A BREADCRUMB NETWORK FOR ASSISTING WITH AUTONOMOUS ROBOT LOCALIZATION

by

David Grabowsky

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Electrical Engineering

Charlotte

2021

Approved by:

Dr. James Conrad

Dr. Chen Chen

Dr. Courtney Smith-Orr

Dr. Aidan Browne

ABSTRACT

DAVID GRABOWSKY. A BreadCrumb Network for Assisting with Autonomous Robot Localization. (Under the direction of DR. JAMES CONRAD)

Localization and communication are critical components for functioning autonomous robots. The infrastructure required for these operations commonly includes global positioning system (GPS) and easily recognizable and re-identifiable landmarks. However, these types of infrastructures are not always readily available. GPS typically uses a low power signal that can be denied intentionally or is unable to penetrate certain materials. Unique landmarks can be difficult to find in unstructured environments like forests, where many potential landmarks can seem nearly identical. To solve this, this research has developed a deploy-able electronic way-point system dubbed 'BreadCrumbs'. BreadCrumbs function as electronic landmarks that can provide localization and communication capabilities to a robot in environments where such infrastructure is not present. When deployed by a forward moving agent with a set destination, the BreadCrumbs also form a series of way-points which reduce the possible state space an autonomous robot must search through when path planning in an unknown or un-mapped environment. The BreadCrumbs are self localizing and have several methods for initial location determination based on the environment they are placed in. GPS is not required for the BreadCrumbs to function and, once established, they can function as landmarks for autonomous robots by providing range data from radio signal strength with a path loss exponenet determined through a Deep Determinsitc Policy Gradient algorithm. The algorithm is designed such that a path loss exponent for each BreadCrumb location is learned during run time.

ACKNOWLEDGEMENTS

Foremost, I want to thank my parents whose constant encouragement and sacrifice ensured I was able to reach the highest level of education.

I would also like to thank my advisor, Dr. James M. Conrad for his guidance and support throughout my years in his lab. In addition, I am thankful for the many opportunities for growth, funding, and experience he provided during my time in graduate school.

I also want to thank my committee, Dr. Chen Chen, Dr. Courtney Smith-Orr, and Dr. Aidan Browne, for spending their valuable time as part of this committee.

Finally, I'd like to thank Dr. Sam Shue who served as a mentor during my formative years in graduate school.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

DDPG An acronym for Deep Deterministic Policy Gradient

ECE An acronym for Electrical and Computer Engineering.

EKF An acronym for Extended Kalman Filter

GNSS An acronym for Global Navigation Satellite System

GPS An acronym for Global Positioning System

ML An acronym for Machine Learning

PLE An acronym for Path Loss Exponent

RF An acronym for Radio Frequency

RL An acronym for Reinforcement Learning

RSSI An acronym for Range Signal Strength Indication

SLAM An acronym for Simultaneous Localization and Mapping

CHAPTER 1: Introduction

1.1 Motivation

Autonomous robots are becoming commonplace across a variety of disciplines and applications. Examples include, search and rescue, cave exploration, and supply trains. In order to operate precisely, these robots must understand their location within an environment. To understand their location, they must recognize and re-observe landmarks which contain enough complexity to describe their location on a global map and within a robots local reference frame. These landmarks are not always readily available given the environment an autonomous robot might operate it. For example, search and rescue autonomous robots may have difficulty recognizing visual landmarks in densely forested unstructured terrain. Buildings or caves that are subjects of natural disasters or un-mapped also present similar challenges. Primarily, the natural layout of the environment is not always particularly conductive to an autonomous robots localization operations. To augment this, wireless signals can be used for localization. Unfortunately, these applications often take place in environments where typically available augmentations, such as, Global Position Systems, are either unavailable, imprecise, or actively denied. Therefore, deploy-able wireless localization systems become and attractive option. Radio signal strength indications are a popular choice for wireless localization systems, but are heavily subject to the difficult to generalize noise and interference presented by the deployed environment. In addition, many parts of these operations take place in environments where a global map is not readily available. In the case of search and rescue in a heavily forested terrain, this presents the risk of wasting time following incorrect paths in a scenario where time is a premium. In the end, the autonomous robot has the burden of lo-

calizing, mapping, and path planing simultaneously in an complex and inhospitable environment. This is a heavy burden for autonomous robots to overcome.

While there is no be-all solution to this problem, it is desirable to lessen the burden placed on an autonomous robot as much as possible. We could leverage the assumption that a human is operating ahead of the autonomous robot in order to deploy wireless landmarks. If these wireless landmarks could provide rough self localization to the autonomous robot, then this could form an initial trail for the autonomous robot, eliminating the risk of the robot wasting time exploring its environment or following incorrect paths. In addition these devices could function as unique landmarks, providing the autonomous robot with much needed localization information.

1.2 Objectives and Scope

This work seeks to aid autonomous robots functioning in environments where: GPS is not available, landmarks are difficult to identify, and path planning mistakes lead to unacceptable amounts of wasted time and energy. Environments could include: large wooded outdoor environments with dense tree line foliage or unmapped indoor environments where typical landmarks may not be consistent and GPS is unavailable. Each of these environments is expected to have significant wireless signal interference, hence this work also seeks to minimize the impact of that error. A primary assumption made to accomplish this is that a human is present and can establish infrastructure designed by this work prior to or during the operation of an autonomous robot. The specific functions we seek to aid are, localization, path planning, and landmark identification. We do not go into specific details on how the autonomous robot is accomplishing any of these functions, but rather show how the developed system can provide a generic tools to aid a variety of algorithm in solving the problems associated with each of theses.

For localization, the challenge present in stated environments begins with the lack of GPS. GPS is widely used by autonomous robots for localization in large outdoor

environments, but is not always reliable or available. Therefore, an autonomous robot must heavily rely on being able to identify landmarks in its surrounding environment. This leads into the landmark identification challenges. Unique visual landmarks in unstructured heavily wooded areas can be difficult to come by, and depending on the length and direction of travel may never be re-observed again, or worse, may be incorrectly re-observed. An autonomous robot that utilizes visual landmarks may build up an extremely large memory footprint where individual landmarks have an extremely low utilization rate. In disaster scenarios, indoor environments may not have consistent landmarks. A block of rubble initially observed may shift or break leading to the loss of landmark that was expected to be re-observed. This work focus on the how wireless radio-frequency (RF) signals emitted from deploy-able devices can be used to tackle this problem. We focus on how wireless landmarks can provide measurements and landmark uniqueness in these environments.

The method by which this localization is accomplished involves the use of wireless RF ranging technology. This technology is especially susceptible to interference and noise produced by the environment. Thick concrete walls can dampen a RF signal while metallic objects can cause unexpected reflections or shifts of the transmission. As this RF ranging is one of the primary pieces of data provided to the autonomous robot by this system, part of this work heavily focuses on how the noise which afflicts these range measurements can be reduced. Specifically, we investigate how a general system can be deployed using neural networks that has the capability to error correct RSSI noise based on the specifics of the environment a device is deployed in.

This work also seeks to provide the robot with location of deployed electronic devices. These electronic devices must be able to localize themselves. Hence, another focus of this work is localization algorithms. Furthermore, we examine the conditions under which initial localization algorithms can be deployed depending on the availability of data.

In addition, this work focuses on the problem of path planning. In outdoor environments the possible space of paths from a starting point to a destination is often massive. Placing the burden of determining the correct path on the autonomous robot when it is not aware of the full details of the environment is difficult. By examining the two specific environment mentioned above, we focus on how a system can be developed to alleviate this burden. We also examine the possibility of a forward moving human agent and how that agent can be used to deploy a network that will narrow down the size of the path space a robot must search through while still providing localization and communication capabilities.

1.3 Contribution

The primary contribution of this work is the development and implementation of the BreadCrumb Network. This is a wireless sensor network designed to be deployed along un-mapped paths that lack easily identifiable landmarks and GPS signals. The BreadCrumbs are designed to provide autonomous robots with the assistance needed to localize and operate within these environments. Multiple factors accomplish this including: an initial location algorithm that builds a system of constraints based on range measurements between deployed BreadCrumbs, then collapses those constrains as additional measurements are taken. The network also provides distance estimations from RSSI measurements that utilize a novel application of a Deep Deterministic Policy gradient for determine environmental factors which impact the model used to estimate distance from RSSI. Finally the network forms a communication chain allowing the autonomous robot the maintain a connection with other devices over long distances. By providing these functions, the BreadCrumb network lessons the burden an autonomous robot would face when operating in a GPS denied unstructured environment.

1.4 Overview and Structure

The remained of this dissertation is organized into four chapters. Chapter Two covers the background theory and relevant works associated with this research. Chapter Three shows the design of the BreadCrumb system. It will walk through all factors of how the system functions and its operational capacity. Chapter 4 covers the experimental setup used to determine the feasibility of BreadCrumb system as well as an analysis of the data gathered. Chapter 5 will go over the conclusion of this research, as well as potential future avenues of research

CHAPTER 2: Background

2.1 Introduction

The last decade has seen an explosion of interest in autonomous robots. These robots are capable of operating without direct control from humans. There are three important factors that enable a robot to be autonomous: perception, decision, and actuation. Perception constitutes devices that enable a robot to observe its environment. In other words, perception devices give robots inputs from their environment which can be used to make decisions. Devices that provide these inputs are wide and varied. They include video cameras which can provide inputs similar to how humans observe the environment in the form of RGB streams of images. Perception devices can also include laser scanners which sweep areas around the robot for planar objects, or even simple bump sensors that only trigger when the robot is physically pressing against a surface. As long as the device gives the robot an input from it environment, it can be considered a perception device. Once the robot gathers inputs from its environment with perception sensors, it must then make decisions based on those inputs. The decision factor is the autonomous robot's ability to make determinations based on inputs and then generate the commands necessary to execute those determinations. 13 system is typically thought of as the component in an autonomous robot that is making the decision. Typically, a computation device or embedded system is the component responsible for making the decisions. The devices utilize tools such as Kalman Filters for determining the accurate position of landmarks, neural networks for identifying specific objects from perception inputs, and many more. Finally, there is the third factor that composes autonomy for robots, actuation. Actuation is how the robot physically interacts with its environment. This generally takes the form of

a motor or pneumatic device that moves the robot or its appendages based on the decisions made by the robot. Thus when a robot such as the one seen in utilizes these three factors, it has the potential to become an autonomous robot.

With the increase in interest in autonomous robots, the burden upon each of these factors has compounded as their capabilities are continually being pushed further and further. Specifically, we consider the burden upon the perception and decision factor of autonomous robots. To understand these burdens, we must first examine the scope of applications certain autonomous robots are being used for. We need also examine the environments where autonomous robotics are used, like those in Figure 2.1, and the challenges they can present.

(a) (b)

(c)

Figure 2.1: Difficult environments for autonomous robot operations (a) Forests with dense and similar foliage [9] (b) Underground cave systems [10] (c) Damaged buildings with large amounts of debris and rubble [11]

One such application involves the mapping, traversal, and communication within underground caves and tunnels. The United States Army has had a increased inter-

est in operations that take place in underground caves/tunnels, especially in scenarios where robotics may be applied [12]. History has shown a variety of conflicts where underground systems have been used to great effect, some examples include the Vietnam War and contemporary conflicts such as Afghanistan. These tunnels and cave systems are fraught with hazards, in fact, there is significant cross-over with the hazards civilian first responders face. Unstable structural conditions can result in collapses as individuals move through caves, poor air quality can make exploration efforts difficult, and un-mapped tunnel layouts can make searching for survivors of accidents a tedious effort. In adversarial conditions, even the act of rounding an unexplored cave bend can be extremely dangerous. Hence, agencies such as the Defense Advanced Research Projects Agency (DARPA), are investing in solving this problem. DARPA recently launched the Subterranean Challenge (currently on-going) [13] which "seeks novel approaches to rapidly map, navigate, and search underground environments during time-sensitive combat operations or disaster response scenarios." Articulation in these scenarios is hard, but the difficulty of perception and decision making should not be understated. For example, an autonomous robot searching for survivor's in a geological cave will face two unique challenges. First, the cave will likely be non-structured and difficult to map. Of particular difficulty will be identifying unique landmarks in the cave and being able to re-observe them with confidence. Further more, once the survivors are located, communication will likely be an issue since Radio Frequency (RF) signals could have trouble reaching the robot depending on the depth and composition of the cave, forcing the robot to attempt to retrace its path out of the cave in order to inform rescuers of the survivors. With the lack of available perception inputs and challenging time constraints, operating in this environment can prove quite challenging to autonomous robots.

Disaster relief and rescue operations comprise a third example of operations where autonomous robots are seeing an increase in use.. After natural disasters occur one

of the first priorities is search and rescue operations. In events such as earthquakes, individuals may have become trapped under rubble and require rescue. While humans will almost certainly be quicker and more nimble then a robot in environments where rubble and debris make motion difficult, autonomous robots still have some key advantages. First, they can be purpose built small and flexible so that they can fit into spaces where humans cant, enabling them to explore blocked off passages and collapsed buildings. Second, robots are disposable. In early disaster relief, the operating environment can be unstable and treacherous. Hence, autonomous robots can be used to build an understanding of the condition of the environment and key search locations before humans are sent in. What complicates this, is the 21 possibility that in a disaster scenarios, the existing infrastructure can potentially become disabled. Radio towers and substations may no longer provide effective communication or power. The robot is then forced to localize and communicate using only the on-board sensors/devices it has. Identifying landmarks in these environments can be tricky. With the established infrastructure damaged, landmarks may have to be identified from items like rubble. Adding to the complexity, the environment may still be in flux, meaning these landmarks could change over time. GPS could be used to combat this issue, but depending on the disaster, autonomous robots may be expected to explore underneath rubble or in buildings that may collapse, thus reducing the usefulness of GPS. The nature of this environment makes it extremely complex for an autonomous root to operate it. The robot must overcome challenges to perception and decision making in an environment that is potentially still in flux. On the topic of rescue operations, we also examine scenarios where it is beneficial to have a human move ahead to an injured individual to provide first aid, while an autonomous robot follows behind the human carrying heavier/bulkier equipment such as a stretcher. In mountainous forest rescue, when the location of an injured individual is known, it is not always possible to directly reach that individual via helicopter. Teams may have to be

deployed miles away and must make their way through unknown terrains as quickly as possible to locate the individual. Bulky and or heavy equipment in these scenarios proves to be a detriment as humans must sacrifice speed and energy carrying it. Lightly equipped individuals can quickly race ahead to provide first aid while other follow more slowly behind carrying the heavier equipment. An autonomous robot robot can be used to replace the individuals carrying the heavy equipment. Even in non-emergency situations, having a autonomous robot carry heavy equipment while individuals move ahead of said robot can be very useful. In worse case scenarios where multiple individuals are injured, an autonomous robot such as in Figure 2.3, could be used to autonomously transport the injured individuals back to a safe location.

Figure 2.2: Robot rescue vehicle developed by Rheinmetall [1] which can autonomously evacuate casualties over long distances.

This scenario creates a high burden on the robots decision making capabilities. Since the rescuers are racing to the injured individual, the robot will likely lose visual sight of the rescuers as they pull ahead. Depending on the foliage and range, the robot may even lose all forms of communication. Since the environment is not known, the

robot will be forced to make difficult decisions when it comes to path planning. Adding to the difficulty is that the best path in an outdoor unstructured environment is not always clear. A straight line to the injured persons location may be the most direct route, but could also lead to obstacles that must be backtracked around. While backtracing it may become difficult to reobserve previously identified landmarks making localization difficult. Essentially, the search space of the unknown map is an extremely daunting challenge for the robot. GPS may be able to aid the robot, but in dense foliage or poor weather conditions GPS could become unreliable. It seems likely given this challenge, that while attempting to follow the rescuers, the robot will could lose localization capabilities, or make poor navigation decisions leading to wasted travel time. The issue here is again, that there is a lack of existing infrastructure for the robot to utilize or take advantage of.

These examples all follow a common theme, in that a robots perception and decision factors are challenged by the environment the robot operates in. Identifying landmarks is critical for an autonomous robot to function. Landmarks play a key roll in localization and path-planning decisions. Furthermore, the environment discussed tend to lack infrastructure many autonomous robots perceive as landmarks such as signage or structured layouts. Existing technology such as GPS which is widely relied on to augment localization and path planning algorithms, proves to either be unreliable or unavailable in many of these environments. Without these tools, the burden placed on the robots decisions making capabilities becomes substantial, especially as it relates to navigation, path planning, and communication. Hence, the BreadCrumb network has been designed to tackle these challenges. The BreadCrumb network is composed of electronic landmarks that can be rapidly deployed to provide robots with infrastructure for localization, path planning, and communication in environments where such tools are not readily available. The goal being to reduce the burden of localization and path planning in challenging environments.

Range signal strength indication is proving to be a popular choice for indoor localization techniques. Primarily because it is widely available from various device such as wireless fidelity (WiFi) routers, low-cost Zigbee devices, and more. Since WiFi routers can already be found setup in many modern buildings and devices like Zigbees are relatively low cost this makes utilizing RSSI for indoor localization an attractive option. However, given the multitude of options, it can be come difficult to determine which device to use. Hence work, such as that conducted by [14], is focused on comparing and contrasting the benefits and draw backs of various peices of this technology. In [14] compares the accuracy and power consumption of four different commonly used devices for indoor RSSI based localization's: bluetooth low energy devices (BLE), WiFi devices, Zigbee devices, LoRaWAN.

Bluetooth Low Energy was specifically designed for applications where power was a premium and data transfer rates were expected to be low. It was introduced by the Bluetooth Special Interest Group with the aim of reducing power consumption and the cost of devices utilizing [15] Since then, there has been an increased focus on how BLE technology can be utilized for indoor localization. [16] distributed 19 BLE beacons over a roughly 600 meter squared area and demonstrated/analyzed how these devices in fixed locations could be used to localize a consumer device. A comparison to WiFi fingerprinting was also provided. The work conducted by [17] was to propose a framework for indoor position utilizing BLE tags which used RSSI to localize devices.

WiFi devices offer a readily available source of infrastructure for robots to utilize for indoor localization. [18] provides an analysis of several indoor localization techniques and shows how a single WiFi access point can be used to support an indoor localization system. Work conducted by [19] focuses on utilizing GraphSLAM to localize and map with signal strength from WiFi devices.

2.2 Localization and Path Planning

Range signal strength indications are the backbone measurement utilized by this, and many other works. Fundamentally, this is a value that represents the signal strength of a received transmission from a transmitter. As the signal radiates out from the transmitter, its power decreases. In fact, this is typically stated as the signal power being inversely proportional to the distance traveled squared. However, this only holds true when there are no sources of interference, which is highly unlikely for any practical uses. As shown in [20] this relationship can quickly breakdown when sources of interference are present, making RSSI difficult to utilize, especially when the goal is to estimate distance using RSSI for localization purposes. To utilize RSSI for localization through distance measurements other models are forms of analysis are often required. Chapter 3 gives more details on what RSSI is, the environmental factors that influence it, as well as the models which are used to estimate distance from the measurements. The remaining subsections of this section detail other works that have utilized RSSI for localization. It is important to reiterate that the localization goal of BreadCrumb network is to provide initial locations of the BreadCrumb for path planning purposes for the autonomous robot. Enhanced localization through techniques like Kalman Filtering are left as a technique to the device utilizing the BreadCrumb network. In addition, the BreadCrumb network operates under the constraint that each BreadCrumb can only communicate with its nearest neighbor, as defined by the prototype work conducted in [21].

Indoor locations are fraught with obstacles and materials that influence RSSI. These range from objects like walls to simple pieces of furniture or metallic objects like filling cabinets. The interference introduced by these objects makes the RSSI distance relationship extremely difficult to model. As shown in [22], the difficulty in determining the quality of a signal link between a mobile and static device can prove quite challenging. To the point where RSSI may seem like an unusable tool, despite its

seeming rise in popularity. One of the primary reasons for this is the changing nature of environments where these measurements take are gathered. While the transmitter and receiver have line of sight of each other in the same room the readings may be consistent, but as soon as it move to another room the environment factors change enough that any initial calibration done in the beginning room become untenable.

Hence, work conducted by the authors in [23] where a virtual calibration method is used to incorporate the interference introduced by objects and walls. They also present a probabilistic blueprint for the density of sensors needed in order to achieve a localization accuracy. However, that work, and many others, require the location of the deployed nodes to be known ahead of time. Work conducted also utilized a fixed devices for determining an environmental specific propagation model which is used to more accurately localize a stationary device. While these methods are impressive, the reliance on known deployment locations makes them limited. Given that our BreadCrumb network is meant to be deployed on the fly, the location of the BreadCrumbs will not be known ahead of time.

There has been work done that explores other types of trails which can be created and followed by robot agents. Russell proposed that a heated trail could be deployed by a robotic agent which would allow other robotic agents to sense paths previously taken [24]. Another interesting method proposed by Russel, Thiel, and Mackag-Slim was to have a robotic agent lay down a trail of volatile chemicals that could be detected by an odor sensor [25]. However, both of these proposals focus on indoor environments and would likely not efficiently scale to outdoor environments. Instead of a physical trail, work conducted by [26] used two UAVs, one of which had a GPS module. The UAV without the GPS module was able localize itself by communicating with the GPS equipped UAV. Other works have used fixed landmarks to plot paths from a start point to and end point such as in [27]. In this work cellular towers were used as landmarks to assist in localizing and planning the path of a UAV operating

in a GPS-denied environment.

Another key aspect of this work is the difficulty determining a path from the initial way-point to the end way-point when many possible paths exists. To clarify, this challenge presents itself when a robotic agent is aware of its starting location and the end location, but has no knowledge of the physical layout of the terrain between the start and end. Work conducted by [28] uses multiple robots to search for a path to the end way-point by having each robot communicate observed landmarks across a shared network until a suitable path is found. [2] also uses multiple robots to find a path to an end destination, however, the author also utilizes small physical beacons that can store information and pass that information to passing by robots.

Figure 2.3: Multiple robots path optimizing by moving physical beacons [2].

The author uses the robots to manipulate and place the beacons in locations that optimizes the path between the starting way-point and ending way-point. Both of these works emphasize creating efficient paths from a start way-point to an end way-point using multiple robots. These are suitable solutions to problems where efficient ferrying is required between two locations, but not suitable when a path between

way-points is needed to quickly be found and only traversed a few times.

2.2.1 Deployable Landmarks

Work has been done utilizing landmarks to localize robotic agents in outdoor environments. Ross and Hoque [3] propose that Fiducials can be used to augment GPS such that the measurement of error of the GPS can be reduced while the robot observes the Fiducials as in Figure 2.4.

Figure 2.4: Robot observing Fiducials to augment GPS [3].

Fiducials were also used by [29] along side visual measurements of an underwater structure in order to localize an autonomous underwater vehicle. Work conducted by [30] sought to localize an robot by combining odometry data and utlra-wide band ranging to a landmark. Instead of Fiducials, [31] utilized magnetic sensors for localization with landmarks. This method has the advantage of functioning where visual and radio-frequency localization methods are limited. In lieu of artificial landmarks, [32] used pre-existing satellite imagery to localize an UAV with a downward facing camera for areas where GPS was unreliable.

Research has also been conducted for creating and deploying a network with the purposes of communication or localization. The author of [33] developed a prototype network, part of the inspiration for this work, which could determine the location on

a line between a transmitter and receiver that was the least distance from an RSSI tag. LifeNet [34] is an ad-hoc network that can be used in emergency situations. It is a static network that is required to be deployed in the building to establish communication between firefighters and a base station. In addition, the Fire Information and Rescue Equipment project aimed to design and implement a set pf decision support tools for the assistance of firefighters [35]. They created a network called SmokeNet to trace the firefighters in a large building. Liu et. al conducted work aimed at solving a similar problem but suggested the use of an automatic dispenser system to deploy their network at appropriate locations [36]. A similar method of automatic deployment is used by Lai et. al for pipeline monitoring called TriopusNet [37]. In addition, research conducted by [38] sought to extend the range of wireless mesh networks in situations where an entity needs to move outside of the range of the network. Research by Souryal et al. examined a similar problem where they sought to extend the range of single-hop communication [39] via relays.

2.3 Radio Wave Propagation

The primary information being provided by the BreadCrumb network to autonomous robot comes in the form of distance/range estimations gathered from radio signal strength between a transmitting and receiving device. The radio signal strength decays as it propagates through the environment. In order to understand utilize these radio wave signals for determining distance, we must understand how they propagate through environment as well as how they are effected by the composition of the environment.

2.3.1 Free Space Electromagnetic Radiation Propagation

Wireless signals such as radio waves behave in certain ways as they travel through the environment, this is called radio propagation. A fundamental principle governs how electromagnetic radiation, including radio waves, function in free space. Free

space describes an environment where there are no obstacles or sources of interference between a transmitting and receiving device. The energy of general electromagnetic radiation follows the inverse-square law in these environments. The inverse-square law fundamentally describes how the power density of the radiation is proportional to the inverse of the square of the distance from the transmitter or source. Assuming that the radiation is uni-formally emitted from a single point. This is given as:

$$power = \frac{1}{distance^2} \tag{2.1}$$

As the the radiation propagates, the power is continuously displaced over an ever increasing spherical surface area the traveling emission. Hence, the power density of a radiation after propagating over a set distance can be described as:

$$Pd = \frac{Pt}{4\pi d^2} \tag{2.2}$$

Where, P_d is the calculated power density per unit area, P_t is the power at the transmitter, and d is the distance the radiation has propagated from the source. A visualization of this is given in Figure 2.5.

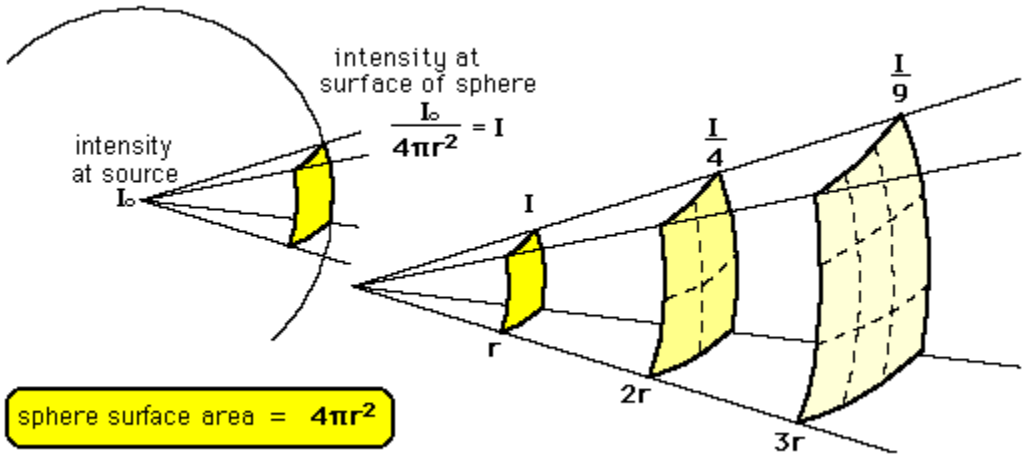


Figure 2.5: Visualization of the inverse square law showing the exponential decrease in intensity I as a multiple of the distance r [4].

This model is the fundamental explanation for how electromagnetic radiation energy propagates through free space. The Friss Transmission Equation utilizes this in order to create a model for determining the expected power at a receiving device radio wave device.

2.3.2 Friss Transmission Equation

In order to properly model the received power between the antenna of a transmitting and receiving device, it is not enough to only use the utilize the inverse power law. Instead, in addition to this law, the physical characteristics of the antenna being used from transmission must be taken into account. Like the power inverse law, the Friss Transmission Equation makes the assumption of free space, meaning the area over which the radio wave propagates is void of any sources of interference that may reflect, refract, or otherwise interfere with the signal. This includes phenomena such as Earths atmosphere or humidity. The equation is given as:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2} \tag{2.3}$$

where P_r is the power at the receiver, P_t is the power at the transmitter, G_t is the gain at the transmitter, G_r is the gain at the receiver, λ is the wavelength of the signal, and d is the distance between transmitter and receiver [40]. There are certain criteria that must be met for the equation to hold. The antennas are assumed to be in the far field from each other, meaning that the distance is expected to be greater than the wavelength $d >> \lambda$, the antenna are expected to be aligned with identical polarization, and the wavelength is assumed to be consistent give the bandwidth is narrow. It is important to note that Friss transmission equation does not incorporate any system losses that could occur from impedance's and imperfections in factors like the antenna construction. To account for this, an extension to the equation known as the Friss free space propagation model is given:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2 L} \tag{2.4}$$

where the variable L has been added to incorporate these losses to the system.

2.3.3 Environmental Impact on Radio Wave Propagation

In practical environments, there are rarely scenarios where free space actually exists. Under most circumstances, items like foliage, walls, and furniture will exists between a transmitter and receiver. These obstacles can have a significant impact on the way a radio wave signal propagates. In fact, there are many sources which can impact these signals as they propagate.

Reflection can occur when a radio wave impacts a conductive surface and experiences a change in medium, i.e air to metal. Part of the signal may be transmitted through the new medium while another part is reflected off of it. This can also occur when the wave impacts a large surfaces who's surface area is greater than the wavelength of the signal.

In a similar vein, radio waves can also experience refraction, where a gradual change in something like ions and free electrons, causes the waves to 'bend' as they travel. The phenomena is dependent on other factors such as electron density and signal wavelength, but the basic out come is the same. Essentially, as the wave enters a medium its course can become adjusted causing it to exit the medium at a different angle than the one it entered.

There is also diffraction, which is when a radio wave signal can be received even though the direct line of site between a transmitter and receiver is blocked. The receiver is considered shadowed by the block object, and depending on the wavelength of the transmitting signal, it may diffract around the obstacle and still reach the receiver. This can commonly be seen with buildings that have well-defined sharp corners and edges.

Scattering occurs when the radio wave has a similar wavelength with an object to the size of a given object with irregular geometry causing the signal ti 'scatter' unpredictably. This is especially common occurrence with forest foliage.

All of these various methods of interference, some of which can be seen visualized in Figure REF, can be grouped into a property called fading.

2.3.4 Fading

Fading is the result of the various phenomena which impact radio signal propagation. The end result is that as the signal propagates from the transmitter to the receiver, the expected power dissipation of the signal takes on added properties such that it is no longer solely dependent on distance and the characteristics of the physical system doing the transmitting and receiving. In the end, these phenomena cause the transmitting signal to appear to be received at the receiver from multiple paths as visualized in Figure 2.6.

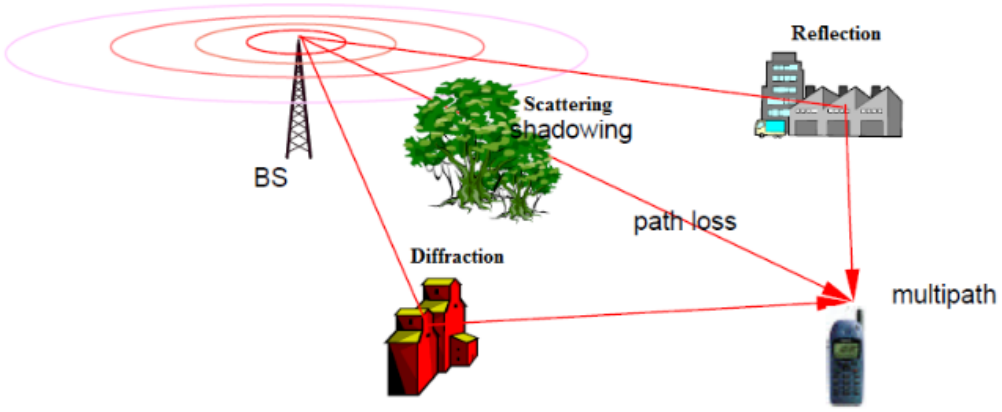


Figure 2.6: Visualization of the impact of phenomena such as reflection, diffraction, and scattering on the propagation of a radio wave signal from a transmitting base station to a receiver [5].

These signals arriving from multiple paths can become superimposed upon one another causing construct rive or destructive interference. These interference's are subject to how the new signals phase or amplitude is impacted.

Fading itself can be broken down into two broad categories to better describe the impact on the propagation: Large Scale Fading and Small Scale Fading. Each of these categories can also be broken down into various sub-categories as illustrated in Figure 2.7.

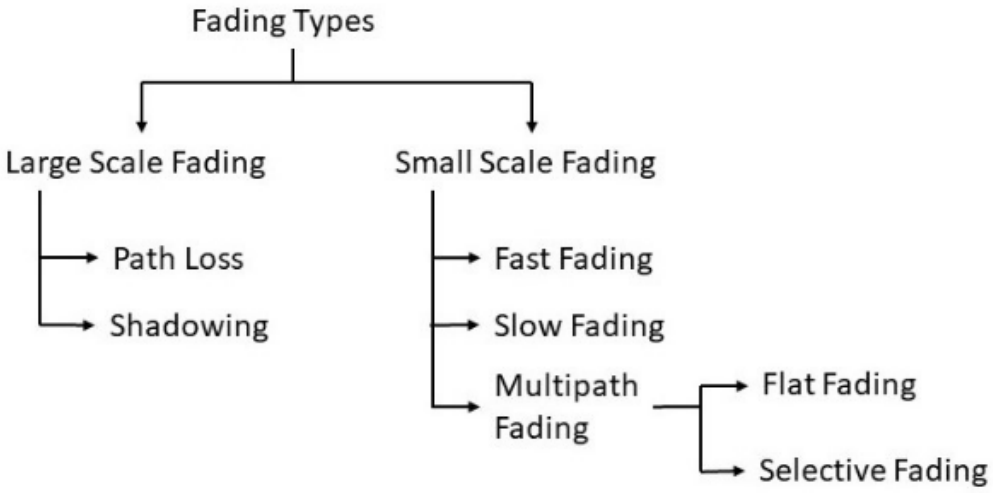


Figure 2.7: Categorical breakdown of the multiple types of fading.[6].

Large scale fading primarily incorporates the loss of signal power resulting from obstacles in between the transmitting and receiving device as well as the loss resulting from transmission over long distances. It is broken down into path loss which covers power loss over unit area per distance and shadowing which covers the loss of power due to obstructions in the path of the signal.

Small Scale Fading, sometimes refereed to as Rayleigh Fading, covers the impact on the signal over small distances and duration's of time. This is especially prevalent indoors. Three subcategories of Small Scale Fading include: Fast Fading, Slow Fading, and Multipath Fading. Fast fading occurs due to the movement of the transmitter or receiver, as well as surface reflection. Linear distortions are caused by this via high Doppler spread when the bandwidth of the signal is comparable or less than the Doppler bandwidth. Slow fading occurs primarily when line of sight is obstructed between the receiver and transmitter. This can be caused by buildings, geographical obstacles, etc. In this instance we see the impact of low Doppler spread which occurs when the bandwidth of the signal is greater than the Doppler bandwidth. Finally there is Multipath Fading, which itself is broken down into Flat and Selective Fading. Multipath Fading occurs when, as the name implies, a signal from a transmitter

reaches a receiver from multiple paths. The affect on the signal could result in Flat Fading, were all received signal frequency components are affected equally and amplitude fluctuates over time. Or the result could be selective fading, where only certain frequency components of the signal will have been affected.

Hence, it can be seen that in any practical scenario, environmental knowledge is a must have to properly model radio wave propagation and the impact on received power. Technically, all of these interference patterns are deterministic, meaning if we have precise knowledge of our environment then the impact on the signal is predicable. However, this knowledge can be extremely costly to obtain. Factors such as obstacle location, material composition, dimensions, temperature, humidity, and ionization would all need to be known. With absolute knowledge of the environment then with a method such as ray tracing it may be possible to predict exactly how a radio wave will interact with the environment. However, the cost of such an endeavour is simply too high to ever be practical. Thus, to narrow down this burden, we select what the most prominent source of interference will be, and examine methods for minimizing the error introduced by that interference. Since our operational range between devices will generally be under 1 kilometer, and our devices are expected to operate indoors or in areas with dense foliage, we focus our attention on how to overcome Multipath interference. Previously, the Rayleigh Fading was mentioned. This is an example of model for describing the impact of multi-path on an environment, showing the necessity for which such interference must be overcome. The Rayleigh fading model assumes there is no direct line of sight and that there are enough obstacles in an environment such that signals will be highly scattered to the point where each signal has an equal probability of reaching the receiver from any angle. Environments where this could happen include cities, dense forests, or indoor buildings. Under this assumption, central limit theorem describes that channel response can be defined as a Gaussian process incorporating the delays associated with the paths, the Rayleigh

distribution function can be given as :

$$p(r) = \begin{cases} \frac{r}{\sigma^2} \exp\left[-\frac{r^2}{2\sigma^2}\right] & \text{for } r \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

where r is the envelope amplitude or the received signal and 2σ is prediction mean power of the multi-path signal [41]. The Rician Fading Model extends Rayleigh by incorporating a dominant line of sight component which is present within the highly scattering environment. This can be given according to [42] as:

$$p(r) = \begin{cases} \frac{r}{\sigma^2} e^{-\frac{\left(r^2+A^2\right)}{2\sigma^2}} I_0\left(\frac{Ar}{\sigma^2}\right) & \text{for } (A \geq 0, r \geq 0) \\ 0 & \text{for } (r < 0) \end{cases} \tag{2.6}$$

where A denotes the peak amplitude and I is the modified Bessel function of first kind zero-order. With that, we are brought back to the problem of properly understanding radio wave propagation's and the impacts the environment will have on them. The questions then becomes, what can be done to analytically minimize the impact of these factors on our signal measurements.

2.3.5 Models for Overcomming Multi-path interference

In section 2.3.1 we described the fundamental inverse squares law which governs how electromagnetic radiation power changes over distance. Then section 2.3.2 described how the received power of a radio wave can be determined by using the previously stated law, as well as knowledge of the physical properties of the transmitting and receiving device through the Friss Free Space Propagation Model. However, this model does not factor in environmental impact such as multi-path interference, which we have shown has a large impact on signal accuracy. Two models are presented in order to discuss this.

