, Dubai, UAE, 2008.

[24] C. Zhou and J. D. Griffin, "Accurate Phase-Based Ranging Measurements for Backscatter RFID Tags," *IEEE Antennas and Wireless Propagation Letters,* vol. 11, no. doi: 10.1109/LAWP.2012.2186110, pp. 152-155, 2012.

[25] J. T. Prothro, G. D. Durgin and J. D. Griffin, "The Effects of a Metal Ground Plane on RFID Tag Antennas," in *IEEE Antennas and Propagation Society International Symposium*, Albuquerque, NM doi: 10.1109/APS.2006.1711302, 2006 .

[26] U. Kkarthaus and M. Fischer, "Fully Integrated Passive UHF RFID Transponder IC With 16.7-microWatt Minimum RF Input Power," *IEEE Journal of Solid-State Circuits,* vol. 38, no. 10, pp. 1602-1608, 2003.

[27] J. D. Griffin, G. D. Durgin, A. Haldi and B. Kippelen, "Radio Link Budgets for 915 MHz RFID Antennas Place on Various Objects," *Texas Wireless Symposium,* 2005.

[28] S. Subedi, Y. D. Zhang and M. G. Amin, "Precise RFID localization in impaired environment thorugh sparse signal recovery," *Proceedings of SPIE,* vol. 8753, pp. 87530H-1 doi:10.1117/12.2018144, 2013.

[29] E. Pauls and Y. D. Zhang, "Experimental studies of high-accuracy RFID localization with channel impairments," *Proc SPIE,* vol. 9497, p. Art. no 94970I, 2015.

[30] A. Lázaro, D. Girbau and D. Salinas, "Radio Link Budgets for UHF RFID on Multipath Environments," *IEEE Transactions on Antennas and Propagation,* vol. 57, no. 4, pp. 1241-1251 doi: 10.1109/TAP.2009.2015818, 2009.

[31] P. V. Nikitin and K. Rao, "Theory and Measurement of Backscattering from RFID Tags," *IEEE Antennas and Propagation Magazine,* vol. 48, no. 6, pp. 212-218 doi: 10.1109/MAP.2006.323323, December 2006.

[32] R. Want, "An Introduction to RFID Technology," Pervasive Computing, Jan-March 2006. [Online]. Available: https://www.cs.colorado.edu/~rhan/CSCI_7143_001_Fall_2002/Papers/rfid_intro_01593568.pdf. [Accessed January 2019].

[33] X. Chen, C. Wang and S. Lu, "Adaptive Accurate Indoor-Location Using Passive RFID," in *International Conference on Parallel and Distributed Systems*, Seoul, South Korea doi: 10.1109/ICPADS.2013.44, 2013.

[34] P. V. Nikitin, R. Martinez, S. Ramamurthy, H. Leland, G. Spiess and K. Rao, "Phase Based Spatial Identification of UHF RFID Tags," *IEEE International Conference on RFID,* pp. 102-109 doi:10.1109/RFID.2010.5467253, 2010.

[35] F. Martinelli, "A Robot Localization System Combining RSSI and Phase Shift in UHF-RFID Signals," *IEEE Transactions on Control Systems Technology,* pp. 1782-1796, 2015.

[36] S. Sarkka, V. V. Viikari, M. Huusko and K. Jaakkola, "Phase-Based UHF RFID Tracking With Nonlinear Kalman Filtering and Smoothing," *IEEE Sensors Journal,* vol. 12, no. 5 doi: 10.1109/JSEN2011.2164062, pp. 904-910, 2012.

[37] W. Li, X. Wang and B. Moran, "Distance Estimation Using Wrapped Phase Measurement Noise," *IEEE Transactions on Signal Processing,* vol. 61, no. 7 doi: 10.1109/TSP.2013.2238934, pp. 1676-1688, 2013.

[38] V. Viikari, P. Pursula and K. Jaakkola, "Ranging of UHF RFID Tag Using Stepped Frequency Read-out," *IEEE Sensors Journal,* vol. 10, no. 9 doi: 10.1109/JSEN.2010.2045497, pp. 1535-1539, 2010.

