Autonomous Navigation of a Ground Vehicle to Optimize Communication Link Quality

Cheryl Lynn Bauman

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
Mechanical Engineering

Dr. Charles F. Reinholtz, Chairman
Alumni Distinguished Professor of Mechanical
Engineering and Engineering Education

Dr. Alfred L. Wicks, Co-Chairman
Associate Professor of Mechanical Engineering

Dr. Dennis W. Hong
Assistant Professor of Mechanical Engineering

December 8, 2006
Blacksburg, Virginia

Keywords: autonomous vehicle teaming, communication-sensitive navigation, mobile ad-hoc network, multi-robot

Autonomous Navigation of a Ground Vehicle to Optimize Communication Link Quality

Cheryl Lynn Bauman

ABSTRACT

The wireless technology of today provides combat systems with the potential to communicate mission critical data to every asset involved in the operation. In such a dynamic environment, the network must be able maintain communication by adapting to subsystems moving relative to each other. A theoretical and experimental foundation is developed that allows an autonomous ground vehicle to serve as an adaptive communication node in a larger network. The vehicle may perform other functions, but its primary role is to constantly reposition itself to maintain optimal link quality for network communication. Experimentation with existing wireless network hardware and software led to the development, implementation, and analysis of two main concepts that provided a signal optimization solution. The first attracts the communication ground vehicle to the network subsystems with weaker links using a vector summation of the signal-to-noise ratio and network subsystem position. This concept continuously generates a desired waypoint for repositioning the ground vehicle. The second concept uses a-priori GIS data to evaluate the desired vehicle waypoint determined by the vector sum. The GIS data is used primarily for evaluating the viewshed, or line-of-sight, between two network subsystems using elevation data. However, infrastructure and ground cover data are also considered in navigation planning. Both concepts prove to be powerful tools for effective autonomous repositioning for maximizing the communication link quality.

Acknowledgments

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

I would like to express my deepest gratitude to Dr. Charles Reinholtz and Dr. Alfred Wicks. Their advice and encouragement have been invaluable throughout my studies. They have compiled a group of amazing students who continue to work together and have fun advancing autonomous technologies. Brett Leedy was also the best graduate mentor, team leader, and friend that I could have asked for. Without these three individuals, I would not have pursued my graduate degree.

I also have to thank the 2005 Virginia Tech Grand Challenge team. There was never a more devoted and enjoyable senior design team. Being a part of this team led to my continued interest in autonomous vehicles. More specifically, thanks to Andrew Bacha as a graduate mentor and software guru. Thanks to Joe Putney and Mike Webster for all the long hours debugging code and testing in the field, as fun and painful as they were. And finally, thanks to Dave Anderson for his work on the base vehicle and box building.

Next, I have to thank David Van Covern, Ben Dingus, and Tom Alberi. These highly motivated individuals made up the Signal Optimization team. Their hard work provided the needed support for researching, developing, and testing the ideas presented in this thesis.

Finally, thanks to Dr. Dennis Hong who offered guidance and support in my autonomous system research. Thanks to Chris Terwelp and Mike Fleming for their leadership and support of all the projects at the Unmanned Systems Lab. And, thanks to all of my fellow graduate students who have helped create a productive and enjoyable unmanned systems family at Virginia Tech.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

A critical component of future combat systems is effective networking and communication. As vehicles and personnel are on the move, they must communicate with the other agents to carry out their duties. Unfortunately, range and bandwidth limitations frequently restrict the ability of one agent to effectively communicate with another. This presents the opportunity to implement strategies for optimizing the configuration of agents based on communication requirements. It also offers the chance to introduce additional agents to serves as communication subsystem node in the network. Unmanned vehicles can provide a solution by automatically adapting to dynamic network conditions. Used as a mobile communication subsystem in a mobile ad-hoc network (MANET), a ground vehicle can reposition itself autonomously to maintain the link quality between the other subsystems in a network. Figure 1.1 is a conceptual drawing showing an assortment of possible agents, or subsystems. For an autonomous communication subsystem to be successful, behaviors and algorithms must be developed to fuse autonomous navigation with real-time signal-to-noise ratio analysis and Geographic Information System (GIS) databases.