2.3.5.1 Okumura Model

The Okumura model focuses on radio propagation in urban environments from data collected in Tokyo, Japan. The model also has extensions for suburban and open areas. The model is entirely based on empirical measurements, and does not have an analytical build up, however, based on empirical performance the model has proven to be considered on of the simplest and best in terms of accurate path loss prediction for these urban environments. [42]. The model is described as:

$$L(db) = L_F + A_{mu}(f, d) - G(h_{te}) - G(h_{re}) - G_{AREA} \tag{2.7}$$

where L is the median value of the propagation path loss, L_F is the free space propagation loss, A_{mu} is the median attenuation relative to free space, $G(h_{te})$ is the base station antenna gain factor, $G(h_{re})$ is the mobile antenna gain factor, and G_{AREA} is the gain due to the environment type, i.e. urban or open space. However, this model is limited in that it adapts poorly to rural areas were terrain change rapidly occurs, and is only applicable for frequency ranges between 150MHz to 1920 MHz as it is based off empirical data, although extrapolations have been done up to 3000 MHz.

2.3.5.2 Log-Distance Path Loss Model

Of particular interest to this work is the Log-Distance Path Loss Model. This model describes how received signal power decreases logarithmically with distance both indoors and outdoors [42]. This is expressed through a critical variable n, known as the path-loss exponent, which is used to parameterize how the power decreases with distance given the environment a signal propagates through. The model can be described as:

$$P_L = P_L(d_0) + 10n \log \frac{d}{d_0} \tag{2.8}$$

where P_L is the overall path loss in decibals, $P_L(d_0)$ is the path loss measured between a transmitter and receiver at reference distance d_0, and d is the current distance between transmitter and receiver. While this does not provide a holistic representation of the impact on radio wave interference, its generalization has proven effective and is very popular as it has seen wide use in the literature [43] [44] [45] [46]. It should be noted that a random zero-mean Gaussian noise variable X_g with standard deviation ω is typically added to model interference resulting from shadowing. When added the model is sometimes referred to as the Log-Normal Shadow Model. In other cases such as fast fading, it is normally appropriate to replace X_g with something that better models the interference such as Rayleigh or Ricean random variables. However, we exclude these variables form the formulation for the sole reason that we are interested in determining d, not the path loss, and these variables only help to model the path loss.

Even with this generalization, the path-loss exponent is still deeply tied to the the specific composition of the environment where the transmitter and receiver are operating. For example, the path-loss exponent for transmitter and receiver on either side of a large office building will be much different than the path-loss exponent on either side of a wall inside that building. Determining an accurate estimation of the path-loss exponent for a given environment is extremely important. Works such as [7] have shown that through extensive empirical testing, accurate estimations of a path loss exponent can be determined for specific environments.

	n	σ(dB)	Number of Locations
All Buildings:			
All Locations	3.14	16.3	634
Same Floor	2.76	12.9	501
Through 1 Floor	4.19	5.1	73
Through 2 Floors	5.04	6.5	30
Through 3 Floors	5.22	6.7	30
Grocery Store	1.81	5.2	89
Retail Store	2.18	8.7	137
Office Building 1:			
Entire Building	3.54	12.8	320
Same Floor	3.27	11.2	238
West Wing 5th Floor	2.68	8.1	104
Central Wing 5th	4.01	4.3	118
West Wing 4th Floor	3.18	4.4	120
Office Building 2:			
Entire Building	4.33	13.3	100
Same Floor	3.25	5.2	37

Figure 2.8: Path-loss exponents n determined through series of empirical data gathering experiments[7]

While this provides useful insights and settings for that specific environment, it does not broadly generalize. For most deployments, it is required that at minimum a series of test measurements be taken and used to estimate a path-loss exponent for the environment of operation. Hence it is critical that the path loss exopnent be properly determined.

CHAPTER 3: System Theory and Design

In this section, the design of the BreadCrumb will be fully explained and analyzed.

3.1 Log-Distance Path Loss for Distance Estimation

Distance estimation from RSSI is heavily utilized in this work, hence the relation between the log-distance path loss model and distance estimation need be explained. The log-distance path loss model typically predicts the power loss between a transmitter and receiver, but the formula can be rearranged to determine the distance between a transmitter and receiver is the power loss is already known. Assuming a reference distance d_0 of 1 meter is used then equation 2.8 can then be re-written as:

$$d = 10^{\frac{P_L(d_0) - P_L}{10 * n}} \tag{3.1}$$

where d is calculated distance between transmitter and receiver, $P_L(d_0)$ is the power loss measured at a reference distance of 1 meter, P_L is the power loss measured at the current distance, and n is the path loss exponent.

3.2 Trilateration

Trilateration in the context of two dimensional wireless localization is the process by which three devices at known locations measure the range to, and determine the location of a fourth device at an unknown location. This is illustrated in Figure 3.1 where D1, D2, and D3 have known locations while D4 has an unknown location.

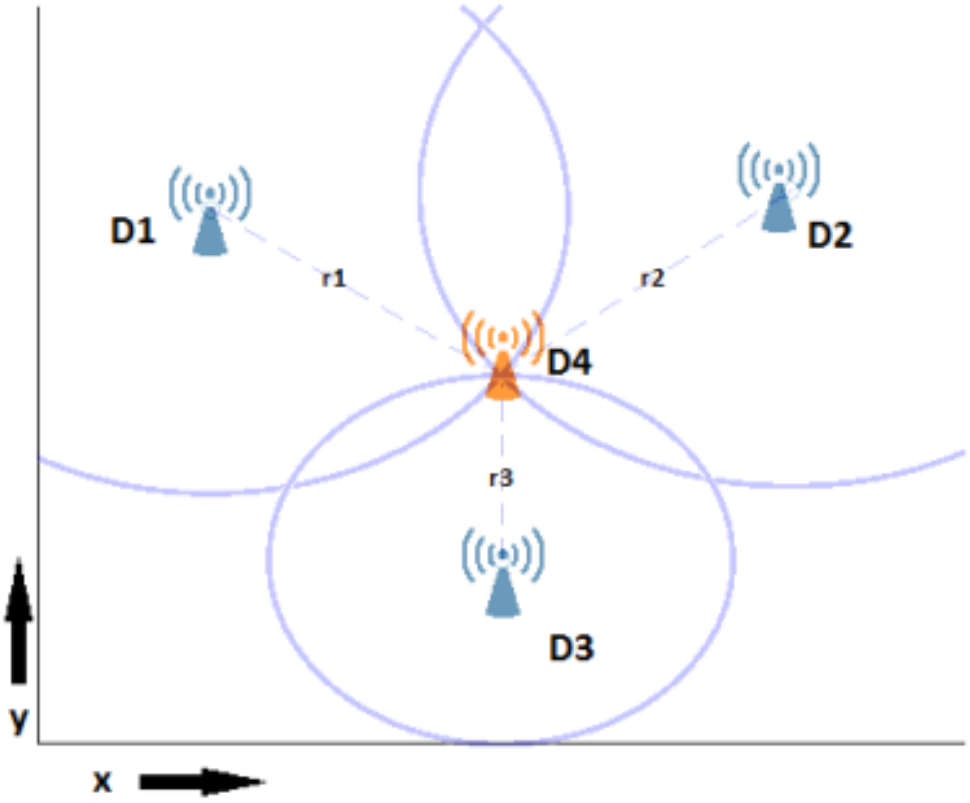


Figure 3.1: Visualization of true range trilateration [8].

The problem can be expressed as a system of equations:

$$\begin{bmatrix} \sqrt{(x_{D_1} - x_{D_4})^2 + (y_{D_1} - y_{D_4})^2} \\ \sqrt{(x_{D_2} - x_{D_4})^2 + (y_{D_2} - y_{D_4})^2} \\ \sqrt{(x_{D_3} - x_{D_4})^2 + (y_{D_3} - y_{D_4})^2} \end{bmatrix} - \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.2}$$

where r_1, r_2, and r_3 are the ranges from D_1, D_2, and D_3 to D_4. Solving this system of equations will result in the x and y location of D_4 on the Cartesian plane presented in Figure 3.1.

However, this formulation utilizes true ranging. This assumes that the range measurements are perfect, with zero error introduced. If error were to be introduced to the ranges r_1, r_2, and r_3. Then a circumstance could arise such as in Figure 3.2 could arise.

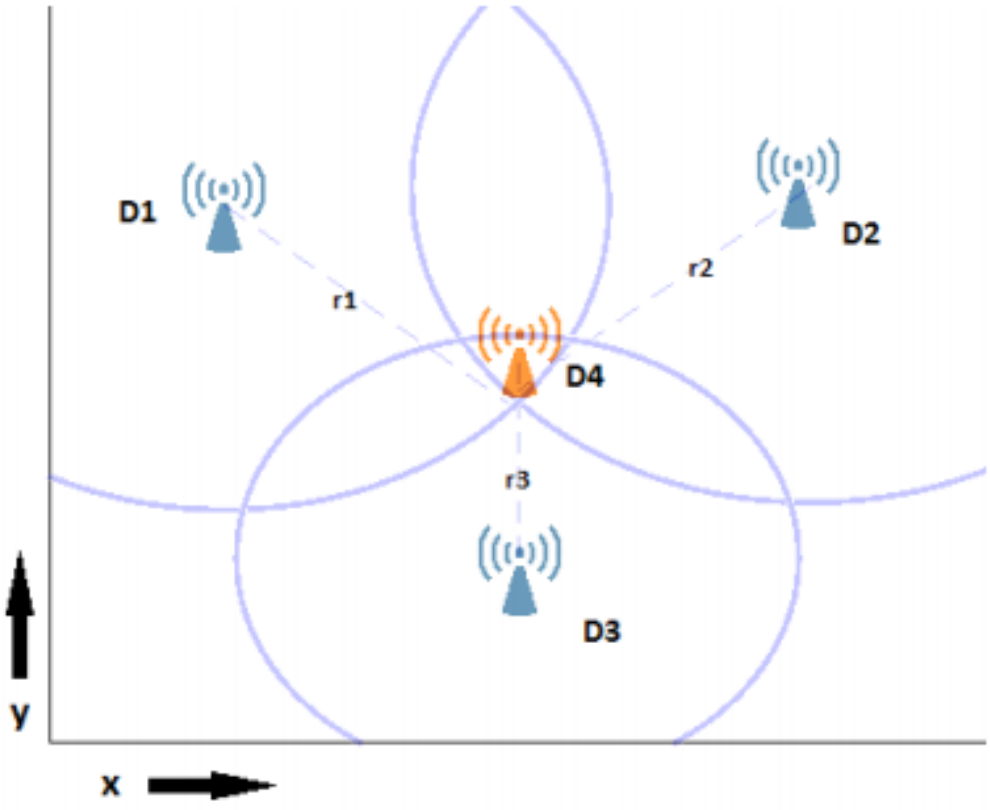


Figure 3.2: No singular solution for trilateration due to noise [8].

As it is extremely unlikely to have consistent perfect ranging measurments in any practical sense, the situation in Figure 3.2 should be considered the norm rather than an exception. Especially considering the previous explanations on fading and sources of signal interference. With that in mind, the system of equations for trilateration can be reformed as:

$$\begin{bmatrix} \sqrt{(x_{D_1} - x_{D_4})^2 + (y_{D_1} - y_{D_4})^2} \\ \sqrt{(x_{D_2} - x_{D_4})^2 + (y_{D_2} - y_{D_4})^2} \\ \sqrt{(x_{D_3} - x_{D_4})^2 + (y_{D_3} - y_{D_4})^2} \end{bmatrix} - \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = E \tag{3.3}$$

Where e_1, e_2, and e_3 represent the error introduced my sources of noise or interference, and E is the vector of these errors. Therefore a method for determining the best possible solution given the data must be utilized. The goal being to determine a

value for x_D4 and y_D4 that minimizes E. To accomplished this, Non-Linear Least Squares is utilized.

3.2.1 Non-Linear Least Squares

Non-linear least squares is used in to minimize the sum of squares in the rror of a function. In this facse the function can be represented as $f_i(x, y)$ where:

$$f_i(x, y) = f_i(\theta) := d_i(\theta) - r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} - r_i \tag{3.4}$$

with i being the i_{th} device with a known location x_i, y_i. r_i is the range measurement from the i_{th} device to the device at an unknown location. The sum of the squared error is then defined as:

$$F(\theta) = F(x, y) = \sum_{i=1}^{n} f_i(x, y)^2 \tag{3.5}$$

where n is the total number of devices with known locations. In order to minimize $F(\theta)$ we need to linearize it. This is first accomplished by the first order partial derivative with respect to x and y which respectively produces:

$$\frac{\partial F(x, y)}{\partial x} = 2 \sum_{i=1}^{n} \frac{\partial f_i(x, y)}{\partial x} = 2 \sum_{i=1}^{n} \frac{\partial d_i(x, y)}{\partial x} \tag{3.6}$$

$$\frac{\partial F(x, y)}{\partial y} = 2 \sum_{i=1}^{n} \frac{\partial f_i(x, y)}{\partial y} = 2 \sum_{i=1}^{n} \frac{\partial d_i(x, y)}{\partial y} \tag{3.7}$$

Next we solve for:

$$\nabla F = 2J^T f = 0 \tag{3.8}$$

With ∇ F as the linearized vector of the partial derivatives, J ,as the Jacobian:

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \vdots & \vdots \\ \frac{\partial f_i}{\partial x} & \frac{\partial f_i}{\partial y} \end{bmatrix} \tag{3.9}$$

where f is the error function:

$$f(x,y) = \begin{bmatrix} f(x,y)_1 \\ f(x,y)_2 \\ \vdots \\ f(x,y)_n \end{bmatrix} \tag{3.10}$$

Which finally yields:

$$\nabla F = 2 \begin{bmatrix} \sum_{i=1}^{n} \frac{(x-x_1)\partial f_i}{\partial x} \\ \sum_{i=1}^{n} \frac{(y-y_1)\partial f_i}{\partial x} \end{bmatrix} \tag{3.11}$$

With a linearized function, we can utilize Newtons method to solve for ∇F.

3.2.2 Newtons Method

Newtons method provides an iterative process that searches for better approximations to the roots of a real function f given as:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{3.12}$$

Where x_0 is an inital guess for the roots of f, and x_1 is an iteration of improved approximation. By dividing the function f with its derivative a tangent line line of the function is produced. The root for this is found from the x intercept of the tangent line. As the tangent x intercept approaches the true x intercept of the function the

approximate roots become more accurate. By iteratively repeating this process with:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{3.13}$$

roots can be determined up to a desired accuracy threshold. However, in the previous derivation Newtons method has not been defined for vectors. Thus, we must extend it such that it can handle the vector formed from trilateration. We begin by defining:

$$x_{n+1} = x_n - [J(x_n)]^{-1} f(x_n) \tag{3.14}$$

with x as our current set of paramters, $f(x_n)$ being the function to be minimized, and $J(x_n)$ being the Jacobian of that function. We've already defined this, hence to minimize for two dimensional trilateration we describe our function as:

$$x_{n+1} = x_n - \left[J(x_n)^T J(x_n)\right]^{-1} J(x_n)^T f(x_n) \tag{3.15}$$

with n as the iteration number, x the set of parameters, f the error function defined previously, and J the Jacobian of f

3.2.3 Converting GPS to local

In some scenarios the location of SB and EB may be given in the form of GPS coordinates, hence they will need to be converted to the BreadCrumbs frame of reference. To accomplish this an equirectangular projection [47] is used. Since the distance between BreadCrumbs is on the order of 10s of meters, the distortion introduced by this projection is negligible for our purposes. The elevation of the BreadCrumbs is assumed to be nearly equal. The projection is accomplished using the following:

$$x = r * Long * cos(\frac{pi * Lat}{180}) * \frac{pi}{180} \tag{3.16}$$

$$y = r * Lat * \frac{pi}{180} \tag{3.17}$$

Where r is the radius of the earth, $Long$ is the longitude, and Lat is the latitude.

3.3 Deep Deterministic Policy Gradients

Deep Deterministic Policy Gradient (DDPG), first introduced by [48], is a reinforcement learning (RL) method that focuses on learning a policy through an endowment interaction reward paradigm. DDPG overcomes the finite state and action space of most RL algorithms, allowing for states and actions in the continuous space to be utilized. Deep Q-learning (DQN) [49] and policy gradients are the two factors responsible for this. What follows is a brief review of DDPG.

Machine learning has been utilized in various areas such as controls [50], computer vision [51] and even signal processing [52]. Reinforcement learning is an extension of this which, at its core, can be determined as a discrete-time Markove Decision Process (MDP) with tuple $M = (S, A, p, r, \gamma)$ where S is the state space, A the action space, p the transition function, r the reward function, and γ the discount factor. A transition function describes the probability of obtaining a next state, given a current state and current action: $P(s_{t+1}|s_t, a_t)$ where t is the current time instance. The reward function r is defined as the reward achieved at a state-action pair: $r(s_t, a_t)$. The goal is to find the optimal policy which maximizes the total discounted reward:

$$J = \mathbb{E}_{r_i, s_i \sim E, a_i \sim \pi} [R_1] \tag{3.18}$$

Where π is the policy and R_t is an episodes cumulative discounted reward:

$$R_t = \sum_{T}^{i=t} \gamma^{i-t} r(s_i, a_i) \tag{3.19}$$

The expected return after taking an action in a current state following policy π is given by the Bellman equation. When the target policy is assumed to be deterministic, as μ, the inner expectation of the equation can be avoided, hence formulated as:

$$Q^{\mu}\left(s_t, a_t\right) = \mathbb{E}_{r_t, s_{t+1} \sim E}\left[r\left(s_t, a_t\right) + \gamma Q^{\mu}\left(s_{t+1}, \mu\left(s_{t+1}\right)\right)\right] \tag{3.20}$$

Furthermore, Q-Learning uses function approximates parameterized by θ^Q which are optimized by minimizing the loss as shown:

$$L\left(\theta^Q\right) = \mathbb{E}_{s_t \sim \rho^{\beta}, a_t \sim \beta, r_t \sim E}\left[\left(Q\left(s_t, a_t \mid \theta^Q\right) - y_t\right)^2\right] \tag{3.21}$$

Where ρ^{β} is the distribution of s_t given the deterministic policy β, where y_t and the variables the deep Q-Network θ^Q are as follows:

$$y_t = r\left(s_t, a_t\right) + \gamma Q\left(s_{t+1}, \mu\left(s_{t+1}\right) \mid \theta^Q\right) \tag{3.22}$$

However, since the action space is continuous, satisfying the greedy policy $\mu(s) = \arg\max\ {}_a Q(s, a)$ requires a_t to be optimized at every time step which is extremely impractical in the extreme. Hence, an actor-critic approached based on the DPG algorithm introduced by [53] is used.

The actor-critic can be divided into the following two features: an actor and a critic. The actor is given the current state of an environment and uses a policy to predict an action to take. The critic also receives the state as well as the reward from the environment from that state. The actors policy is then updated based on the evaluation from the critic.

The actor $\mu(s|\theta_\mu)$ details the current policy via deterministically mapping states to actions. The critic $Q(s, a)$ learns with the Bellman equation is in Q-learning while

the actor is updated as:

$$\nabla_{\theta^\mu} J = \mathbb{E}_{s_t \sim \rho^\beta} \Big[\nabla_a Q\left(s, a \mid \theta^Q\right)\big|_{s=s_t, a=\mu(s_t)} \nabla_{\theta_\mu} \mu\left(s \mid \theta^\mu\right)_{s=s_t}\Big] \quad (3.23)$$

The training of the algorithm then follows, where the action $a_t = \mu(s_t) + n_t$ is generated. A reward and next state are received from that action on the environment and the set is stored in an experience replay buffer $(s_t, a_t, r_t, s_t + 1)$. A random minibatch is selected from the replay buffer based on N which is input to the actor and critic net. The target net of the actor net then outputs an action $\mu'(s_i + 1)$. The target of the critic net can then calculate y_i based on the mini batch and action. The loss is minimized as according to:

$$L = \frac{1}{N} \sum_i \left(y_i - Q\left(s_i, a_i \mid \theta^Q\right)\right)^2 \quad (3.24)$$

Then the actor policy is updated according to the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q\left(s, a \mid \theta^Q\right)\Big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu\left(s \mid \theta^\mu\right)\Big|_{s_i} \quad (3.25)$$

Finally, the target actor and critic network are updated:

$$\begin{aligned} \theta^{Q'} &\leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'} \end{aligned} \quad (3.26)$$

Incorporating a small constant as τ

3.4 Physical BreadCrumb

There are two different version types of the BreadCrumb. Each BreadCrumb from version one is composed of the components listed Table 3.1 and each BreadCrumb from version two is composed of the components listed in Table 3.2.

Table 3.1: BreadCrumbV1 Components List

Amount Per Crumb	Item
1	RaspberryPi Model 2 B+
1	Xbee S2
1	Ceramic GPS with 1575R-A GPS
1	Indicator Panel
1	Solar Rechargeable Battery
1	Modular 3D Printed Case

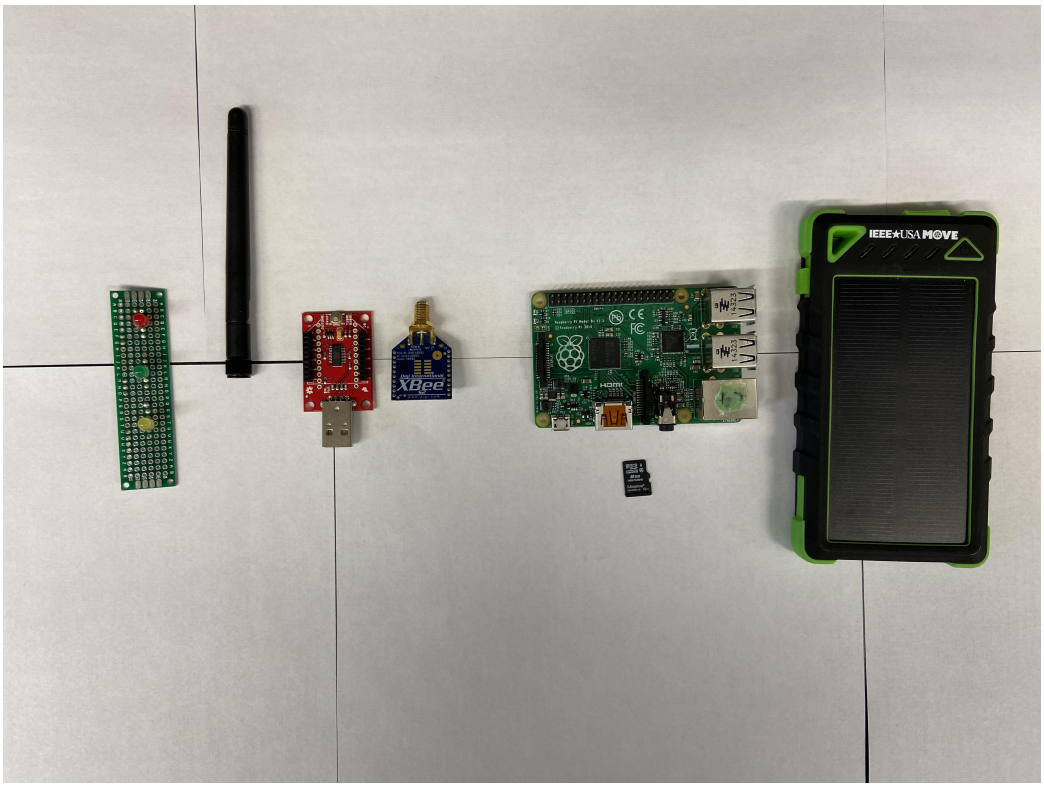
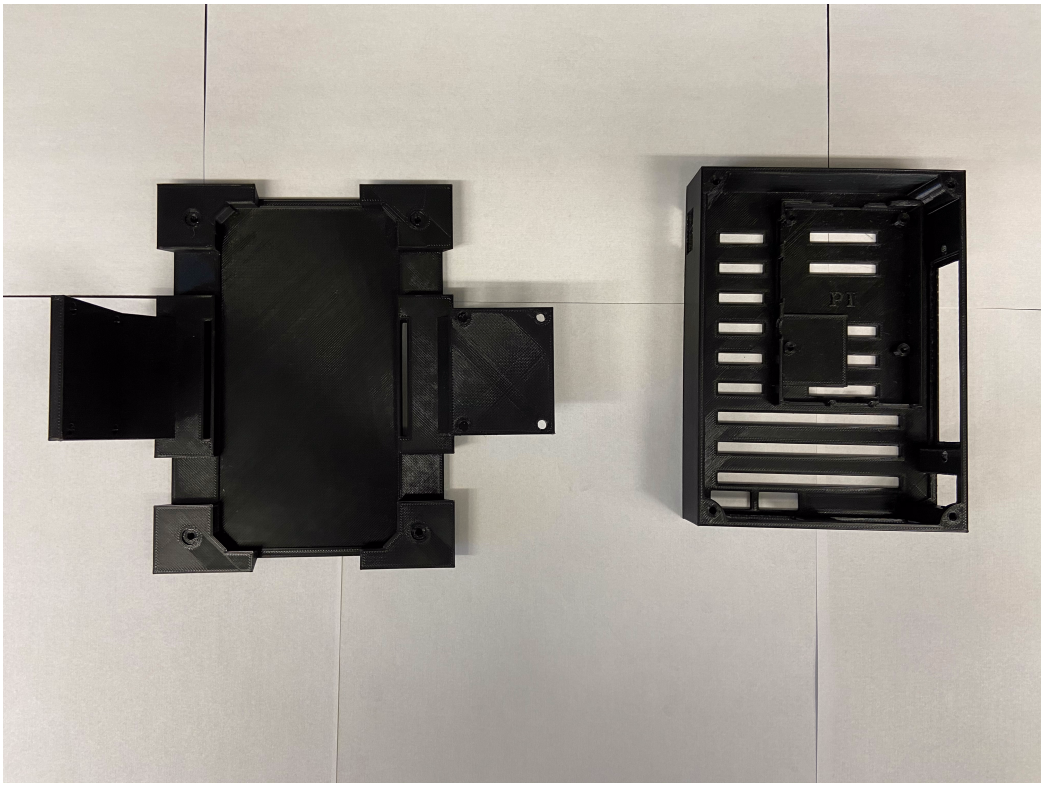


Figure 3.3: BreadCrumb V1 modular case



(a) (b)

Figure 3.4: BreadCrumb V1 Assembled a) Top view, b) Side view

Table 3.2: BreadCrumbV2 Components List

Amount Per Crumb	Item
1	RaspberryPi Model 4
1	Xbee S3
1	RaspberryPi Camera V2
1	Indicator Panel
1	Solar Rechargeable Battery
1	Modular 3D Printed Case
1	3-Axis Compass Magnetometer

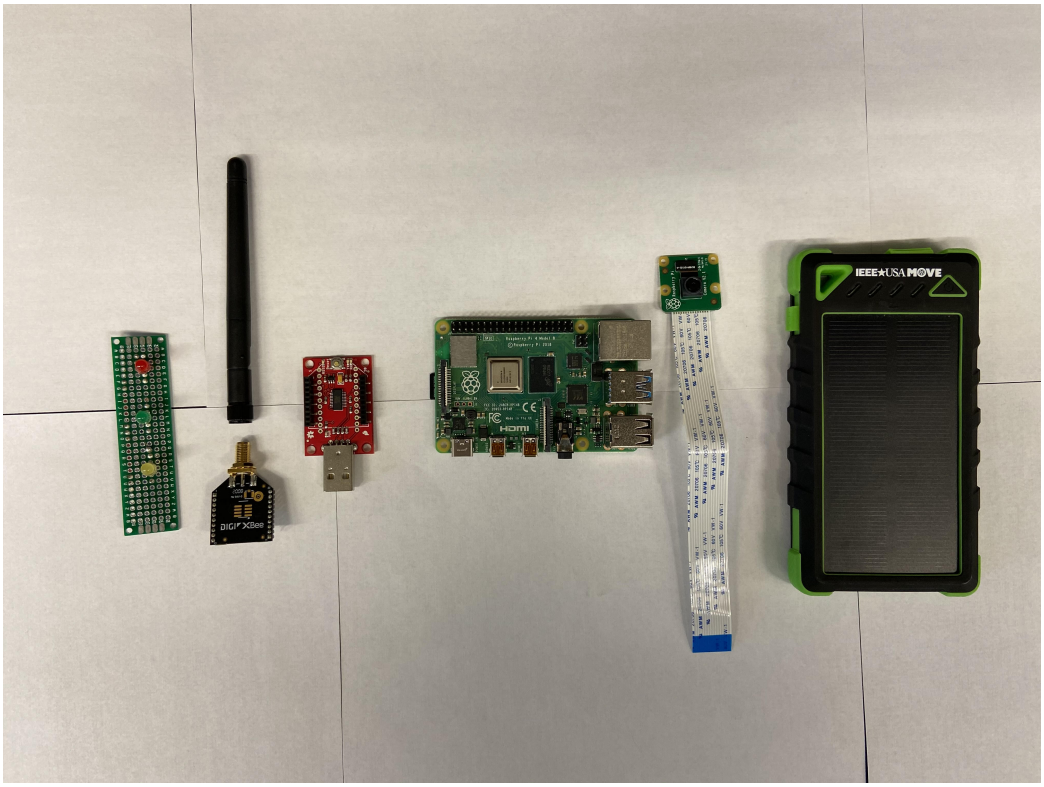


Figure 3.5: BreadCrumb V2 component parts

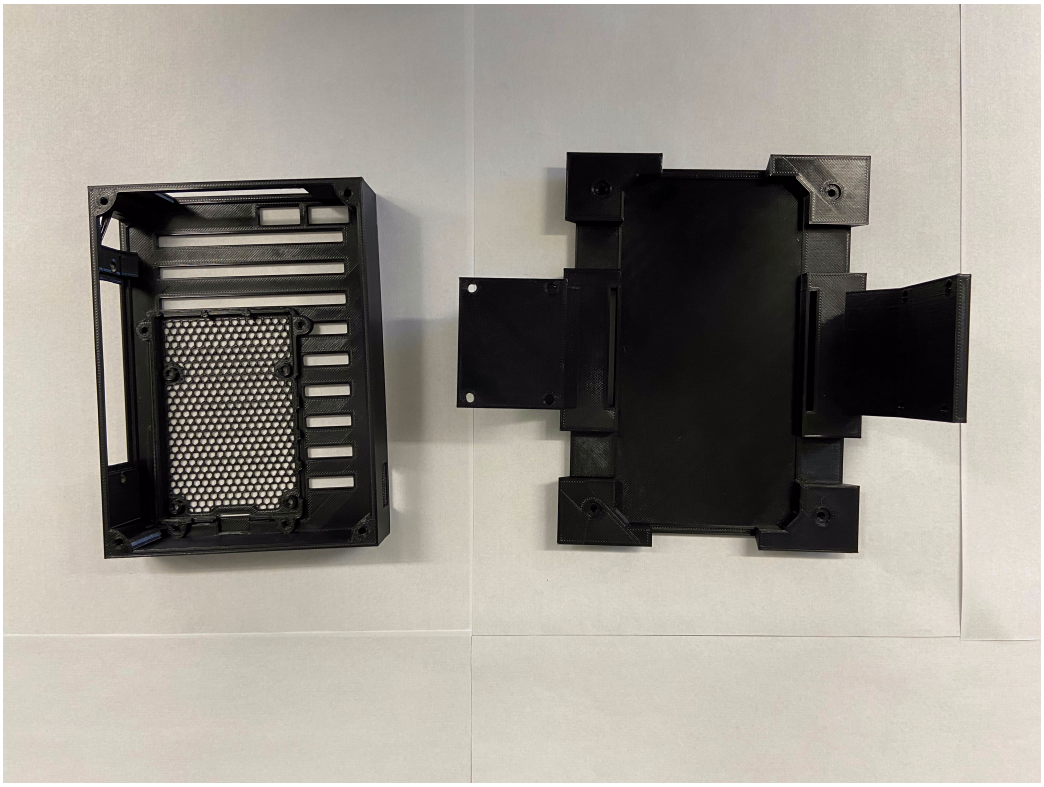


Figure 3.6: BreadCrumb V2 modular case

3.5 Concept of Operations

This section details the concept of operations for this BreadCrumb. It covers how the network is formed, the operational phases of the BreadCrumb, and the communication/command protocol the BreadCrumbs use. The basic ConOps follows what is shown in Figure 3.7.

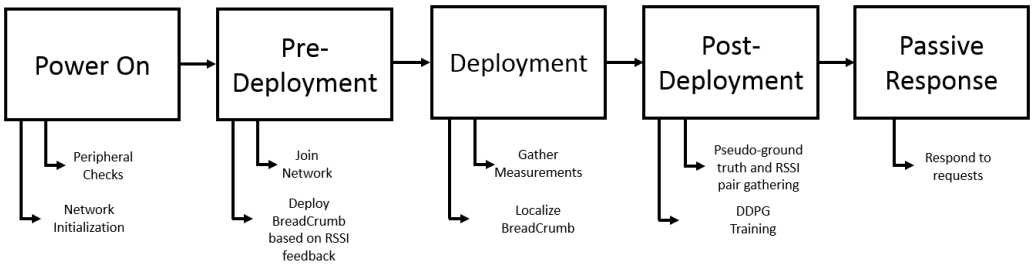


Figure 3.7: Basic ConOps of BreadCrumbs

The main phases are: power on, pre-deployment, deployment, post-deployment, passive response. Several of these phases break down into sub-phases which will be

further described through the next sections.

3.5.1 Power On

3.5.1.1 BreadCrumb Coordinator

The BreadCrumbs communicate using Xbee radios. These Xbee radios follow the Zigbee protocols. The wireless network over which the BreadCrumbs communicate must be established by a network coordinator. The network coordinator is a unique BreadCrumb that only exists at the start of the BreadCrumb trail. It is responsible for determining an appropriate traffic-free channel for the network to operate over, selecting the ID of the network, and authenticating and distributing network keys to nodes as they join the network. The network coordinator BreadCrumb must be the first BreadCrumb powered on in the trail. All other BreadCrumbs simply join this network once powered on and have no special network formation functions. The BreadCrumb acting as the network coordinator also can utilize a 2 inch LCD screen that displays information on the network which can be used for troubleshooting. All but one Xbees are configured using Zigbee Router API. A particularly important setting is the broadcast hops. When the Xbee mounted to the robot is broadcasting range data, steps must be taken to ensure that the range readings which are specific to each device do on hop along the Xbee chain. The single Xbee used as the coordinator is configured with the Zigbee Coordinator API.

3.5.1.2 All Other BreadCrumbs

All other BreadCrumbs once powered on join the network established by the coordinator. They then run a series of system checks to ensure proper operational capacity. Of these system checks the following are critical and will cause a system restart if failed.

1. No Xbee radio detected

2. No Zigbee network found

3. Zigbee network found, but could not join network

4. If BreadCrumb V2 and no RaspberryPi camera detected

Other noncritical checks include flashing the display lights so an operator can check for burnout, and checking that directories for data and measurements have been properly established. If the proper directories are not established then additional time is allocated to create data folders. After successfully connecting to the network and completing the peripheral checks, the green LED indicator will flash three time to indicate that power on was successful.

The Power On phase of a BreadCrumb should be performed after the previous BreadCrumb in the trail has entered the deployment phase. In other words, there should never be a time when two BreadCrumbs are simultaneously in the power on or pre-deployment phase. The Power On phase of the current BreadCrumb should take place adjacent to the most recently deployed BreadCrumb to ensure the Zigbee network is discovered and joined as quickly as possible.

3.5.2 Pre-Deployment

Once a BreadCrumb is powered on and connected to the network, it requests the deployment number of its nearest neighbors. The deployment number numerically represents the order the BreadCrumbs were deployed in, with the coordinator having a deployment number of zero. It then sets its deployment number equal to the highest deployment number returned plus one. Since the current BreadCrumb was powered on next to the previous BreadCrumb, the current BreadCrumb should end up with the highest deployment number of the network at this state. The deployment number is used as an identifier for the BreadCrumb.

Next, the current BreadCrumb begins to request received signal strength indicator (RSSI) readings from the previous BreadCrumb at a rate of one Hz. The RSSI reading is used to indicate an reasonable deployment range for the device. The goal being

to maximize the range of the BreadCrumb network while maintaining enough signal integrity for reliable communication.

RSSI readings with a signal strength greater than -50 dB are considered to be too close to the previous BreadCrumb, and cause the yellow LED to light up. This indicates that the current location is fit for deployment, but not optimal as more distance could be covered while maintaining signal integrity. RSSI readings between -50 and -80 dB are considered optimal locations for deployment and cause the green LED to light up. RSSI readings less than -80 dB are considered to be signal risk distances. While signal strength for communication may be mainted for further distance, the risk of losing the signal entirely and forcing the user to back track is deemed to great to increase the limit . This causes the red LED to light up indicating that the BreadCrumb should not be deployed here. Once a location has been determined the current BreadCrumb is put down. Then a switch on the current BreadCrumb is flipped to in-service, indicating that the BreadCrumb has been deployed. The current BreadCrumb sends its deployment number to its nearest neighbors, those neighbors continue to chain the information back to the coordinator. informing them that another BreadCrumb has been deployed. The current BreadCrumb then gathers a series of 20 RSSI measurements from the previous BreadCrumb. These will be used for future localization calculations. If this BreadCrumb is to be the final of the network, then after the initial switch flip, the switch should be flipped back to the out-of-service position. This will trigger a broadcast that informs the Coordinator the network is complete. Note, the coordinator BreadCrumb does not participate in this phase.

3.5.3 Deployment

This phase is broken down into separately for the current BreadCrumb activated and the coordinator BreadCrumb.

3.5.3.1 Current BreadCrumb

During the deployment phase, the current BreadCrumb waits to receive a deployed signal from the next BreadCrumb. Once received, it requests 20 RSSI measurments from the next BreadCrumb. Once gathered, gathered, the current BreadCrumb will send the 20 measurements to the previous and next BreadCrumb down the chain back to the coordinator. The current BreadCrumb then waits and responds to other BreadCrumb requests until it receives a location update from the coordinator. Once the current BreadCrumb receives its location, it moves onto the post-deployment phase.

3.5.3.2 Coordinator BreadCrumb

During the deployment phase, the Coordinator BreadCrumb is responsible for localizing the deployed network. Depending on conditions this can be performed a number of ways. The conditions breakdown in the flowchart below. To further understand this flow chart, the following scenario is provided. An autonomous supply train is expected to deliver materials to a staging ground in an un-mapped forest with poor GPS reliability. The staging ground location has been determined. Human personal have been dispatched to secure the grounds and deploy the BreadCrumb network. There is no time constraint with this work. The robot begins operating once the trail complete flag is raised the robot begins operating.

