Hybrid Wi-Fi Localization using

RFID and UWB sensors

A Major Qualifying Project Report

Submitted to the faculty of

Worcester Polytechnic Institute

In partial fulfillment of the requirements

for the Degree of Bachelor of Science

By:

Chuka Ebi

Sebastian Franco

Tianxiong Wang

Project Advisor: *Professor* Kaveh Pahlavan

Abstract

The purpose of our major qualifying project is to develop a hybrid localization algorithm for precise indoor geolocation. There are two main parts to our project. The first involves conducting a performance evaluation on the technologies used: Wi-Fi, RFID, and UWB, to weigh their accuracy against their economic viability. This portion of the project allows us to gain an intimate understanding of each technology, their strength and their weaknesses. The second part of the project is to develop an algorithm that would take in data from Wi-Fi, RFID, and UWB, and produce a localization method that is significantly more accurate. For this part, we design and implement a hybrid localization algorithm to incorporate these various technologies to take advantage of their individual tracking abilities. This is achieved by utilizing Wi-Fi as the foundation, while RFID would act as a corrective measure. We combine RFID with Wi-Fi by using their individual RSS. The UWB portion of the algorithm will be activated if the user is in open space, and the UWB readings are within a certain distance. To increase the accuracy of our system even further, we use an Inertial Navigation System (INS) to count our steps in combination with Wi-Fi and RFID, using Kalman filter. The algorithm development and implementation, along with the analysis of the different technologies, is written in MATLAB. We are able to achieve a more accurate localization method than what is available if Wi-Fi alone is used.

Acknowledgements

We wish to acknowledge the guidance of Prof. Pahlavan, supportive help of PhD student Yishuang Geng, Master student Guanxiong Liu, Luyao Niu and Yingyue Fan and all the members of CWINS Labs for their cooperation during this project. We would also like to express our sincere appreciation to Dr. Chunjie Duan and Redpoint Positioning for donating the Ultra-wide Band system, and Prof. Massoud for providing the funding for the purchase of the RFID Reader.

Table of Contents

Chapter 1: Introduction

With an increase of interest in the field of indoor localization, companies are acquiring new assets, conducting research and developing systems that will be the foundation for the next generation of this technology. As demand increases for indoor localization [1], Google released its Indoor Maps, and many other companies have an interest in indoor Wi-Fi localization such as Skyhook Wireless, and Apple. These companies are currently tapping into this technology by trying to map Wi-Fi access points inside buildings. Although Wi-Fi localization is the most popular technique, there are numerous other techniques that can be used. The two others we will focus on are Radio Frequency Identification (RFID) and Ultra Wide-Band (UWB). Some companies have developed their system by these technologies for tracking packages and robots, inside a warehouse environment. For example, Kiva systems, which was recently purchased by Amazon, uses RFID to utilize their system. RedPoint positioning uses UWB to track mobile nodes within an open environment.

Precise localization utilizing those techniques is paramount for indoor mobile devices, such as smartphones, mobile robots etc. However, Wi-Fi alone does not provide the accuracy good enough to identify the location of the user due to multipath fading characteristic. Furthermore, it is also subjected to significant short term variations. A new system can be developed that balances both cost and accuracy based on the characteristics of Wi-Fi, RFID and UWB.

1.1 Project Description

In this project, we first collected three kinds of reference databases which were all manually taken. For Wi-Fi localization we stored MAC addresses and their relative RSS values. For RFID localization, each tag represented a location, with a corresponding RSS reading. For UWB, we set up all of the system nodes and anchors to test its precision, all locations were given in an x, y, z format. Upon testing, we combined Wi-Fi with RFID due to their similar characteristics, while UWB would be used in open areas.

Upon completion, all of the testing and scenarios took place in the third floor of Atwater Kent and we only used one reference database representing location. To identify location, priority was given to UWB, followed by RFID and Wi-Fi. We equipped the CWINS lab with an UWB system, and whenever an individual entered the CWINS laboratory, the system displayed the UWB indicated position. When an individual is outside of the CWINS laboratory, the RFID-WiFi system goes into effect. If an RFID tag is read, then the RFID tag location is assumed, otherwise we assumed the calculated WiFi location. In order to smooth out the trace of our position, we used a Kalman filter.

1.2 Project Outline

There are five chapters that will be covered in this report. Chapter one details the reasons why we chose this project, a general description of the MQP, and our goals. In Chapter two, we studied the three fundamental technologies which are Wi-Fi, UWB & RFID and give an overview on the history of modern localization methods outdoors and indoors. In Chapter three we will discuss the performance analysis of the three different technologies, specifically the development of the scenarios for the RFID, performance analysis of the Wi-Fi localization, and the performance analysis of UWB. Chapter four details the results of the Hybrid Wi-Fi localization system. It includes the final testing and scenario along with the graphical representation of our data, which supports our hybrid indoor localization system. In the end of the report, Chapter five, we address the conclusion, future work and suggestions for the next MQP team. We also have an Appendices chapter, which includes the code that was developed and used in this project. Lastly, there is a reference chapter stating all the various references and sources used to complete this report.

Chapter 2: Background in Localization Technologies

This chapter covers the information needed to better understand modern localization in outdoor and indoor environments. The chapter will provide a brief overview on the history of localization. It also provides a detailed introduction to the three fundamental technologies Wi-Fi, RFID, and UWB. It will cover the advantages and limitation of each technology and how we ultimately came to a decision on how to move forward with the project.

2.1 Modern Localization

The means of identifying ones location, whether inside a building, or outdoors, has proved extremely helpful, mainly for purposes of navigation, in reaching point B from point A. Although we most commonly relate navigation to modern instruments, many tools have already been used and invented for purposes of travel. Tools such as the Sun or the stars in the night sky, and instruments like the chronometer, astrolabe and compass all took part in designing the modern, more technically advanced methods of navigation.

2.1.1 Outdoor Localization

Localization in environments outside of our homes and buildings, dates as far back as 1983, this was a time when GPS technology began to emerge to the public, the United States Department of Defense understood the importance of this technology, the positive impact it would have on the public sector and the advantages it would bring to the development of the world.

GPS proved to work extremely well for outdoor environments, and it is now the means by which most outdoor navigation devices work. It is a set of 24 satellites which are owned and operated by the U.S Department of Defense, but is available for world-wide use. These satellites orbit our planet allowing ground receivers to identify their respective position/location. Each one of these Satellites contains an atomic clock, a radio and a computer. Its internal clock and computer (which keeps an account of its orbit), allow the satellite to identify its own position, which is then broadcasted by the radio.

The signal sent out by these Satellites is then accepted by a receiver, which traces and measures the received information through a process known as triangulation, in which these signals (containing positions, and distances from one another in space) are compared to one another, in order to pinpoint a location. Since most of the devices that require GPS are handheld and mobile, it is the receivers' duty to calculate direction and speed of travel. More modern applications of the GPS receivers are now being used for research purposes, these vary from monitoring moving glaciers to tracking volcanic activity. [1]

2.1.2 Indoor Localization

The GPS Satellite signal is often too weak to penetrate buildings, making GPS use for indoor applications practically useless, many indoor localization models have been proposed of which Wi-Fi localization has gained the most popularity. Wi-Fi localization, being the most popular type of indoor position, takes advantage of large amount of wireless access points found in buildings and urban areas. Providers of this service include Google, AlterGeo and Navizon.

This method is based on the measurement of RSS (Received Signal Strength), combined with fingerprinting, which identifies the device being used based on its MAC address or SSID.

Wi-FI positioning can be combined with cell phone tower triangulation and Global Positioning Systems, in order to provide a more accurate location indoors. However some of its limitations are that the device one is trying to localize, must be within the range of the Wi-Fi signal.[2]

[1] (What Is Global Positioning System (GPS))

[2] (Indoor/Outdoor Localization.)

2.2 RFID Localization

As the advances in technology continue at a magnificent trajectory, so does the need for our devices to be constantly connected. Recent deployment of radio frequency identification (RFID) technology for efficient asset tracking and management has made RFID tags and associated devices widely available with low cost and low energy usage. For example, there are active RFID tags that typically last for five to seven years with a compact battery as a reliable wireless signal transmitter; obviously passive RFID tags have practically no lifetime limit. Clearly RFID tags, at a coarser level, provide a cost-effective and energy-efficient way of solving the environment sensing problem. One straightforward solution is to attach one or more RFID tags to each object of interest in the environment. As RFID tags have a limited range of readability, by reading all the tags in the proximity, using a reader or similar device, a computer can approximate its environment based on the sensed objects. Additionally, a unique advantage of RFID technology over vision and other sensor based methods is that RFID tags do not require line of sight in order to be "seen" and thus avoid problems associated with occlusion.[3]

For the majority of us, card access to buildings, using a key to start a vehicle or scanning a bus or subway ticket have become part of our everyday routine. Many times we use automatic data capture technology that works on radio frequency electromagnetic technology. This field is most commonly known as (RFID) Radio frequency identification. This same type of technology is used to transit and track merchandise and objects from all the way from its manufacturing, down to its point of sale. RFID has many applications and cannot be ultimately defined for one sole purpose. Through this project, radio frequency identification will be studied in order to develop a localization system. Once a localization system has been developed, it will be merged with both Wi-Fi and UWB technology to develop a more accurate indoor

[3] (How RFID Works)

localization system. A basic RFID system is composed of RFID tags, RFID reader antenna and RFID reader control application software for user interface.

2.2.1 RFID Tags

RFID tags are composed of an integrated circuit or chip which is attached to an antenna that has been printed/stamped or vapor-deposited into what is often a paper substrate or PolyEthylene Therephlate (PET). When the antenna and chip are combined, these two are then placed between labels with adhesive on one side as seen on Figure 2. , or constructed into a more durable structure as seen in Figure 1. RFID tags can be powered through two different methods. An Active RFID tag is powered by the RF signal send by the RFID reader.

A Passive RFID tag has its own power source. In Table 1.1 we see a detailed comparison between the two different types of tags.

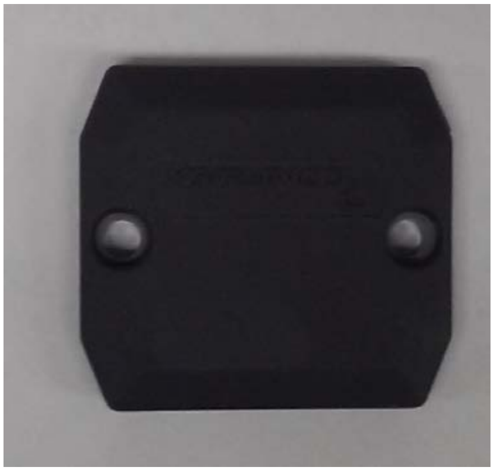


Fig 2.1 Solid Encasing Passive RFID tag

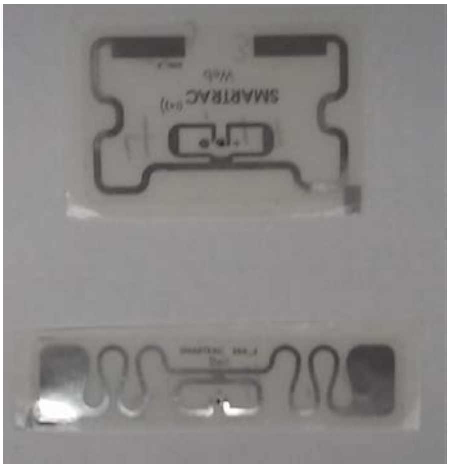


Fig 2.2 Flexible label Passive RFID tag

RFID TAGS

	Passive	Active
Read Range	Up to 40 feet (fixed readers) Up to 20 feet (handheld readers)	Up to 300 feet or more

Power	No power source	Battery powered
Tag Life	Up to 10 years depending upon the environment the tag is in	3-8 years depending upon the tag broadcast rate
Ideal Use	For inventorying assets using handheld RFID readers (daily, weekly, monthly quarterly, annually). Can also be used with fixed RFID readers to track the movement of assets as long as security is not a requirement.	For use with fixed RFID readers to perform real-time asset monitoring at choke-points or within zones. Can provide a better layer of security than passive RFID.
Readers	Typically higher cost	Typically lower cost

Table 2.1 (RFID Tags) Passive vs Active

2.2.2 RFID Reader

The RFID reader is composed of a scanning antenna that is responsible for putting out radio-frequency signals. These RF signals provide a method of communication between the RFID reader and the RFID tags. Once communication between the RFID reader and RFID tags is established, the reader can display all the information being transmitted and received through a software user interface.

RFID readers most commonly use two different types of antennas, polarized linear and circular linear. **Polarized linear antennas** broadcast on a single plane (either vertical or horizontal). These antennas tend to have a higher read range when pointed in the direction of the tag, its signal distribution is represented by Figure 3. **Polarized circular antennas** emit electromagnetic fields in a corkscrew –like fashion. These broadcast electromagnetic waves in two planes making a complete revolution in a single wavelength. These types of antennas lose about 3 dB per read now that they split the power across two separate planes. Figure 4 below shows the circular polarization.

2.2.3 RFID Localization Systems

Inventory Localization: KIVA Systems is a company located in North reading MA. Recently bought by Amazon, this company helps supply operations and reduce costs while increasing strategic flexibility. Current solutions store items in fixed locations which results in wasted time and energy spent keeping the facility organized.

With Kiva, inventory is free from physical location constraints, locations and positions are virtual and move and adapt to the products. The result is that any item can be delivered to any operator at any time. As part of the Mobile robotic fulfillment system, KIVA uses RFID tags in combination with robots, in order to track, send and receive packages in a warehouse environment.

Access Application: Most of the access cards implement the usage of an RFID tag, this tag contains information in regards to the identification of the individual. When the RFID tag is read by the RFID reader, it allows access to the building.

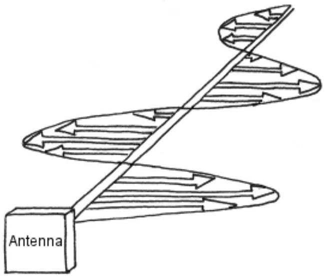


Fig. 2.3a Linear Polarized Antenna [4]

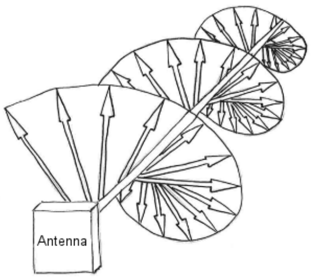


Fig. 2.3b Circular Polarized Antenna [4]

Upon completing the technical background research for RFID technology, several points were concluded. Passive RFID tags would be best to use now that the group will be working in a small environment, and a smaller reading range of RFID tags would also increase accuracy in terms of localization. When it came down to choosing a reader, the team decided to go with a Linear Polarized antenna. This decision was made now that most of the passive tags that will be read will have a known

[4] *(Circular Polarization vs Linear Polarization., Which is right?)*

location. Since the location of these tags is known, we will be able to point the antenna to their respective location, therefore increasing accuracy, and preventing other false tag reads.

2.3 Wi-Fi Localization

Indoor localization has been becoming an interesting and popular topic in past decades. The are many ways to realize the Wi-Fi positioning by measuring different parameters of Wi-Fi, such as time of arrival (TOA), angle of arrival (AOA) and received signal strength (RSS) etc. The most popular and fundamental method for Wi-Fi localization is using the measurement of received signal strength (RSS). The most widely applied algorithms for indoor localization also involve using RSS. There are two fundamental methods for indoor localization. These methods are k-nearest neighbor and kernel algorithm.

2.3.1 Wi-Fi Localization Algorithms

K-nearest neighbor method is for classification. According to the introduction of k-nearest neighbor by Altman, the output of k-nearest neighbor is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors, which k is usually small positive integer. If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor[5].

For indoor localization application, k-nearest neighbor locates the object inside of a building by measuring RSS of the object, and comparing it with the RSS of the access points (AP). It estimates the position of the object by selecting a K number of access points with the closest RSS to the object, and calculates the coordinates of the object by the weighted average coordinates of the K selected access points. The nearest neighbor method is a straightforward and efficient method, but sometimes not accurate enough for precise situations. **Kernel** methods are a class of algorithms for pattern analysis, which is to find and

[5] (How Does a Wi-Fi Positioning System Work?)

study of general types of correlation or classification in datasets[2]. It is based on the idea of using a probability mass function and the Gaussian curve to estimate the position of the object. The entire flow chart of our localization system is as shown below. The first step in building our Wi-Fi localization system is to start with constructing the database, which is illustrated below. [6]

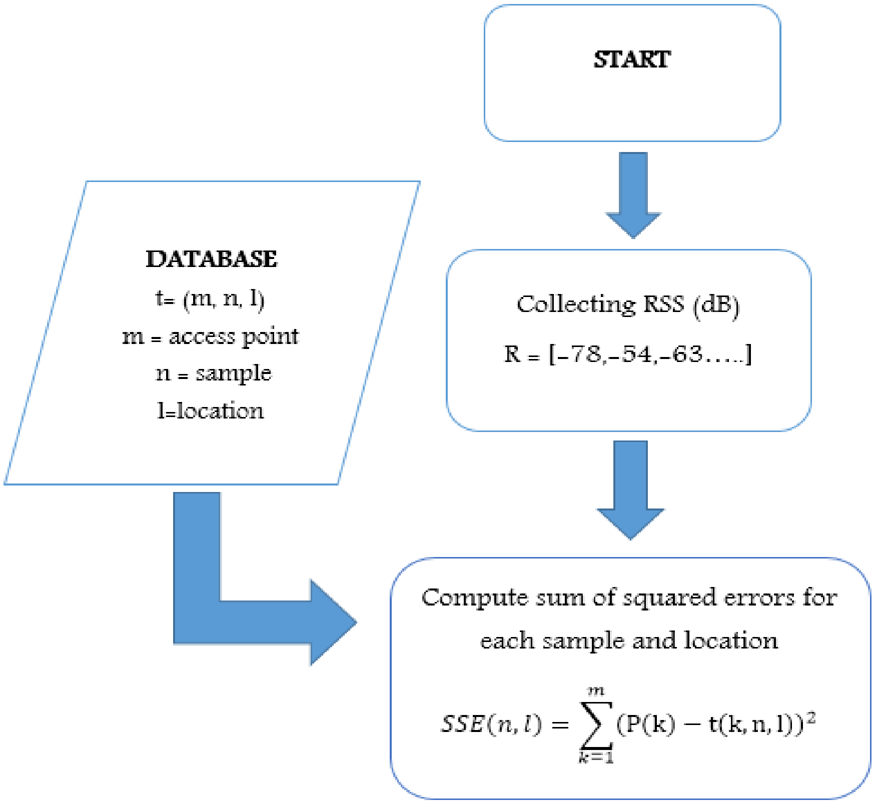


Figure 2.4: Diagram for constructing the database and computing the sum of squared errors for each sample and location using the received signal strength readings

We recorded the received signal strength measurements of 8 locations on the third floor of Atwater Kent Laboratory. At each location we collected training samples for one minute from 65 different access points. Using the received signal strength measurements from the object, we first calculated the sum of

[6] An introduction to kernel and nearest-neighbor nonparametric regression

square errors. Next, we conducted the calculations depicted on the next figure.

