Design of a Control System for Multiple Autonomous Ground Vehicles to Achieve a Self Deployable Security Perimeter

John Scott Clemmensen Jr

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
In
Electrical Engineering

July 11, 2007
Blacksburg, Virginia

Keywords: Formation Control, Zigbee, Range Based, Unmanned Vehicles

Abstract

Design of a Control System for Multiple Autonomous Ground Vehicles to Achieve a Self Deployable Security Perimeter

John Scott Clemmensen Jr.

Due to the limitations of GPS in areas where line of sight to the sky is obstructed the development of a GPS-free algorithm for relative formation control is an asset to collaborative vehicles. This paper presents a novel approach based on the Received Signal Strength Indication (RSSI) measurement between broadcast and receive nodes to calculate distance and using the data transfer capability to allow each vehicle to develop a table of relative positions. These relative positions are used to create a potential field that results in an absolute minimum at the vehicles desired position. All vehicles are numbered sequentially. The numbering defines the order in which they will broadcast their data, as well as their position along the perimeter. This thesis looks at two control methods for achieving a formation. The first is the circular motion method that puts perimeter nodes in an orbit around around the perimeter center. The second is a gradient descent method that calculates the gradient of the potential field. Both methods achieve a formation when all perimeter nodes are at their absolute minimums in the potential field. Tests were conducted to analyze RSSI measurements using the 802.15.4 protocol, and a mathematical simulation was conducted for each control algorithm.

Acknowledgments

This thesis, as well as my graduate career, would not have been possible without the following individuals. First, Id like to thank my family Scott and Joanne Clemmensen for their interest and pride in my work and success throughout my undergraduate and graduate careers. Without their love, support and constant nagging, I could never have made it as far as I am today.

I would like to express my gratitude to Dr. Alfred Wick and Dr. Charles Reinholtz for taking me on as a graduate student and for their advice and encouragement. I would like to thank Dr. William Baumann for my interest in controls, his enthusiasm encouraged my pursuit in the field.

I also need to thank Ben Hastings for encouraging me to purse a graduate degree, which is one of the best decisions I have made, and for encouraging me through out. I would like to thank those I have shared a lab with for the past two years, Mike Avitabile, Andrew Culhane and Adam Sharkasi, who were always there to lend a helping hand and made the time in lab enjoyable. I would epically like to thank John Bird who I shared a cubical with during my studies, and was one of my most valuable resources.

And, thanks to all of my fellow graduate students who have helped create a productive and enjoyable working environment at Virginia Tech.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 2001, the Senate directed the armed forces to increase the number of unmanned systems in its air and ground combat vehicles to one-third over the next decade [1]. This has increased the development in the area of unmanned and autonomous systems. Advances have been made in unmanned aerial, ground, underwater and surface vehicles for military applications. As a result development of control systems for these vehicles has become a growing field of research.

One area in particular is multi-vehicle formation control. There has been an increase in research in control theory and sensor integration that allows for multiple simple vehicles to interact with each other in what looks like intelligent behavior; also known as emergent behavior. A use for this behavior is to create a formation. A formation is a specific alignment of vehicles.

The formation control system described in this thesis is a solution to the problem of aligning multiple vehicles in formation with limited sensor capabilities. The package described integrates a sensor and control in a small device that can be quickly implemented on any unmanned vehicle running Labview. Implementation on multiple vehicles enables vehicle cooperation in achieving a formation.

1.1 JOUSTER

The Joint Unmanned Systems Test, Experimentation, and Research (JOUSTER) facility was created in 2004 to support the scientific testing and evaluation of unmanned vehicle systems through experimentation [2]. One of the goals of JOUSTER was to facilitate the development of unmanned platforms and the control and sensors necessary to run them.

This design work was part of a project aimed toward the design and development of a team of lab vehicles that were small and easy to reproduce. These vehicles would be used for testing collaborative control of multi-vehicle systems.

1.2 Motivation

A constant need throughout history for any combat force is perimeter security. It is of such high importance that the first item executed by a unit as part of its Standard Operating Procedure (SOP) is securing a perimeter. This allows the unit to defend itself and execute its main objective. This is currently accomplished by a listening post/observation post or blocking position all manned by soldiers [3]. To help reduce the number of soldiers necessary to man a perimeter Intrusion Detection Devices (IDD) have been developed [5]. IDD's can range from as simple as a wire with noise makers, or flares, to the more complicated Mobile Remote Sensing System developed by Millennium Sensor[4]. These sensors integrate several different sensors to detect an intruder. These systems are effective for increasing security, and decreasing the number of soldiers needed to stand watch. However they still require that a soldier set them up. With a current emphasis on the armed forces on rapid deployment, this is a time consuming process that puts soldiers on the outskirts of a secure perimeter and increases their chances of getting injured.

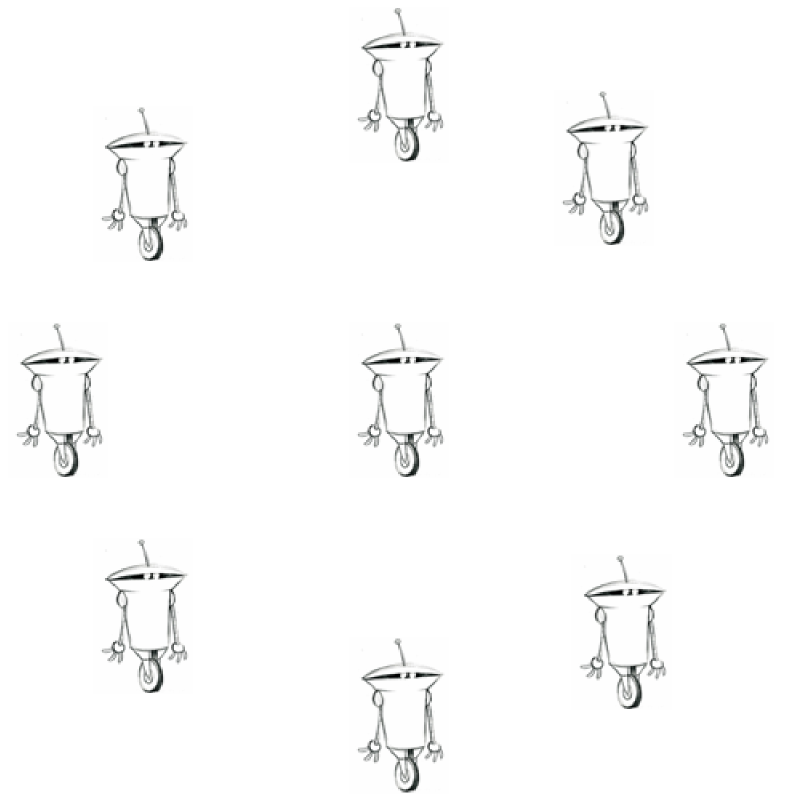


Figure 1.1: A sketch of the desired formation

A self deployable security perimeter would eliminate these potential hazards. With a vehicle small enough for a soldier to carry, which could be deployed without an operator, the solider is then free to perform other tasks necessary for mission success. The problem solved by this system is organizing the nodes into a perimeter using only the distance between nodes, without orientation of the vehicle or to other nodes. The solution currently used to align a team of nodes into a formation is a Global Positioning System (GPS), which uses satellites to locate each node in a global coordinate system, with these known positions a formation coordinator can compare each nodes desired location with its current location and give it the optimal vector to its final location. However, there has been a push to find alternative methods for accomplishing the same task as seen in papers discussed in the following chapter. However these all make assumptions on knowledge about location or orientation, or communication abilities that do not exist. Also none have been tested using a simulation that only uses the knowledge that is actually available. The achievement of this paper is the testing of a system that will use only the distance, as measured using Received Signal Strength Indication (RSSI) data, between individual nodes to create a field that can be navigated by the nodes to direct them into a formation.

1.3 Thesis Overview

The goal of this thesis is a simple and inexpensive solution creating a module that can direct autonomous vehicles into a formation. The first step toward accomplishing this was developing the hardware to handle communications (wired and wireless) and power. This required selection of wireless and wired communication modules, as well as voltage regulators, battery protection and charging, and the battery itself. The power circuits were chosen for size and reliability. The wired communications are designed for RS232 serial to interface with a vehicle controller. A converter was also developed so the modules are USB capable as well. The chosen on board wireless communication device is the Maxstream XBee Pro module which is capable of measuring the strength of the received radio signal and returning a Received Signal Strength Indication (RSSI).

To coordinate the flow of messages each node is designated a number in the system between 1 and the total number of perimeter nodes. It is the job of the user to make sure no number is assigned twice. The number has two purposes. The first is the order in which it will transmit in relation to the other nodes, and second is its relative position to the other nodes in the formation. The formation is a circle of evenly spaced vehicles with the vehicle numbers increasing in the counter clockwise direction. Only the Beacon Node (BN), described in section 3.2, needs to know the total number of nodes in the system a priori.

Next, methods of control were investigated that could deal with the restrictions on sensor information. The control methods had to be able to direct a perimeter node to its assigned position without knowing a heading to the position or knowing where it or the other nodes are positioned in a global or local coordinate system. Two forms of vehicle control met these criteria and are compared. The first is the circular motion method. This method puts the perimeter nodes in an orbit around BN of the desired perimeter radius. By calculating a change in the distance from BN between two measurements and by measuring the distance traveled a correction angle for the heading is determined. The second method is recognized as an optimal control method called Gradient Descent. This method creates a field based on the information from all the nodes and through a method of sampling, steers the node down the greatest slope in the field, leading to minimum that should be the vehicles designated position in the formation.

Testing was then conducted. The first test was performed on the relationship between RSSI and distance between nodes. Next the control methods went through various simulations that measured settling time, and convergence.

Chapter 2

Literature Review

A review of existing research was conducted to find out what has already been accomplished and where the most significant contributions could be made. This literature review also examined what was necessary to complete the project. This was done by first looking at publications describing the measurements that could be expected from the sensor and filters being used, then looking at publications involving control theory and implementation.

2.1 Measurements and Filter

The low-power of Zigbee wireless chips benefit power consumption, size and cost, with no additional hardware [31], and allow the use of the RSSI feature as a sensor. While the use of RSSI as a way to estimate distance between nodes has been a possibility for a while, it has not been until recently that the use of small-wireless devices featuring low-power radios has attracted a lot of attention. A study performed last year by Yale [12] tested Zigbee

(802.14.5) radios, to get a better understanding of how RSSI can be used for localization. They learned as others have that the effects of noise must be taken into account [15].

To compensate for the noise in the RSSI measurements a filter will be needed to smooth the data. A weighted windowing filter was investigated, and is used to smooth out the measurements [16] [17]. Even with a filter on the measurements being taken a precise formation will not be achievable. However, these sensors will enable the nodes to get into a desired configuration. This should not present a problem as long as sensor coverage of each node is able to overlap the coverage of the surrounding nodes.

2.2 Control

To implement any control system it is necessary to know how a decentralized team works together and shares information [10] [24]. A decentralized strategy requires that each node has its own ability to collect data, and communicate with the other nodes directly or indirectly.

Formation control has broad applications, including security patrol and search and rescue in hazardous environments. To achieve a formation from a group of coordinated robots, different control topologies can be adopted depending on the specific scenarios [6]. One control technique for getting multiple vehicles into formation includes creating a local coordinate system[26] [27] [28]. Creating a local coordinate system is an effective method for calculating vectors between nodes and control inputs to get nodes to their position in the formation. This technique is possible with the limited sensor data that is presented as part of the current design constraints. However, mirroring is a problem which limits its effectiveness in this constrained system.

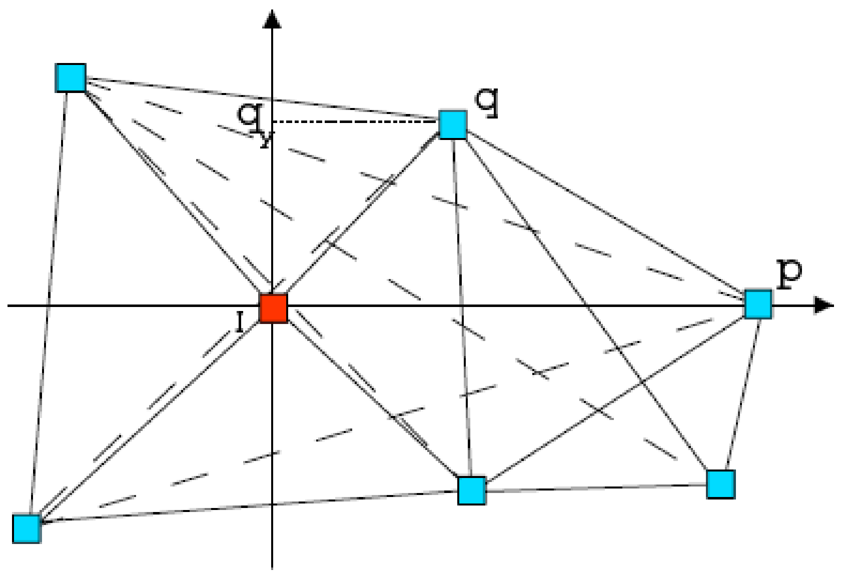


Figure 2.1: An Example of All The Measurements Necessary for a Local Coordinate System [26]

Mirroring is a result of the lack of orientation in the global coordinate system. As a result the local coordinate system can be rotated about any axis and all the measurements would still be the same. This is demonstrated in figure 2.1. Where the coordinate system shown is an a guess, the measurements would still work in any orientation rotated around node I. The mirroring would be possible if a orientation was established to one node such as node P, the coordinate could be flipped over the I-P axis. This means while each node knows the angle between all the nodes surrounding it, it is incapable of establishing its orientation in the system, and therefore unable to make the proper moves to align itself. This can be overcome with a small system but the message size increases exponentially with the number of nodes because each node needs to know the distance from itself to all surrounding nodes and the distances between all surrounding nodes. This also means corrections become computationally expensive as the system grows. Because of this the articles use information stored priori to give initial orientation, or more extensive sensors are used. The sensors used range from adding cameras to locate nodes, to scanning laser range finders, to external controllers that monitor the positions of the nodes [23] [21] [22] [11] [25].

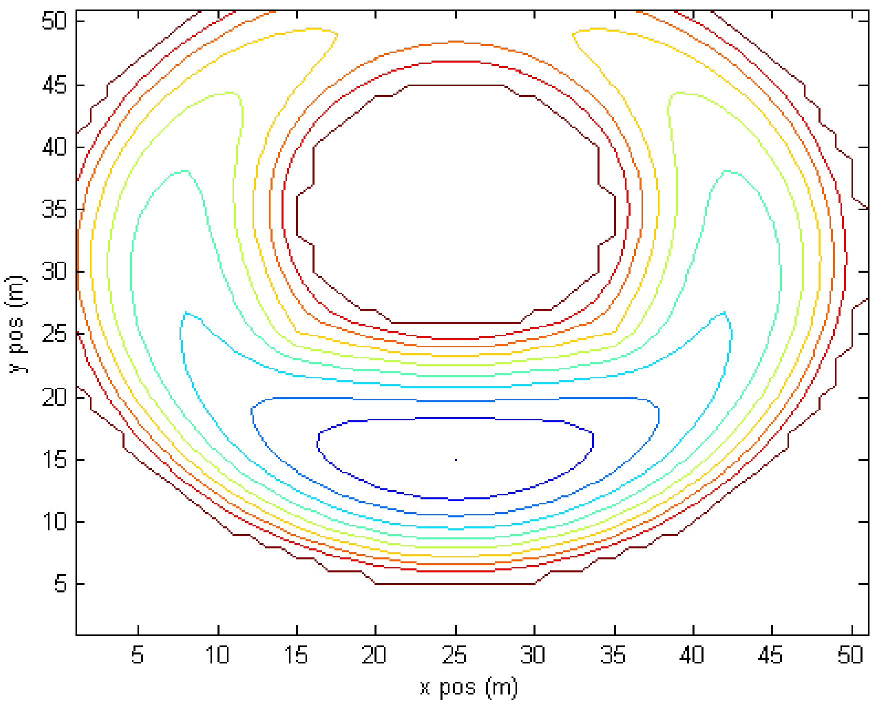


Figure 2.2: The Potential Field For A Node With A Position In The Formation At [15,25]

An alternative to this method is to create potential fields [9]. This method uses the sum of all the direct measurements to each node to calculate a nodes current magnitude in the field. The measured value of the field is a function of the node's position relative to the other nodes in the system. The field has a minimum at the nodes position in the formation, and the field value increases the further away from that position the node travels, as seen in figure 2.2.

The potential field method works with the limited sensor capabilities that are being implemented. One method for navigating the fields is an angular motion technique [7]. This method only requires distance to a central node to maintain a direction with the ability to handle the RSSI measurement noise, and uses the field measured for controlling speed. Another method for navigating the potential field is the method of gradient descent [13] [14]. For this thesis the two potential field methods were investigated further because they can work with the sensor limitations of this system.

Chapter 3

Formation Control Design

This chapter describes the need for obtaining a position in a formation from a distance measurement without direction. It also describes the classifications of the nodes in the system based on measurement constraints, as well as a description of the vehicle the system was being designed for. This chapter will also look at the expected values of an ideal RSSI measurement and the noise that can be expected. Then it describes the Circular Motion and Gradient Descent control methods that will be tested in this thesis.

3.1 Formation Based on Relative Position

To properly align nodes relative to each other, each node must know its proper alignment relative to each other node. The purpose of this section is to get each node to this alignment. While this is taking place there are several things to consider, the first is that the number of nodes is scalable, the second is that a perimeter circle is being created. But before these can be addressed there is another problem that needs to be addressed. Using a distance from a single location to establish a desired position does not result in a unique solution. The result of this is a circle around the first node with a radius of the desired location. By adding a second node that resides at one of the infinite possible locations along the perimeter of the first node, it is then possible to limit the possible positions for the third node down to just two, in a 2 dimensional space. Once the third nodes picks one of the two possible positions designated for it, the system will now have three set nodes. With three nodes in position in a 2 dimensional space, that are not in a line, all following nodes will have a unique position in their alignment to all the other nodes. The figure below demonstrates the need for three nodes before a unique position can be established.

To establish this system each node has to know its job, and what information it has to work with. Without this classification system the nodes would be unstable. Because every node would be dependent on all the other nodes to settle before it could settle. Since there is an infinite number of possible orientations for the formation around the center of the formation, no nodes would ever settle into a formation. They would instead be in a constant state of motion.

3.2 Node Classification

Defining the nodes and giving them different criteria for achieving their position in the formation will ensure that all nodes converge to an alignment with one another.