Next, we will walk through how this process unfolds from the coordinators perspective. The starting BreadCrumb (SB) has been deployed at a known location. The ending BreadCrumb has not yet been deployed but the location where it will be deployed is known. This is visually represented in Figure 3.9.

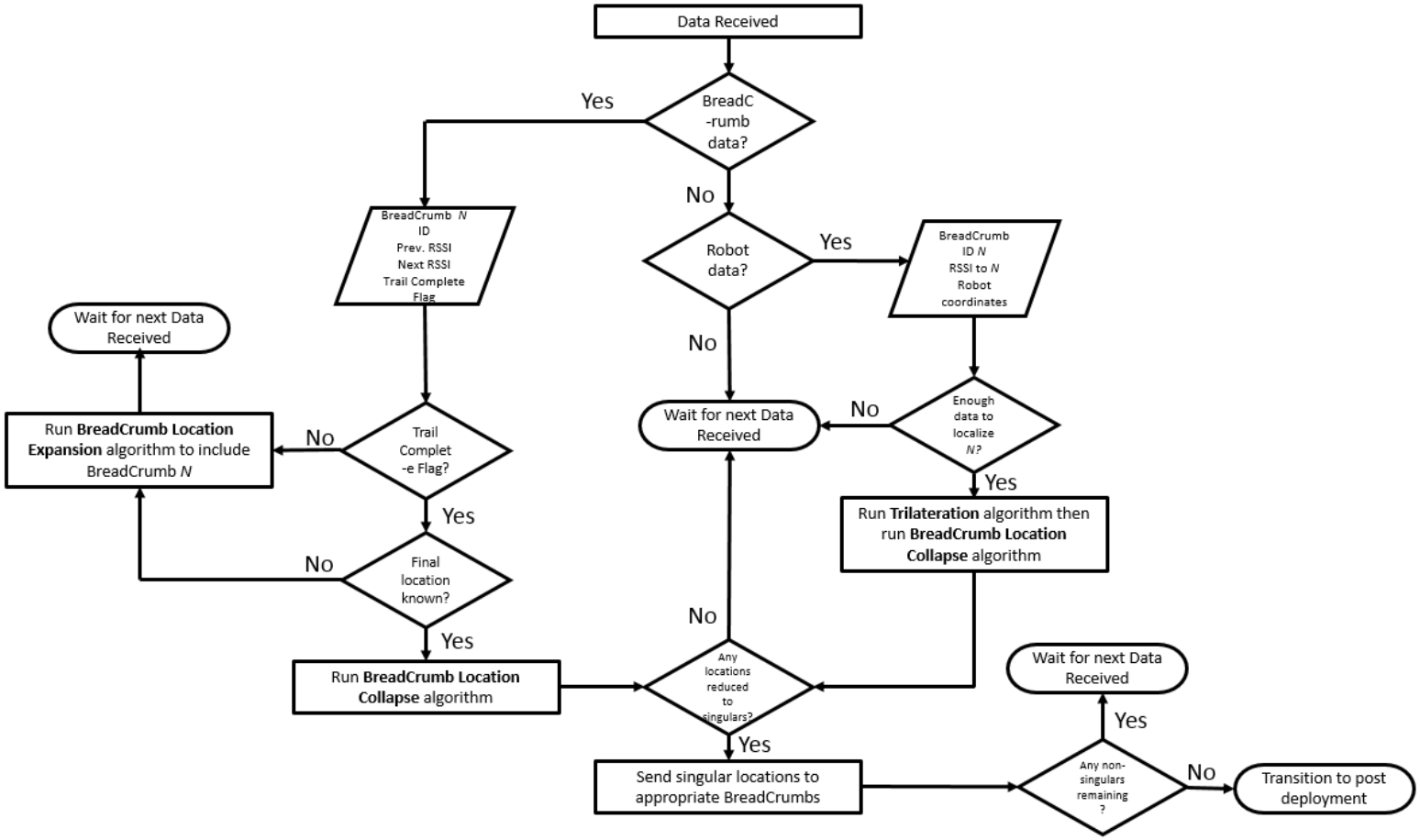


Figure 3.8: Flow chart depicting the process the Coordinator undergoes as it is localizing the network.

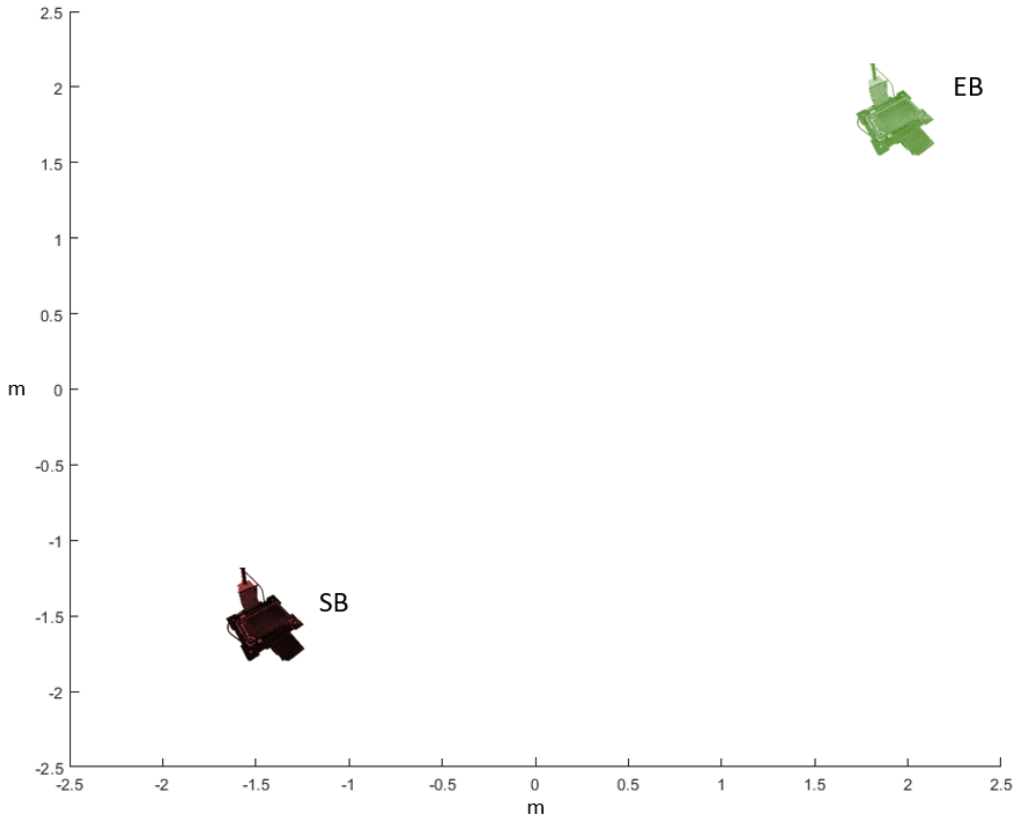


Figure 3.9: Perspective of starting BreadCrumb when location of deployed starting and undeployed ending BreadCrumb (SB and EB) is known.

Next a BreadCrumb labeled $UB1$ is powered on, goes through the pre-deployment phase, and is deployed. It gathers 20 RSSI measurements to SB. Another BreadCrumb labeled $UB2$ is powered on, goes through the pre-deployment phase, and is deployed. It gathers 20 RSSI measurements to $UB1$. This also causes $UB1$ to gather 20 RSSI measurments to $UB2$. Once $UB1$ has gathered its measurements it sends the following information to the coordinator: BreadCrumb Deployment Number, RSSI measurements to previous BreadCrumb, RSSI measurements to next BreadCrumb. This begins the flowchart in Figure 3.8. The data received is determined to be BreadCrumb data with the affor mentioned information. As $UB1$ is not the final BreadCrumb of the trail the trail complete flag has not be set. Thus, the BreadCrumb Location Expansion algorithm is run.

Algorithm 1: BreadCrumb Location Expansion

1: Uniform Circle Point Generation:
2: $th = 0 : \frac{\pi}{N} : 2 * \pi$
3: $x_{circ} = r * \cos th$
4: $y_{circ} = r * \sin th$
5: Expansion:
6: LocList=concate{LocList,zeros(N),$ID + 1$}
7: **for** ii=1:*length* (LocList{ID}):
8: LocList{ID+1}(ii)=LocList{ID}+$\{x_{circ}, y_{circ}\}$
9: **return** *LocList*

In algorithm 1, N is the number of points to generate uniformly around a circle with radius r as determined using the log-distance path loss model from the RSSI value sent by $UB1$. x_{circ} and y_{circ} are those points. *LocList* is the current list of all possible BreadCrumb locations. It has a number of dimensions equal to the total number of BreadCrumbs which have been deployed and sent measurement data. Whenever a new BreadCrumb sends measurment data, an additional dimension is concatened to the *LocList* of size N. Every element of LocList is a cell array containing a possible x,y coordinate of a BreadCrumb. Each dimension of LocList corresponds to an individual BreadCrumb with respect to the location of the previous BreadCrumb denoted by the layer above the current BreadCrumb. This is determined via the deployment number of ID that is encoded in the BreadCrumbs data measurement. At this point LocList will only contain the location of the BreadCrumb coordinator and making the number of dimensions equal to 1. After the Location Expansion Algorithm is run with the data from $UB1$, the size of LocList will expand to $1xN$. Where each element of the second dimension is an theoertical location from $UB1$ may be located with respect to the coordinator. The visualization of this process can be seen in Figure 3.10:

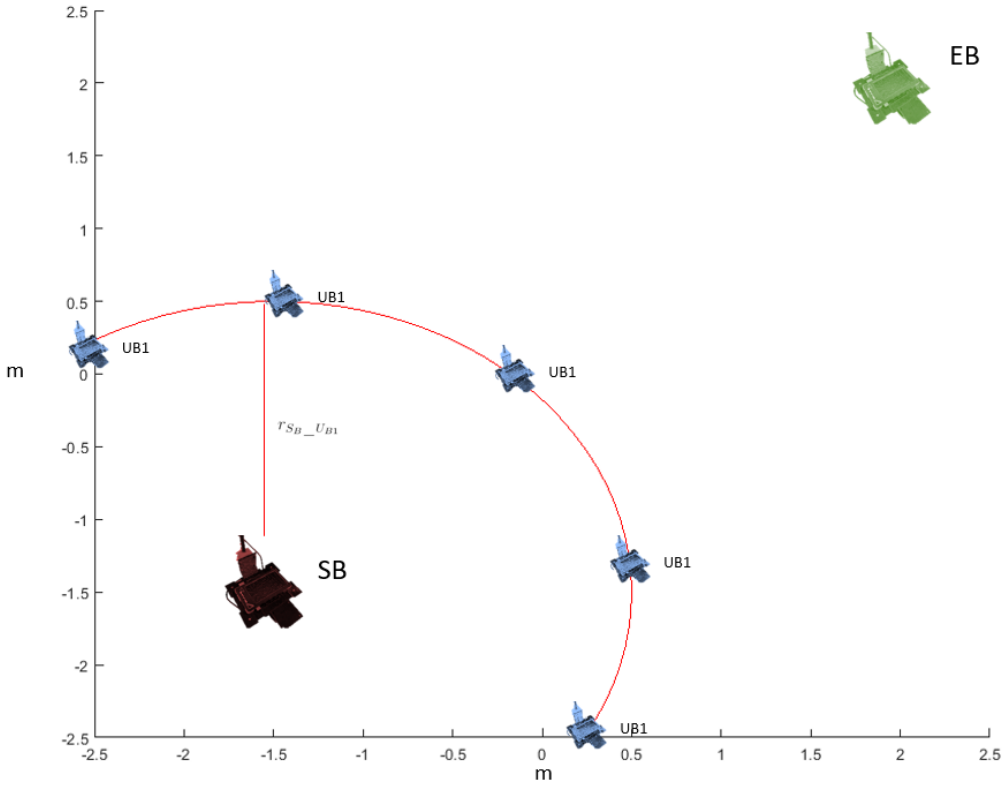
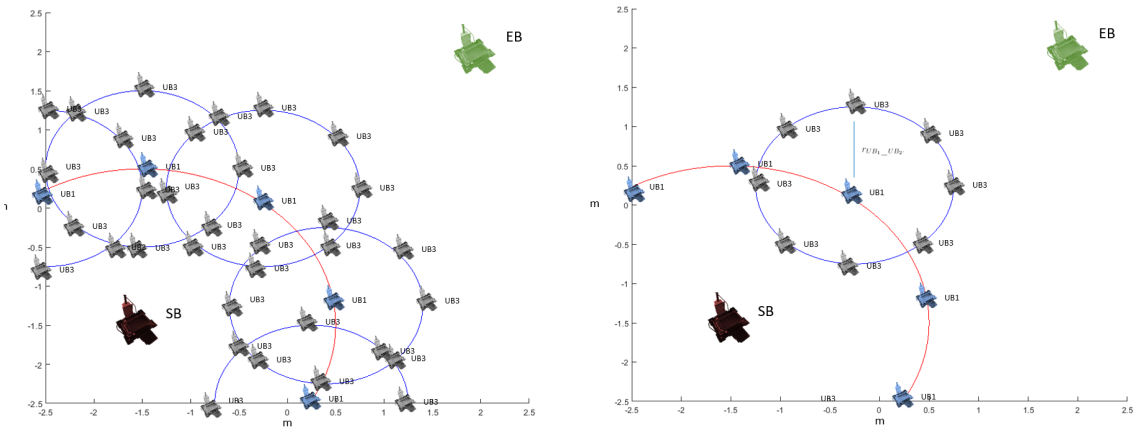


Figure 3.10: n theoretical locations of $UB1$ along the circumference of the circle generated by $r_{S_B\_U_{B1}}$.

Where each instance of $UB1$ is a potential x, y location currently stored in the second dimension of LocList with respects to SB. Once the BreadCrumb at the end of the trail is deployed a similar process will happen with $UB2$. LocList will be expanded to be dimension $1xNxN$ where each element of the third dimension corresponds to a potential location of $UB2$ with respect to its index in the second dimension, which corresponds to each theoretical instance of $UB1$. This process is visualized in Figure 3.14



(a) (b)

Figure 3.11: (a) n theoretical locations of UB_2 along the circumference of the circle generated by $r_{UB_1\_UB_2}$ for each theoretical instance of UB_1 (b) n theoretical locations of UB_2 along the circumference of the circle generated by $r_{UB_1\_UB_2}$ for a single theoretical instance of UB_1

Since this can quickly become unwieldy to visualize, we will continue the explanation using 3.14 which only examines a single instance of $UB1$. However, the following will still apply to all instances. At this point, EB is deployed and will broadcast that information back to SB. This will trigger SB to begin the BreadCrumb Location Collapse Algorithm detailed below.

Algorithm 2: BreadCrumb Location Collapse

```
candidateList=[]
for ii=1:length(LocList{EB_ID − 1}
xy = LocList{ID}(ii)
dist_calc = sqrt((xy(1) + EB_x)^2 + (xy(2) + EB_y)^2)
if abs(dist_calc − dist_meas) < 1 then
canidateList=[candidateList;LocList{1:ID-1}]
return candidateList
```

In Algorithm 2 a series of comparisons are done between the calculated distance from theoretical values of the each xy location in the deepest dimension of LocList. This process can be seen in Figure 3.12

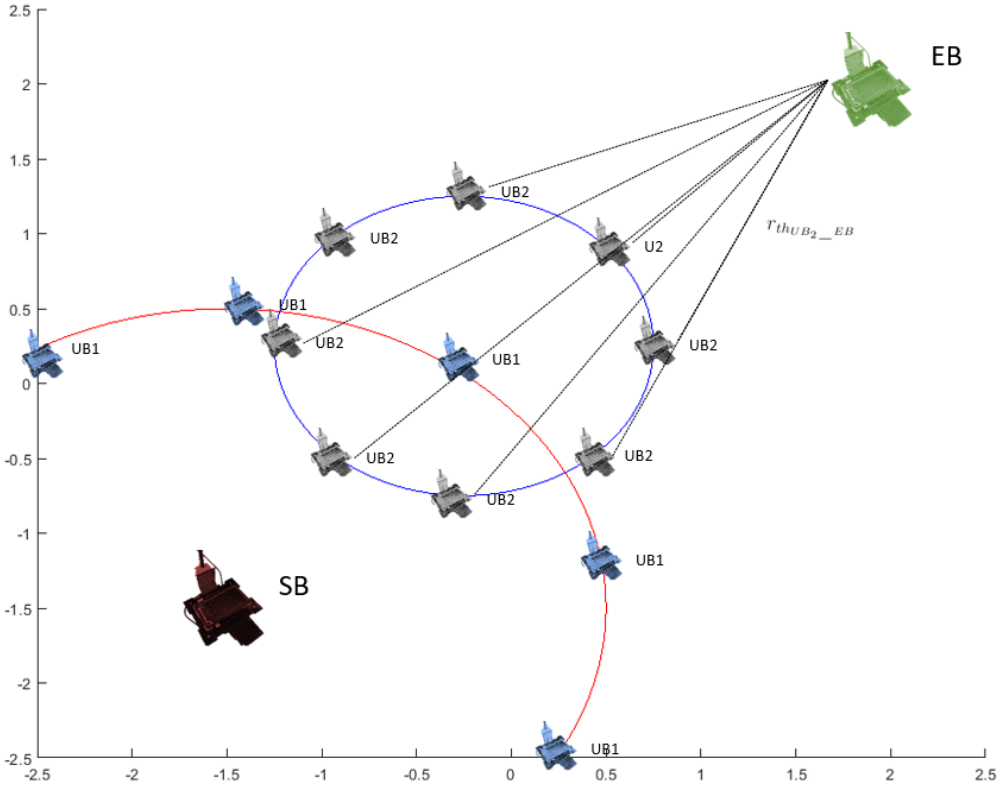


Figure 3.12: $r_{th_{UB_2\text{-}EB}}$ calculated based on the location of each theoretical instance of UB_2.

Those calculated distances are then compared to the measured distance from the end deployed BreadCrumb and the BreadCrumb deployed before it. If the difference between the measured and calculated distance is less than an arbitrary threshold, one meter in this case, than that instance of $UB2$ and its parent instance of $UB1$ are considered candidates. For example, if $UB2$ at LocList1,4,7 meets this criteria, then both 4th instance of $UB1$ and the 7th instance of $UB2$ spawning from the fourth instance of $UB2$ will be considered candidates. The process is visualized in Figure 3.13

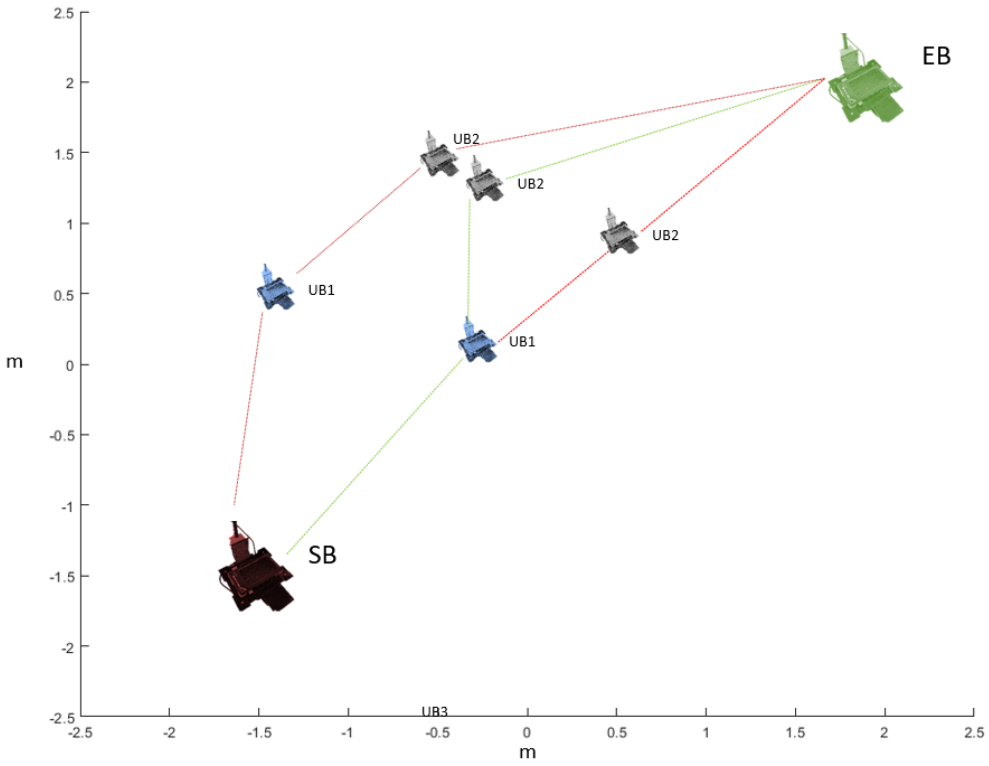
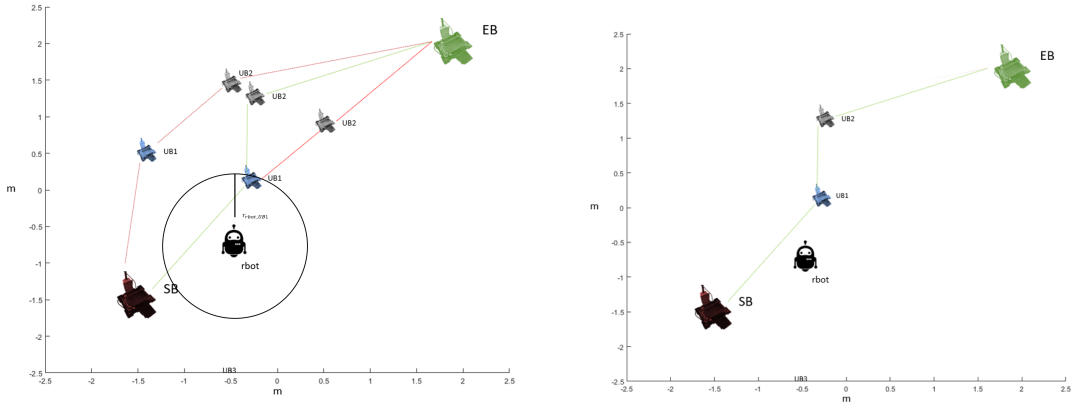


Figure 3.13: Candidate locations for UB_1 and UB_2. The candidates connected by the path of green lines are closet to the acutal location of the BreadCrumbs.

At this point the possible locations for $UB1$ and $UB2$ have been reduced down to two and three locations respectively. For this example it is enough to allow the robot to begin path planning and operations. As the robot being to follow its path, we assume it is running its own localization algorithm and is generating estimates for its location. As the robot comes into range of $UB1$ it gets a range measurement to $UB1$ and sends it to the coordinator SB with the robots current location. The trilateration and BreadCrumb Location Collapse algorithm are both triggered according to the flow chart. However, not enough information is available for the trilateration algorithm at this point, so it simply fizzles. In this iteration of the BreadCrumb Location Collapse algorithm, EB_x and EB_y is replaced with $rbot_x$ and $rbot_y$. Figure 3.14 a) visualizes this process with the final result being Figure 3.14 b).



(a) (b)

Figure 3.14: (a) Robot triggering the BreadCrumb Location Collapse algorithm by providing the coordinate with range measurement from the robot the $UB1$ (b) Final resulting locations of $UB1$ and $UB2$

Now that all BreadCrumbs have been reduced down to having singular x, y coordinate values, this concludes the example of a deployment when the starting and ending location are not. However, this raises the question, what if the ending location was not known. This could be a very common scenario when a broad objective is given, such as locate an individual in an unexplored cave, where the precise destination is not known ahead of time. In that instance, only the location of the starting BreadCrumb SB would be known. This would look similar to Figure 3.16

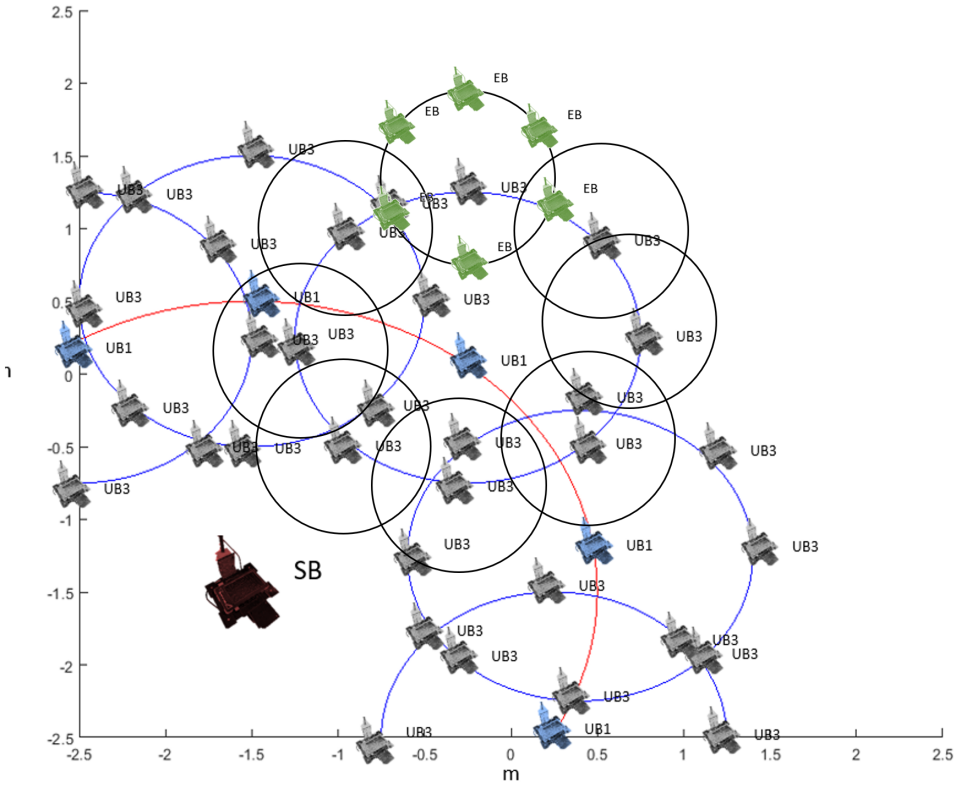


Figure 3.15: Theoretical locations of $UB1$, $UB2$, and EB determined through Bread-Crumb Location Expansion algorithm. Note that for visual clarity not all instances of EB were included.

In this scenario, once the robot was deployed near SB it would begin to get range measurements to $UB1$. From the robots perspective it would appear as in Figure

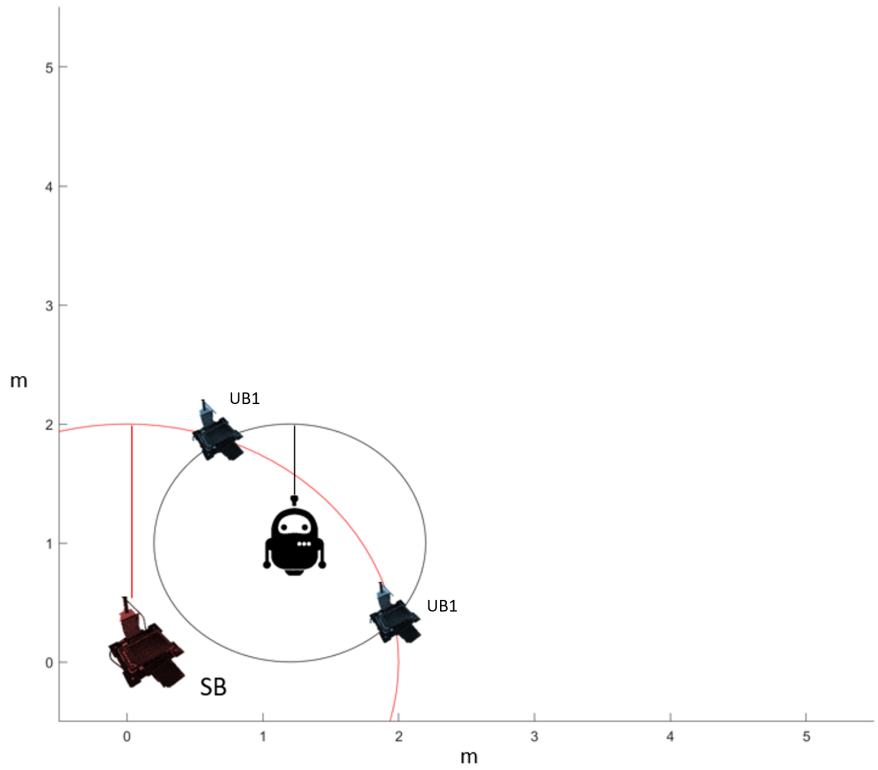


Figure 3.16: Range measurements from robot and SB to $UB1$ at unknown location.

There are two options in this scenario. Either SB can use the BreadCrumb Location Collapse algorithm or from the robots perspective a circle-circle intersection can be solved such that the two visualized locations of $UB1$ are determined. Those two locations can then be fed to the BreadCrumb Location Collapse algorithm. The circle-circle intersection can be solved with Algorithm 3

Algorithm 3: Circ-Circ Intersection

1: $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
2: $l = \frac{r_1^2 - r_2^2 + d^2}{2d}$
3: $h = \sqrt{r_1^2 - l^2}$
4: $x = \frac{l}{d}(x_2 - x_1) \pm \frac{h}{d}(y_2 - y_1) + x_1$
5: $y = \frac{l}{d}(y_2 - y_1) \pm \frac{h}{d}(x_2 - x_1) + y_1$
6: **return** [x,y]

Where x_1, y_1 and x_2, y_2, are objects with known locations, r_1 and r_2 are the ranges from those locations to an object at an unknown location, and x, y are the pair of

solutions for where that point might be. Hence, when the robot moves to another location and gets another range measurement to $UB1$ the possible locations can be reduced to a singular either through trilateration or the BreadCrumb Collapse Algorithm, with the solution appearing as in Figure 3.17

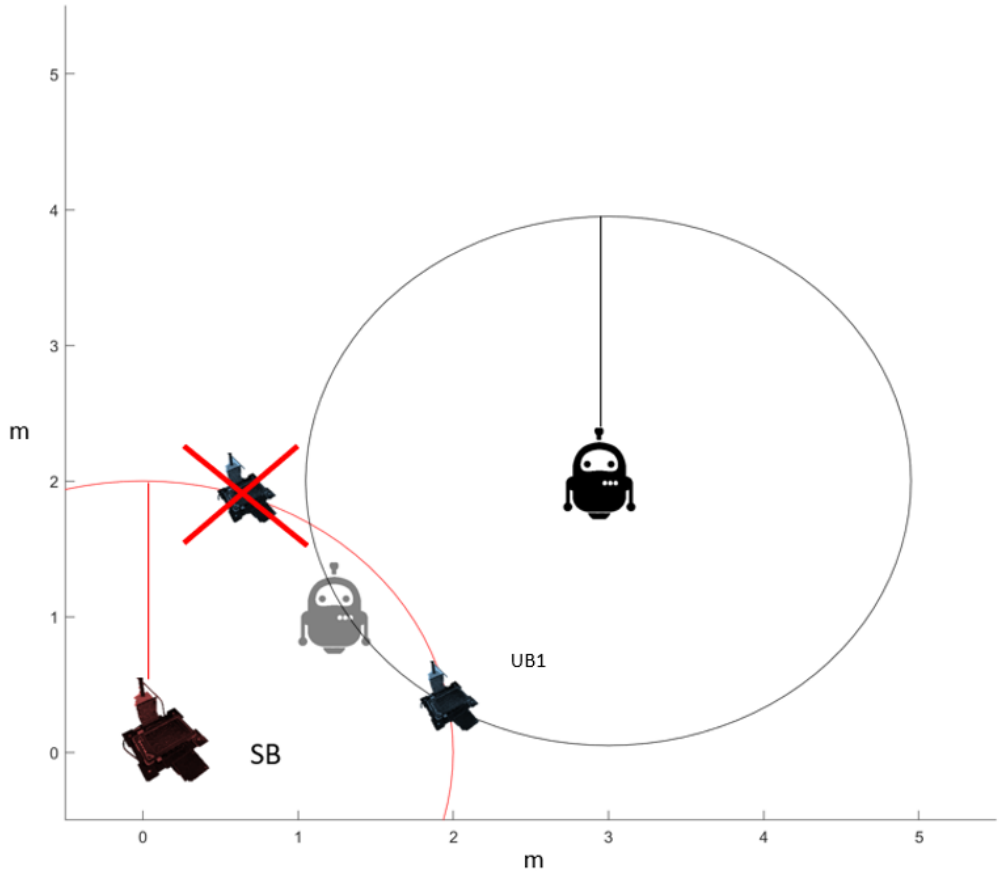


Figure 3.17: Range measurements from robot $UB1$ reducing possible of locations of $UB1$ to a single location.

This brings the deployment phase to an end. At this point, all BreadCrumbs should be localized, thus transitioning them to the Post-Deployment Phase.

3.5.4 Post-Deployment

During the Post-Deployment phase, all BreadCrumb_V2 begin to gather visual location measurment/RSSI range measurement pairs. The visual location measurment is gathered by periodically taking pictures with the raspberrypi camera. The mobile

robot is equipped with an ArUco tag. This is a popular fiducial marker used for finding correspondences between 2D images and real world environments [54]. Utilizing OpenCV we are able to quickly and efficiently determine the local translation vector from the camera to the ArUco tag. When a BreadCrumb observes an ArUco tag it immediately requests and RSSI measurement from the BreadCrumb on the robot, termed the mobile BreadCrumb. The values are then saved together and applied to our Deep Deterministic Policy Gradient algorithm once a certain threshold of visual location measurment/RSSI range measurements have been gathered. This is done to learn an appropriate path loss exponent to be applied to the log-distance path loss model based on the robots location around the BreadCrumb. However, if the BreadCrumb already has line of site to the robot, this may seems unhelpful. However, since the BreadCrumbs have already been localized, we can transform the visual location measurement from the BreadCrumbs local frame to the global frame. Then, any BreadCrumbs in range of BreadCrumb observing the mobile node can gather an RSSI measurement to the mobile node, and use the global coordinates of the mobile node to determine their local distance to said node. Figure 3.18 visualizes this.

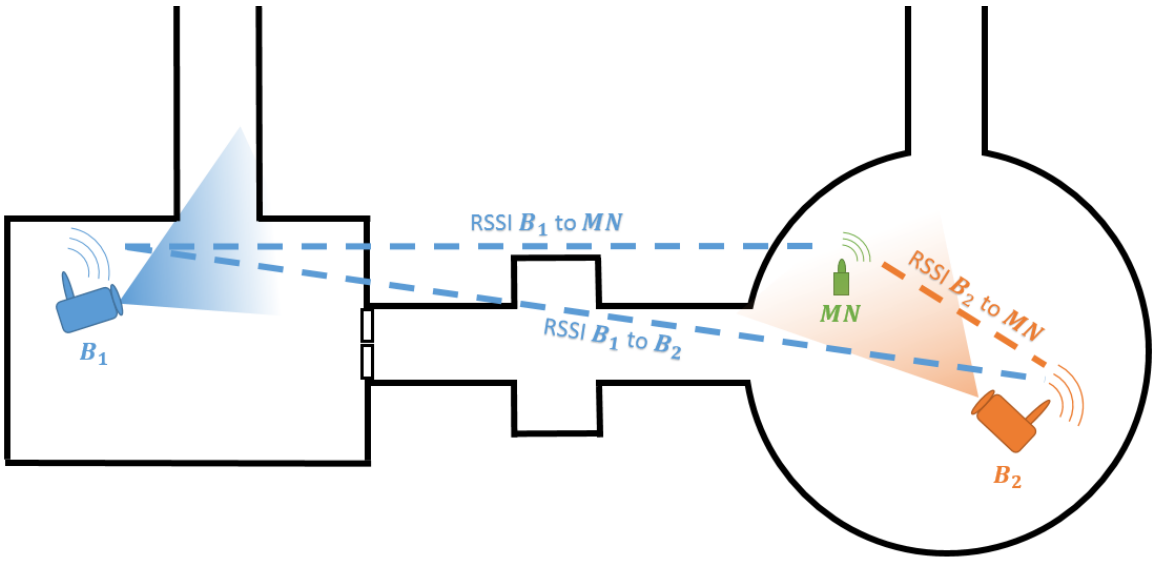


Figure 3.18: BreadCrumb B_2 visually observes MN, gathers a visual location measurment, transforms that measurement to the global frame, and sends it to $B1$ who can gather an visual location measurment/RSSI measurement pair to the mobile node without direct line of sight.

Thus, several BreadCrumbs can gather data even when the mobile node is only visually observed by a single BreadCrumb. This information is fed to the Deep Determinstic Policy Gradient which is detailed below:

3.5.4.1 Agent

The agent is composed of two parts. The first is the physical or simulated manifestation of the agent which performs actions. The manifestation in this case is represented by the augmentation made to the path loss exponent when calculating the range from the RSSI of a transmission made by the Xbee radio. The second part, in this application, is the neural network which is responsible for learning actions based on the signal transmissions interactions in the environment. The agent can be broken down into an actor and critic network as described below:

3.5.4.2 Actor and Critic Network

The actor and critic network are both composed of a series of network layers. In the case of the actor, the network was structured as seen in Figure 3.19. The critic

network was constructed as seen in Figure 3.20.

