ANGLE BASED LOCALIZATION OF AN AUTONOMOUS LAWNMOWER VIA RADIO FREQUENCY BEACONS AND A DIRECTIONAL ANTENNA

by

DANIEL ALVIN BENNETT

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

Thesis Adviser: Dr. Roger Quinn

Department of Mechanical and Aerospace Engineering

CASE WESTERN RESERVE UNIVERSITY

August, 2010

CASE WESTERN RESERVE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

We hereby approve the thesis/dissertation of

Daniel A. Bennett

candidate for the Master of Science degree *.

(signed) R. D. Quinn
(chair of the committee)

Francis Merat

J. M. Prahl

(date) 5/28/2010

*We also certify that written approval has been obtained for any proprietary material contained therein.

Table of Contents

1 Introduction 7

2 Background 11

2.1 Existing Robotic Lawn Mowers 12

2.2 Localization Techniques 14

2.2.1 GPS/Radio Frequency Time of Flight 14

2.2.2 Ultra-sonic Sensors: 16

2.2.3 Using RSSI Readings for Distance Based Localization 17

2.2.4 Using RSSI Readings for Angle Based Localization 18

3 Methods 21

3.1 Triangulation Techniques 21

3.2 Antenna Test Device 27

3.3 Obtaining RSSI Readings 32

3.3.1 Windows Management Instrumentation 32

3.3.2 Native Wifi API 34

3.3.3 Roving Networks WiFly Module 36

3.4 Angle-Finding Algorithm 39

4 Results 46

4.1 Data Acquisition and Filtering 46

4.2 Data Logs 49

4.2.1 Inside Logs 50

4.2.2 Outside Logs 51

4.2.3 Truncated Trials 56

4.2.4 Evaluation of Different Fit Methods 58

5 Analysis and Discussion 60

5.1 Indoor Accuracy 60

5.2 Effect of Chain Link Fences 61

5.3 Other Obstacles 62

5.4 Expected Triangulation Accuracy 64

6 Conclusion 67

6.1 Future Work 68

7 Appendix 70

7.1 Geometric Triangulation Algorithm 70
7.2 Polynomial Fit Method Comparison (all errors in degrees) 71
7.3 Estimated Cost 72
8 References 73

List of Figures

Figure 1-1 Competition Field for 2010 (30) 8
Figure 2-1: Examples of commercial lawnmowers 12
Figure 2-2: LawnBott cutting pattern (2) 13
Figure 2-3: Error From RSSI Ranging Localization 17
Figure 2-4: Localization Problem Posed in (16) 19
Figure 3-1 Typical Setup of the Localization Problem (19) 22
Figure 3-2: Illustration of the ambiguity that occurs when beacons and robot are collinear 24
Figure 3-3: Locations where triangulation solution is undefined 24
Figure 3-4: Radiation pattern for a parabolic dish antenna (28) 28
Figure 3-5: Radiation pattern for a waveguide antenna (25) 28
Figure 3-6: Schematic of directional waveguide antenna (29) 29
Figure 3-7: An inside view of the prototype waveguide antenna 30
Figure 3-8: Completed antenna assembly 32
Figure 3-9: Roving Networks WiFly Module (23) 36
Figure 3-10: Schematic of WiFly Circuit 37
Figure 3-11: Typical Data Set and Polynomial Fit 43
Figure 4-1: Antenna Testing Board 46
Figure 4-2: Log 1 Data 50
Figure 4-3: Log 2 Data 50
Figure 4-4: Outside Log 1 Data 51
Figure 4-5: Outside Log 2 Data 51
Figure 4-6: Outside Log 3 Data 52
Figure 4-7: Outside Log 4 Data 52
Figure 4-8: Outside Log 5 Data 53
Figure 4-9: Outside Log 6 Data 53
Figure 4-10: Outside Log 7 Data 54
Figure 4-11: Outside Log 8 Data 54
Figure 4-12: RMS Error and Offset for Inside (IL) and Outside (OL) Logs 55
Figure 4-13: Truncated Trial Locations 56
Figure 4-14: Error and Offsets for Truncated Logs (TL) 57
Figure 4-15: Fit Comparison (RMS Error) 58
Figure 4-16: Fit Comparison (Offset) 59
Figure 5-1: Representative Data Plot for IL 2 60
Figure 5-2: Representative Data Plot for OL 4 61
Figure 5-3: Representative Data Plot for OL 8 62
Figure 5-4: Representative Data Plot for TL 5 63
Figure 5-5: Error Plots for 6 Beacons, ± 4 Degrees 64
Figure 5-6: Error Plots for 8 Beacons, ±4 Degrees 65
Figure 5-7: Error Plots for 8 Beacons, ±2.5 degrees 66

Figure 7-1: Generalized Geometric Triangulation Algorithm (19) 70
Figure 7-2: Polynomial Fit Comparison Data 71
Figure 7-3: Cost Estimate Table 72

Acknowledgments

I would like to thank the following people for their help on this thesis:

My adviser, Dr. Roger Quinn, for getting me started in the Biorobotics lab and this project

My committee, Dr. Frank Merat and Dr. Joseph Prahl

Arkady Polinkovsky, for his help in finding components for my antenna

Ian Charnas and Ed Burwell, for their assistance in configuring the WiFly module

Jim Green and MTD for providing funding for the CWRU Cutter project

The rest of the Biorobotics lab for creating such a relaxed and fun work environment

Angle-Based Localization of an Autonomous Lawnmower via

Radio Frequency Beacons and a Directional Antenna

Abstract

by

DANIEL ALVIN BENNETT

Intelligent robots have the potential to become a major part of everyday life by performing menial household tasks, but are hindered by the high cost of sensors. Several companies have already started somewhat successful lines of autonomous robots, but as yet the functionality is too limited and the price is too high for widespread use. As a less expensive alternative to GPS for lawnmower navigation, this paper proposes the use of a rotating directional antenna to plot signal strengths verses angular displacement for several beacons of known locations to find their beacons relative to the mower. These angles are then used to triangulate the mower's position. Preliminary testing of this method indicates possible antenna error of less than 4 degrees, which corresponds to a simulated position error of less than 1.5 meters. Such a system has the potential to be superior to inexpensive GPS units in both accuracy and cost.

1 Introduction

There are many household chores related to maintenance and upkeep of the home and property that many people would rather avoid. Tasks such as cleaning, vacuuming, painting, and lawn-mowing are tedious and repetitive, but must be completed regularly to keep a house livable. Though there will always be some people who enjoy these activities, most would relish the prospect of technology eliminating the need for human involvement in household tasks. To this end, in recent years, advances in sensor and processing technology have enabled robots to start expanding to the field of consumer electronics. Beginning with products like automatic vacuum cleaners and autonomous lawnmowers, companies have shown that there is a market for low cost, robust robots that perform menial tasks (1).

Despite this advancement and the limited proliferation of these mobile robots, the accuracy of sensors and processing power is still constrained enough to significantly hinder the effectiveness and robustness of these products. As a result of the vast performance gap between robots and human operators, for now these devices are still relegated to the role of a novelty. In order for the average person to purchase a robotic lawnmower or vacuum cleaner, they will expect the robot to perform its task with a reasonable level of effectiveness and expediency, meaning that the task completed would not take outrageously longer or be of vastly inferior quality. In addition, it is important that the price be accessible to the most number of people possible. As it now stands, not only is there is a significant trade-off between effectiveness and price, but even the most expensive robots are having difficulties serving as adequate replacements for human operators.

For the last several years, a team at Case Western Reserve University has been developing an autonomous lawnmower in conjunction with our corporate sponsors, MTD. It is the hope of

our team to develop a robotic lawn-mowing platform (dubbed the "CWRU Cutter") that can overcome both the problems with functionality and price that have plagued previous attempts to produce a marketable autonomous lawnmower. The chief test of this functionality has been the annual ION Autonomous Lawnmower Competition. This competition has been challenging teams of researchers over the last seven years to develop a lawn-mowing robot that will cut a lawn reliably and intelligently. Over the years the competition has been held, the rules have slowly changed to be more demanding of the lawn mowers' performance. A sample course for the current competition is shown in Figure 1-1. As can be seen from the diagram, there are

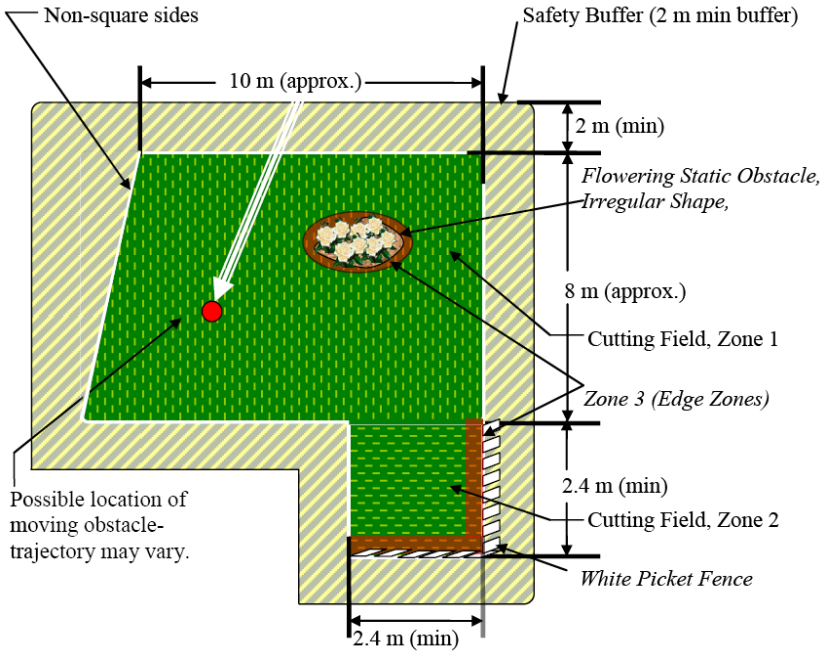


Figure 1-1 Competition Field for 2010 (30)

several obstacles inside the boundaries of the course. Penalties are incurred if any obstacle is struck, so an emphasis is placed not only on the ability of the mower to mow the entirety of the lawn, but also on its ability to recognize both stationary and mobile obstacles and effectively plan around them. Additionally, areas that require more precision, such as zones 2 and 3 in the diagram, have a considerably higher impact on the final score than the larger zone 1. These

scoring requirements are specified so the competition will test the mower on its ability to handle environments that are likely to be encountered in a typical yard.

In addition to the completeness of cut and the avoidance of obstacles, a commercial mower would be expected to have a high quality of cut as well. To properly emulate a human operator, the mower should not mow haphazardly, but rather it should have a set plan and mow in orderly paths. To accomplish this, CWRU Cutter handles navigation in two separate phases. First, a global map of the mowing area is generated with GPS coordinates. With this information, the desired cutting pattern is laid out as a series of GPS waypoints and path types. All of this is done offline, before the mower is set loose on the field. Obstacles are ignored in this stage, as paths are plotted directly through them. The second phase of navigation relates to the avoidance of obstacles. Since the mower is told to go straight through the obstacles, once an obstacle is detected by sensors, the mower follows a reflex to veer around the obstacle and back onto the path. These two navigation schemes, the local and the global, work independently: the local obstacle avoidance will only override the global path plan when an obstacle is detected. After the obstacle is passed, the global path takes over again and guides the mower via GPS.

With this navigation scheme, it is imperative that CWRU Cutter be able to reliably determine its position relative to the rest of the yard or course. It is this position that determines start and end conditions of the current mowing path, as well as keeps the mower from straying too far away from the predefined path. In past years the concentration has been to make a lawnmower that performs all tasks as reliably as possible, regardless of the cost. To this end, localization was achieved to a high degree of accuracy with an expensive GPS setup. For this and future iterations of CWRU Cutter, however, we will attempt to achieve the same results by

integrating a combination of much less expensive sensors. The purpose of this thesis is to investigate cost effective alternatives to GPS. Specifically, it will investigate the viability of using a rotating directional antenna in combination with several wireless beacons of known location to provide an accurate estimation of the position of the mower. This method, if successful, will provide a very inexpensive method of localization which can be combined with other methods to further increase the accuracy of the calculated position of the mower.

2 Background

Though the reality of robust and inexpensive intelligent mobile robots is far from being realized, much work has been done in pursuit of this goal over the last several decades. Sensor and processing technology has made great advances, and enabled new methods for navigation and localization. Utilizing these advances, several companies have even started product lines of autonomous lawnmowers. Though price restrictions still limit the availability and intelligence of these mowers, these companies have shown that there is a market for autonomous machines. These products are limited in their intelligence and functionality by the costs and limits of technology, but there is also much research being done in the area of inexpensive localization. There are pros and cons to the different methods available at this time, and this chapter will discuss these benefits and drawbacks, as well as the decisions that led to the localization technique described in this thesis.

2.1 Existing Robotic Lawn Mowers

Figure 2-1: Examples of commercial lawnmowers
Top Left: LawnBott LB3510 by Kyodo America (2)
Top Right: Automower Solar Hybrid by Husqvarna (3)
Bottom: Robomow RL2000 by Friendly Robotics (4)

Using robots to mow lawns is not a new idea, and in fact, several companies have lines of autonomous lawnmowers targeted at the average homeowner. While the cost is significantly higher than a normal lawnmower, much care has gone in to making the mowers simple to use and effective at their intended task. Some of the most popular robotic lawnmowers are made by companies such as Friendly Robotics, Husqvarna, and Kyodo America (1). Though each brand has features that distinguish it from the others, they all function in a similar basic manner (2), (3), (4). The mower starts from a base charging station, and travels in a straight line or some other simple predefined pattern until it strikes an obstacle or boundary. Once it does so, it will turn to a random heading, and travel in a straight line until it encounters another obstacle. If allowed to continue this series of random turns long enough, it is expected to mow the entirety

of the lawn. Once the mower has run out of battery power or is recalled, it will return to the base station to recharge. An illustration of this cutting pattern is shown in Figure 2-2.

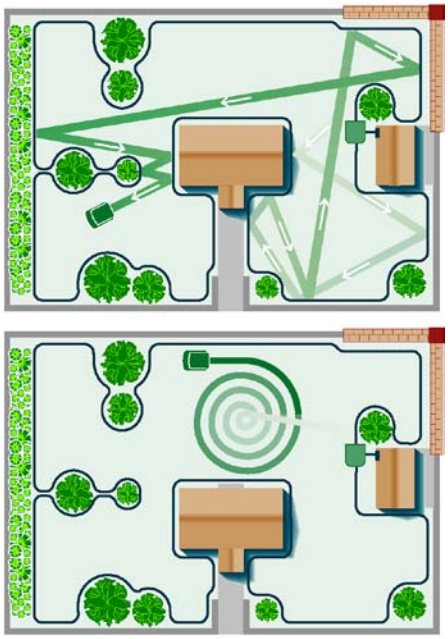


Figure 2-2: LawnBott cutting pattern (2)

The sensory information available to these commercial mowers is very limited. Most of the navigation and obstacle avoidance is accomplished with contact sensors placed around the mower. These sensors detect when the mower impacts a rigid obstacle that is taller than the cutting deck of the mower. For all other obstacles and for demarcating the boundaries of the cutting area, an electric wire is laid around the perimeter of all areas from which the mower is restricted. This “electric fence” works in a similar manner to invisible dog fences: when the mower encounters the wire, it is detected with onboard sensors and is interpreted as an obstacle. This limited suite of sensors allows these mowers to be cheaper and very robust with respect to their intended functionality, but they lack the intelligence to remember any of the obstacles that are encountered, or to efficiently plan a route through the yard. Without this capability to plan ahead, autonomous lawnmowers will have a difficult time becoming popular with anybody who cares greatly about the cut of their lawn.