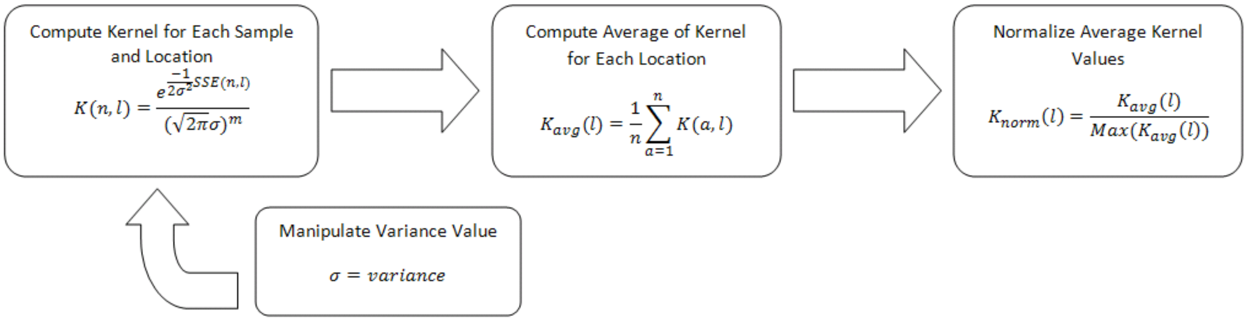

Compute Kernel for Each Sample and Location
$K(n,l) = \frac{e^{\frac{-1}{2\sigma^2}SSE(n,l)}}{(\sqrt{2\pi}\sigma)^m}$
Compute Average of Kernel for Each Location
$K_{avg}(l) = \frac{1}{n}\sum_{a=1}^{n} K(a,l)$
Normalize Average Kernel Values
$K_{norm}(l) = \frac{K_{avg}(l)}{Max(K_{avg}(l))}$
Manipulate Variance Value
$\sigma = variance$

Figure 2.5: Creating and normalizing the Kernel for probability estimate []

Using the SSE data and the variance value, we created the average Gaussian Kernel at each of the 8 locations. In addition, we had to tweak this value many times in order to achieve the most accurate localization estimations. Next, with the average Gaussian Kernel, we normalized it and calculated the probability as shown in the next figure.

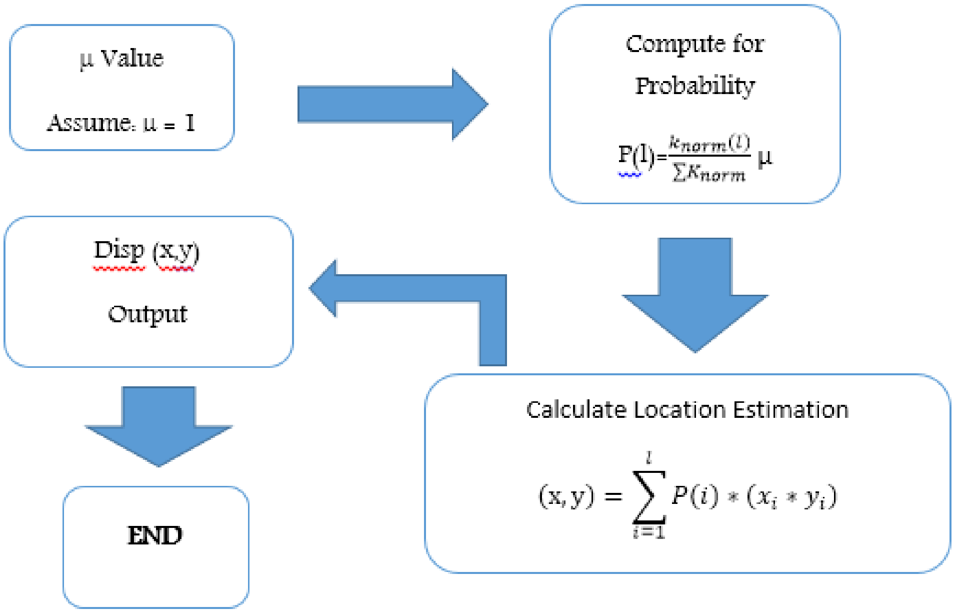

µ Value
Assume: µ = 1
Compute for Probability
P(l)=$\frac{k_{norm}(l)}{\sum K_{norm}}$ µ
Disp (x,y)
Output
END
Calculate Location Estimation
$(x,y) = \sum_{i=1}^{l} P(i) * (x_i * y_i)$

Figure 2.6: Calculating the probability of the robot in each location using the Kernel

2.3.2 Kalman Filter

Kalman filter is an efficient recursive filter. That means the filter can estimate the state of a dynamic system from a series of semi contained noise measurements. It is a non-linear dynamic system with time concern, it is often used in target tracking system. Later scholars were carried out a number of improvements, one of these improvements is the extended Kalman filter which can be applied to time nonlinear dynamic systems.

The basic idea of an Extended Kalman Filter is to linearize the nonlinear system, then apply the Kalman filter, so the EKF can be considered a suboptimal filtering tool. Subsequently, a variety of second-order extended Kalman filter method may be applied, these further improvements promote the performance of the Kalman filter to estimate of nonlinear systems.

2.4 UWB Localization

UWB is a radio technology and others which may be used at a very low energy level for short-range, high-bandwidth communications using a large portion of the radio spectrum.[1] UWB has traditional applications in non-cooperative radar imaging. Most recent applications target sensor data collection, precision locating and tracking applications. Recently, UWB has been utilized for localization applications. (UWB) radios have relative bandwidths larger than 20%or absolute bandwidths of more than 500 MHz. Such wide bandwidths offer a wealth of advantages for both communications and radar applications. In both cases, a large bandwidth improves reliability, as the signal contains different frequency components, which increases the probability that at least some of them can go through or around obstacles. Furthermore, a large absolute bandwidth offers high resolution radars with improved ranging accuracy. For communications, both large relative and large absolute bandwidth alleviate small-scale fading.[2] UWB is perceived as one of the enabling technologies for robust and accurate localization, especially in harsh channel environments like e.g. indoor areas.

An UWB localization system includes several fixed readers (UWB receiver), a set of moving tags (transmitter) and the processing center. The processing center, which connects the readers with wires, consists a baseband FPGA board and one computer (for graphic display and data processing). During the ranging detection, the tag is sending pulse stream to all the readers. After pulse detection, the readers send the data to the FPGA to measure the time difference of arrival (TDOA). Although UWB has its benefits for solving this task, several conditions have to be fulfilled to take advantage of them: the signal structure must enable the receiver to detect the arrival time as exactly as possible, the signal must have the possibility to travel a direct Line of Sight (LOS) path, and the receiver must detect the correct LOS path in multipath and interference environments.[7]

For UWB systems, OLOS(Obstructed Line Of Sight) Propagation does exist. When the direct LOS between two nodes is obstructed, only reflections of the UWB pulse from “strays” reach the receiving node. Therefore, the delay of the first arriving pulse does not represent the true TOA. Since the pulse travels an extra distance, a positive bias called the OLOS error is present in the measured time delay. [8]

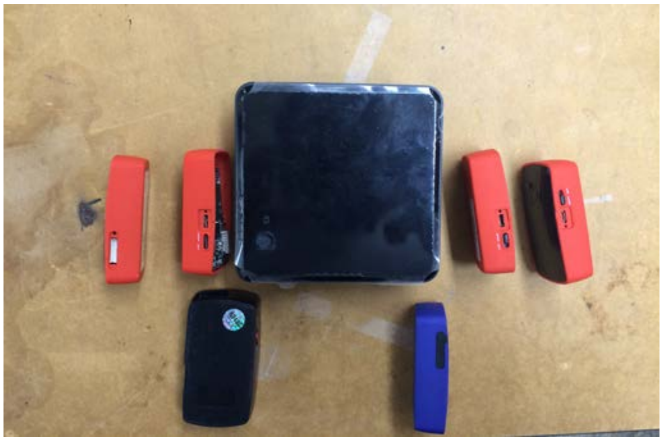


Fig. 2.7 Showing the Ultra Wide-Band System. The black box is the Base station(Black box), the fixed nodes(Orange nodes), the bridge node(black node), and the mobile node(blue node).

[7] (UWB localization - active and passive approach)

[8] Localization via Ultra-Wideband radios a look at positioning aspects for future sensor networks)

2.4.1 UWB Localization Techniques

Most Ultra Wideband technology use time-of flight based algorithms to estimate the location of an object. Two of the most popular are time-of-arrival (TOA), and time-difference-of-arrival (TDOA). In both the time of arrival (TOA) and time difference of arrival (TDOA) arrangements, omnidirectional antennas are not used and location is found by trilateration using distance data only.

2.4.2. Time of Arrival (TOA)

Distance can be estimated using received signal strength (RSS) data, or time-of-flight measurements. Theoretically, at least two fixed terminals are needed in order to pinpoint a location in two dimensions. In a multilateral system, like the system that will be described later on, fixed terminal receivers are used to estimate distance to a transmitting target.

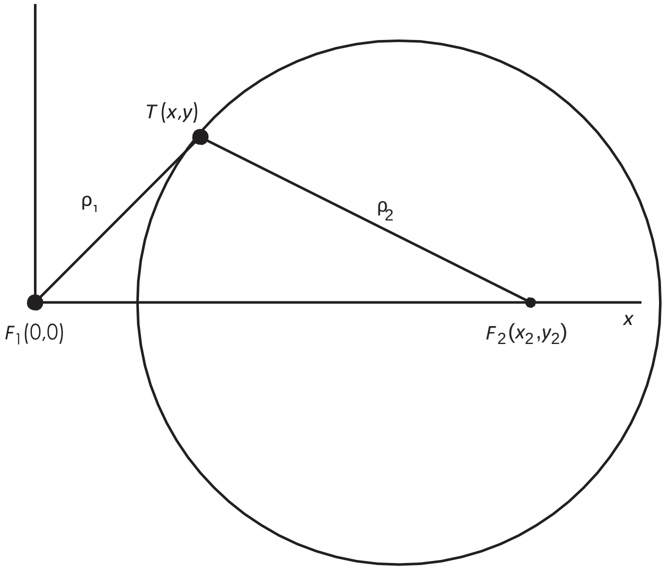


Figure 2.8. TOA location measurement configuration

The geometry of determining two-dimensional location from distance measurements is shown in Figure 2.13. The coordinates of two fixed terminals, *F*1 and *F*2 are known in a given frame of reference specified by the *x*-*y* axis and the origin at *F*1. If we can find the ρ_1 and ρ_2, we can determine the coordinates of *T* from a point of intersection of two circles. Since the circles intersect in two locations, we will assume the ambiguity is solved by knowing that *T* is in the upper half of the *x-y* plane. If there is no other knowledge to eliminate the ambiguity, a third fixed terminal is required. Equations to calculate the distances ρ_1 and ρ_2 are below:

$$p_1 = (t_1 - t0) \cdot c$$
$$p_2 = (t_2 - t0) \cdot c$$

Eq 2.1 Distance Calculation

Using the method denoted by TOA, we find the distances ρ_1 and $\rho_{2.}$ In the case of TOA, the one-way distance between *T* to *F*1 or *F*2 is determined as follows. Assume that all three stations have high precision clocks that are set to exactly the same time and that *F*1, *F*2 are receivers. A pulse sent from *T* at time *t*0 is received at *F*1 at time *t*1 and at *F*2 at time *t*2. *T* notifies *F*1 and *F*2 of the time of transmission, *t*0, by time-stamping its message. Now *F*1 and *F*2 can calculate the distances ρ_1 and ρ_2 from the transmit and receive times and the known propagation speed, the speed of light, *c*

The equations for the two circles with radiuses of ρ_1 and ρ_2 are:

$$\rho_1{}^2 = x^2 + y^2$$

$$\rho_2{}^2 = (x - x_2)^2 + (y - y_2)^2$$

Eq 2.2 Circle Equations for P_1 and P_2

2.4.2 Time Difference of Arrival (TDOA)

While TOA gives a straightforward way to find location from distance measurements, it does have disadvantages for many applications. Accurate, synchronized clocks must be maintained in all stations participating in the measurements. Information must be passed from the initiator to the receiver specifying when the transmission was started. Another geometric location method, TDOA, does not possess these disadvantages. All TDOA needs is a transmission that has a recognizable unambiguous starting point. The data used in the location calculations is the time difference in the reception of that starting point at the several base stations, and not the actual time of flight of the signal from the target to the fixed stations. In an arrangement having a mobile target whose coordinates are to be determined and two fixed base stations, as we had in the example of TOA, we can find the time difference of arrival of a signal sent from the mobile and received at the base stations. This one time difference value is not enough to calculate the two coordinate values of the mobile's position. So, in order to have sufficient data to find two unknowns—the mobile's coordinates—TDOA requires one more base station than TOA. The clocks of the fixed stations must be synchronized, but not that of the target. TDOA is used unilaterally—the target finds its own position from fixed station transmissions or multilaterally, where time difference data is collected from target transmissions by fixed base station receivers.

The figure above shows the geometric layout for TDOA in two dimensions. Target *T* transmits a pulse at *t*0 that is received at *F*1 at *t*1 and at *F*2 at *t*2. The clocks of *F*1 and *F*2 are synchronized, but *T*'s clock is not, so *t*0 is not known. However, the time difference of arrival, which can be calculated. The times on the right side of the equation are proportional to the distances *d*1 and *d*2 shown in Figure 2.14 since the distance is the time of flight times the speed of light, *c*. Therefore the difference of the distances between the two fixed stations and the target is:

$$\Delta d = d2 - d1 = c(t2 - t1)$$

Eq 2.3 Distance difference between two fixed stations

The time difference of arrival that is obtained from times of arrival measured at two synchronized fixed stations indicates that the target is located somewhere on a hyperbola. The particular branch of the hyperbola that the target is on is the one that is closest to the fixed station that received the signal first. Figure 2.14 is drawn with *F*1 and *F*2 on the *x*-axis and each at equal distance, *D*/*2*, from the origin. The expression for the hyperbola is:

$$\frac{x^2}{\mathrm{a}^{2}} - \frac{y^2}{b^2} = 1$$

Eq 2.4 Hyperbola Equation

Where a, and b are in terms of the known quantities Δd and D. Which amount to:

$$a^2 = (\Delta \mathrm{d}\ /2)^2$$

$$b^2 = \left(\frac{D}{2}\right)^2 - a^2$$

Eq 2.5 Hyperbola Equation Variables

Since the time difference of arrival found from TOA measurements by two terminals places the target on a locus of positions, it is necessary to use the time of arrival at a third fixed station, *t*3, to pinpoint the target location. With the addition of this one station, we can now find three time differences of arrival: between *F*1 and *F*2, *F*2 and *F*3, and *F*1 and *F*3. The intersection of a minimum of two hyperbolas, constructed from two times of arrival determinations and drawn on the same coordinate system, gives the location of *T*, as shown in the above Figure. The second hyperbola, shown as a solid curve, is based on the time difference of arrival between *F*2 and *F*3.

2.4.3 Red Point Positioning

The UWB system implemented for our MQP was provided by Red Point Positioning Corporation; a start-up based in Cambridge, Massachusetts. Red Point Positioning's Real-Time Location System (RTLS) uses UWB technology to provide indoor position information in 2D, 2.5D and 3D.

For the purpose of this MQP, we will pay more attention to the 2D positioning. For most applications, an RTLS is comprised of a single base station and multiple radio nodes. The radio nodes create a wireless network from which the system gathers localization data. The base station acts as a configuration and management hub for the network and stores the location and sensor data for each radio node in the system. There are three type of radio nodes: **Bridge Nodes**: A bridge node stays fixed in place and uses an Ethernet connection to serve as a gateway between the wireless network and the base station. **Anchor Nodes**: Anchor nodes are mounted on the walls and ceilings in a predetermined configuration to serve as fixed reference points and make up the rest of the network infrastructure. **Mobile Nodes**: Mobile nodes are attached to objects or people as tags and move around the network. The system tracks their location in order to provide precision tracking within the network [9]

As mentioned earlier, the anchor nodes and bridge nodes make up the network infrastructure. They dynamically establish a mesh network comprised of all active radio nodes in the system. Mobile and anchor nodes exchange messages wirelessly to determine the distance between them. This process is known as *ranging*. Once a mobile node has ranged with enough anchors, it computes its 2D, 2.5D, or 3D location. The mobile nodes then either send that location data back to the base station or out to an external device such as a phone or tablet.