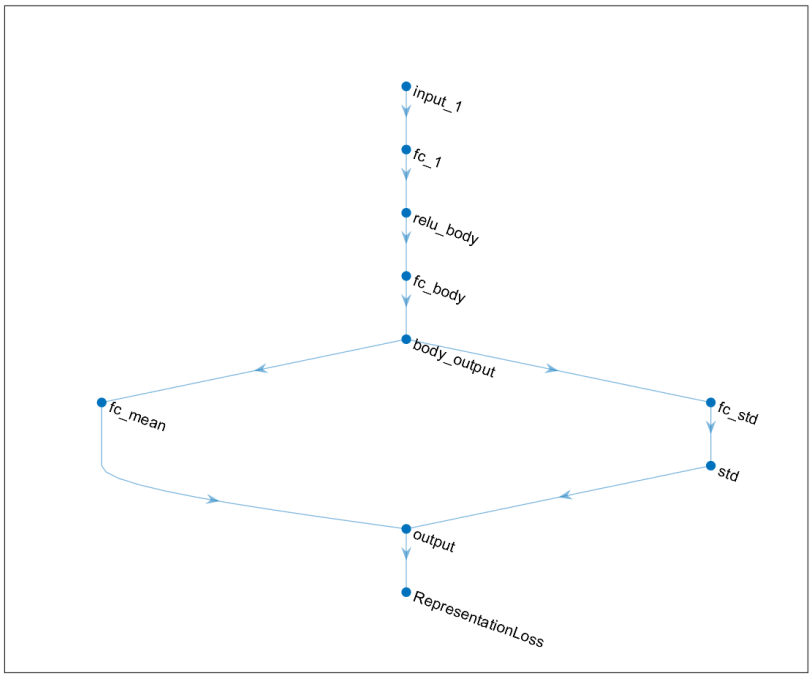


Figure 3.19: Actor network layout

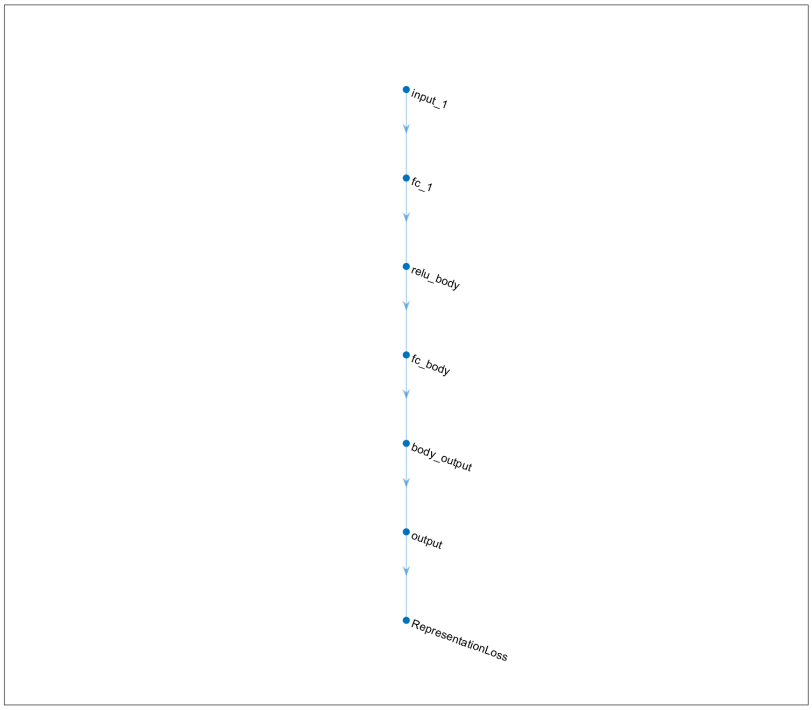


Figure 3.20: Critic network layout

The actor network layer details can be seen in Table 3.3 while the critic network layer details can be seen in Table 3.4.

Table 3.3: Actor Network Layers Details

Layer Name	Value
Feature Input	3 input features s_t
Fully Connected	256 unit
ReLU	ReLU
Fully Connected	256 unit
ReLU	ReLU
Fully Connected	256 unit
Tanh	Hyperbolic Tangent
Scaling Layer	Scaling Layer
Regression Output	Mean Squared Error

Table 3.4: Critic Network Layers Details

Layer Name	Value
Input1	3 feature observation input
Fully Connected 1	256 unit
Input2	1 feature action input
Fully Connected2	256 unit
Concatenation	Concatenation of Fully Connected1 and Fully Connected2
ReLU	ReLU
Fully Connected	256 unit
ReLU	ReLU
Fully Connected	1 unit
Regression Ouput	Mean Squared Error

The actor network training options can be seen in Table 3.5 and the critic network training options can be seen in Table 3.6

Table 3.5: Actor Network Options

Name	Value
Learn Rate	0.005
Gradient Threshold Method	l2norm
L2RegularizationFactor	1.0000e-04
Optimizer	Adam

Table 3.6: Critic Network Options

Name	Value
Learn Rate	0.001
Gradient Threshold Method	l2norm
L2RegularizationFactor	1.0000e-04
Optimizer	Adam

3.5.4.3 The Environment

The environment is the general term for the physical or simulated areas where the agent operates. In this work, the environment is physically outdoor woodland locations or indoor office space. Signals are transmitted by Xbee radios, those signals interact with the various elements of their environment and are received by other Xbee radios. Each Xbee has a local environment within its signal transmission range which is what is considered its environment in terms of the reinforcement learning algorithm.

3.5.4.4 The State

The state is a description of the variables which denote points of interest within our environment. In this case the state is described by the x and y coordinates of the autonomous robot on a Cartesian plane as provided by the localization of that robot and the current RSSI measurement received from the robot to the BreadCrumb at that location.

$$s = \{x_{bot}, y_{bot}, RSSI_{bot}\} \tag{3.27}$$

3.5.4.5 Action

This describes the possible actions that will effect the state as the agent operates in the environment. The possible actions include only changing the pass-loss exponent within the range of two to four.

$$a = \{PLE\} \tag{3.28}$$

3.5.4.6 Reward

The reward function is dictated by the euclidean distance between the final predicted range and the actual range as determined by the camera mounted on the BreadCrumb that observes the robot. The agent is incrementally rewarded as the predicted and actual range converge, and given an large reward when the difference is less than 0.05 meters. The reward function can be defined as:

$$r(s,a) = \begin{cases} -1 & |d| > 2 \\ 2 - d & |d| > 0.05 \text{ and } |d| < 2 \\ 10 & |d| < 0.05 \end{cases} \tag{3.29}$$

Where $r(s,a)$ is the reward of the action taken in a given state, and d is the euclidean distance between the visual location measurment and calculated range with the PLE selected by the action.

3.5.5 Passive Response

Now that the BreadCrumb knows its current location it only provides two functions. The first is to respond to requests from other BreadCrumbs or the robot. The details of these requests can be as follows:

The second is to enable the RSSI reinforcement learning mitigation algorithm. This algorithm is designed to utilize machine learning to learn the specific RSSI noise

pattern of the local environment the BreadCrumb has been deployed to. Details on the specifics of this algorithm can be seen in the next section.

3.5.6 BreadCrumb Communication Commands

The BreadCrumbs can each request or command other BreadCrumbs to perform certain actions or send information. The autonomous robot utilizing the BreadCrumbs also has access to these commands. When a BreadCrumb receives a communication the first byte of that message will be a command ID. The ID will dictate the format of the rest of the message as well as what functions are required to be taken. The commands are summarized as the following, with more details given below the summary:

Command ID	Command Description
1	request deployment number
2	Request hardware ID
3	Send RSSI measurement to device
4	Request RSSI measurement from device
5	Set device as next BreadCrumb
6	Send location to device
7	Set device as previous BreadCrumb
9	Send ready to transmit flag

The general format of each message is seen in Figure 3.21, however, depending on the specific command message, the format may change.

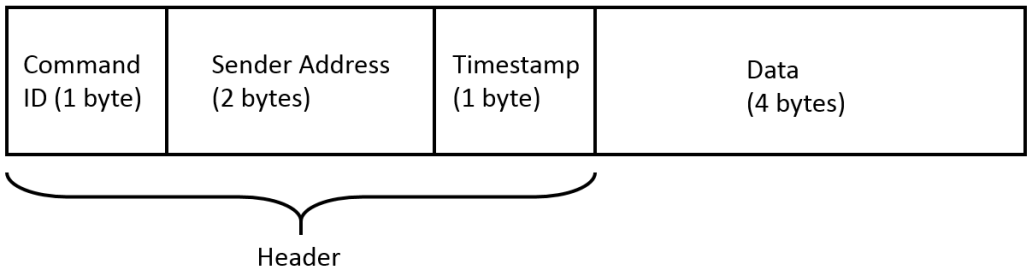


Figure 3.21: General format of command messages that can be sent to BreadCrumbs

Command ID 1 instructs the receiving BreadCrumb to send its deployment number to the BreadCrumb who sent it the command message.

Command ID 2 performs a similar function, except the hardware ID of the Xbee device attached to the BreadCrumb is sent instead.

Command ID 2 is primarily used for trouble shooting.

Command ID 3 tells the BreadCrumb to send an RSSI measurement to the BreadCrumb who sent the command. Command ID 4 stipulates that the BreadCrumb who received the message request an RSSI measurement from a BreadCrumb specified by the deployment number attached.

Command ID 5 states that the sender of the message should be set as the BreadCrumb that was deployed immediately after the receiving BreadCrumb.

Command ID 6 command the receiving BreadCrumb to transmit its location, if available, to the sender of the command.

Command ID 7 states that the sender of the message should be set as the BreadCrumb that was deployed immediately before the receiving BreadCrumb.

Command ID 9 is used when the BreadCrumb that sent the command requires critical information from the receiving BreadCrumb, such as location or confidence scores, but the receiving BreadCrumb does not have this information ready yet. Thus, when the information is ready, the receiving BreadCrumb will send the transmitting BreadCrumb a flag to prepare it to receive the critical information transmission.

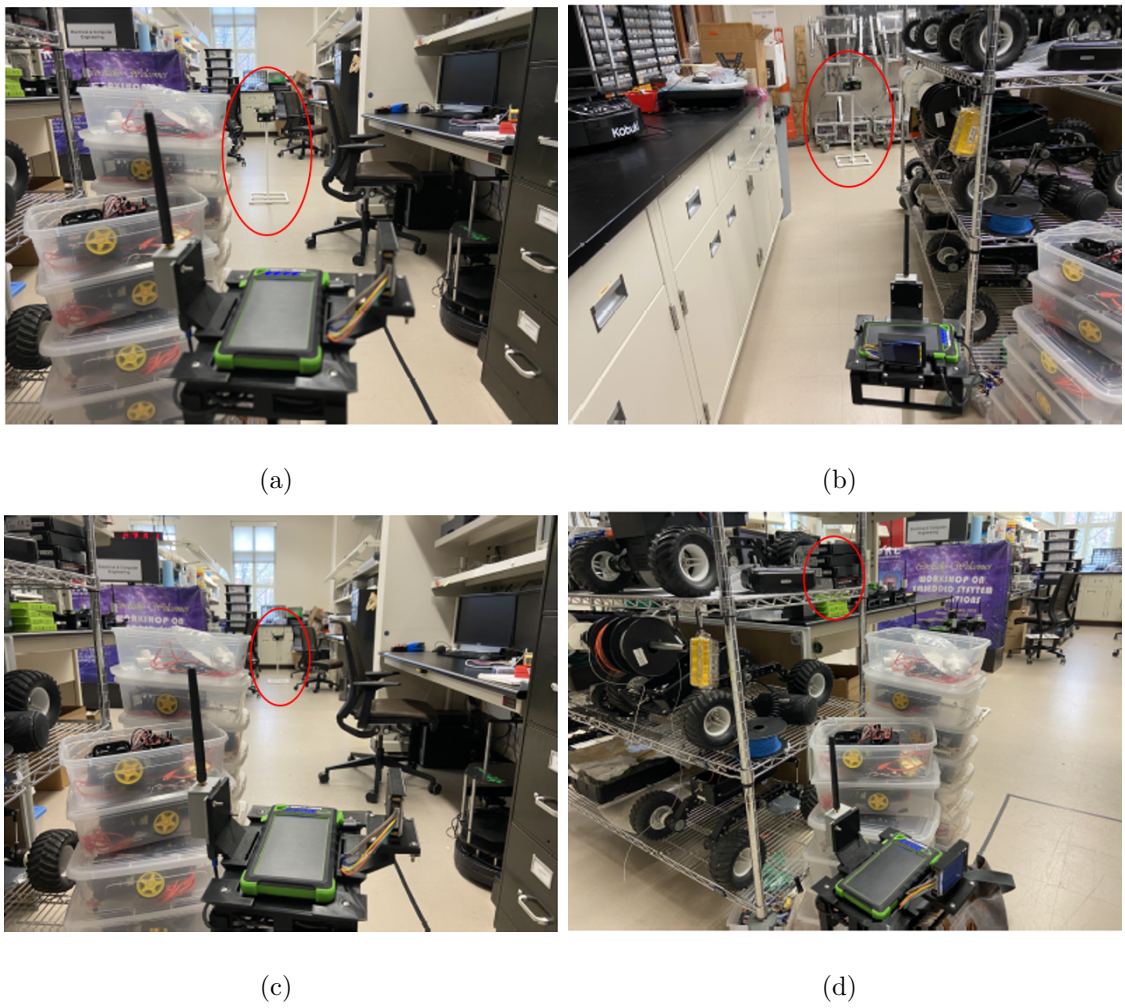
CHAPTER 4: Experimental Setup and Results

This section details the experimental setup of various elements for analysis.

4.1 Path Loss Exponent Impact

The path loss exponent is the primary adjustable factor in the log-distance path loss model. In most work, the PLE is considered either pre-determined based on a reference model, or is empirically determined from experimentation in a known environment where the device will operate.

A series of tests was conducted to determine the PLEs impact. The test was conducted in an indoor environment cluttered with many sources of multi-path between a series of four devices receiver and a transmitter are placed in four separate locations a that are 3.3528m, 3.657m 6.7056m, and 7.497m away respectively. These four locations are given the color codes teal, lime, purple, pink. Figure 4.1 shows the point of view from the transmitter to each receiver at a given distance.



(a) (b)

(c) (d)

Figure 4.1: Point of view of transmitter to receiver located in cluttered indoor area a) teal receiver 3.3528m away, b) lime receiver 3.657m away, c) purple receiver 6.7056m away, d) pink receiver 7.497m away

Roughly 200 measurements were taken of the RSSI values measured at each of the four locations at 1 second intervals. Using the log-distance path loss model, the distance was calculated using a reference path-loss value -31 Decibels measured at a distance of 1 meter away. In addition, a path-loss exponent of 2 was used. Once the distance was calculated, it was compared to the actual distance. The error between these two can be seen in Figure 4.2

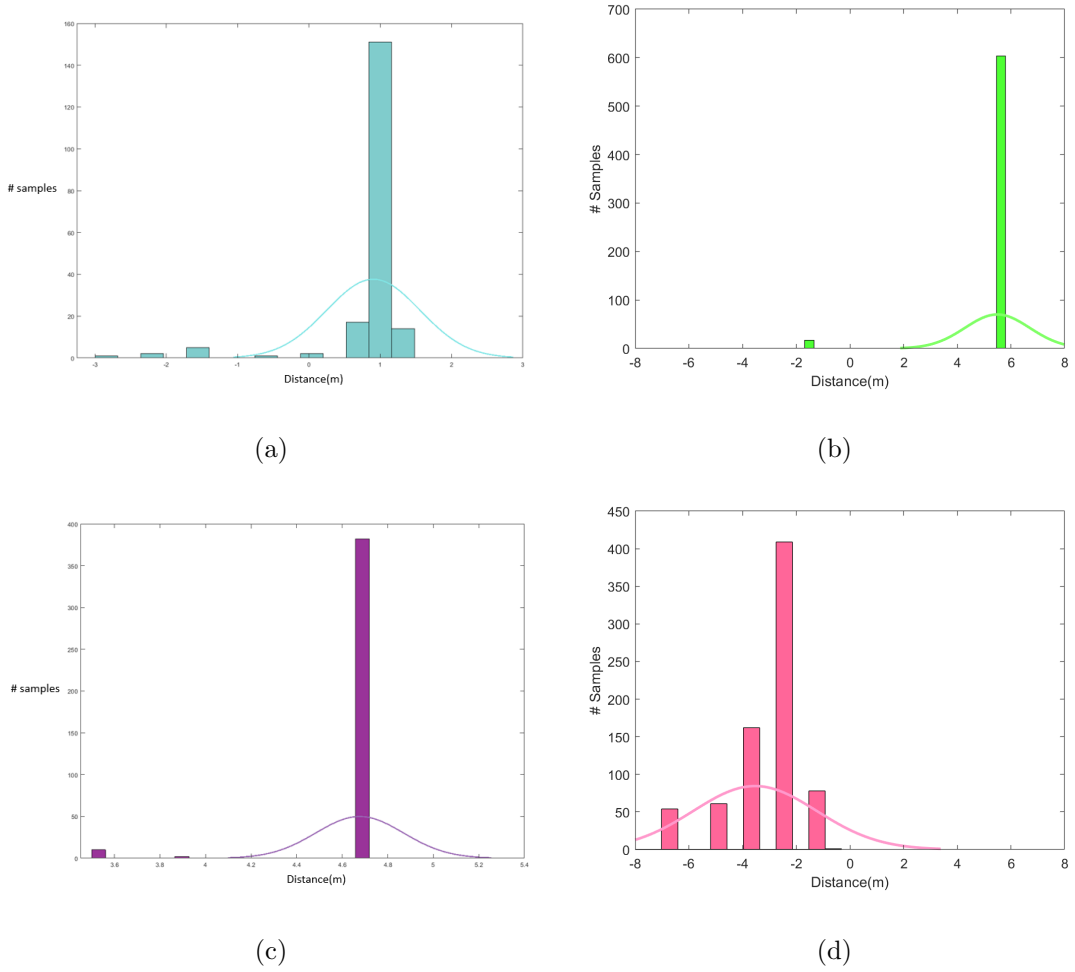


Figure 4.2: a) Distance error for teal receiver with normal distribution fit, b) Distance error for lime receiver with normal distribution fit, c) Distance error for purple receiver with normal distribution fit, d) Distance error for pink receiver with normal distribution fit

Examining these we see there is relatively consistent error measured at each receiver, and worse it is centered at different locations. This becomes more obvious if overlap the graphs together as in Figure 4.3

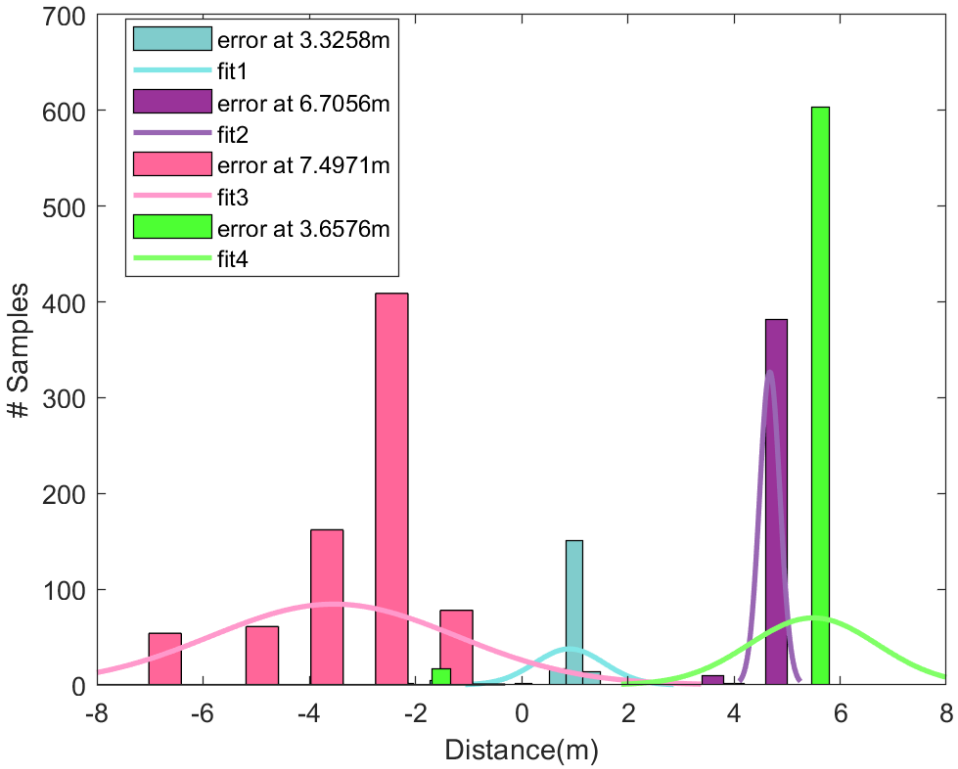


Figure 4.3: Overlapped errors of receivers from Figure 4.2 with path-loss exponent of 2.0

Next, we change the path-loss exponent to something better reflecting the environment, such as 2.4, then recalculate this errors in distance. The result is displayed in Figure 4.4

Now it can be seen that the errors have slightly shifted closer to 0. Proving of course that the path-loss exponent plays an critical role in the distance estimation

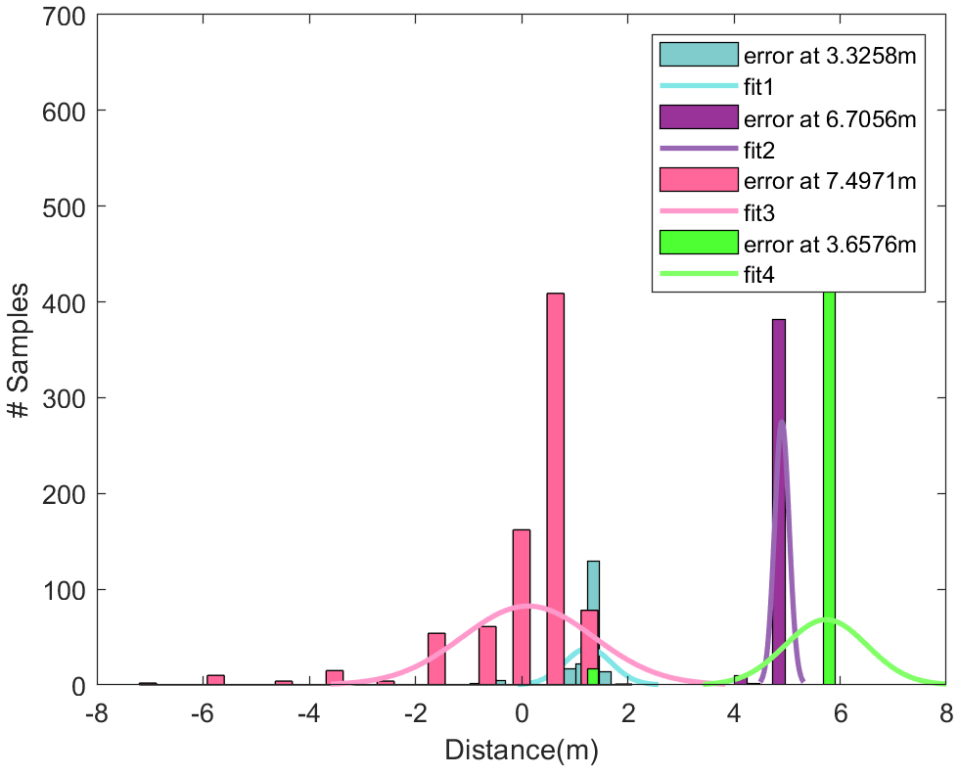
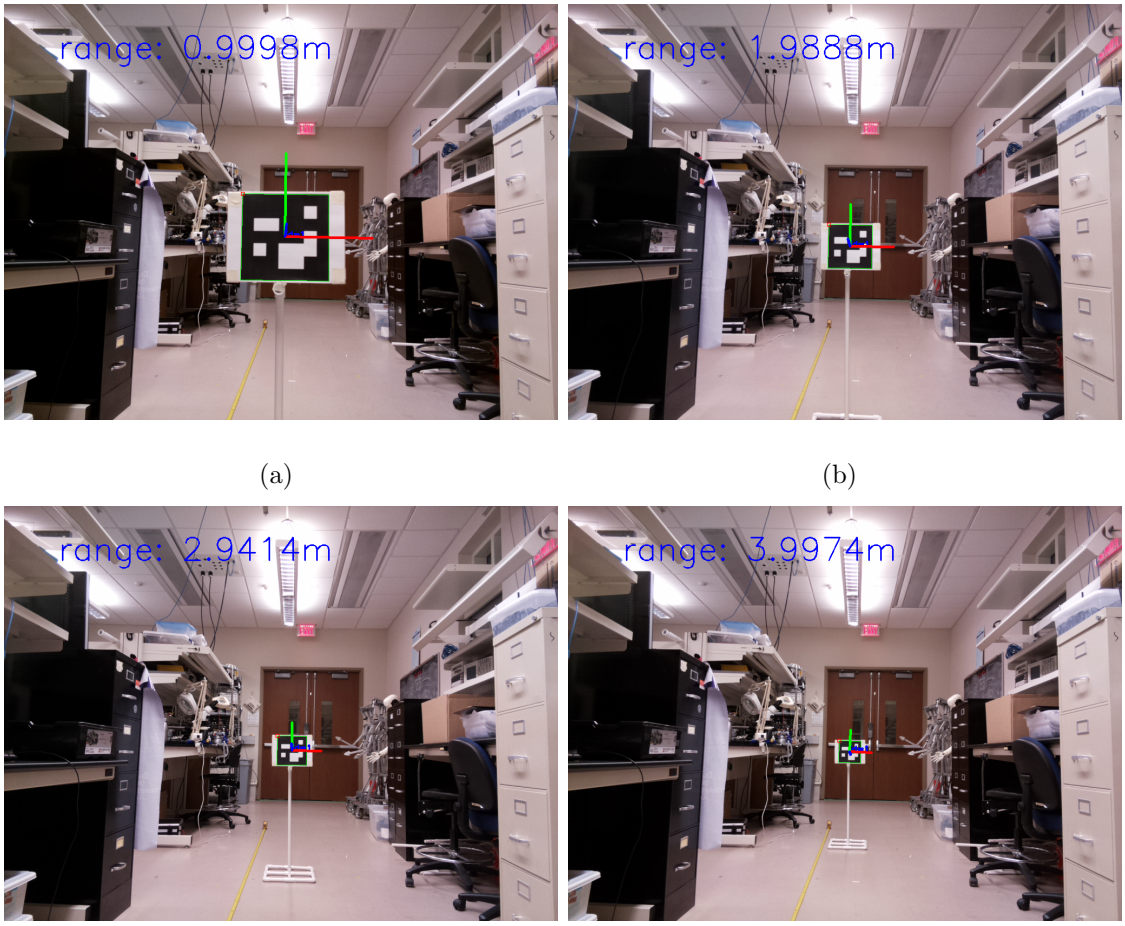


Figure 4.4: Overlapped errors of receivers with path-loss exponent of 2.4

Now it can be seen that the errors in Figure 4.4 have slightly shifted closer to 0 compared to the errors in Figure 4.3. Proving of course that the path-loss exponent plays an critical role in the distance estimation.

4.2 Range to AcUro Tag

An Aruco tag is used as the target identifiable object. Utilizing the AcUco OpenCv library a translation matrix from the AcUro tag to the cameras frame of reference can be determined. Examples of this can be seen in Figure 4.5.



(a) (b)

(c) (d)

Figure 4.5: Examples of identifying ArUco tags and the range resulting from the translation vector at ranges of: a) 1.0 meter, b) 2.0 meters, c) 3.0 meters, d) 4.0 meters

Next a series of 200 measurements was gathered at distances of 1, 3, 5, and 8 meters. The error between the known distance and the measured distance was calculated and is plotted in Figure 4.6.

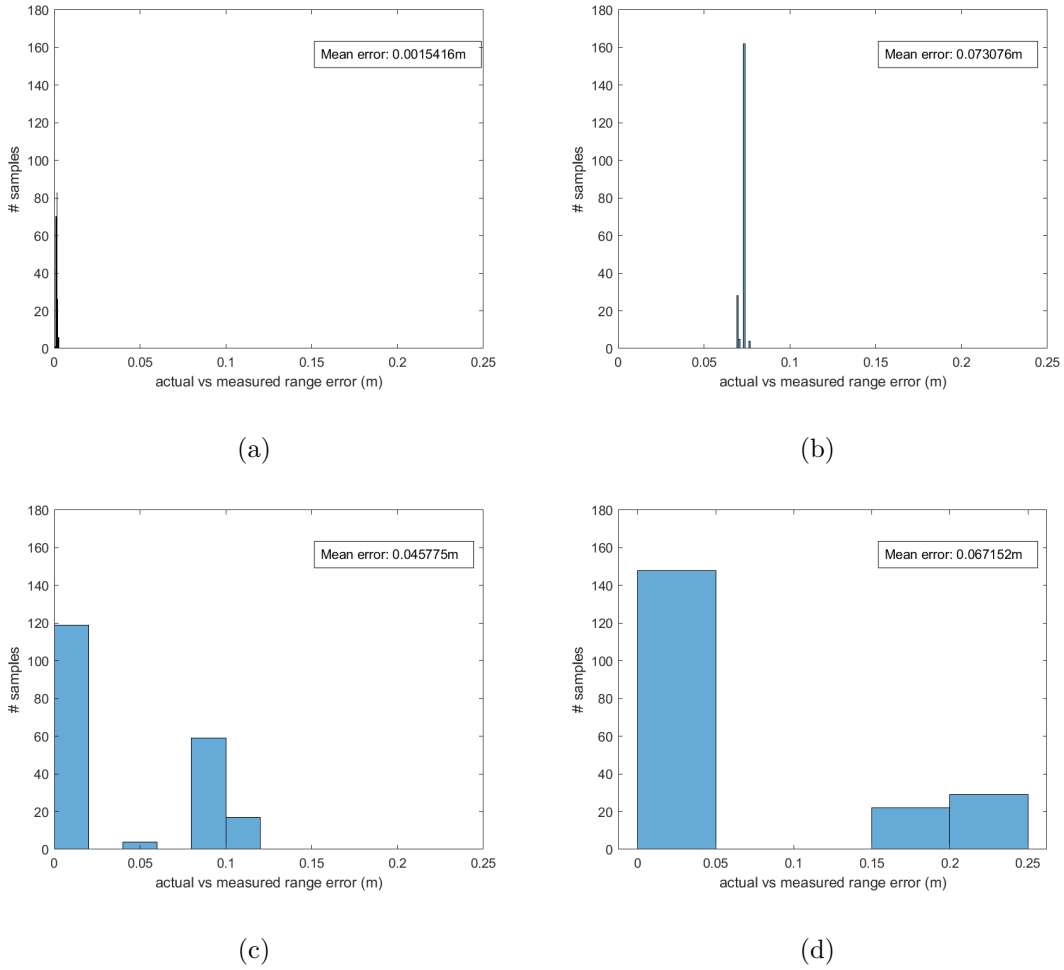


Figure 4.6: Histograms of the error between the actual range and the range measured to the ArUco tag at distances of: a) 1m, b) 3m, c) 5m, d) 8m

4.3 Path Loss Exponent Learn-ability

The Deep Deterministic Policy Gradient is expected to learn appropriate PLEs based on the visual location measurement/RSSI range measurement pairs. To that end, a simple test was conducted to confirm this is feasible. Two BreadCrumb_V2s were placed on either side of a door according to Figure 4.7.

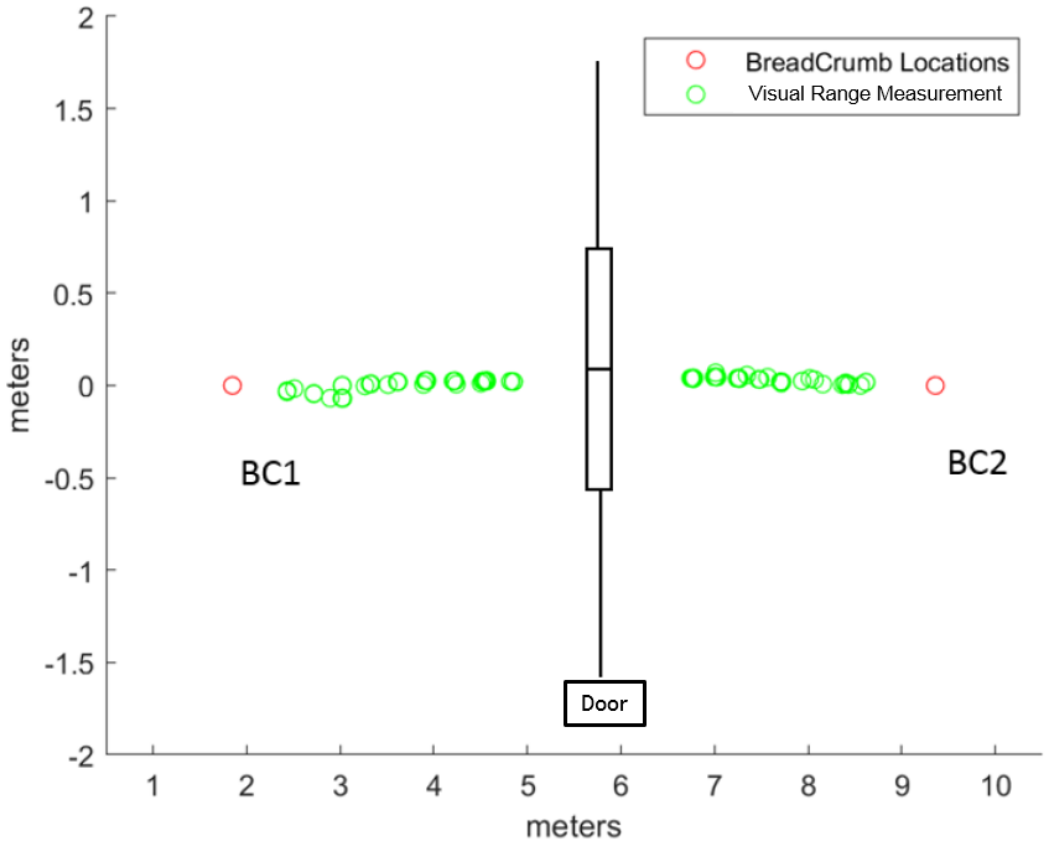


Figure 4.7: Visual range measurements taken from BC1 and BC2 in the global frame

Figure 4.8: Visualization of arUco tag moving through door as measurements are taken. (Note the annotated dates are not meaningful)

Roughly two thousand five hundred measurements were taken at various point in a straight line between BC1 and BC2. The process of moving the tag is visualized in Figure **??**. The RSSI value of a visual location measurement vs the distance from that visual location measurement is plotted in Figure 4.9. The distribution of the PLEs selected vs the visual range measurements can be seen in Figure 4.10.

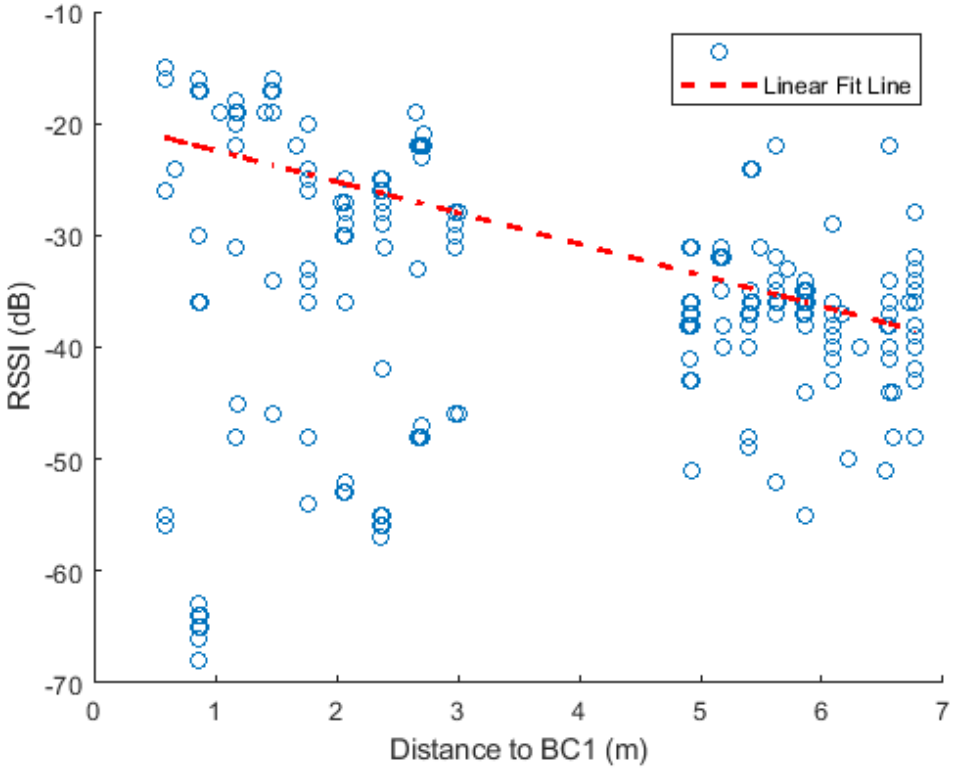


Figure 4.9: Distance to BC1 a visual location measurment was taken vs the RSSI measurement taken at that location. A linear best fit line was included to show outliers.

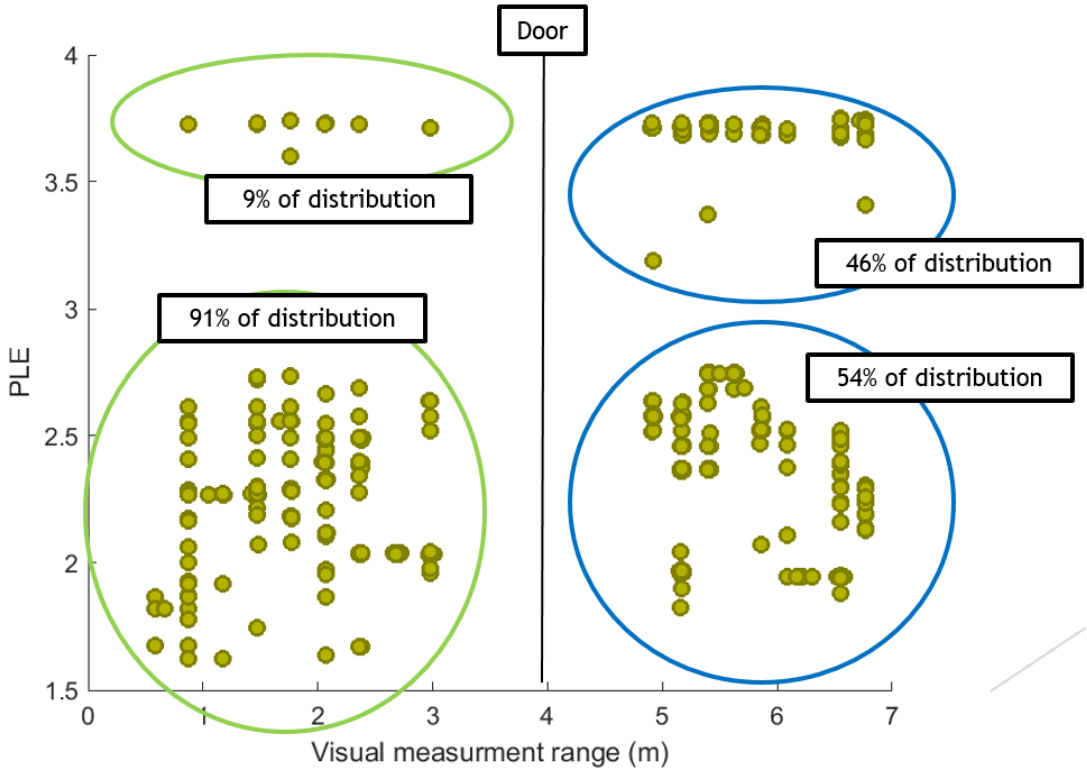


Figure 4.10: Selected PLE in through door experiment vs visual range measurements. Visualizes the distribution shift of selected PLEs before and after the range where the physical door is located. Hence, showing networks ability to learn obstacles' that it is not previously aware of.