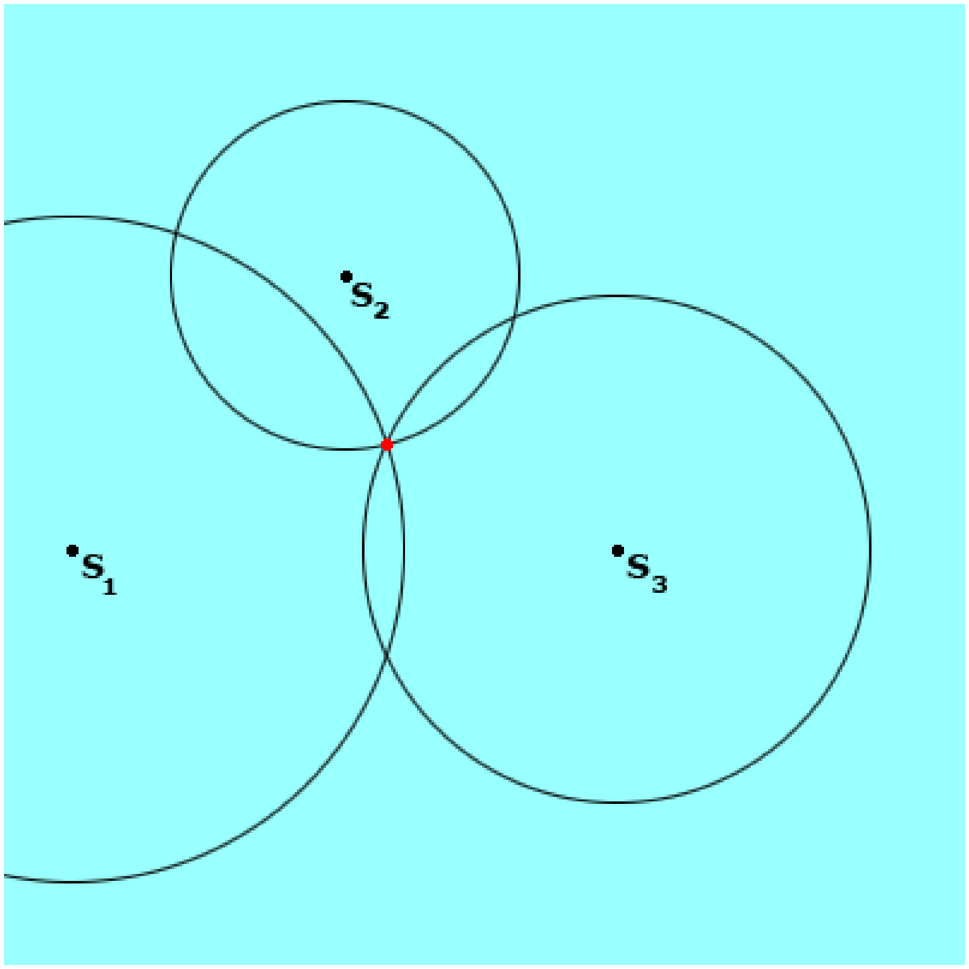


Figure 3.1: Three Intersecting Circles Are Needed To Locate A Point

3.2.1 Beaconing Node (BN)

This is the central node of the formation. It will be the node that is tasked with knowing the total number of nodes, and it restarts the communications cycle when the last node has transmitted.

3.2.2 Perimeter Nodes

Under Constrained Node 1 (PN1)

This node would make up the second node in the system, and the first of the distance based control nodes. This node's position is based on its distance from the beaconing node. There are infinite possible locations for this node creating a circle, assuming working only in a plane and not three dimensional space, around BN with a radius of the desired distance. Its goal is to get to the radius distance and stop.

Under Constrained Node 2 (PN2)

This node is the third node in the system, and the second of the distance based control nodes. This nodes position is based on its distance from BN and PN1. There are two minimums in the potential field that the node might find, the one it goes to is dependent on its starting position. The two minimums in the field are a result of the two intersection points of the circles drawn around BN and PN1 of the desired distance from each.

Constrained Nodes (PN $\sharp$)

These nodes will make up the rest of the system, and their locations in relation to the rest of the system can specified. With three other nodes already in the system, drawing circles of radius equal to desired distance from each will result in only one intersection point.

3.3 Vehicle Modeling

One of the simplest forms of modeling a vehicle is the planar unicycle model. This model acts like a point mass. The equations for change in position listed below.

$$\begin{aligned} \dot{x} &= v \cos\theta(t) && (3.1) \\ \dot{y} &= v \sin\theta(t) && (3.2) \\ \dot{\theta} &= u(t) && (3.3) \end{aligned}$$

Where

$$[x \quad y \quad \theta] \in \Re^2 \quad \times \quad [-\pi, \pi) \tag{3.4}$$

Represents the nodes position in real space, v is the forward velocity, and $u(t)$ is the angular velocity. Both the forward velocity, v, and the angular speed $u(t)$ are control inputs.

3.4 RSSI Modeling

There are several sources of error when obtaining a RSSI. They include multipath, transmitter variability, receiver variability, antenna orientation [12].

The ideal equations for calculating RSSI value as a function of distance between two nodes are[29]:

$$Loss(d) = 10 \cdot log_{10}\left(\left(\frac{4\pi}{c}df\right)^2\right) \tag{3.5}$$

$$Loss(d) = 20 \cdot log_{10}(d) + 20 \cdot log_{10}(f) + 20 \cdot log_{10}\left(\frac{4\pi}{c}\right) \tag{3.6}$$

$$Loss(d) = 20 \cdot log_{10}(d) + 20 \cdot log_{10}(f) - 147.55 \tag{3.7}$$

$$RSSI(d) = TS - Loss(d) \tag{3.8}$$

Where d is the distance between nodes in meters, and f is the frequency of the signal. For these tests f was equal to 2.4GHz because that is the frequency of the 802.15.4 (Zigbee) modules, and c is the speed of light.

3.4.1 Sources of RSSI Error

When modeling RSSI response as a function of distance, there are several sources of error that can be expected.

Multipath

Multipath occurs when a signal reaches a receiver by more then one path. In an ideal situation the receiver will get the message once, directly from the transmitter, through a line of sight path. However, what often occurs is the signal will bounce off objects that will redirect a non-line of sight signal to the receiver, thus the receiver will get the message more then once, resulting

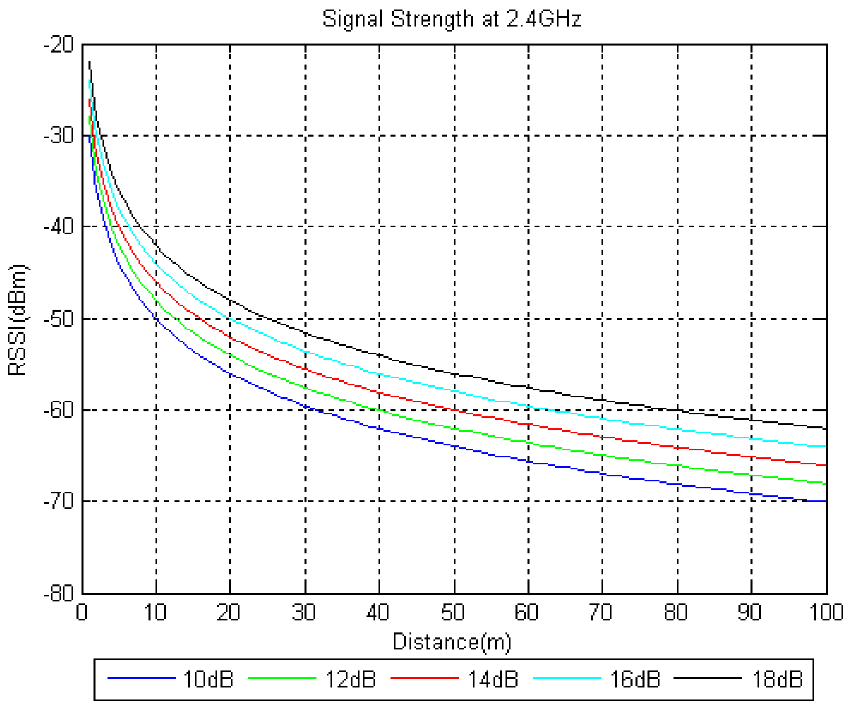


Figure 3.2: Ideal RSSI Values

in error of the RSSI.

Transmitter variability

Since not all transmitters are made exactly the same and perform exactly the same, there will be variability in the power level of transmissions from each node, given the same configurations on transmission dBm level. This will have an effect on the signal strength at the receiver thus altering the received signal strength indication, and error in measured values.

Receiver variability

This is a result of the sensitivity of the receiver will vary with each chip. This means that even if all other parameters are the same receivers could return different received signal strength indication.

Antenna Orientation

Because each antenna is made from a wire, there are several factors that could affect it. Each antenna will have its own radiation pattern that is not uniform, thus over a constant distance, the RSSI could fluctuate depending on what side of the antenna the other nodes are on.

Testing showed that looking at transmitter/receiver variability, and antenna orientation, resulted in a standard deviation of 0.4ft on measurements [12]. Multipath will depend heavily on the environment in which the nodes are being used.

3.5 Control Methods

The following control methods are closed loop control. Closed loop control is a basic form of control with feedback. This type of guidance directs the system based on what it thinks is going on with sensor feedback to confirm these assumptions were correct, and allowing for corrections based on the sensor measurements.

3.5.1 Circular Movement

The first step to achieving a formation between multiple vehicles is to create motion for a vehicle to align itself with a single vehicle. Due to the sensor limitations discussed earlier, this is an under constrained control problem without an unique solution. There would be an infinite number of solutions to this problem that would create a circle around the central node of a radius equal to the desired spacing. This section will describe a control algorithm that will allow a vehicle to align in any of these positions.

The best way to enable a vehicle to achieve alignment with possible locations forming a circle is for the vehicles movements to be circular. Using the circular motion control method this can be achieved by measuring the distance between the vehicle and the Beacon Node(BN). The vehicle can calculate the difference between its current heading and a heading to the BN by comparing the distance traveled between two measurements with those measurements, this difference angle is called $\gamma \in [0, 2\pi)$ and is found by

$$\gamma = \pi - \cos^{-1}\left(\frac{\rho_(t-1)^2 + d_t^2 - \rho_t^2}{2 \cdot d_f \cdot d_t}\right) \tag{3.9}$$

Where ρ is the distance measurement between the vehicle and BN, and d_t is the distance traveled. However, this angle is without direction so the vehicle does not know if the BN is γ radians to the clockwise or counter-clockwise direction. It is assumed that the vehicle is traveling in a straight line between measurements.

Assuming a constant velocity, a control input to steer the vehicle into an orbit of a desired distance around the BN would be [7]

$$u(t) = \begin{cases} k \cdot g(\rho(t)) \cdot \alpha_{dist}(\gamma(t)) & \rho(t) > 0 \\ 0 & \rho(t) = 0 \end{cases} \tag{3.10}$$

When

$$g(\rho) = ln\left(\frac{(c-a)\cdot\rho + \rho_o}{c \cdot \rho_o}\right) \tag{3.11}$$

Where $g(\rho)$ steers the vehicle toward BN if $\rho > \rho_o$, where ρ_o is the desired distance between the control node and the reference node, and away from the reference node if $\rho < \rho_o$, and $k > 0, c > 0\rho_o > 0$ and $\psi \in (\frac{3}{2}\pi, 2\pi)$ are all given constants.

To ensure a direction of orbit a bias is set using $\alpha_{dist}(\gamma)$ where ψ sets the direction.

$$\alpha_{dist}(\gamma) = \begin{cases} \gamma & if\ 0 \le \gamma \le \psi \\ \gamma - 2\pi & if\ \psi < \gamma < 2\pi \end{cases} \tag{3.12}$$

$$r = r_b - r_v = \rho e^{i\Gamma} \tag{3.13}$$

$$\gamma = (\Gamma - \Theta) mod(2\pi) \tag{3.14}$$

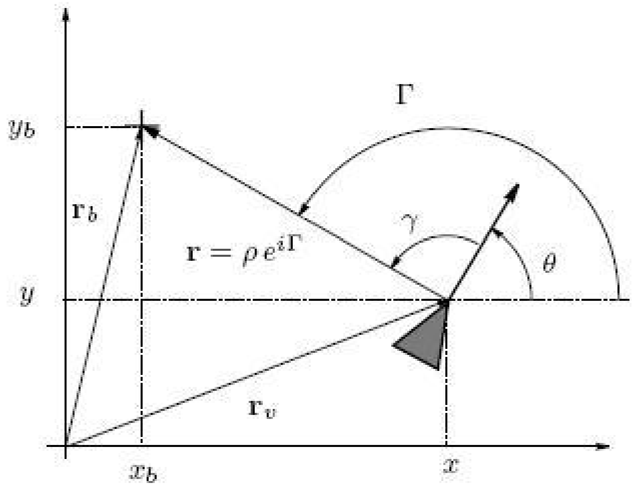


Figure 3.3: Single Node Relation to Reference Node [7]

for simulation *theta* is a random orientation for the initial vehicle heading, and ρ is

$$\rho = \sqrt{(x - x_r)^2 + (y - y_r)^2} \tag{3.15}$$

where (x_r, y_r) is the reference node position, and (x, y) is the control vehicle position

which according to [7] ends in an equation for calculating the change in gamma, $\dot{\gamma}$

$$\dot{\gamma} = \dot{\Gamma} - \dot{\Theta} \begin{cases} \frac{v}{\rho}\sin\gamma - kg(\rho) & if\ 0 \leq \gamma \leq \psi \\ \frac{v}{\rho}\sin\gamma - kg(\rho)(\gamma - 2\pi) & if\ \psi < \gamma < 2\pi \end{cases} \tag{3.16}$$

These equations give a frame work for vehicle motion around a central node, however, they are not adequate to solve this problem. The accuracy of $\rho(t)$ to ρ_o is dependent on k. The higher the value of k the quicker the control vehicle's ρ value approaches ρ_o and settles more closely to ρ_o as well. But under certain conditions a k value that may work for a majority of test positions and orientations will result in a noncovergent solution. This is usually the result of the control node getting too close to the reference node, and the large control input changes the node direction so drastically that it can not orient itself to get away from it. However given the same initial conditions and a smaller k, this problem will not occur, the vehicle can pass near the reference node without large corrections that would disorient it. To achieve accuracy in orientation as well as not getting disoriented while too close to the reference node, the control gain, k, is set to a function of ρ, this also benefits when $\rho >> \rho_o$ and brings the control vehicle to a more direct trajectory.

$$k = \begin{cases} 0.1; & if\ \rho \leq \rho_o \\ .25 & if\ \rho \geq \rho_o \end{cases} \tag{3.17}$$

3.5.2 Gradient Descent

Gradient descent is an algorithm to find a local minimum of a potential field. Gradient descent is accomplished by taking steps relative to the negative of the locally calculated gradient. A gradient is from vector calculus and points in the direction of the greatest rate of increase of a vector field [13].

$$b = a + \gamma \Delta F(a) \tag{3.18}$$

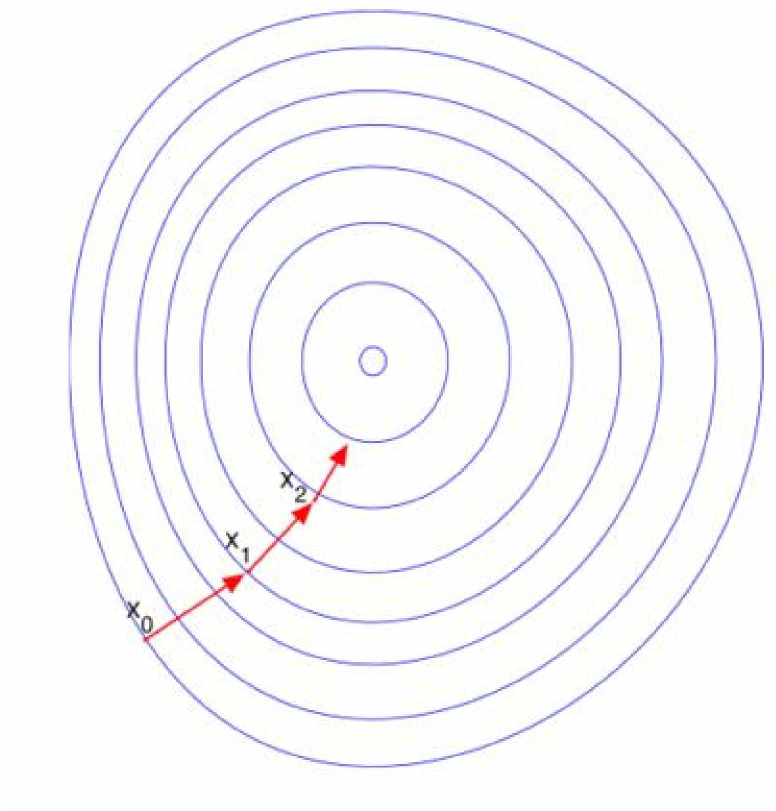


Figure 3.4: Gradient of field leading to a local minimum

$$x_{n+1} = x_n + \gamma_n \Delta F(x_n), \quad n \geq 0 \tag{3.19}$$

$$F(x_n) \geq F(x_{n+1}) \geq F(x_{n+2}) \tag{3.20}$$

There are weaknesses to this approach, as there are to any, that make convergence to zero time consuming, as well as finding the optimal path by calculating an exact gradient. To counter these weaknesses the vehicle will not have to reach the absolute minimum of the potential fields but get within a desired tolerance of it. As for calculating the exact gradient, this would be almost impossible because the gradient of the vehicles location is never constant. Therefore the vehicle will calculate as best it can an approximation of the gradient based on three different field samplings creating a 90^o angle with respect to its previous heading and calculate a gradient. Then move along that path until the measure field values increase, indicating that the node is no longer traveling toward a minimum.

3.5.3 The Gradient

The necessary information for gradient descent control is the gradient that the node lies on. This is unobtainable through a single measurement or calculation because the node does not know where in the field it lies. Each node has to measure the gradient that it lies on through a series of measurements. First the node measures the field where it currently is, it then proceeds forward a distance, x, and takes another measurement. Next it makes a 90 deg turn, for these calculations it turns left, but a right turn would work just the same. After the turn the node proceeds forward a distance, x, and takes another measurement. A gradient is then calculated as a vector field whose components are the partial derivatives of the field. However, since each node is making relative measurements in relation to its position, a temporary local coordinate system is established where the first and second measurements are along the x-axis, and the second and third measurements are along the y-axis, since a 90 deg turn occurs. The resulting equation is

$$\nabla f = \left(\frac{f_{n+1} - f_n}{x}, \frac{f_{n+2} - f_{n+1}}{x} \right) \tag{3.21}$$

Where x, is the distance traveled between measurements, and f_n represents the measurements taken. Because the goal is to descend the gradient the direction of travel is the opposite of the gradient.

$$(\lambda_x, \lambda_y) = -\nabla F = (\Delta F_x, \Delta F_y) \tag{3.22}$$

The λ values are the weights for the nodes movements, so that it proceeds down the gradient to the absolute minimum.

3.5.4 Creating an Artificial Potential Field

For the Gradient Descent method to work, equations are needed to define the potential field. The most common way this is done is by using a linear relation to the distance values calculated. But, because of the variability of the number of inputs, and the dynamic nature of the positions of the inputs to each nodes potential field, it was necessary to develop a more robust generic set of equations. Another issue that was taken into consideration was the elimination of local minimums. The Gradient Descent control method searches out minimums, if the minimum it finds is a local minimum and not the absolute minimum it will get stuck and never reach its position in the formation. To prevent this the equations were designed so that no local minimums would exist. Another method for fixing this is injecting a random vector field that will dislodge a node from a local minimum. But this is not guaranteed, and the size of the pocket that would be escapable by this method would be limited to the size of the random field. Since the node does not know if it is trapped in a local minimum, this method would also cause error during normal navigation, resulting in a suboptimal path to the absolute minimum.

The field measured by a node at any point is the sum of the field from the beacon node and all other perimeter nodes that are less constrained then itself (have a lower assigned number in the order of communication).

$$F_n(t) = F_{BN}(t) + \sum_{i=1}^{n-1} F_i(t), \quad if n > 1 \tag{3.23}$$

Where F_{BN} is the force from the beacon node at a give time, n is the measuring node's assigned number in the order, and F_i is the force from the node with the assigned number i.