[39] S. Preradovic and N. C. Karmakar, "RFID Readers- A Review," in *International Conference on Electrical and Computer Engineering*, Dhaka, Bangladesh doi: 10.1109/ICECE.2006.355300, 2006.

[40] J. C. Carrick, "Methods and Apparatuses for RFID Tag Range Determination". US Thing Magic Inc Asignee Patent US20100109903A1, 6 May 2010.

[41] J. Fessler, Z. Kratzer, J. Parker, A. Whitney and J. Whitney, "Methods and Systems for Estimating Distance of Radio Frequency Identification Tag". US Lexmark International Assignee Patent US20160102964, 14 April 2016.

[42] K. Aspelin, "Establishing Pedestrian Walking Speeds," 25 May 2005. [Online]. Available: https://www.westernite.org/datacollectionfund/2005/psu_ped_summary.pdf. [Accessed 24 Aug 2009].

[43] D. Hahnel, W. Burgard, D. Fox, K. Fishkin and M. Philipose, "Mapping and Localization with RFID Technology," in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, USA doi: 10.1109/ROBOT.2004.1307283, 2004.

[44] J. Wang, M. Amin and Y. Zhang, "Signal and array processing techniques for RFID readers," *Proc SPIE,* vol. 6248, p. Art no. 624807 doi:10.1117/12.666712, 2006.

[45] Y. Zhang, X. Li and M. Amin, "Principles and techniques for RFID positioning," in *RFID Systems, Research Trends and Challenges*, John Wiley, 2010, pp. 389-415.

[46] C.-H. Huang, L.-H. Lee, C. C. Ho, L.-L. Wu and Z.-H. Lai, "Real-Time RFID Indoor Positioning System Based on Kalman-Filter Drift Removal and Heron-Bilateration Location Estimation," *IEEE Transactions on Instrumentation and Measurement,* vol. 64, no. 3, pp. pp. 728 - 739 doi: 10.1109/TIM.2014.2347691, March 2015.

[47] Q. Wu, Y. D. Zhang, M. G. Amin and F. Ahmad, "Through-the-wall radar imaging based on modified Bayseian compressive sensing," *Proc. IEEE China SIP, Xi'an China,* pp. 232-236 doi:10.1109/ChinaSIP.2014.6889238, July 2014.

[48] P. Setlur, G. E. Smith, F. Ahmad and M. G. Amin, "Target localization with a single sensor via multipath exploitation," *IEEE Transactions Aerospace Electronic System,* vol. 28, no. 3, pp. 1996-2014 doi: 10.1109/TAES.2012.6237575, 2012.

[49] ThingMagic, "M6e Hardware Guide," Trimble Navigation Limited, Cambridge, MA, 2012.

[50] R.-H. Wu, Y.-H. Lee, H.-W. Tseng, Y.-G. Jan and M.-H. Chuang, "Study of Characteristics of RSSI Signal," in *IEEE International Conference on Industrial Technology*, doi: 10.1109/ICIT.2008.4608603, 2008.

[51] ThingMagic, "Mercury6e UHF RFID Module Family Performance, Efficiency, and Flexibility," 20 March 2017. [Online]. Available: http://www.thingmagic.com/images/pdfs/M6e-Modules-datasheet.pdf. [Accessed 2018].

[52] Z. Ghahramani and M. I. Jordan, "Learning from Incomplete Data," DSpace@ MIT, 1995. [Online]. Available: http;//hdl.handle.net/1721.7202. [Accessed 27 February 2019].

[53] Z. J. Viharos and T. Vincze, "Training and Application of Artificial Neural Networks with Incomplete Data," in *IEA/AIE 2002: Developments in Applied Artificial Intellegence*, Berlin Heidelberg, Springer-Verlag, 2002, pp. 649-659.

[54] R. Heydon, Bluetooth Low Energy, The Developer's Handbook, Boston: Prentice Hall, 2013.

[55] A. Ng, *Neural Networks and Deep Learning,* Stanford University: Coursera, 2018.

[56] C. Fuhrer, "Cubic Spline," 2005. [Online]. Available: www.maths.lth.se/na/courses/FMN081/FMN081-06/lecture11.pdf. [Accessed Jan 2019].