Using a free space RSSI of -24 dB and path loss exponent of 2.0, which was empirically determined at a range of 1 meter, we can calculate the range from the RSSI measurements using the log-distance path loss model. After splitting the data 20/80 percent for training and validation respectively, the difference between calculated distances and visual location measurment distances were determined and plotted in Figure 4.11.

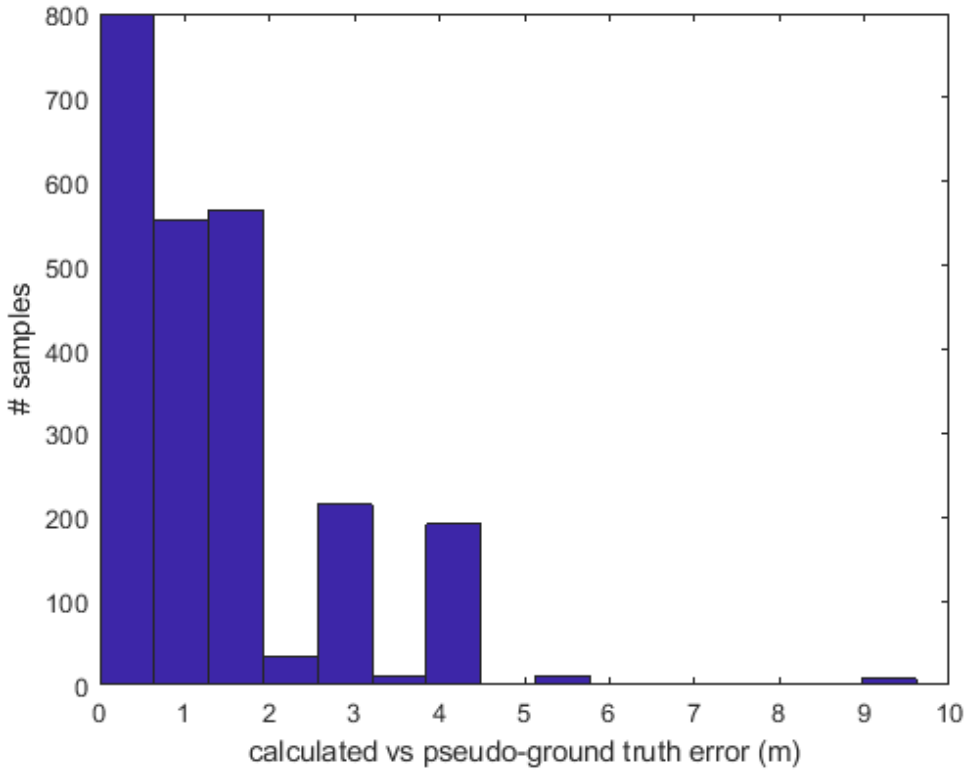


Figure 4.11: Error between the calculated distance from RSSI and visual location measurments with ideal path loss exponent of 2.0. Mean error of 1.2525 meters.

Next, the DDPG was trained with 20 percent of the dataset. The network was fed the same data as was used to generate Figure 4.11. The resulting PLEs were then used to calculate the distances from the measured RSSI values. Those calculated distances were then compared to the visual location measurment gathered distances. The results can be seen in Figure 4.12.

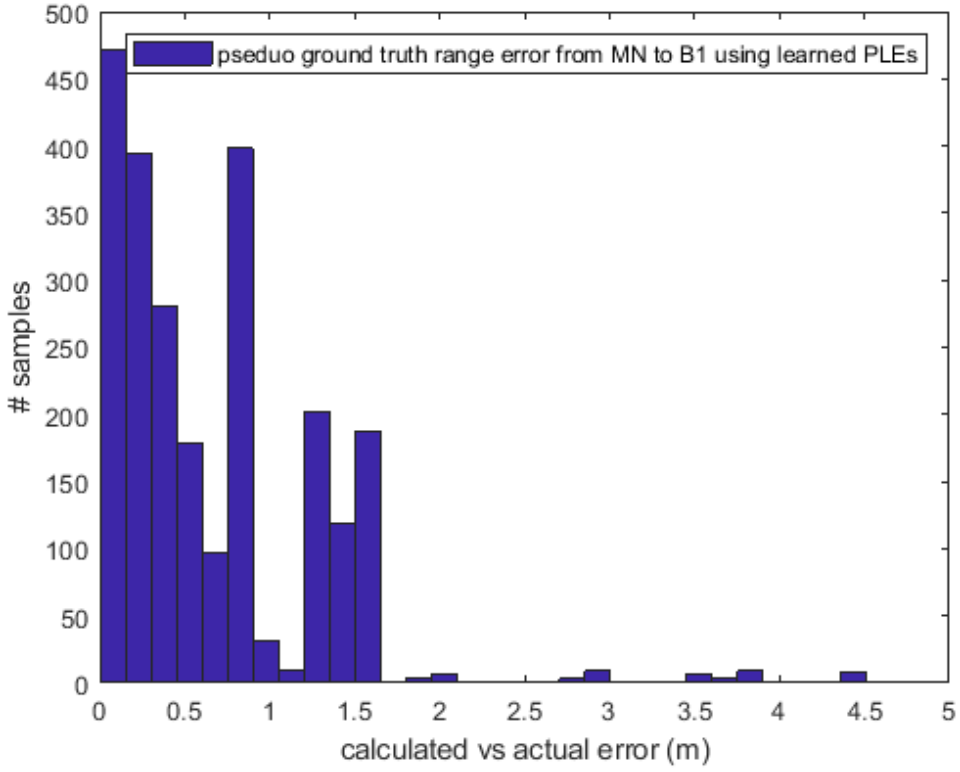


Figure 4.12: Error between the calculated distance from RSSI and visual location measurments with path loss exponents determined with the learned DDPG network. Mean error of 0.6792 meters.

From this, we can see that the underlying assumption of a Deep Deterministic Policy Gradient being able to learn a path loss exponent is valid.

4.4 Indoor Data

The indoor data collection was done over a series of three trials. Trials will have varying numbers of BreadCrumbs at varying locations but will all have the following information: BreadCrumb RSSI measurements to nearest neighbors, deployed BreadCrumb ground truth locations, initial BreadCrumb location as determined by the BreadCrumb localization algorithm, visual range measurement/RSSI measurements to a mobile node equipped with a BreadCrumb in the form of a Turtlebot2. This is visualized in Figure 4.13.

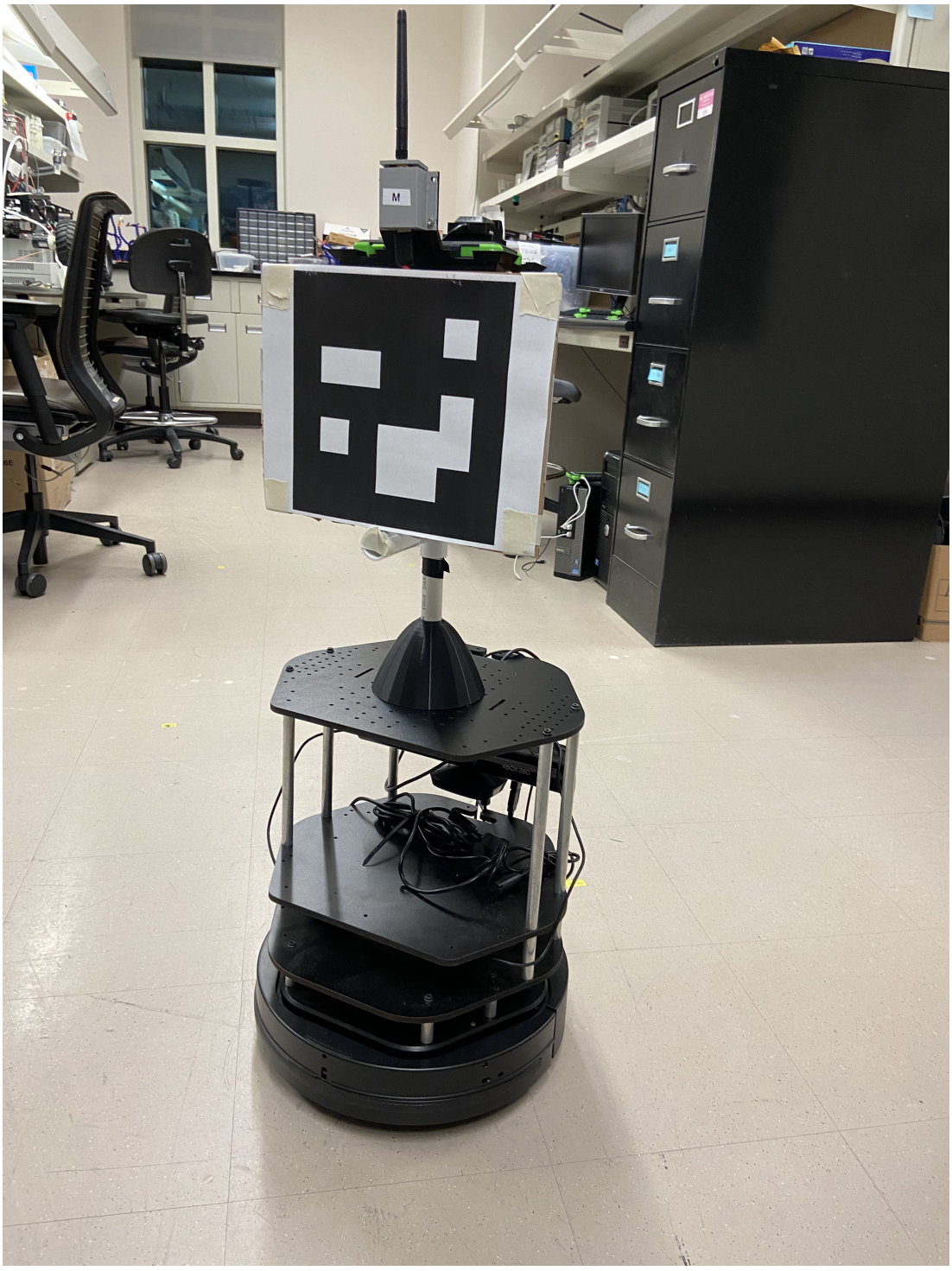


Figure 4.13: Turtlebot2 equipped with Mobile BreadCrumb Node to be used for data collection indoor

4.4.1 Trial 1: Limited Lab

Four BreadCrumbs V2s were deployed over a limited space in an indoor lab. A Turtlebot2 equipped with the mobile BreadCrumb was translated linearly for about 9 meters. This is visualized in Figure 4.14. The number of measurements gathered in the experiment can bee seen in Table 4.1. The initial locations calculated are visualized in Figure 4.15. The initial location errors are tabulated in Table 4.3. The mean error of the log-distance path loss model utilizing learned PLEs is tabulated in Table 4.12.

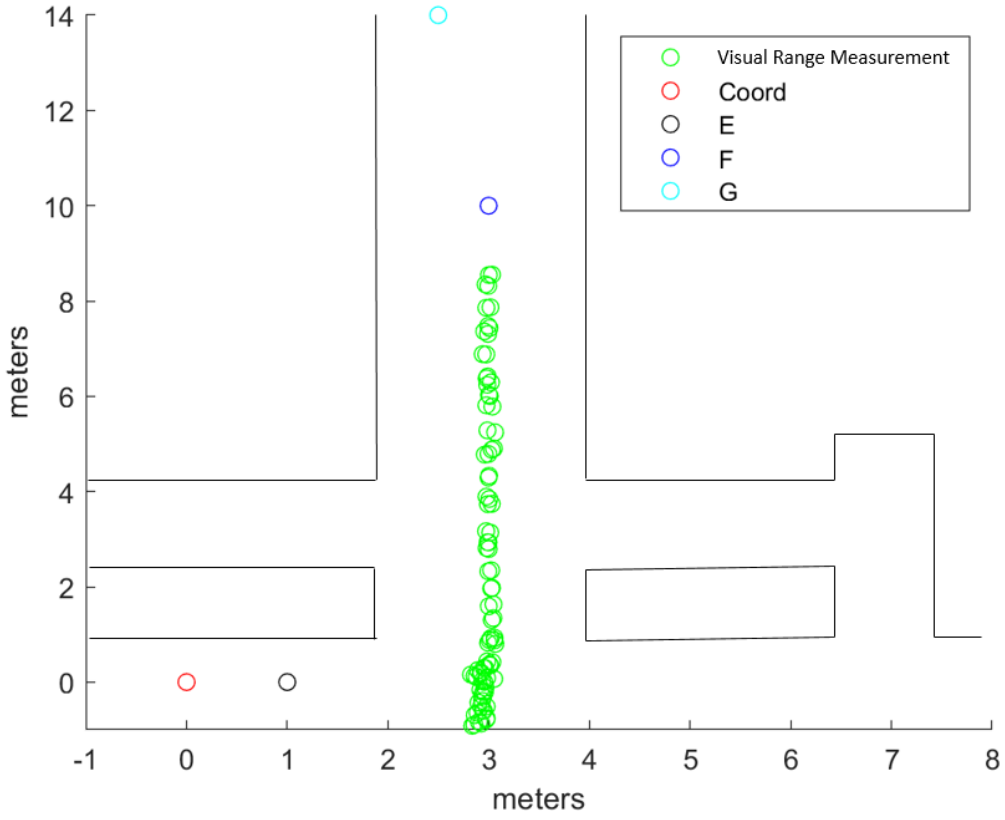


Figure 4.14: Indoor trial 1 deployed BreadCrumbs and visual location measurements. Walls included are for graphical purposes only and are not to scale.

Table 4.1: Indoor trial 1 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	73
F	142

Table 4.2: Indoor trial 1 BreadCrumb log-distance path loss error with learned PLEs

BreadCrumb ID	Error (meters)
E	0.9654
F	0.6458

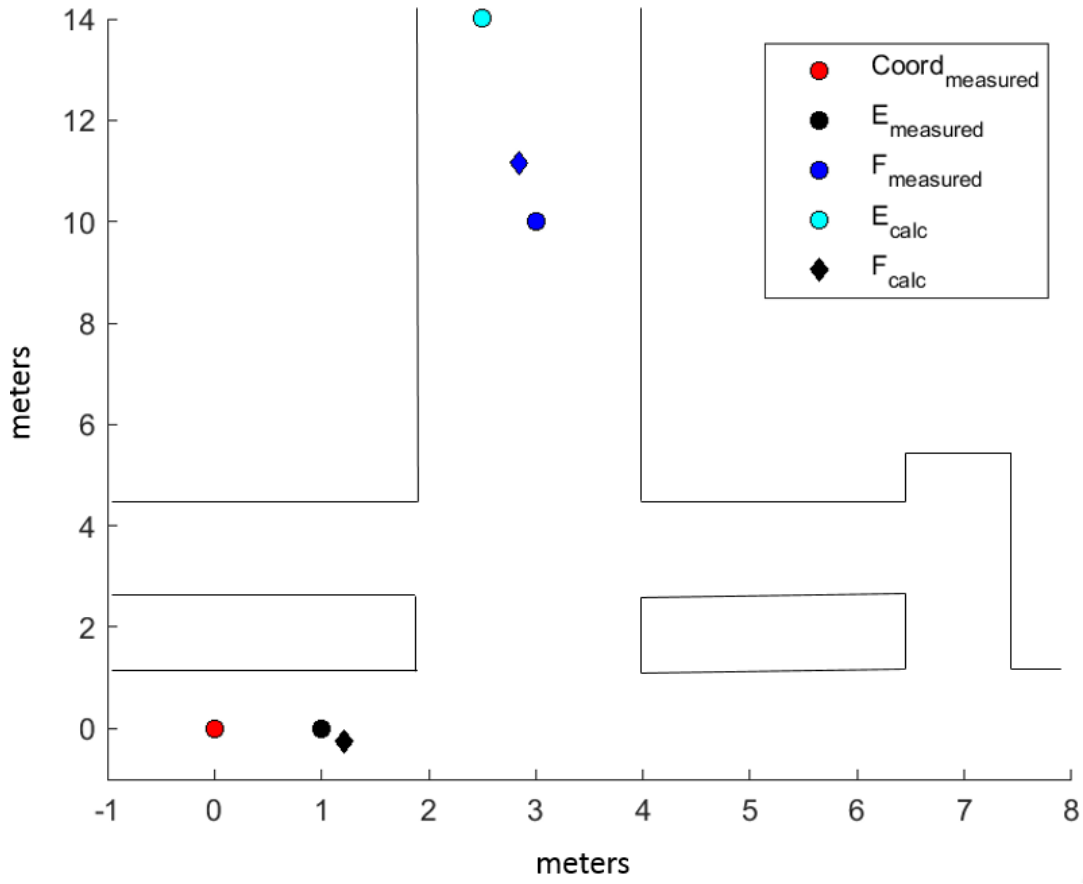


Figure 4.15: Indoor trial 1 BreadCrumb initial location results. Mean error: 0.7503.

Table 4.3: Indoor trial 1 BreadCrumb initial location and errors

BreadCrumb ID	[x,y] (meters)	Error (meters)
E	[1.2154, -0.2546]	0.8954
F	[2.8456, 11.1568]	1.0564

4.4.2 Trial 2: Lab

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed over a indoor lab space. A Turtlebot2 equipped with the mobile BreadCrumb was remotley controlled and moved around the environment. This is visualized in Figure 4.16. The number of measurements gathered in the experiment can bee seen in Table 4.4. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.17, 4.18, **??**, and 4.20. The initial locations calculated are visualized in Figure 4.21. The initial location errors are tabulated in Table 4.5. The mean error of the log-distance path loss model utilizing learned PLEs is tabulated in Table 4.15.

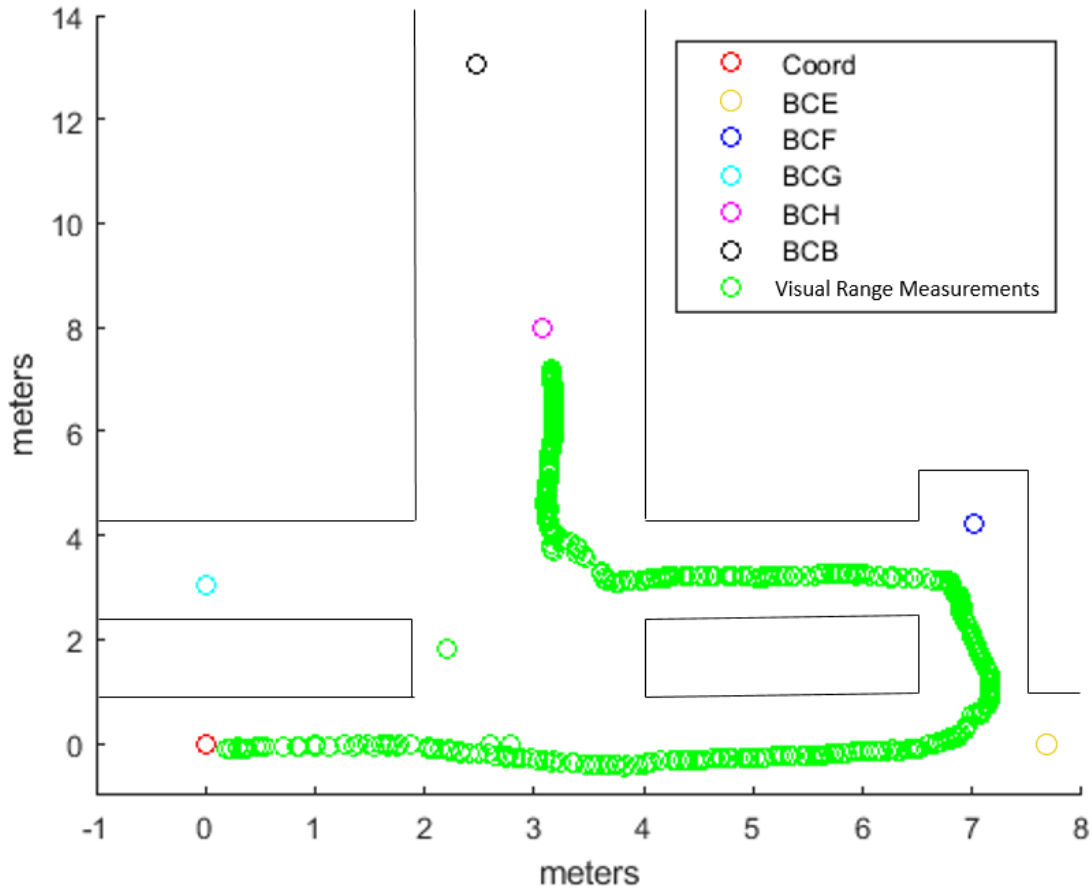


Figure 4.16: Indoor trial 2 deployed BreadCrumbs and visual location measurements. Walls included are for graphical purposes only and are not to scale.

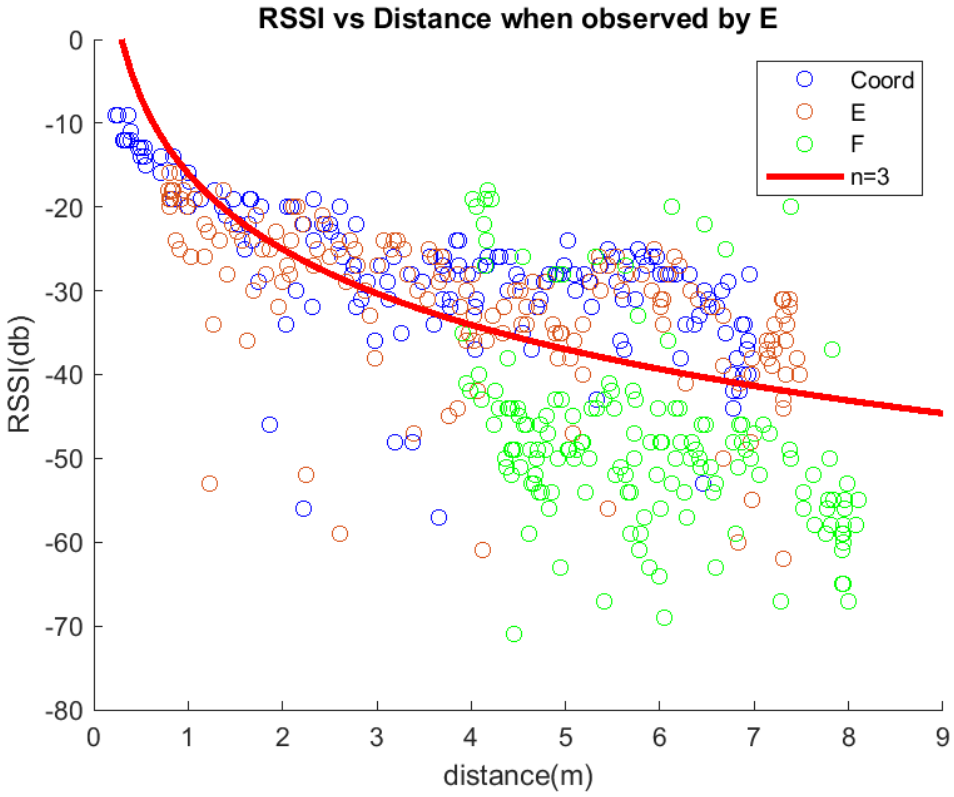


Figure 4.17: Indoor trial 2 RSSI vs range from visual measurements BreadCrumb E

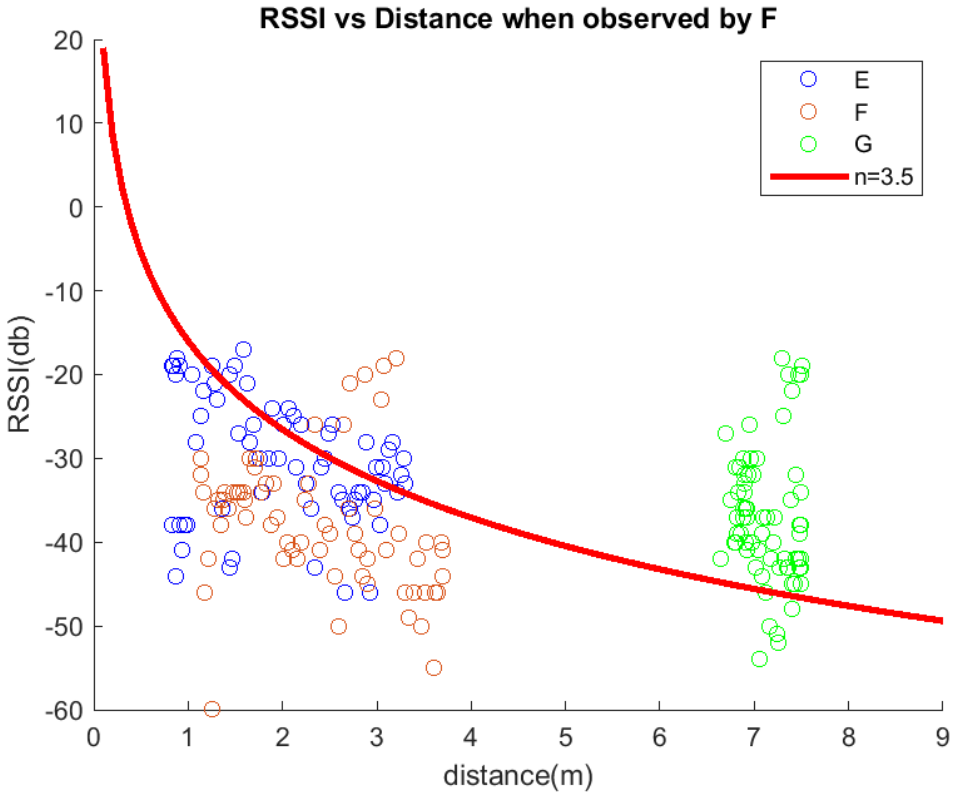


Figure 4.18: Indoor trial 2 RSSI vs range from visual measurements BreadCrumb F

Table 4.4: Indoor trial 2 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	231
F	329
G	345
H	277

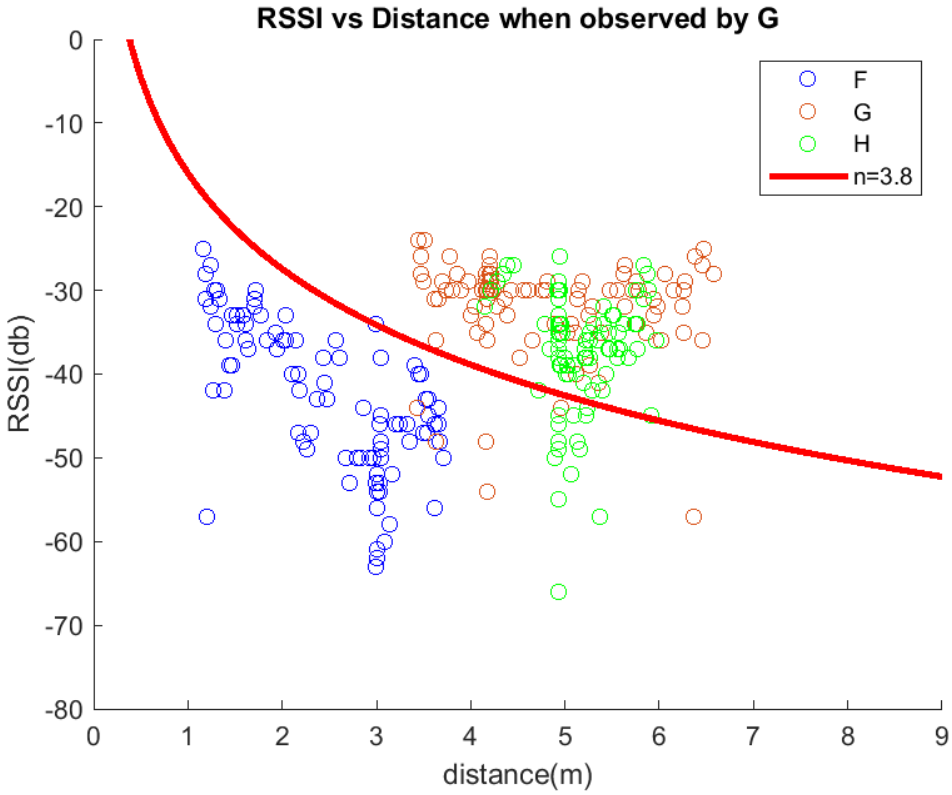


Figure 4.19: Indoor trial 2 RSSI vs range from visual measurements BreadCrumb G

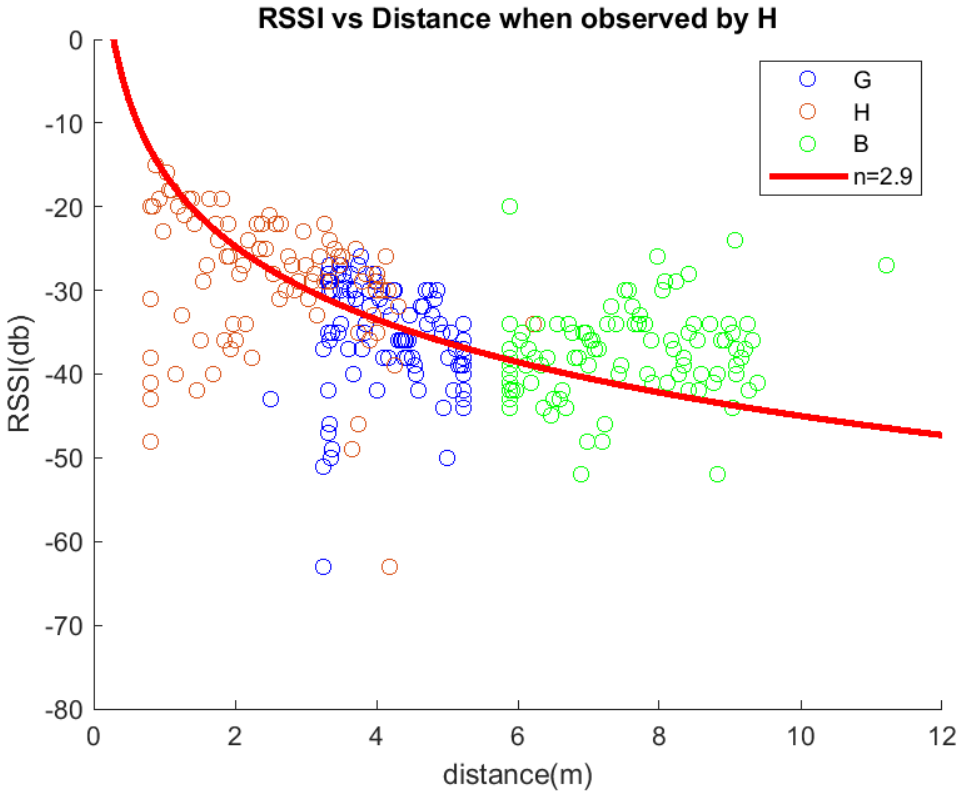


Figure 4.20: Indoor trial 2 RSSI vs range from visual measurements BreadCrumb H

Table 4.5: Indoor trial 2 BreadCrumb initial locations and errors

BreadCrumb ID	[x,y] (meters)	Error (meters)
E	[8.1486,-0.4567]	0.5298
F	[8.3425,4.1578]	0.3328
G	[0.3254,4.2214]	1.1378
H	[3.6721,8.9654]	0.3725

Table 4.6: Indoor trial 2 BreadCrumb mean log-distance path loss error with learned PLEs

BreadCrumb ID	Error (meters)
E	0.6543
F	1.3251
G	1.2254
H	1.1378

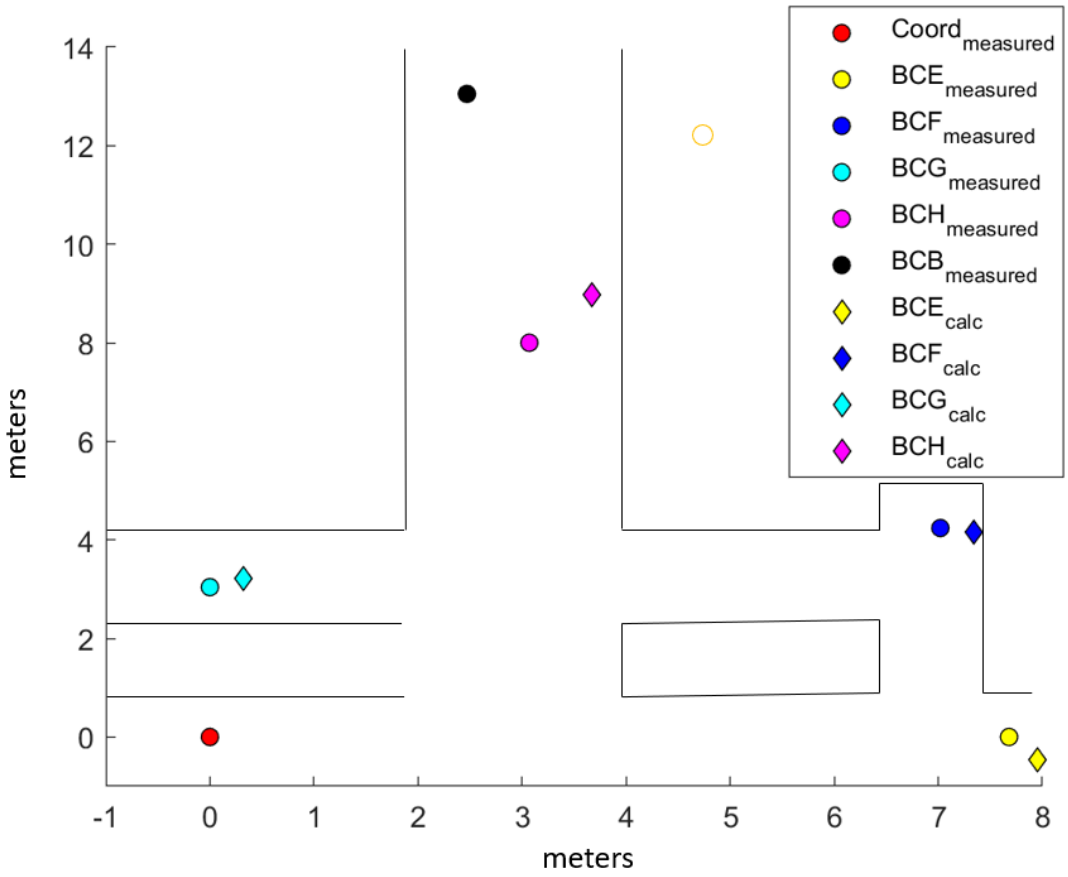


Figure 4.21: Indoor trial 2 initial location results. Mean error: 1.0856 meters.

4.4.3 Trial 3: Hallway Exterior

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed over a indoor hallway. A Turtlebot2 equipped with the mobile BreadCrumb was remotely controlled and moved around the environment. This is visualized in Figure 4.22. The number of measurements gathered in the experiment can bee seen in Table 4.7. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.23, 4.24, 4.25, and 4.26. The initial locations calculated are visualized in Figure 4.27. The initial location errors are tabulated in Table 4.8. The mean error of the log-distance path loss model utilizing learned PLEs is tabulated in Table 4.18.

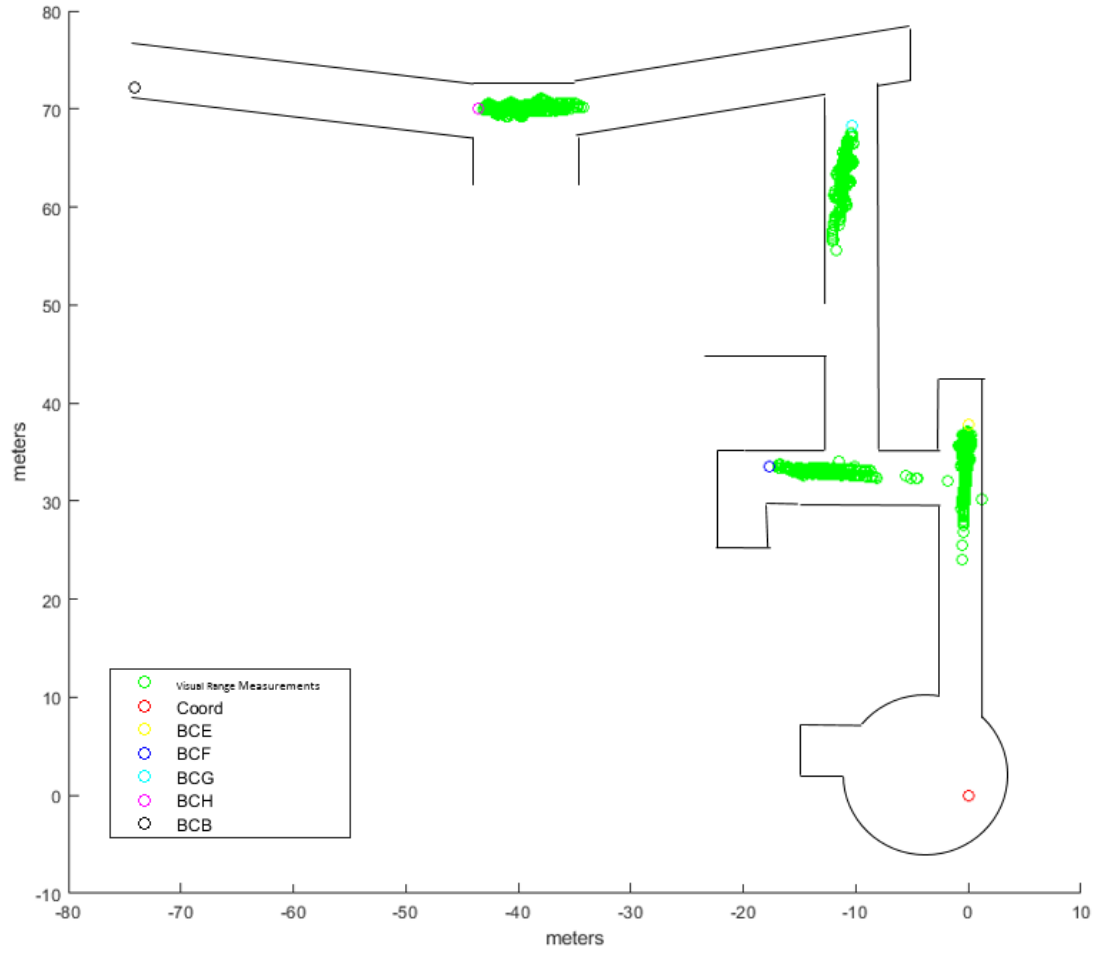


Figure 4.22: Indoor trial 3 deployed BreadCrumbs and visual location measurements. Walls included are for graphical purposes only and are not to scale.