The field from the beacon node is

$$F_{BN}(t) = g_{BN} \cdot |d_{nBN}(t) - d_{BNo}|^3 \tag{3.24}$$

Where g_{BN} is the beacon nodes assigned gain, d_{BNo} is a predefined value that represents the radius of the security perimeter, and d_{nBN} is the current measured distance

$$g_{BN} = 1 \tag{3.25}$$

The fields from perimeter nodes can be defined generally. Each field value is a function of a gain multiplied by the square of the difference between the current measured distance and the desired distance between the two nodes.

$$F_i(t) = g_i \cdot (d_{ni}(t) - d_{nio})^2 \tag{3.26}$$

Where g_i is the gain for the node being evaluated, d_{ni} is the current measured value between the two nodes, and d_{nio} is the desired distance.

The gain for a perimeter node is defined as

$$g_i = \frac{Nodes - i}{\frac{d_{BNo}}{d_{nio}}} \tag{3.27}$$

Where 'Nodes' is the total number of perimeter nodes.

The desired distance between the current node and node being measured is shown here.

$$d_{nio} = \sqrt{2 \cdot d_{BNo}^2 - 2 \cdot d_{BNo}^2 \cdot \cos\left(\frac{\pi(n-1)}{Nodes}\right)} \tag{3.28}$$

This equation takes advantage of the law of cosines to create an equation that is scalable to the number of nodes in the system as a function of a node's position in the communication order, and the desired radius of the perimeter.

Chapter 4

Hardware Design

The custom hardware is one of the major benefits of this system because it consists of a single board that has its own power supply, can handle wired and wireless communications, and measure range to other nodes. All the hardware is on a board that is 8cm by 5cm, and stands 1cm tall. The assembled hardware is pictured in figure 4.1.

4.1 Communications Hardware

The communications, schematic shown in figure 4.2, are adaptable to the most common communication protocols including RS232 and USB. This ensures that the system can communicate with any vehicle controller. Chapter 5, describes the software that handles the communications.

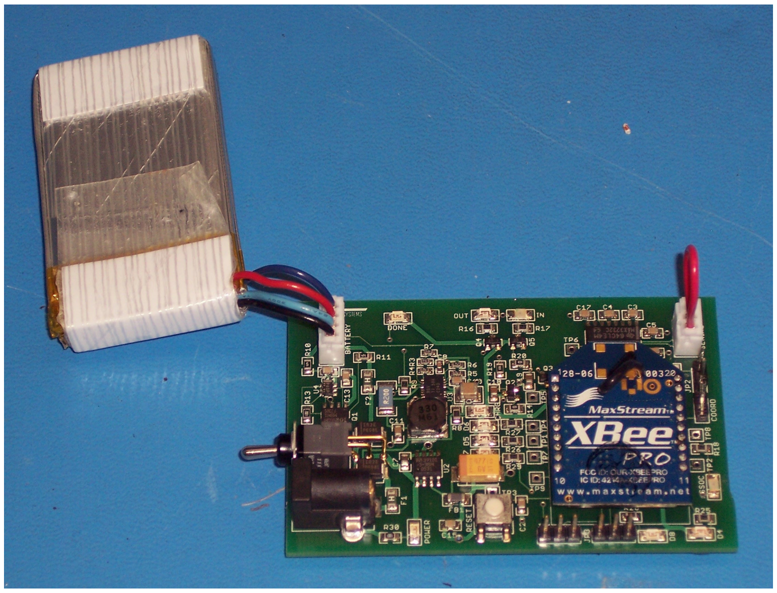


Figure 4.1: Assembled Board with Battery

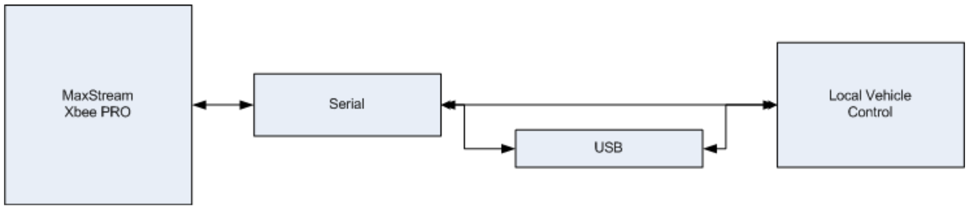


Figure 4.2: Communications Layout

4.1.1 Wireless Communications

The Maxstream ZigBee Pro was chosen because it enables communications with all other nodes at once, and can calculate RSSI, which is being used to calculate range between nodes. It has a 3cm, single wire, monopole antenna. Operates in the 2.4 GHz ISM band at an effective data rate of 250Kbps. Was designed for low power applications and uses AT commands. The module is compact measuring 3.2 x 2.5cm. It supports multiple transmission power levels: 10dBm to 18dBm at which the consumed power varies from 137mA to 215mA @ 3.3V [31].

4.1.2 Serial Communications

The system is setup to handle data transfer over a standard RS-232 serial interface. Since the XBee Pro communicates using UART and not serial, a transceiver was needed. The Maxim MAX3232 was chosen, it is a low power true RS-232 transceiver [34]. It translates the UART communications coming from the XBee Pro to RS-232 without the need of drivers by the vehicle controller.

4.1.3 Universal Serial Bus

The USB communications standard was also made available. If the controller does not have serial communications or all the serial ports are taken, then USB can be used. This conversion was accomplished using the FTDI USB UART Chip [35]. USB does require a FTDI USB driver be installed on the controller and is usually only available on higher level processors. USB also offers the advantage of being an alternate power source for the system on later board revisions.

4.2 Power Supply

The board was designed to handle power from two different sources. The first being power from the vehicle it is supporting if external power is not available it is powered by a 7.4V 850mAh Lithium Polymer battery.

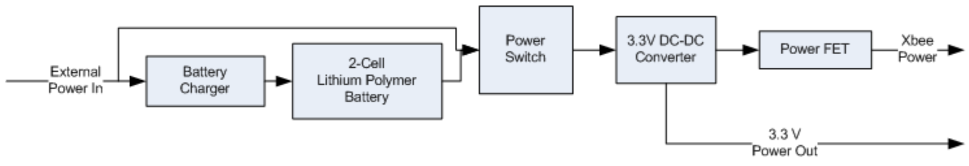


Figure 4.3: 3.3 Volt Power Layout

4.2.1 3.3 Volt

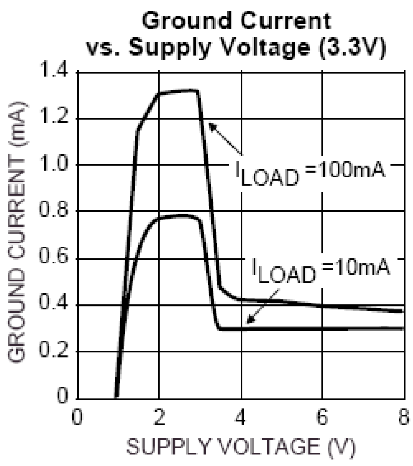
The supply voltage is regulated by Micrel 3.3V 1A low-dropout linear voltage regulators. The regulator operates at a set voltage, has a small foot print and requires just two capacitors.

The power dissipated by the linear regulator is

$$P_D = (V_{IN} - V_{OUT})I_{OUT} + V_{IN} \cdot I_{GND} \tag{4.1}$$

Where V_{IN} is the voltage of the battery, and V_{OUT} is the output voltage of the regulator. I_{out} is calculated by summing the currents in Table 4.1 these variables are known except I_{GND} which is found by comparing I_{OUT}

and V_{IN} on Figure 4.4.



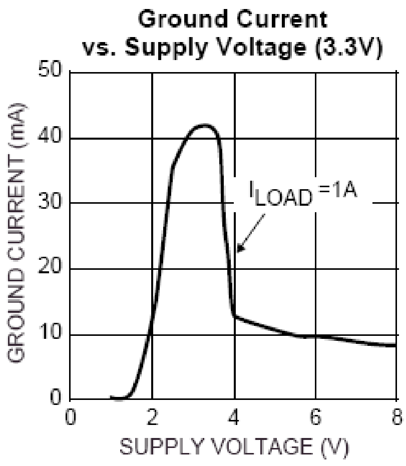


Figure 4.4: Ground Current of Linear Regulator at 3.3 Volts [32]

Therefore, the power dissipated by the linear regulator to give the board a 3.3V supply from a 7.4V battery and an output current of 252.6mA is

$$P_D = (7.4V - 3.3V)252.6mA + 7.4V \cdot 8mA = 1.099W \qquad (4.2)$$

4.3 Battery

The Lithium Polymer(LiPo) type of battery chemistry was chosen because of its high energy density. It has the ability to hold a lot of charge in a small, lightweight package. The battery being used supplies 850mAh at 7.4V. However, unlike some other battery chemistries, a specific charging procedure must be followed to avoid damaging the cells [36]. 7.4 volts is more then the 3.3 volts required by the system, but it is the standard voltage of a 2 cell LiPo battery, and a single cell at 3.7 volts would have a short run time.

4.3.1 Run Time

A run time for the system can be calculated by comparing the battery with the demands of the board. With a Lipo's batteries characteristics it is capable of supplying 22.6kJ.

$$850mAh \cdot 7.4volts \cdot 3600\frac{s}{hr} = 22.6kJ \tag{4.3}$$

The energy consumed by the board is documented in 4.1, and totals to 1.93 watts.

Device	Current(A)	Volts(V)	Power(mW)
XBee Pro	215mA	3.3V	710mW
MAX3232	35mA	3.3V	115mW
4 LED's	2.6mA	3.3/7.4V	9.13mW
MIC39101	See Eqn. 4.2		1,100mW
Total Power	252.6mA		1,930mW

Table 4.1: Calculations of Power Used

A run time for the board is calculated in eqn. 4.4

$$35.2kJ/1.93watts \cdot \frac{1hr}{3600s} = 3.26Hoursruntime \tag{4.4}$$

The battery provides a run time of 3.26 hours. The LED's consume power that is small compared to the other components on the system, but because they are unnecessary could be removed to increase the run time.

4.3.2 Battery Charger

The charging circuitry allows the Lithium Polymer battery to be charged when using external power.

The BQ24123[33] is a single chip switch mode lithium polymer charge management IC with enhanced EMI performance. It provides high accuracy current, voltage regulation, charge preconditioning, charge status, and charge termination. This all prevents overcharging which would result in battery failure.

A sample lithium polymer charging curve is shown in figure 4.5. The circuit charges the battery in three stages, conditioning, constant current, and constant voltage. The chip operates in one of three states; auto shut off, auto restart if battery voltage drops below threshold, or sleep mode when no external supply is provided.

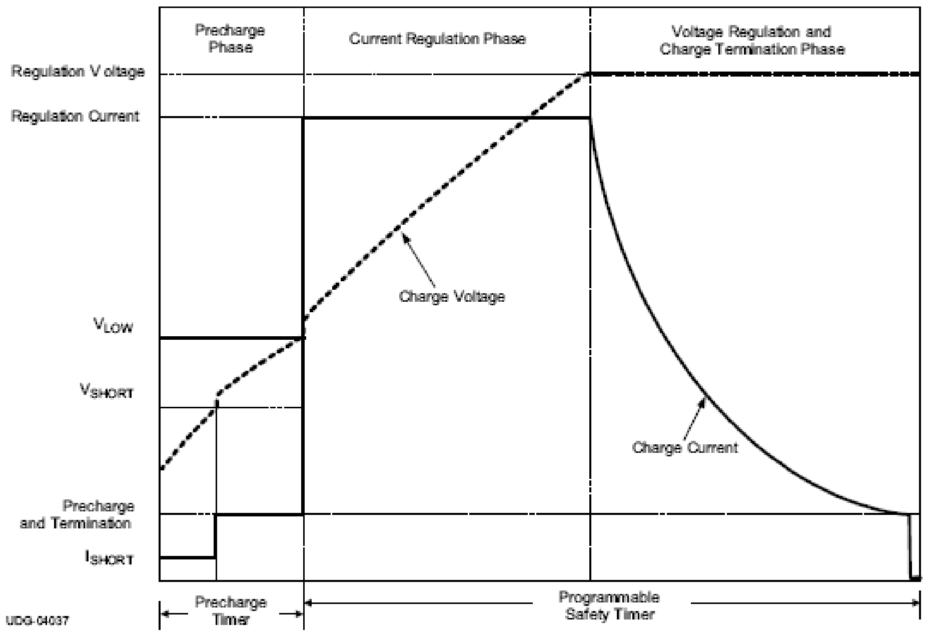


Figure 4.5: Charging Curve for Battery Circuit[33]

Chapter 5

Communications Software

The need for custom communications software was to create a simple way to pass node identification information and measured data from the current transmitting node to all receiving nodes. This section describes the settings for the Maxstream XBee Pro so that it will broadcast this information so that all available nodes will receive it. It will also overview the format that the messages are sent in, and how Labview features were used. Finally it will describe how each node uses the received information, uses the received message to make a measurement, and how the order for sending messages is coordinated.

5.1 XBEE Network Settings

For the Maxstream XBee Pro modules to communicate they must be set to interact properly. All the nodes have to be set to have the ability of transmitting to all the nodes in range at once, and being able to receive any message that is transmitted.

5.1.1 NonBeacon

By default, XBee/XBee-PRO RF Modules are configured to support Non-Beacon communications. NonBeacon systems operate within a Peer-to-Peer network topology and therefore are not dependent upon Master/Slave relationships. This means that modules remain synchronized and each module in the network shares both roles of master and slave. MaxStream's peer-to-peer architecture features fast synchronization times and fast cold start times.

If the default settings have been changed a peer-to-peer network can be established by configuring each module to operate as an End Device (CE = 0), allowing every node to receive transmissions. Then disabling End Device Association on all modules (A1 = 0) sets all nodes to be identical across the network. If all the nodes look identical to the transmitting node no pairing of nodes will takes place, which would limit the distribution of information [31].

5.1.2 Broadcast Mode

For the XBee modules to accept transmissions they receive they must operate in broadcast mode. When in broadcast mode receiving modules do not send acknowledgments(ACKs) and transmitting modules do not automatically resend packets as is the case in Unicast Mode. If these ACKs or resending of data were to occur it would cause a failure in the sequence of node transmissions. If a node does miss a transmission it will be able to get the information when the next node broadcasts. To send a broadcast packet to all modules regardless of addressing, set the destination addresses of all the modules as shown below[31]:

o DL (Destination Low Address) = 0x0000FFFF

o DH (Destination High Address) = 0x00000000 (default value)

5.2 Message Format

The necessary information to send when a node is transmitting is its own ID number in the system. This ID number is the BN or PN# from section 3.2 on node classification. The message also contains all the measurements that the transmitting node has collected to the other nodes.

String 1	String 2	⋯	String N+1
Transmitting Node ID	Measured Value to Node 1	⋯	Measured Value to Node N

Table 5.1: Message Format for up to N nodes

The measured values are ordered the same regardless of the ID number of the the node they were measured from. This way every receiving node will know the two nodes that the measurement was between without having to specify in the transmission.

5.3 Received Signal Strength Indicator

After a node receives a message, it goes into command mode and measures the received signal strength indicator(RSSI). The software then parses the message to find what node sent the message and adds the measurement to its table of measurements, that it will later transmit to all the other nodes.

The DB command is a diagnostics function of the software, and it returns the RSSI in dBm of the last RF packet received. The values that are returned are between -36dBm and the RF module's receiver sensitivity of -100dBm.

5.4 Information Fusion

When a node receives a packet of data of the form shown in figure 5.1 it stores it in the table shown in figure 5.2. This table holds all the ranges calculated between nodes. The measured ranges in the table should create a symmetrical matrix, but because of noise in the measurements it will not. To help reduce error the two ranges from the two halves of the table are averaged together. The resulting table is used to calculate the potential field measurements.

	BN	PN1	PN2	$\cdots$	PNx
BN	0	d_{BN1}	d_{BN2}	$\cdots$	d_{BNx}
PN1	d_{1BN}	0	d_{12}	$\cdots$	d_{1x}
PN2	d_{2BN}	d_{21}	0	$\cdots$	d_{2x}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
PNx	d_{xBN}	d_{x1}	d_{x2}	$\cdots$	0

Table 5.2: Table of Stored Measured Values

5.5 Message Coordination

There are a couple options for coordinating messages. One of them is the built in functions of the Maxstream XBee Pro. However, while these functions will coordinate message sending and confirm that each node receives the message, it limits the transmission to one node at a time. The method that was implemented was a broadcast mode, so all the nodes could receive the transmission at once. By taking advantage of the assigned alignment positions an order could be established for sending messages. Each node transmits after it receives a message from the node before it in the alignment order, starting with the Beacon Node(BN). Once the last node has transmitted, the Beacon Node will start the order off again with its transmission. Therefore the Beacon Node is the only node that needs to know the total number of Perimeter Nodes(PNs) in the system, which makes the addition or subtraction of nodes while in use a simple task. A problem that could occur is if a node in the sequence is lost, or the next node in the order is unable to receive the transmission, which would halt the transmission sequence. This is accounted for through a timeout setting, that monitors the time from the last transmission received, after a period of time that node is skipped and the next node will transmit its packet.

Chapter 6

Testing and Results

This chapter will describe the measurements that were taken and simulations that were run. The first part of this chapter describes the simulation language used. Next is an overview of how the RSSI measurements were collected, and how the conclusion drawn from those measurements were put to use in the vehicle simulations. This is followed by a series of simulations comparing the Circular Motion and Gradient Descent methods, in an ideal and noisy environment. The tests compare position with measured potential field values, as well as compare performance in a noisy environment with and without a filter.

6.1 Computer Modeling

A simple way to test algorithms without using real vehicles is to use a computer simulation. This allows for results quicker, which means more tests can be run, and to test with more vehicles then are actually available.

This section describes the testing that occurred prior to the running of simulations to model the effects seen by the real system, and integrate them into the simulations that were run.

6.1.1 Simulation Language Description

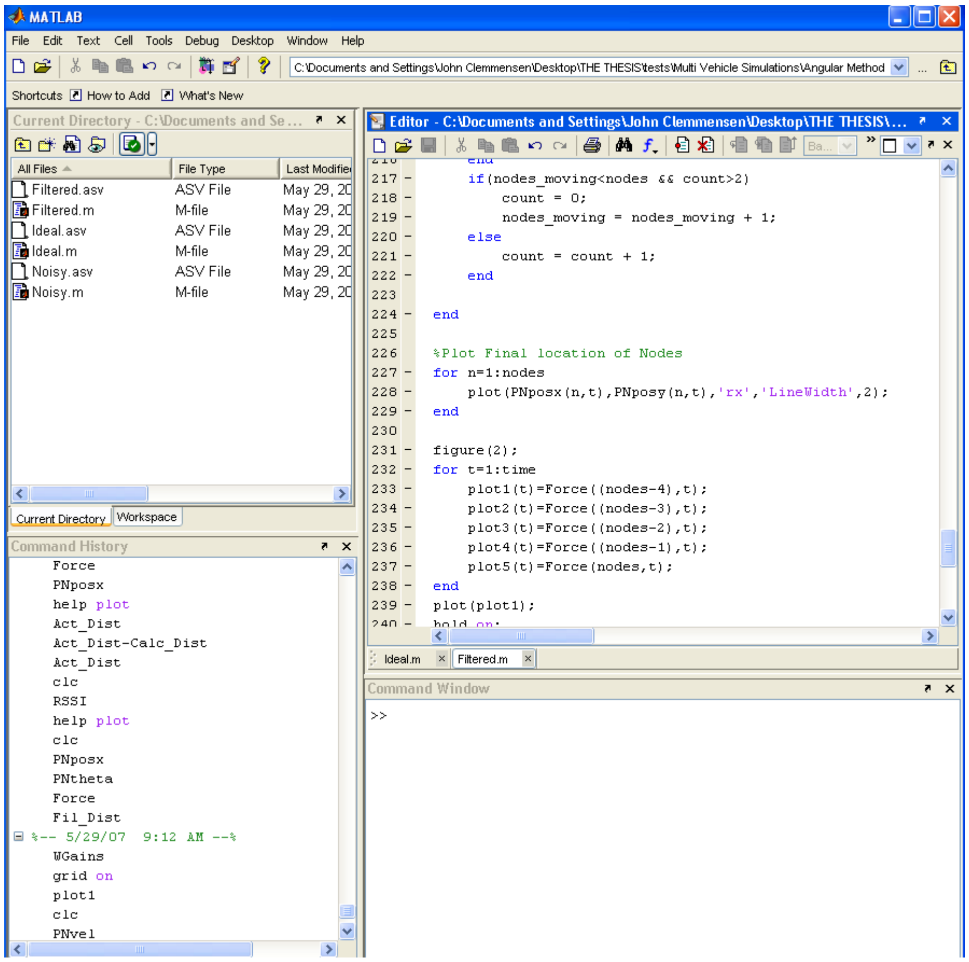
Matlab

The programming language that is being used to run the simulations for testing the control algorithms is Matlab developed by MathWorks with control testing in mind. Matlab is "the language developed for technical computing" [37]. It allows for algorithm development, data visualization, and data analysis.