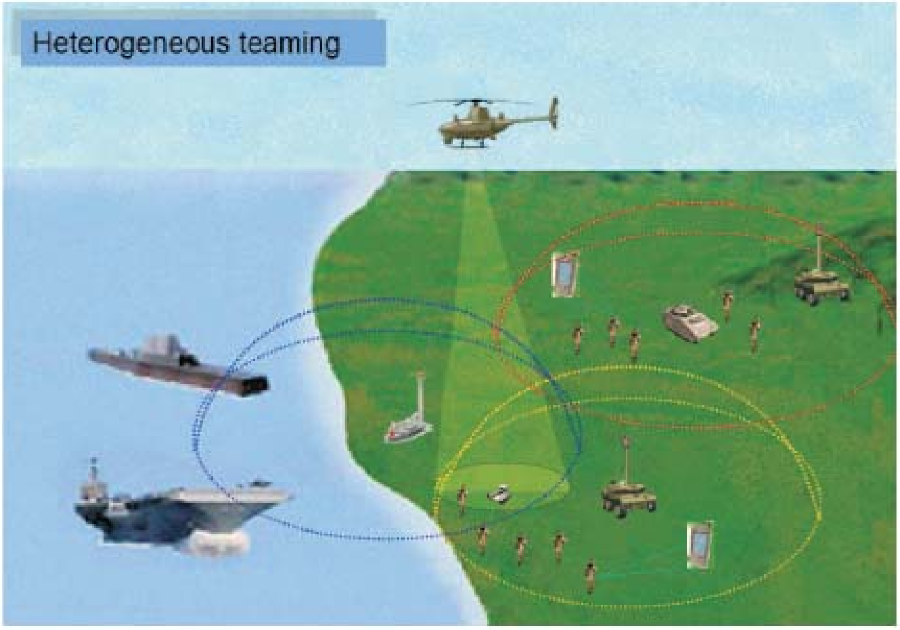


Figure 1.1: Heterogeneous Teaming Example [19]

Funded by the Office of Naval Research (ONR), this signal optimization research is part of a larger autonomous systems research project proposed by the Virginia Center for Autonomous Systems (VaCAS). The project goal is to advance sensing and control for heterogeneous teams of autonomous vehicles. Some of the sub-projects include research in perception with acoustic sensing and computer vision. Others deal with control and coordination for adaptive sampling, vision-enabled control, and adaptive path following [19].

1.2 Thesis Overview

After reviewing related work in signal optimization, the first goal of this research project was to demonstrate the ability to conduct an autonomous site survey. This involved creating a signal-to-noise ratio map of a test field and then returning to the point with the best communication with the other subsystems in the network. Completing the site survey of a static network environment provided the quickest way to get involved with most aspects required for signal optimization. This experiment served as an initial milestone and learning experience.

The site surveys occurred in two main steps. The first was using existing software, Ekahau [4], that was developed specifically to perform site surveys for monitoring and troubleshooting 802.11 networks. The hope was that Ekahau would provide a partial solution to the problem. However, there was no easy way to output raw data from the Ekahau software package for real-time processing. Autonomous site surveys were performed, but the vehicle was not able to reposition itself for optimal communication. Much was learned in the process, but another alternative had to be found.

The next step for the site survey was to use NetStumbler [14]. This is another tool developed for 802.11 monitoring and troubleshooting. NetStumbler allowed signal-to-noise ratio values to be input into custom developed mapping and optimization software in real-time. Using this setup, an autonomous site survey was performed and the vehicle repositioned itself for optimal communication in a static network environment. However, a three to five second lag in the signal-to-noise ratio acquisition system was discovered. After attempts to minimize the lag were unsuccessful, a new solution was required. The ITT wireless mesh card [9] minimized signal-to-noise ratio lag, provided a complete mobile ad-hoc network solution, and could be used for real-time processing.