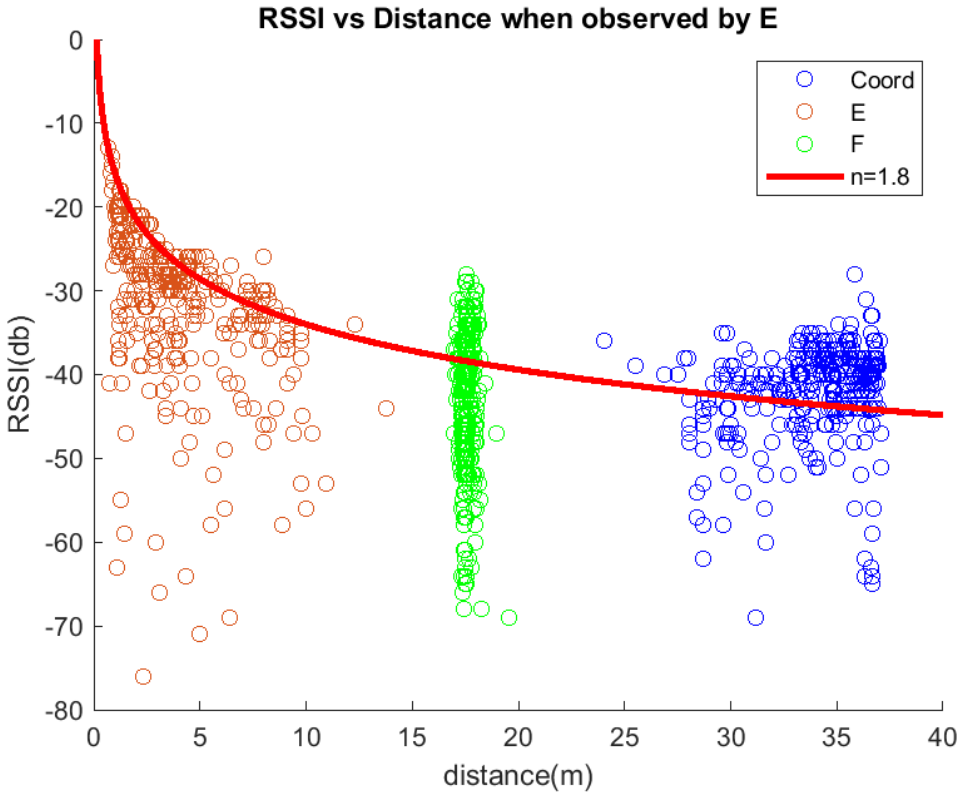


Figure 4.23: Indoor trial 3 RSSI vs range from visual measurements for BreadCrumb E

Table 4.7: Indoor trial 3 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	575
F	876
G	830
H	571

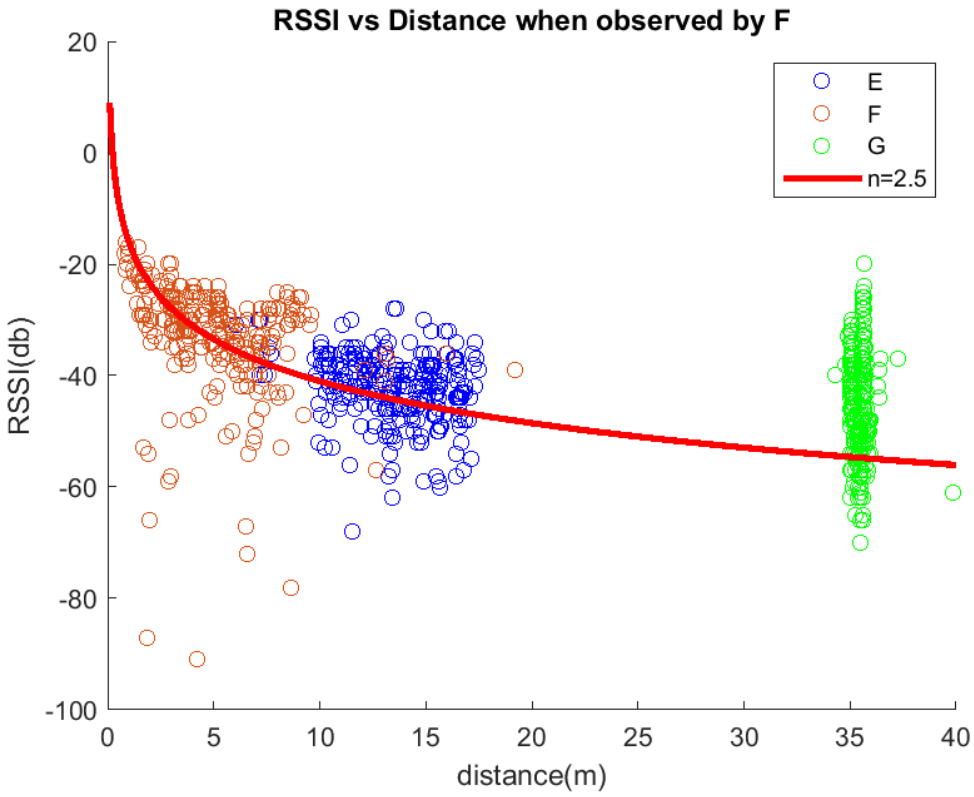


Figure 4.24: Indoor trial 3 RSSI vs range from visual measurements for BreadCrumb F

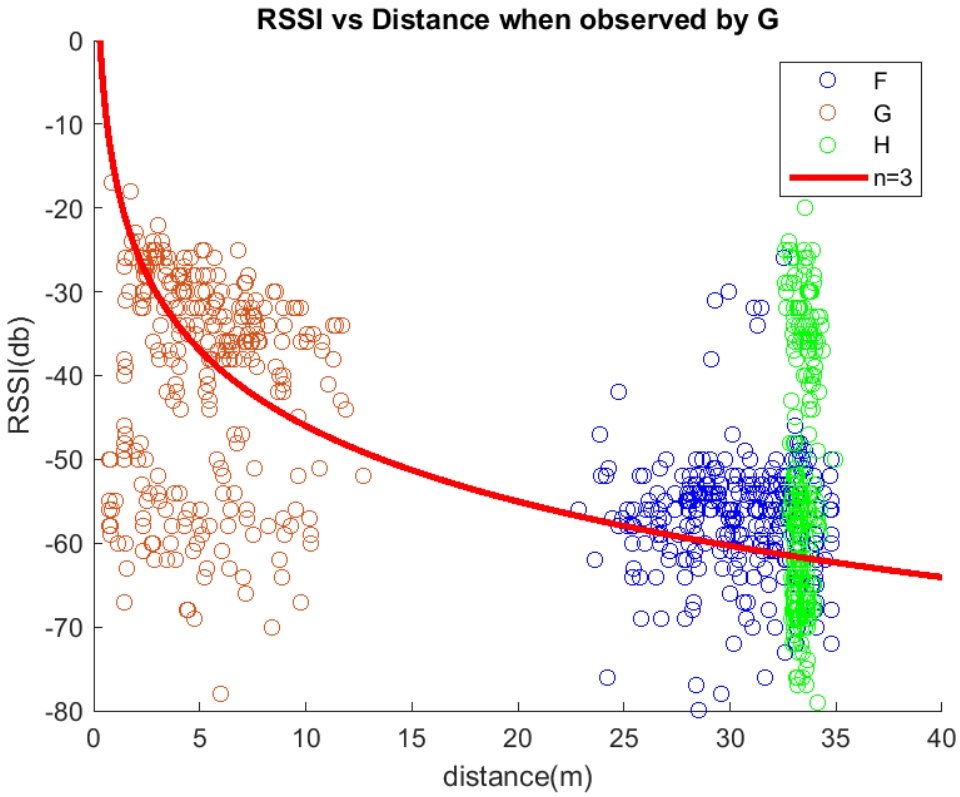


Figure 4.25: Indoor trial 3 RSSI vs range from visual measurements for BreadCrumb G

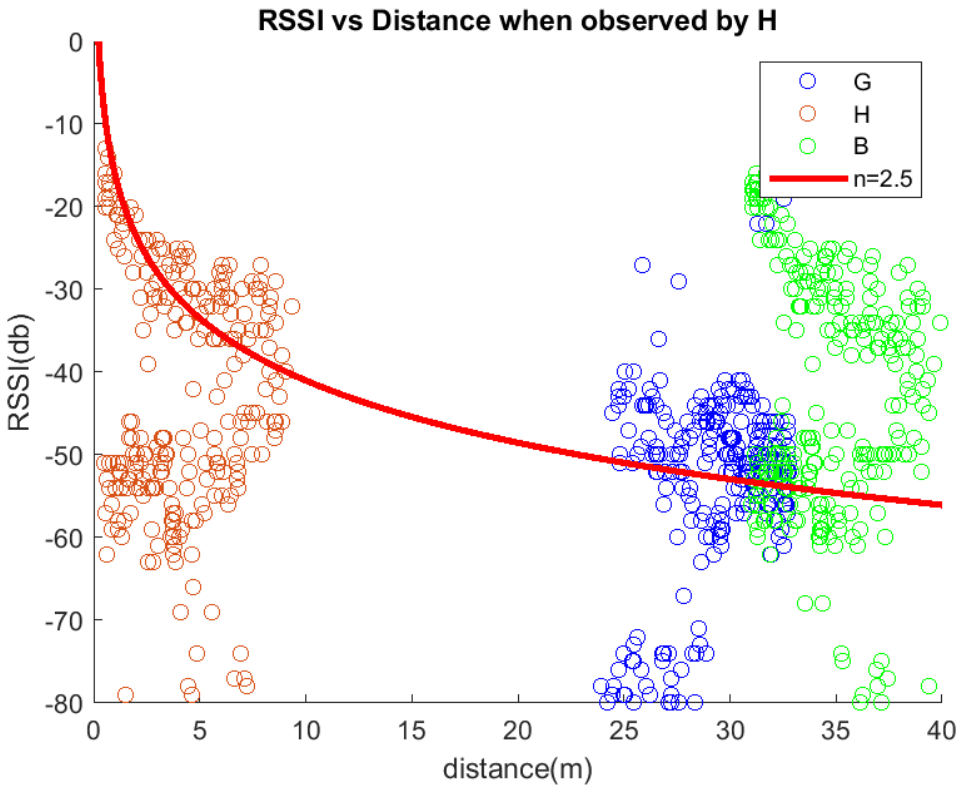


Figure 4.26: Indoor trial 3 RSSI vs range from visual measurements for BreadCrumb H

Table 4.8: Indoor trial 3 BreadCrumb initial locations and errors

BreadCrumb	[x,y] (meters)	Error (meters)
E	[1.5263, 46.4501]	8.7884
F	[-18.6592, 37.8684]	4.4498
G	[-11.8654, 74.4596]	6.3545
H	[-48.7168, 69.1564]	5.2172

Table 4.9: Indoor trial 3 BreadCrumb mean log-distance path loss error with learned PLEs

BreadCrumb	Error (meters)
E	4.2641
F	2.4569
G	2.7387
H	1.5735

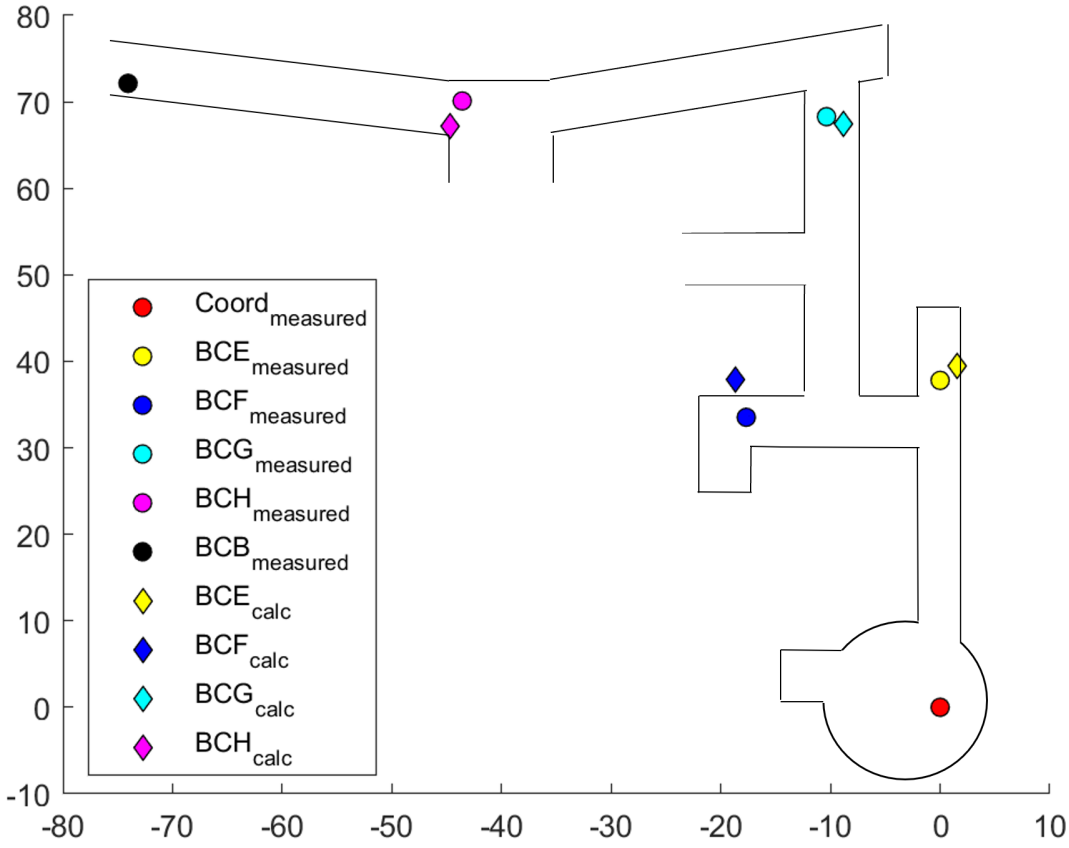


Figure 4.27: Indoor trial 3 initial location Results. Mean Error: 6.2025 meters.

4.5 Outdoor Data

The outdoor data was gathered from a wooded forest area with a nearby pond. A mobile robot was not available, so the mobile BreadCrumb was attached to a PVC pipe pole, as seen in Figure 4.28, and physically moved around the deployed BreadCrumbs while visual location measurement/RSSI range measurements were gathered.

Figure 4.28: Mobile BreadCrumb mounted on PVC pipe to be used for outdoor data gathering.

4.5.1 Trial 1: Fitness Trail

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed over a outdoor trail running along a wooded forest area and outdoor pond. A mobile BreadCrumb mounted on a PVC pipe was translated along this trail. This is visualized in Figure 4.29. A zoomed in view of BreadCrumbs F and H can bee seen in Figure 4.30. The number of measurements gathered in the experiment can bee seen in Table 4.10. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.31, 4.32, 4.33, and 4.34. The initial locations calculated are visualized in

Figure 4.35. The initial location errors are tabulated in Table 4.11. After training, the mean error of the log-distance path loss model utilizing learned PLEs was tabulated in Table 4.12.

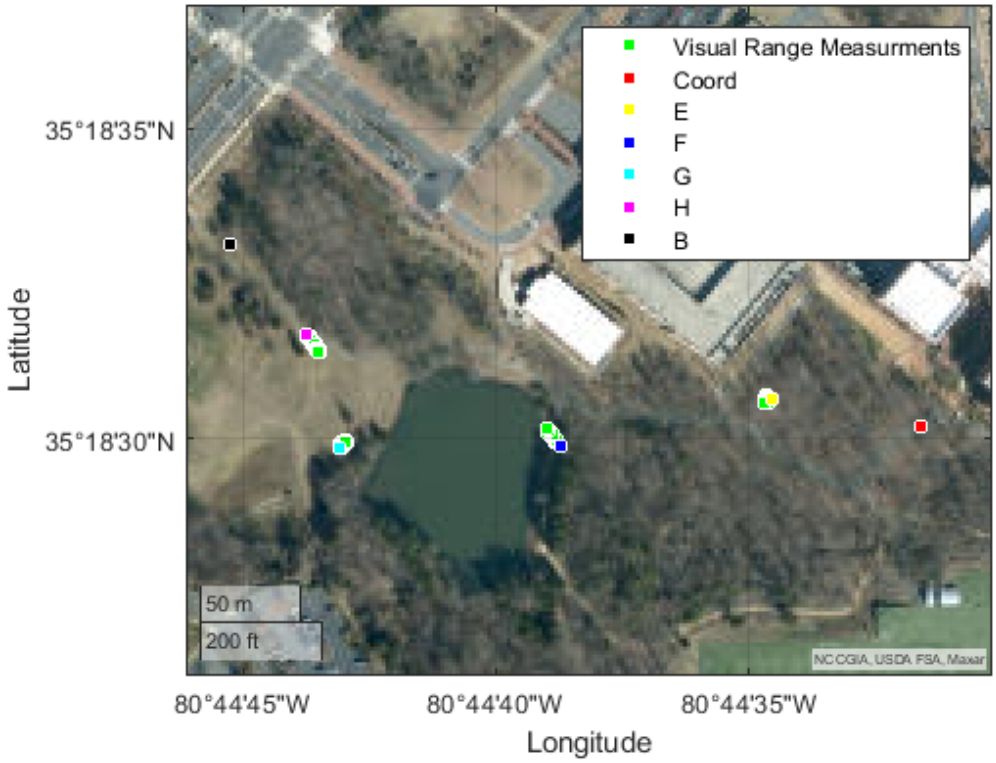
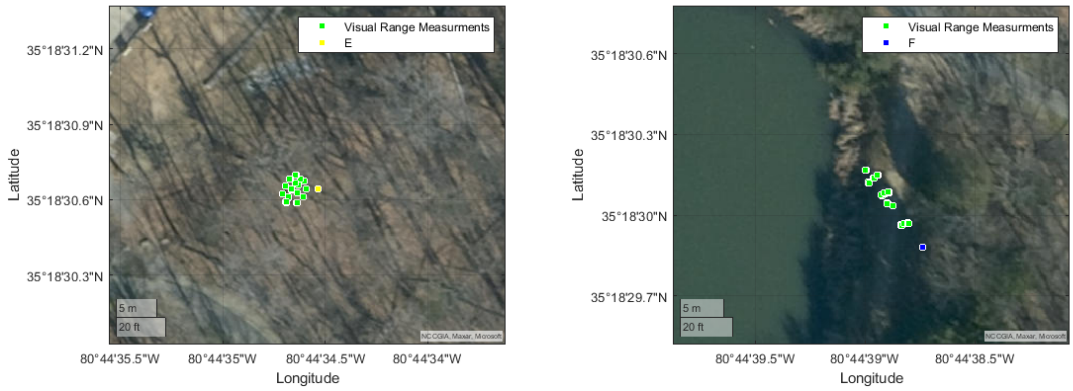


Figure 4.29: Outdoor trial 1 deployed BreadCrumbs and visual location measurements.



(a) (b)

Figure 4.30: Zoomed in view of BreadCrumb E and F deployed along fitness trail with visual location measurements. a) BreadCrumb E, b) BreadCrumb F

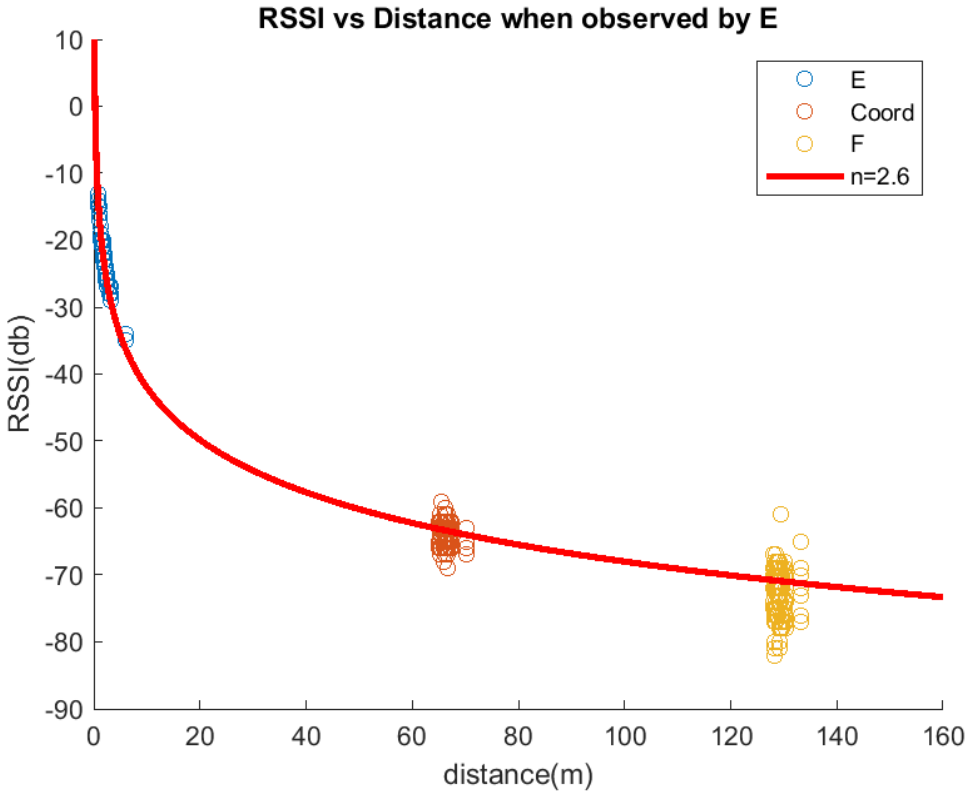


Figure 4.31: Outdoor trial 1 RSSI vs range from visual measurements for BreadCrumbs E

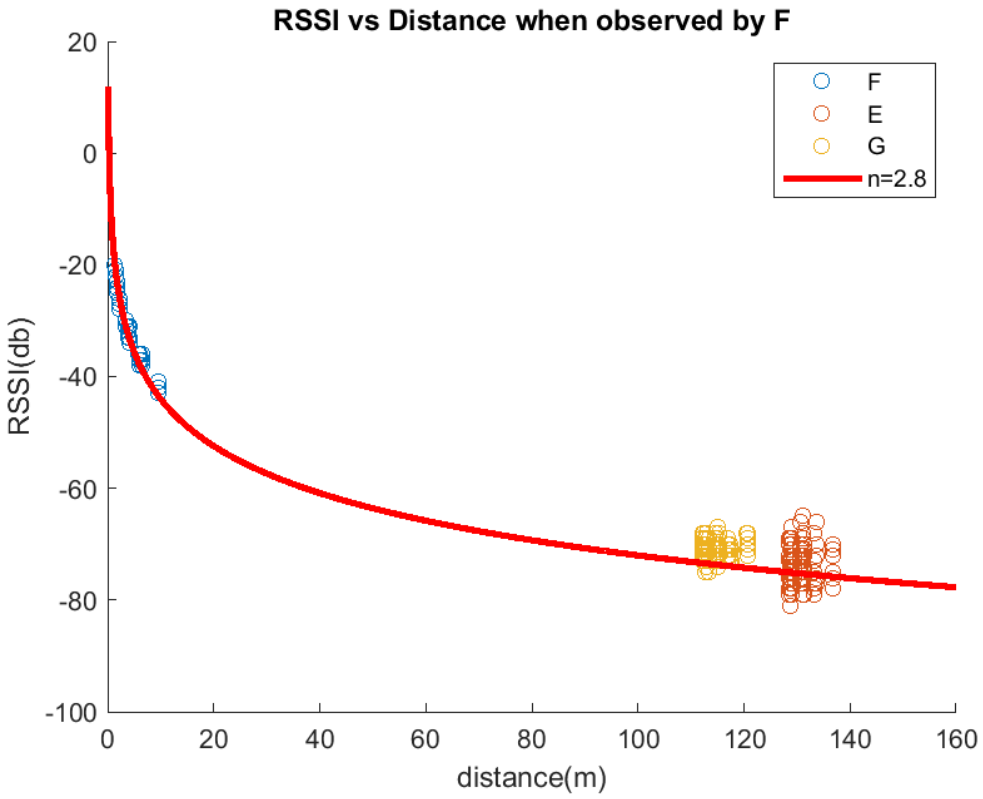


Figure 4.32: Outdoor trial 1 RSSI vs range from visual measurements for Bread-Crumbs F

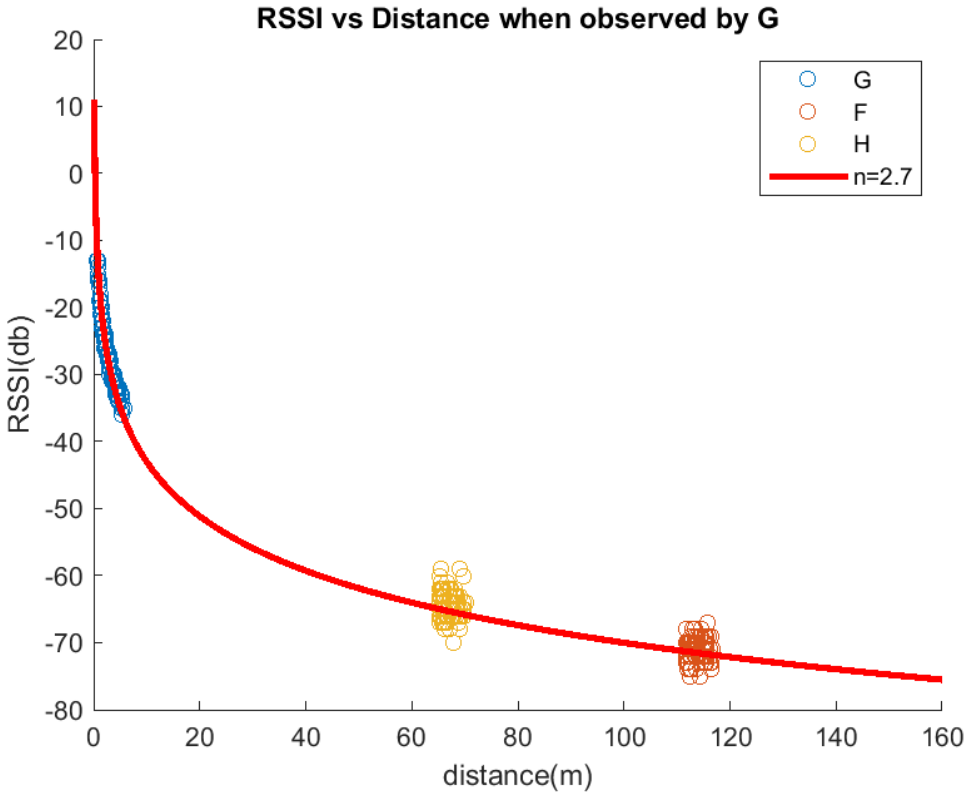


Figure 4.33: Outdoor trial 1 RSSI vs range from visual measurements for Bread-Crumbs G

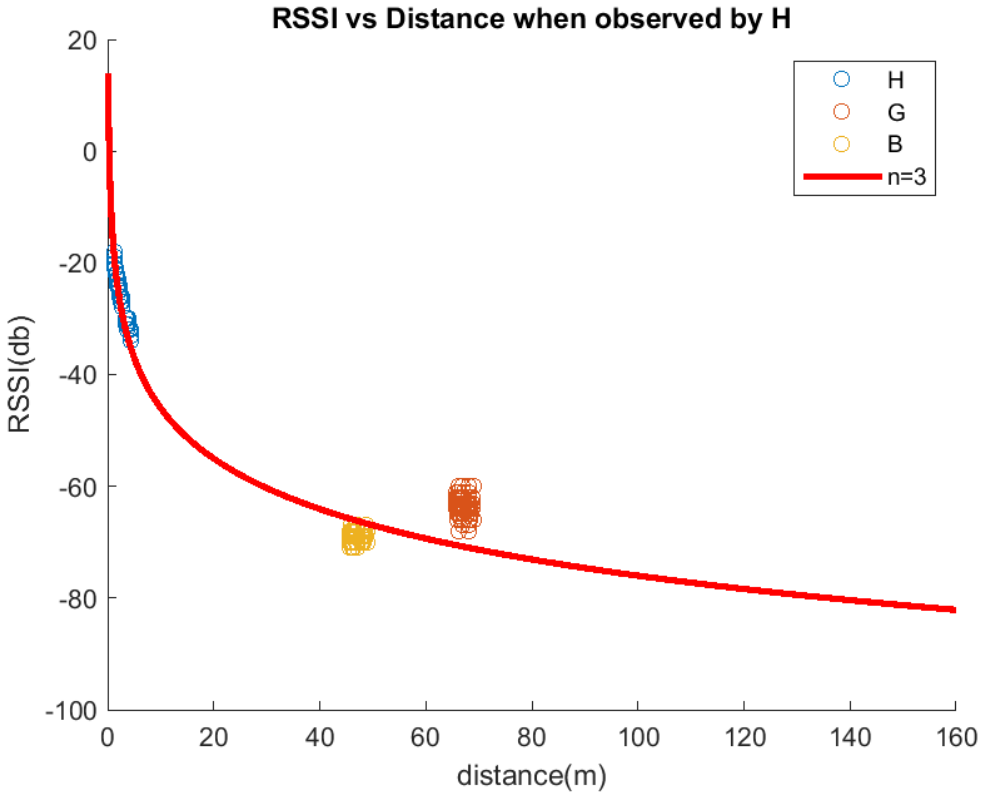


Figure 4.34: Outdoor trial 1 RSSI vs range from visual measurements for BreadCrumbs H

Table 4.10: Outdoor trial 1 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	429
F	462
G	480
H	560

Figure 4.35: Outdoor trial 1 BreadCrumb initial location results. Mean Error: 10.1031 meters.

Table 4.11: Outdoor trial 1 BreadCrumb initial locations and errors

BreadCrumb ID	Calculated [Lat,Long] (deg)	Error (meters)
E	[35.308479N, 80.742984W]	8.8877
F	[35.308321N, 80.744033W]	7.7647
G	[35.308218N, 80.745190W]	8.9576
H	[35.308923N, 80.745539W]	14.8020

4.5.2 Trial 2: Woods

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed over a outdoor heavily forested trail. A mobile BreadCrumb mounted on a PVC pipe was translated along this trail. This is visualized in Figure 4.36. The number of measurements

Table 4.12: Outdoor trial 1 BreadCrumb mean log-distance path loss error with learned PLEs

BreadCrumb	Error (meters)
E	6.5423
F	12.4583
G	3.1254
H	9.1253

gathered in the experiment can bee seen in Table 4.13. The distribution of the visual range measurements and measured RSSI is visualised in Figure **??**. The initial locations calculated are visualized in Figure 4.41. The initial location errors are tabulated in Table 4.14. After training, the mean error of the log-distance path loss model utilizing learned PLEs was tabulated in Table 4.15.

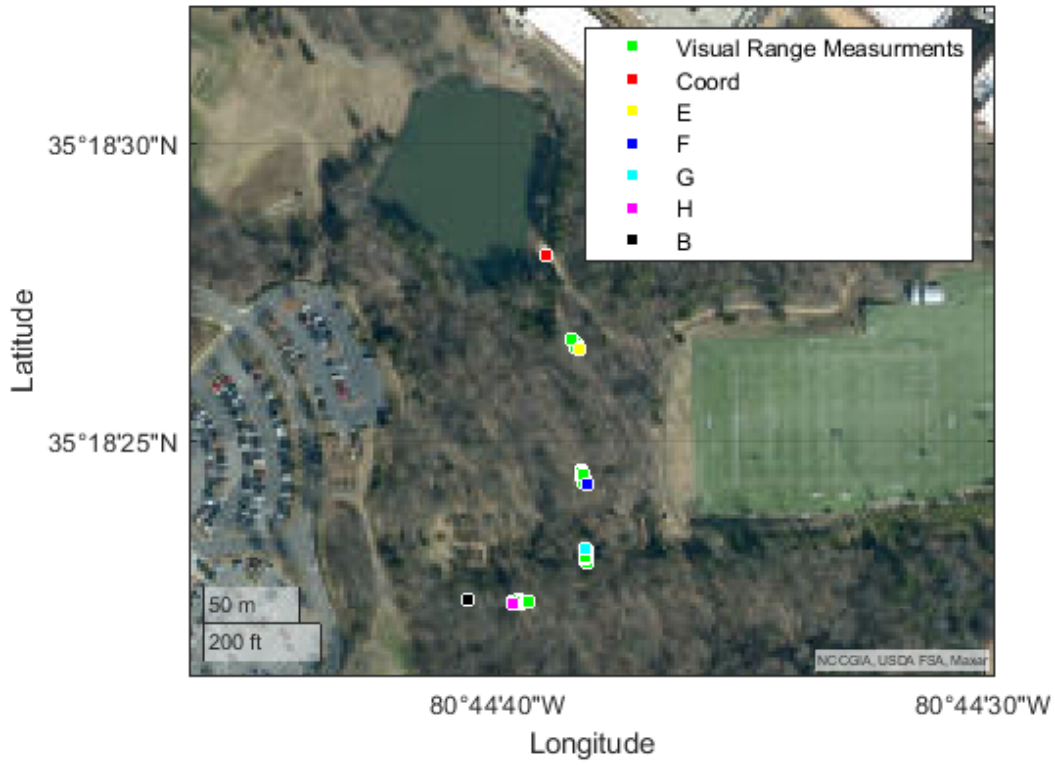


Figure 4.36: Outdoor trial 2 deployed BreadCrumbs and visual location measurements.