2.2 Localization Techniques

In order to plan ahead and mow a lawn as intelligently as a person, a lawnmower would require sensors that not only detect obstacles before contact, but also provide a way to achieve an accurate estimation of its absolute position in the yard. Some mobile robots are designed to generate maps of the terrain as they move, with no a priori information of the yard (5), (6), (7). Rather than have sensors that directly localize the robot in a global frame, these robots use local sensors to detect environmental features, and use the features to ascertain its location. This method was deemed overly complex for this application, since there is no reason not to use any prior information about the environment the lawn mower will operate in. Since it will most likely be operating in a fixed area that will not change significantly day to day, giving the mower the ability to map unknown areas is unnecessary.

A more common localization strategy for mobile robots is to have a number of beacons of known position in the vicinity of the robot broadcast some sort of signal to a receiver on the robot. Then, usually with either distance or direction information from the beacons, the robot can triangulate its position with respect to some absolute frame. There are various techniques already in existence that accomplish this, but many are too expensive, too inaccurate, or too unreliable when implemented. The following is a brief survey of these localization techniques.

2.2.1 GPS/Radio Frequency Time of Flight

The global positioning system is probably the best example of an RF time of flight distance-based triangulation technique. Satellites in orbit serve as beacons which broadcast signals to each other and to receivers on the ground. By comparing times the signal was sent to when it is received, a distance to each satellite can be determined, which can then be used to triangulate the receiver's position. GPS has the advantage of already being a well-established solution to

the localization problem. As such, there are many companies that sell GPS modules that require little configuring to be used for a mobile robot. CWRU Cutter uses a differential GPS system with one stationary receiver and one receiver on the robot. This setup provides position accuracy on the order of a few centimeters while moving, or a few millimeters while stationary. However, this accuracy comes at a price: a differential GPS setup with this level of accuracy costs approximately $40,000 (8). Needless to say, this system will not be used for commercial mowers in the near future. In addition to price constraints, GPS receivers have several other drawbacks to other methods. In order for the system to function, the receiver must be within the line of sight of at least four satellites overhead, which makes GPS almost useless in locations surrounded by tall buildings or large trees (9).

To overcome the price constraint there are less expensive GPS modules available for use, but positional accuracy is sacrificed. U-Blox makes modules that have an accuracy of up to 2 meters (10), but they still require relatively clear skies to operate. An inexpensive GPS module with this accuracy would necessarily have to be used in conjunction with other sensors to give a better estimate of position.

An alternative to GPS that would overcome the satellite reception issue would be to implement a beacon system on the ground. Placing beacons much closer to the robot in known locations around a yard would effectively eliminate the problem of lost signals. Unfortunately, such a beacon system would present its own problems. Over a distance of about ten to twenty meters, which is the distance the mower would be expected to operate from the beacon, the time of flight of any radio signal would be approximately 3×10^{-8} seconds. Comparatively, the time of flight of a signal from a GPS satellite in medium earth orbit (approximately 20,200 kilometers), would be about .07 seconds. Such a small time difference for the close beacons

would require very precise equipment to measure reliably, and all of the beacons would have to be very accurately synchronized. This level of precision would necessitate a much higher cost, making such a method infeasible for now.

2.2.2 Ultra-sonic Sensors:

MIT has recently developed an ultra-sonic sensor based localization system dubbed “Cricket.” Like GPS, this system uses a time of flight based distancing method to discern a moving platform’s location. In the application of this system described in (11), concern was not for exact triangulation of a robot, but rather generally localizing a robot within rooms or sections of a room. However, it is certainly possible the system could be adapted with more beacons to accomplish more specific localization.

Instead of timing RF pulses, the Cricket system uses ultrasonic signals that are sent from beacons placed around a room. This requires significantly less complicated circuitry, as sound waves travel much slower than radio waves, and it has the potential to be a very accurate distancing method. Unfortunately, there are several problems with using ultrasonic sensors reliably that the MIT team discovered. First of all, the speed of sound in air depends greatly on temperature and atmospheric effects. This can be accounted for in some capacity by temperature and humidity sensors on the robot, but there is no feasible way to precisely account for the temperature gradients between the beacons and the robot. The Cricket system was developed primarily as a system for indoor localization, so it was deemed that it would not be likely for there to be extreme spatial temperature gradients, and that the error incurred by such gradients would be tolerable. Outside, however, temperature and environmental factors can vary greatly from time to time and position to position, which introduce potentially large errors that are difficult to predict well.

2.2.3 Using RSSI Readings for Distance Based Localization

Received Signal Strength Indicator (RSSI) readings give the signal strength of wireless connections using the 802.11 standard. Usually, this reading is expressed in decibels, which describe the signal to noise ratio of the signal. The most simplistic way to use this reading is to simply take the value itself as a measurement of distance. Since a wave propagating outwards in a sphere will dissipate in proportion to the square of the distance from the source, this can be used to translate the RSSI values from each beacon into an estimation of distance. Even without a set reference point, the difference in the distances can be used to estimate change in position. With many beacons over a large area, this method has been used to achieve moderate accuracy in several implementations. The mobile robot in (12) was able to localize with a mean error between .5 meters and 1 meter while in a small indoors environment. The robot in (13), on the other hand, was able to achieve an accuracy of about 4 meters, as shown in Figure 2-3 , for a large outdoor field with 25 beacons surrounded by tall trees.

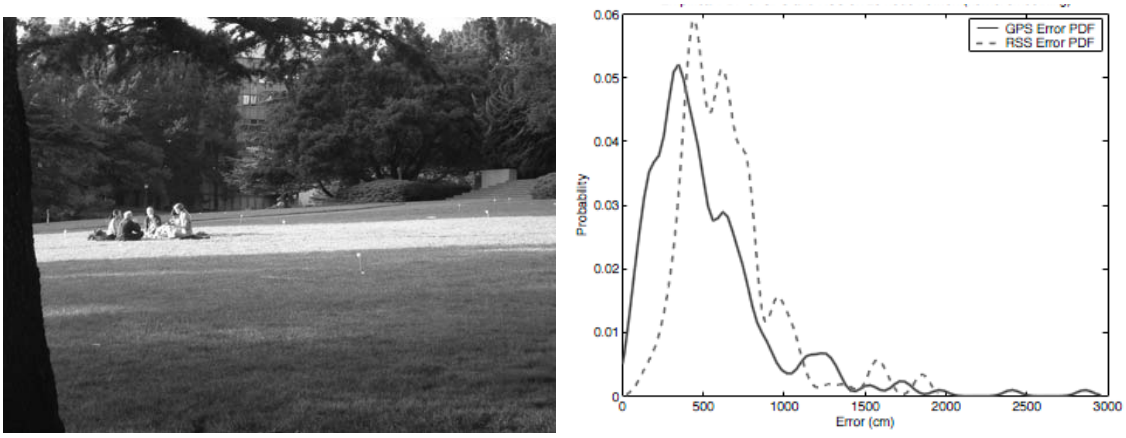


Figure 2-3: Error From RSSI Ranging Localization
Left: Test Field
Right: Probability Density Plot of RSSI Error vs GPS Error (13)

The problem with using only the magnitudes of the RSSI reading to calculate distance from a beacon is that much like ultra sonic sensors, this reading can be very sensitive to the

environment, and it can also be very noisy. Slight changes in the environment can drastically alter how the signal strengths relate to the distance to the beacons. Thus, any such system would most likely have to be calibrated before each use, which would be impractical in any commercial application.

Another localization scheme utilizing signal strength readings would be the use of a system of radio frequency identification (RFID) tags. These tags placed in the field or yard would allow a reader placed on the mower to ascertain its position based on which tags were nearby. Such a system was implemented for localization in (14). Unfortunately, RFID usually works over relatively short distances, so for these methods to be viable in an autonomous lawnmower application there would have to be a large number of tags placed in many locations around the yard. In this way, they would serve only slightly better than the invisible fences present on current model lawnmowers.

2.2.4 Using RSSI Readings for Angle Based Localization

The largest problem with using RSSI readings for localization is the signal noise: signal strength is heavily dependent on atmospheric conditions, obstacles, and other subtle factors that can change from day to day. These problems can be remedied somewhat by using angle-based triangulation methods. If the RSSI values are plotted verses angular position, their absolute magnitudes are irrelevant, as is their relative magnitudes over time. All that matters are the instantaneous relative magnitudes of the signal received from each direction. This is the basic idea behind LORAN C, a localization system for ships at sea. The LORAN system was able to provide a position accuracy of less than .25 nautical miles before it was shut down in 2010 (15).

In (16), this idea was implemented to calculate positions of nodes in wireless sensor networks. In this implementation, rather than just localizing one mobile platform, the task was to

determine the location of many randomly distributed sensor nodes. In this system there also exist several beacon nodes, whose locations are known. Since there are so many nodes of unknown locations, rather than mount the directional antennas on the sensor nodes, they were added to each beacon node. The antenna attached to each beacon node would rotate at a constant rate plotting the RSSI values from each sensor node verses direction. During execution, a central control unit could aggregate this information, and triangulate the location of each beacon.

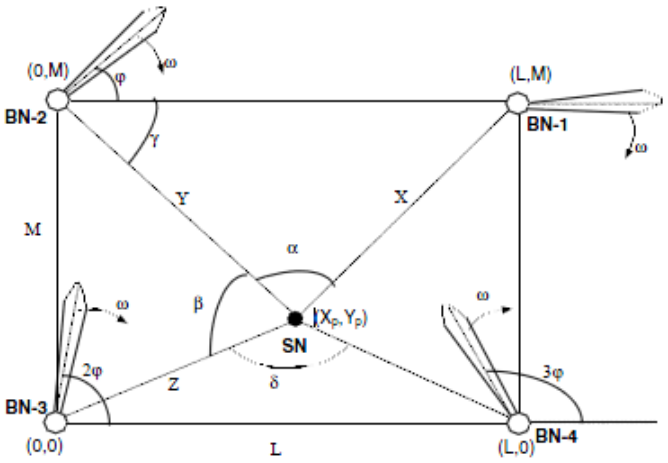


Figure 2-4: Localization Problem Posed in (16)

The simulated results presented in (16) show that this method is promising for localizing sensor networks, but there are several changes that are necessary for this method to be applicable for a mobile robot. One issue is the fact that the wireless sensor nodes in (16) are all stationary, so the accuracy of the position estimate should approach the best possible accuracy as time goes on. For adapting the same method to a mobile platform, there will be added complexity in the direction finding algorithm to account for motion and orientation. Additionally, the algorithm must be able to decide how to use a much smaller set of data to make an accurate estimation of the robot's position.

Additionally, it would not make sense for every beacon to have rotating directional antennas in the case of a mobile robot. Since there is only one robot but many beacons, it is more feasible that the directional antenna would be placed on the robot, and not on every beacon. This has the advantage of greatly reducing the mechanical complexity of the system. This is the method that will be investigated in this thesis. Using a rotating directional antenna with stationary beacons has previously been suggested as a localization system in a patent for an autonomous lawnmower (17), (18). The patent covers a triangulation algorithm that finds the perceived angles between beacons by searching for peaks in the signal strength, and then triangulates the position of the mower by finding the intersection of two circles that each contains the mower (discussed in Section 3.1, and in depth in (19)). In this case, a simple loop antenna was proposed for finding the bearings of the beacons.

This method differs from the one investigated in this thesis in several ways. First, the technique proposed to find the bearing of the beacons seems to only detect peaks, meaning that the method would be very susceptible to multipath effects from reflected signals. Without using any additional data when filtering the signal, it is impossible to tell which received bearing is accurate. Additionally, the algorithm proposed to triangulate position based on relative angles between the mower and the beacons is not ideal when additional beacons are available for triangulation. It was discovered through simulating different triangulation algorithms that the method described in (20) is significantly more accurate in the presence of more than three beacons. Most importantly, however, the method suggested in (17) and (18) was never implemented or tested on an autonomous lawnmower, so it is impossible to assess the validity or robustness of the method.

3 Methods

This thesis will be investigating an angle based localization method similar to the ones just described. A directional antenna rotating at a fixed speed will be mounted to the lawnmower, and wireless beacons of known position will be distributed about the yard. As the antenna rotates, the signal strength received from the beacons will be plotted verses angular displacement. From this measurement, the bearing of each beacon can be found, and a triangulation scheme will be used to calculate the position of the robot. This section will describe the methodology of this calculation, beginning with an overview of angle based triangulation algorithms using three or more beacons. Next, the construction of the directional antenna will be covered, followed by a description of the method used to get accurate RSSI values. Finally, the actual data acquisition and angle finding algorithm for a single beacon will be explained.

3.1 Triangulation Techniques

Angle based triangulation methods are not new in the area of localization, and there are several methods that have been in use for a long time. There is a distinct mathematical disadvantage in relation to distance based methods however: given an error in angle Δθ, the perceived location of the beacon would have an error in position equal to:

$$\Delta s = d * \sin(\Delta\theta) \tag{1}$$

Where d is the distance from the robot to the beacon. As a result, even if sensor measurement error remained constant between the two methods, the localization error will increase as the robot moves farther from a beacon. Given this fundamental tendency towards inaccuracy, it was decided to investigate the potential accuracy of triangulation techniques given various

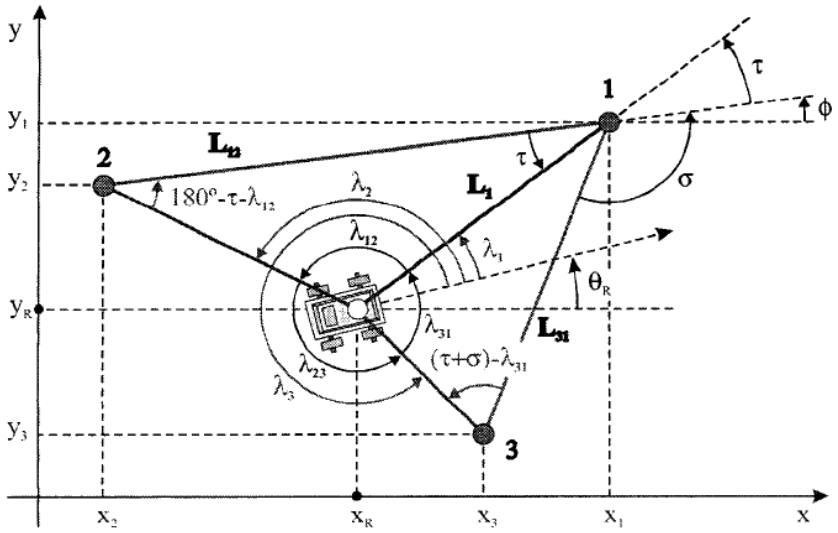


Figure 3-1 Typical Setup of the Localization Problem (19)

sensor errors. Once angle based triangulation was deemed viable, the algorithms developed could also be used to project expected localization error once the performance of angle finding techniques were characterized.

Since methods of triangulation are rather well-established, this project will be modifying and adapting algorithms found in papers rather than creating a new method. There are two main methods of triangulation that will be explored here. The first is a more geometric approach, whereas the second involves using the angle to calculate the distance from the robot to each beacon. For either method, in order to fully define the position and heading of the robot there must be at least three beacons visible. A diagram of the triangulation problem is shown in Figure 3-1. Both algorithms take as arguments the absolute coordinates of each beacon, and the bearing of each beacon relative to the robot. These bearings are expressed in the figure as λ_1, λ_2, and λ_3.

The first algorithm is labeled as a “geometric” algorithm because it requires determining the intersection of two circles. Since a circle can be defined by any three points, there exists a unique circle that is defined by the robot and two of the beacons. Likewise, there is another

unique circle that is defined by the robot and any other combination of two beacons. The location of the robot can then be determined by the intersection of the two circles. This intersection can be easily found using trigonometric arguments for any specific case, but the difficulty of the algorithm lies in correctly defining input values such that it returns valid results for every possible scenario. The method of calculation is dependent not only on which beacons are chosen to fit each circle, but also where the robot is with respect to them. It is difficult to account for every possibility programmatically. The exact algorithm used to account for this was taken from (19), and is reproduced in the appendix (Figure 7-1).