Matlab was chosen over other languages such as C or C++, because it can solve technical computing problems faster and has more built in functions. It was also chosen over Labview which was used in other areas of this project because it is easier for working with matrix algorithms and plotting results. It has an interface that is similar to most programing software. A screen shot is shown in figure 6.1.

6.2 RSSI Measurements

RSSI measurements were taken using the X-CTU software that came with the XBee Pro. All measurements were taken with a clear line of sight. There were two nodes for all tests, the first was a laptop with a node communicating over the X-CTU software. The second node was a feedback node that transmitted back to the laptop node what it received. The transmission


MATLAB
File Edit Text Cell Tools Debug Desktop Window Help
Shortcuts How to Add What's New
Current Directory - C:\Documents and Se...
Editor - C:\Documents and Settings\John Clemmensen\Desktop\THE THESIS\...
Current Directory Workspace
Command History
Ideal.m Filtered.m
Command Window

C:\Documents and Settings\John Clemmensen\Desktop\THE THESIS\tests\Multi Vehicle Simulations\Angular Method

All Files	File Type	Last Modified
Filtered.asv	ASV File	May 29, 20
Filtered.m	M-file	May 29, 20
Ideal.asv	ASV File	May 29, 20
Ideal.m	M-file	May 29, 20
Noisy.asv	ASV File	May 29, 20
Noisy.m	M-file	May 29, 20

```
    if(nodes_moving<nodes && count>2)
        count = 0;
        nodes_moving = nodes_moving + 1;
    else
        count = count + 1;
    end

end

%Plot Final location of Nodes
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'rx','LineWidth',2);
end

figure(2);
for t=1:time
    plot1(t)=Force((nodes-4),t);
    plot2(t)=Force((nodes-3),t);
    plot3(t)=Force((nodes-2),t);
    plot4(t)=Force((nodes-1),t);
    plot5(t)=Force(nodes,t);
end
plot(plot1);
hold on;
```

```
    Force
    PNposx
    help plot
    Act_Dist
    Act_Dist-Calc_Dist
    Act_Dist
    clc
    RSSI
    help plot
    clc
    PNposx
    PNtheta
    Force
    Fil_Dist
%-- 5/29/07  9:12 AM --%
    WGains
    grid on
    plot1
    clc
    PNvel
```

```
>>
```

Figure 6.1: Matlab Simulation Interface

back to the laptop node from the feedback node was the transmission that was measured. The measuring occurred by the laptop node, and displayed by the X-CTU program on the laptop. Both the laptop node and the feedback node were the custom designed boards, from chapter 4, powered by LiPo batteries. The transmitting power setting was left at the default value of 18dBm. The range between nodes was calculated using the Professional Laser Rangefinder by Laser Technology Inc.. It has an accuracy of 0.3m and a max range of 1000m.

6.2.1 RSSI Distance Curve

Measurements were taken at incremental distances to measure how close the measured values were to the expected value. For each range 4 measurements were taken. The spacing between measurements were based on the slope of the ideal curve formed by the expected values, seen in figure 6.2. The greatest rate of change occurs in the first 20 meters, so samples were taken every meter. The slope lessens from 20 to 50 meters, so measurements were taken every two meters. From 50 to 100 meters the line starts to level off, so measurements were taken every 5 meters. From 100 to 150 meters the line is leveling so measurements were taken every 10 meters. The measured values for the test are in appendix B.

In figure 6.2 T1, T2, T3, T4 represent the four samples taken at each distance. A visual inspection shows how noisy the data is going to be, but does produce a curve that resembles what was expected. The noisy data is not a surprise and continues over large distances, as shown by similar tests conducted at Carnegie Mellon University[15] in figure 6.3.

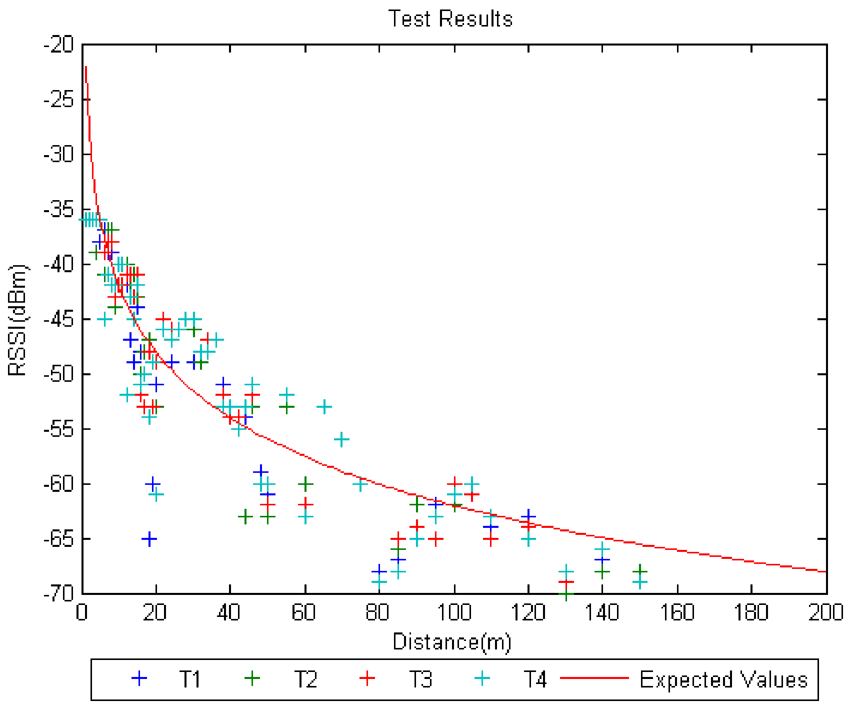


Figure 6.2: Measured RSSI Values vs Expected Values

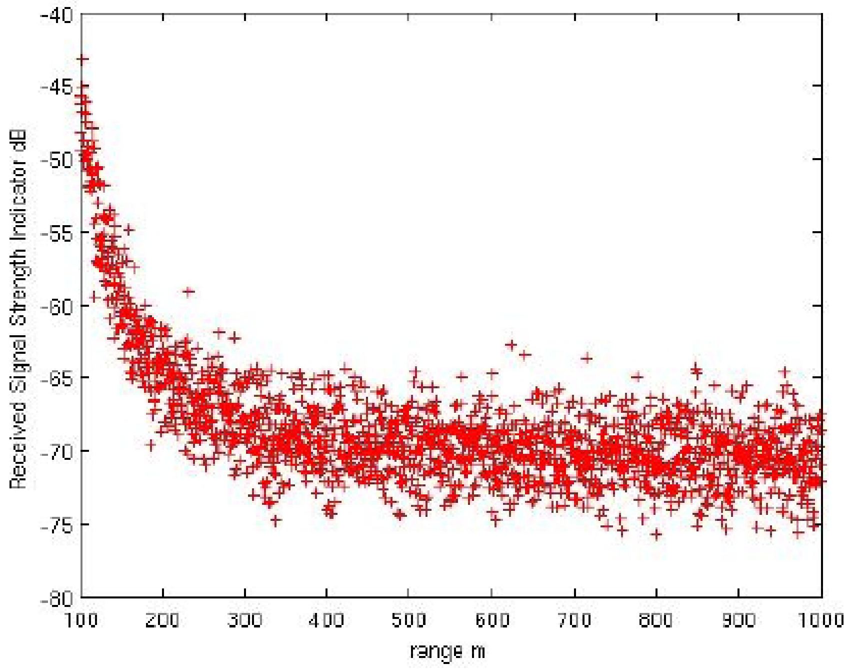


Figure 6.3: Measured RSSI Values Over a Long Range [15]

Figure 6.3 shows the results when measuring RSSI over a distance of 1000m, a range that exceeds the projected distances covered by this system, and the testing capabilities of this thesis.

6.2.2 RSSI Measurement Noise

Another source of error is measurement noise. Measurement noise occurs in the XBee Pro. This noise was tested by taking measurements at three distances, and a large number of samples were taken until a trend emerged.

The first distance measured was 25 meters. This distance was chosen because it is the lower limit of perimeter node spacing. A histogram showing

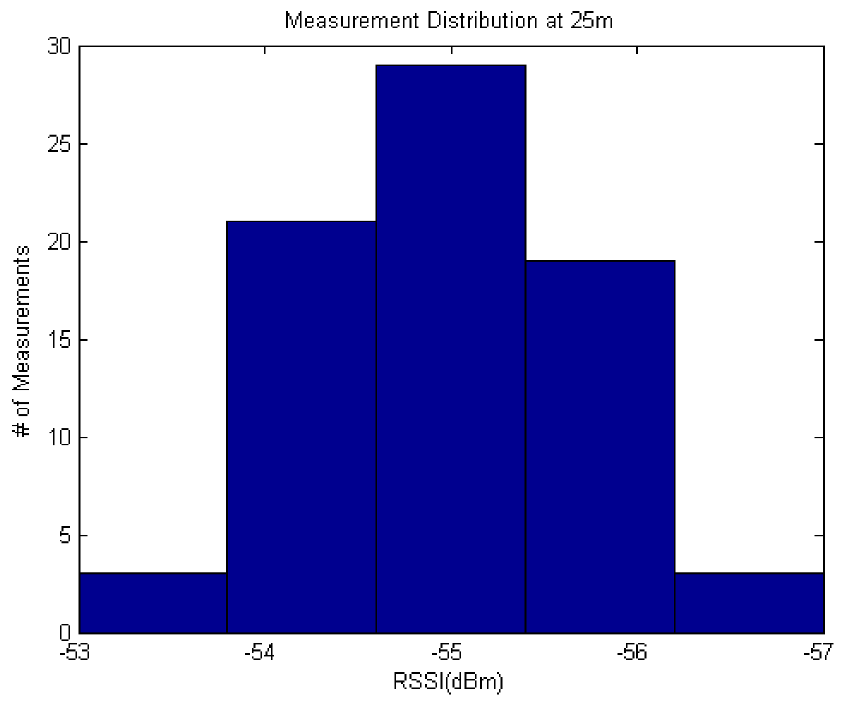


Figure 6.4: Histogram of Measured RSSI Values at 25m

the measurement distribution is shown in figure 6.4 the actual measured values for all the test are in appendix B. The first test at this distance of 30 samples was inconclusive and and a second test was performed taking 75 samples. The second test resulted in the normal distribution shown.

The second distance measured was 50 meters. This distance was chosen because it is on the outer limit of perimeter node spacing. A histogram showing the measurement distribution is shown in figure 6.5. This test also resulted in a normal distribution.

The third distance measured was 100 meters. This distance was chosen because it a likely range between a perimeter node and the beacon node. A

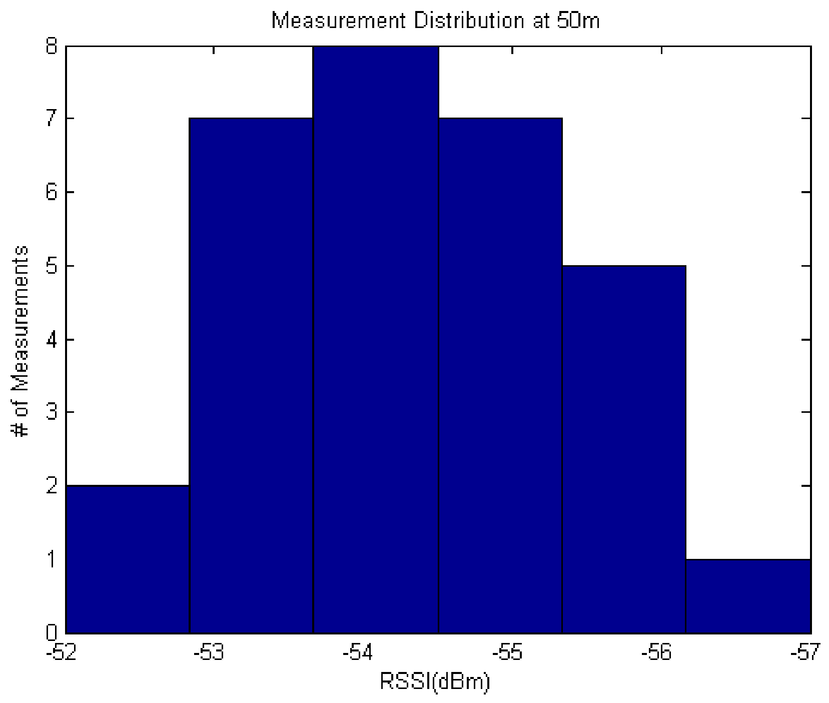


Figure 6.5: Histogram of Measured RSSI Values at 50m

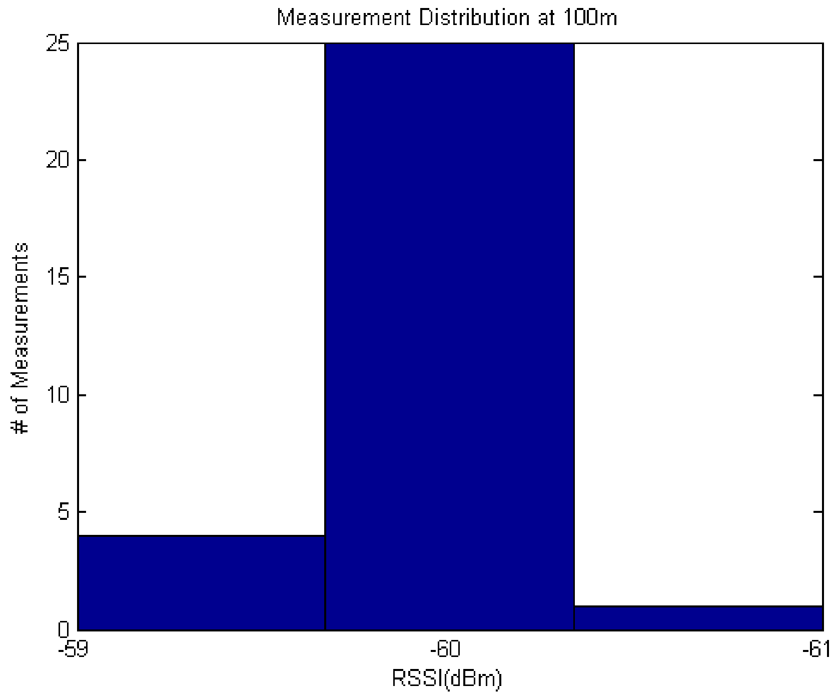


Figure 6.6: Histogram of Measured RSSI Values at 100m

histogram showing the measurement distribution is shown in figure 6.6. This test resulted in a more pronounced normal distribution, which is attributed to dBm calculations being on a log scale. The log scale is also the reason for the difference in ranges of measured values.

Analysis concluded that the measurement noise has a normal distribution.

6.2.3 RSSI Data Analysis and Modeling

Now that there is actual data to work with, corrections can be made to the ideal assumptions to make the system more robust to the real world.

The first assumption that can be altered is using the ideal RSSI loss over distance equation for calculating vehicles relative positions to each other. A more accurate equation for modeling purposes.

Best Fit Line

The ideal RSSI equation for loss vs. distance fits the shape of the data, but a better equation is found when modeling the system using Matlab. Using the function *CFTool* in Matlab, and averaging together the four samples taken in section 6.2.1 a more accurate equation is derived. This equation is a best fit line to the data, that matches with 95% confidence. The Matlab results are shown below:

```
Matlab Analysis:
Best fit from cftool
General model Power2:
       f(x) = a*x^b+c
Coefficients (with 95% confidence bounds):
       a =       34.91  (-27.12, 96.94)
       b =      0.1457  (-0.03153, 0.3229)
       c =      -5.685  (-72.19, 60.82)

Goodness of fit:
  SSE: 704.8
  R-square: 0.8569
  Adjusted R-square: 0.8509
```

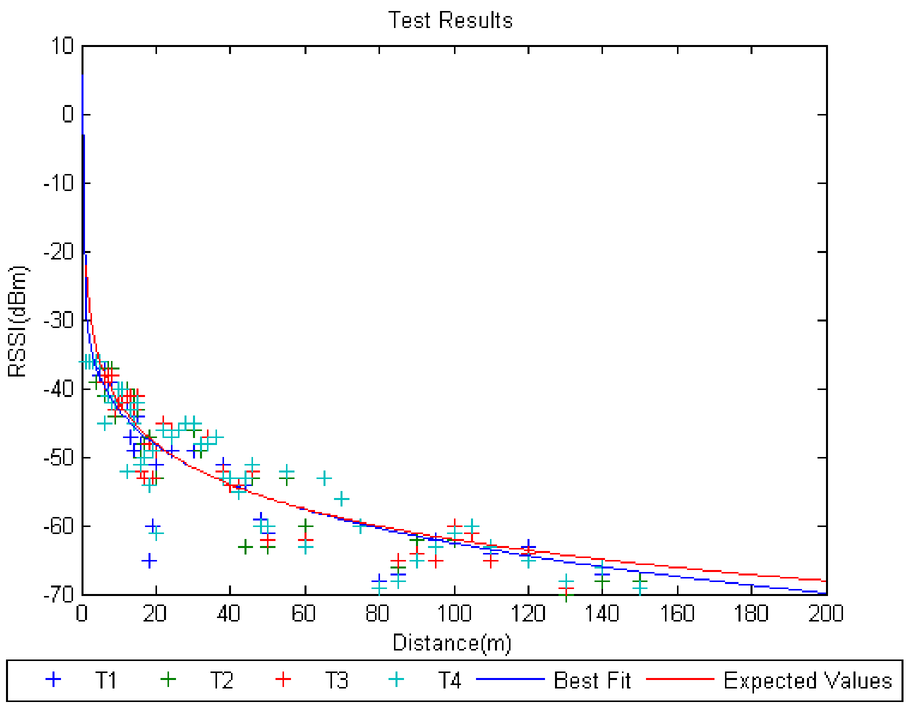


Figure 6.7: Measured RSSI Values With Best Fit Line

```
RMSE: 3.832
```

The resulting equation used in simulation for modeling RSSI value as a function of distance was

$$f(x) = 34.91 \cdot x^{0.1457} - 5.685 \tag{6.1}$$

Resulting in a equation that is very similar to the ideal equation but will give more results closer to those actually going to be measured over a range of 200 meters. A comparison of the two equations and the measured points is shown in figure 6.7.