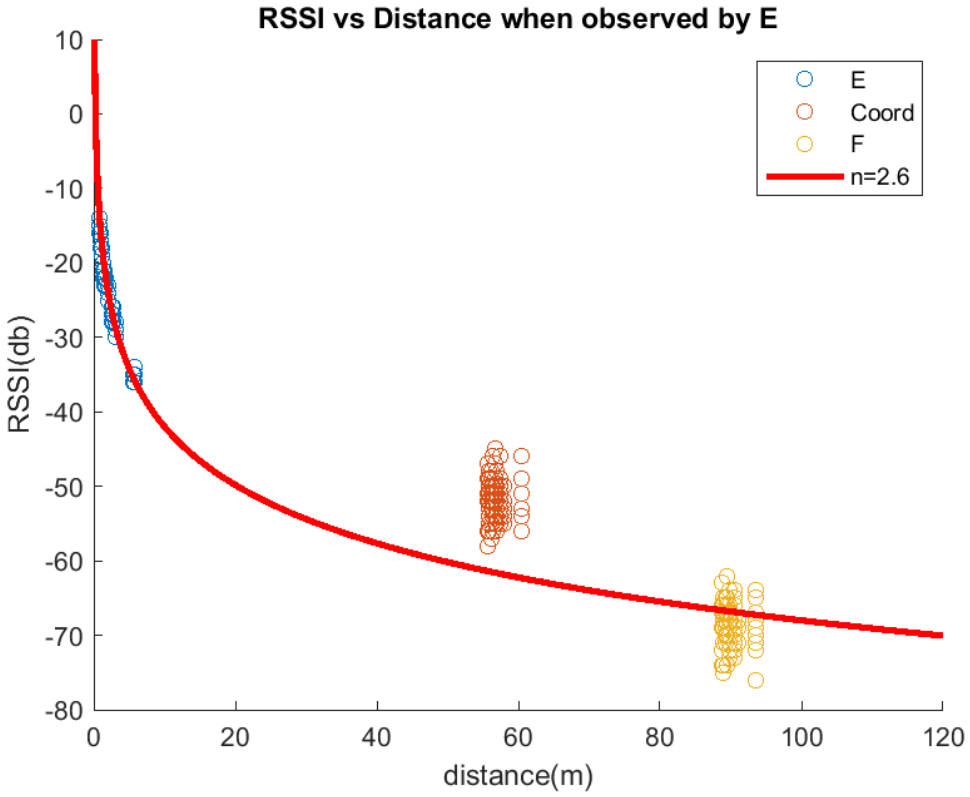


Figure 4.37: Outdoor trial 2 RSSI vs range from visual measurements for BreadCrumb E

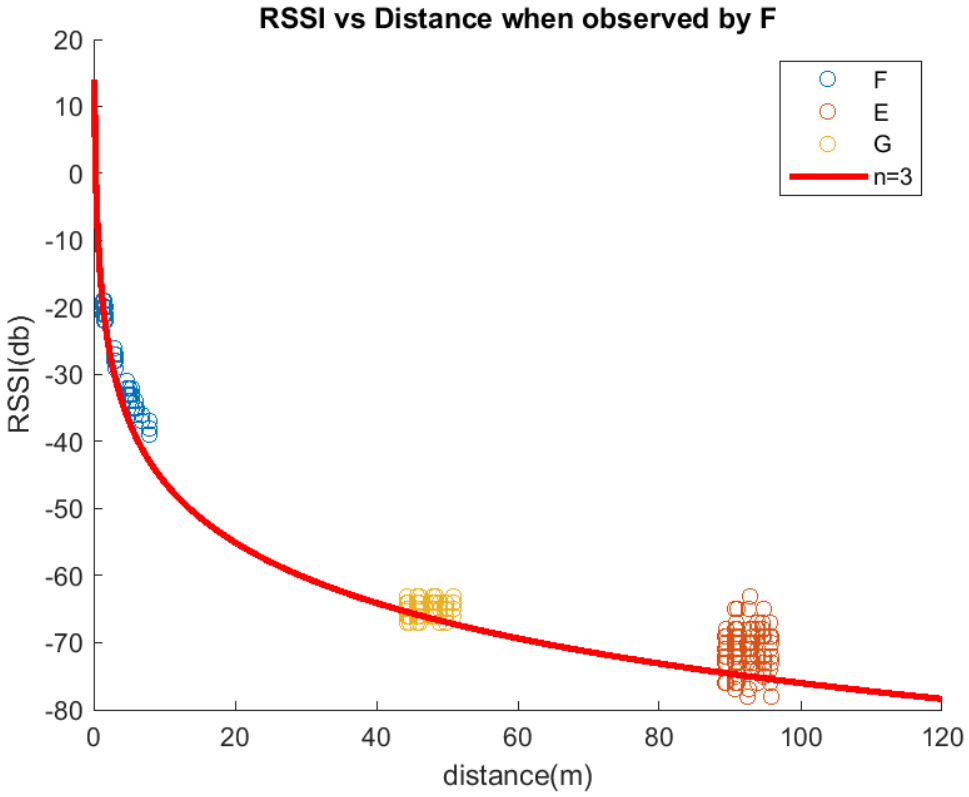


Figure 4.38: Outdoor trial 2 RSSI vs range from visual measurements for BreadCrumb F

Table 4.13: Outdoor trial 2 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	231
F	329
G	345
H	277

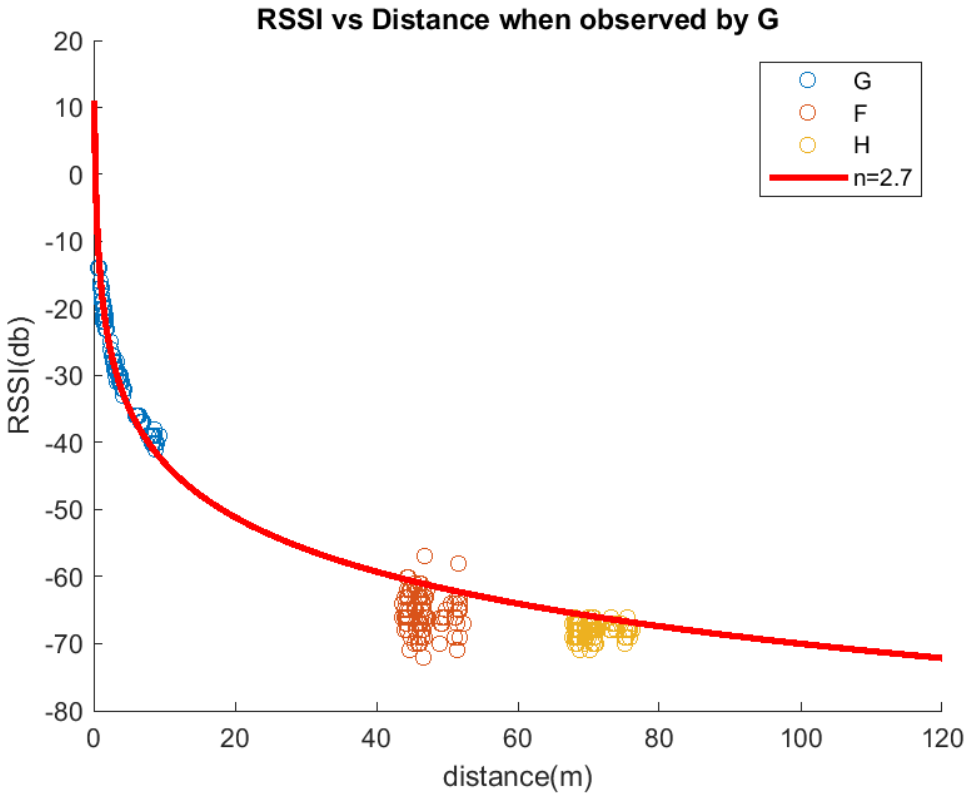


Figure 4.39: Outdoor trial 2 RSSI vs range from visual measurements for BreadCrumb G

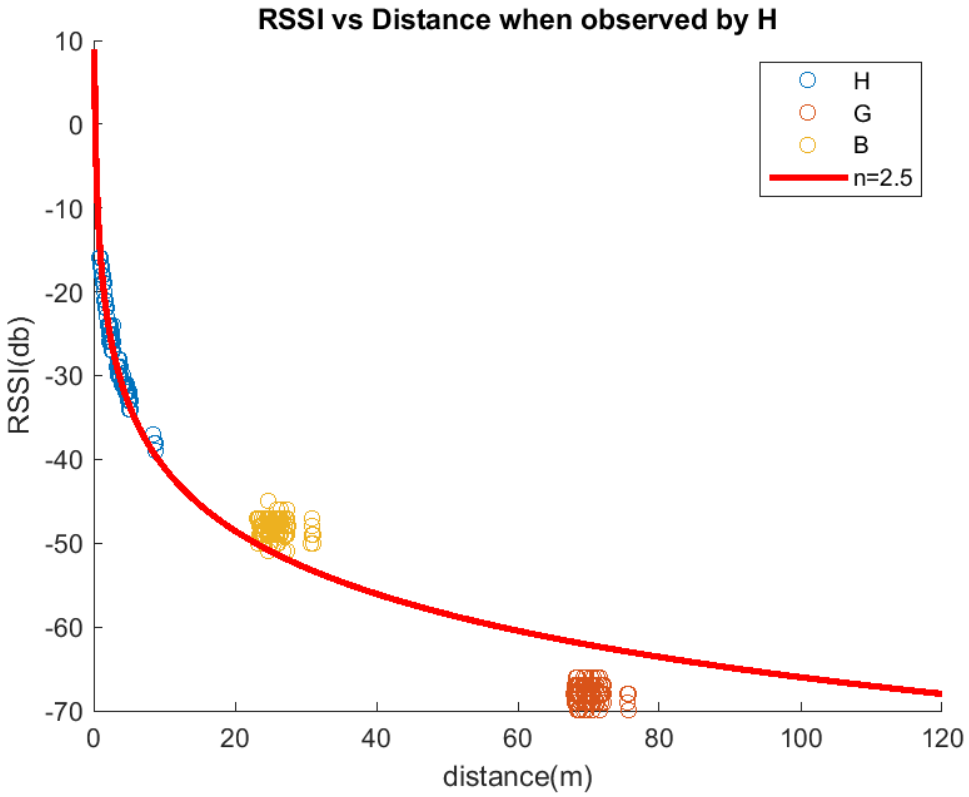


Figure 4.40: Outdoor trial 2 RSSI vs range from visual measurements for BreadCrumb H

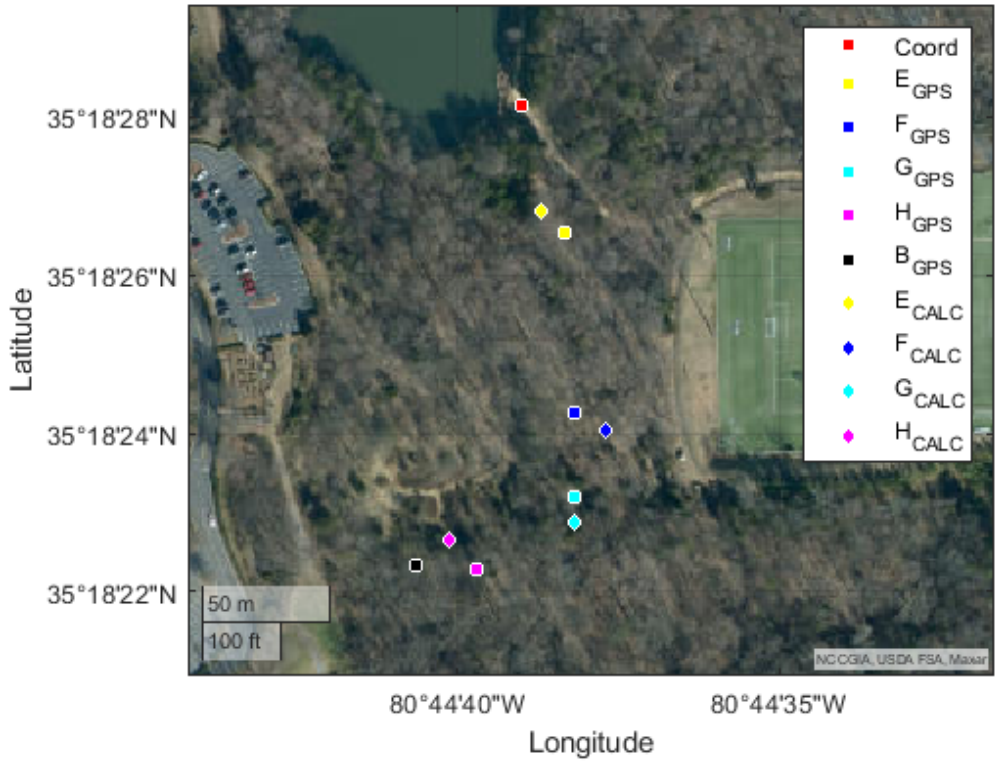


Figure 4.41: Outdoor trial 2 BreadCrumb initial location results. Mean Error: 10.3341 meters.

Table 4.14: Outdoor trial 2 BreadCrumb initial locations and errors

BreadCrumb ID	Calculated [Lat,Long] (deg)	Error (meters)
E	[35.3074431N, 80.744084W]	8.3899
F	[35.3066794N, 80.743805W]	9.3868
G	[35.3063561N, 80.743935W]	11.9523
H	[35.3062937N, 80.744478W]	11.6075

Table 4.15: Outdoor trial 2 BreadCrumb mean log-distance path loss error with learned PLEs

BreadCrumb ID	Error (meters)
E	12.5864
F	20.1462
G	22.1568
H	19.1254

4.5.3 Trial 3: Pond

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed over a trail around a pond. A mobile BreadCrumb mounted on a PVC pipe was translated along this trail. This is visualized in Figure 4.42. The number of measurements gathered in the experiment can bee seen in Table 4.16. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.43, 4.44, 4.45, and **??**. The initial locations calculated are visualized in Figure 4.47. The initial location errors are tabulated in Table 4.17. After training, the mean error of the log-distance path loss model utilizing learned PLEs was tabulated in Table 4.18.

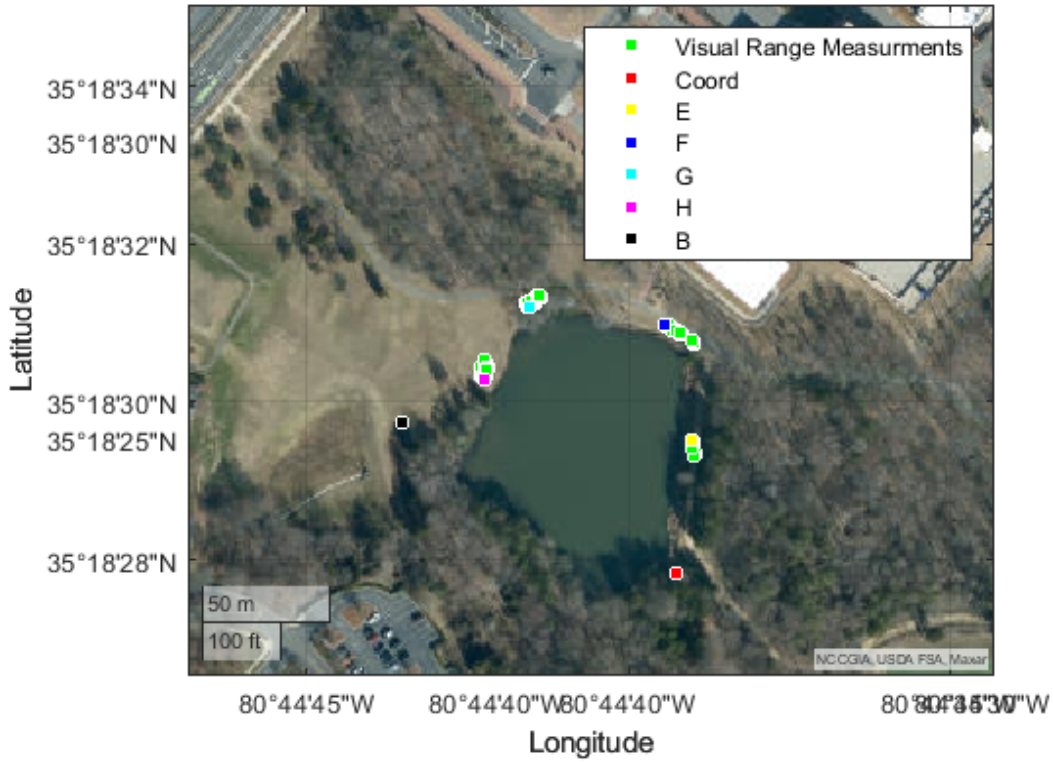


Figure 4.42: Outdoor trial 3 deployed BreadCrumbs and visual location measurements.

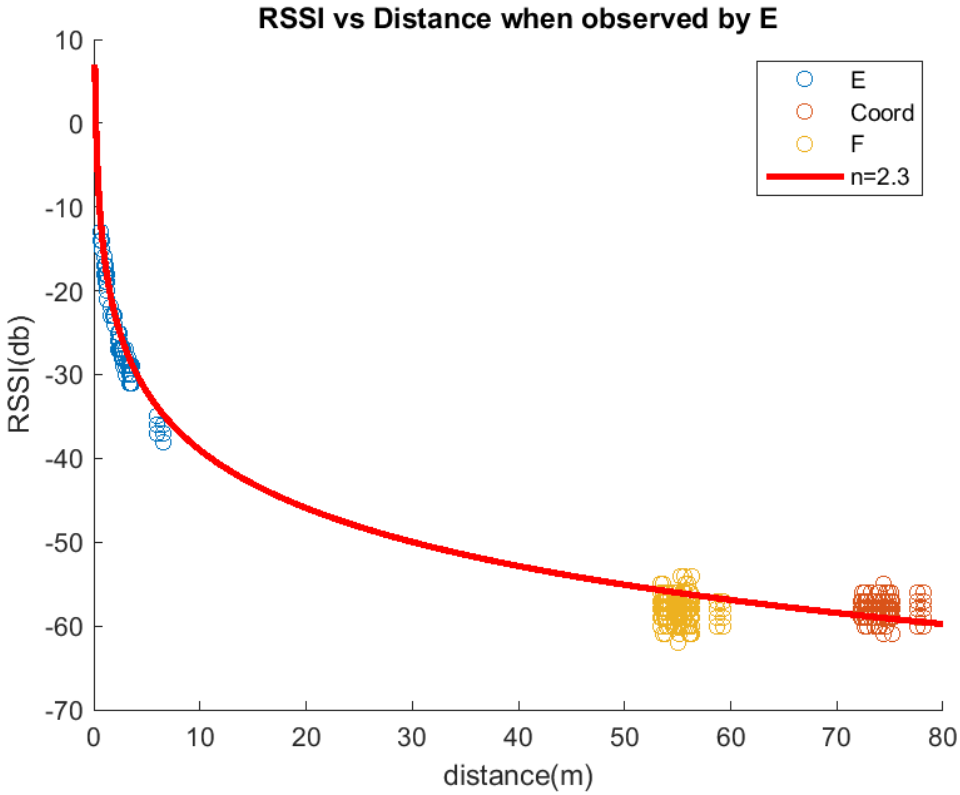


Figure 4.43: Outdoor trial 3 RSSI vs range from visual measurements for BreadCrumb E

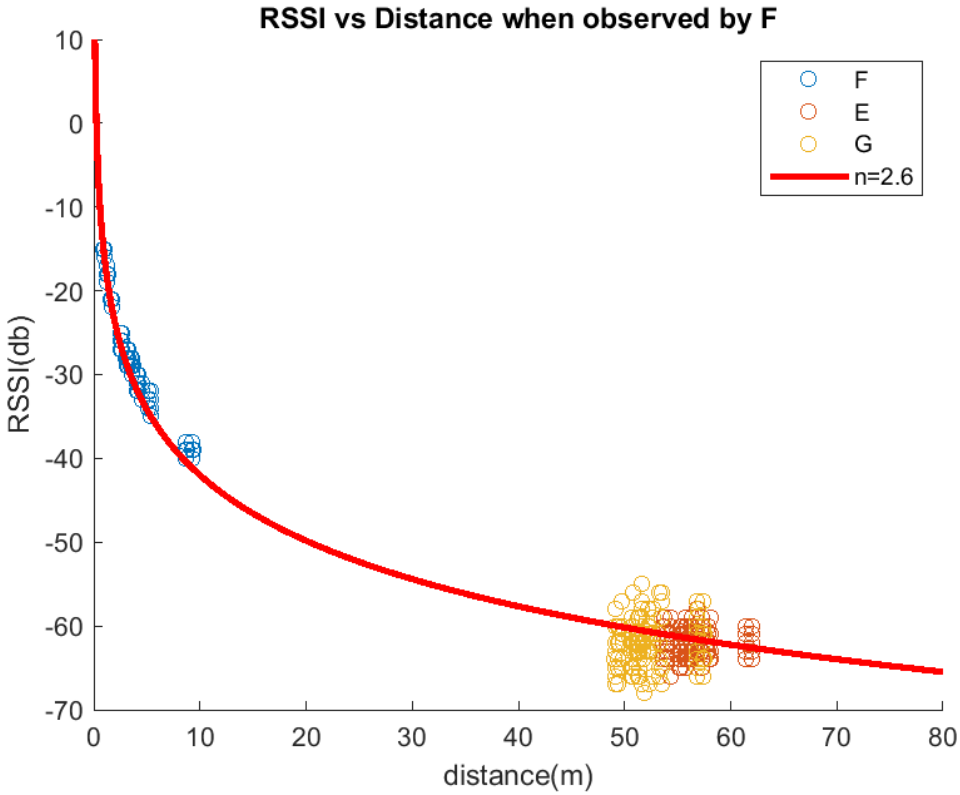


Figure 4.44: Outdoor trial 3 RSSI vs range from visual measurements for BreadCrumb F

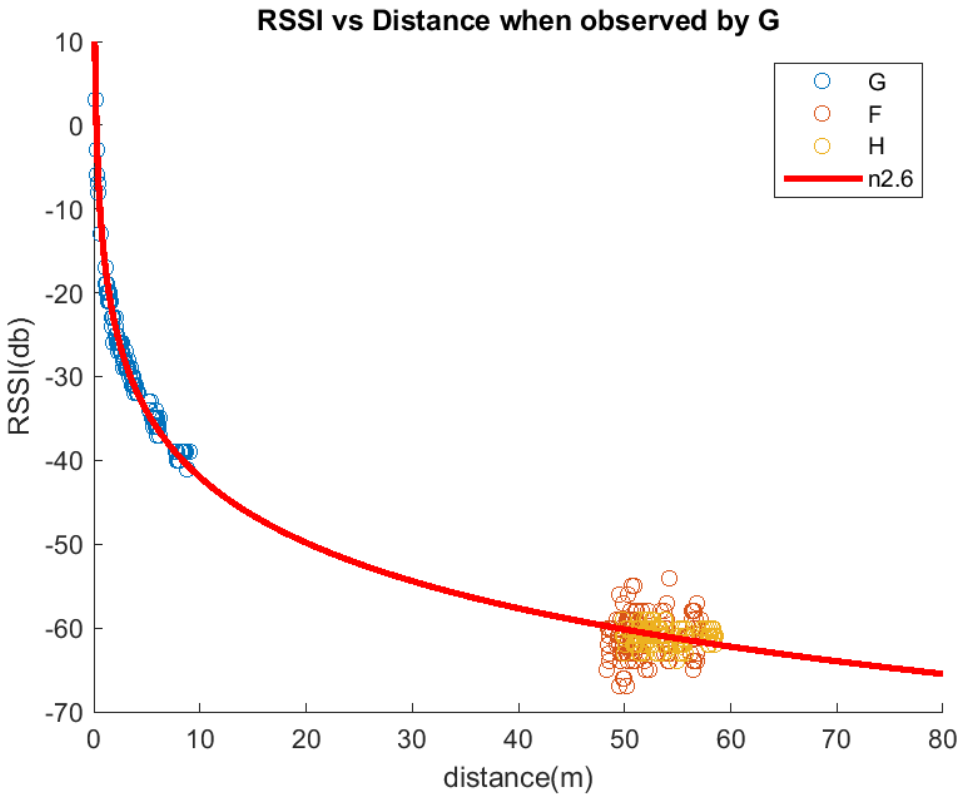


Figure 4.45: Outdoor trial 3 RSSI vs range from visual measurements for BreadCrumb G

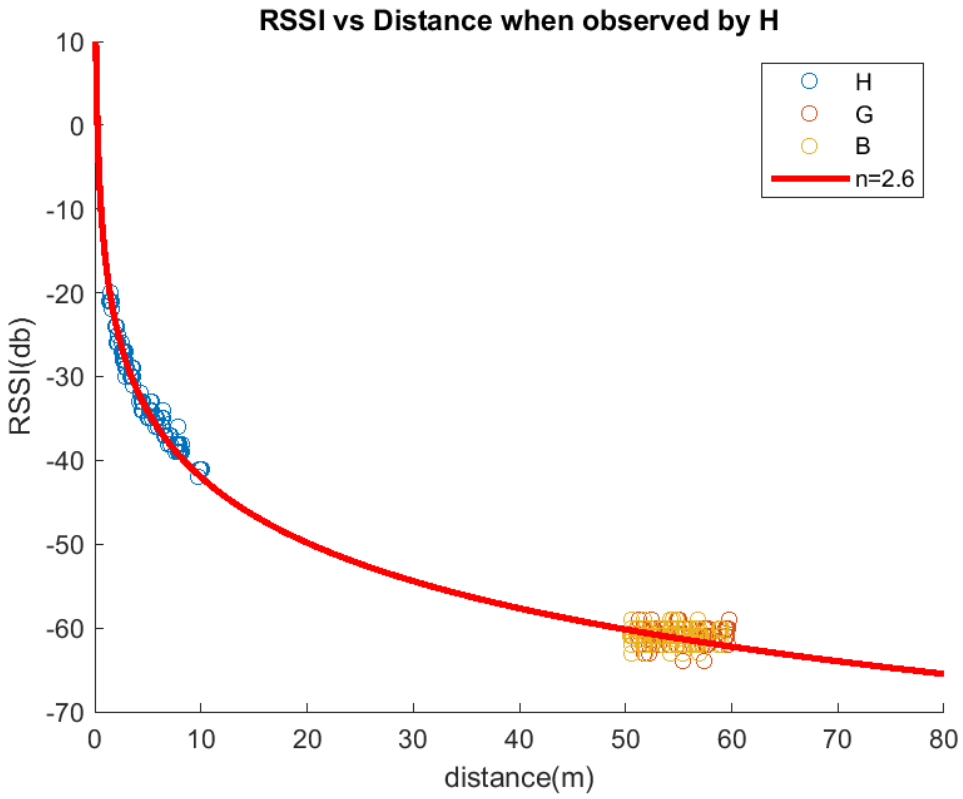


Figure 4.46: Outdoor trial 3 RSSI vs range from visual measurements for BreadCrumb H

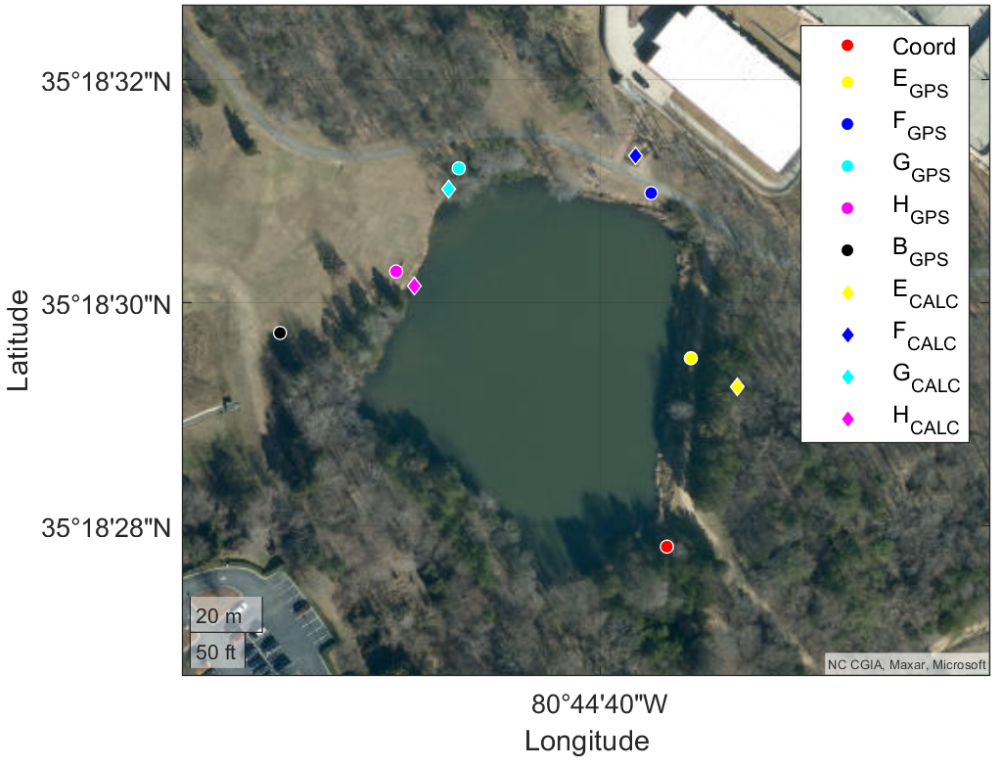


Figure 4.47: Outdoor trial 3 BreadCrumb initial location results. Mean Error: 10.1031 meters.

Table 4.16: Outdoor trial 3 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	245
F	364
G	305
H	254

Table 4.17: Outdoor trial 3 BreadCrumb initial locations and errors

BreadCrumb ID	Calculated [Lat,Long] (deg)	Error (meters)
E	[35.30812401N, 80.74402723W]	10.1580
F	[35.30869975N, 80.74433727W]	11.0695
G	[35.30861718N, 80.74490742W]	9.4979
H	[35.30837590N, 80.74501162W]	4.3810

Table 4.18: Outdoor trial 3 BreadCrumb mean log-distance path loss error with learned PLEs

BreadCrumb ID	Error (meters)
E	9.1684
F	12.9842
G	17.6812
H	14.6842

4.5.4 Trial 4: Fitness Trail P2

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed along an extended fitness trail. A mobile BreadCrumb mounted on a PVC pipe was translated along this trail. This is visualized in Figure 4.48. The number of measurements gathered in the experiment can bee seen in Table 4.19. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.49, 4.50, 4.51, and 4.52. The initial locations calculated are visualized in Figure 4.53. The initial location errors are tabulated in Table 4.20. After training, the mean error of the log-distance path loss model utilizing learned PLEs was tabulated in Table 4.21.

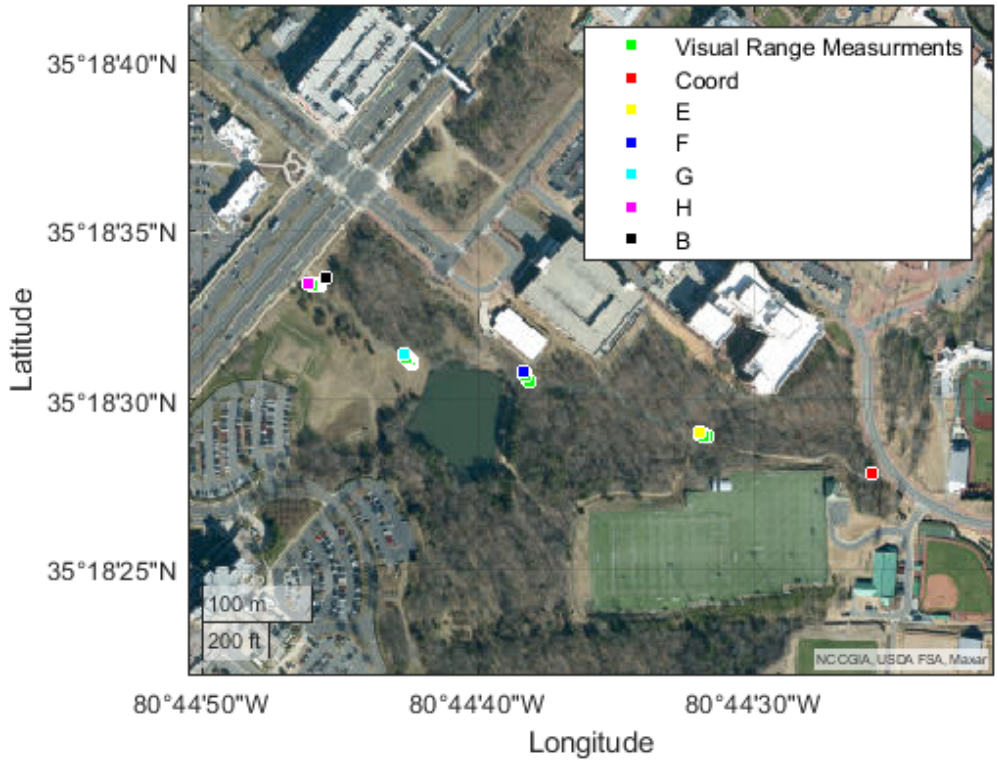


Figure 4.48: Outdoor trial 3 deployed BreadCrumbs and visual location measurements.

Table 4.19: Outdoor trial 4 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	450
F	574
G	541
H	320

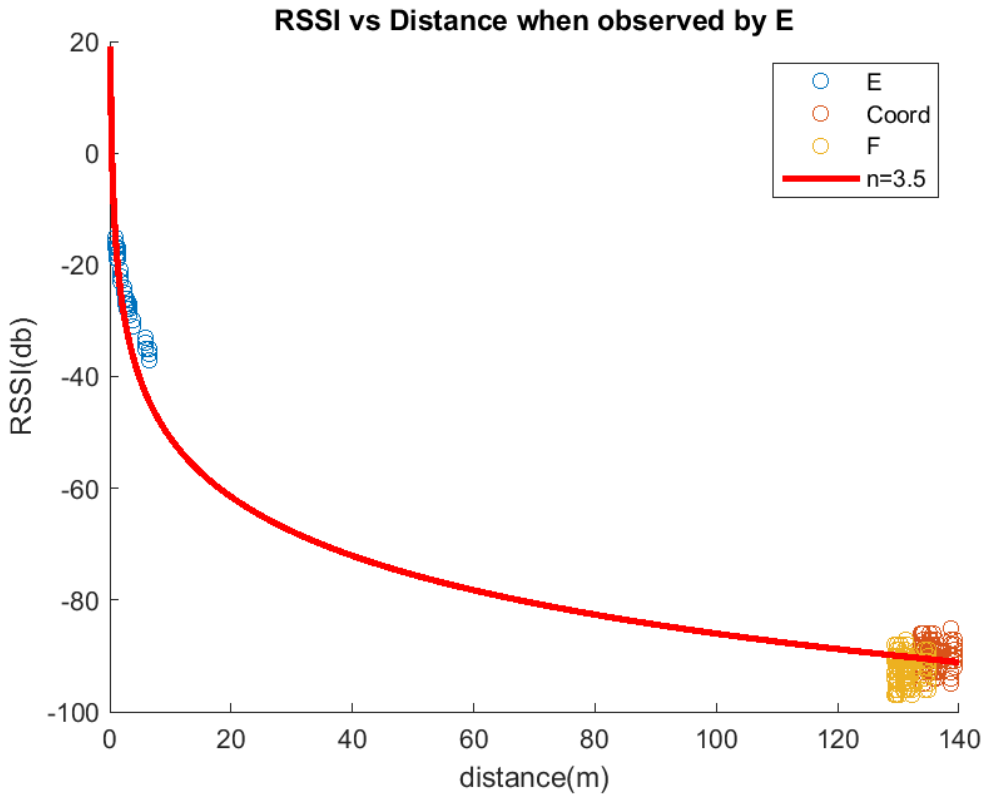


Figure 4.49: Outdoor trial 4 RSSI vs range from visual measurements for BreadCrumb E

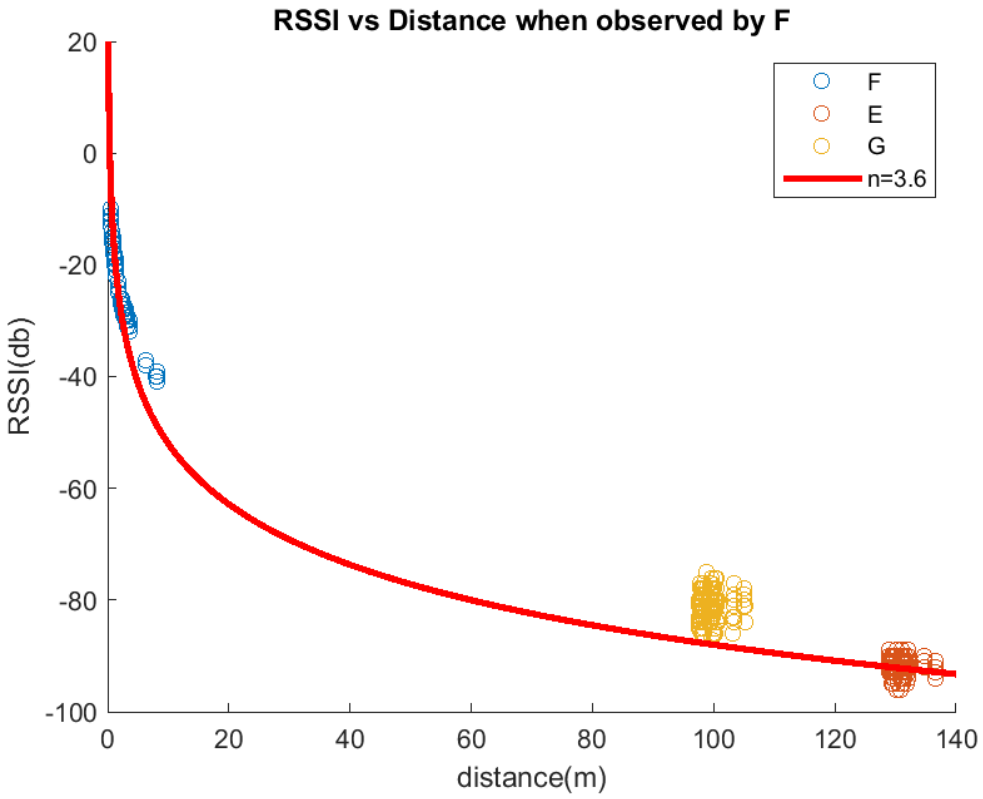


Figure 4.50: Outdoor trial 4 RSSI vs range from visual measurements for BreadCrumb F

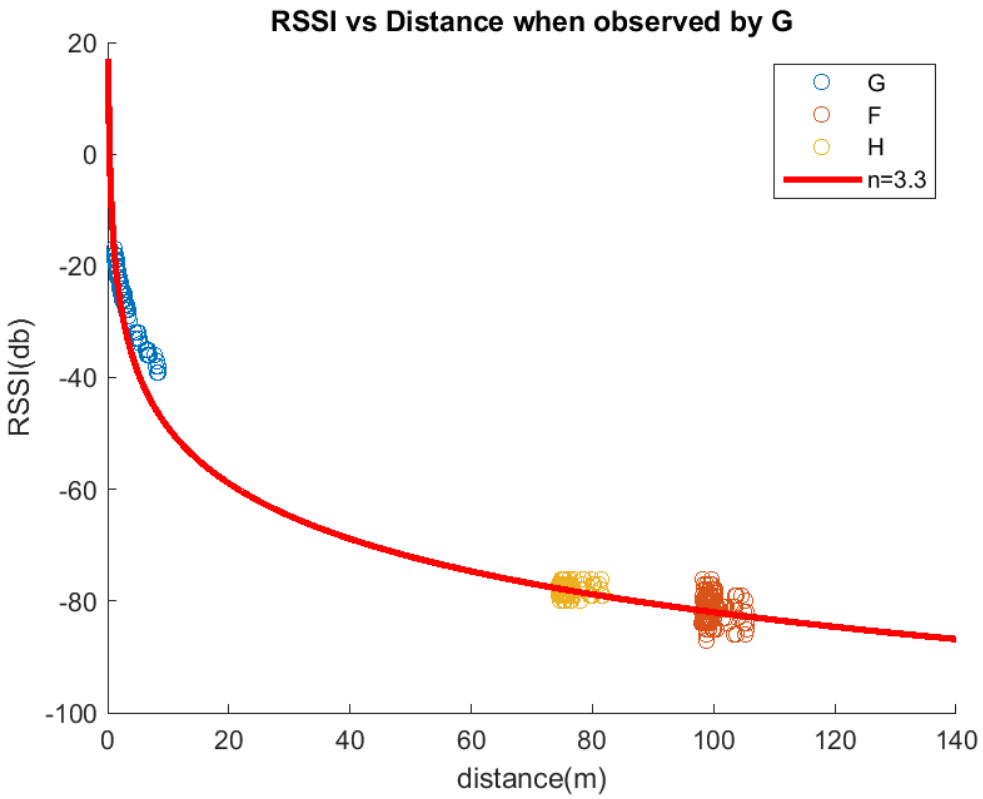


Figure 4.51: Outdoor trial 4 RSSI vs range from visual measurements for BreadCrumb G

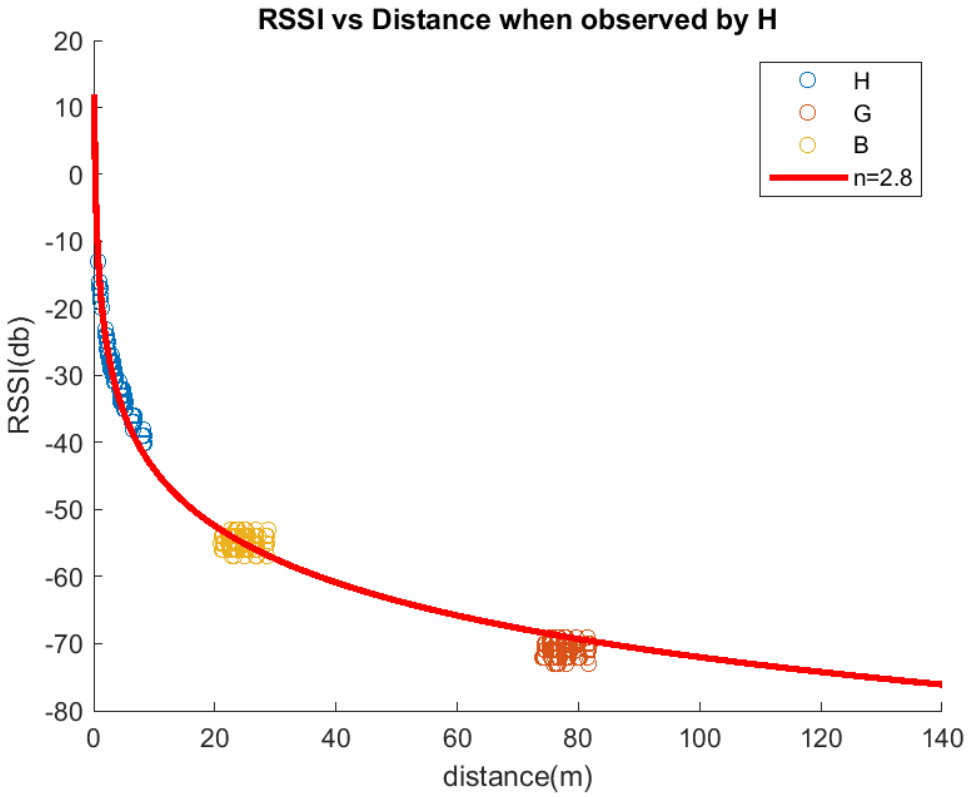


Figure 4.52: Outdoor trial 4 RSSI vs range from visual measurements for BreadCrumb H

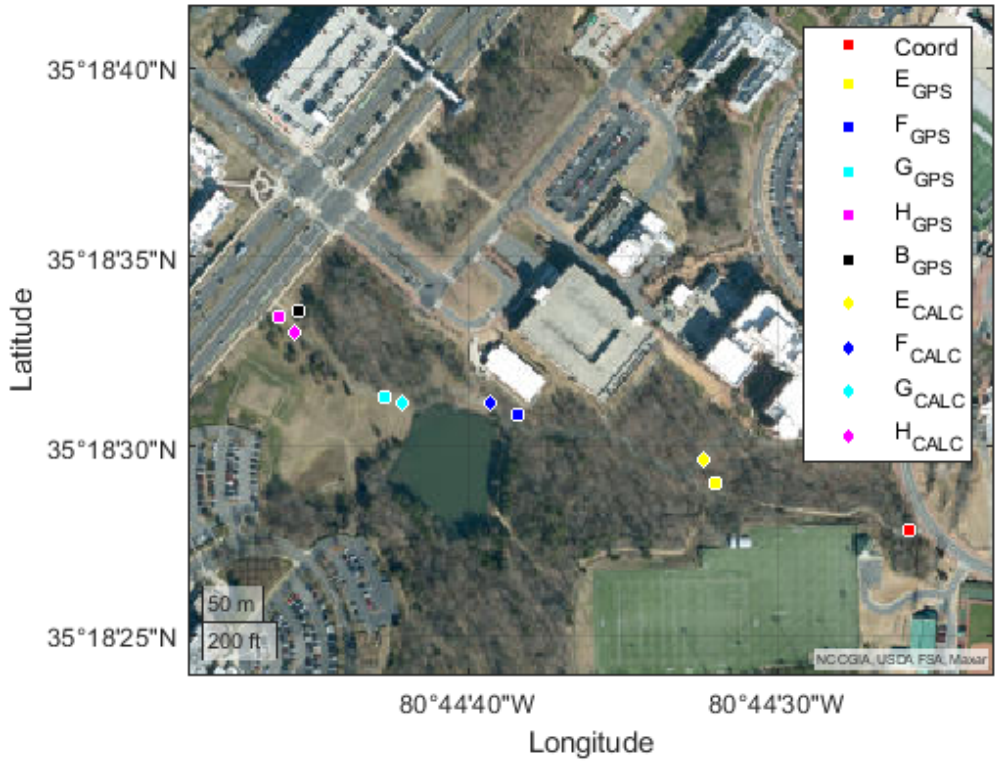


Figure 4.53: Outdoor trial 4 BreadCrumb initial location results. Mean Error: 15.5710 meters.