The final phase of development and experimentation was a signal optimization algorithm. The algorithm evolved in two parts. The first developed and tested an algorithm

for navigating the improve signal-to-noise ratio. Using an attraction to weaker signal-to-noise ratio values and the positions of the other network subsystem, vector addition was used to reposition the communication subsystem. The second part investigated the ability to process GIS data for assisting with signal optimization. Software was written to determine line-of-sight between two vehicles, maintain the desired vehicle boundary, and avoid difficult-to-traverse areas were developed. Using these two concepts, a ground vehicle successfully reposition itself autonomously to optimize communication line quality. Finally, recommendations for future work are presented.

Chapter 2

Literature Review

2.1 Mobile Ad-hoc Networks

A mobile ad-hoc network (MANET) is a self-configuring, wireless network of mobile nodes that form an arbitrary topology [12]. Message are routed to their destination by hopping between network nodes on an optimal path through the current topology. As the nodes continue to move, communication links may be lost causing the topology of the wireless network to change suddenly. Routing tables must be able to keep up with the changing network topology to ensure the best path for the message. A mobile ad-hoc network can operate both as an independent network as subsection of a larger network.

2.2 Signal-to-Noise Ratio

There are many metrics by which to evaluate communication link quality, such as data rate and signal-to-noise ratio (SNR). In the end, data rate is the most important factor because communication is only as good as the information that is reaching its goal. However, the data rate is directly proportional to the signal-to-noise ratio and is dependent on modulation techniques of the 802.11 family. The technique of 802.11b provide four data rate increments of 1, 2, 5.5, or 11 Mbps. On the other hand, 802.11g has eleven data rate increments of 1, 2, 5.5, 6, 9, 12 18, 24, 36, 48, or 54 Mbps. For both 802.11 b and g, the data rate increments cover the entire signal-to-noise ratio range of 0 to 100 dB. Therefore, 11 Mbps is a signal-to-noise ratio greater than 10 dB for 802.11b [16] and a signal-to-noise ratio of about 12 dB for 802.11g [1]. The signal-to-noise ratio metric was chosen to ignore the differences between the 802.11a/b/g and to provide better resolution, 0 to 100 dB instead of the few data rate increments, for navigation algorithm decisions.

An initial thought for evaluating communication was to find a way to estimate the signal-to-noise ratio. The signal-to-noise ratio (in dB) is signal level (in dBm) minus the noise level (in dBm). However after research, this task proved to be very difficult. There are too many variables to consider, including distance, transmit power, and antenna gains. Along with these variables, there are many other factors involved, such as interference, multipathing, and line-of-sight. Interference is the presence of unwanted radio frequency (RF) signals that disturb the normal operations of a system. Standard 802.11 networks operate at 2.4 GHz along with many other commonly used devices, including microwave ovens, wireless phones, bluetooth devices, and other WLANs (Wireless Local Area Networks) [5]. Multipathing can also occur when an RF signal takes a non-direct path from its source to its destination. If an object, such as a building, gets in the way, some of the signal may bounce off the building creating a longer path and confusion for the receiver [6]. Though interesting topics, this research focuses on developing a navigation algorithm to improve the communication quality. Due to the difficultly in predicting signal strength, it was decided that reacting to current signal-to-noise ratio value and considering line-of-sight between network subsystems was the best solution for optimizing the communication link.