[9] (Redpoint Positioning Localization, “Developers Guide”)

Chapter 3: Performance Analysis of the Different Technologies

This chapter performs a comparative performance evaluation of the three different technologies. We went through each one of the technologies, set up a scenario, and performed testing accordingly. For the RFID reader, we developed an antenna model, and implemented RFID tag writing, for Wi-Fi we tested two different localization algorithms, and for the UWB we developed models for its localization behavior.

3.1 RFID Modelling and Scenario Setup

The USB Plus+RFID Reader sold by the company ThingMagic is a low cost platform for developing interactive read/write applications. It contains a reader with a linear polarized antenna and a set of sample passive RFID tags which made it perfect for the team's desired application. This device is driven by ThingMagic's Mercury 5e-Compact UHF RFID reader module, the USB *Plus+* is controlled and powered by a host PC or laptop through a USB interface and is compatible with ThingMagic's application development tools, permitting rapid creation of RFID solutions. With a software adjustable read distance up to 3 ft (0.91 m), the USB *Plus+* supports a variety of applications, including RFID tag commissioning, manufacturing WIP, document tracking, library book check in/out, retail point of sale, event and hospitality services, hospital patient workflows, and more. The high-performance internal antenna of the USB *Plus+* is also ideal for commissioning high memory tags and reading small form factor RFID tags more effectively. Figure 3.1 below shows the RFID reader with the respective sample tags. [10]

[10] (USB Plus+ RFID Reader)

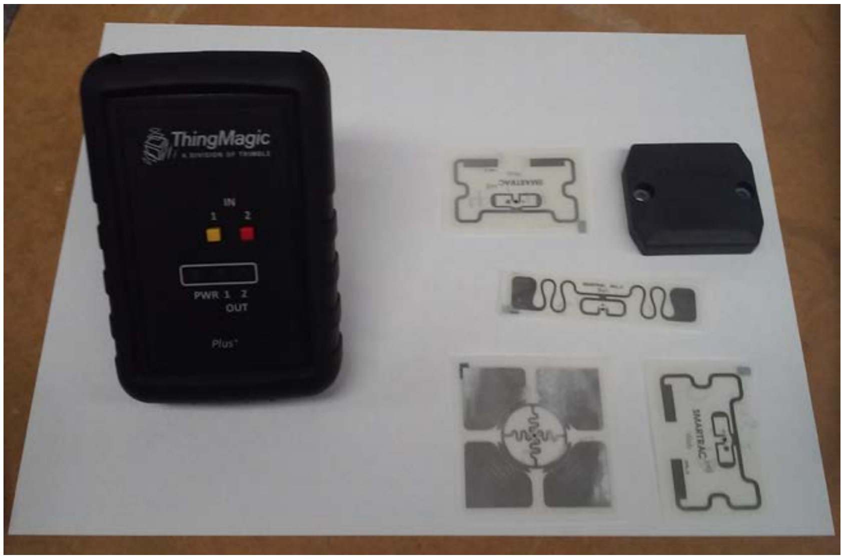
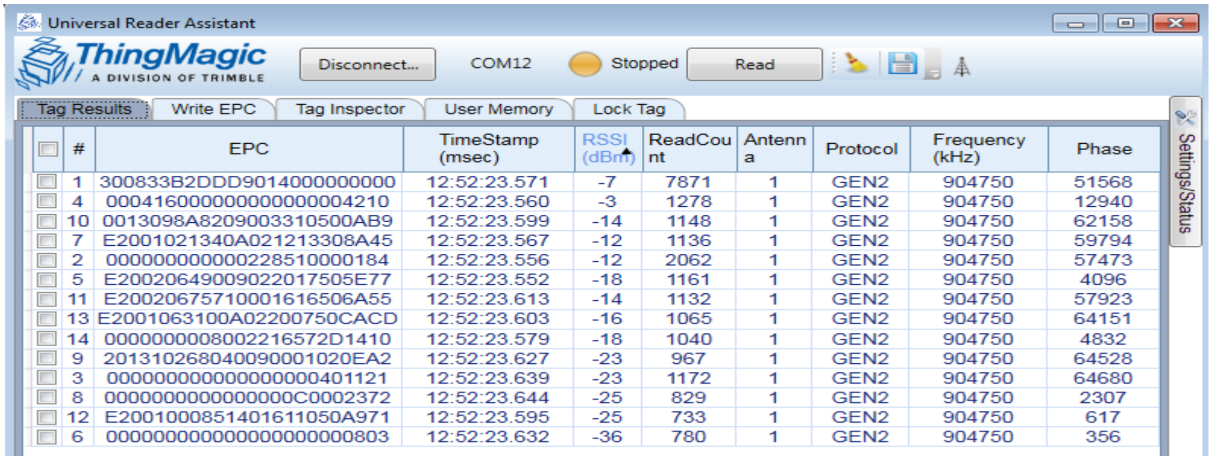


Fig. 3.1 RFID Reader and an assortment of RFID tags which were used

3.1.1 Software Interface

Through the (URA) Universal Reader Assistance software user interface, we are able to analyse all of the tag reads. Since this software is open source, we will be able to make adjustments to the data that is read. The ThingMagic's reader assistant will also allow for tag editing which will make it easier for the group to add location data to each of these tags, data that will then be plotted. Figure 6 below shows the reader software assistant.



☐	#	EPC	TimeStamp (msec)	RSSI (dBm)	ReadCount	Antenna	Protocol	Frequency (kHz)	Phase
☐	1	300833B2DDD9014000000000	12:52:23.571	-7	7871	1	GEN2	904750	51568
☐	4	000416000000000000004210	12:52:23.560	-3	1278	1	GEN2	904750	12940
☐	10	0013098A8209003310500AB9	12:52:23.599	-14	1148	1	GEN2	904750	62158
☐	7	E2001021340A021213308A45	12:52:23.567	-12	1136	1	GEN2	904750	59794
☐	2	000000000000228510000184	12:52:23.556	-12	2062	1	GEN2	904750	57473
☐	5	E20020649009022017505E77	12:52:23.552	-18	1161	1	GEN2	904750	4096
☐	11	E20020675710001616506A55	12:52:23.613	-14	1132	1	GEN2	904750	57923
☐	13	E2001063100A02200750CACD	12:52:23.603	-16	1065	1	GEN2	904750	64151
☐	14	000000008002216572D1410	12:52:23.579	-18	1040	1	GEN2	904750	4832
☐	9	20131026804009000102OEA2	12:52:23.627	-23	967	1	GEN2	904750	64528
☐	3	000000000000000000401121	12:52:23.639	-23	1172	1	GEN2	904750	64680
☐	8	000000000000000C0002372	12:52:23.644	-25	829	1	GEN2	904750	2307
☐	12	E2001000851401611050A971	12:52:23.595	-25	733	1	GEN2	904750	617
☐	6	000000000000000000000803	12:52:23.632	-36	780	1	GEN2	904750	356

Fig. 3.2 Universal Reader assistant Displaying Tags.

3.1.2 RFID tag writing.

All tags that came with the purchased kit are identified by an EPC number. The first step was to re-assign a value to these EPC identification numbers. The group then used the URA to edit the tags, the last four digits of each EPC was changed to “XX , YY” location, and the remaining digits to zero. A tag that is placed at the location (meters) X= 12 & Y=6, would have an equivalent EPC value of, “000000000000000000001206” . Initially the team used 6 RFID tags and placed these between the WiFi access points for calibration purposes.

The following 6 tags were assigned X, Y coordinates, with respect to the WPI 3rd floor blue prints:

TAG # and TAG ID

Tag 1 to 6 “ 0004” “0013” “0417” “0817” “1517” “1817”

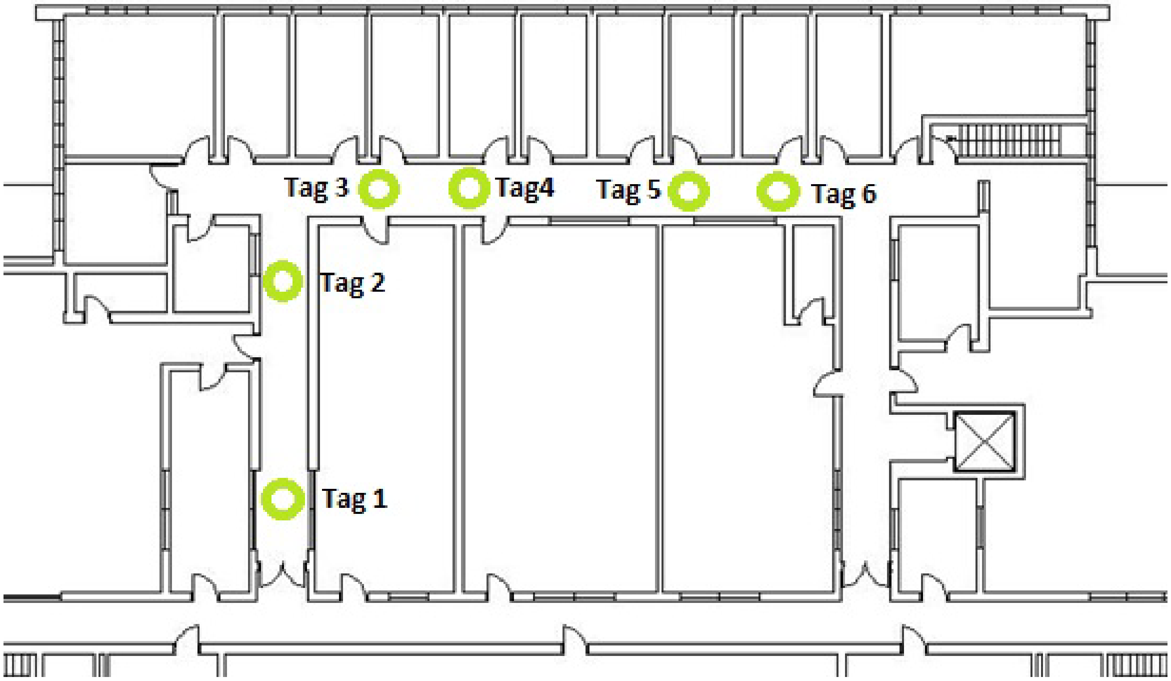
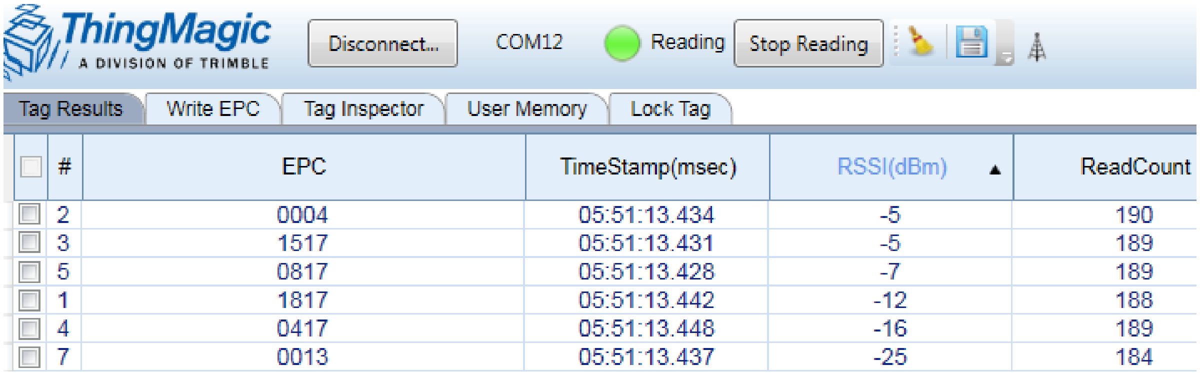


Fig. 3.3 – Third Floor of Atwater Kent Laboratory with the RFID tag locations



□	#	EPC	TimeStamp(msec)	RSSI(dBm) ▲	ReadCount
□	2	0004	05:51:13.434	-5	190
□	3	1517	05:51:13.431	-5	189
□	5	0817	05:51:13.428	-7	189
□	1	1817	05:51:13.442	-12	188
□	4	0417	05:51:13.448	-16	189
□	7	0013	05:51:13.437	-25	184

Fig. 3.4 – RFID and URA tag EPC location assignation.

The figure above shows the new EPC tag value of a total number of 6 tags. Each tag is representative of a location in the X and Y axis, the first two digits represent an X coordinate, and the last two digits represent a Y coordinate.

3.1.3 Real Time Implementation

To evaluate a real time implementation, it is first important to understand the development tools that are currently available. The Universal Reader assistant is written using the high level MercuryAPI in C# .NET. All the source code for the URA is available as part of the MercuryAPI SDK, which is a platform that allows for software development for all ThingMagic products. The MercuryAPI supports Java, .NET and C programming environments.

The first option is utilizing the URA through its development tool “MercuryAPI”, would allow us to implement all code in real time, now that any type of filter has an equivalent C interpretation of its MATLAB form. Once the filter has been implemented into C or (C# , .NET) , a grid can be created, where plotting of data is displayed.

Simple Code for creating a 20/20 grid (C Language)

```
const int cellsize = 20; // 20 pixels wide/high cells.
void grid(int gridsize)
{
int i;
for(i = 0; i < size; i++)
{
DrawLine(0, i * cellsize, size * cellsize, i * cellsize);
DrawLine(i * cellsize, 0, i * cellsize, size * cellsize}
```

The code above can be set to a scale representative of the dimensions in Atwater Kent 3rd floor. As soon as the above process is completed, we can proceed to the addition of an algorithm, that when developed, will readjust and recalibrate depending on the locations of the RFID tags, where the current (X,Y) Value will be set equivalent to tag location.

Option two is that the URA provides demo functionality to stream data, as a server, to a network port where the client applications can listen and receive tag read data.

When enabled, tag read data will be sent to the specified network port in a tab-delimited format containing any specified information " [EPC ID] [Time Stamp] [RSS] etc.] an example of this is the following:

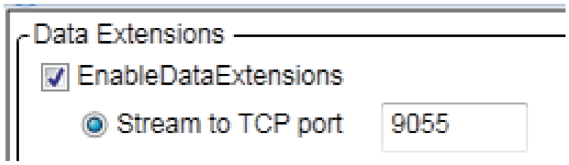
```
0001  2/14/2015 1:55:03 PM -22
0002  2/14/2014 1:55:03 PM -31
0003  2/14/2014 1:55:04 PM -22
```

Any client that can connect to the specified network port can connect and receive the data

It is very important to note 3 things. The current functionality is limited to a single client, currently it does not support a MATLAB extension**,** although, connection must be made in order to stream the data.

3.1.4 Testing Data Extensions & PuTTy for Real Time Implementation.

By clicking “Enable Data Extensions” in the URA, the software is ready to transmit to a specified port



Data Extensions

☑ EnableDataExtensions

● Stream to TCP port 9055

Fig. 3.5-Data extension

A test client can be used “telnet” for testing purposes. This can be done by entering the command

```
$ telnet [IP address of URA host] 9055
```

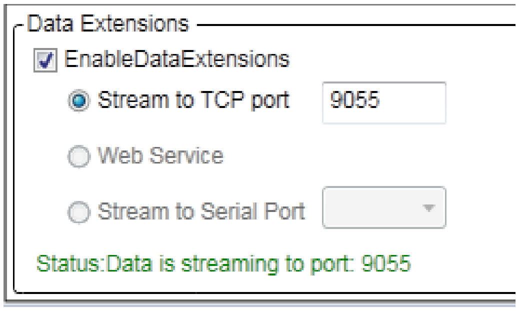
On windows, a telnet client application such as “putty” or “teraterm” can be used.

Just by assigning :

host: [URA host IP address]

port: 9055

With a client connected, read output is now pending. Once we start reading tags, these should be displayed at the other end of the connection displaying the signal below.



Data Extensions

☑ EnableDataExtensions

● Stream to TCP port 9055

○ Web Service

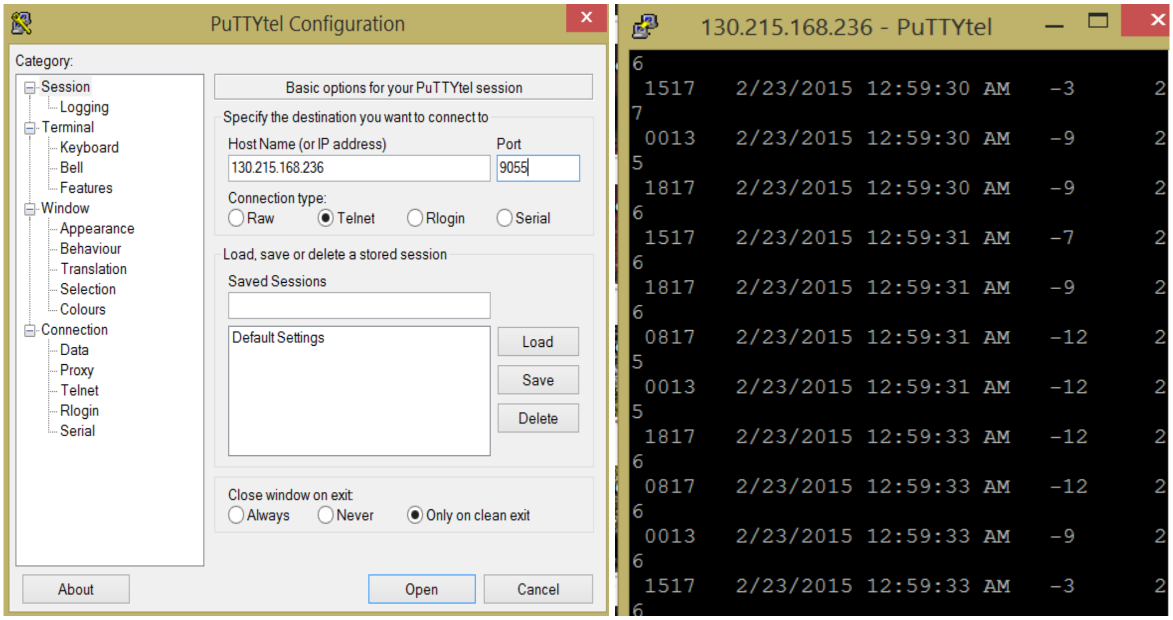
○ Stream to Serial Port

Status:Data is streaming to port: 9055

Fig. 3.6- Enable Data extension

The Method specified above can be used to transmit data to a host computer in charge of WiFi localization. Where both of these methods can be implemented simultaneously. For Offline mode, all the data available for each independent tag can be exported in EXCEL format and combined with WIFI through MATLAB.