Table 4.20: Outdoor trial 4 BreadCrumb initial locations and errors

BreadCrumb ID	Calculated [Lat,Long] (deg)	Error (meters)
E	[35.308236N, 80.742324W]	20.9356
F	[35.308645N, 80.744248W]	17.7398
G	[35.308645N, 80.745045W]	10.3671
H	[35.309165N, 80.746012W]	13.2464

Table 4.21: Outdoor trial 4 BreadCrumb mean log-distance path loss error with learned PLE

BreadCrumb ID	Error (meters)
E	56.9841
F	38.1586
G	17.4159
H	19.8956

4.5.5 Trial 5: Woods P2

Five BreadCrumbs V2s and one BreadCrumb V1 were deployed along an trail in the woods that led to the perimeter of a soccer field. A mobile BreadCrumb mounted on a PVC pipe was translated along this trail. This is visualized in Figure 4.54. The number of measurements gathered in the experiment can bee seen in Table 4.22. The distribution of the visual range measurements and measured RSSI is visualised in Figure 4.55, 4.56, 4.57, and 4.58. The initial locations calculated are visualized in Figure 4.59. The initial location errors are tabulated in Table 4.23. After training, the mean error of the log-distance path loss model utilizing learned PLEs was tabulated in Table 4.24.

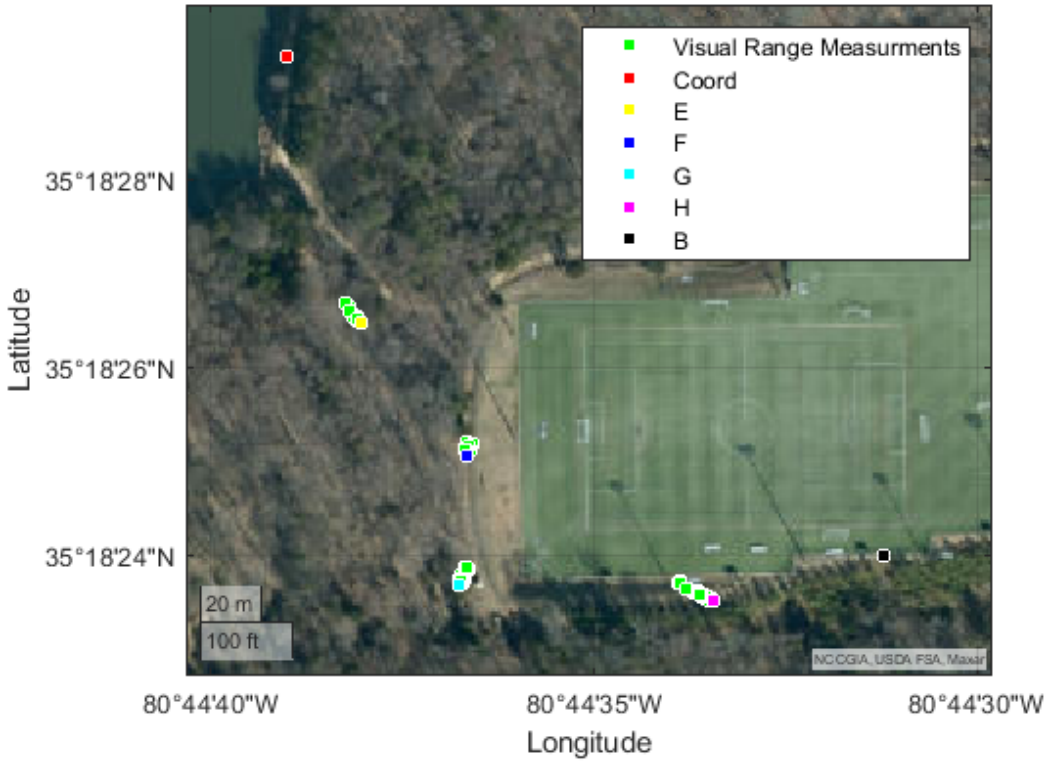


Figure 4.54: Outdoor trial 5 deployed BreadCrumbs and visual location measurements.

Table 4.22: Outdoor trial 5 visual location measurement/RSSI measurement pairs per BreadCrumb

BreadCrumb ID	# measurements pairs
E	320
F	417
G	402
H	295

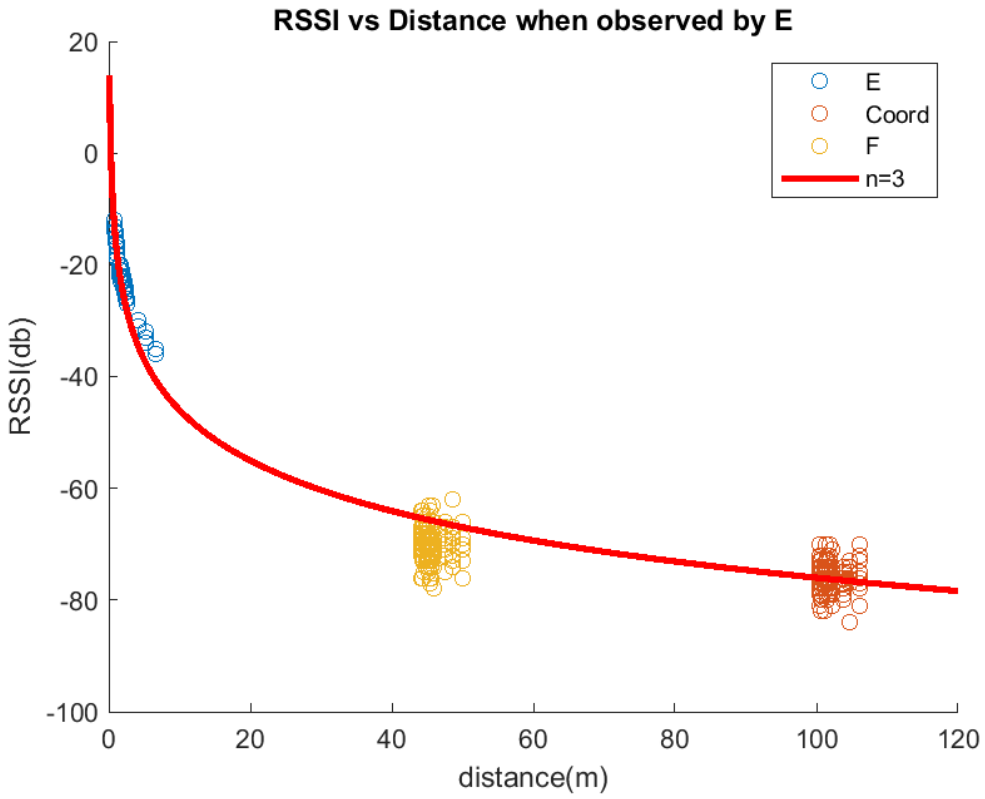


Figure 4.55: Outdoor trial 5 RSSI vs range from visual measurements for Bread-Crumbs E

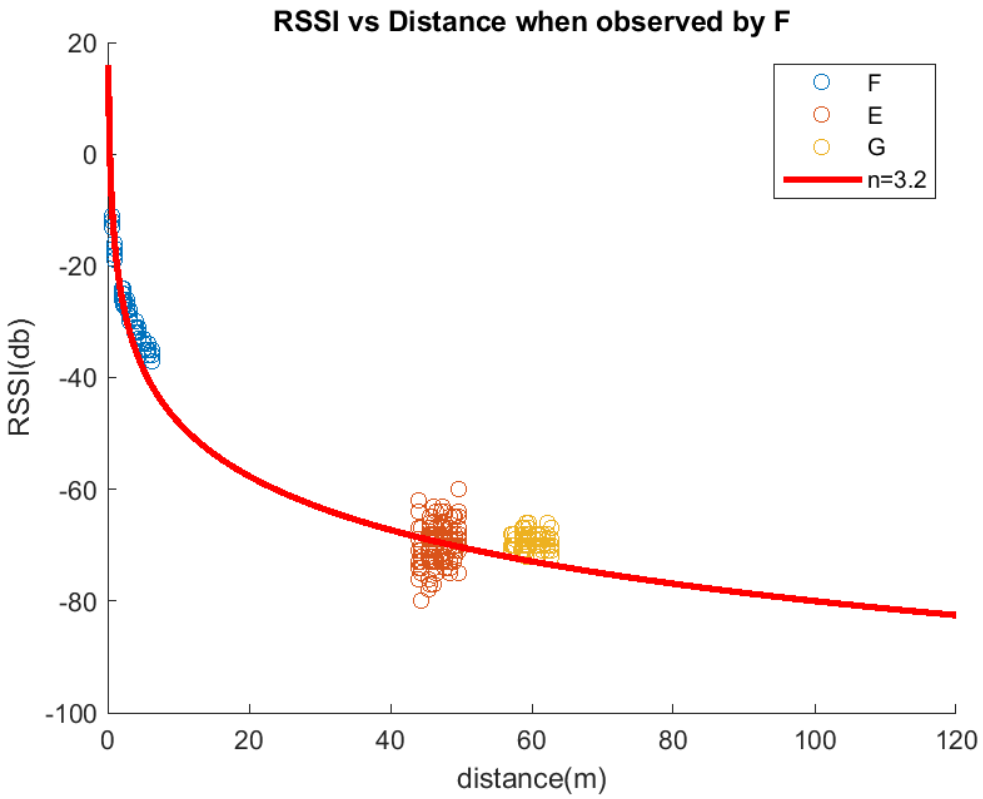


Figure 4.56: Outdoor trial 5 RSSI vs range from visual measurements for Bread-Crumbs F

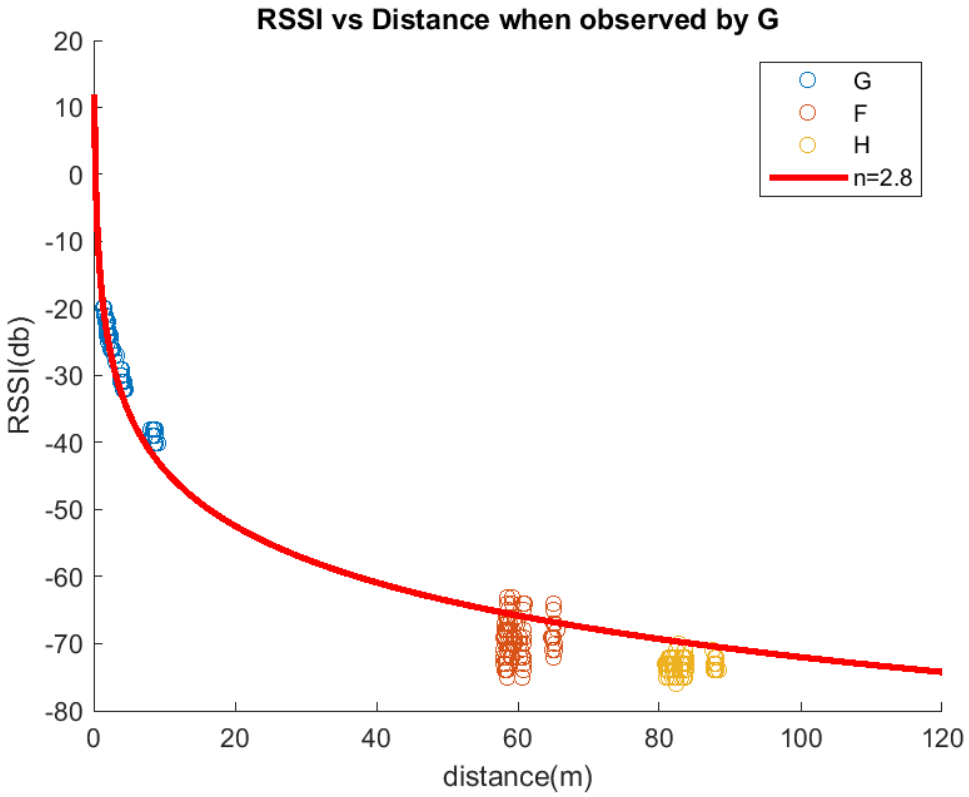


Figure 4.57: Outdoor trial 5 RSSI vs range from visual measurements for Bread-Crumbs G

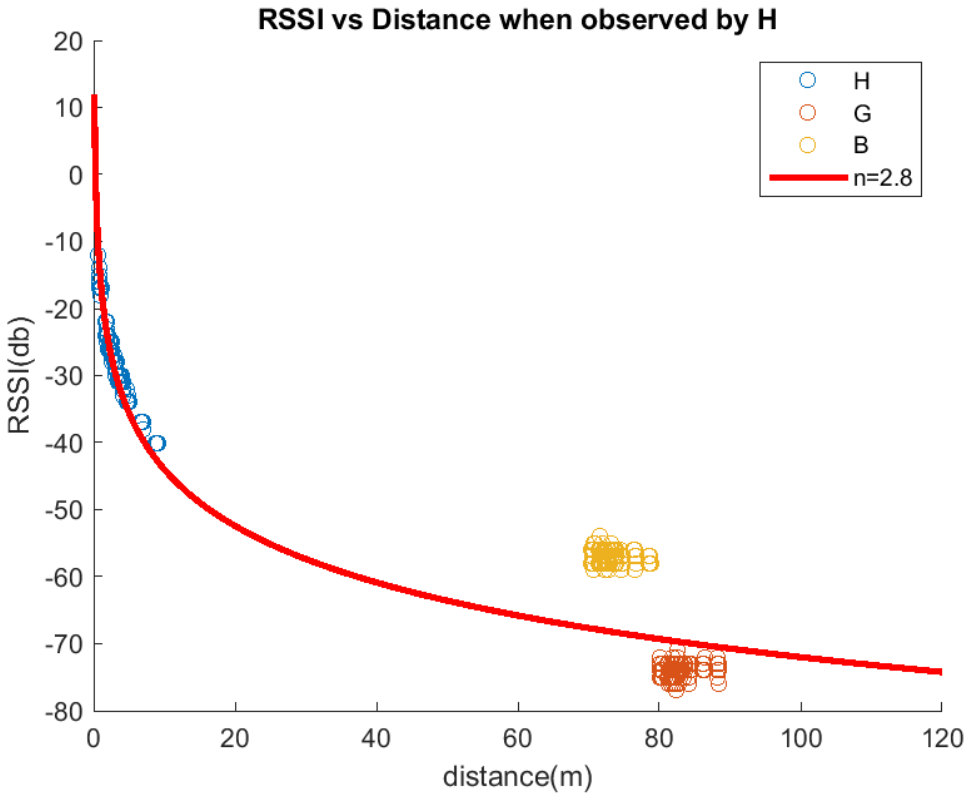


Figure 4.58: Outdoor trial 5 RSSI vs range from visual measurements for Bread-Crumbs H

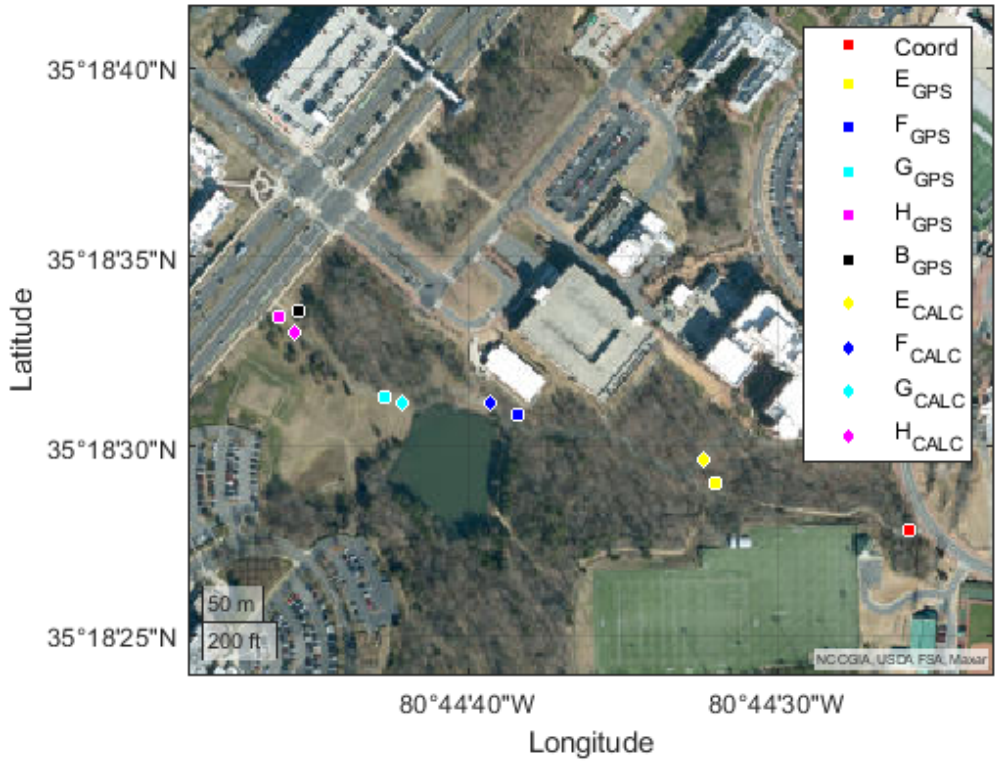


Figure 4.59: Outdoor trial 5 BreadCrumb initial location results. Mean Error: 8.08072903 meters.

Table 4.23: Outdoor trial 5 BreadCrumb initial locations and errors

BreadCrumb ID	Calculated [Lat,Long] (deg)	Error (meters)
E	[35.307310N, 80.743790W]	7.8812
F	[35.306901N, 80.743434W]	7.1829
G	[35.306684N, 80.743688W]	12.2296
H	[35.306486N, 80.742574W]	5.02855

Table 4.24: Indoor aggregate mean initial location Error for trials 1 through 3 for BreadCrumbs E through H.

BreadCrumb ID	Error (meters)
E	9.1981
F	16.4892
G	118.1655
H	19.6564

4.6 Results Aggregate Discussion

The aggregated results are grouped based on the experimentation done inside and outside. Table 4.25 shows the aggregate initial location determination error for each BreadCrumb located indoors. Indoor trial 1 and trial 2 show promising results for indoor location determination. In fact, they are on par with some contemporary localization algorithms. Trial 3 shows an increased error on the meters scale, which while not on an appropriate level for precise localization purposes, it does provide enough reasonable accuracy for initial path planning, which is the purpose of the BreadCrumb trial at this stage. It is also important to note that the map size of trial 3 is much larger than trials 1 and 2. For example, trial 2 took place over a roughly 10x10 meter area where as trial 3 took place over a roughly 80x80 meter area. Therefore, the change in scale of error is acceptable with the proportional change in scale of map size for our purposes of initial location determination for path planning. This also generally holds for the aggregate mean log-distance path loss error with learned PLEs shown in Table 4.26. Trials 1 and 2 show a reasonable level or error for distance measurements estimated form the log-distance path loss model. There is a spike in error for trail 3, however since the model utilized is an empirical one, it is difficult to find comparative results for this specific environment. All we have to consistently compare against is the physically measured distance vs the estimated distance. That

being said, we believe the scale of the error to be well within an acceptable range for indoor as a distance estimation measurement. The results from the outdoor experimentation differ from the indoor experimentation quite significantly. Table 4.27 shows a range of relatively high initial location errors for the BreadCrumb with the highest being 20.9356 meters. Examining the aggregate mean log-distance path loss error, as shown in Table 4.28, with learned PLEs also shows significant error. We've believe this to be the result of two primary factors. First, the distance between each BreadCrumbs in these outdoor experiments was upwards of 60 meters on average. This is a significant increase over the indoor experimentation. When examining the log-distance path loss model, we can see that at higher distances the resolution of RSSI measurements decreases. Therefore, slight changes in the PLEs used can lead to massive amounts of introduced error at this distance scale. Hence, one possible source of these larger errors is that the error introduced by the learned PLEs is simply greater than the error introduced by using a constant PLE in these complex outdoor environments. A possible solution to this would be to increase the power of our transmitters or decrease the distance between BreadCrumbs. That being said, there is a second source of possible error. All of the measured locations used to compare against were gathered using a GPS device with an error range of 7 to 25 meters. Given the distance the outdoor experimentation was done over, this GPS device was the only method available for gathering any comparative forms of measurement. While our results generally fall within this range of error, it is extremely difficult to determine the empirical impact this error introduced to our error comparisons. Therefore, it is difficult to strictly quantify whether the outdoor experiments can be considered a success or failure and further testing with a method for more accurate ground truth determination may be required. Examining the current results, we consider the initial location error to be within an acceptable range for initial path planning purposes. While an error of 10 meters is significant, when put on a map with a scale of hundreds

of meters, we believe it to be acceptable for solely path planning purposes. However, again, to make a more confident determination we believe higher accuracy ground truth measurements would be required. Finally, the season during which the outdoor experimentation was done may also have played a role in these results. Data was primarily gathered during the fall and winter season, meaning that the amount of foliage was vastly decreased. During the spring and summer season we expect the amount of foliage to increases, leading to an increase in the scattering effect on our radio wave transmissions. The additive error introduced by this effect may cause of methods PLE determination to produce more accurate results than a statically determined PLE.

Table 4.25: Indoor aggregate mean initial location error for trials 1 through 3 for BreadCrumbs E through H.

BreadCrumb ID	T1 Error (m)	T2 Error (m)	T3 Error (m)
E	0.8954	0.5298	8.7884
F	1.0564	0.3328	4.4498
G	N/A	1.1378	6.3545
H	N/A	0.3725	5.2172

Table 4.26: Indoor aggregate mean log-distance path loss error with learned PLEs for trials 1 through 5 for BreadCrumbs E through H.

BreadCrumb ID	T1 Error (m)	T2 Error (m)	T3 Error (m)
E	0.9654	0.6543	4.2641
F	0.6458	1.3251	2.4569
G	N/A	1.2254	2.7387
H	N/A	1.1378	1.5735

Table 4.27: Outdoor aggregate mean initial location error for trials 1 through 5 for BreadCrumbs E through H.

BreadCrumb ID	T1 Error (m)	T2 Error (m)	T3 Error (m)	T4 Error(m)	T5 Error (m)
E	8.8877	8.3899	10.1580	20.9356	7.8812
F	7.7647	9.3868	11.0695	17.7398	7.1829
G	8.9576	11.9523	9.4979	10.3671	12.2296
H	14.8020	11.6075	4.3810	13.2464	5.02855

Table 4.28: Outdoor aggregate mean log-distance path loss error with learned PLEs for trials 1 through 5 for BreadCrumbs E through H.

BreadCrumb ID	T1 Error (m)	T2 Error (m)	T3 Error (m)	T4 Error(m)	T5 Error (m)
E	6.5423	12.5864	9.1684	56.9841	9.1981
F	12.4583	20.1462	12.9842	38.1586	16.4892
G	3.1254	22.1568	17.6812	17.4159	118.1655
H	9.1253	19.1254	14.6842	19.8956	19.6565

CHAPTER 5: Conclusion

5.1 Conclusion

A network of devices dubbed BreadCrumbs was developed with the purpose of assisting autonomous robots operating in GPS denied environments with poor visually recognizable landmarks. To this end, the BreadCrumbs network determines and provides the following to autonomous robots: location information for path planning, range measurements utilizing the log-distance path loss model, generalized path loss exponent for the location where the BreadCrumb was deployed to be used for reducing the error of distance estimation from the log-distance path loss model. To provide the PLEs for the log-distance path loss model, a novel application of a DDPG was utilized and trained with visual location measurements/RSSI range measurement pairs gathered during run-time by each BreadCrumb.

Through empirical analysis, this work has demonstrated the feasibility of such a network. Experimentation was done to determine the accuracy of the initial localization method and accuracy of the distance estimation acquired with the learned PLE. The system showed an successful improvement on indoor localization and distance estimation over measurements gained through empirical analysis. However, the results outdoors were not as significant. We believe the reason for this is two fold. First, the scale of deployment distance was much greater, leading to a decrease in log-distance path loss model resolution an an increase in the error introduced by the learned PLE. Second, the error of the GPS utilized to ascertain the location of the outdoor BreadCrumbs for comparison was too high to be able to properly compare the results against. That additive error may be skewing the outdoor results, however given the circumstances this was the only method for gathering data for compari-

son against. Hence, while the error rate for the outdoor experimentation appeared quite high, it is still too early to make a definitive statement about the success of the network outdoor without further ground truth measurements for comparison.

5.2 Future Work

One of the primary avenues of future work involves examining the impact of the BreadCrumb network when utilized for a localization algorithm, such as with Extended Kalman Filter Simultaneous Localization and Mapping. This research was focused on the development and implementation of the BreadCrumb network and an analysis on its ability to provide better range measurements. While the current work has shown that the network provides a higher accuracy source of RSSI measurements through environmentally determining a PLE, it is of interest to see the impact that will have on actual localization algorithms. A comparison of other techniques is also desirable. Including the use of technologies such as ultra-wide band. Comparing the accuracy of the BreadCrumb networks RSSI based distance estimation to ultra-wide band distance estimation would be an interesting avenue for future work as it would provide an avenue for comparison against other works that is not purely empirical. Furthermore, additional testing in outdoor environments during different seasons is desired. With this scale of experimentation, we we believe that foliage will play a prime factor by introducing error from the scattering effect. Hence, future works seeks to conduct experimentation during seasons where foliage will be most present and introduce to most amount of error.

REFERENCES

[1] "Rheinmetall defence: Shorten the chain of rescue." https://www.rheinmetall-defence.com/en/rheinmetall_defence/public_relations/themen_im_fokus/ugv/index.php. (Accessed on 03/05/2021).

[2] K. Russell, M. Schader, K. Andrea, and S. Luke, "Swarm robot foraging with wireless sensor motes," in *AAMAS*, 2015.

[3] R. Ross and R. Hoque, "Augmenting GPS with Geolocated Fiducials to Improve Accuracy for Mobile Robot Applications," *Applied Sciences*, vol. 10, no. 1, 2020.

[4] "The inverse square law." https://www.cyberphysics.co.uk/general_pages/inverse_square/inverse_square.htm. (Accessed on 03/05/2021).

[5] F. E., "Mobile Radio Propagation Prediction for Two Different Districts in Mosul-City," in *MATLAB - A Fundamental Tool for Scientific Computing and Engineering Applications - Volume 2*, 2012.

[6] "What are the different types of fading? - everything rf." https://www.everythingrf.com/community/what-are-the-different-types-of-fading. (Accessed on 03/05/2021).

[7] S. Y. Seidel and T. S. Rappaport, "914 MHz Path Loss Prediction Models for Indoor Wireless Communications in Multifloored Buildings," *IEEE Transactions on Antennas and Propagation*, 1992.

[8] D. Grabowsky, "Utilizing orientation estimation from trilaterated poses over time to improve ro-ekf slam," Master's thesis, 2018.

[9] "Rocky mountain national park (u.s. national park service)." https://www.nps.gov/romo/index.htm.

[10] "Carlsbad caverns national park (u.s. national park service)." https://www.nps.gov/cave/index.htm.

[11] J. Young, "Will 2018 be a really bad year for earthquakes?: Beyond infinity podcasts." https://beyondinfinity.com.au/will-2018-be-a-really-bad-year-for-earthquakes/, Dec 2017.

[12] Headquarters, Department of the Army, Washington, NC, *Subterranean Operations*, atp 3-21.51 ed., 2019.

[13] "Darpa subterranean challenge." https://www.subtchallenge.com/index.html. (Accessed on 11/21/2020).

[14] S. Sadowski and P. Spachos, "RSSI-Based Indoor Localization with the Internet of Things," *IEEE Access*, 2018.

[15] "Bluetooth core specifications." https://www.bluetooth.com/specifications/bluetooth-core-specification/. (Accessed on 03/05/2021).

[16] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE Journal on Selected Areas in Communications*, 2015.

[17] A. A. Kalbandhe and S. C. Patil, "Indoor Positioning System using Bluetooth Low Energy," in *International Conference on Computing, Analytics and Security Trends, CAST 2016*, 2017.

[18] C. Yang and H. R. Shao, "WiFi-based indoor positioning," *IEEE Communications Magazine*, 2015.

[19] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor WiFi GraphSLAM," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1038–1043, 2011.

[20] A. T. Parameswaran, M. I. Husain, and S. Upadhyaya, "Is RSSI a reliable parameter in sensor localization algorithms - an experimental study," *IEEE International Symposium on Reliable Distributed Systems*, 2009.

[21] D. Grabowsky, J. M. Conrad, and A. F. Browne, "A breadcrumb system for assisting outdoor autonomous robots with path identification and localization," in *2021 SoutheastCon*, pp. 1–6, 2021.

[22] Qian Dong and W. Dargie, "Evaluation of the reliability of rssi for indoor localization," in *2012 International Conference on Wireless Communications in Underground and Confined Areas*, pp. 1–6, 2012.

[23] P. Barsocchi, S. Lenzi, S. Chessa, and G. Giunta, "A novel approach to indoor rssi localization by automatic calibration of the wireless propagation model," in *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pp. 1–5, 2009.

[24] R. A. Russell, "Mobile robot guidance using a short-lived heat trail," *Robotica*, 1993.

[25] A. Russell, D. Thiel, and A. Mackay-Sim, "Sensing odour trails for mobile robot navigation," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 2672–2677 vol.3, May 1994.

[26] J. S. Russell, M. Ye, B. D. O. Anderson, H. Hmam, and P. Sarunic, "Cooperative Localisation of a GPS-Denied UAV using Direction-of-Arrival Measurements," *IEEE Transactions on Aerospace and Electronic Systems*, 2019.

[27] S. Singh and P. B. Sujit, "Landmarks based path planning for UAVs in GPS-denied areas," *IFAC-PapersOnLine*, 2016.

[28] R. T. Vaughan, K. Støy, G. S. Sukhatme, and M. J. Matarić, *Blazing a trail: Insect-inspired resource transportation by a robot team*, pp. 111–120. Tokyo: Springer Japan, 2000.

[29] J. Jung, J. H. Li, H. T. Choi, and H. Myung, "Localization of AUVs using visual information of underwater structures and artificial landmarks," *Intelligent Service Robotics*, 2017.

[30] V. Magnago, P. Corbal, G. P. Picco, L. Palopoli, and D. Fontanelli, "Robot Localization via Odometry-assisted Ultra-wideband Ranging with Stochastic Guarantees," no. Section VI, pp. 1607–1613, 2019.

[31] T. Mitterer, H. Gietler, L. M. Faller, and H. Zangl, "Artificial landmarks for trusted localization of autonomous vehicles based on magnetic sensors," *Sensors (Switzerland)*, 2019.

[32] H. Goforth and S. Lucey, "GPS-denied UAV localization using pre-existing satellite imagery," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2019.

[33] M. Chaudhary, "A wireless sensor network breadcrumb trail for an autonomous vehicle," Master's thesis, 2018.

[34] M. Klann, T. Riedel, H. Gellersen, C. Fischer, M. Oppenheim, P. Lukowicz, G. Pirkl, K. Kunze, M. Beuster, M. Beigl, O. Visser, and M. Gerling, "LifeNet: an Ad-hoc Sensor Network and Wearable System to Provide Firefighters with Navigation Support," *Adjunct Proc Ubicomp 2007*, vol. M, no. 1, pp. 124–127, 2007.

[35] J. Wilson, V. Bhargava, A. Redfern, and P. Wright, "A wireless sensor network and incident command interface for urban firefighting," *Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous 2007*, 2007.

[36] H. Liu, J. Li, Z. Xie, S. Lin, K. Whitehouse, J. A. Stankovic, and D. Siu, "Automatic and robust breadcrumb system deployment for indoor firefighter applications," *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*, p. 21, 2010.

[37] T. Lai, W.-J. Chen, K.-H. Li, P. Huang, and H.-H. Chu, "TriopusNet: automating wireless sensor network deployment and replacement in pipeline monitoring," *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pp. 61–72, 2012.

[38] A. Giroux and J. Frolik, "In situ channel modeling for real-time repeater node placement," in *2016 IEEE 17th Annual Wireless and Microwave Technology Conference, WAMICON 2016*, 2016.

[39] M. R. Souryal, J. Geissbuehler, L. E. Miller, and N. Moayeri, "Real-time deployment of multihop relays for range extension," *Proceedings of the 5th international conference on Mobile systems, applications and services - MobiSys '07*, p. 85, 2007.

[40] H. T. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, 1946.

[41] B. Sklar, "Rayleigh fading channels in mobile digital communication systems Part I: Characterization," *IEEE Communications Magazine*, 1997.

[42] T. S. Rappaport, *Wireless Communications: Principles and Practice (Prentice Hall Communications Engineering & Emerging Technologies Series)*. 2001.

[43] V. Malyavej, W. Kumkeaw, and M. Aorpimai, "Indoor robot localization by rssi/imu sensor fusion," in *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pp. 1–6, 2013.

[44] S. Sadowski and P. Spachos, "Rssi-based indoor localization with the internet of things," *IEEE Access*, vol. 6, pp. 30149–30161, 2018.

[45] B. Yang, L. Guo, R. Guo, M. Zhao, and T. Zhao, "A novel trilateration algorithm for rssi-based indoor localization," *IEEE Sensors Journal*, vol. 20, no. 14, pp. 8164–8172, 2020.

[46] M. Pajovic, P. Orlik, T. Koike-Akino, K. J. Kim, H. Aikawa, and T. Hori, "An unsupervised indoor localization method based on received signal strength (rss) measurements," in *2015 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2015.

[47] M. Kennedy, *Understanding map projections*, vol. 8. ESRI, 2000.

[48] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[49] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, 2015.

[50] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *33rd International Conference on Machine Learning, ICML 2016*, 2016.

[51] A. V. Bernstein and E. V. Burnaev, "Reinforcement learning in computer vision," 2018.

[52] D. Grabowsky, J. M. Conrad, and A. F. Browne, "Limited log-distance path loss model path loss exponent estimation using deep deterministic policy gradient," in *2021 SoutheastCon*, pp. 1–8, 2021.

[53] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *31st International Conference on Machine Learning, ICML 2014*, 2014.

[54] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, 2014.