2.3 Autonomous Communication Behaviors

2.3.1 MARS 2020 Demonstration

The MARS 2020 Demonstration is a Defense Advanced Research Projects Agency (DARPA) sponsored event to demonstrate Mobile Autonomous Robot Software. Unmanned aerial and ground vehicle teams are given missions that must include communication sensitive planning and behaviors. The vehicles must operate in urban environments with unreliable or unavailable GPS. The teams of vehicles cooperate to achieve a single mission by each vehicle completing individual tasks while considering communication strength with the others. For the research described in this thesis, the signal optimization vehicle will be operating in a fairly open field with GPS and will act solely as the communication subsystem for the entire network. Though the MARS 2020 teaming exercise has a slightly different end goal, behaviors and navigation strategies can be similar. The paper summaries below focus only on the behaviors of the communication aspect developed for the MARS 2020 demonstration. Based on the literature, creating a signal strength map is extremely helpful. However, creating such a map is time consuming and is unrealistic for a real-time system operating in a dynamic network environment. Reactive behaviors to assist when network subsystems are lost seem to be the most useful concept from the MARS 2020 demonstration that can be applied to this signal optimization research.

Deploying Air-Ground Multirobot Teams in Urban Environments

The robot teams in this research use a three-phase process [2]. First, exploration waypoints are determined for each robot using an overhead surveillance map. The points, represented as red circles in Figure 2.1, are chosen in such a way to assist in predicting line-of-sight propagation characteristics in the environment. Second, the robots navigate to those waypoints to obtain signal strength measurements and create a radio connectivity map. This map, shown below, is a function that returns the signal-to-noise ratio between two points. The blue lines show where the signal strength measurements were collected. Also displayed are the corresponding signal strength values that appear next to each of the lines. The third phase uses the prediction to increase the probability of a reliable communication network during mission execution. Future work includes eliminating the need to predetermine exploration waypoints and to incorporate the signal strength map creation into other tasks or mission goals.

Figure 2.1: “Preliminary experimental radio connectivity map for the Fort Benning MOUT site” [2]

Internalized Plan for Communication-Sensitive Robot Team Behaviors

This research developed a hybrid deliberative/reactive architecture to address the problem of communication link optimization [20]. It implements reactive control with a-priori map knowledge. Prior to the run, an experimenter visually selects areas such as alleys and parked cars that will likely attenuate communication. Once this knowledge is stored, the vehicles set off to complete their reconnaissance mission. The goal for future research is to develop a selection mechanism based on automated terrain analysis. The algorithm was evaluated based on connection time, area coverage, and mission time. The research was successful in demonstrating communication sensitive behaviors. Based on these metrics, the experiments resulted in a trade-off between communication and area coverage.

When Good Comms Go Bad

This paper describes four reactive behaviors for communication recovery [17]. The two general-purpose behaviors are Retrotraverse and Move-to-Nearest-Neighbor. Retrotraverse provides an attraction to past waypoints when communications is lost. Move-to-Nearest-Neighbor is an attraction to the last known position of the closest teammate. Both of these behaviors move the vehicle back to where it last had communication with the network. The other two behaviors, Probe and Move-to-Higher-Ground, are context-specific. Probe draws the vehicle to nearby open spaces, and Move-to-Higher-Ground moves the vehicle to nearby inclines. These behaviors position the vehicle in areas that are more likely to have line-of-sight with the other vehicles in the team.

Various experiments were run with different combinations of behaviors and robot responsibilities. The evaluation was based on mission success rate, area covered, and communication recovery time. In general, the most successful behavior was Move-to-Nearest-Neighbor. Retrotraverse often produced oscillatory behavior around a communication obstruction.

Chapter 3

Base Vehicle

3.1 DARPA Grand Challenge

This chapter describes the base vehicle systems that were used to conduct this research. Both vehicles were originally developed to participate in the DARPA Grand Challenge. It was motivated, in part, by a congressional mandate that one-third of the operational ground combat vehicles must be unmanned by 2015 [3]. The goal of the competition was to spark interest and motivation in unmanned technology. The Grand Challenge consisted of a 150 mile, off-road race through the Mojave Desert. The vehicles were required to follow a series of GPS waypoints, and they encountered obstacles, including k-barriers, tank traps, and desert vegetation. No one won the 2004 competition, but it was successful in generating momentum in unmanned research and development. The 2005 competition brought back many teams determined to win the race and the two million dollar prize. Virginia Tech was successful in qualifying two vehicles in the final event, Cliff and Rocky.