A test was performed using PuTTytel configuration in order to look at the TCP connection, this software connects to the HOST computers " IP address" and "port number" which we assigned a value of 9055 and displays all the data being streamed.



Specify the destination you want to connect to

Host Name (or IP address): 130.215.168.236

Port: 9055

Connection type: ○ Raw ● Telnet ○ Rlogin ○ Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

```
6
 1517   2/23/2015 12:59:30 AM   -3      2
7
 0013   2/23/2015 12:59:30 AM   -9      2
5
 1817   2/23/2015 12:59:30 AM   -9      2
6
 1517   2/23/2015 12:59:31 AM   -7      2
6
 1817   2/23/2015 12:59:31 AM   -9      2
6
 0817   2/23/2015 12:59:31 AM   -12     2
5
 0013   2/23/2015 12:59:31 AM   -12     2
5
 1817   2/23/2015 12:59:33 AM   -12     2
6
 0817   2/23/2015 12:59:33 AM   -12     2
6
 0013   2/23/2015 12:59:33 AM   -9      2
6
 1517   2/23/2015 12:59:33 AM   -3      2
6
```

Fig 3.7 The figure above shows a correct connection made between URA and PuTTytel

3.1.5 Implementing the USB Plus+RFID Reader

The initial procedures involved setting up the RFID reader and software interface. After installing the driver for the USB reader and the Universal Reader Assistant software, we could begin taking measurements. The group initially studied the URA, we were able to set up the software to display EPC (Tag ID), time of arrival, RSS (Received Signal Strength), frequency and phase. This is all very valuable information that will come in handy as the project progresses.

The most important feature of the URA is the ability to edit RFID tags, by using this feature we can not only change the identification number of the tag, but also add data to them. The next step was to develop a model for the antenna, this step is extremely important now that it will determine the orientation of the reader when reading RFID tags. The model for the antenna was developed by taking measurements at different distances and angles between the reader and the tags. Two equations for the Antenna model were used. The first equation used an “b” unknown antenna loss Eq.1, and the second equation used an estimated antenna loss for our reader of -31.7dB Eq.2.

3.1.6 RFID Antenna Modeling

Two equations for the Antenna model were used. The first equation used an “b” unknown antenna loss Eq.1, and the second equation used an estimated antenna loss for our reader of -31.7dB Eq.2.

$b - 10 * \text{Alpha}*\text{Log}_{10}(\text{Distance})$ **Eq 3.1**a Path loss model unknown b

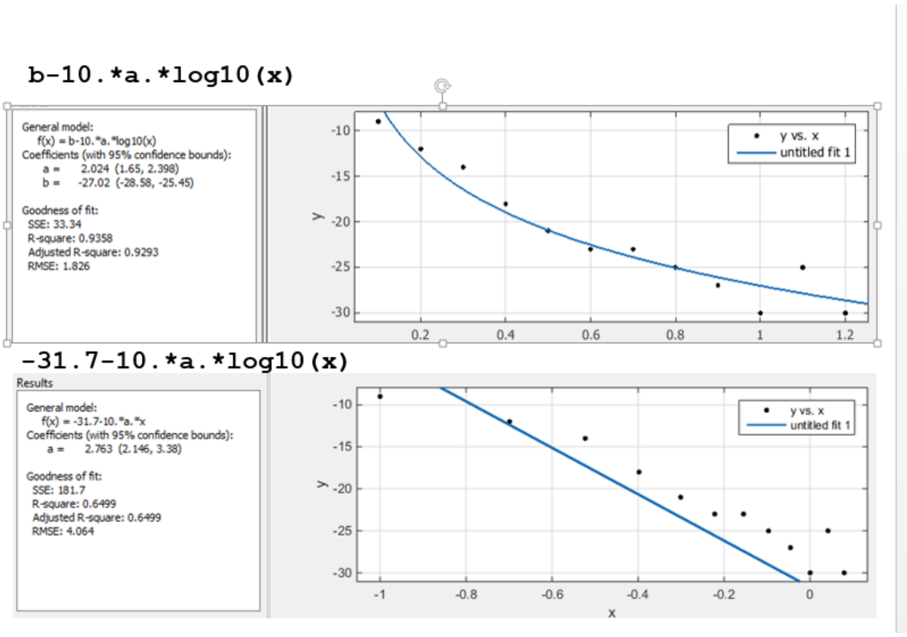
$-31.7 - 10 * \text{Alpha}*\text{Log}_{10}(\text{Distance})$ **Eq 3.1**b Path loss model calculated b

For the Antenna modeling, we initially placed the RFID tag right in front of our reader. From there we took measurements, then did two rotations, first to 45 degrees, then 90 degrees. The goal of this was to understand the behavior of the reader and its antenna in the presence of an RFID tag

LOS (based on longest measurable distance)

LOS (RFID tag placed directly in front of reader)

Distance (cm)	dBm (loss)
0	0
10	9
20	12
30	14
40	18
50	21
60	23
70	23
80	25
90	27
100	30
110	25
120	30



General model:
f(x) = b-10.*a.*log10(x)
Coefficients (with 95% confidence bounds):
a = 2.024 (1.65, 2.398)
b = -27.02 (-28.58, -25.45)

Goodness of fit:
SSE: 33.34
R-square: 0.9358
Adjusted R-square: 0.9293
RMSE: 1.826

Results

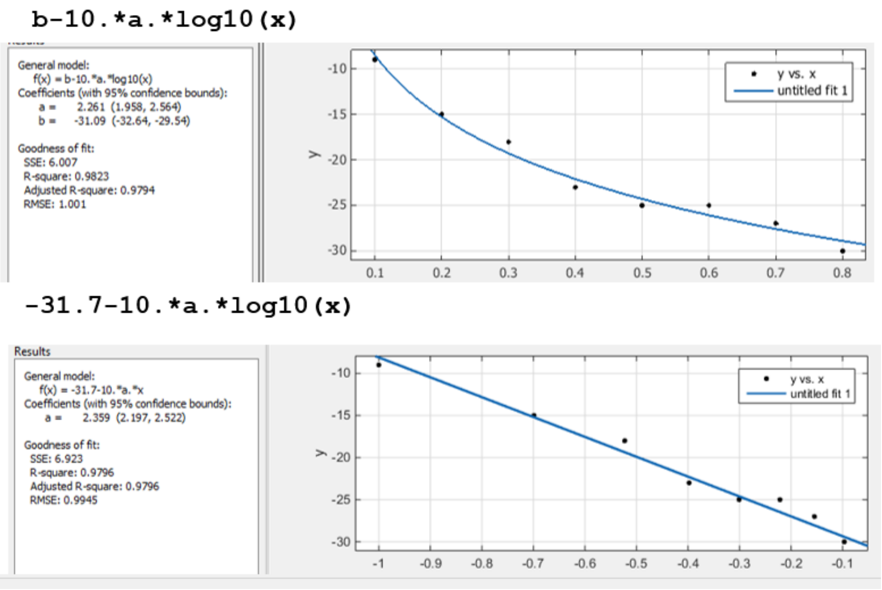
General model:
f(x) = -31.7-10.*a.*x
Coefficients (with 95% confidence bounds):
a = 2.763 (2.146, 3.38)

Goodness of fit:
SSE: 181.7
R-square: 0.6499
Adjusted R-square: 0.6499
RMSE: 4.064

Fig. 3.8 Table for measurements and Graphs for Curve Fitting at LOS

+45° Angle from LOS

Distance (cm)	dBm (loss) +45
0	0
10	9
20	15
30	18
40	23
50	25
60	25
70	27
80	30



General model:
f(x) = b-10.*a.*log10(x)
Coefficients (with 95% confidence bounds):
a = 2.261 (1.958, 2.564)
b = -31.09 (-32.64, -29.54)

Goodness of fit:
SSE: 6.007
R-square: 0.9823
Adjusted R-square: 0.9794
RMSE: 1.001

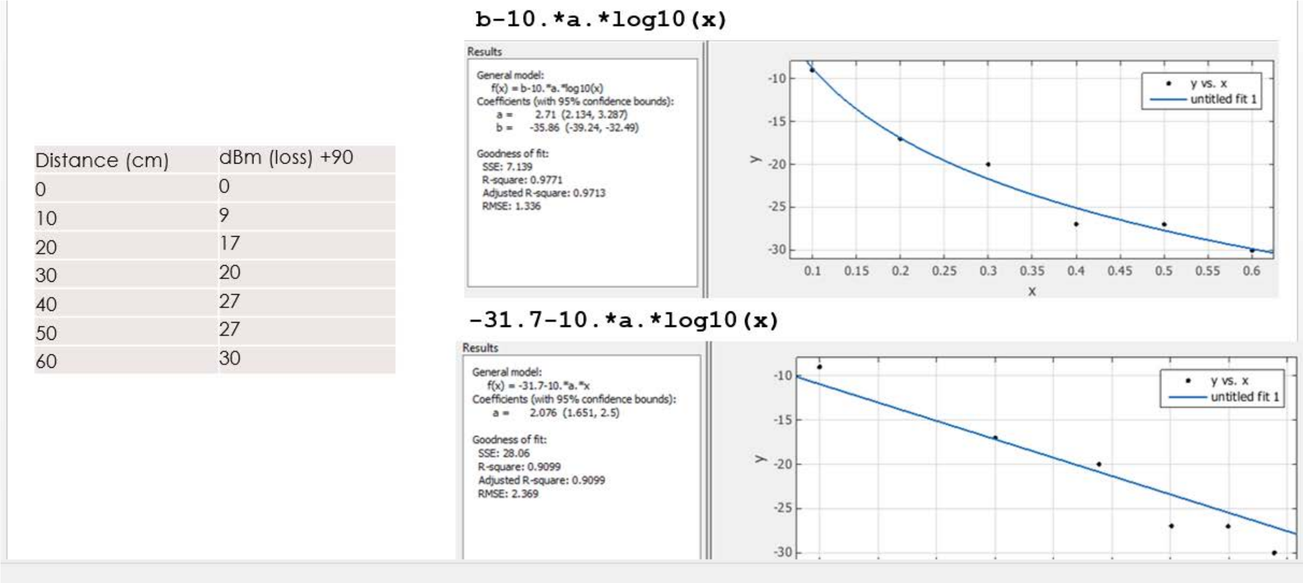
Results

General model:
f(x) = -31.7-10.*a.*x
Coefficients (with 95% confidence bounds):
a = 2.359 (2.197, 2.522)

Goodness of fit:
SSE: 6.923
R-square: 0.9796
Adjusted R-square: 0.9796
RMSE: 0.9945

Fig. 3.9 Table for measurements and Graphs for Curve Fitting at LOS **+45°**

+90º Angle from LOS



Distance (cm)	dBm (loss) +90
0	0
10	9
20	17
30	20
40	27
50	27
60	30

b-10.*a.*log10(x)

```
Results
General model:
    f(x) = b-10.*a.*log10(x)
Coefficients (with 95% confidence bounds):
      a =       2.71  (2.134, 3.287)
      b =     -35.86  (-39.24, -32.49)

Goodness of fit:
  SSE: 7.139
  R-square: 0.9771
  Adjusted R-square: 0.9713
  RMSE: 1.336
```

-31.7-10.*a.*log10(x)

```
Results
General model:
    f(x) = -31.7-10.*a.*x
Coefficients (with 95% confidence bounds):
      a =       2.076  (1.651, 2.5)

Goodness of fit:
  SSE: 28.06
  R-square: 0.9099
  Adjusted R-square: 0.9099
  RMSE: 2.369
```

Fig. 3.10 Table for measurements and Graphs for Curve Fitting at LOS **+90º**

Based on the measurements taken above, the LOS for the antenna was found to be the set of measurements that reached the longest distance away from the antenna for which we still had a tag read. The calculated (b) value from the measurements above and Eq.1 had an average value of -28 dBm which was close to the calculated DB loss of -31.7 in Eq 3.1b. Now that we have found that we are working with a directional antenna, we know in which way to position the reader in order to achieve a maximum reading range and accuracy.

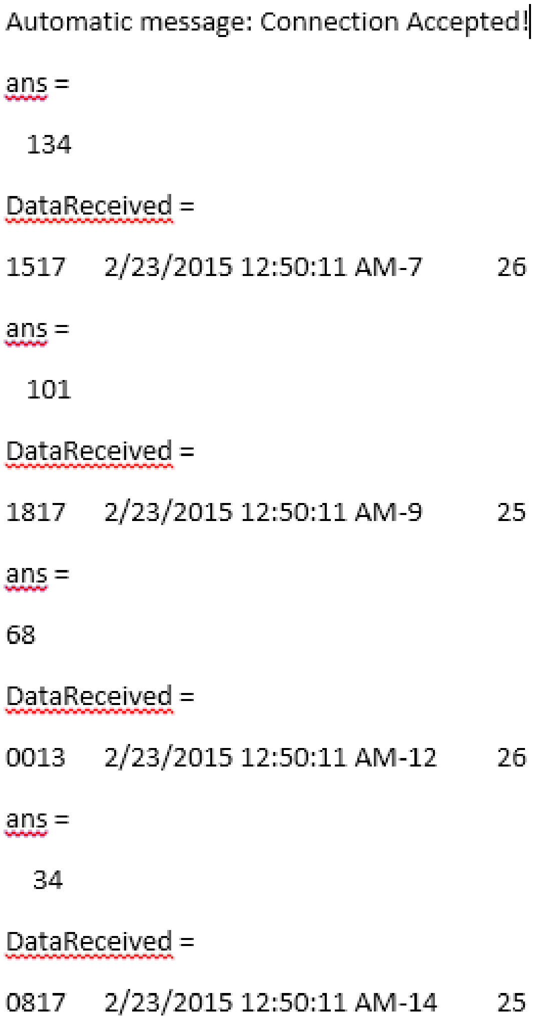
3.1.7 MATLAB RFID Implementation

The following MATLAB code allowed for a connection to be established between the Universal reader assistant, and MATLAB.

MATLAB TCP/IP CONNECTION

```
% Create TCP/IP object 't'. Specify server machine and port number.

t = tcpip('130.215.168.236', 9055);


% Set size of receiving buffer, if needed.

set(t, 'InputBufferSize', 30000);

% Open connection to the server.

fopen(t);

% Transmit data to the server (or a request for data from the server).

fprintf(t, 'GET /');

% Pause for the communication delay, if needed.

pause(1)

% Receive lines of data from server

while (get(t, 'BytesAvailable') > 0)

t.BytesAvailable

DataReceived = fscanf(t)

end

% Disconnect and clean up the server connection.

fclose(t);

delete(t);

clear t
```

OUTPUT



```
Automatic message: Connection Accepted!

ans =

  134

DataReceived =

1517    2/23/2015 12:50:11 AM-7        26

ans =

  101

DataReceived =

1817    2/23/2015 12:50:11 AM-9        25

ans =

68

DataReceived =

0013    2/23/2015 12:50:11 AM-12       26

ans =

   34

DataReceived =

0817    2/23/2015 12:50:11 AM-14       25
```

By implementing the code above, as a result, we received the following output in MATLAB, which shows that a data connection has been established and RFID tag data is ready to be processed.

3.3 Implementation and Performance Evaluation of Wi-Fi localization

This part is about algorithms implementation and data collecting. In the beginning, we select reference points for our Wifi localization. As the graph shown below, we pick eight reference points, four vertex points and four middle points, as suggested by. The Figure 18 shows the location of each reference point in red circle. The dashed rectangular is our desired trajectory in the third floor of Atwater Kent Laboratory where all experiment was conducted. The Figure 19 is the plot of 86 coordination points of desired trajectory with the unit meter.

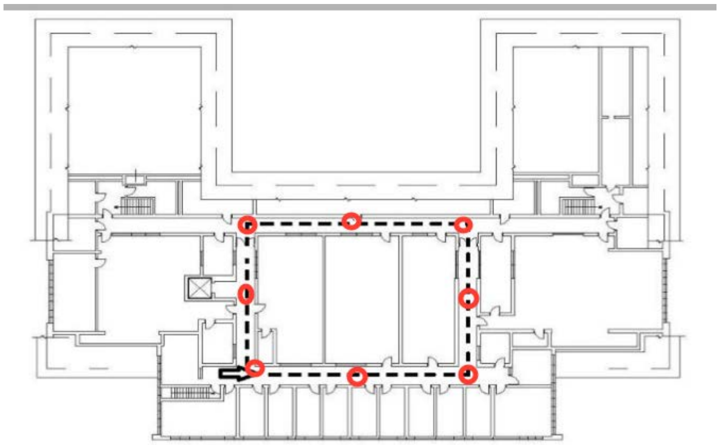


Fig. 3.11 Third Floor of Atwater Kent Laboratory with locations of reference points.