While robust with respect to robot orientation and position relative to the beacons, there are several locations that are impossible to account for with this algorithm. Two arcs will typically have two intersections: one will be the robot, and the other will be the beacon shared by both circles. If the robot is coincident with any beacon the two circles will only have one intersection at the location of both the beacon and the robot, but if the robot lies on the circle defined by all three beacons, there should be an infinite number of intersections. In practice, this results in the position error given by the algorithm increasing to the point of uselessness as the robot is closer to the arc.

The algorithm likewise encounters problems when the robot is collinear with any two of the beacons. When this occurs, the bearing of the two collinear beacons will be the same, and there will be two possible locations for the robot. From the robot's perspective, the third beacon could be in either of two positions, and without any other information it is impossible to determine which is valid. This situation is illustrated in Figure 3-2. The arrows represent the direction of travel of the robot, the beacons are labeled A, B, and C, and α is the bearing of beacon C relative to the robot. The robot knows it must be on the line AB, but cannot

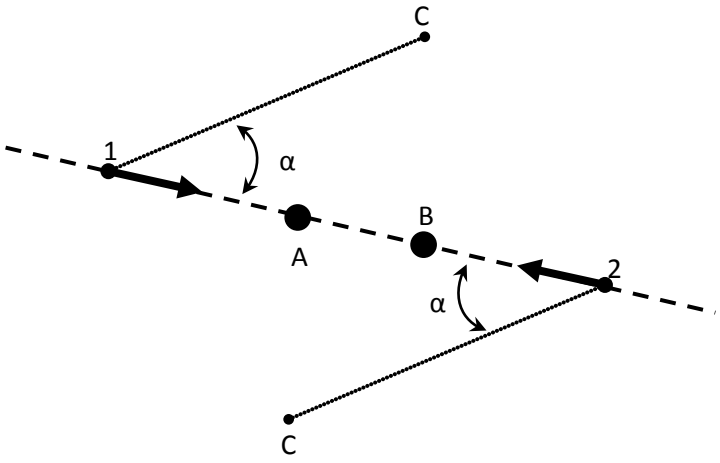


Figure 3-2: Illustration of the ambiguity that occurs when beacons and robot are collinear

determine whether it is in position 1 or 2, since both are valid. Being collinear with two beacons has a similar effect as being on a circle with all three: as the robot approaches the line, the error increases dramatically, making accurate localization impossible. These areas of inaccuracies are illustrated in Figure 3-3.

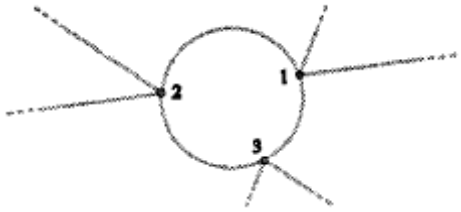


Figure 3-3: Locations where triangulation solution is undefined

These limitations are present not only with this method, but also with the second method that was explored, indicating that three-beacon localization would be sub-optimal for many cases. Fortunately, these errors are mostly overcome when more beacons are introduced. In addition to increasing the overall accuracy of localization additional beacons add redundancy to fall back on when the robot is in any of the troublesome areas. The algorithm outlined in (19) was strictly for three beacons, so it was modified to use the extra information provided by more beacons. In the modified triangulation program, the original algorithm was iterated over every

possible combination of three beacons. Therefore, if there are N beacons, there will be P different bearing calculations, where P is:

$$P = \frac{N!}{(N-3)!*3!} \tag{2}$$

Unless the beacons are deliberately placed in such a way as to be in a circle or collinear, it is very likely that there will be several beacon combinations that are not near an error prone area. As a result of this, the algorithm can safely exclude any bearings obtained from beacon combinations that are likely to have large error without jeopardizing its ability to triangulate.

Once the inaccurate beacon combinations are excluded, the remaining positions are combined to give an aggregate location. In preliminary tests of triangulation algorithms, it was observed that the solution is more accurate when the robot's location is closer to the centroid of beacon set. Thus, to increase accuracy the calculated positions are combined with a weighted average, where each coordinate has a weight proportional to its distance from the centroid of its corresponding beacon set.

For the alternative triangulation method presented in (20), less emphasis is placed on having a geometric description of the procedure. While the first method had to be adapted to handle more than three beacons, this method uses all the beacons at once to solve for position and heading. The basic idea behind the algorithm is to use a complex representation of vectors to create a system of equations that can be solved for the position of the robot. Using this complex representation, a vector from the robot to a beacon z_i can be expressed as:

$$z_i = l_i * e^{j*\tau_i} \quad for\, i = 1,\ldots,n \tag{3}$$

Where l_i is the magnitude of the vector, and τ_i is the angle relative to the robot that is given by the sensor. If each of these equations is divided by the complex representation of a reference beacon z_0, some additional algebra will produce an over-determined set of n(n-1) equations where the only unknowns are the ratios of the lengths of the z vectors. Since the equations are over defined, a least squares approach was used to find the robot location that will minimize the error from all of the equations. The exact algorithm and its derivation are not reproduced here, but can be found in (20).

The output given by this algorithm takes the form of a vector starting at the robot and going to the reference beacon, from which the position of the robot can easily be found. The problem with this approach is that its accuracy depends strongly on the accuracy of the reference beacon. In (20) Betke and Gurvits suggest using the most accurate bearing measurement, if available, to choose which beacon to use as reference. With the hardware intended for this project, it is not likely there will be one beacon with a significantly higher accuracy, so this is not a promising strategy. However, while at the time this algorithm was developed there were significant constraints on hardware power, today there is little worry that in realistic situations available processing power will be outstripped by this algorithm. This is especially true in this case, as this algorithm will not be required to run in real-time. Thus, in order to achieve a greater accuracy, the algorithm is iterated several times, much like the previous method. Each iteration chooses a different beacon to act as a reference, and once all iterations are complete, any outliers are removed before returning the average of the calculated positions.

Once both of these methods were implemented, a program was created to iterate through various bearing errors in order to assess the potential accuracy of the method. Initial results were promising, but as expected there was a strong dependence on the accuracy of the

directional antenna used to measure the bearings. Since there are no products currently available which use directional antennas in this way, it was difficult to get a good idea of the accuracy that could be expected from this application. Therefore, to test the viability of this angle based triangulation method, it was necessary to construct a specialized test rig.

3.2 Antenna Test Device

Preliminary simulations of the triangulation algorithm indicated that localization could be achieved provided the angle data can be obtained reliably and accurately. Several possibilities for determining the bearing of the reference beacons were considered. One possibility was to use a light based system, where infrared beacons placed at known locations emit a unique frequency. A rotating detector on the mower would then be able to match each beacon with its angular displacement. This method can potentially be very accurate, but the main problem is that it requires strict line-of-sight for the beacons to be seen by the detector. This limits the viable locations for beacons within a yard, and could pose problems for yards with many trees or obstacles. Also, the operating frequency range would be severely limited by solar radiation, meaning interference and false signals could become a large problem. Another possibility was to use ultrasonic sensors, but this would encounter the same problems already discussed: namely, that ultrasonic behavior is highly dependent on atmospheric conditions.

A third option would be to use a directional antenna mounted on a servomotor, and beacons placed at known locations that broadcast a wireless signal. RSSI attenuation based distance measurements were already found to be highly unreliable, but rather than attempt to get distance data directly from the RSSI readings, this method would use a plot of the readings versus the angular displacement of the motor. Assuming the antenna has a high degree of directionality, it is expected that the plot would have a distinct peak of RSSI values in the

direction of the beacon. If such an antenna could be constructed which is also small enough to be easily rotated, this method has the potential to be robust with respect to non-metallic obstacles. Since high-frequency radio waves pass through most materials, it should not be necessary to have a direct line of sight between the antenna and the beacon. This method is also advantageous in that it can be implemented for a very low cost. Wireless beacons can be mass produced very cheaply, especially since their functionality would only need to be very limited, so there would be no problem with increasing the accuracy of the system by having many beacons in the yard.

There are several different models of directional antennas, each with different sizes and typical beam width. All directional antennas have a radiation pattern similar to the one shown in Figure 3-4 for a parabolic dish antenna. This figure plots the gain of the antenna against the angular position, where 0 degrees is where the antenna is aimed. Most antennas have a primary lobe in front, and secondary lobes of lower gain to the side, and possibly behind. The

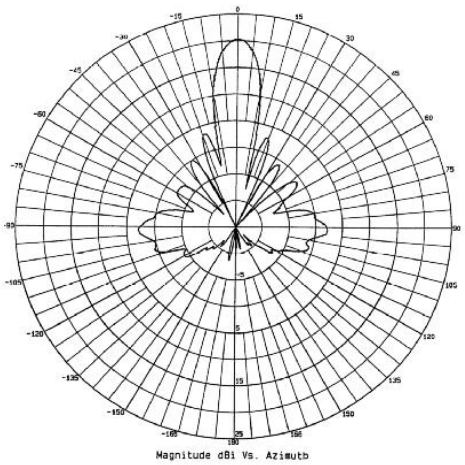


Figure 3-4: Radiation pattern for a parabolic dish antenna (28)

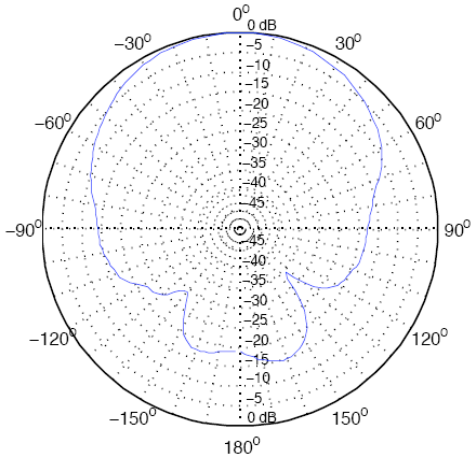


Figure 3-5: Radiation pattern for a waveguide antenna (25)

highest degree of directionality is given by parabolic dish antennas, followed by yagi antennas, flat panel antennas, and finally waveguide antennas. A parabolic dish antenna seems like the

best choice based on beam width alone, but their large sizes make them too unwieldy to be mounted and rotated on a mid-sized mobile robot. The other possibilities can come in much smaller sizes, and can be easily attached to a motor to rotate. For the experimental set-up used to test this method a waveguide antenna was chosen, due to the simplicity of its design and its lightweight construction. In future applications an enclosed yagi antenna would be a better choice, and can be nearly as compact as a waveguide, if slightly heavier. The loss of directionality is especially apparent from the plots of the radiation pattern shown in Figure 3-5. Though not ideal long term, the waveguide antenna was deemed acceptable for the experimental set-up to prove the concept, especially since there were certain advantages to constructing the antenna instead of purchasing it.

A waveguide antenna consists of a radiating element that sends and receives signals, which are guided in a specific direction by a narrow metallic cylinder. The performance of the antenna is strongly dependent on the geometry of both the waveguide and the radiating element. Even

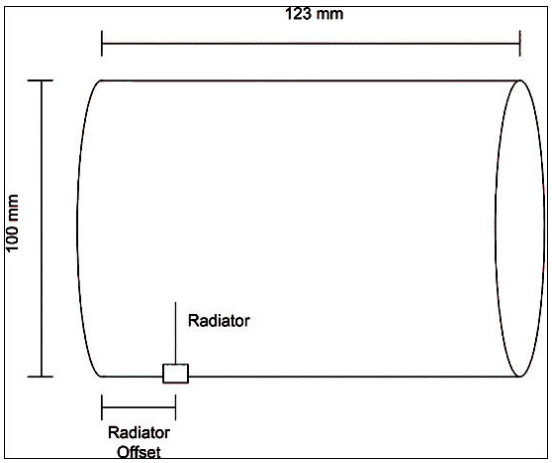


Figure 3-6: Schematic of directional waveguide antenna (29)

though there are very few components that make up a waveguide antenna, the theory that dictates the ideal size and spacing for these components is very complex and abstract, so the

antenna used here was constructed based on a design found on the internet. The design that was found was specific for operating in the 2.4 GHz frequency range, which is the frequency range used by common wireless routers, and is also the range being used for the prototype beacons. A schematic of the design is shown in Figure 3-6. The design called for a metal cylinder with a diameter of 90-110 mm, and a length of about 123 mm (half of a wavelength) or a full wavelength. Many of the designs investigated suggest using coffee cans or soup cans for the waveguide portion, and luckily, there was an empty coffee can conveniently at hand that fit the exact specifications in the design.

The radiating element proved more complicated to construct correctly. It had already been determined that the best way to attach the cable to the antenna would be through a bearing attached between the waveguide and the radiating element, so that as the can rotated, the radiator and cable would remain stationary as the waveguide revolves around them. This prevents wires from getting wound up and allows the antenna to rotate constantly in a single direction. Since the radiating element will be stationary with respect to the ground, but not the

Figure 3-7: An inside view of the prototype waveguide antenna

waveguide, it is essential that the element be axially symmetric, so that it will always present the same surface area to the opening of the waveguide. According to the design, there are three options for the shape of the element: a cylinder, a wedge, or a cone. The cone and wedge provide the best frequency coverage and gain, but the wedge does not satisfy the symmetry constraint, and the cone is difficult to make. A cone element was attempted at first, using a thin sheet of copper wrapped in the cone shape, but there were performance issues related to the discontinuity caused by the seam. The seam from the two sides of the cone coming together caused the element to have a different profile at different angular displacements, making it impossible to get consistent data. Thus, for later trials the cone element was replaced by a round wire, which yielded significantly better results. A close-up photo of the inside of the antenna and the radiating element is shown in Figure 3-7. As per the specifications, the element was 30.7 mm long, and it was placed 27 mm from the back of the can.

Once the antenna itself was constructed, it was attached to a Maxon servomotor with a built-in encoder, and the entire assembly was mounted on several pieces of aluminum Bosch stock to serve as a base. Also mounted to the Bosch stock was a micro switch which is actuated when the antenna completes one revolution. This serves to initialize the encoder readings, ensuring that 0 degrees is the same direction every time data is collected. This was later replaced by a non-contact IR sensor for greater precision, and also so that the antenna would no longer lurch in response to the resistance encountered as it actuates the sensor. The encoder ticks and the input from the IR sensor were read by an NI USB-6008 DAQ board. The board has binary inputs for the IR sensor as well as a built-in counter input which simplifies the interpretation of the encoder signal. Since the antenna would be rotated in only one direction, it was only necessary to read one channel of the encoder to get an accurate picture of angular displacement. A picture of the completed antenna assembly is shown in Figure 3-8.

Figure 3-8: Completed antenna assembly

3.3 Obtaining RSSI Readings

Rapidly and reliably obtaining RSSI value updates proved to be the most challenging aspect of this project. Since this triangulation method relies on matching RSSI readings to the exact encoder count on which they are updated, it is imperative that the exact time the RSSI value is updated is known and controlled. Though it seems like a simple part of a wireless driver, nobody has used this method to analyze the RSSI values before, so there were no tools readily available that provided the needed information by default. Several methods of retrieving the data were explored before finding one that worked, and all of them encountered the same problem either with the maximum speed of the data acquisition, or in the uncertain timing of the retrieved data.

3.3.1 Windows Management Instrumentation

The first method investigated was to use the Windows Management Instrumentation (WMI) tools included with the Windows operating system to query driver information. WMI is a set of

tools related to managing a Windows computer, intended to facilitate sharing computer management information between separate management programs (21). As a result it has many built-in functions that allow programs to obtain information from drivers. The first attempt at retrieving RSSI values was to invoke WMI functions via LABVIEW's .NET tools to query the RSSI information from the 802.11 wireless drivers, which in turn received the information from a USB wireless adapter connected to the antenna assembly. Tutorials on using .NET functions in LABVIEW are readily available, and Microsoft has posted extensive documentation on WMI online, so at first glance this method seemed promising. LABVIEW was able to successfully communicate with WMI and reliably obtain RSSI information from the "MSNdis_80211_ReceivedSignalStrength" class. Unfortunately, several problems soon surfaced with this method. In the future application of this localization software, it will be necessary to simultaneously retrieve RSSI information from several different wireless access points. WMI seemed to be capable of this, as there was documentation of a "MSNdis_80211_BSSIList" class which indicated that this function would return an array of SSID's and their corresponding RSSI values. However, after much investigation and troubleshooting, it was found that there is a bug in this very WMI class which prevents it from functioning as described. Instead of providing all access points and their information, the method displayed only the first in the list correctly and returned gibberish for the rest. Since this has been an outstanding bug for a very long time it cannot be assumed that Microsoft will repair it in a timely fashion. Moreover, it would be impossible to work around the bug with other WMI functions, which casts serious doubt on the viability of this method long-term.