Unfortunately, both vehicles came to a stop approximately forty miles into the race due to mechanical failures. Twenty-three teams qualified for the final event; Cliff placed eighth, and Rocky placed ninth. This chapter provides a brief overview of the vehicles and why they were was chosen for the signal optimization research.

3.2 Virginia Tech Grand Challenge 2005

Virginia Tech entered two vehicles, Cliff and Rocky (Figure 3.1), into the 2005 Grand Challenge [10]. Originally, the nearly identical base platforms were being developed to run two different navigation algorithms. The goal of the team was to compare reactive and deliberative navigation approaches. At competition, the reactive algorithm was chosen because it was more reliable when presented with new, untested situations.

Figure 3.1: Virginia Tech Grand Challenge Vehicles Cliff (right) and Rocky (left)

3.2.1 Base Platform

The base platform is a Ingersoll-Rand Club Car XRT1500. This utility vehicle is extremely rugged and maneuverable. It also has an ample payload required for autonomous conversion equipment. These vehicles provided a great platform to develop and gain experience with unmanned systems.

3.2.2 Drive-By-Wire Conversion

Both vehicles were modified with steering, brake, and throttle actuators to allow the computer to control motion. The steering and throttle are controlled by DC servo motors with quadrature encoder feedback. After removing the steering wheel and column, the steering motor was coupled directly to the shaft of the steering rack and pinion system. Using a pulley, the throttle motor was attached to the throttle cable of the vehicle. Finally, the electric brake was actuated using a hydraulic pump. The stock emergency brake was modified to be used as a fail-safe, mechanical brake for emergency stops.

3.2.3 Sensor Suite

The two vehicles have slightly different sensor arrangements for the two navigation algorithms. Each arrangement consists of the same basic sensors: Inertial Navigation System (INS), Light Detection And Ranging (LIDAR), and computer vision. The INS, used for localization, is a Novatel Propak LBplus Global Positioning System (GPS) receiver and a Novatel IMU-G2 enclosure that houses a Honeywell HG1700 Inertial Measurement Unit (IMU). Together, this system provides an accuracy of 10 cm CEP (Circular Error Probable) in applications where GPS satellites are generally in view. The LIDAR is a SICK LMS-290 that is used for obstacle detection. It is capable of returning the distances for a 180^o, single plane scan. Computer vision is used for detecting and following roads. Using a Point Grey Bumble Bee stereo vision camera, the center points of the road can be calculated based on the pixel offset between the two camera images.

3.2.4 Navigation Algorithm

Up until the National Qualifying Event (NQE), both the reactive and deliberative algorithms were being developed. However, the reactive navigation strategy, referred to as

Dynamic Expanding Zones (DEZ) [15], was the algorithm used on both vehicles at the NQE and the final event. As opposed to a deliberative approach, reactive navigation responds only to the current environment. It does not predict future paths or remember where it has been in the past. Simple behaviors are combined to produce intelligent navigation. The DEZ algorithm is a combination of the following behaviors: waypoint navigation, obstacles avoidance, and road following. It also incorporates a high-level rollover prevention to ensure that it is does not command unsafe steering angles based on the current and desired speed of the vehicle. After many hours of testing and fine tuning, DEZ provided a reliable, fully autonomous navigation solution.

3.2.5 Motion Control

The motion control software coverts the desired vehicle vector (speed and steering angle), calculated by the navigation algorithm, into motor commands. The throttle percent is determined by a closed-loop PID controller that monitors the desired and current speed of the vehicle. The brake is an open-loop controller that translates the desired speed reduction to the appropriate brake percent. Steering is achieved by converting the desired angle to desired motor counts. The software also include various safety checks, including rollover prevention. Rollover prevention ensures the vehicle will not perform maneuvers that will jeopardize the stability of the vehicle.