[57] W. Navidi, W. S. Murphy Jr. and W. Hereman, "Statistical Methods in Surveying by Trilateration," *Computational Statistics & Data Analysis,* pp. 209-227, 1998.

[58] P. Barsocchi, S. Lenzi, S. Chessa and F. Furfari, "Automatic Virtual Calibration of Range-Based Indoor Localization Systems," *Wireless Communications and Mobile Computing,* pp. 1546-1557, 2012.

[59] H. Long, J. Lu and Q. Xu, "A New RSSI-based Centroid Localization Algorithm by Use of Virtual Reference Tags," *The Third International Conference on Advanced Communications and Computations,* pp. 122-128, 2013.

[60] P. Cotera, M. Velazquez, D. Cruz, L. Medina and M. Bandala, "Indoor Robot Positioning using an Enhanced Trilateration Algorithm," *International Journal of Advanced Robotic Systems,* pp. 1-8, 2016.

[61] K. Yu, I. Sharp and Y. J. Guo, "Appendix A: Hyperbolic Navigation," in *Ground-Based Wireless Positioning*, John Wiley & Sons, 2009, pp. 409-415.

[62] M. Z. Rahman, "Beyond Trilateration: GPS Positioning Geometry and Analytical Accuracy," in *Global Navigatoin Satellite Systems - Signal, Theory, and Applications*, P. S. Jin, Ed., Intech.com, 2012, pp. 241-257.

[63] Z. W. Mekonnen and A. Wittneben, "Localization via Taylor Series Approximation for UWB based Human Motion Tracking," *Positioning Navigation and Communications,* 2011.

[64] H. Li, "Low-Cost 3D Bluetooth Indoor Positioning with Least Square," *Wireless Personal Communications,* 2014.

[65] K. Chawla, G. Robins and L. Zhang, "Object Localization Using RFID," *IEEE International Symposium on Wireless Pervasive Computing,* 2010.

[66] J. Zhou and J. Shi, "Localization of Stationary Objects Using Passive RFID Technology," *International Journal of Computer Integrated Manufacturing,* vol. 22, no. 7, pp. 717-726, 2009.

[67] V. Weerackody and L. Gonzalez, "Motion Induced Antenna Pointing Errors in Satellite Communications On-The-Move Systems," *40th Annual Conference on Information Sciences and Systems,* pp. 961-966, 2006.

[68] L. C. Chen, R. K. Sheu, H. C. Lu and Y. P. Chu, "Object Finding System Based on RFID Technology," in *AP Web 2006 Lecture Notes in Computer Science*, doi: 10.1007/11610496_51, Springer-Verlag Berlin Heidelberg, 2006, pp. 383-396.

[69] E. Menegatti, A. Zanella, S. Zilli, F. Zorzi and E. Pagello, "Range-only SLAM with a Mobile Robot and Wireless Sensor Networks," in *IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[70] J. Mo, J. Lee and M. Simie, "Directional discrimination in radio frequency identification system for materials flow control in manufacturing and supply chain," *Proceedings IMechE, Journal of Engieering Manufacture,* vol. 223 Part B, no. doi: 10.1243/09544054JEMI1282, pp. 875-883, 2009.

[71] D. M. Dobkin and S. M. Weigand, "Environmental Effects on RFID Tag Antennas," in *Microwave Symposium Digest, IEEE MTT-S* , International doi: 10.1109/MWSYM.2005.1516541, 2005.

[72] G. Calis, B. Becerik-Gerber and A. B. Goktepe, "Analysis of the variability of RSSI values for active RFID-based indoor applications," *Turkish Journal of Engineering and Environmental Sciences,* vol. 37, no. doi:10.3906/muh-1208-3, pp. 186-210, 2013.

[73] Wikipedia, "Gradient Decent," Wikimedia, [Online]. Available: https://www.datacamp.com/search?q=neural. [Accessed 2 January 2019].

[74] N. Nygard, "Indoor Positioning Utilizing Bluetooth Smart," Stockholm, Sweden, 2015.

[75] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan and B. Ravidran, "Accurate Mobile Robot Localization in Indoor Environments using Bluetooth," in *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, 2010.