Despite this major shortcoming of WMI, it was already demonstrated that it could return the RSSI value of the current network, so it seemed possible for it to be used for testing the ability of the antenna to provide accurate bearings. Unfortunately, another shortcoming was

discovered shortly after beginning testing with the antenna. The RSSI data coming from the driver would provide the same value repeatedly before changing, creating a step-like shape. Also, when interpreted by a preliminary peak finding algorithm, the returned peak would always be significantly shifted in the direction of rotation of the antenna. Since the shift seemed to be random, there was no way to account for it when the bearing was calculated. This unknown bias made matching RSSI values to encoder ticks impossible, rendering this method useless for the purposes of finding the bearing of the access point.

It was later determined that this behavior was caused by the way the RSSI query function works in WMI. Rather than refreshing the RSSI when the method is called, it returns a stored value for the RSSI that is updated at set intervals specified by the wireless driver software. Furthermore, the drivers provide limited response to any commands issued with WMI functions; scan functions did not work through the LABVIEW interface, and it proved impossible to specify when a driver should update the RSSI variable. Since for most management applications the precise timing of the RSSI reading is irrelevant, it would seem that this particular application is beyond the scope of the WMI tools.

3.3.2 Native Wifi API

After WMI, the next method explored was to use the Native Wifi API that is included in the software development kit (SDK) for Windows XP, Vista, and 7. The Native Wifi API contains many functions to manage wireless network connections (22). In this way, its functionality somewhat overlaps that of WMI, but since the SDK also contains classes that can be used to create drivers for Windows, it is likely that the SDK would contain functions that could circumvent the problems encountered with WMI. Once the SDK was installed, several promising functions were found, but due to the complicated data types the methods returned

the library functions could not be called directly by LABVIEW. To circumvent this, a dynamic link library (.dll) was written in Visual C++ that contained several “helper” methods to interface between LABVIEW and the Native Wifi API. These functions called the Native Wifi methods, selected the desired information from the data and returned the information as simple data types that could be easily interpreted by LABVIEW such as integers, strings (char arrays), and double precision numbers. LABVIEW programs could then import this library and call these functions instead of the default Native Wifi API libraries. The helper functions were intended to replicate all of the behavior that was previously desired from WMI; there were functions to read the RSSI value of the current network, to display the SSID of every visible network and its corresponding RSSI value, and to scan for available networks to refresh the stored RSSI values.

Though initially promising, this method eventually fell to many of the same shortcomings as the WMI method. When RSSI values were requested by the Native Wifi API functions, only stored values were returned. Again, there was no way to determine when the returned values were refreshed. While the scan function seemed promising as a way around this problem, it proved to be unstable when called by LABVIEW. It took far too long to execute, which would have been a fatal problem in itself, but the long delay also caused LABVIEW to crash when the function was placed inside of a loop. Thus the Native Wifi API, much like WMI, proved unworkable. It is worth noting here, however, that there are tools present in the Windows SDK to create a wireless driver from scratch, where the functionality could be stripped down to include only what is required. It is possible such a driver could be fast and reliable enough to be used in the bearing calculations, but it would require a person with far more programming and driver experience to be implemented.

3.3.3 Roving Networks WiFly Module

After the attempts to use Windows drivers to get the needed information failed, it was decided to try a lower-level approach in the form of a Roving Networks WiFly GSX module (Figure 3-9). The WiFly module is a low-power 802.11 b/g transceiver typically used in conjunction with an

Figure 3-9: Roving Networks WiFly Module (23)

Arduino microprocessor board (23). The WiFly module can connect to wireless networks and display information about the connection, such as SSID, RSSI, and MAC addresses. Communication with the WiFly is achieved via a UART serial connection operating at 3.3V for input. In order to communicate with standard RS-232 serial ports, a MAX 3232 chip was installed between the WiFly and the serial connector. A schematic of the functioning circuit is shown in Figure 3-10.

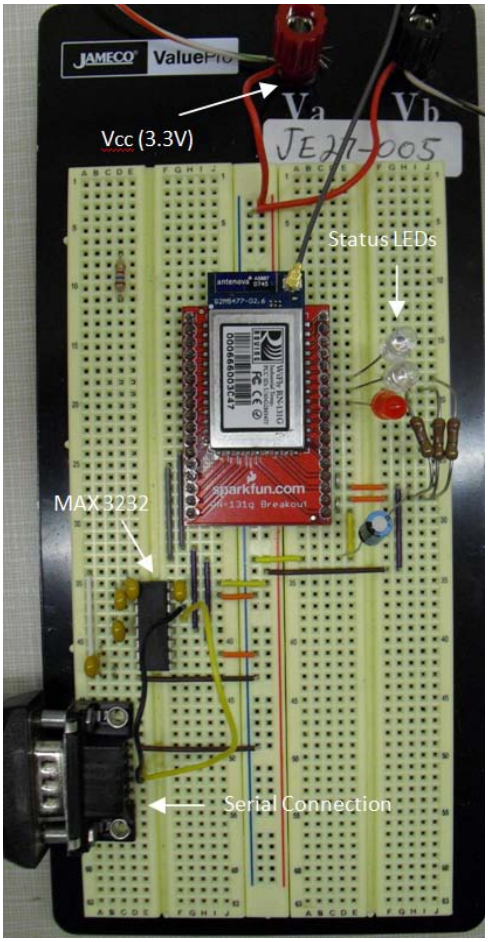


Figure 3-10: Schematic of WiFly Circuit

The WiFly module has many built-in commands to retrieve and manage information about wireless networks, but the functions of interest perform essentially the same tasks as the functions from previous methods. The command “show rssi” has exactly the same functionality and drawbacks as previous RSSI functions, but the “scan” command differs from prior functions in that it takes an integer as argument that specifies the length of time in milliseconds to spend scanning each channel. The default value is 200 ms/channel, resulting in a total scan time of 3 seconds. This is far too slow to be practical in this application, but with scan times of 30 ms/channel or less, the total time can be reduced dramatically. The scan time could be reduced even further if individual channels of interest could be scanned and all others ignored, but this is beyond the capability of the WiFly module. Thus, there is a distinct tradeoff as scan times get

smaller. As the total scan time approaches useful values, the scan is less likely to detect the signal broadcast by the wireless beacon; the WiFly module will not spend enough time on each channel to ensure the detection of the beacon's SSID broadcast. As a result, there were no values of the scan time that resulted in an acceptable data acquisition frequency when the dropped signals were accounted for.

Using the straightforward commands of the WiFly resulted in the same problems that were encountered with both previous methods: calling the stored RSSI values is fast enough but results in unpredictable delays, while the scan function gives RSSI values on command, but is far too slow for practical use. Since calling the stored RSSI values would work if the exact time the readings were taken were known, the "join" command was used to attempt to have the WiFly refresh the RSSI readings on command. Once the join command was used to associate the WiFly with the beacon, the show rssi command could be used to read the newly updated RSSI reading. Unlike any other method attempted thus far, the WiFly is able to provide RSSI readings updated on demand at a frequency of approximately 4-5 data points per second. Since the antenna is typically rotated at about .5 Hz, and 10 revolutions are used to determine the bearing, there will be 80-100 data points used in the bearing calculation, which is more than adequate. However, while this method is effective for characterizing the antenna and algorithm performance in the presence of one beacon, for practical applications with multiple beacons this method would become increasingly inefficient. Since the WiFly module would have to obtain readings from each beacon in series with one another, rather than parallel, the frequency of the data acquisition would be inversely related to the number of beacons, making the method infeasible even with only one or two additional beacons. This could be solved either by using a different WiFly chip for every one or two beacons, or by writing a custom driver for a wireless adapter that could scan individual channels quickly.

3.4 Angle-Finding Algorithm

Once the WiFly chip was configured so as to provide RSSI data with a high enough frequency, this data had to be integrated with the encoder data so as to provide a reasonable estimate of the bearing of the beacon. The algorithm needed to be able to read the encoder data from the DAQ board, the RSSI readings from the WiFly, and calculate the bearing based on this data, but in such a way that no task interfered with the timing of the others. This was especially important in the RSSI and encoder reading, because in order for any reasonable calculation of the bearing to be possible, the algorithm has to be able to match an RSSI reading exactly with its corresponding encoder count. This is further complicated by the fact that it is not possible to determine exactly how long it will take for the WiFly chip to update the RSSI value. To solve this problem, the algorithm operates in three loops: one is constantly updating the encoder count and storing the value to a global variable, another is reading the RSSI values as they become available and matching them to the latest encoder count, and the last uses this data to calculate the bearing after every complete revolution of the antenna.

In order to ensure synchronization, no loop executes until both the serial connection and the connection to the DAQ board are initialized. Once they have been initialized, the program waits until the antenna passes the sensor at 0 degrees to start all three loops. This ensures a constant reference point between trials. The encoder loop is relatively simple, as it is only necessary to update the encoder count global variable as fast as the information is available. To restrict the encoder count, and therefore the calculated angle, to a set range of 360 degrees, the total number of encoder ticks is divided by the number of ticks per revolution and the remainder is given as the encoder data. In order to account for any drift that may occur, the encoder count is periodically reset by the sensor placed at 0 degrees.

The RSSI data loop is executed in much the same way as the encoder data loop: the RSSI values are read and recorded as fast as possible, and stored with the corresponding encoder ticks to be analyzed in the third loop. The difficulty arises in deciding exactly when to read the encoder count so as to have it align correctly with the received signal strength. Since the RSSI value is updated when the WiFly joins the network, and not when it outputs its stored signal strength, the global encoder count variable must be read within the subprogram that updates the RSSI value. At exactly which point in this subprogram to update the variable had to be determined by trial and error. The last task of this loop is to check for inconsistent serial communication. Occasionally the serial communication with the WiFly module will drop out of synch and must be reset. If the program detects that no data has been retrieved from the WiFly for more than a few iterations, it will terminate the connection and restart it.

The third loop is where the data is processed to determine the bearing of the beacon. This loop is kept separate from the other two so that the calculations can run at their own pace and not interfere with the timing of the data acquisition loops. Rather than having to wait for the calculations to finish before retrieving more data, the RSSI loop can continue to collect data as fast as possible, while the previous data is interpreted simultaneously. Each revolution, the data received from the encoder and the WiFly are added to previous values in an array, which is then used to calculate the bearing. The main assumption of the method used to calculate the bearing is that the data will have one dominant peak (where the maximum value of the peak is equal to the maximum received value of that cycle) that is aligned with the direction of the beacon. If there is more than one dominant peak, the algorithm is not able to discern the true direction, but will usually indicate that the reading is unreliable.

The challenge for calculating a bearing based on data that is usually fairly noisy is how to use as much useful information for the calculation as possible while disregarding the data that is incorrect or otherwise useless. Since the only part of the data that really matters is the dominant peak and the data in its immediate vicinity, the first step is to the find the peak and filter everything else out. There is a property of the RSSI reading that helps out when it comes to this. The RSSI reading is given by the WiFly (and by most driver software) in units of decibels, represented as a negative integer, which are then converted to a linear scale by the following equation before any calculations are done.

$$RSSI_{lin} = 10^5 * 10^{\frac{RSSI_{dB}}{10}}$$

This conversion is made in order to get a better idea of the relative magnitudes of the signal for use in the bearing calculation, but since the readings started out in a logarithmic scale represented by integers, the values stored for the RSSI value maintain the discrete nature of the integers they were calculated from. Thus, instead of a continuum of signal strength readings, there are many discrete steps. This makes it a simple matter to find all primary and secondary peaks in the data, as they will all have the same value. Initially, this property was used to find the peak location simply by finding the centroid of the maximum values and the second-highest values and averaging them. This method proved to be too susceptible to outliers in the data, and it didn't use very much information about the shape of the peak to determine the bearing. It was eventually discarded in favor of an algorithm which located the dominant peak that was most likely correct, and fitting a polynomial to the data surrounding it.

To find this optimum data segment for the peak calculation, an assumption is made that the true peak would have a larger number of other high values around it, whereas any outlier

caused by random signal noise would probably not. Therefore, each peak and its surrounding data points (given by a tunable constant) is given a score W equal to:

$$W = \frac{\sum_{i=1}^{N} \left(R_i\right)^3}{N}$$

Where N is the number of data points in the vicinity of the peak in question, and R_i represents each individual RSSI reading in the vicinity. The peak with the highest value of W is then determined to be the ideal peak to use for the bearing calculation.

Once the irrelevant data are filtered out, the remaining data is fit with a 4^{th} order polynomial using the least squares method, with each point given a weight equal to the RSSI reading cubed. A polynomial fit is used so that the entire shape of the peak is used in the calculation, which would not be true in the case of a simple average of the maximum value locations. Since the points that make up the shape of the peak are usually greatly outnumbered by the points with lower values, the weights are added so more attention is paid to the peak, and the baseline values are ignored. This results in a fit that is much sharper and well defined. Once the data is fit with the polynomial, the maximum value is found, and its location corresponds to the direction of the wireless beacon. This process is illustrated in Figure 3-11. The polynomial fit can be executed after every revolution, but several iterations are usually necessary to converge onto a single solution. The number of revolutions per bearing calculation can be specified, but preliminary experiments indicated that ten revolutions per calculation were sufficient for convergence in most cases.

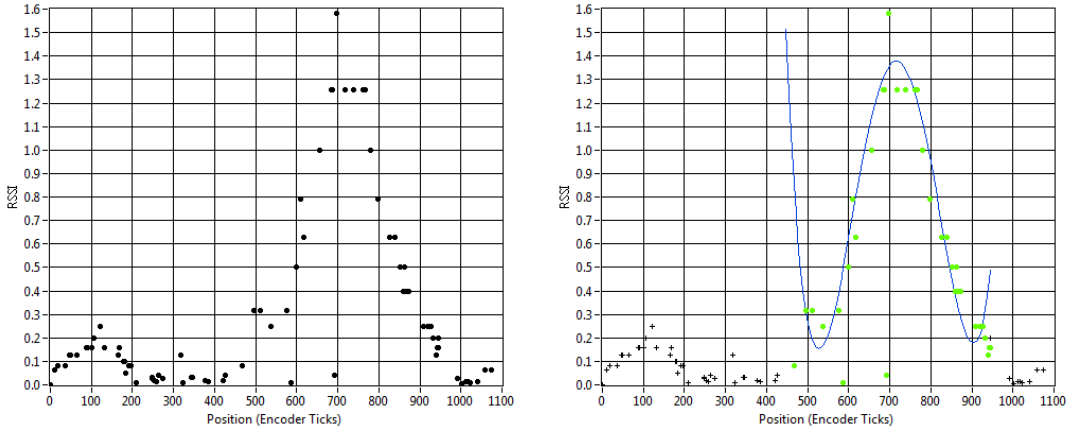


Figure 3-11: Typical Data Set and Polynomial Fit
Left: RSSI data points for 10 revolutions. Notice the primary peak as well as the secondary peak from the side lobes in the antenna radiation pattern
Right: The data around the primary peak (green) and the polynomial fit to determine the bearing (blue)

Once ten revolutions worth of encoder and signal strength data are collected, there are several ways the data can be used to determine the bearing. One possibility is to simply aggregate the data after ten revolutions and fit the polynomial to all the data at once. This ensures that the most data possible will be used in the polynomial fit, but this method makes it more difficult to determine convergence in the case of noisy data. Since the algorithm will only be determining the bearing once, there is no other information to evaluate the reliability of the measurement. Another descriptor of convergence would have to be devised if outliers and faulty data are to be ignored.