Noise

Now that actual results can be looked at, a better idea of the values that can be expected during operation can be predicted. Using the best fit line calculated in the previous section as the ideal response, instead of the standard ideal RSSI calculation, a more accurate base line is set. Next error was injected into that equation to model the noise seen in actual measurements. Using the 'randn' function in Matlab to create a normally distributed noise to replicate the results seen in section 6.2.2. The results injecting this noise are shown in figure 6.8, where the red marks are the average of the measured results at each distance, the light blue line is the best fit line from the previous section, and the blue marks are the modeled noisy data for each distance. Since the noise uses the 'randn' function the data will have a normal distribution but return different values during each simulation.

6.3 Measurement Filter

The results of the tests preformed in the previous section resulted in noisy results, which would also be expected to occur during actual usage of the system. Analyzing the data unfiltered would impair the ability of the control system to efficiently direct the vehicle to its assigned position. To get better results from each measurement of the RSSI a FIR filter, Finite Impulse Response filter, was implemented.

A FIR filter was chosen because the span of previous of samples that are considered can be limited. Since the nodes are going to be in motion, samples are only valid for a period of time in proportion to the distance

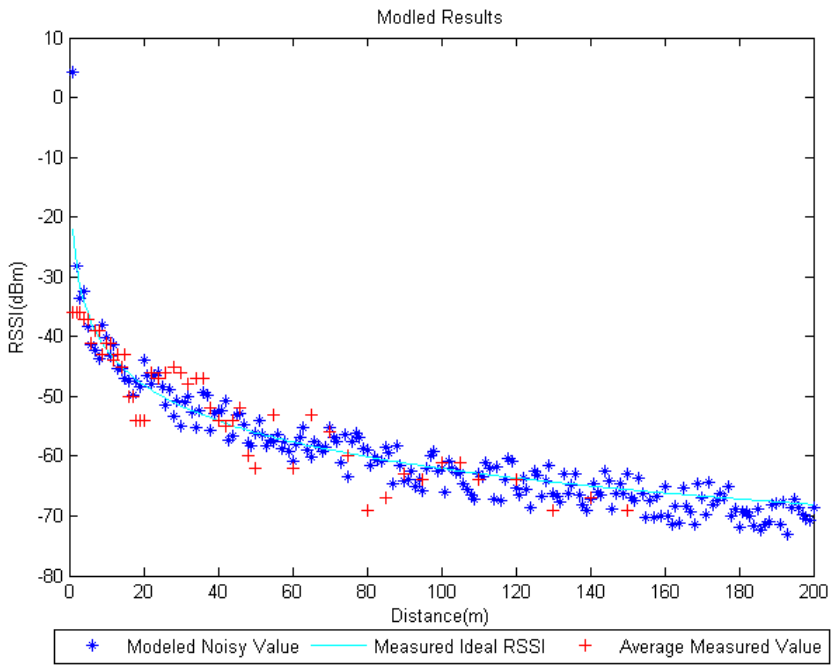


Figure 6.8: Overlay of Measured, Modeled, and Ideal RSSI Values

traveled and the sample rate. Therefore, the use of an IIR filter, Infinite Impulse Response filter, which considers all previous samples, would not be as efficient. The filter, given a window size, weights each sample in the window view and weights them with the largest weight going to the most recent sample and the smallest to the last measurement in the window. They are then summed, and divided by the factorial of the current weight [16] [17].

$$FRSSI(t) = \frac{\sum_{i=0}^{w} RSSI(t-i) \times (w-i)}{w!} \tag{6.2}$$

$$w = \begin{cases} w\ if & w \leq \ \ t \\ t\ if & w > \ \ t \end{cases} \tag{6.3}$$

6.4 Testing Formation Algorithms

The following sections demonstrate the movement and coordination of multiple nodes in several different environments, ideal simulation, real world simulation that has error, and finally a demonstration of the final design using filters to help compensate for the noisy measurements. All simulations use the vehicle model from section 3.3, and the XBee Pro communication modules for ranging, with the noise measured in section 6.2.

6.4.1 Circular Motion

These tests are of the circular motion algorithms described in section 3.5.1. For each test the first figure shows the vehicle movement, and the second shows measured values in the potential field.

Ideal Environment

This simulation was performed using a BN, and nine PN's. Figure 6.9 tracks the position of PN5 - PN9 in a local coordinate system. For clarity only the last five nodes were plotted because the plot is too crowded with all nine and these are the nodes of interest. These nodes are dependent on all the nodes that come before them in ranking, therefore any problems that would result in earlier nodes would show up in these nodes.

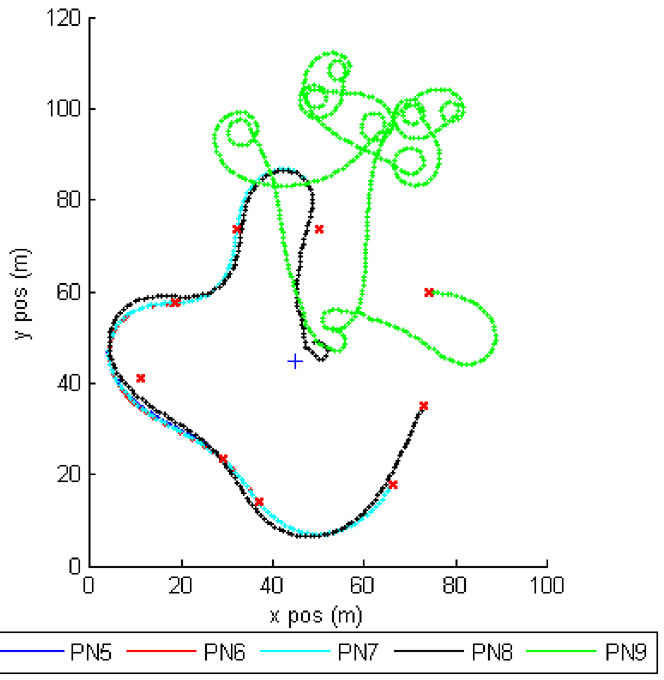


Figure 6.9: 9 Ideal Circular Motion Nodes moving into Formation

Figure 6.10 shows the gradient field measured by each node as a function of time. When compared with figure 6.9 it can be seen that the nodes do not navigate to the point with the least measured force, or potential field, but move in a circle around the beacon node until they find their orientation.

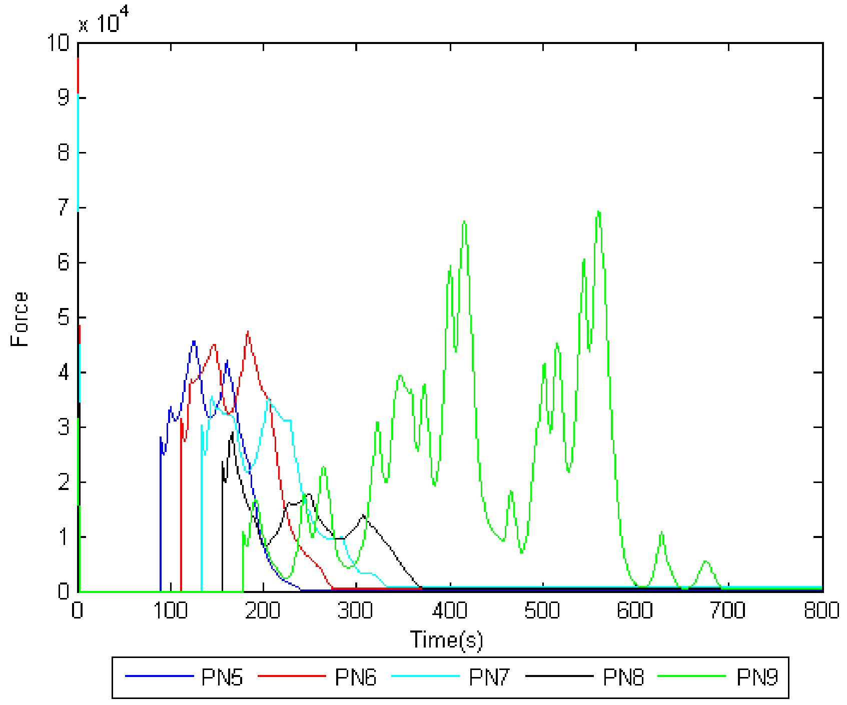


Figure 6.10: Perimeter Nodes 5-9 Measured Forces in the Potential Field using Ideal Circular Motion

In this example all the nodes eventually find their alignment in the formation. However even for an ideal system a perfect formation can not be achieved. To achieve a perfect formation the control gains must be tuned higher for more precise turns to keep the desired distance from the beacon node. This example demonstrates what happens when the control gains get tuned too high. Perimeter node 9 is an example, as shown the gains are so aggressive that its search pattern for the perimeter leads it to turn in tight circles missing the desired distance and the proper orientation to maintain it. Because of the nodes inability to precisely align all the nodes were given a threshold value to approximate their positions. Otherwise the nodes would continue to circle the beacon node and never settle.

Unfiltered Noisy Environment

This section demonstrates the control system in a real world environment. It pulls together all the data that has been tested involving a circular motion based control algorithm and ranging information using signal strength. This simulation was performed using a BN, and nine PNs. Figure 6.11 tracks the position of PN5 - PN9 in a local coordinate system.

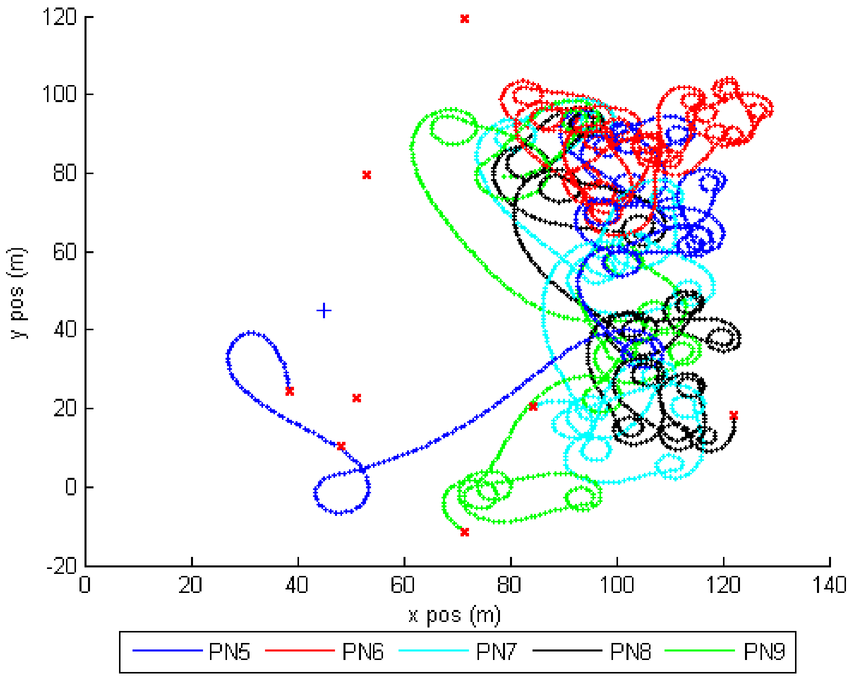


Figure 6.11: 9 Circular Motion Nodes moving into Formation with Unfiltered Noisy Measurements

Figure 6.12 shows the gradient field measured by each node as a function of time. When compared with figure 6.11 it can be seen, as in the previous test, that the nodes do not navigate to the point with the least measured force, or potential field, but moves in a circle around the beacon node until it finds its orientation. However, in this test it takes longer for the orbit to be achieved and the radius of the orbit is less precious.

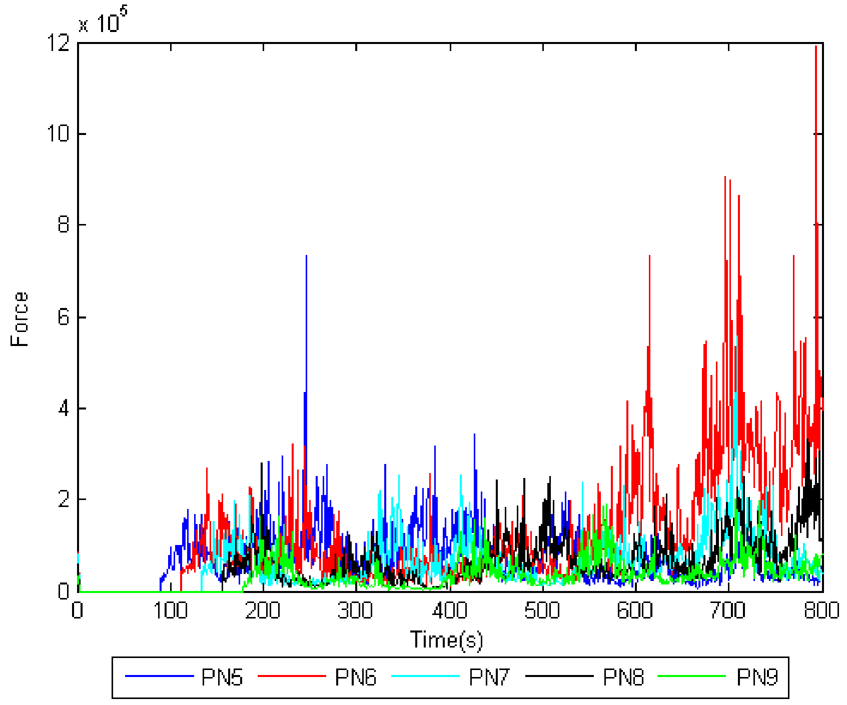


Figure 6.12: Perimeter Nodes 5-9 Measured Forces in the Potential Field using Circular Motion with Unfiltered Noisy Measurements

The perimeter nodes are not able to keep an orbit around the beacon node because of the noise in the measurement. Therefore, the nodes never settle into a formation.

Filtered Noisy Environment

This section demonstrates the control system using the measurement noise and a filter to smooth it as in a real world environment. It pulls together all the data that has been tested involving the circular motion control algorithm, RSSI noise measurements, and adds a filter to help compensate. This simulation was performed using a BN, and nine PNs. Figure 6.13 tracks the position of PN5 - PN9 in a local coordinate system.

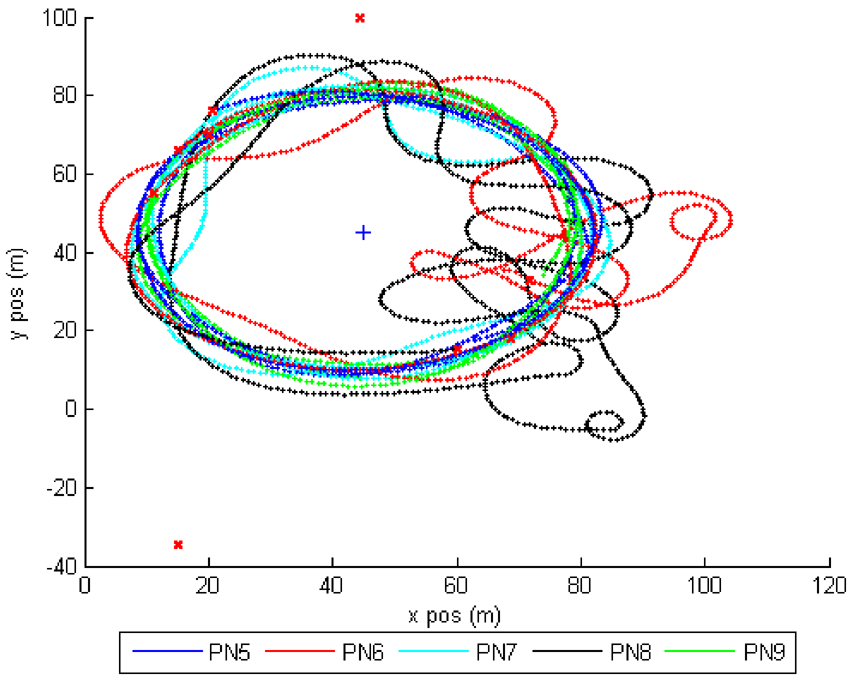


Figure 6.13: 9 Circular Motion Nodes moving into Formation with Filtered Noisy Measurements

Figure 6.14 shows the gradient field measured by each node as a function of time. When compared with figure 6.13 it can be seen, as in the previous test, that the nodes do not navigate to the point with the least measured force, or potential field, but moves in a circle around the beacon node until it finds its orientation.

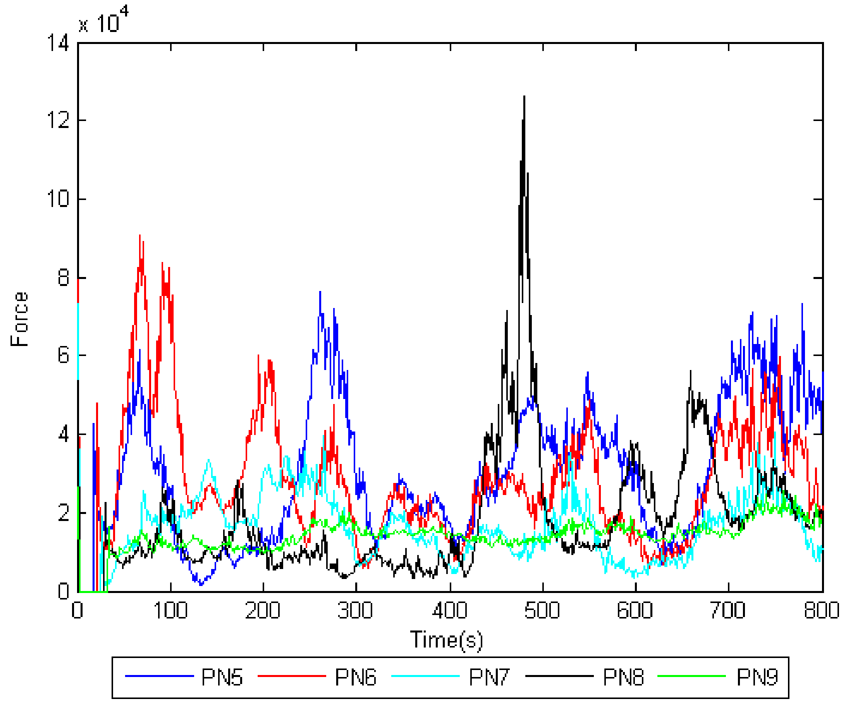


Figure 6.14: Perimeter Nodes 5-9 Measured Forces in the Potential Field using Circular Motion with Filtered Noisy Measurements

It can be concluded that the nodes do not move in an optimal path to their orientations in the formation. The nodes move along the perimeter but are unable to achieve a formation because the measurement noise does not allow for a threshold to be set that will keep the nodes in formation.

6.4.2 Gradient Descent

These tests use the gradient descent algorithms described in section 3.5.2. For each test the first figure shows the vehicle movement, and the second shows measured field values.

Ideal Environment

This simulation was performed using a BN, and nine PNs. Figure 6.15 tracks the position of PN5 - PN9 in a local coordinate system. For clarity only the last five nodes were plotted because the plot is too crowded with all nine and these are the nodes of interest. These nodes are dependent on all the nodes that come before them in ranking, therefore any problems that would result in earlier nodes would show up in these nodes.

In figure 6.16 is the gradient field measured by each node as a function of time. When compared with figure 6.15 it is shown how the nodes navigate to the point with the least measured force in the potential field.

Unfiltered Noisy Environment

This section demonstrates the control system in a real world environment, meaning that there will be noise in the measurements. It pulls together all the data that has been tested involving a gradient descent based control algorithm and ranging information using signal strength. This simulation was performed using a BN, and nine PNs. Figure 6.17 tracks the position of PN5 - PN9 in a local coordinate system. The difference with this test is that noise is introduced to the system.