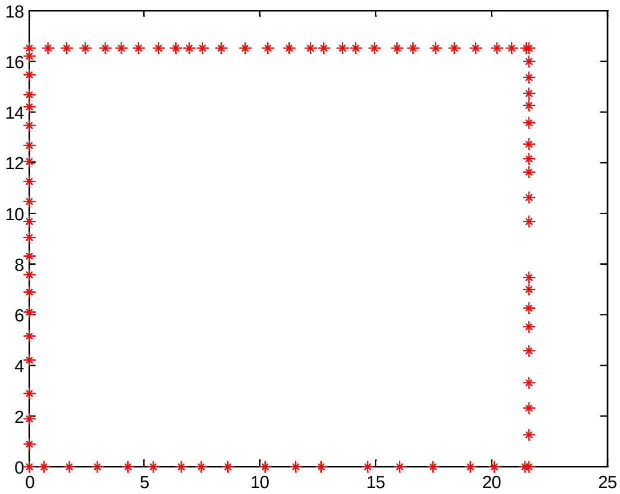


Fig. 3.12 86 coordination points of desired trajectory

Next step is to collect the Wifi data of each reference point, which are MAC and RSS at each reference point. The device we use for collecting data is a mobile phone with specific software, which developed by Guanxiong Liu, CWINS graduate student researcher. The program interface is as shown in Figure 3.

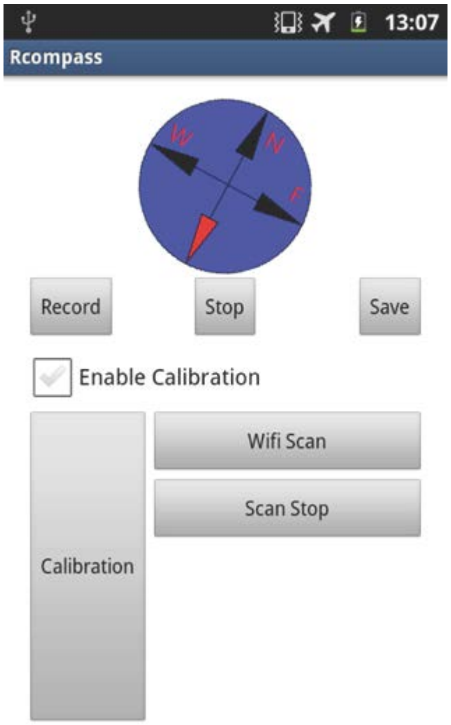
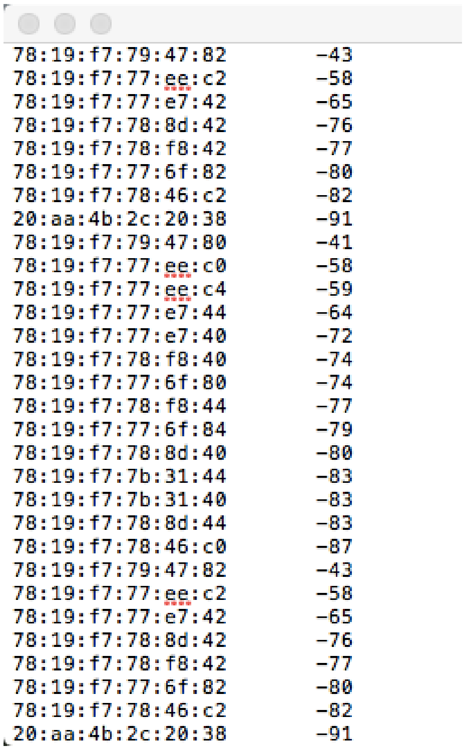


Fig 3.13a User Interface of Wi-Fi Compass. Program used to collect Wi-Fi data



```
78:19:f7:79:47:82        -43
78:19:f7:77:ee:c2        -58
78:19:f7:77:e7:42        -65
78:19:f7:78:8d:42        -76
78:19:f7:78:f8:42        -77
78:19:f7:77:6f:82        -80
78:19:f7:78:46:c2        -82
20:aa:4b:2c:20:38        -91
78:19:f7:79:47:80        -41
78:19:f7:77:ee:c0        -58
78:19:f7:77:ee:c4        -59
78:19:f7:77:e7:44        -64
78:19:f7:77:e7:40        -72
78:19:f7:78:f8:40        -74
78:19:f7:77:6f:80        -74
78:19:f7:78:f8:44        -77
78:19:f7:77:6f:84        -79
78:19:f7:78:8d:40        -80
78:19:f7:7b:31:44        -83
78:19:f7:7b:31:40        -83
78:19:f7:78:8d:44        -83
78:19:f7:78:46:c0        -87
78:19:f7:79:47:82        -43
78:19:f7:77:ee:c2        -58
78:19:f7:77:e7:42        -65
78:19:f7:78:8d:42        -76
78:19:f7:78:f8:42        -77
78:19:f7:77:6f:82        -80
78:19:f7:78:46:c2        -82
20:aa:4b:2c:20:38        -91
```

Fig 3.13b Sample of data collected by Wi-Fi Compass. The left column is the MAC Address of the access point and the right column is the RSS(dB) of the access point

In order to collect the Wifi data, we step on each reference point for one minute, and started the program with pressing “Wifi scan”. Then we pressed the “scan stop” to end the data collecting. The program will automatically generate a text file of the Wifi data, which it collected in one minute. The data contains MAC of Wifi hotpots and RSS from each MAC, which are the mobile phone received in one minute. A sample of collected data is as shown below, the first column is MAC, and the second column is RSS of corresponding MAC.

3.3.1 Algorithm Testing

In this part, we started to test our algorithms. Initially, we randomly stand on four points as our test points, as the result shows below, the point we stand on in the graph is the red circle, and we collect the Wifi data of these points. After the data collecting, we plug in our reference points data and each test points data. The algorithm will estimate the coordination of each test point. The estimate coordination of each test point is “x” in the graph. The green dots are coordination points.

For K-NN algorithm, when K=3, the overall performance of the KNN has error of 2 to 5 meters.

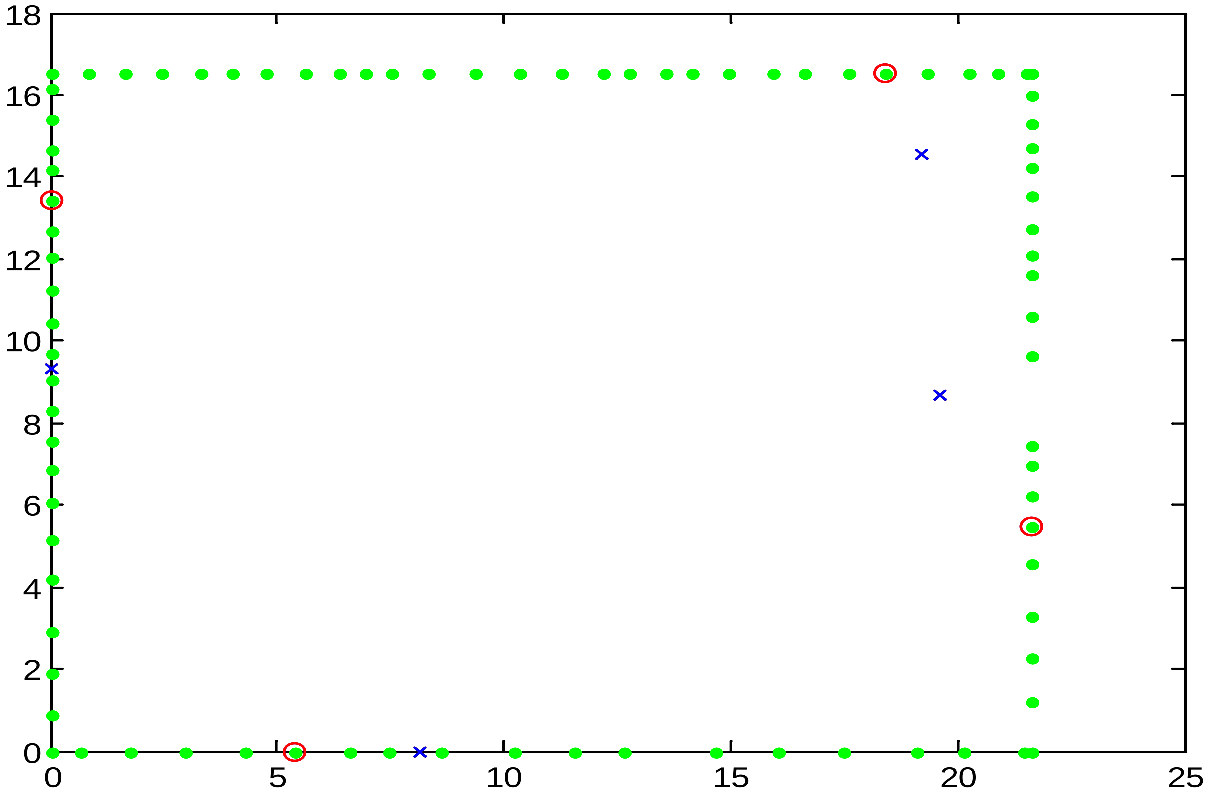


Fig 3.14 – Estimated results of K-NN algorithm

Table 3.1 Table of the calculated errors for K-NN

Stand Point (x,y)	Estimate Point (x,y)	Error (Meter)
5.397,0	8.133,0	2.74
21.6,5.49	19.59,8.68	3.765
18.42,16.51	19.22,14.57	2.165
0,13.45	0,9.30	4.15

For Kernel algorithm, we recollected the data, because of Kernel algotithm, we need to turn around when we collecting the Wifi data. In our kernel algorithm with sigma = 20, the overall performance of the Kernel has error of 0.53 to 8 meters.

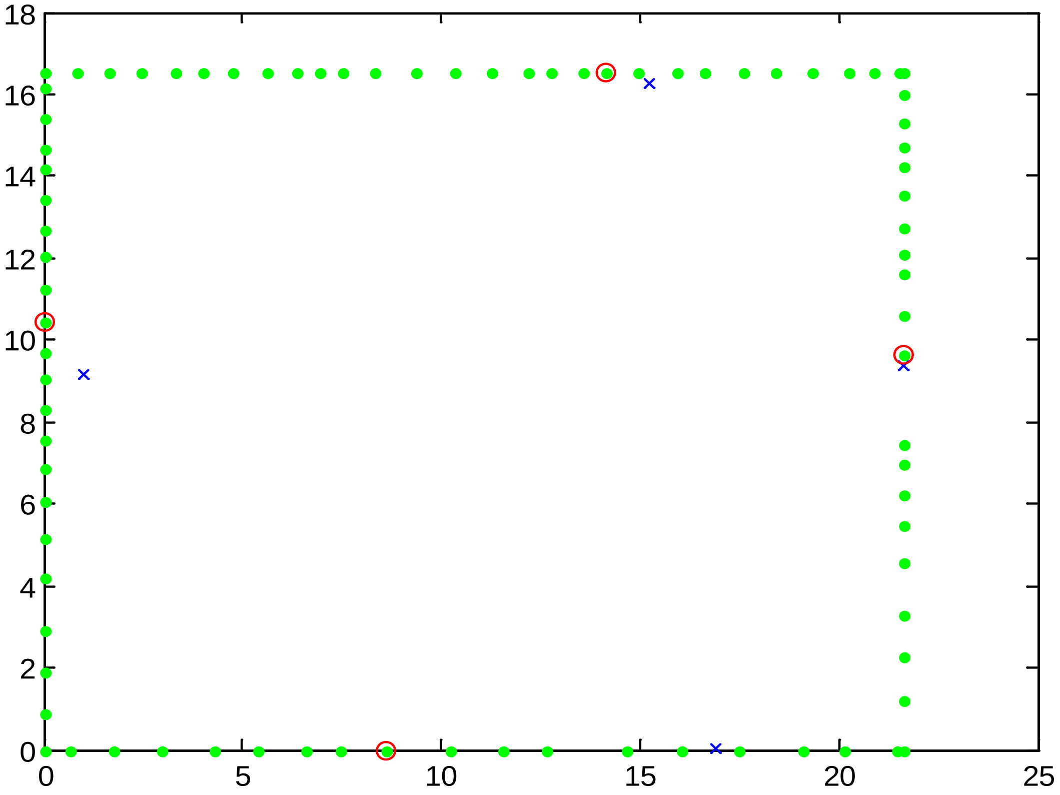


Fig 3.15 – Estimated results of Kernel algorithm

Table 3.2. Table of the calculated errors for Kernel

Stand Point (x,y)	Estimate Point (x,y)	Error (Meter)
8.64,0	16.92,0	8.28
21.61,9.66	21.6,9.37	0.29
14.15,16.51	15.2,16.3	1.02
0,10.43	1.0233,9.159	1.73

Based on the previous test, K-NN algorithm has the error about 2-3 meters, and Kernel algorithm possesses an error of approximately 1 meter. However, Kernel has an outlier with an error of 8 meters. In order to make an accurate evaluation of each algorithm, we take 16 more test points, and create a CDF (Curriculum Distribution Function) graph to evaluate error of these two algorithms. The first part, as shown below in Fig 24 and Fig 25, is the result of K-NN algorithm and CDF graph of the error of K-NN algorithm, where K=3

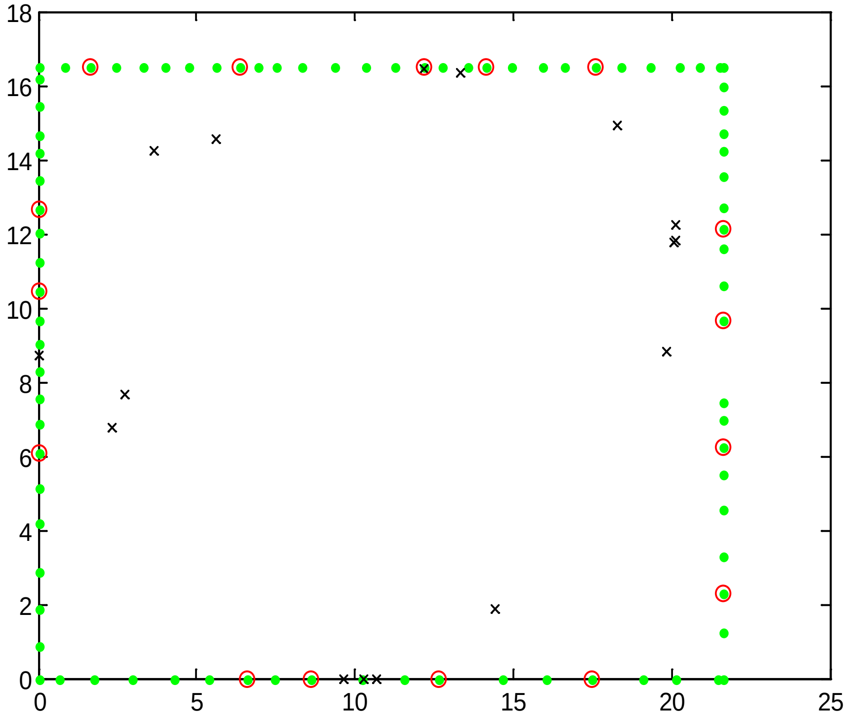


Fig. 3.16 – Estimated results of K-NN algorithm

From the CDF graph as shown below in Fig 20, with 50% probability, the error of K-NN algorithm is about 2.6 meters

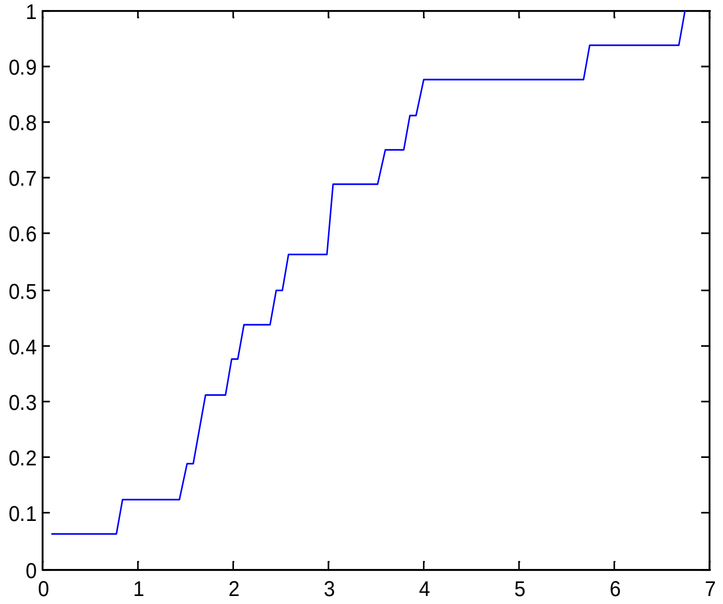


Fig. 3.17 – CDF graph of error of K-NN algorithmThe second part, as shown below in Fig 21 and Fig 22, is the result of Kernel algorithm and CDF graph of the error of K-NN algorithm, where sigma is equal to

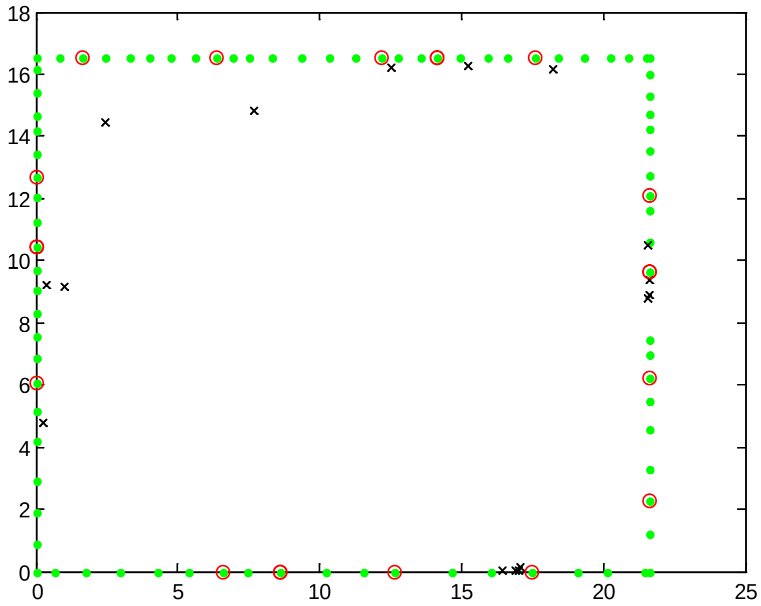


Fig. 3.18 – Estimated results of Kernel algorithm

From the CDF graph as shown below in Fig 22, with 50% probability, the error of Kernel algorithm is about 2.1 meters.