Although in most cases the polynomial fit of all of the data combined turns out to be the most accurate, averaging polynomial fits of several different subsets of the data rather than merely aggregating the data and conducting one fit can be advantageous in several circumstances. It seems probable that the segmented calculations would be more robust in the presence of multiple dominant peaks caused by multipath effects. This is a consequence of how the dominant peak is determined. Since the score of each peak is based on the values of all

surrounding data, if there is more than one dominant peak due to multipath effects or egregious signal noise, it is likely the maximum score will oscillate between the peaks over several subsets of the data, resulting in a high standard deviation. This would make it easy to recognize data sets which do not converge. An obvious choice for the division of these subsets would be to divide the data points into sets based on the revolution in which they were received. However, individual revolutions do not provide enough data to make an accurate fit. Alternatively, the data could be decimated into subsets by taking every other data point, or every third data point, etc…

Finally, the bearing could be calculated via a progressive fit on the data. Each revolution, the polynomial would be fit to all of the data received over every revolution thus far. At the end of the ten revolutions, each calculated bearing would be averaged for the final output. This method serves somewhat as a compromise between the other two. Much like the first method, this would use all of the data at once to calculate the bearing, but it will also average that with bearings calculated with different subsets of the data, resulting in a more robust calculation in the presence of false dominant peaks.

A major drawback with the second two methods is that with smaller subsets of data, it is increasingly difficult to make an accurate fit to the data that captures its shape. Thus, it is likely that there will be fits made with insufficient data that result in grossly incorrect bearings. In order for these methods to be effective, the algorithm must reject all of the bearings calculated from incorrect fits. This is done with a function designed to reject outliers from any data array. This function calculates the mean and standard deviation of the array, and then removes every point that is less than the mean minus two standard deviations, or greater than the mean plus two standard deviations. The mean and standard deviation of the new array is then

recalculated, and the array is filtered again. This iterative process continues until the function returns a stable mean and standard deviation. This function has proved very useful in filtering outliers from data sets that should normally be grouped somewhat tightly around their mean. For a normally distributed data set, two standard deviations about the mean describes about 96% of the data, so it is assumed that any values outside of this range are not representative of the rest of the data. Even if the set in question does not strictly follow a normal distribution, it is still reasonable to assume that data obtained from a sensor in identical circumstances should be somewhat tightly clustered around their mean.

Once the outliers have been removed from the bearing array, convergence is determined based on its standard deviation. If after the set number of revolutions the standard deviation is less than ten, the bearing is determined to have converged. This convergence check ensures that if there is too much noise, and thus no clear signal peak from which to calculate a bearing, the data given by the program will be disregarded as inaccurate.

Since there seem to be benefits and drawbacks for each proposed averaging method, one of the objectives of data analysis will be to assess the accuracy of each method. Also, in the methods where fitting function is iterated over several subsets, the size of the data sets will be varied to find an optimum size, if it exists.

4 Results

4.1 Data Acquisition and Filtering

Once the prototype antenna became functional, experiments were performed in order to determine the potential accuracy of this triangulation method. A Zonet 802.11 a/g wireless router operating on channel 6 served as the beacon in these trials, and was tested in several environments an autonomous lawnmower would likely encounter. Of primary importance was discovering the effect of common yard objects on the directional antenna signal. The objects tested were bushes, trees, and general plant life, chain link fences with metal poles, people, and buildings. An open field with no obstructions or objects in the vicinity was used as a baseline case.

There were two strategies used to acquire data. In the first, each scenario was classified as

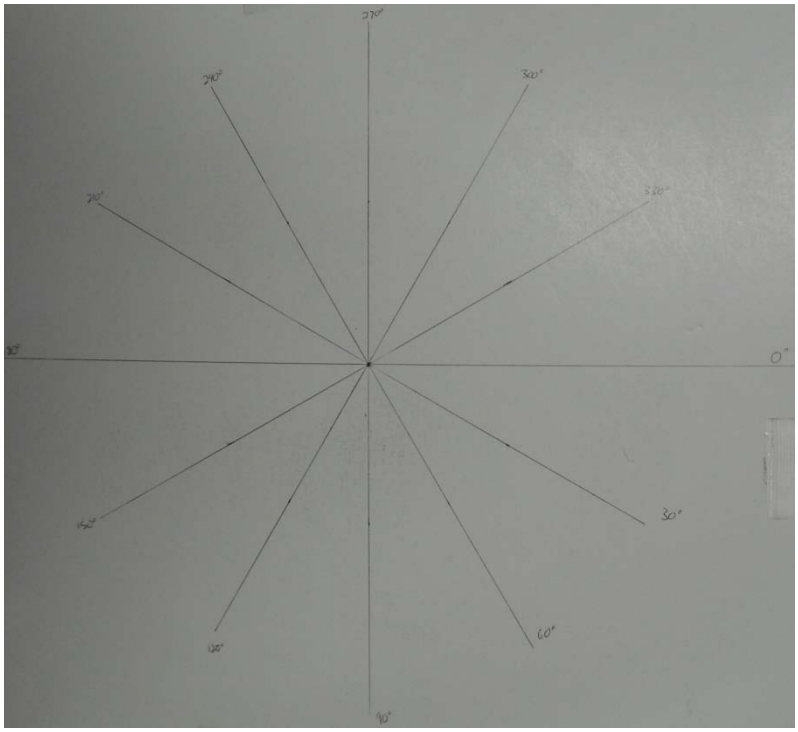


Figure 4-1: Antenna Testing Board

completely as possible. For each obstacle type the beacon was tested at distances of 15 and 30 feet from the antenna to classify the effects of range. Once the beacon was placed in each trial, a board with angle demarcations for every 30 degrees from 0 to 360 (Figure 4-1) was fixed to the table such that the beacon was aligned with 0°. This provided an absolute reference frame from which the antenna could measure the bearing of the beacon. Since the algorithm will give an angle only with respect to the support it is mounted on, it is important that this support be fixed to a reference frame that has a known displacement from the beacon. This allows the received bearing to be checked against the actual bearing to test for any bias that is consistent between different orientations. For each angle demarcation, the direction finding algorithm was executed fifteen times. Since the antenna rotates ten times per execution of the algorithm, a total of 150 revolutions worth of data were collected per demarcation.

There are two main properties of the data used to quantify the accuracy of each reading for these trials: the average offset of the measured values from the reference values, and the error between the data and this offset. These two numbers measure how accurate the antenna is with respect to the absolute reference frame, and its own relative frame, respectively. It is expected that over the course of a trial, with all else held equal, the antenna should give consistent readings of the location of the beacon relative to the absolute frame. In other words, the relationship between the reference angle and the measured angle should be strictly linear, and the slope should be equal to one. As the antenna base is rotated a fixed amount, its perception of the beacon's location should change by the same amount. From this information, a line can be fit to the data by finding the average offset of the measured values from the reference values. This is the first number used to quantify the accuracy of the antenna: if everything is properly aligned, the value of the average offset should be zero or very close to

zero. If it is not zero, it is possible some obstacle is introducing a bias into the signal received from the beacon.

Even in the presence of a constant bias, however, it is still expected that the readings from the antenna should follow a straight line. Therefore, the data will also be judged based on the average square error between the measured values and the expected linear relationship. This is similar to the statistical calculation of the variance of a sample. The variance from the mean of a sample is calculated by:

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2 \tag{4}$$

Where $\overline{x}$ is the mean of the sample, N is the size of the sample, and x_i represents each data point in the sample. For quantifying the error of the antenna, the value obtained from the linear model was used instead of the mean, resulting in the mean squared error as follows:

$$s^2 = \frac{1}{N}\sum_{i=1}^{N}(\theta_i - \phi_i - \overline{\phi}_{os})^2 \tag{5}$$

Where θ_i is a measured angle, ϕ_i is its corresponding reference angle, and $\overline{\phi}_{os}$ is the average offset angle calculated by:

$$\overline{\phi}_{os} = \frac{1}{N}\sum_{i=1}^{N}(\theta_i - \phi_i) \tag{6}$$

The root mean squared error, s, should give a good idea of the magnitude of error that can be expected from this localization technique.

In the second data acquisition strategy more emphasis was placed on testing a diversity of environments. Rather than test every angle demarcation, the antenna was set such that the beacon was at exactly 180 degrees. Next, for each environment there were ten iterations of the angle finding algorithm run. Since each iteration consists of ten revolutions, each trial consisted of 100 revolutions of the antenna. This truncated sample size enables trials to be taken in many more environments without sacrificing too much useable data. This assumption is corroborated by all prior experiments with the antenna which indicated that throughout a given trial, each angle demarcation exhibits similar behavior to all the others. Thus, every demarcation for a given environment need not be checked, and the behavior of the antenna can be classified well with just one orientation. Once the data is collected for this case, it is interpreted in the same manner as before: via the average offset from the reference angle, and the root mean squared error with respect to the mean.

4.2 Data Logs

For each data log, the calculated angle is plotted verses the reference angle. A line representing the average offset is drawn over the data to give a reference for how consistent the measured bearings are with the others from the same trial. The closer the data is to the line, the more consistent the measurements. The value of the RMS error under each plot is calculated using the progressive fit method. The different methods will be compared in a later section. For each trial, the antenna was rotating in a clockwise fashion, so clockwise represents the positive direction for angular displacement. The beacon location is highlighted in yellow in each picture.

4.2.1 Inside Logs

All of the logs taken inside were intended to test the relative accuracy of the antenna rather than the absolute accuracy, so the absolute frame was not aligned with the beacon prior to the trials.

Log 1: 17 feet, 8 inches

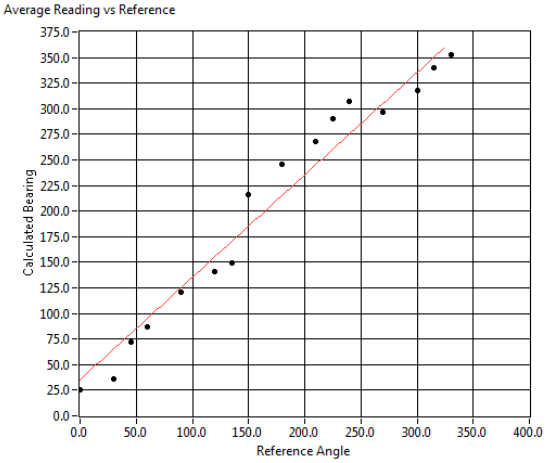


Figure 4-2: Log 1 Data
RMS Error: 19.86 degrees

Log 2: 29 feet

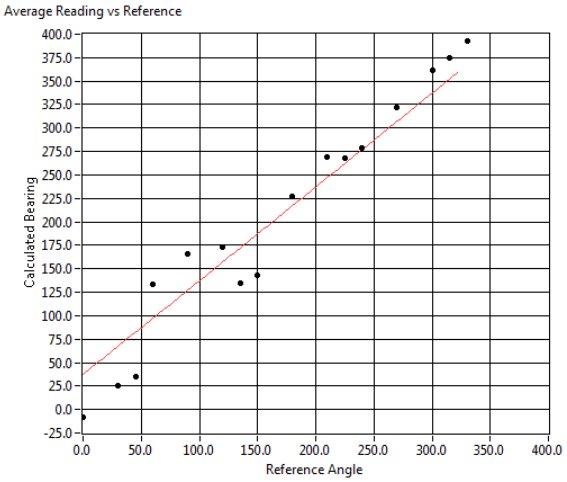
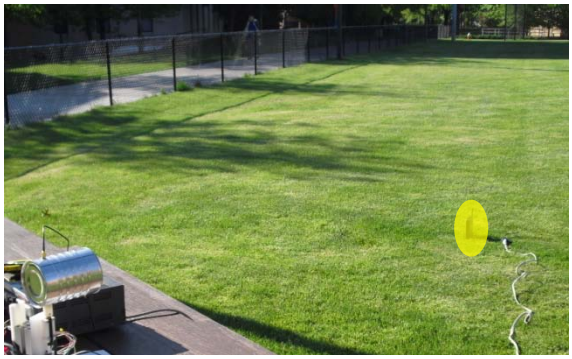


Figure 4-3: Log 2 Data
RMS Error: 32.9 degrees

4.2.2 Outside Logs

From the results of indoor trials, it was discovered that the metal objects and clutter of an interior environment produces too much interference to provide a good classification of the method. Thus, additional trials were taken outdoors with various likely obstacles. The first two trials were not aligned with the beacon direction, and so only give a measure of the relative accuracy without accounting for any constant bias.

Log 1: 15 feet, no obstructions



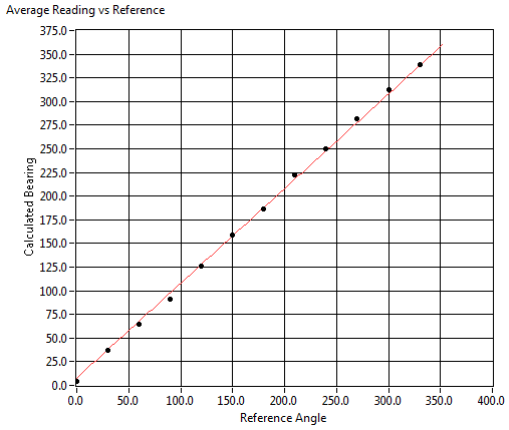


Figure 4-4: Outside Log 1 Data
RMS Error: 3.50 degrees

Log 2: 30 feet, obstructed by bushes

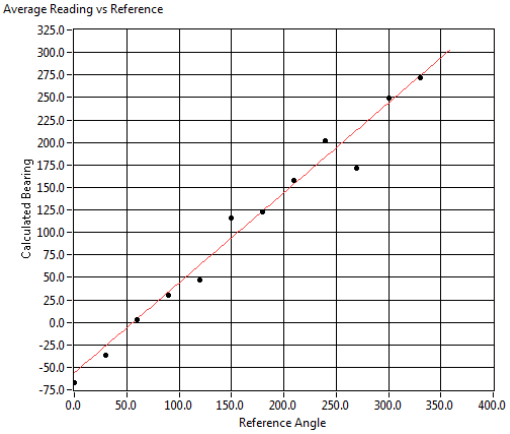


Figure 4-5: Outside Log 2 Data
RMS Error: 16.79 degrees

Log 3: 30 feet, unobstructed
This is the first log with an absolute reference angle