3.3 Communication Subsystem

The Virginia Tech Grand Challenge vehicles provide an excellent autonomous platform for the signal optimization research. The vehicle serving as the communication subsystem must be able to traverse rugged terrain. The fully autonomous capability of the Grand Challenge vehicle allows the research to focus on optimizing communication. Changes had to be made in order to integrate the signal optimization algorithm into the existing software architecture. The main change was that the new software needed to accept dynamic waypoints instead of a file of predetermined waypoints.

Chapter 4

Autonomous Site Survey

The initial milestone of the research was to perform an autonomous site survey. The unmanned ground vehicle traversed the entire field collecting signal-to-noise ratio values from two static access points using predetermined file of GPS waypoints similar to the RDDF used in the DARPA Grand Challenge. Once the scan was complete, the vehicle was required to reposition itself at the optimal point in the field for communication with both network subsystems. The main goal of this experiment was to help understand the hardware, software, and inherent problems associated with the chosen wireless system.

4.1 Ekahau Site Survey

Ekahau Site Survey software is a commercially available tool used to quickly and easily analyze 802.11 a/b/g networks [4]. Using GPS data from an external receiver, the software can generate signal-to-noise ratio, data rate, and other maps overlaid on aerial photographs. Though Ekahau is a great wireless tool, this product does not meet the complete needs of this project because the data cannot be output in real-time for external processing. Still, the software provided a starting point for understanding some of the fundamental issues associated with measuring and optimizing communication link quality. This section provides a description of the experiment and results from Ekahau Site Survey.

4.1.1 Hardware Overview

Rocky, one of the Virginia Tech entries to the 2005 DARPA Grand Challenge, was used as the unmanned ground vehicle. This fully autonomous platform is capable of both waypoint navigation and obstacle avoidance in rugged off-road environments [11]. The unmanned ground vehicle autonomous navigation software was provided a predetermined waypoint file for the desired site survey path. The vehicle was also equipped with a Tablet PC, Proxim ORiNOCO gold 802.11b/g wireless PC card, and a 2.4GHz omnidirecitional antenna to handle the wireless network monitoring. Using GPS data from the vehicle's Inertial Navigation System, the Ekahau Site Survey software was used to create an signal-to-noise ratio map of the Mobile Instrumentation Platform (MIP) in an open test field. The MIP, is an 802.11 wireless communication hub and modular platform developed by Virginia Tech researchers to support instrumentation at a test site [7]. The three main hardware components used in the Ekahau site survey are displayed in Figure 4.1.

Figure 4.1: Ekahau Site Survey Hardware. UGV Rocky (left), Tablet PC running Ekahau (center), and MIP (right).

4.1.2 Results

The Ekahau site survey experiment was repeated twice using the same RDDF, but changing the external antenna used for acquiring signal strength. Figure 4.2 shows the signal-to-noise ratio map including vehicle path of the first run that used a 4 dBi omnidirectional antenna. A fine grid resolution was used for the signal-to-noise ratio map. This provides a map that covers only the path of the vehicle during the survey. A 3 dBi omnidirectional antenna was used to generate a signal-to-noise ratio map of the second run illustrated in Figure 4.3. This run used a coarser grid resolution for the map; therefore, it estimates the signal-to-noise ratio values in areas not covered during the survey. Both runs provide a good visual understanding of the autonomous site surveys performed and the data available for post-processing. However, neither of these outputs are useful in real-time repositioning of an autonomous vehicle.