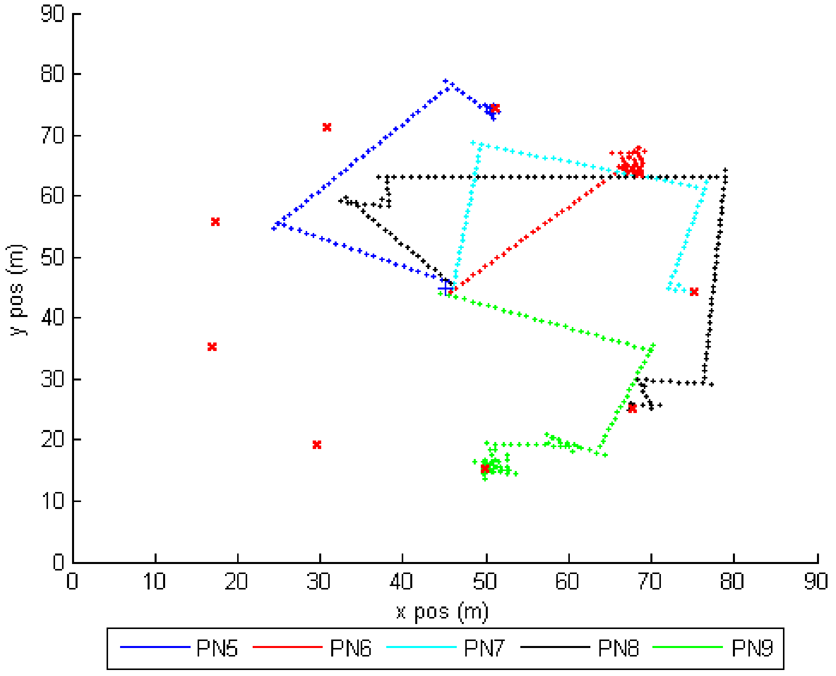


Figure 6.15: 9 Ideal Gradient Descent Nodes Moving into Formation

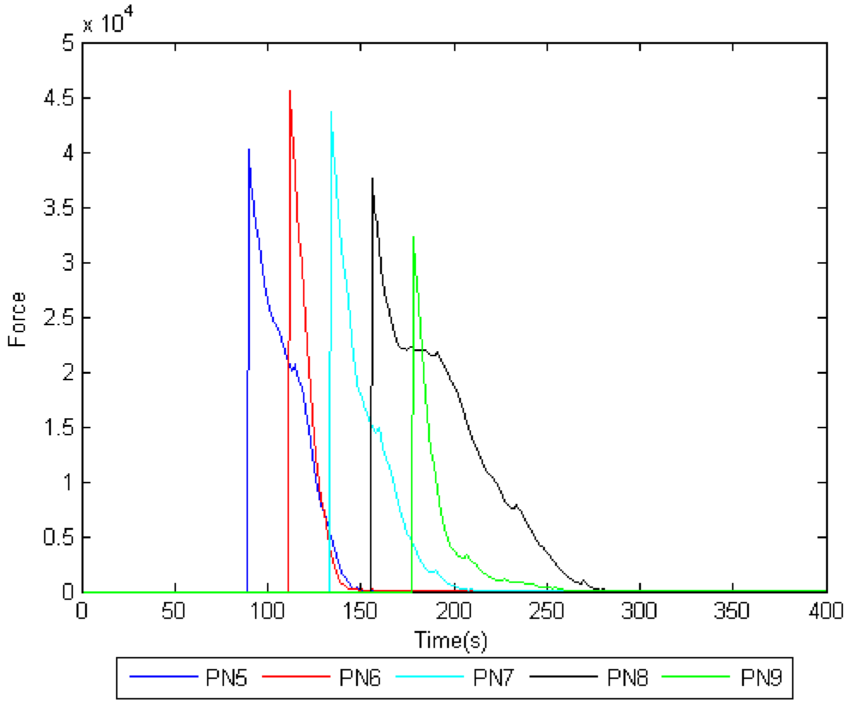


Figure 6.16: Perimeter Nodes 5-9 Measured Forces in the Potential Field Using Ideal Measurements

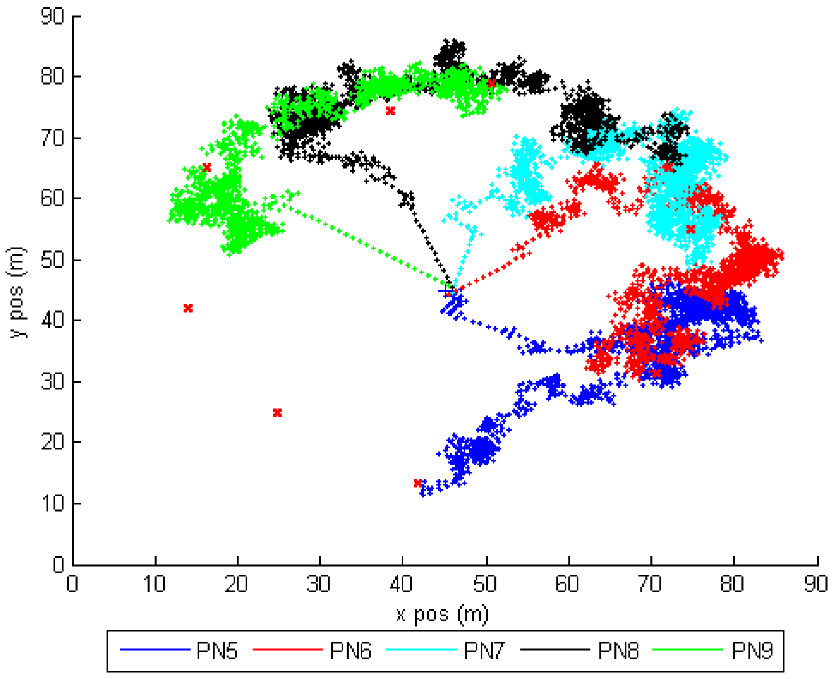


Figure 6.17: 9 Gradient Descent Nodes moving into Formation with Unfiltered Noisy Measurements

Figure 6.18 shows the gradient field measured by each node as a function of time. When compared with figure 6.17 it can be seen how the nodes navigate to the point with the least measured force in the potential field.

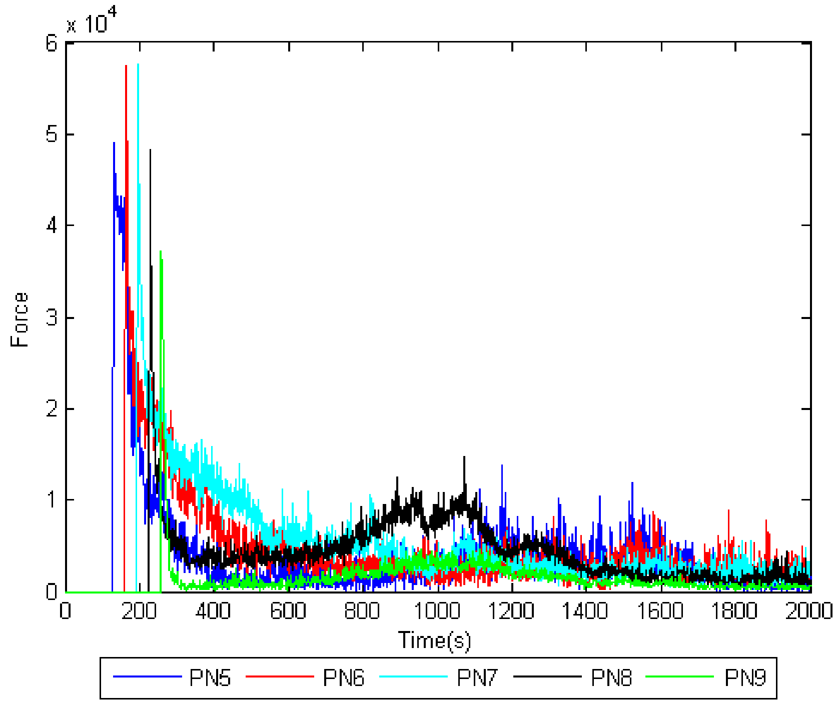


Figure 6.18: Perimeter Nodes 5-9 Measured Forces in the Potential Field using Gradient Descent with Unfiltered Noisy Measurements

This test demonstrates that a very loose formation is achievable in a noise environment. To improve the performance a filter is added. The filter is tested in the following section.

Filtered Noisy Environment

This section demonstrates the control system with noise, as in a real world environment like the previous section but adds a filter to improve performance. This simulation was performed using a BN, and nine PNs. Figure 6.19 tracks the position of PN5 - PN9 in a local coordinate system.

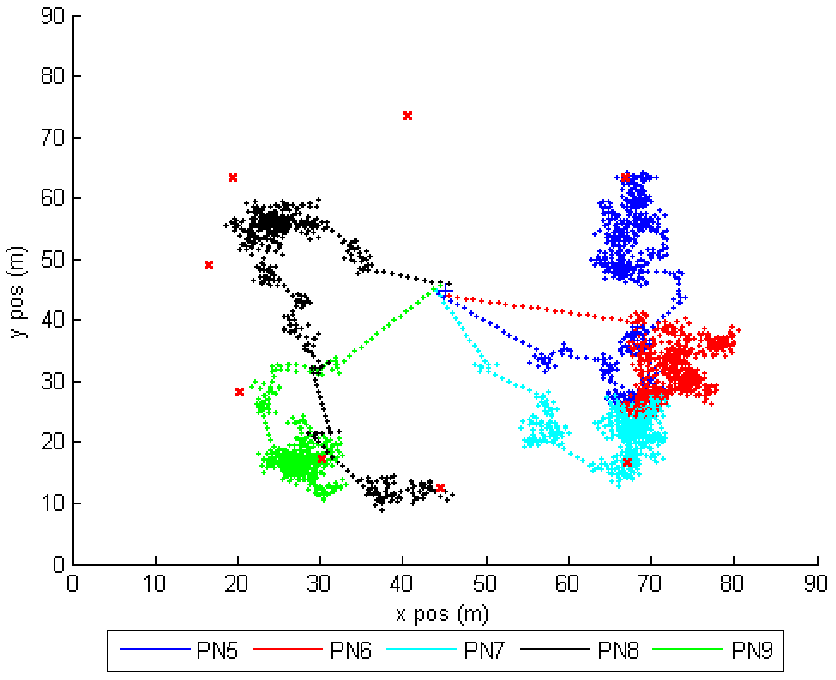


Figure 6.19: 9 Gradient Descent Nodes moving into Formation with Filtered Noisy Measurements

Figure 6.20 shows the gradient field measured by each node as a function of time. When compared with figure 6.19 it can be seen how the nodes navigate to the point with the least measured force in the potential field.

From the figures it can be concluded that the nodes act intelligently in their task of finding their position in the formation. All the nodes take a path that reduces their current measured value in the potential field and the measurement approaches zero as best it can with noise in the measurements.

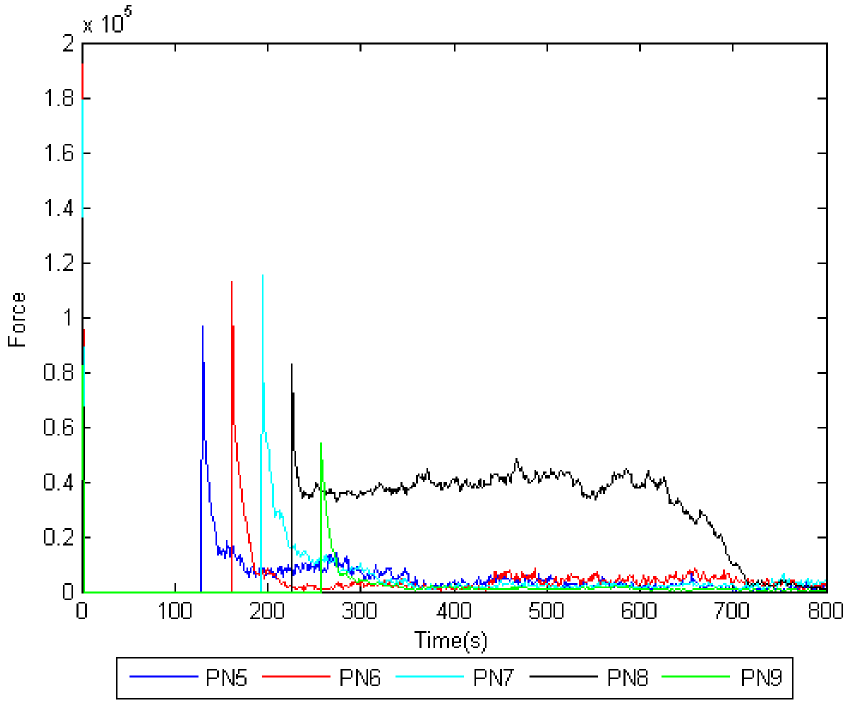


Figure 6.20: Perimeter Nodes 5-9 Measured Forces in the Potential Field using Gradient Descent with Filtered Noisy Measurements

Chapter 7

Conclusion

Global Positioning Systems have become relied upon to navigate autonomous vehicles into a formation. The goal of this paper was to develop an alternative to GPS by aligning vehicles based on measured relative distances between nodes. This goal was accomplished by comparing control algorithms to determine the most effective and designing custom hardware and software for implementation. An unintended benefit to the designed system is that it will keep all vehicles in communication range of one another. The weighting of the potential field is such that vehicles are encouraged to work inside the perimeter thus there is little risk of a vehicle moving out of communication range of the other vehicles.

7.1 Control Comparison

Both of the control algorithms tested in chapter 6.4 would offer solutions in an ideal environment. However, the gradient descent method had the shortest settling time and was the only one to converge to a formation when measurement noise was introduced. The simulations showed the gradient descent method would be the most effective of the two in navigating the vehicles into a formation.

7.2 Future Work

The next step in the development of this design would be integration on actual vehicles. This would require the use of several autonomous ground vehicles running Labview Embedded with a local control that can avoid obstacles. To improve upon the system further work could be done on the measurement noise and on the communication protocols.

7.2.1 Sensor Measurement

An improved sensor for measuring RSSI would greatly increase the effectiveness of this design. The current sensor's variation when measuring the same distance under the same conditions was a hindrance that could be corrected. A measurement that was repeatable would increase the accuracy of the gradient that is calculated thus reducing the settling time and resulting in a more precious formation.

7.2.2 Communications

A more advanced messaging protocol would add a great deal of flexibility to the system by allowing for the number of vehicles to be dynamic. Currently the number of nodes is set prior to operation, and if a node is lost or added the system is unable to correct the vehicle alignments while in operation. A future protocol would also have the ability to assign each node its order in the perimeter. An optimal way of assigning nodes would be based on a comparison of current position based on potential field measurements to all possible positions. Thus each node would be assigned a position in the formation that is closest to it.

References

[1] United States Senate, *Fiscal Year 2001 Defense Authorization Bill*, Sec 217.

[2] B. Gombar, C. Terwelp, M. Fleming, A. Wicks, C. Reinholtz, "JOUSTER: Joint Unmanned Systems Test, Expermentation, and Research Facility," *ITEA Journal*, Dec 2004/Jan 2005, pp. 9-13.

[3] *Standard Operating Procedures for Military Operations in Urban Terrain* Available Online: www.usmc.mil

[4] Millennium Sensor, LLC www.msensor.com

[5] B. Schott, R. Parker, "Defense Applications for Large Numbers of Distributed Small Sensors," *2005 IEEE Microwave Symposium Digest*, 12-17 June 2005.

[6] Y. Chen, Z. Wang,"Formation Control: A Review and A New Consideration," *IEEE Internation Conference on Intelligent Robots and Systems*, August 2-6, 2005, pp. 3181-3186.

[7] N. Ceccarelli, M.D. Marco, A. Garulli, A. Giannitrapani, "Collective Circular Motion of Multi-Vehilce Systems with Sensory Limitations," *Proceddings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Dec 2005, pp. 740-745.

[8] N.E. Leonard, E. Fiorelli, "Virtual Leaders, Artificial Potentials and Coordinated Control of Groups," *Proceddings of the 40th IEEE Conference on Decision and Control*, Dec 2001, pp. 2968-2973.

[9] V. Stoian, M. Ivanescu, E. Stoian, C. Pana, "Using Artificial Potential Field Methods and Fuzzy Logic for Mobile Robot Control," *12th International Power Electronics and Motion Control Conference*, Aug. 2006, pp. 385-389.

[10] D.J. Stilwell, B.E. Bishop, C.A. Sylvester, "Redundant Manipulator Techniques for Partially Decentralized Path Planning and Control of a Platoon of Autonomous Vehicles," *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol. 35, No. 4*, Aug. 2005, pp. 842-848.

[11] H. Ohtake, K. Tanaka, H.O. Wang,"Polar Coordinate Fuzzy Model and Stability Analysis," *Proceedings of the American Control Conference*, June 2003, pp. 1914-1919.

[12] D. Lymberopoulos, Q. Lindsey, and A Savvides, "An Empirical Characterization of Radio Signal Strength Variablity in 3-D IEEE 802.15.4 Networks Using Monopole Antennas," *EWSN 2006, LNCS 3868*, Available Online: www.eng.yale.edu, 2006.

[13] J. A. Snyman, "Practical Mathematical Optimization: An introduction to basic optimization theory and classical and new gradient-based algorithms," Springer, Cambridge, Massachusetts, February 2005, (ISBN 0-387-24348-8)

[14] K. Shimizu, K. Tamura "Control of Nonholonomic Systems via Direct Gradient Descent Control," *Proceedings of the 2004 IEEE International Conference on Control Applications*, September 2-4, 2004, Taipei, Taiwan, pp. 837-841.

[15] Scerri P., Glinton R., Owens S., Scerri D., and Sycara K., "Geolocation of RF Emitters by Many UAVs," *AIAA Conference and Exhibit* 7-10 May 2007, Rohnert Park, CA.

[16] M.J.T. Smith, "A Novel FIR. Filter Design Method Based on Windowing," *Circuits and Systems, 1989., IEEE International Symposium* 8-11 May 1989, Portland, OR, pp. 347-350 vol.1

[17] C. Jing, Y. Jinsheng, D. Runtao, "Fuzzy Weighted Average Filter," *Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference* 21-25 Aug. 2000, Beijing, China, pp. 525-528 vol.1

[18] A.S. Watkins, R. J. Prazenica, A.J. Kurdila, G.J. Wiens, "Vision-Based RHC for an Autonomous Vehicle in an Urban Environment," *2006 American Control Conference*

[19] Y. Chung, C. Park, F. Harashima, "A Position Control Differential Drive Wheeled Mobile Robot," *IEEE Transactions on Industrial Electronics*, vol. 48, No. 4, August 2001.

[20] T. Balch, R. Arkin, "Behavior-Based Formation Control for Multirobot Teams," *IEEE Transactions on Robotics and Automation*, vol, 14, No. 6, December 1998.

[21] A. Howard, M. Mataric, G. Sukhatme, "Putting the 'I' in 'Team': an Ego-Centric Approach to Cooperative Localization," *IEEE Internation Conference on Robotics and Automation*, September 14-19, 2003,Taipei, Taiwan, pp. 868-874.

[22] C. Jones, M. Mataric, "From Local to Global Behavior in Intelligent Self-Assembly,"*IEEE Internation Conference on Robotics and Automation*, September 14-19, 2003,Taipei, Taiwan, pp. 721-726.

[23] W. Chen, T. Wu, H. Chao, "Virtual Polar Coordinate Space with Fuzzy Preserving Consideration for Sensor Networks," *Wireless and Optical Communication MultiConference*,July 8-10, 2004.