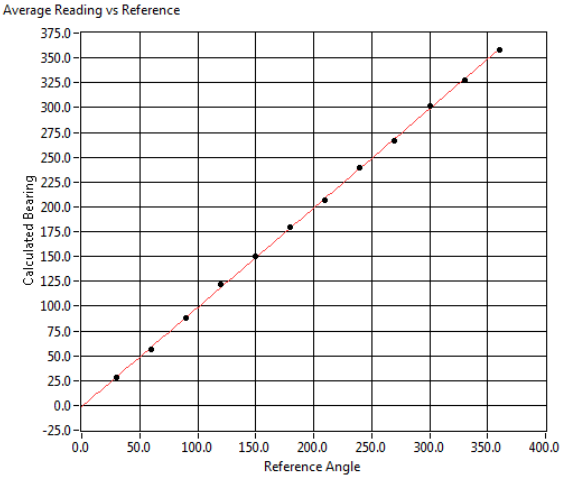


Figure 4-6: Outside Log 3 Data
RMS: 2.13 degrees
Offset: -1.34 degrees

Log 4: 15 feet, obstructed by chain link fence

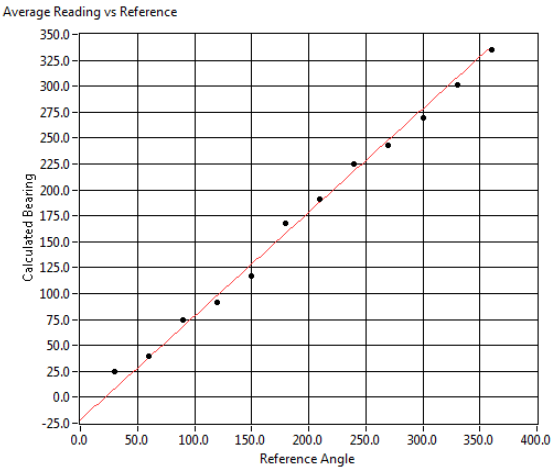


Figure 4-7: Outside Log 4 Data
RMS: 8.63 degrees
Offset: -21.63 degrees

Log 5: 60 feet, beacon in same location as Log 4

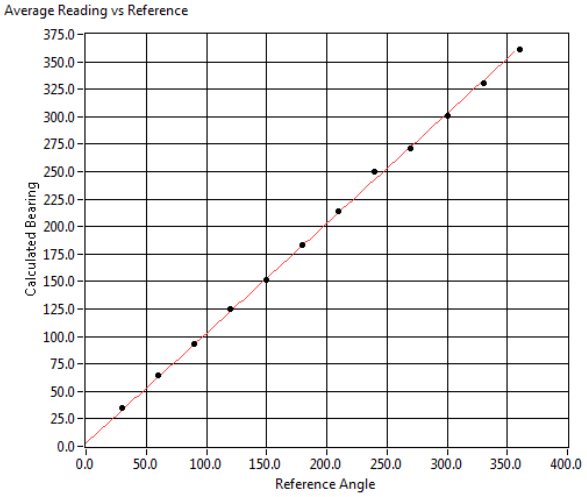


Figure 4-8: Outside Log 5 Data
RMS: 3.23 degrees
Offset: 2.91 degrees

Log 6: 30 feet, obstructed by chain link fence

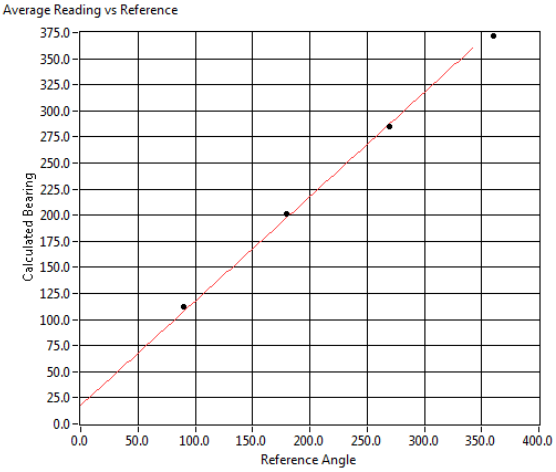


Figure 4-9: Outside Log 6 Data
RMS: 4.80 degrees
Offset: 17.77 degrees

Log 7: 15 feet, beacon in same location as Log 6

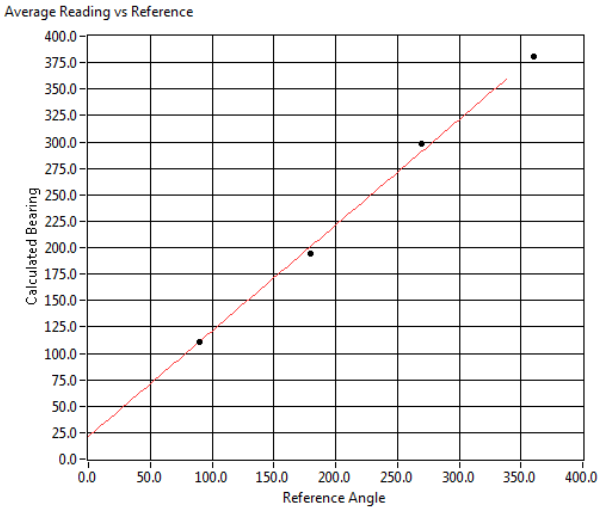


Figure 4-10: Outside Log 7 Data
RMS: 6.25degrees
Offset: 21.21degrees

Log 8: 20 feet, obscured by bushes

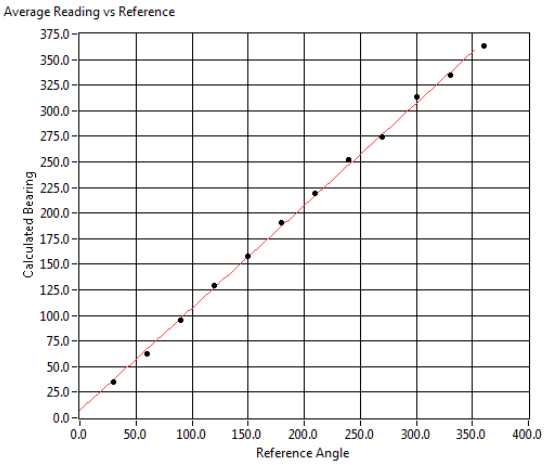


Figure 4-11: Outside Log 8 Data
RMS: 3.66 degrees
Offset: 7.29 degrees

Summary (errors in degrees):

Trial	RMS Error	Offset
IL 1	19.86	N/A
IL 2	32.9	N/A
OL 1	3.5	N/A
OL 2	16.79	N/A
OL 3	2.13	-1.34
OL 4	8.63	-21.63
OL 5	3.23	2.91
OL6	4.8	17.77
OL7	6.25	21.21
OL 8	3.66	7.29

Figure 4-12: RMS Error and Offset for Inside (IL) and Outside (OL) Logs

4.2.3 Truncated Trials

The truncated trials tested the antenna in the environments shown in Figure 4-12, with the following obstacles:

- Truncated Logs (TL) 1-4: These trials tested operating the antenna at 4 different speeds with the beacon in an open field to see if any delay was introduced
- TL 5: Human obstacle halfway between beacon and antenna
- TL 6: Human obstacle halfway between beacon and antenna, but standing approximately 30 degrees to the left
- TL 7: Human obstacle level with beacon, but standing approximately 1 foot to the left

Figure 4-13: Truncated Trial Locations
Top Left: Trials 1-8, Distance: 15 feet
Top Right: Trial 9, Distance: 30 feet
Bottom: Trial 10, Distance: 30 feet

- TL 8: The beacon was placed underneath a cardboard box
- TL 9: The beacon was placed behind bushes and ivy, and near a tree and a brick building
- TL 10: The beacon was placed directly in front of glass and metal doors

Results (errors in degrees):

Trial	RMS Error	Offset
TL 1	1.85	-2.25
TL 2	3.6	-2.04
TL 3	1.23	1
TL 4	0.91	-0.14
TL 5	3.33	-26.99
TL6	6.65	2.18
TL 7	2.16	-0.77
TL 8	1.87	-2.02
TL 9	0.63	-2.25
TL 10	1.59	-1.25

Figure 4-14: Error and Offsets for Truncated Logs (TL)

4.2.4 Evaluation of Different Fit Methods

Figure 4-15 and Figure 4-16 show charts relating the RMS error and the offset, respectively, of each of the fit methods that was discussed in Section 3.4 for each trial. The progressive method fits the data as it arrives, and then averages the bearings from each revolution, the aggregate method calculates one fit using all of the data, and the decimation method takes data points at set intervals to create the specified number of sub-arrays. The data used to generate these plots is shown in Figure 7-2 in the appendix.

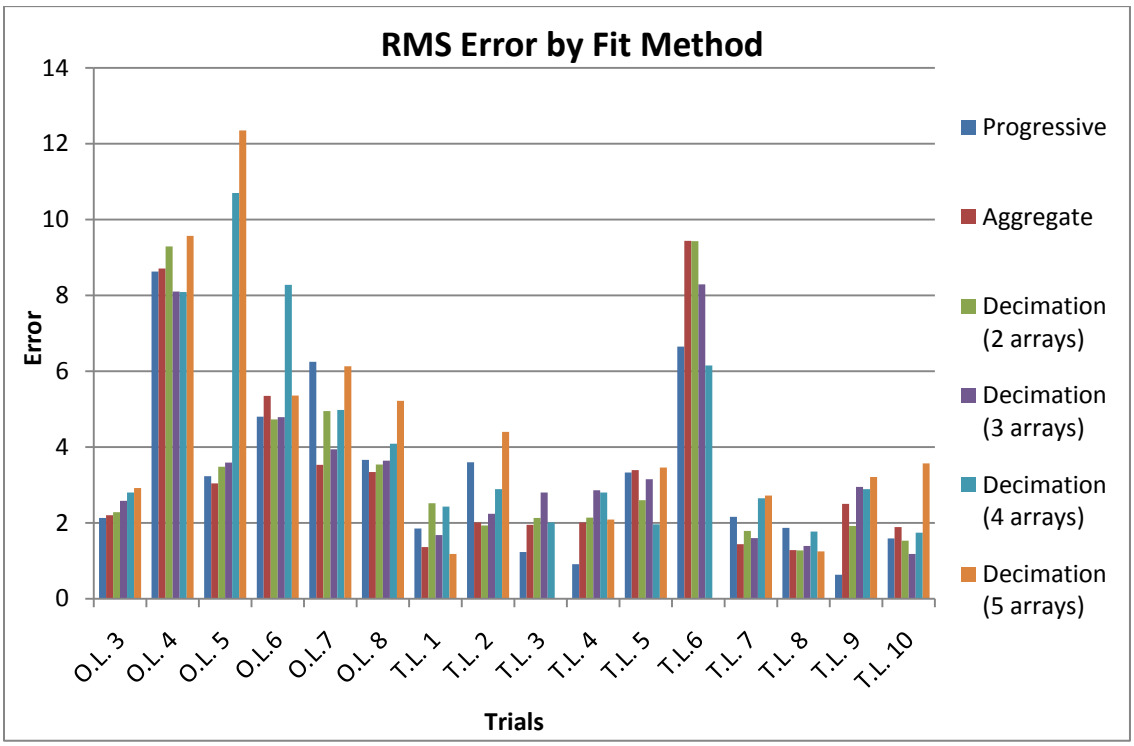


Figure 4-15: Fit Comparison (RMS Error)

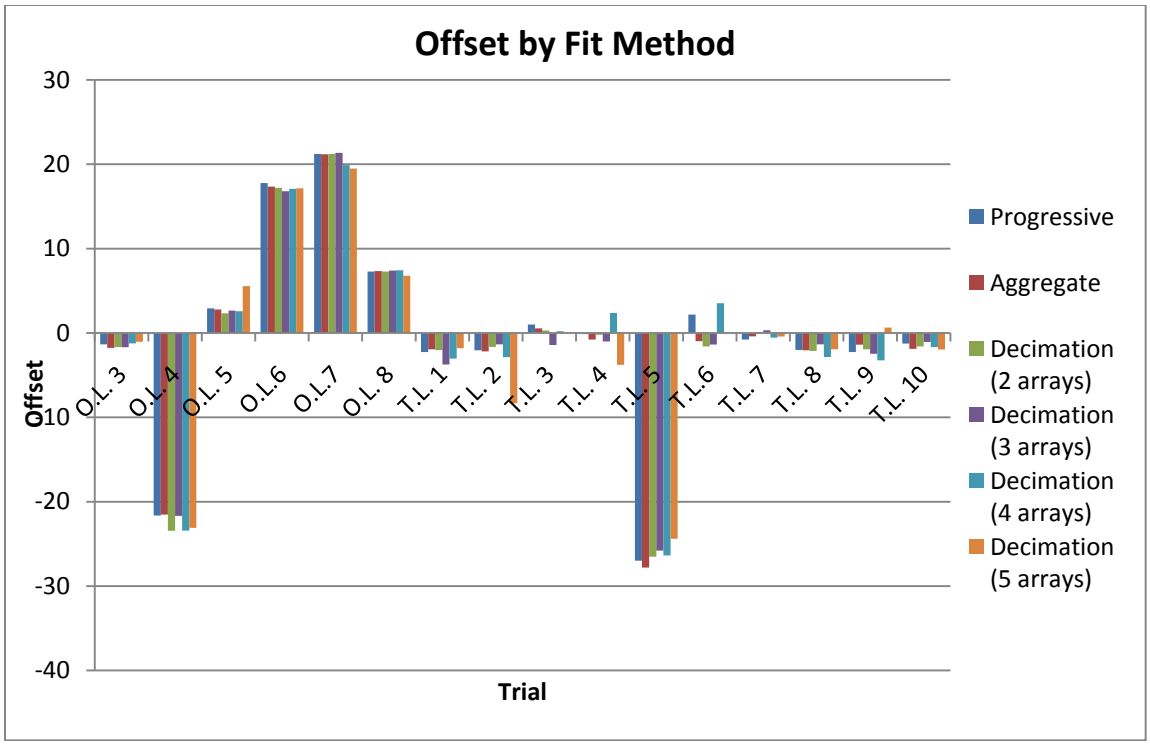


Figure 4-16: Fit Comparison (Offset)

From this data, it does not seem as if one fit is better than the others. There are situations where each algorithm has the advantage, and where each has a disadvantage. Though no method is optimal, it does appear that 4 and 5 array decimation are not viable methods, since there are several instances where they result in significantly higher errors than any other method. Other than this exception, it would seem that all of the methods are of comparable accuracy.

5 Analysis and Discussion

Overall, the results obtained from the trials demonstrate a very good level of accuracy. For every trial without obstructions (OL 1, OL3, TL 1-4), the RMS error is less than 4 degrees, with the mean being approximately 2.2 degrees. Additionally, in each case the offset from the true value is within the error given by the RMS. This would seem to indicate that in an unobstructed environment a typical error of less than 3 degrees could be expected from this angle finding method, with a worst case error of around 4-5 degrees. TL 1-4 also demonstrated that there is no discernible effect on the antenna accuracy caused by varying the speed between .48 and .91 Hz. However, RSSI values can only be read at a fixed speed, so as the speed of the antenna increases there will be fewer data points with which to calculate the bearing, so overall accuracy would be expected to fall off as speed continued to increase.

5.1 Indoor Accuracy

As could be expected, the beacon sensing is rather ineffective when used indoors. With RMS errors on the order of 20-30 degrees or more, accurate localization would be impossible. This is no doubt due to the large amount of RF interference that can be found indoors from metal

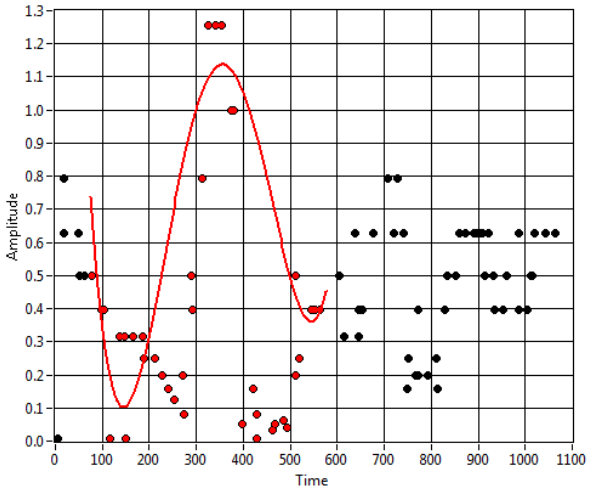


Figure 5-1: Representative Data Plot for IL 2

furniture, computer equipment, and the frame of the building. This interference is particularly apparent in Figure 5-1, which is a typical data set from the indoor trials. The plot shows one dominant peak that was fit with the polynomial, but there are also several secondary peaks that add much noise to the measurements. In addition, the magnitude of the primary peak is not much larger than the secondary peaks. This shows how much noise is introduced when the beacon is surrounded by a lot of clutter and metal objects. Fortunately, an autonomous lawnmower should not have to operate indoors, so this drawback will not be a huge problem.