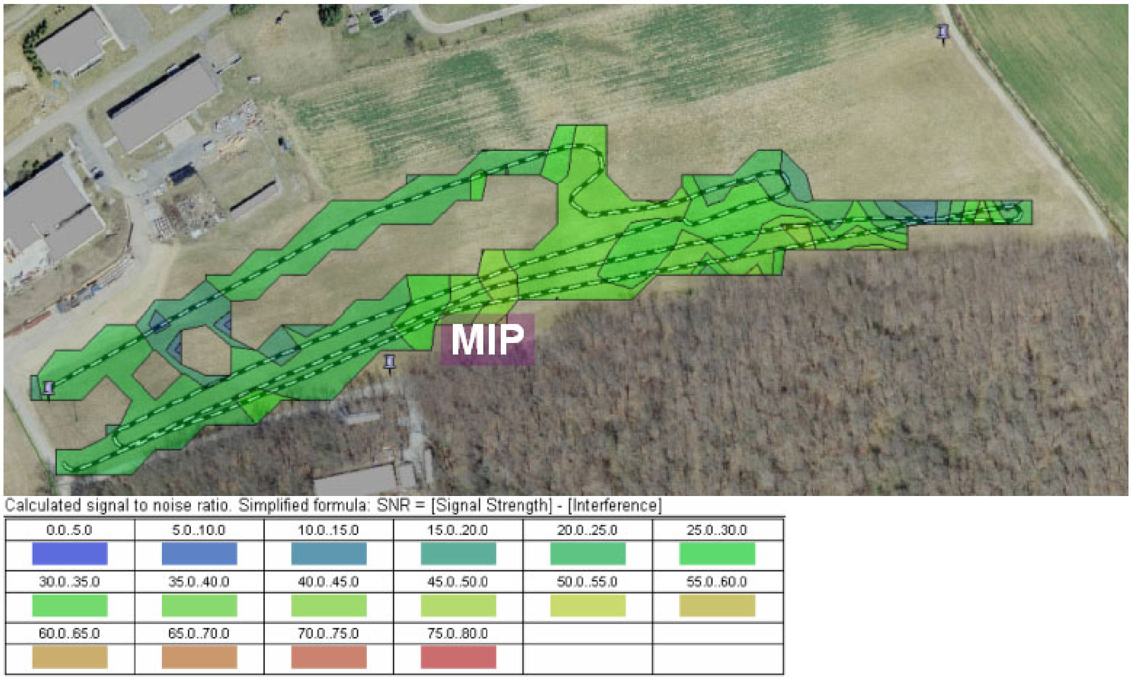


Figure 4.2: SNR Map of Ekahau Site Survey Results (with path of the ground vehicle) using a 4 dBi Omnidirectional Antenna and Fine Grid Resolution.

4.2 NetStumbler Site Survey

NetStumbler Site Survey was the next iteration of the autonomous site survey experiments. Using a combination of freeware and in-house developed software, a signal-to-noise ratio map was created by the autonomous ground vehicle as it traversed the test field in a predefined pattern. The vehicle then repositioned itself in the location that was found to produce optimal communication.

4.2.1 Hardware Overview

Commerical, off-the-shelf 802.11 wireless systems were used to establish the network for the site survey. Rocky was equipped with a Proxim ORiNOCO gold 802.11b/g wireless PC card and a 2.4GHz 3dBi omnidirectional antenna. The additional network subsystem agents were simulated with wireless routers - Linksys WRT54G (SSID: ONR AP) and