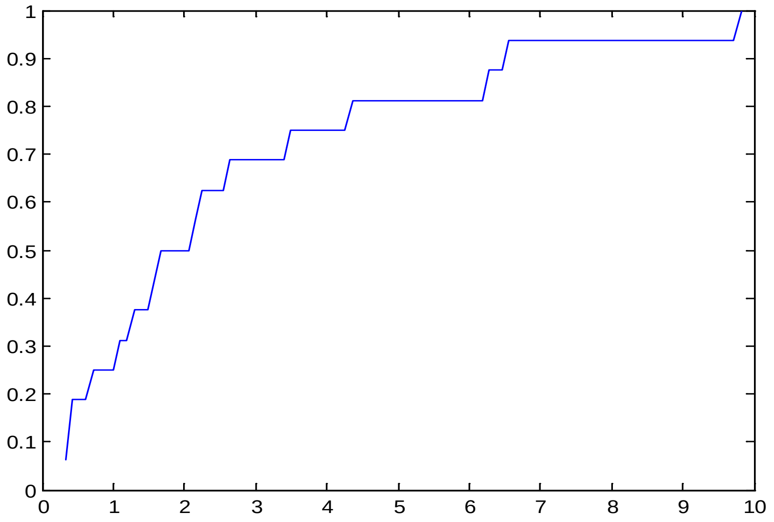


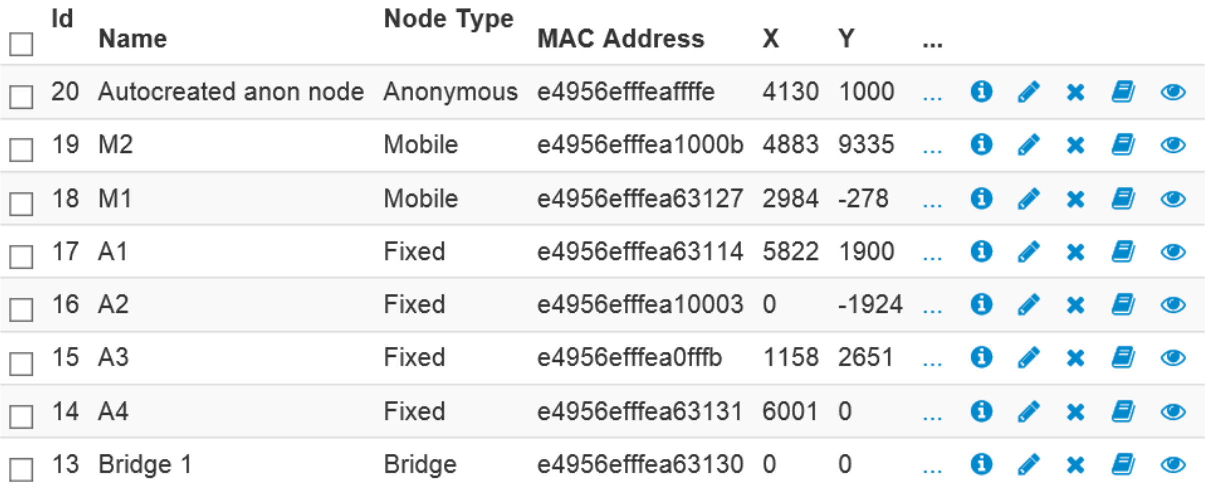
Fig. 3.19 – CDF graph of error of Kernel algorithm

From the statistical analysis, we can conclude that Kernel algorithm has better performance than K-NN algorithm, although, the difference is almost negligible. In practice, the Kernel only has an error of approximately 1 meter on most test points, but some outlier errors with more than 6 meters on the bottom side increase the average error on the overall performance of the Kernel algorithm.

3.4 UWB Performance Evaluation and Scenario Setup

This part concerns the set up and testing of the UWB system discussed in the background. In order to set up the system, we have to register the nodes through the Red Point Administrator tool. This allows us to calibrate the nodes in accordance with our needs. For the purpose of this report, we set up the system within the CWINS Laboratory

.



□	Id	Name	Node Type	MAC Address	X	Y	...					
□	20	Autocreated anon node	Anonymous	e4956efffeaffffe	4130	1000	...	ℹ	✎	✖	▤	👁
□	19	M2	Mobile	e4956efffea1000b	4883	9335	...	ℹ	✎	✖	▤	👁
□	18	M1	Mobile	e4956efffea63127	2984	-278	...	ℹ	✎	✖	▤	👁
□	17	A1	Fixed	e4956efffea63114	5822	1900	...	ℹ	✎	✖	▤	👁
□	16	A2	Fixed	e4956efffea10003	0	-1924	...	ℹ	✎	✖	▤	👁
□	15	A3	Fixed	e4956efffea0fffb	1158	2651	...	ℹ	✎	✖	▤	👁
□	14	A4	Fixed	e4956efffea63131	6001	0	...	ℹ	✎	✖	▤	👁
□	13	Bridge 1	Bridge	e4956efffea63130	0	0	...	ℹ	✎	✖	▤	👁

Fig 3.20. The list of nodes on the server.

We begin with a reference point. For us, that reference point is the Bridge node, as seen in the above figure where the bridge node has 0 value for the x, and y axis. We measured the distance of each fixed node relative to the bridge node and placed the fixed nodes in a rectangular fashion around the bridge node. We measured the distance of the nodes using the Bosch Distance Measurer (depicted in Figure), a laser device that is accurate within 1mm. Accurate placement of the fixed nodes is necessary in order to better gauge the error of the system. [11]

Fig 3.21. Bosch Distance Measurer

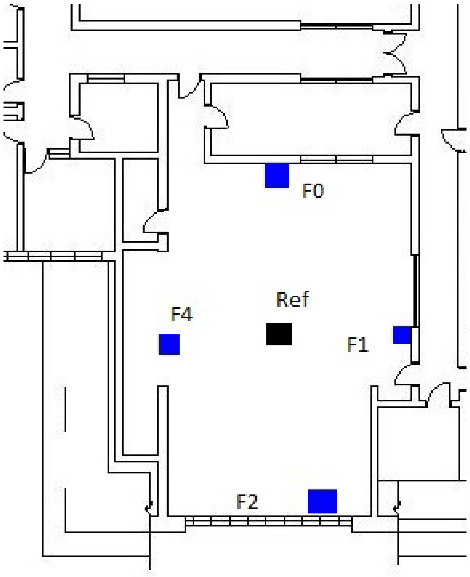
[11] UWB localization - active and passive approach

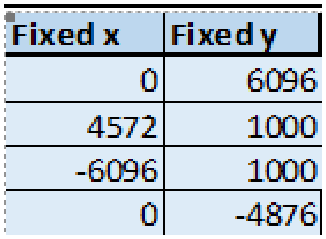
For the nodes in the figures below, a helpful key is provided to be able to identify them:

- Blue Squares – Fixed Node
- Black Square– Bridge Node

The measurements we have taken are from the mobile node.

3.4.1 LOS Measurements





Fixed x	Fixed y
0	6096
4572	1000
-6096	1000
0	-4876

Fig 3.22 Locations of the fixed nodes and bridge node for Line of sight measurement in CWINS Lab

The mobile nodes would send a “heartbeat” to the bridge node every 10 seconds. Hence, the readings above are an average location after one minute situated in one place. The points shown above are LOS. The error fell between 41mm to 437mm. These numbers are well below a meter thus, highlighting the high accuracy of Ultra Wideband.

Table 3.3 A table containing the errors for Line of Sight localization In CWINS Lab

Real x	Real y	Uwb x	Uwb y	Error
-3962	-2133	-3392	-1344	973.3555
-4876	914	-4752	930	125.028
-3890.96	-4419.6	-3552	-3680	813.5736
-2133.6	3352	-2256	3232	171.4111
-6004.56	-4632	-4208	-2976	2443.351
-5029.2	-1767	-4416	-1152	868.4695
1127.76	2712	2000	2720	872.2767
914.4	-4343.4	1344	-4576	488.5273
3505.2	-1117.7	4416	-1264	922.4751
2926.08	-4038.6	3584	-3968	661.6971
3048	1676.4	1888	1072	1308.014
-304.8	3200.4	-688	3500	486.4179
1700.784	2565.41	1296	2144	584.3257
1574.9	4673.5	1760	5728	1070.622
381	4597.4	-448	4624	72.08717

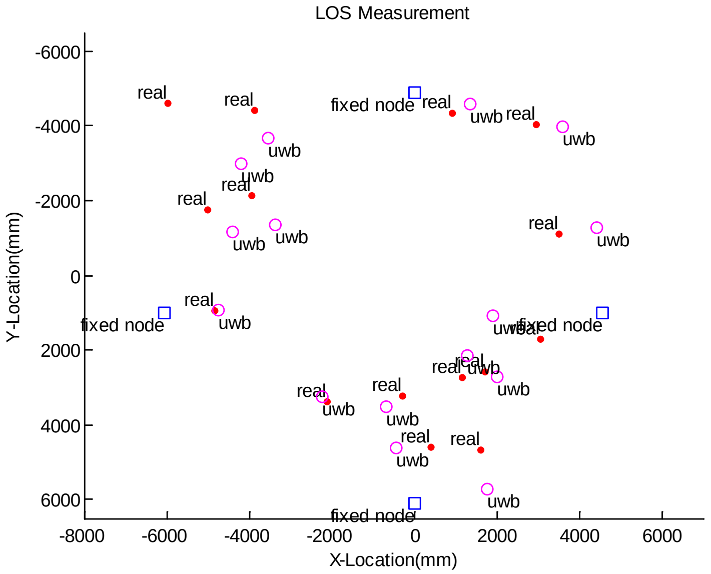


Fig 3.24 Graphical representation of the UWB measurement in CWINS Lab for Line of sight

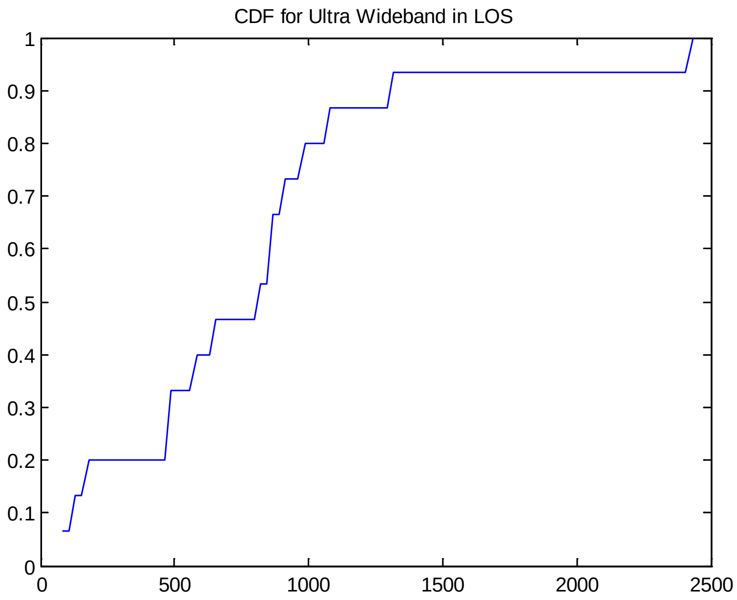
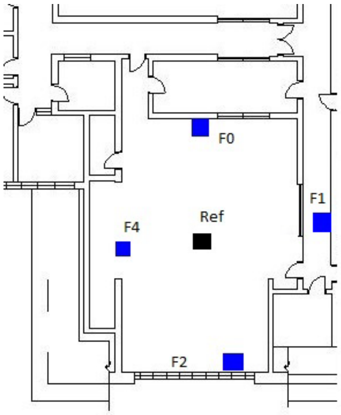
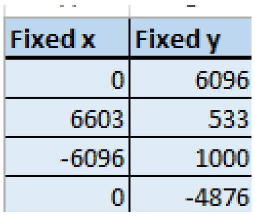


Fig 3.25 CDF for Ultra Wideband in Line of sight

The mobile nodes would send a “heartbeat” to the bridge node every 10 seconds. Hence, the readings above are an average location after one minute situated in one place. The points shown above are LOS. The error fell between 41mm to 437mm. These numbers are well below a meter thus, highlighting the high accuracy of Ultra Wideband.

3.4.2 OLOS Measurements



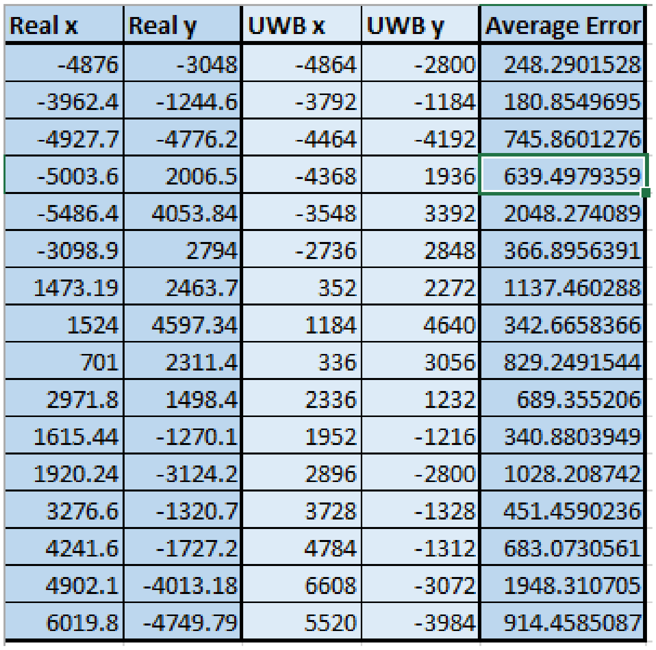


Fixed x	Fixed y
0	6096
6603	533
-6096	1000
0	-4876

Fig 3.26 Locations of the fixed nodes and bridge node for Obstructed Line of sight measurement in CWINS Lab

To test the OLOS capability of the technology, we set up the nodes around the hallway of the third floor of Atwater Kent. Theoretically, the error of Ultra Wideband in OLOS should be exponentially high, due to the technology utilizing TOA (Timing of Arrival). Error should hover around 3 to 5 meters.

Table 3.4 A table containing the errors for Obstructed Line of Sight localization In CWINS Lab



Real x	Real y	UWB x	UWB y	Average Error
-4876	-3048	-4864	-2800	248.2901528
-3962.4	-1244.6	-3792	-1184	180.8549695
-4927.7	-4776.2	-4464	-4192	745.8601276
-5003.6	2006.5	-4368	1936	639.4979359
-5486.4	4053.84	-3548	3392	2048.274089
-3098.9	2794	-2736	2848	366.8956391
1473.19	2463.7	352	2272	1137.460288
1524	4597.34	1184	4640	342.6658366
701	2311.4	336	3056	829.2491544
2971.8	1498.4	2336	1232	689.355206
1615.44	-1270.1	1952	-1216	340.8803949
1920.24	-3124.2	2896	-2800	1028.208742
3276.6	-1320.7	3728	-1328	451.4590236
4241.6	-1727.2	4784	-1312	683.0730561
4902.1	-4013.18	6608	-3072	1948.310705
6019.8	-4749.79	5520	-3984	914.4585087

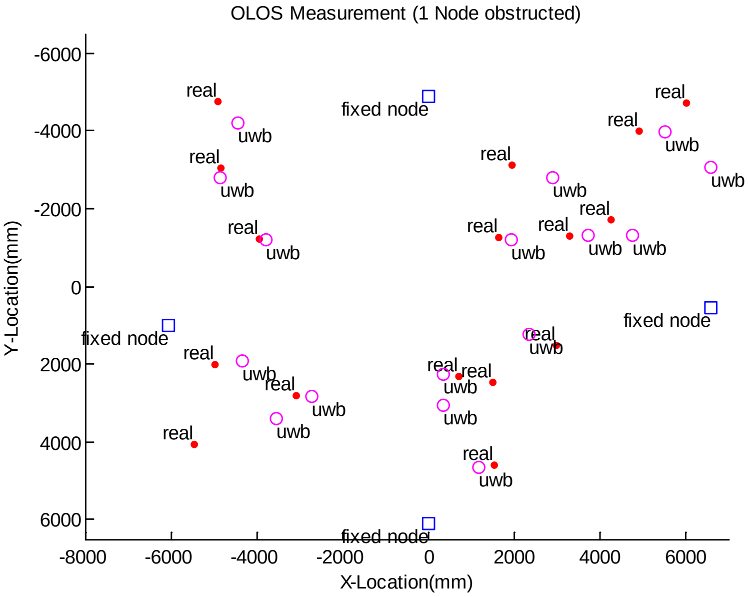


Fig 3.28 Graphical representation of the UWB measurement in CWINS Lab for Obstructed Line of sight

To test the OLOS capability of the technology, we set up the nodes around the hallway of the third floor of Atwater Kent. Theoretically, the error of Ultra Wideband in OLOS should be exponentially high, due to the technology utilizing TOA (Timing of Arrival). Error should hover around 3 to 5 meters.