5.2 Effect of Chain Link Fences

Like metal objects indoors, the data also show that chain link fences have the potential to interfere with the triangulation. Outside logs (OL) 4-7 were taken specifically to test the effect of chain link fences on the signal. While each of these trials had a relatively low RMS (mean of 5.7 degrees), the offset in each case was significant, except for trial 5, where the offset was only 2.91 degrees. Based on this information, it seems likely that rather than introduce a lot of noise into the data, the posts of the chain link fence act as secondary antennas, appearing as

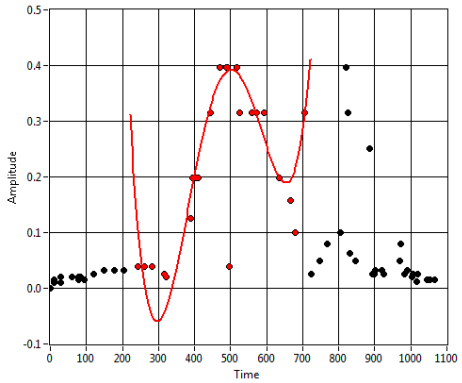


Figure 5-2: Representative Data Plot for OL 4

alternate sources of the signal, while the links dampen out the contribution from the true source. From the picture of the beacon placement in each of these trials, it appears that when the offset is accounted for, the antenna perceives the signal source as originating from the left

chain link fence post in OL 4, and the light post in OL 6 and 7. As is shown in Figure 5-2, there are two peaks in this data set, and it seems likely that they correspond to the two chain link fence posts. However, since there were more high-valued data points around the left peak, it was chosen for the polynomial fit. In this figure, it would seem that the true source is between the two visible peaks, but there are no data points there since the chain link fence is blocking the true source. Similarly, in this data set from OL 8 (Figure 5-3) there are clearly two peaks, one aligned with the true bearing (in this case 360 degrees) and one pointing towards the large metal light post. In this case, the polynomial was fit to the false peak instead of the true peak. This behavior can also account for the larger RMS values, since in one iteration the algorithm might choose the true peak, while in the next iteration it chooses the false peak. As for the surprising accuracy of OL 5, it is possible (but just speculation) that the signal damping from the chain link fence is only a short-range effect, and at larger distances the antenna is able to satisfactorily read the signals from the beacon.

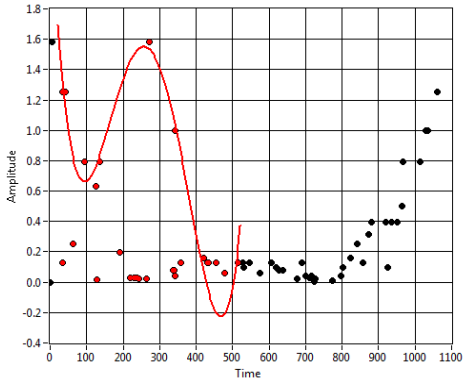


Figure 5-3: Representative Data Plot for OL 8

5.3 Other Obstacles

Chain link fences and metal obstacles have the potential to interfere with the beacon's signal, but the system has demonstrated that it is rather resistant to other types of obstacles. The effect of a person on the signal was tested in TL 5-7, which showed that people will greatly

affect the result of the direction finding algorithm only if they are standing directly between the antenna and the beacon. There was a lesser effect on accuracy when the person was standing near the line of sight but not blocking it, but the effect disappeared entirely when the person was just off to the side of the beacon. Figure x shows the flattening effect that the person has on the signal from the beacon. There are no definitive peaks, so it makes it difficult to calculate a bearing, especially when this plateau seems to have replaced the left slope of the peak. This results in a fit that is drastically skewed to the left, since the polynomial is weighted by the magnitude of the signal cubed. In this case, the calculated bearing was 146.5 degrees, where it should have been 180.

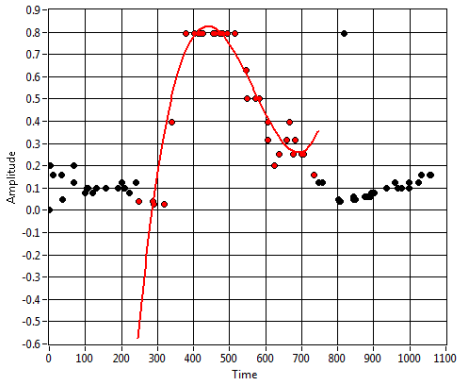


Figure 5-4: Representative Data Plot for TL 5

Non-human obstacles do not seem to interfere with the signal much at all. Even when completely obscured by a cardboard box (TL 8), the RMS was less than 2, and the offset was around -2 degrees. Interference from trees, bushes and ivy (TL 9), and a brick building with glass doors (TL 10) was similarly low. On the other hand, there was a significant RMS error for OL 2, in which the beacon was placed behind several feet of bushes. This would seem to indicate that small barriers are negligible, but if there are enough obstructions blocking the beacon's signal, the accuracy of the algorithm will be compromised.

5.4 Expected Triangulation Accuracy

If the obviously incorrect bearings are removed from the data logs, the resulting RMS error comes out to about 2.4 degrees. Taking a conservative estimate, typical accuracies of approximately 4 degrees could be expected. In order to translate this angular error into a corresponding Cartesian error in the final triangulation process, the triangulation algorithm described in 3.1 was used. The first simulation used last year's ION Autonomous Lawnmower Competition field as the input, and assumed that 6 beacons would be placed at each corner of the field. Iterating through the worst possible positional errors based on the given angular accuracy, the following error plots were produced:

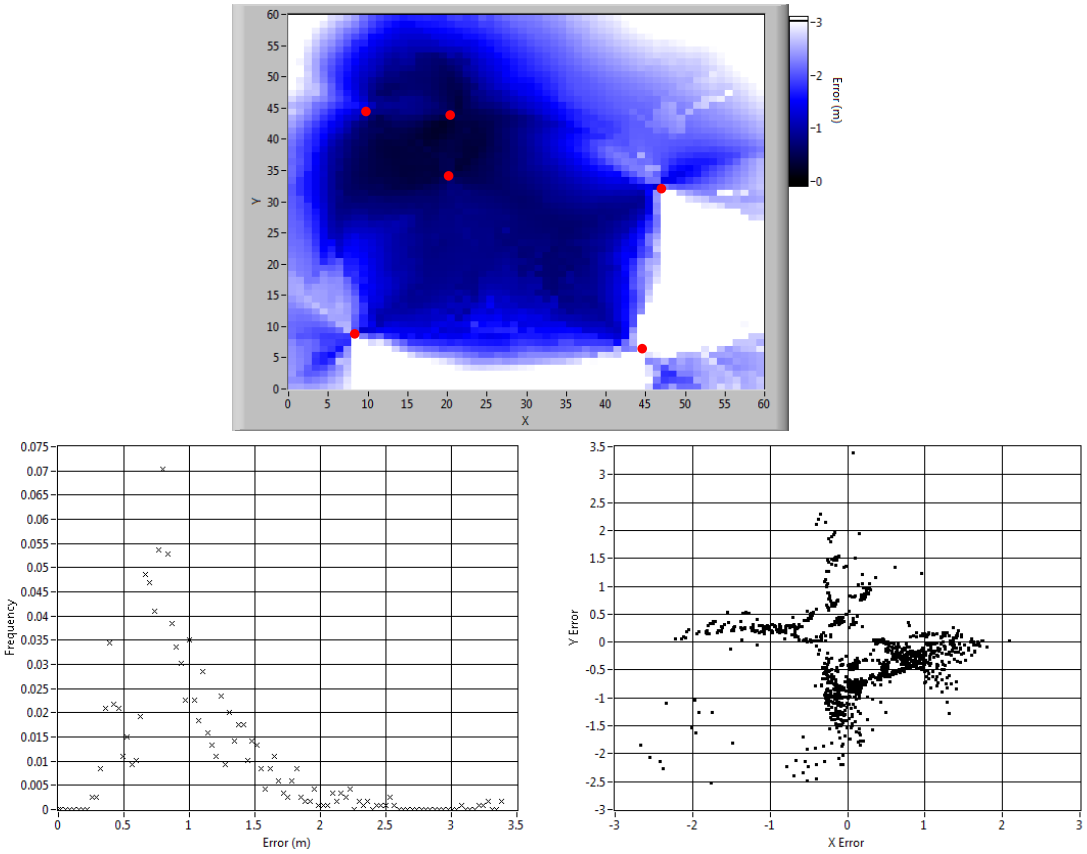


Figure 5-5: Error Plots for 6 Beacons, ± 4 Degrees
Top: Intensity plot, X and Y represent position, and darker points indicate lower error. Red dots indicate beacon placement
Bottom Left: Histogram, frequency of error plotted verses error magnitude
Bottom Right: Error in X verses Error in Y for each point
Mean Error for Interior Points: 0.97 meters
Standard Deviation: 0.45 meters

The more analytical triangulation method was used to create these plots, as it generates less error for the same inputs in every case tested thus far. The histogram and the sample statistics show that most of the error is between about .5 and 1.5 meters. This is rather good, and is on par with what can be expected from less expensive GPS units. As can be seen from the intensity plot, this lesser error is mostly located in the regions that are surrounded by the beacons. This would factor into beacon placement decisions in the final application; beacon locations on the perimeter of the mowing area would take priority. As a result of this, the calculation of the bottom two error plots, the mean, and the standard deviation considered only those points surrounded by at least 3 beacons. Additionally, as can be seen by the much lower error in the region with a high beacon density, error depends significantly on the distance of the mower from the beacons.

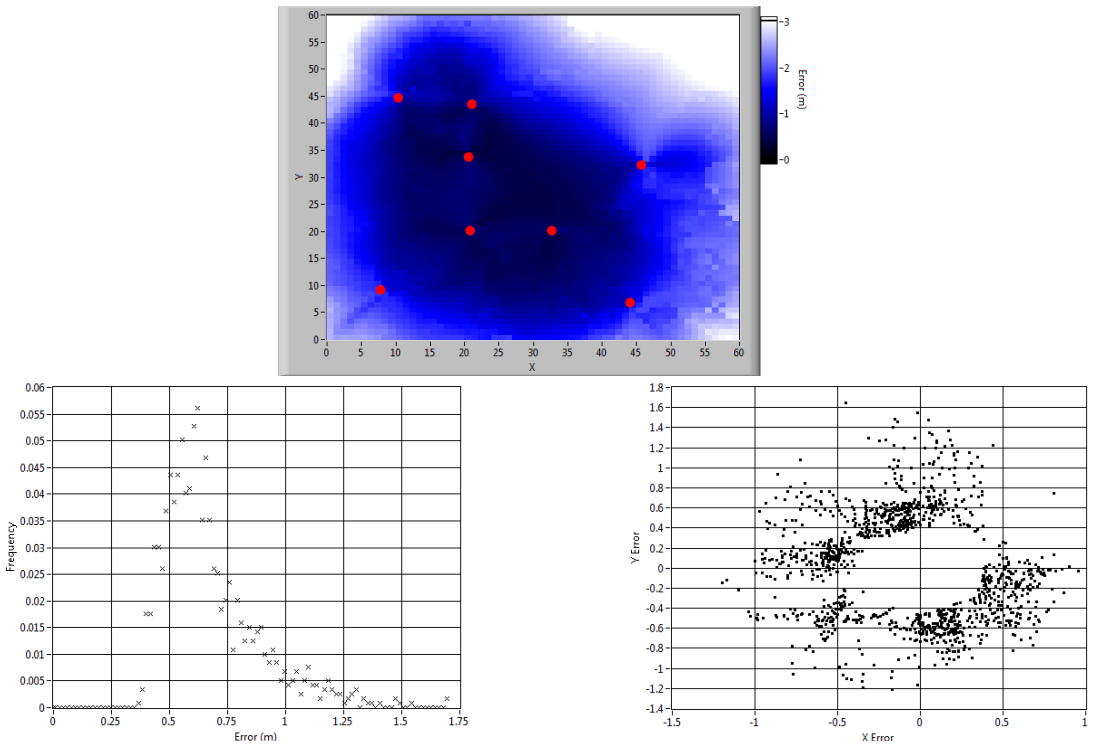


Figure 5-6: Error Plots for 8 Beacons, ±4 Degrees
Top: Intensity plot
Bottom Left: Histogram
Bottom Right: Error in X verses Error in Y for each point
Mean Error for Interior Points: 0.67 meters
Standard Deviation: 0.2 meters

These considerations were taken into account for a second simulation, where two extra beacons were placed in the middle of the field to make the beacon density more uniform. The new error plots are shown in FIGURE X. With the addition of the two beacons, both the mean error and its standard deviation dropped considerably. For this case, most of the error from interior points would be between .45 and .9 meters. This shows that larger antenna errors can be compensated for by adding more beacons to the mowing area. The results also indicate that this could be a viable triangulation method, with accuracies that surpass the less expensive GPS units.

Finally, a less conservative estimate of angular accuracy was taken as just the average of the RMS errors, or approximately 2.5 degrees. This should give an indication of the errors dependence on the accuracy of the antenna. The error plots of this are shown in Figure 5-7. As expected, the errors are again drastically reduced.

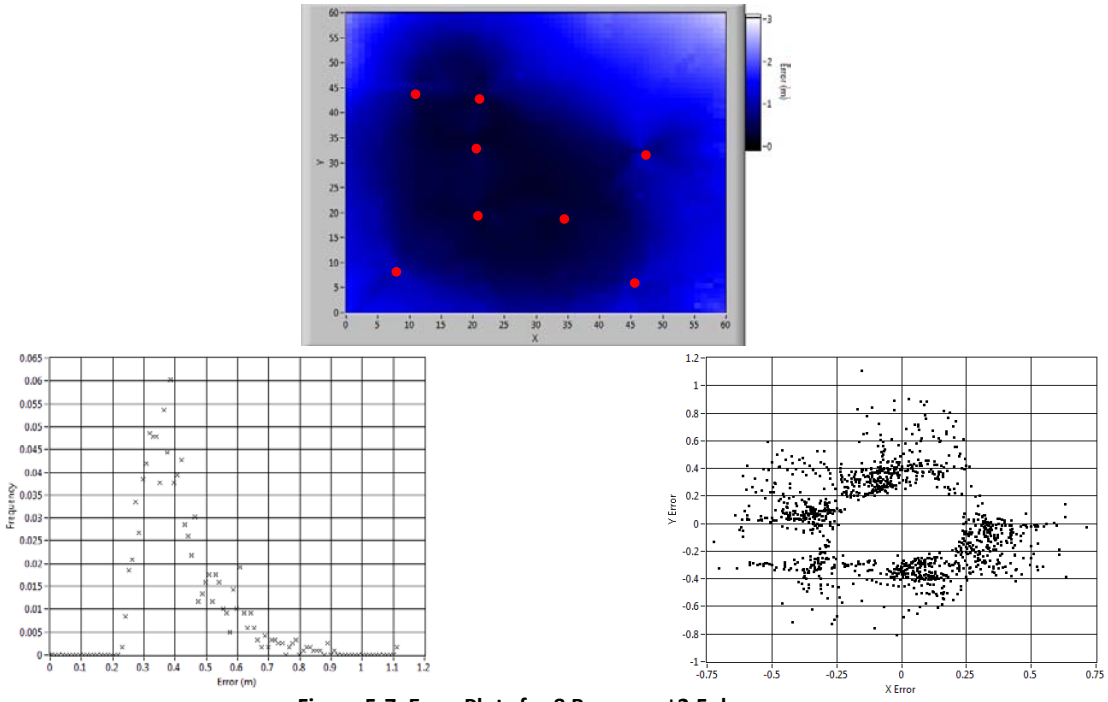


Figure 5-7: Error Plots for 8 Beacons, ±2.5 degrees
Mean Error for Interior Points: 0.41m
Standard Deviation: 0.12m

6 Conclusions

This thesis has presented and investigated a method of localizing an autonomous lawnmower that is less expensive than existing high-precision sensors, and is more accurate than current inexpensive alternatives. Using a rotating directional antenna to plot beacon signal strength verses displacement, angular positions of several beacons can be ascertained and used to triangulate the mower's position. RSSI values are obtained using a serial connection to a Roving Networks WiFly chip, which are then matched with their corresponding encoder positions. The bearings are calculated by fitting a fourth order polynomial to this data, and averaging the fit over several cycles.

There are several shortcomings to this method that would have to be overcome before implementation. First, the signal strength data is read from the WiFly chip too slowly for this method to be used to localize the mower in real time. It generally takes around 15-20 seconds to acquire enough data to make an accurate fit. As such, this algorithm would not be able to give the mower constant feedback on its position, since by the time enough data was collected, the mower would have moved a significant amount. Therefore, it makes sense to implement this as an intermittent update. The mower would localize as well as it could from other sensors while moving, and then periodically stop for a few seconds to get a more accurate position estimation.