[24] Z. Lin, M. Broucke, B. Francis, "Local Control Strageties for Groups of Mobile Autonomous Agents," *IEEE Transactions on Automatic Control*, vol. 49, No. 4, April 2004.

[25] J. Fredslund, M. Mataric, "Robot Formations Using Only Local Sensing and Control," *Proceeding so 2001 IEEE International Symposium on Computational Intelligence in Robotics and Automation* July 29-August 1, 2001, Banff, Alberta, Canada, pp. 308-313.

[26] S. Capkun, M. Hamdi, J. Hubaux, "GPS-Free Positioning in Mobile Ad-Hoc Networks," *Proceedings of the 34th Hawaii International Conference on System Sciences - 2001.*

[27] R. Iyengar, B. Sikdar, "Scalable and Distributed GPS free Positioning for Sensor Networks," *IEEE International Conference on Communications, 2003*, vol. 1, May 11-15, 2003, pp. 338-342.

[28] V. Kukshya, H. Krishnan, C. Kellum, "Performance Evaluation of a System for Estimating Relative Positions of Vehicles During GPS Outages," *Intelligent Vehicles Symposium 2006*, June 13-15, 2006, Tokyo, Japan, pp. 394-399.

[29] Free-Space Loss on Wikipedia, Available Online: www.wikipedia.com

[30] ZigBee; www.zigbee.org

[31] Maxstream, Inc., *XBee/XBee-PRO OEM RF Modules Product Manual* v1.xAx Available Online: www.maxstream.net

[32] Micrel Semiconductors, *MIC39100 Data Sheet*, August 2005 Available Online: www.micrel.com

[33] Texas Instruments, *BQ24123 Data Sheet*, March 2006 Available Online: www.ti.com

[34] Maxim, *MAX3232 Data Sheet*,rev. 7, January, 2007 Available Online: www.maxim-ic.com

[35] FTDI Chip, *FT232L Data Sheet*, 2005 Available Online: www.ftdichip.com

[36] Ben Hastings, "Design of a Micro Wireless Instumented Payload for Unmanned Vehicle Testing," Master's thesis, Virginia Tech, 2006

[37] Matlab, www.mathworks.com

Appendix A

Code Listing

A.1 Angular Motion Software

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Angular Motion Software%%%%%%%%
%%%%       JSC 6/2/07      %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all; clear all; clc;
%Length of Simulation
time = 800;
%Number of Perimeter Nodes
nodes = 9;
%Perimeter Radius in Meters
radius = 30;
%Max speed of a Node m/s
vel = 1;
%Transmission Frequency
freq= 2400000000;%Hz = 2.4GHz
%Transmission Strenght
Tstrength = 18;%dB
%Window Size of filter
window = 4;
%Window Gains
for x = 1:window
    WGains(x) = 2^(window - x);
end
%other inital conidtions
c = 3;
psi = 7/4*pi;
kE = .15;

count = 0;
%Position of central beacon
BNx = 1.5*radius;
BNy = 1.5*radius;
%Plot Size
MAXx = 2*BNx;
```

```
MAXy = 2*BNy;

%Starting Position and Orientation for all perimeter nodes
%Start them all at the center
t=1;
for n=1:nodes
    PNposx(n,t) = BNx+30-60*rand(1);
    PNposy(n,t) = BNy+30-60*rand(1);
    PNtheta(n,t) = 2*pi*rand(1);
    PNvel(n,t) = vel;
    PNint(n) = 0;
    PNalpha(n,t)=1;
    PNgamma(n,t)=2;
end

%Desired Final Spacing Between Nodes
%All PNs want to be the dist of the Radius away from BN
for n=2:nodes
    for d=1:(n-1)
        desir_dist(n,d)= sqrt(2*radius^2-2*radius^2*cos(2*(n-d)*pi/nodes));
        gain(n,d) = (nodes-(n-1))/(radius/desir_dist(n,d));
    end
end

%Plot Starting Positions of Nodes
figure(1);
plot(0,0,MAXx,MAXy)
ylabel('y axis(m)');xlabel('x axis(m)');
hold on;
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'o');
end
plot(BNx,BNy,'r+');

%Current Actual Distance Between Nodes
for n=1:nodes
    Act_Dist_BN(n,t) = sqrt((PNposx(n,t)-BNx)^2+(PNposy(n,t)-BNy)^2);
    for d=1:nodes
        Act_Dist(n,d,t) = sqrt((PNposx(n,t)-PNposx(d,t))^2+(PNposy(n,t)-PNposy(d,t))^2);
    end
end

%Converting Actual Distance to RSSI, Making it Noisy
%Dont use actual distance here because they all start at zero and that
%causes error, will need to change if they dont all start at the same spot
for n=1:nodes
    RSSI_noisy_BN(n,t) = Tstrength - 20*log10(Act_Dist_BN(n,t))+20*log10(freq)-147.55+ 4
    - (rand(1)) -(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1));
    for d=1:nodes
        RSSI(n,d,t) = Tstrength - 20*log10(1)+20*log10(freq)-147.55;
        RSSI_noisy(n,d,t) = RSSI(n,d,t) + 4 - (rand(1)) -(rand(1))-(rand(1))-(rand(1))
        -(rand(1))-(rand(1))-(rand(1))-(rand(1));
    end
end

%Calculating Distance from RSSI (No Filter)
```

```
for n=1:nodes
    Calc_Dist_BN(n,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20 - RSSI_noisy_BN(n,t)/20);
%     Calc_Dist_BN(n,t) = n;
    for d=1:nodes
        Calc_Dist(n,d,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20 - RSSI_noisy(n,d,t)/20);
    end
end

%Calculate Current Field Felt by each Node
for n=1:nodes
    Force(n,t)= abs(sqrt((PNposx(n,t)-BNx)^2+(PNposy(n,t)-BNy)^2)-radius)^3;
    for d=1:(n-1)
        Force(n,t) = Force(n,t)+ gain(n,d)*(Calc_Dist(n,d,t)- desir_dist(n,d))^2;
    end
end

%Simulate Vehicle Movement
nodes_moving = 1;
for t=2:time
    for n=1:nodes
         if n<=nodes_moving
            %Update Positions
            PNposx(n,t) = PNposx(n,(t-1)) + PNvel(n,(t-1))*cos(PNtheta(n,(t-1)));
            PNposy(n,t) = PNposy(n,(t-1)) + PNvel(n,(t-1))*sin(PNtheta(n,(t-1)));
            plot(PNposx(n,t),PNposy(n,t));

            %Update Actual Distance Between Nodes
            Act_Dist_BN(n,t) = sqrt((PNposx(n,t)-BNx)^2+(PNposy(n,t)-BNy)^2);
            for d=1:nodes
                Act_Dist(n,d,t) = sqrt((PNposx(n,t)-PNposx(d,t))^2+(PNposy(n,t)-PNposy(d,t))^2);
            end

            %Converting Actual Distance to RSSI, Making it Noisy
            RSSI_noisy_BN(n,t) = Tstrength - 20*log10(Act_Dist_BN(n,t))+20*log10(freq)-147.55+ 4
            - (rand(1)) -(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1));
            for d=1:(n-1)
                RSSI(n,d,t) = Tstrength - 20*log10(Act_Dist(n,d,t))+20*log10(freq)-147.55;
                RSSI_noisy(n,d,t) = RSSI(n,d,t) + 4 - (rand(1))-(rand(1))
                -(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1))-(rand(1));
            end

            %Calculating Distance from RSSI (No Filter)
            Calc_Dist_BN(n,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20
            - RSSI_noisy_BN(n,t)/20);
                if t >=window
                    tmax = window;
                elseif t < window
                    tmax = t;
                end

                f = t;
                fil = 0;
                Gsum = 0;
                for x = 1:tmax
                    fil = fil + Calc_Dist_BN(n,f)*WGains(x);
                    Gsum = Gsum + WGains(x);
                    f= f - 1;
                end
                Fil_Dist_BN(n,t) = fil/Gsum;

            for d=1:(n-1)
                Calc_Dist(n,d,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20
                - RSSI_noisy(n,d,t)/20);

                if t >=window
                    tmax = window;
                elseif t < window
                    tmax = t;
                end

                f = t;
                fil = 0;
                Gsum = 0;
                for x = 1:tmax
                    fil = fil + Calc_Dist(n,d,f)*WGains(x);
                    Gsum = Gsum + WGains(x);
                    f= f - 1;
                end
                Fil_Dist(n,d,t) = fil/Gsum;
            end

            %Calculate Current Field Felt by each Node
            Force(n,t)= abs(Fil_Dist_BN(n,t)-radius)^3;
            for d=1:(n-1)
                Force(n,t) = Force(n,t)+ gain(n,d)*(Fil_Dist(n,d,t)- desir_dist(n,d))^2;
            end

            %Velocity
            if Force(n,t) > 2
               PNvel(n,t) = PNvel(n,(t-1));
            else
               PNvel(n,t) = 0.000000001;
            end

            %Control inputs of velocity and direction
            calibrate = mod(t , 200);
            if(calibrate<3 && Fil_Dist_BN(n,t)>3 || Fil_Dist_BN(n,t)>1.5*radius)
                PNgamma(n,t) = pi-real(acos((Fil_Dist_BN(n,t)^2+PNvel(n,t)^2-Fil_Dist_BN(n,(t-1))^2)
                /(2*Fil_Dist_BN(n,t)*PNvel(n,t))));
            else
                PNgamma(n,t) = PNgamma(n,(t-1));
            end
                gain(n,t) = log(((c-1)*Fil_Dist_BN(n,t) + radius)/(c*radius));
                %Calculate change in theta
                if (PNgamma(n,t)>=0 && PNgamma(n,t)<=psi)
                    PNgamma(n,t) = PNgamma(n,(t-1)) + PNvel(n,t)/Fil_Dist_BN(n,t)*sin(PNgamma(n,t))
                    -kE*gain(n,t)*PNgamma(n,t);
                else
                    PNgamma(n,t) = PNgamma(n,(t-1)) + PNvel(n,t)/Fil_Dist_BN(n,t)*sin(PNgamma(n,t))
                    -kE*gain(n,t)*(PNgamma(n,t)-2*pi);
                end
```

```
                %Find alpha
                PNgamma(n,t)=mod(PNgamma(n,t),2*pi);
                if (PNgamma(n,t) >= 0 && PNgamma(n,t) <= psi)
                    PNalpha(n,t) = PNgamma(n,t);
                elseif (PNgamma(n,t) > psi && PNgamma(n,t) < 2*pi)
                    PNalpha(n,t) = PNgamma(n,t) - 2*pi;
                end
                %Find u, Control force
                %Angle
                if (Fil_Dist_BN(n,t) == 0)
                    PNtheta(n,t) = PNtheta(n,(t-1)) + 0;
                else
                    PNtheta(n,t) = PNtheta(n,(t-1)) + kE*gain(n,t)*PNalpha(n,t);
                end
            % end


        else
            %Update Positions
            PNposx(n,t) = PNposx(n,(t-1)) ;
            PNposy(n,t) = PNposy(n,(t-1)) ;
            PNtheta(n,t) = PNtheta(n,(t-1));
            PNvel(n,t) = PNvel(n,(t-1));
            PNgamma(n,t) = PNgamma(n,(t-1));
            PNalpha(n,t) = PNalpha(n,(t-1));
         end
    end
    if(nodes_moving<nodes && count>2)
        count = 0;
        nodes_moving = nodes_moving + 1;
    else
        count = count + 1;
    end

end

%Plot Final location of Nodes
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'rx','LineWidth',2);
end

figure(2);
for t=1:time
    plot1(t)=Force((nodes-4),t);
    plot2(t)=Force((nodes-3),t);
    plot3(t)=Force((nodes-2),t);
    plot4(t)=Force((nodes-1),t);
    plot5(t)=Force(nodes,t);
end
plot(plot1);
hold on;
plot(plot2,'r');
plot(plot3,'c');
plot(plot4,'k');
plot(plot5,'g');
ylabel('Force');xlabel('Time(s)');
```

```
legend('PN5','PN6','PN7','PN8','PN9','Location','SouthOutside','Orientation','horizontal');

figure(3);
hold on;
for t=1:time
    plot(PNposx((nodes-4),t),PNposy((nodes-4),t),'LineWidth',1);
    plot(PNposx((nodes-3),t),PNposy((nodes-3),t),'r','LineWidth',1);
    plot(PNposx((nodes-2),t),PNposy((nodes-2),t),'c','LineWidth',1);
    plot(PNposx((nodes-1),t),PNposy((nodes-1),t),'k','LineWidth',1);
    plot(PNposx((nodes),t),PNposy((nodes),t),'g','LineWidth',1);
end

plot(0,0,MAXx,MAXy)
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'rx','LineWidth',2);
end
plot(BNx,BNy,'b+');
ylabel('y pos (m)');xlabel('x pos (m)');
legend('PN5','PN6','PN7','PN8','PN9','Location','SouthOutside','Orientation','horizontal');
```

A.2 Gradient Decsent Software

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Gradient Decsent Software%%%%%%%%
%%%%       JSC 6/2/07      %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all; clear all; clc;
%Length of Simulation
time = 800;
%Number of Perimeter Nodes
nodes = 9;
%Perimeter Radius in Meters
radius = 30;
%Max speed of a Node m/s
vel = 1;
%Transmission Frequency
freq= 2400000000;%Hz = 2.4GHz
%Transmission Strenght
Tstrength = 18;%dB
%Window Size of filter
window = 4;
%Window Gains
for x = 1:window
    WGains(x) = 2^(window - x);
end

count = 0;
%Position of central beacon
BNx = 1.5*radius;
BNy = 1.5*radius;
%Plot Size
MAXx = 2*BNx;
MAXy = 2*BNy;
```

```
%Starting Position and Orientation for all perimeter nodes
%Start them all at the center, if this is changed the initial starting
%distance between nodes in the first RSSI section needs to be changed
t=1;
for n=1:nodes
    PNposx(n,t) = BNx;
    PNposy(n,t) = BNy;
    PNtheta(n,t) = 2*pi*rand(1);
    PNvel(n,t) = vel;
    PNint(n) = 0;
end

%Desired Final Spacing Between Nodes
%All PNs want to be the dist of the Radius away from BN
for n=2:nodes
    for d=1:(n-1)
        desir_dist(n,d)= sqrt(2*radius^2-2*radius^2*cos(2*(n-d)*pi/nodes));
        gain(n,d) = 2*(nodes-(n-1))/(radius/desir_dist(n,d));
    end
end

%Plot Starting Positions of Nodes
figure(1);
plot(0,0,MAXx,MAXy)
ylabel('y axis(m)');xlabel('x axis(m)');
hold on;
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'o');
end
plot(BNx,BNy,'r+');

%Current Actual Distance Between Nodes
for n=1:nodes
    for d=1:nodes
        Act_Dist(n,d,t) = sqrt((PNposx(n,t)-PNposx(d,t))^2+(PNposy(n,t)-PNposy(d,t))^2);
    end
end

%Converting Actual Distance to RSSI, Making it Noisy
%Dont use actual distance here because they all start at zero and that
%causes error, will need to change if they dont all start at the same spot
for n=1:nodes
    for d=1:nodes
        RSSI(n,d,t) = Tstrength - 20*log10(1)+20*log10(freq)-147.55;
        RSSI_noisy(n,d,t) = RSSI(n,d,t) + 2 - (rand(1)) -(rand(1))-(rand(1))-(rand(1));
%         RSSI_noisy(n,d,t) = RSSI(n,d,t);
    end
end

%Calculating Distance from RSSI (No Filter)
for n=1:nodes
    for d=1:nodes
        Calc_Dist(n,d,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20 - RSSI_noisy(n,d,t)/20);
    end
end
```

```
%Calculate Current Field Felt by each Node
for n=1:nodes
    Force(n,t)= abs(sqrt((PNposx(n,t)-BNx)^2+(PNposy(n,t)-BNy)^2)-radius)^3;
    for d=1:(n-1)
        Force(n,t) = Force(n,t)+ gain(n,d)*(Act_Dist(n,d,t)- desir_dist(n,d))^2;
    end
end

%Simulate Vehicle Movement
nodes_moving = 1;
for t=2:time
     for n=1:nodes
         if n<=nodes_moving
            %Update Positions
            PNposx(n,t) = PNposx(n,(t-1)) + PNvel(n,(t-1))*cos(PNtheta(n,(t-1)));
            PNposy(n,t) = PNposy(n,(t-1)) + PNvel(n,(t-1))*sin(PNtheta(n,(t-1)));
            plot(PNposx(n,t),PNposy(n,t));

            %Update Actual Distance Between Nodes
            for d=1:(n-1)
                Act_Dist(n,d,t) = sqrt((PNposx(n,t)-PNposx(d,t))^2+(PNposy(n,t)-PNposy(d,t))^2);
            end

            %Converting Actual Distance to RSSI, Making it Noisy
            for d=1:(n-1)
                RSSI(n,d,t) = Tstrength - 20*log10(Act_Dist(n,d,t))+20*log10(freq)-147.55;
                RSSI_noisy(n,d,t) = RSSI(n,d,t) + 2 - (rand(1)) -(rand(1))-(rand(1))-(rand(1));
%                 RSSI_noisy(n,d,t) = RSSI(n,d,t);
            end

            %Calculating Distance from RSSI (Then Filtered)
            for d=1:(n-1)
                Calc_Dist(n,d,t) = 10^(Tstrength/20 + log10(freq) - 147.55/20 - RSSI_noisy(n,d,t)/20);

                if t >=window
                    tmax = window;
                elseif t < window
                    tmax = t;
                end

                f = t;
                fil = 0;
                Gsum = 0;
                for x = 1:tmax
                    fil = fil + Calc_Dist(n,d,f)*WGains(x);
                    Gsum = Gsum + WGains(x);
                    f= f - 1;
                end
                Fil_Dist(n,d,t) = fil/Gsum;
            end

            %Calculate Current Field Felt by each Node
            Force(n,t)= abs(sqrt((PNposx(n,t)-BNx)^2+(PNposy(n,t)-BNy)^2)-radius)^3;
            for d=1:(n-1)
                Force(n,t) = Force(n,t)+ gain(n,d)*(Fil_Dist(n,d,t)- desir_dist(n,d))^2;
            end
            %Control inputs of velocity and direction
            if (Force(n,t) > 0.01)
                %velocity
                PNvel(n,t) = vel*Force(n,t)/n;
                if PNvel(n,t)> vel
                    PNvel(n,t) = vel;
                end

                %Theta
                if PNint(n) == 0
                    if Force(n,t)<Force(n,(t-1))
                        PNtheta(n,t) = PNtheta(n,(t-1));
                    else
                        PNtheta(n,t) = PNtheta(n,(t-1)) + pi/2;
                        PNint(n) = 1;
                    end
                elseif PNint(n) == 1
                    if (Force(n,t)-Force(n,(t-1)))>0
                        PNtheta(n,t) = PNtheta(n,(t-1)) + pi - acos((Force(n,t)-Force(n,(t-1)))
                        /sqrt((Force(n,t)-Force(n,(t-1)))^2+(Force(n,(t-1))-Force(n,(t-2)))^2));
                    else
                        PNtheta(n,t) = PNtheta(n,(t-1)) + acos((Force(n,t)-Force(n,(t-1)))
                        /sqrt((Force(n,t)-Force(n,(t-1)))^2+(Force(n,(t-1))-Force(n,(t-2)))^2));
                    end
                    PNint(n) = 0;
                end
            else
                PNtheta(n,t) = 0;
                PNvel(n,t) = 0;
            end
         else
            %Update Positions
            PNposx(n,t) = PNposx(n,(t-1)) ;
            PNposy(n,t) = PNposy(n,(t-1)) ;
            PNtheta(n,t) = PNtheta(n,(t-1));
         end
     end
    if(nodes_moving<nodes && count>30)
        count = 0;
        nodes_moving = nodes_moving + 1;
    else
        count = count + 1;
    end
end

%Plot Final location of Nodes
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'rx','LineWidth',2);
end

figure(2);
for t=1:time
    plot1(t)=Force((nodes-4),t);
    plot2(t)=Force((nodes-3),t);
    plot3(t)=Force((nodes-2),t);
    plot4(t)=Force((nodes-1),t);
    plot5(t)=Force(nodes,t);
end
plot(plot1);
hold on;
plot(plot2,'r');
plot(plot3,'c');
plot(plot4,'k');
plot(plot5,'g');
ylabel('Force');xlabel('Time(s)');
legend('PN5','PN6','PN7','PN8','PN9','Location','SouthOutside','Orientation','horizontal');

figure(3);
hold on;
for t=1:time
    plot(PNposx((nodes-4),t),PNposy((nodes-4),t),'LineWidth',1);
    plot(PNposx((nodes-3),t),PNposy((nodes-3),t),'r','LineWidth',1);
    plot(PNposx((nodes-2),t),PNposy((nodes-2),t),'c','LineWidth',1);
    plot(PNposx((nodes-1),t),PNposy((nodes-1),t),'k','LineWidth',1);
    plot(PNposx((nodes),t),PNposy((nodes),t),'g','LineWidth',1);
end

plot(0,0,MAXx,MAXy)
for n=1:nodes
    plot(PNposx(n,t),PNposy(n,t),'rx','LineWidth',2);
end
plot(BNx,BNy,'b+');
ylabel('y pos (m)');xlabel('x pos (m)');
legend('PN5','PN6','PN7','PN8','PN9','Location','SouthOutside','Orientation','horizontal');
```