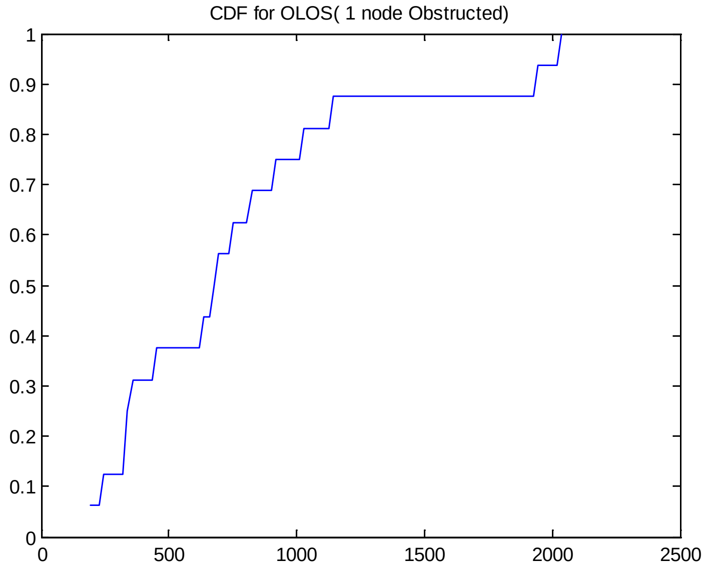


Fig 3.29 CDF for UWB measurement Obstructed Line of Sight in CWINS Labs

3.4.3 UWB MATLAB Implementation

The UWB system will work conjunction with Wi-Fi and RFID. The technology will be used predominantly within the “open space” of the CWINS laboratory. The algorithm used for Ultra Wideband would be a geometric algorithm. The algorithm will be based on distance: If the mobile node is within a certain range, the algorithm will detect it and begin to plot the data. Once the mobile node is out of range, the algorithm switches to Wi-Fi and RFID.[12]

[12] NLOS detection algorithms for Ultra-Wideband localization

Chapter 4: Hybrid Wi-Fi Localization

This chapter details our results, and the combination of the three technologies into one hybrid system. It includes the merging of Wi-FI and RFID, Kalman filter smoothing and the final testing and scenario along with the graphical representation of our data, which supports our hybrid indoor localization system.

4.1 RFID & Wi-Fi Combination Results

Since RFID tags have a maximum RSS loss of -30 dB, and WiFi can have RSS loss values less than -30 dB, we can set a hierarchy system in which the current location prioritizes and RFID reading, followed by WiFi. The algorithm for this is attached to the Appendix, and it takes into an account measured RSS, and RFID location. After setting up the RFID tag locations, a new set measurements can be taken by running the MATLAB algorithm. The new set of data can be set in a side by side comparison, for a better understanding of the results.

Then, we did a test to compare the Wi-Fi results and the results with RFID calibration. We randomly stood on 12 positions as your testing points, where deployed the RFID tags. From the testing graph as shown below, the RFID did the correction when I stood in its range.

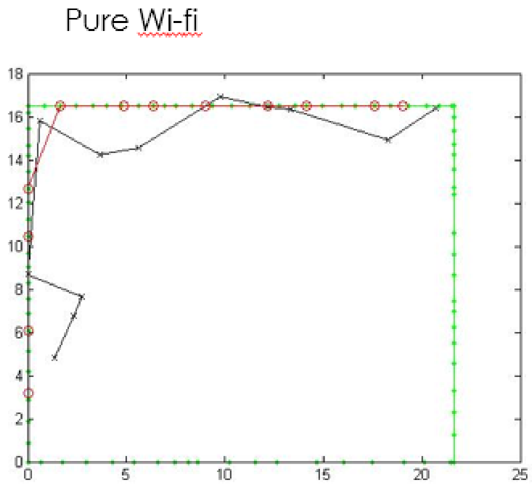


Fig 4.1a WiFi

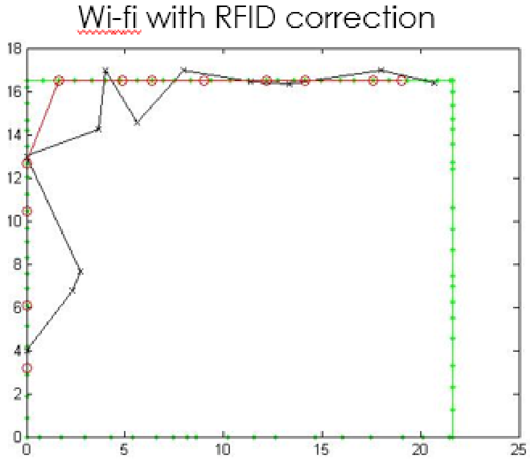


Fig 4.1b Wi-Fi with RFID Correction

The graphs above can be translated into the CDF Graphs below to visualize the improvements. From the graph, at the 50% chance, the error of Wi-Fi estimation by Kernel method is 1.8 meter. For RFID calibration result is 1.1 meter.

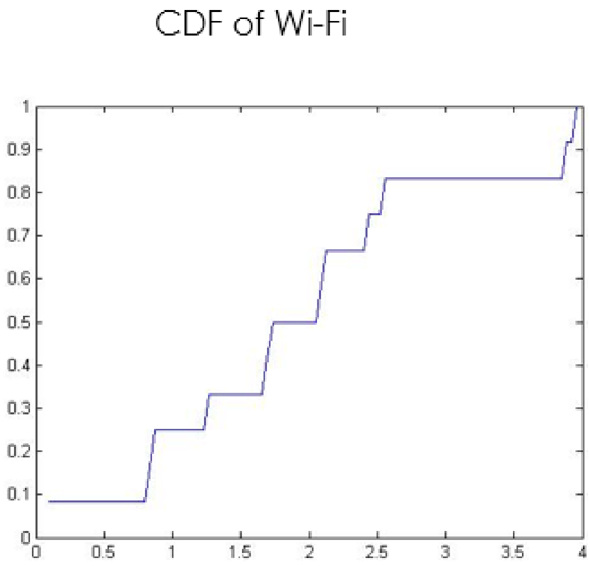


Fig 4.2a CDF graphs of result of WiFi

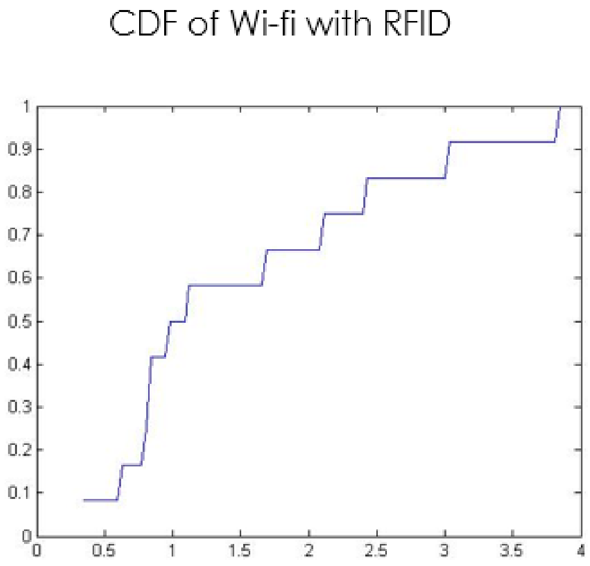
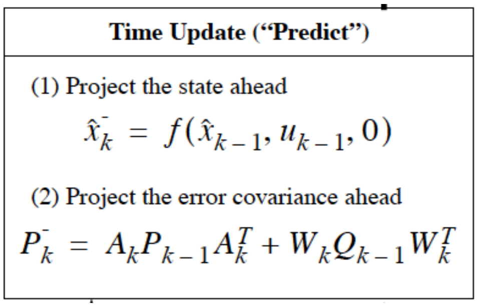


Fig 4.2b CDF of WiFi with RFID

4.2 Kalman Filter Results

To combine the Kalman filter with the project, we integrated it with the received signal strength of the Wi-Fi, a gyroscope, and the average distance covered by one of our steps. This Kalman filter will take care of providing a (non-linear) graphical representation, of our changes in steps and movement. Our goal with this filter is to implement it to utilize it in separate stages: Initialization and measurement update stage.



Time Update ("Predict")

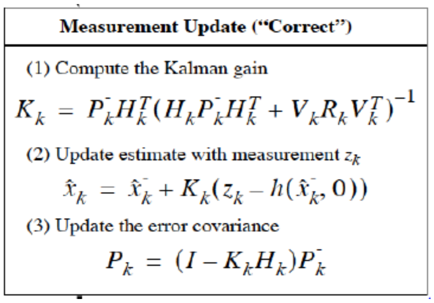
(1) Project the state ahead

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0)$$

(2) Project the error covariance ahead

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$$

Fig 4.3 -Time update, initially estimates X_{k-1} and P_{k-1}



Measurement Update ("Correct")

(1) Compute the Kalman gain

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$$

(2) Update estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0))$$

(3) Update the error covariance

$$P_k = (I - K_k H_k) P_k^-$$

Fig 4.4- Movement Update

Initialization Process

In the initial phase, the state and covariance is calculated. The information for the state, is calculated while the time is being updated, then, it is combined with the length of the step and the gyroscope, for a more accurate representation of direction and displacement.

$$B = \begin{bmatrix} step(i) & 0 \\ 0 & step(i) \end{bmatrix}$$

$$\mathrm{U} = [\sin\big(theta(i)\big)\,;\quad \cos(theta(i))]$$

Eq 4.1 The 1st State

As seen in the equations above, the 1st state carries the X and Y coordinates, and their respective angles, represented by the Sine and Cosine. It is very important to note, that as new sets of data arrive, these measurements get constantly updated as we use the wireless estimations of location and the magnetometer to setup up our states. At first we collected data across 23 steps in AK's third floor hallway, this was followed by another set of data collection across 33 steps in a hallway perpendicular to the previous, the data collected carried information from the Wi-Fi Kernel method outputs, with the x and y locations.

Then, by using the average and standard deviation of those measurements, the calculated position covariance is shown below.

$$E\_x = [\begin{matrix} sigma\_x{}^\wedge 2 & 0 \\ 0 & sigma\_y{}^\wedge 2 \end{matrix}], \qquad E\_z = [\begin{matrix} sigma\_x\_ker{}^\wedge 2 & 0 \\ 0 & sigma\_y\_ker{}^\wedge 2 \end{matrix}]$$

Eq 4.2 Average standard deviation placed into position convergence

The equation above models a path which determines where to display the Wi-Fi measurements. As the covariance results get greater and greater, each Wi-Fi measurement accuracy will increase until the covariance results are reduced down to a certain point.

In this process we update the state by just utilizing the length of our steps and the information given by the gyroscope. The two following equations are used to provide a probabilistic approach to determine a prediction of the future location.

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k, 0, 0)$$
$$\mathbf{P}_k^- = \mathbf{A}_k \mathbf{P}_{k-1} \mathbf{A}_k^T + \mathbf{B}_k \mathbf{\Gamma}_{k-1} \mathbf{B}_k^T + \mathbf{Q}_{k-1}$$

Eq 4.3 Prediction of location

$$p_{bar} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * p + [\sin(theta(i));\quad \cos(theta(i))]\text{*B}$$

Eq 4.4 Combination of x ,y and Theta

Because our goal is to combine these measurements in terms of X, Y, theta, step length and sigma of step length and sigma of theta, we can proceed with the Eq.5 to have an estimate of our future position. Also, the state function is only used in the prediction phase to predict where the object is located better.

In the Kalman Filter, the state transition matrix is the derivative of the state, which is shown in great detail above. Note, this matrix is only used during the covariance update phase. Now that all the elements in the prediction state are explained, let us move on to the measurement update phase.

Measurement Update Stage

In the Measurement Update part of the Kalman Filter, we used the following equations

$$p = p_{bar} + Kalman\ Gain * (kernel(x, y) - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * p_{bar}$$

Eq 4.5 Kalman Filter Implementation

The first part of the measurement update phase is to determine what is actually being measured. In our case, we chose to measure the Wi-Fi x and y coordinates and we will only be using these values to update the KF position only when the covariance values become too large.

$$\mathrm{E} = \left(\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} - Kalman\ Gain * \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) * E\_bar$$

Eq 4.6 Wi-Fi reading applied directly to current state

During the measurement update phase, the raw Wi-Fi readings will be directly applied to the current state only if the covariance becomes too high. If the covariance is not high, then the Wi-Fi measurements will not have much of an impact on the position coordinates. With the development of the KF equations, measurements and state complete, we have to determine the covariance values.

$$\mathrm{E\_bar} = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} * E + \begin{bmatrix}sigma\_x\hat{}2 & 0\\0 & sigma\_y\hat{}2\end{bmatrix}$$

Eq 4.7 Covariance value determination

With every time update, the covariance term, E, will be updated with the noise from the matrix, E-bar. Once the noise from the x and y position reaches a certain point, then the Wi-Fi measurement update will have a larger impact. The results of our KF algorithm will be further discussed later on.

4.3 Final Scenario

Our final scenario is as shown below in the figure that I took a walk from a corner of hallway on third floor in AK down to the end, and did the same thing in the other hallway.

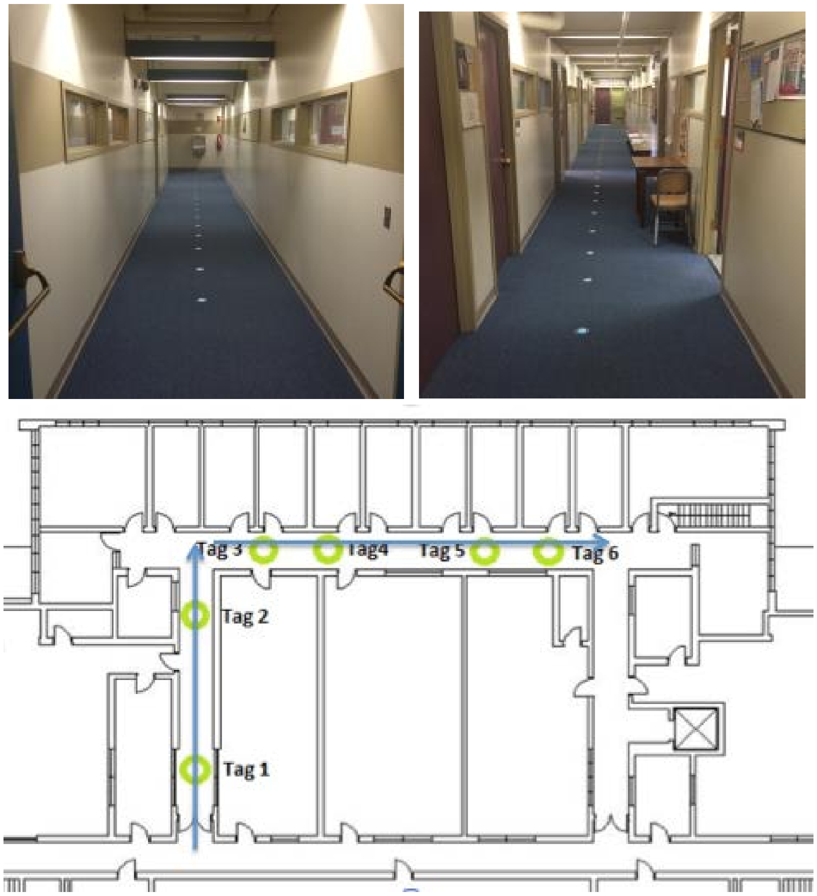


Fig 4.5 The demonstration of our final scenario in the project

To obtain better result of the system, we combine inertial navigation system and Wi-Fi result using Kalman filter get better Wi-Fi estimation. The methodology is discussed in a previous chapter. Then, we add RFID calibration to get the refined result as our final estimation.

4.3.1 The result and analysis of final scenario

As the diagram shown below, the green dots in graph is the real path in our final scenario, the red line is the result of Kernel method estimation. The left graph is result of Kalman filter, the result is as shown as black ‘x’ in the graph. The right graph show the result of Kalman filter with RFID calibration, as shown as blue line in the graph.

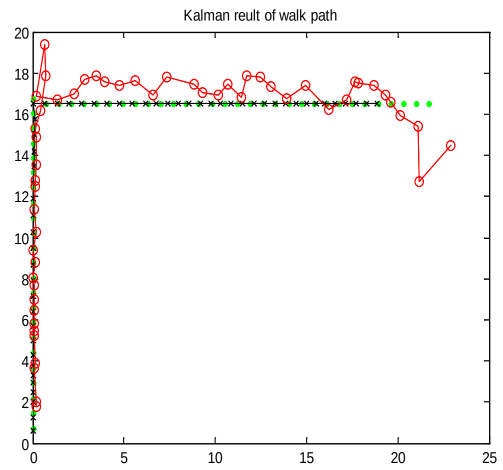


Fig 4.5a Kalman result of walk path

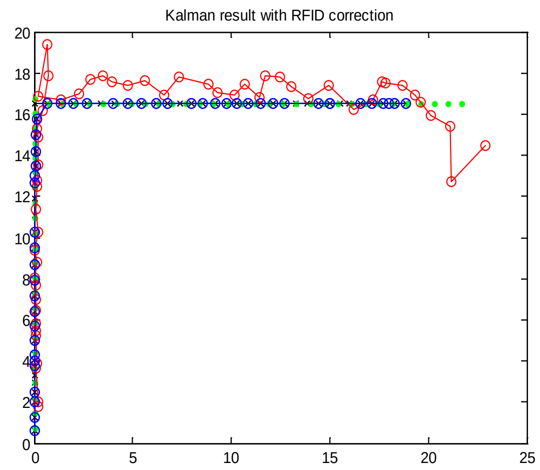


Fig 4.5b Kalman Result with RFID correction

The results of final scenario of using Kalman Filter. Left graph is the Kalman Filter result; Right graph is the result of Kalman Filter with RFID calibration. To visualize the improvement we built a CDF graph, as shown below, to compare each method.