Secondly, metal obstacles have been shown to interfere with the signal from the beacons. From the data, it appears that metal poles tend to act as false sources, tricking the algorithm into thinking they are the true beacon. Though it would require more testing, a possible solution to this would be to supply users of the lawnmower with magnetic beacons, so that

they are able to be attached to any metal poles or fences that may be present in the yard. This would allow the obstacles to act as antennas for the beacon, rather than false sources.

Another consideration is the amount of beacons to be used. Results from triangulation simulation show that adding beacons greatly increases the accuracy of the localization, but more beacons also imply higher cost and complexity. However, in (24) researchers have developed a wireless platform that is compact, low-power, and effective, at a cost that is significantly less than comparable test-beds. Since only limited functionality is required for the beacons in this application, this indicates that beacons could be produced for very little cost. With a low beacon cost, a user could be supplied with a large number of beacons, to be added to the yard at their discretion.

The intended purpose for this method is to be implemented on a commercial autonomous lawnmower. With this in mind, the chief concerns should be cost and robustness. This thesis has already demonstrated the accuracy of the method with respect to many different obstacles, and while the cost of the prototype was significant (Figure 7-3 in appendix), in production this amount would be reduced considerably. Many of the components used in the prototype were excessively effective and robust, and such hardware would not be necessary on a production model. There is still work to be done before this method is implemented, but angle based triangulation of a robotic lawnmower has been shown to be a viable localization technique.

6.1 Future Work

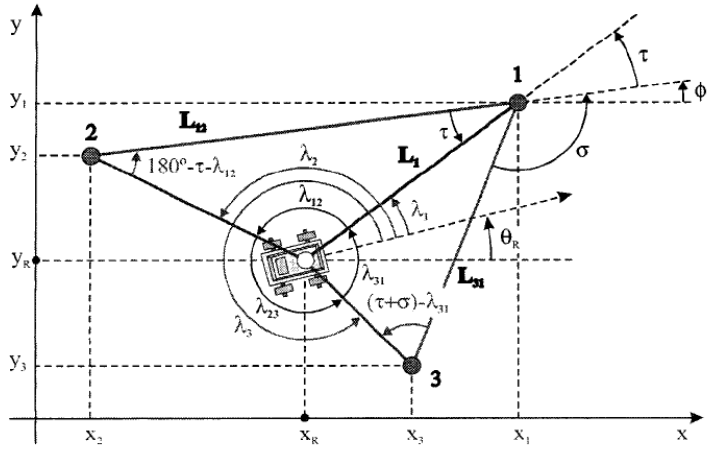
As mentioned before, much work still must be done before this method can be implemented:

- Investigate new integrated circuit design to read RSSI and encoder ticks

- Create efficient driver for wireless circuit that will return desired values efficiently
- Test angle finding algorithm with multiple beacons
- Implement more directional enclosed yagi antenna

7 Appendix

7.1 Geometric Triangulation Algorithm



Angle Ranges

$0° \le \lambda_1 < 360°$
$0° \le \lambda_2 < 360°$
$0° \le \lambda_3 < 360°$

$0° < \lambda_{12} < 360°$
$0° < \lambda_{23} < 360°$
$0° < \lambda_{31} < 360°$

$-180° < \phi \le 180°$
$-180° < \sigma \le 180°$
$-180° < \tau \le 180°$

$-180° < \theta_R \le 180°$

Generalized Geometric Triangulation Algorithm

1. If there are less than three visible beacons available. then return a warning message and stop.
2. $\lambda_{12} = \lambda_2 - \lambda_1$
3. If $\lambda_1 > \lambda_2$ then $\lambda_{12} = 360° + (\lambda_2 - \lambda_1)$
4. $\lambda_{31} = \lambda_1 - \lambda_3$
5. If $\lambda_3 > \lambda_1$ then $\lambda_{31} = 360° + (\lambda_1 - \lambda_3)$
6. Compute L_{12} from known positions of beacons 1 and 2.
7. Compute L_{31} from known positions of beacons 1 and 3.
8. Let ϕ be an oriented angle such that $-180° < \phi \le 180°$. Its origin side is the image of the positive x semi-axis that results from the translation associated with the vector which origin is (0, 0) and ends on beacon 1. The extremity side is the part of the straight line defined by beacons 1 and 2 which origin is beacon 1 and does not go by beacon 2.
9. Let σ be an oriented angle such that $-180° < \sigma \le 180°$. Its origin side is the straight line segment that joins beacons 1 and 3. The extremity side is the part of the straight line defined by beacons 1 and 2 which origin is beacon 1 and does not go by beacon 2.
10. $\gamma = \sigma - \lambda_{31}$
11. $\tau = \tan^{-1}\left[\dfrac{\sin\lambda_{12}\cdot(L_{12}\cdot\sin\lambda_{31} - L_{31}\cdot\sin\gamma)}{L_{31}\cdot\sin\lambda_{12}\cdot\cos\gamma - L_{12}\cdot\cos\lambda_{12}\cdot\sin\lambda_{31}}\right]$
12. If $\begin{cases}\lambda_{12} < 180° \\ \tau < 0°\end{cases}$ then $\tau = \tau + 180°$
13. If $\begin{cases}\lambda_{12} > 180° \\ \tau > 0°\end{cases}$ then $\tau = \tau - 180°$
14. If $|\sin\lambda_{12}| > |\sin\lambda_{31}|$ then $L_1 = \dfrac{L_{12}\cdot\sin(\tau + \lambda_{12})}{\sin\lambda_{12}}$
15. else $L_1 = \dfrac{L_{31}\cdot\sin(\tau + \sigma - \lambda_{31})}{\sin\lambda_{31}}$
16. $x_R = x_1 - L_1\cdot\cos(\phi + \tau)$
17. $y_R = y_1 - L_1\cdot\sin(\phi + \tau)$
18. $\theta_R = \phi + \tau - \lambda_1$
19. If $\theta_R \le -180°$ then $\theta_R = \theta_R + 360°$
20. If $\theta_R > 180°$ then $\theta_R = \theta_R - 360°$

Figure 7-1: Generalized Geometric Triangulation Algorithm (19)

7.2 Polynomial Fit Method Comparison (all errors in degrees)

	Progressive		Aggregate		Decimation (2 arrays)		Decimation (3 arrays)		Decimation (4 arrays)		Decimation (5 arrays)			
Trial	RMS Error	Offset	RMS Error	Offset	RMS Error	Offset	RMS Error	Offset	RMS Error	Offset	RMS Error	Offset	**Lowest Error**	**Lowest Offset**
O.L. 3	2.13	-1.34	2.2	-1.77	2.28	-1.67	2.58	-1.69	2.8	-1.24	2.92	-1.04	Progressive	Dec. (5 arrays)
O.L. 4	8.63	-21.63	8.71	-21.52	9.29	-23.44	8.1	-21.67	8.09	-23.42	9.57	-23.09	Dec. (4 arrays)	Aggregate
O.L. 5	3.23	2.91	3.04	2.78	3.48	2.34	3.59	2.66	10.7	2.58	12.35	5.55	Aggregate	Dec. (2 arrays)
O.L.6	4.8	17.77	5.35	17.36	4.73	17.19	4.79	16.8	8.28	17.08	5.36	17.137	Dec. (2 arrays)	Dec. (2 arrays)
O.L.7	6.25	21.21	3.53	21.18	4.95	21.22	3.94	21.36	4.98	19.89	6.13	19.49	Aggregate	Dec. (5 arrays)
O.L. 8	3.66	7.29	3.34	7.34	3.54	7.29	3.64	7.4	4.09	7.43	5.22	6.79	Aggregate	Dec. (5 arrays)
T.L. 1	1.85	-2.25	1.36	-1.93	2.52	-2.03	1.68	-3.73	2.43	-3.03	1.18	-1.81	Dec. (5 arrays)	Dec. (5 arrays)
T.L. 2	3.6	-2.04	2.01	-2.19	1.93	-1.66	2.24	-1.35	2.89	-2.87	4.4	-8.32	Dec. (2 arrays)	Dec. (3 arrays)
T.L. 3	1.23	1	1.95	0.54	2.13	0.31	2.8	-1.42	2.01	0.21	NaN	NaN	Progressive	Dec. (4 arrays)
T.L. 4	0.91	-0.14	2.01	-0.78	2.14	-0.21	2.86	-1	2.8	2.37	2.09	-3.78	Progressive	Progressive
T.L. 5	3.33	-26.99	3.39	-27.8	2.6	-26.51	3.15	-25.8	1.96	-26.36	3.46	-24.39	Dec. (4 arrays)	Dec. (5 arrays)
T.L.6	6.65	2.18	9.44	-0.96	9.43	-1.59	8.29	-1.37	6.15	3.53	NaN	NaN	Dec. (4 arrays)	Aggregate
T.L. 7	2.16	-0.77	1.44	-0.39	1.79	0.097	1.6	0.33	2.65	-0.55	2.72	-0.41	Aggregate	Dec. (2 arrays)
T.L. 8	1.87	-2.02	1.28	-2.04	1.273	-2.12	1.39	-1.34	1.77	-2.84	1.25	-1.93	Dec. (5 arrays)	Dec (3 arrays)
T.L. 9	0.63	-2.25	2.5	-1.39	1.92	-1.93	2.95	-2.46	2.89	-3.24	3.21	0.64	Progressive	Dec. (5 arrays)
T.L. 10	1.59	-1.25	1.89	-1.87	1.53	-1.6	1.18	-1.07	1.74	-1.67	3.57	-1.96	Dec. (3 arrays)	Dec. (3 arrays)

Figure 7-2: Polynomial Fit Comparison Data

7.3 Estimated Cost

Can for Waveguide	$5
Maxon Motor*	$200
Various Connectors	$10
WiFly Module	$84.95
Zonet Router x 8 beacons*	$160
NI DAQ USB-6008*	$169
Total:	**$629**

Figure 7-3: Cost Estimate Table
***These components were used for their ease in implementing with a prototype. For a final implementation, they would be replaced by far cheaper components.**

8 References

1. *A Survey of Robot Lawn Mowers.* **Hicks II, Rob Warren and Hall, Ernest L.** San Diego, USA : s.n., 2000. Proceedings of SPIE. Vol. 4719, pp. 262-269.

2. LB3510 Manual. [Online] June 18, 2008. [Cited: May 17, 2010.] <http://www.lawnbott.com/pdf/LB3510-manual.pdf>.

3. Automower Solar Hybrid. *Husqvarna.* [Online] [Cited: May 18, 2010.] <http://www.husqvarna.com/us/homeowner/products/robotic-mowers/automower-solar-hybrid/>.

4. Robomow RL2000. *Friendly Robotics.* [Online] [Cited: May 18, 2010.] <http://www.robomow.com/robomow/rl2000/>.

5. *Mobile Robot Localization by Tracking Geometric Beacons.* **Leonard, John J and Durrant-Whyte, Hugh F.** 1991. IEEE Transactions on Robotics and Automation. Vol. 7, pp. 376-382.

6. *Localization for Robot Mowers Covering Unmarked Operational Area.* **Zu, Li, Wang, Huakun and Yue, Feng.** Sendai, Japan : s.n., 2004. Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2197-2202.

7. *Nerual Networks-Based Terrain Acquisition of Unmarked Area for Robot Mowers.* **Wang, Huakun, Zu, Li and Yue, Feng.** Kunming, China : s.n., 2004. 8th International Conference on COntrol, Automation, Robotics and Vision.

8. CWRU Cutter 2 Technical Report. [Online] 2009. [Cited: May 18, 2010.] <http://www.ion.org/satdiv/alc/reports2009/CWRU_Cutter_2_Technical_Report.pdf>.

9. **Kleusberg, Alfred and Langley, Richard B.** The Limitations of GPS. *GPS World.* March/April, 1990.

10. LEA-4T: ANTARIS 4 GPS module with Precision Timing. *U-Blox.* [Online] [Cited: May 18, 2010.] <http://www.ion.org/satdiv/alc/reports2009/CWRU_Cutter_2_Technical_Report.pdf>.

11. *The Cricket Location-Support System.* **Priyantha, Nissanka B, Chakraborty, Anit and Balakrishnan, Hari.** 2000. Proceedings of ACM MobiCom'00. pp. 32-43.

12. *Range-only SLAM with a Mobile Robot and a Wireless Sensor Networks.* **Menegatti, Emanuele, et al.** Kobe, Japan : s.n., 2009. IEEE International Conference on Robotics and Automation.

13. **Whitehouse, Kamin, Karlof, Chris and Culler, David.** A Practical Evaluation of Radio Signal Strength for Ranging-Based Localization. *ACM SIGMOBILE Mobile Computing and Communications Review.* January, 2007, Vol. 11, 1.

14. *A Review of RFID Localization: Applications and Techniques.* **Sanpechuda, T and Kovavisaruch, L.** 2008. Proceedings of ECTI-CON 2008. pp. 769-772.

15. LORAN-C User Handbook. *The Navigation Center of Excellence.* [Online] U.S. Coast Guard. [Cited: May 31, 210.] <http://www.navcen.uscg.gov/loran/handbook/h-book.htm>.

16. *A Directionality Based Location Discovery Scheme for Wireless Sensor Networks.* **Nasipuri, Asis and Li, Kai.** Atlanta, Georgia, USA : s.n., 2002. Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications.

17. **Nelson, Russell G.** *Automated lawn mower. 5,974,347* US, October 26, 1999.

18. **Nelson, Russell.** Determining Mobile-Equipment Location with RF Transmitters. *Industrial Control Design Line.* [Online] June 19, 2001. [Cited: May 17, 2010.] <http://www.industrialcontroldesignline.com/showArticle.jhtml?articleID=192200494>.

19. *Generalized Geometric Triangulation Algorithm for Mobile Robot Absolute Self-Localization.* **Esteves, Joao Sena, Carvalho, Adriano and Couto, Carlos.** Rio de Janeiro : s.n., 2003. IEEE International Symposium on Industrial Electronics (ISIE).

20. *Mobile Robot Localization Using Landmarks.* **Betke, Margrit and Gurvits, Leonid.** 2, 1997, IEEE Transactions on Robotics and Automation, Vol. 13, pp. 251-263.

21. Windows Management Instrumentation. *MSDN.* [Online] Microsoft. [Cited: May 20, 2010.] <http://msdn.microsoft.com/en-us/library/aa394582%28VS.85%29.aspx>.

22. Native Wifi. *MSDN.* [Online] Microsoft. [Cited: May 20, 2010.] <http://msdn.microsoft.com/en-us/library/ms706556%28v=VS.85%29.aspx>.

23. WiFly GSX User Manual. [Online] June 15, 2009. [Cited: February 20, 2010.] <http://www.rovingnetworks.com/documents/WiFlyGSX-um.pdf>.

24. *Telos: Enabling Ultra-Low Power Wireless Research.* **Polastre, Joseph, Szewczyk, Robert and Culler, David.** 2005. Fourth International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors.

25. **Berman, Greg, et al.** *A Low-Cost Directional Antenna for IEEE 802.11.* Ann Arbor : University of Michigan.

26. *Multisensor Fusion and Navigation for Robot Mower.* **Cong, Ming and Fang, Bo.** Sanya, China : s.n., 2007. International Conference on Robotics and Biomimetics.

27. **Silver, Samuel, [ed.].** *Microwave Antenna Theory and Design.* London : Peter Peregrinus Ltd, 1984.

28. **Smith, Jeremy.** Antenna Tutorial. *Aerocomm.* [Online] [Cited: May 17, 2010.] <http://www.aerocomm.com/docs/Antenna_Tutorial.pdf>.

29. Building a Wi-Fi Antenna Out of a Tin Can. *ExtremeTech.* [Online] August 31, 2004. [Cited: May 17, 2010.] <http://www.extremetech.com/article2/0,2845,1641185,00.asp>.

30. The Seventh Annual Robotic Lawnmower Competition Rulebook. *The Institute of Navigation.* [Online] 2010. [Cited: May 17, 2010.] <http://www.ion.org/satdiv/alc/rules2010.pdf>.