[76] Y. Zhao, A. LaMarca and J. R. Smith, "A Battery-free Object Localization and Motion Sensing Platform," in *UBICOMP*, Seattle, WA USA, 2014.

[77] Y. Zhang, W. Fu, D. Wei, J. Jiang and B. Yang, "Moving target localization in indoor wireless sensor networks mixed with LOS/NLOS situations," *EURASIP Journal on Wireless Communications and Networking,* vol. 291, 2013.

[78] C. A. Peters, "Statistics for Analysis of Experimental Data," Department of Civil and Environmental Engineering at Princeton University, 2001.

[79] Wikipedia, "Circular Polarization," Wikimedia, 25 February 2017. [Online]. Available: https://en.wikipedia.org/wiki/Circular_polarization.

[80] P. Setlur, G. E. Smith, F. Ahmad and M. G. Amin, "Target localization with a single sensor via multipath exploitation," *IEEE Transactions Aerospace Electronic System,* vol. 28, no. 3, pp. 1996-2014, 2012.

[81] AboutCivil.Org, "Civil Engineering Dictionary," AboutCivil.Org, [Online]. Available: http://www.aboutcivil.org/sources-of-errors-in-gps.html..

[82] W. M. Bohannon, "Signal Loss vs. Antenna Pointing Error," *Journal of Spacecraft and Rockets,* vol. 2, pp. 461-463, 1965.

[83] N. Truong, U. Jayasinghe, T.-W. Um and G. M. Lee, "A Survey on Trust Computation in the Internet of Things," *THE JOURNAL OF KOREAN INSTITUTE OF COMMUNICATIONS AND INFORMATION SCIENCES,* vol. 33, pp. 10-27, 2016.

[84] luminWayMaker, "Conical Sections," [Online]. Available: https://courses.lumenlearning.com/waymakercollegealgebra/chapter/equations-of-hyperbolas/. [Accessed 27 February 2019].

[85] A. Radhi, Data Acquisition and Controlling System by Using Mobile Phone Based Microcontroller via Bluetooth, 2015, pp. 319-339.

[86] G. Dean, "Demystifying iOS location-based services to improve Measurement Accuracy," 22 October 2012. [Online]. Available: https://www.techrepublic.com/blog/software-engineer/demystifying-ios-location-based-services-to-improve-accuracy/. [Accessed 6 March 2019].

[87] Stack Exchange, "Is it better to use a WiFi Channel with High Frequency?," 8 December 2016. [Online]. Available: https://superuser.com/questions/1154134/is-it-better-to-use-a-wifi-channel-with-high-frequency. [Accessed 11 March 2019].

VITA

Ann M. Whitney

EDUCATION

University of Kentucky: Ph.D., Mechanical Engineering **Jan 2012-Present**
Dissertation: Localization of Indoor Wireless Devices

Purdue University: B.S., Mechanical Engineering **Dec 2011**

PUBLICATIONS

3 Published Patent Applications

- US 20160003932
- US 20160102964
- US 20160104013

- "Reducing RF Distance Error by Characterizing Multipath" IEEE Transactions on Instrumentation and Measurement Print ISSN: 0018-9456 Online ISSN: 1557-9662 November, 2018 Digital Object Identifier: 10.1109/TIM.2018.2875899

- "RSSI-Informed Phase Method for Distance Calculations," in IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2015

- "Received Signal Strength Indication Signature for Passive UHF Tags," in IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2011

EXPERIENCE

Louisville Gas & Electric, Louisville Kentucky **Jan 2017- Present**
Research Engineer/Data Scientist

University of Kentucky, and Lexmark International Inc. **Aug 2012- Present**
Research and Development of Algorithms for Improved Localization of Wireless Devices
Advisor: Johné Parker, Ph.D.

Purdue University **Summer 2011**
Undergraduate Research Internship
Research advisor: Prof. George Chiu, Ph.D.

Lexmark International, Lexington, KY **Summers 2008 and 2010**
Hardware Research and Development Engineer, color laser printer development

Aurora Glass Jewelry, Georgetown, KY **2003-2009**
Co-Founder and Co-Owner