Appendix B

RSSI Measurements

RSSI Measurements(dBM)			
	Measured Distance		
Sample	25m	50m	100m
1	49	56	60
2	48	55	59
3	49	56	60
4	48	54	60
5	49	56	60
6	48	56	59
7	48	53	60
8	48	54	60
9	49	53	59
10	49	56	60
11	49	55	60
12	48	55	59
13	50	55	60
14	48	55	60
15	48	54	60
16	48	53	60
17	49	52	60
18	48	53	60
19	49	53	60
20	49	53	60
21	48	53	60
22	49	54	60
23	48	54	60
24	48	57	60
25	50	55	61
26	50	54	60
27	50	54	60
28	51	54	60
29	51	55	60
30	51	52	60
average(dBm)	-48.9	-54.3	-59.9
expected(dBm)	-50.0	-56.0	-56.0
% Dif.	2.2%	3.1%	6.9%
St. Dev.	0.995	1.291	0.403

Table B.1: RSSI Readings Modeling Measurement Noise

RSSI Measurements(dBM) at 25m			
Sample	RSSI(dBm)	Sample	RSSI(dBm)
1	56	41	55
2	55	42	56
3	54	43	56
4	54	44	56
5	55	45	56
6	55	46	54
7	55	47	54
8	55	48	56
9	55	49	55
10	55	50	55
11	55	51	54
12	55	52	55
13	54	53	54
14	53	54	55
15	53	55	54
16	54	56	55
17	54	57	55
18	54	58	56
19	54	59	55
20	53	60	56
21	54	61	56
22	55	62	56
23	54	63	56
24	54	64	55
25	54	65	54
Continued...			

...Continued			
Sample	RSSI(dBm)	Sample	RSSI(dBm)
26	56	66	55
27	57	67	54
28	56	68	54
29	56	69	55
30	55	70	54
31	56	71	55
32	56	72	54
33	56	73	55
34	56	74	54
35	55	75	55
36	57		
37	55		
38	55		
39	57		
40	56		
St. Dev.	0.9296		

Table B.2: RSSI Readings Modeling Measurement Noise Redone at 25m

RSSI vs Distance				
	RSSI Measurements (dBm)			
Distance(m)	Sample 1	Sample 2	Sample 3	Sample 4
1	36	36	36	36
2	36	36	36	36
3	36	36	36	36
4	36	39	36	36
5	38	36	36	36
6	37	41	39	45
7	39	37	38	41
8	39	37	38	42
9	43	44	43	42
10	40	40	42	40
11	40	40	42	40
12	42	40	41	52
13	47	42	41	43
14	49	41	43	45
15	44	43	41	42
16	48	50	52	51
17	50	48	53	50
18	65	47	48	54
19	60	53	53	49
20	51	53	49	61
22	45	46	45	46
24	49	46	46	47
26	46	46	46	46
28	45	45	45	45
30	49	46	45	45
32	48	49	48	48
34	47	47	47	48
36	47	47	47	47
38	51	52	52	53
40	54	54	54	53
42	55	54	54	55
44	54	55	53	53
46	51	53	52	51
48	59	60	60	60
50	61	63	62	60
Continued...				

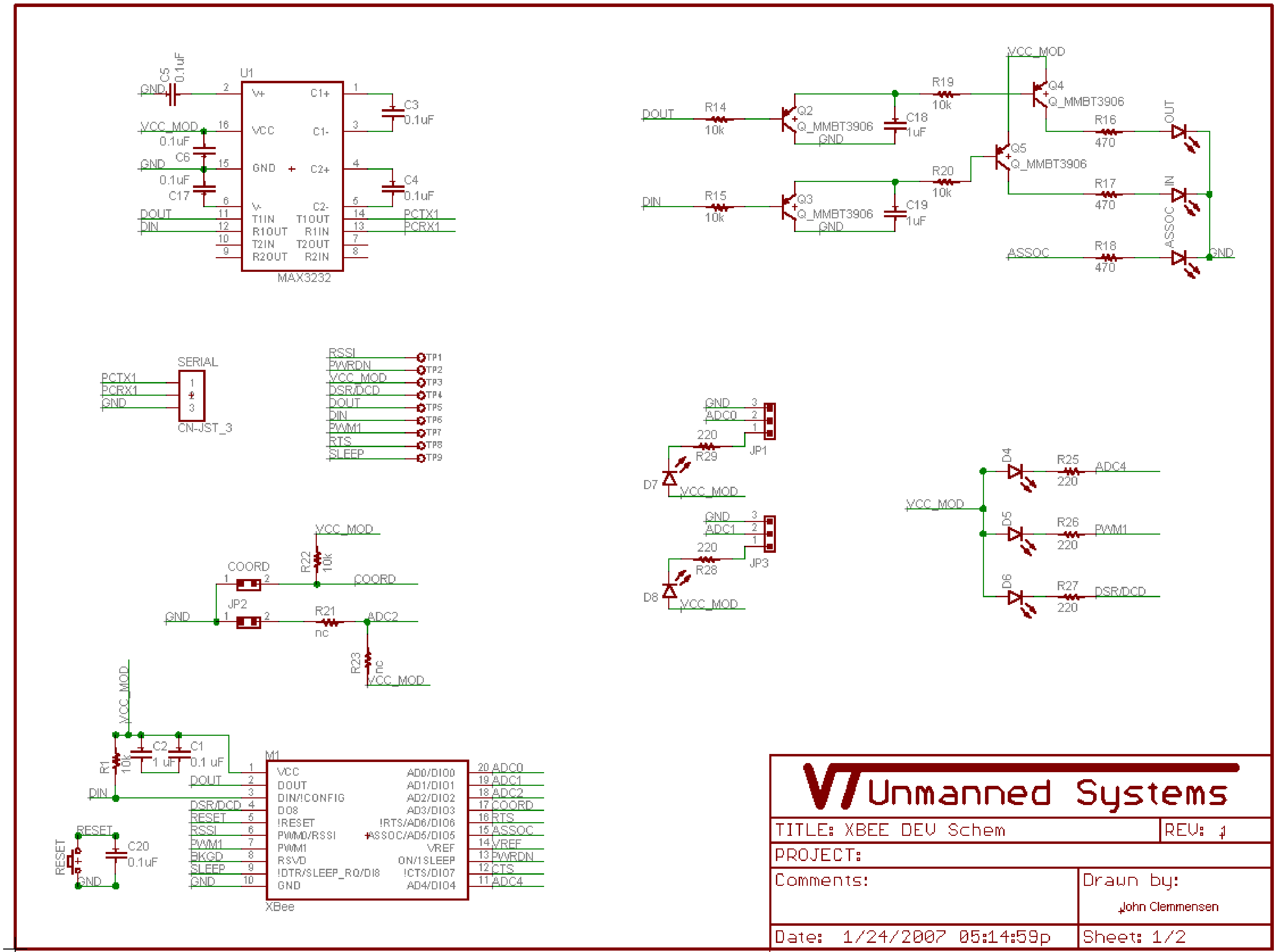
...Continued				
	RSSI Measurements (dBm)			
Distance(m)	Sample 1	Sample 2	Sample 3	Sample 4
55	53	53	52	52
60	63	60	62	63
65	53	53	53	53
70	56	56	56	56
75	60	60	60	60
80	68	69	69	69
85	67	66	65	68
90	62	62	64	65
95	62	65	65	63
100	61	62	60	61
110	64	63	65	63
120	63	65	64	63
130	69	70	69	68
140	67	68	66	66
150	69	68	69	69

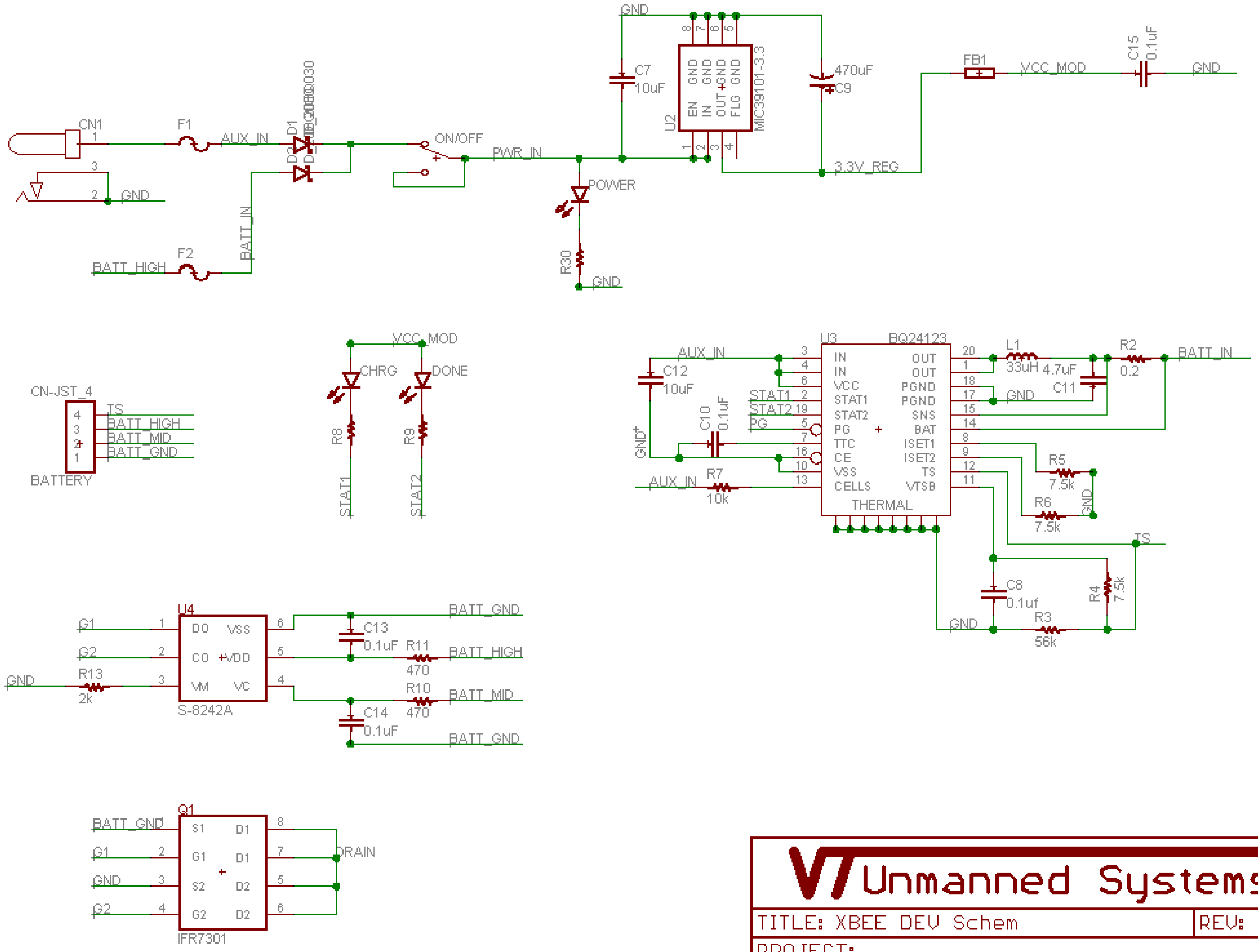
Table B.3: RSSI Readings Modeling Signal Strength Loss vs Distance

Appendix C

Schematic

The schematics for the board supporting the XBee Pro module are shown on the following two pages, the third page is the board layout.


U1
MAX3232
V+
VCC
GND
V-
T1IN
R1OUT
T2IN
R2OUT
C1+
C1-
C2+
C2-
T1OUT
R1IN
T2OUT
R2IN
C5 0.1uF
C6 0.1uF
C17 0.1uF
C3 0.1uF
C4 0.1uF
VCC_MOD
GND
DOUT
DIN
PCTX1
PCRX1
SERIAL
CN-JST_3
RSSI
PWRDN
VCC_MOD
DSR/DCD
DOUT
DIN
PWM1
RTS
SLEEP
TP1
TP2
TP3
TP4
TP5
TP6
TP7
TP8
TP9
COORD
JP2
R22 10k
R21 nc
R23 nc
ADC2
M1
XBee
VCC
DOUT
DIN/!CONFIG
DO8
!RESET
PWM0/RSSI
PWM1
RSVD
!DTR/SLEEP_RQ/DI8
GND
AD0/DIO0
AD1/DIO1
AD2/DIO2
AD3/DIO3
!RTS/AD6/DIO6
ASSOC/AD5/DIO5
VREF
ON/!SLEEP
!CTS/DIO7
AD4/DIO4
20 ADC0
19 ADC1
18 ADC2
17 COORD
16 RTS
15 ASSOC
14 VREF
13 PWRDN
12 CTS
11 ADC4
DSR/DCD
RESET
RSSI
PWM1
BKGD
SLEEP
C1 0.1 uF
C2 1 uF
R1 10k
C20 0.1uF
RESET
R14 10k
R15 10k
Q2 Q_MMBT3906
Q3 Q_MMBT3906
C18 1uF
C19 1uF
R19 10k
R20 10k
Q4 Q_MMBT3906
Q5 Q_MMBT3906
R16 470
R17 470
R18 470
OUT
IN
ASSOC
JP1
JP3
R29 220
R28 220
D7
D8
ADC0
ADC1
D4
D5
D6
R25 220
R26 220
R27 220
ADC4
PWM1
DSR/DCD
VT Unmanned Systems
TITLE: XBEE DEV Schem
REV: 1
PROJECT:
Comments:
Drawn by:
John Clemmensen
Date: 1/24/2007 05:14:59p
Sheet: 1/2



VT Unmanned Systems

TITLE: XBEE DEV Schem | REV: 1

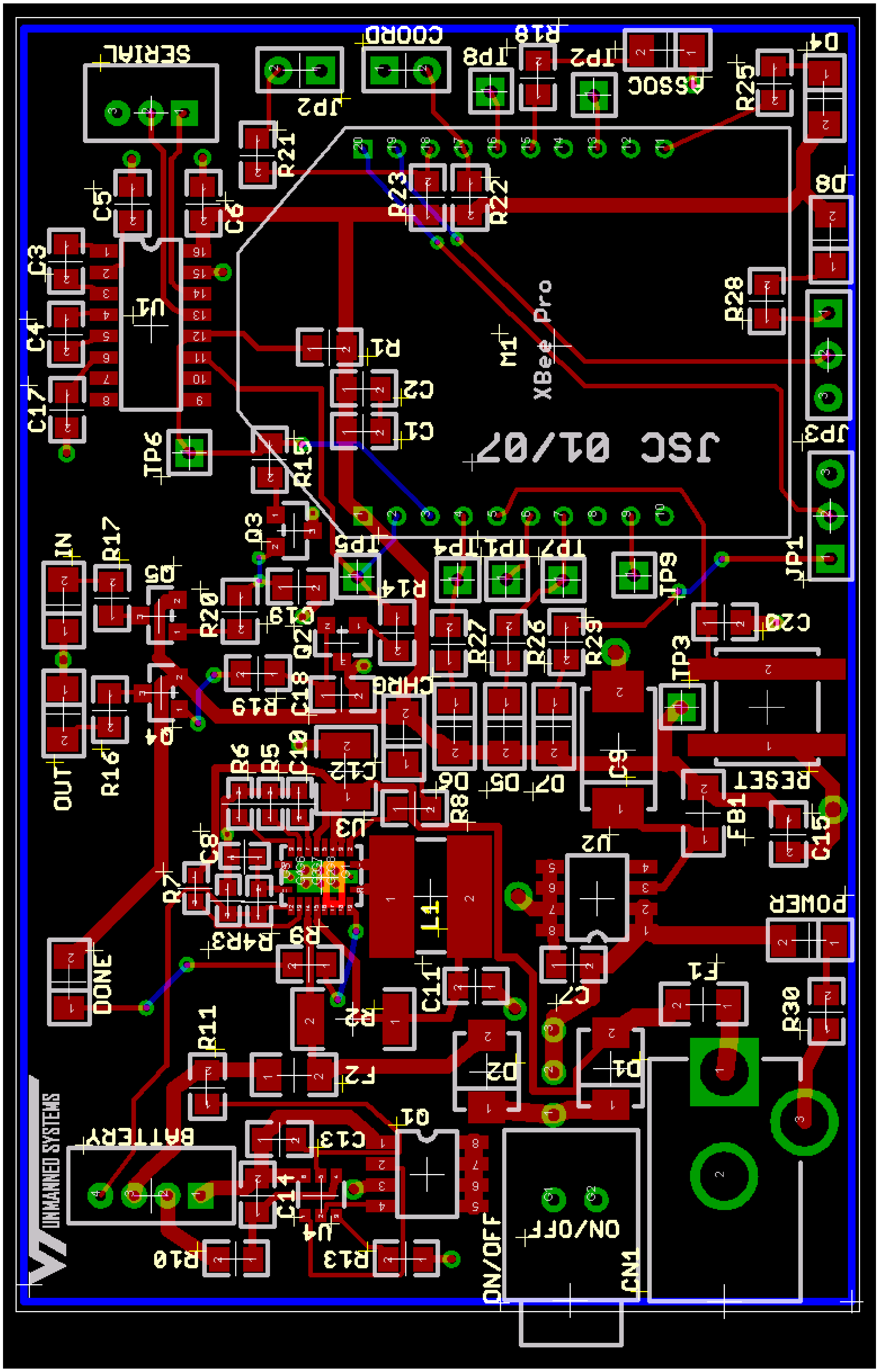
PROJECT:

Comments:

Drawn by: John Clemmensen

Date: 1/24/2007 05:14:59p

Sheet: 2/2



SERIAL
COORD
ASSOC
XBee Pro
M1
JSC 01/07
IN
OUT
DONE
BATTERY
ON/OFF
CN1
POWER
RESET
CHRG
UNMANNED SYSTEMS