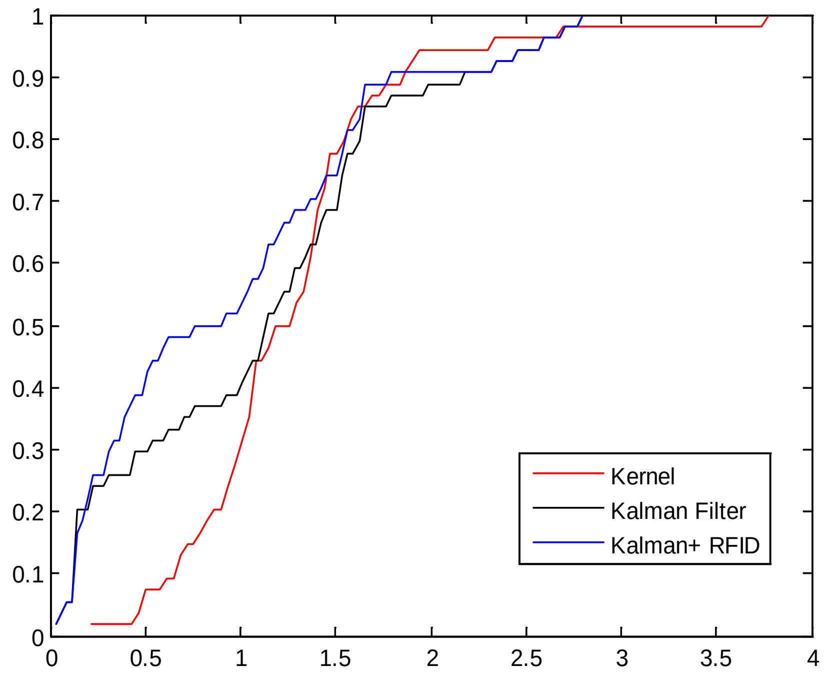


Fig 4.6: The CDF graph of the each method results

As the figure shows above, the blue line shows that the error is half meter with 50% chance, for the Kalman filter with RFID calibration. Black line shows that the error is above 1 meter 50%chance, for the result of only using Kalman filter combining with Wi-Fi localization. Red line shows that the error is slightly more than Kalman result, which is 1.3 meter for Wi-Fi localization by Kernel method.

Chapter 5: Conclusion & Future Work

5.1 Conclusion

In conclusion, we designed, implemented, and tested a hybrid localization algorithm that uses Wi-Fi, RFID, and UWB. We developed the algorithm to take advantage of an existing Wi Fi infrastructure, and incorporates RFID and UWB in order to make the system much more accurate. We were not able to find a method to fully implement Ultra Wide-Band due to its limitations in obstructed line of sight, but we were able to connect the Ultra Wide-Band system with the overall system. Also, we use inertial navigation system to record step count and directional angles to use the Kalman filter to refine the Wi-Fi results. Ultimately, we get better estimation by using the different technologies. It will also be simpler with the aid of a robotic platform (e.g. TurtleBot) in data collection. This would reduce error in collection, thus making the data more accurate.

5.2 Future Work

Overall, we accomplished most of the work we set out to do, in terms of developing a Hybrid Localization System. For the future work, real time implementation would be a huge plus, as of now, only UWB works in real time, RFID & Wi-Fi both require offline data processing. It would also be great to combine all three of these technologies into one hardware solution, perhaps a device that is both a Wi-Fi & RFID reader, and a UWB mobile node. These two main ideas could develop a product that is much more user friendly both in hardware and software.

Appendix

KNN

```
load('RSSMap-T4.mat');
refpoints=RSSMap(:,1:8);
testpoints = RSSMap(:,12);
n1=1;  %#testpoints
n2=8;
K=3;
RP_x = [0 12.437 21.6091  21.6091  21.6091 12.1920 0 0];
RP_y = [0 0 0 8.6614 16 16.51 16.51 9.0371];
trainingpoint(:,1)=RP_x';
trainingpoint(:,2)=RP_y';

for i=1:n1
    M=[];
    a=testpoints(:,i);

    for j=1:n2
        b=refpoints(:,j);
       % b(idx)=[];
        m=a-b;
        M(j)=sqrt(sum(m.^2));
    end
    M1=sort(M);
    ID=[];
    pro=[];
    for j=1:K
        n1=find(M==M1(j));
        pro(j)=atan(1/M1(j));
        n=n1(1);
        ID=[ID n];
    end
    Estimate(i,1)=pro*trainingpoint(ID,1)/sum(pro);
    Estimate(i,2)=pro*trainingpoint(ID,2)/sum(pro);
%     E=testingpointGT(i,:)-Estimate(i,:);
%     ErrKNN(i,1)=sqrt(sum(E.^2));     %error


end
```

Kernel

```
sigma = 20;
kernel_result = [];

RP_num = [1:8];
RP_x = [0 12.437 21.6091  21.6091  21.6091 12.1920 0 0];
RP_y = [0 0 0 8.6614 16 16.51 16.51 9.0371];
NewDatabase_MAC = MAC;
NewDatabase_RSS = RSS;
step_RSS = rss;
step_MAC = ap;
Test = [];

for k = (1:length(NewDatabase_MAC))
    num = find(strcmp(step_MAC, NewDatabase_MAC(k)));
    if not(isempty(num))
        Test = [Test step_RSS(num)];
    else
        Test = [Test -100];
    end
end
% generate the Test vector for Kernel function
```

```
K = [];
for j = RP_num
    Train = NewDatabase_RSS(:, j)';
    K = [K Kernel(Train, Test, sigma)];
end

K = K./sum(K);
kernel_x = K*RP_x';
kernel_y = K*RP_y';
kernel_result = [kernel_x kernel_y];
```

LOS Measurement Plotting

```
clear all; close all; clc;
real_x = [-3962 -4876 -3890.96 -2133.6 -6004.56 -5029.2 1127.76 914.4...
        3505.2 2926.08 3048 -304.8 1700.784 1574.9 381];
uwb_x = [-3392 -4752 -3552 -2256 -4208 -4416 2000 1344 4416 3584 1888 -688 ...
        1296 1760 -448];
labels = cellstr('real');
labels1 = cellstr('uwb');
labels2 = cellstr('fixed node');
real_y = [-2133 914 -4419.6 3352 -4632 -1767 2712 ...
        -4343.4 -1117.7 -4038.6 1676.4 3200.4 2565.41 4673.5 4597.4];
```

```
uwb_y = [-1344 930 -3680 3232 -2976 -1152 2720 -4576 -1264 -3968 1072 3500 ...
            2144 5728 4624];
fixed_x = [0 4572 -6096 0];
fixed_y = [6096 1000 1000 -4876];
figure
hold on
axis( [-8000 6000 -6500 6500] );
scatter(real_x, real_y, '.r');
text(real_x, real_y, labels, 'HorizontalAlignment','right', ...
          'VerticalAlignment','bottom');
scatter(uwb_x, uwb_y, 'mo');
text(uwb_x + 0.1, uwb_y +0.1, labels1, 'HorizontalAlignment','left', ...
          'VerticalAlignment','top');
plot(fixed_x, fixed_y, 's');
text(fixed_x + 0.1, fixed_y +0.1, labels2, 'HorizontalAlignment','right', ...
          'VerticalAlignment','top');
hold off
```

CDF Error in LOS

```
real_x = [-3962 -4876 -3890.96 -2133.6 -6004.56 -5029.2 1127.76 914.4...
           3505.2 2926.08 3048 -304.8 1700.784 1574.9 381];
uwb_x = [-3392 -4752 -3552 -2256 -4208 -4416 2000 1344 4416 3584 1888 -688 ...
           1296 1760 -448];
```

```
real_y = [-2133 914 -4419.6 3352 -4632 -1767 2712 ...
          -4343.4 -1117.7 -4038.6 1676.4 3200.4 2565.41 4673.5 4597.4];
uwb_y = [-1344 930 -3680 3232 -2976 -1152 2720 -4576 -1264 -3968 1072 3500 ...
           2144 5728 4624];


true_x = (abs(real_x) - abs(uwb_x)).^2;
true_y = (abs(real_y) - abs(uwb_y)).^2;


a = sqrt((true_x) + (true_y));


[b x]=hist(a,100);
num=numel(a);
c=cumsum(b/num);
plot(x,c);
```

OLOS Measurement Plotting

```
clear all; close all; clc;
real_x = [-4876 -3962.4 -4927.7 -5003.6 -5486.4 -3098.9 1473.19 1524 701 ...
            2971.8 1615.44 1920.24 3276.6 4241.6 4902.1 6019.8 ];
uwb_x = [-4864 -3792 -4464 -4368 -3548 -2736 352 1184 336 2336 1952 2896 ...
           3728 4784 6608 5520];
labels = cellstr('real');
labels1 = cellstr('uwb');
labels2 = cellstr('fixed node');
real_y = [-3048 -1244.6 -4776.2 2006.5 4053.84 2794 2463.7 4597.34 ...
```

```
      2311.4 1498.4 -1270.1 -3124.2 -1320.7 -1727.2 -4013.18 ...
      -4749.79];
uwb_y = [-2800 -1184 -4192 1936 3392 2848 2272 4640 3056 1232 -1216 ...
    -2800 -1328 -1312 -3072 -3984];

fixed_x = [0 6603 -6096 0];
fixed_y = [6096 533 1000 -4876];
figure
hold on
axis( [-8000 7000 -6500 6500] );
scatter(real_x, real_y, '.r');
text(real_x, real_y, labels, 'HorizontalAlignment','right', ...
     'VerticalAlignment','bottom');
scatter(uwb_x, uwb_y, 'mo');
text(uwb_x + 0.1, uwb_y +0.1, labels1, 'HorizontalAlignment','left', ...
     'VerticalAlignment','top');
plot(fixed_x, fixed_y, 's');
text(fixed_x + 0.1, fixed_y +0.1, labels2, 'HorizontalAlignment','right', ...
     'VerticalAlignment','top');
hold off
```

CDF Error in LOS

```
real_x = [-4876 -3962.4 -4927.7 -5003.6 -5486.4 -3098.9 1473.19 1524 701 ...
```

```
          2971.8 1615.44 1920.24 3276.6 4241.6 4902.1 6019.8 ];

uwb_x = [-4864 -3792 -4464 -4368 -3548 -2736 352 1184 336 2336 1952 2896 ...

         3728 4784 6608 5520];



real_y = [-3048 -1244.6 -4776.2 2006.5 4053.84 2794 2463.7 4597.34 ...

         2311.4 1498.4 -1270.1 -3124.2 -1320.7 -1727.2 -4013.18 ...

         -4749.79];

uwb_y = [-2800 -1184 -4192 1936 3392 2848 2272 4640 3056 1232 -1216 ...

       -2800 -1328 -1312 -3072 -3984];

true_x = (abs(real_x) - abs(uwb_x)).^2;

true_y = (abs(real_y) - abs(uwb_y)).^2;



a = sqrt((true_x) + (true_y));



[b x]=hist(a,100);

num=numel(a);

c=cumsum(b/num);

plot(x,c)
```

Geometric Algorithm for UWB

```
function pos = mob_pos_read(LOG_FILE_PATH, LOG_FILE_NAME, PLOT_EN)



if nargin < 3

  PLOT_EN = false;
```

```
end

fid = fopen([LOG_FILE_PATH LOG_FILE_NAME '.txt']);

num_pos_est = 0;
tline = fgets(fid);
pos = [];

while ischar(tline)
    if ~isempty(strfind(tline(1:5),'@POS:'))
        idx_comma = strfind(tline, ',');
        idx_semic = strfind(tline, ';');
        if ~isempty(idx_semic)
            num_pos_est = num_pos_est + 1;
            pos(num_pos_est,1) = str2num(tline(6:idx_comma(1)-1));
 pos(num_pos_est,2) = str2num(tline(idx_comma(1)+1:idx_comma(2)-1));
 pos(num_pos_est,3) = str2num(tline(idx_comma(2)+1:idx_comma(3)-1));
 pos(num_pos_est,4) = str2num(tline(idx_comma(3)+1:idx_semic(1)-1));
      end
    end
    tline = fgets(fid);
end
fclose(fid);

if(pos(num_pos_est,3) || pos(num_pos_est,2) > 20000)
if(PLOT_EN)
    figure(101);
    hold off;
    for k=1:num_pos_est
        plot(pos(k,1),pos(k,2),'*');
        hold on;
    end
    daspect([1,1,1]);
    grid on;
pos_mean = mean(pos);
plot(pos_mean(1), pos_mean(2), 'ro');

for k=1:num_pos_est
    pos_err(k,:) = pos(k,1:2) - pos_mean(1:2);
    r_err(k) = sqrt(pos_err(k,:)*pos_err(k,:)');
end;

figure(102);
hold off;
hist(r_err, 200); grid on
end
```

Kalman filter Code

```
    % The parameters in KalmanFilter

    p = [0;16.51]; % initial position

    P = [p];

    sigma_sin = var(sin(deg2rad(180+sigma_theta(1).*randn(1000, 1))));

    sigma_cos = var(cos(deg2rad(180+sigma_theta(1).*randn(1000, 1))));

    sigma_x_ins = sigma_sin*sigma_step(1); % std of sin(theta)*step error

    sigma_y_ins = sigma_cos*sigma_step(1); % std of cos(theta)*step error

    sigma_x_ker = 2.9132; % std of kernel error for x (var2)

    sigma_y_ker = 2.7307; % std of kernel error for y


%     sigma_x_ker = 0.029132; % std of kernel error for x (var2)

%     sigma_y_ker = 0.0027307; % std of kernel error for y


    % The matrix for KalmanFilter


    A = [1 0; 0 1];


    C = [1 0; 0 1];


    E_x = [sigma_x_ins^2 0;

           0 sigma_y_ins^2]

    E_z = [sigma_x_ker^2 0;

           0 sigma_y_ker^2];

    E = E_x;
```

```
% Kalman Filter Process

for i = 1:length(theta)

    %  B matrix is changing

    %  B = [step(i) 0; 0 step(i)];

    %  B = [0 0; 0 step(i)];

    B = [step(i) 0; 0 0];


    U = [sin(deg2rad(theta(i))); cos(deg2rad(theta(i)))];


    p_bar = A * p + B * U;


    E_bar = A * E * A' + E_x;


    K = E_bar * C' * (C * E_bar * C' + eye(2) * E_z)^(-1);


    p = p_bar + K * (kernel(i,:)' - C * p_bar);


    E = (eye(2) - K * C) * E_bar;


    P = [P p]; % store the filtering result


end
```

Refrences

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

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

[3] "How RFID Works." *How RFID Works*. N.p., n.d. Web. 30 Apr. 2015. http://goo.gl/u6yv9B

[4] *Circular Polarization vs Linear Polarization., Which is right?, RFID Insider, Shain Armstrong,* Atlas RFID, 2013. Web: http://blog.atlasrfidstore.com/circular-polarization-vs-linear-polarization

[5] "Circular Polarization vs. Linear Polarization: Which Is the Right RFID Antenna?" *RFID Insider*. N.p., 03 Apr. 2013. Web: http://goo.gl/bQrBn6

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

[7] "USB Plus+ RFID Reader." USB Plus+ RFID Reader. Web. http://goo.gl/dAoRQS

[8] Frank, Korbinian, Bernhard Krach, Noel Catterall, and Patrick Robertson. *Development and Evaluation of a Combined WLAN & Inertial Indoor Pedestrian Positioning System*. Working Paper. 2009. Print

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

[10] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[11] USC Electrical Engineering: Innovation and Excellence. (2005, January 1). Retrieved December 15, 2014, from http://viterbi.usc.edu/news/news/2006/usc-electrical-engineering.htm

[12] Zetik, R.; Sachs, J.; Thoma, R., "UWB localization - active and passive approach [ultra wideband radar]," *Instrumentation and Measurement Technology Conference, 2004. IMTC 04. Proceedings of the 21st IEEE* , vol.2, no., pp.1005,1009 Vol.2, 18-20 May 2004 doi: 10.1109/IMTC.2004.1351232

[13] Zhang Guoping; Rao, S.V., "Position localization with impulse ultra wide band," *Wireless Communications and Applied Computational Electromagnetics, 2005. IEEE/ACES International*

Conference on , vol., no., pp.17,22, 3-7 April 2005 doi: 10.1109/WCACEM.2005.1469518

[14] Schroeder, J.; Galler, S.; Kyamakya, K.; Jobmann, K., "NLOS detection algorithms for Ultra-Wideband localization," *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on* pp.159,166, 22-22 March 2007 doi: 10.1109/WPNC.2007.353628

[15] Gezici, S.; Zhi Tian; Giannakis, G.B.; Kobayashi, Hisashi; Molisch, A.F.; Poor, H.V.; Sahinoglu, Z., "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks," *Signal Processing Magazine, IEEE* , vol.22, no.4, pp.70,84, July 2005 doi: 10.1109/MSP.2005.1458289

[16] Bensky, Alan. Wireless Positioning Technologies and Applications. Boston, Mass.: Artech House, 2008. 29 - 31. Print.

[17] Alavi, B.; Pahlavan, K., "Modeling of the TOA-based distance measurement error using UWB indoor radio measurements," *Communications Letters, IEEE* , vol.10, no.4, pp.275,277, Apr 2006 doi: 10.1109/LCOMM.2006.1613745

[18] *RTLS Developer's Guide.* 2014. Red Point Positioning Corporation.