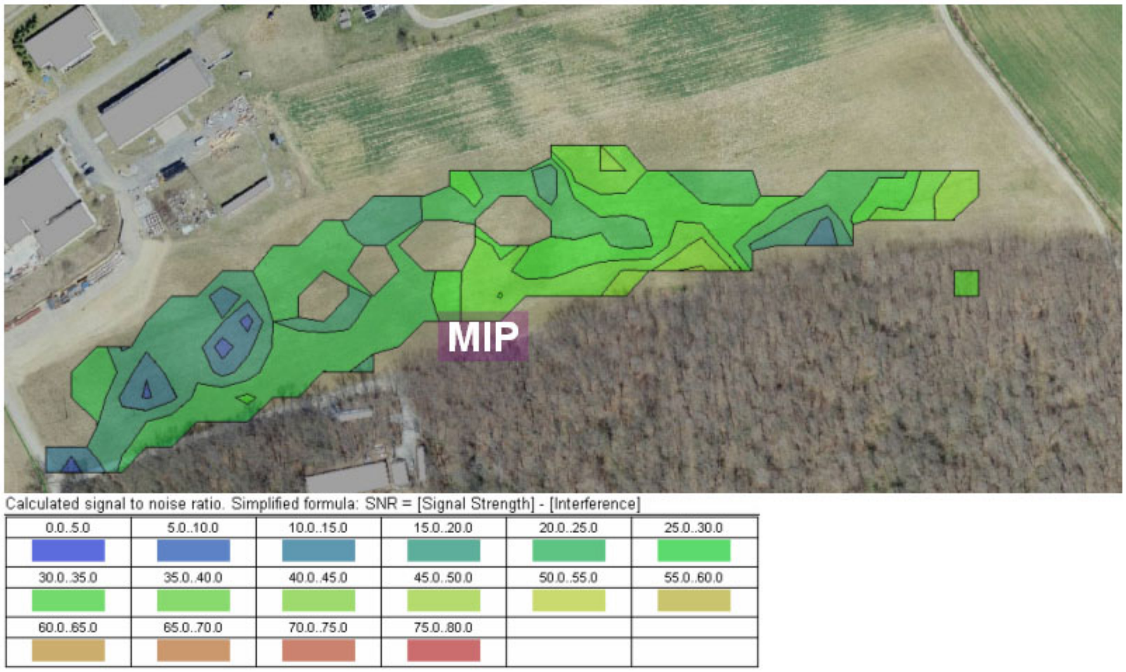


Figure 4.3: SNR Map of Ekahau Site Survey Results using a 3 dBi Omnidirectional Antenna and Coarse Grid Resolution.

D-Link DI-524 (SSID: GC Cliff). The Service Set Identifier (SSID) is the name given by the network administrator to a Wireless Local Area Network (WLAN) , in this case, a wireless router. The SSID of an open wireless network is broadcast to all wireless devices within range of the network access point.

4.2.2 Software Development

Signal-to-Noise Ratio Acquisition

After researching the options available for Windows, NetStumbler [14] was chosen to acquire signal-to-noise ratio. This freeware tool for monitoring 802.11 WLANs provided the simplest way to input the signal-to-noise ratio values for further real-time processing. External scripting allowed for the NetStumbler scan results to be sent over User Datagram Protocol (UDP) to the Site Survey Optimization software.

Signal-to-Noise Ratio Lag An initial idea for the signal optimization problem was to attempt to locate the signal source using antennas. This could be done by rotating a directional antenna or by having an array of static omni-directional antennas. The directional antenna was investigated, but it was never implemented on the vehicle. Due to the two hertz update rate for NetStumbler, a rotating antenna would have to revolve at a significantly slower rate to resolve direction. The vehicle would not be able to move until the antenna rotated a full 360 degrees. This would be an extremely slow process that might work in a static environment, but could not keep up in a dynamic environment. However, the directional antenna experiments were extremely important because they led to the discovery of lag in acquiring the signal-to-noise ratio value. Some of the lag was initially due to software communication buffering. That was quickly fixed, but a three to five second lag was still apparent (Figure 4.4). To examine the lag, a directional antenna was pointed at the access point approximately 800 feet away in an open field until the signal-to-noise ratio value stabilized at its maximum value. The antenna was then instantaneously rotated 180 degrees to point directly away from the Access Point (red line). Once the signal-to-noise ratio stabilized at its minimum value, the antenna was instantaneously rotated back toward the Access Point (green line). This experiment was repeated multiple times at various distances with the same results.

Signal-to-Noise Solution The two main problems with this signal-to-noise ratio acquisition system are the two hertz maximum update rate of NetStumbler and the signal-to-noise ratio lag. At a speed of twenty miles per hour the vehicle would only be receiving one reading every fifteen feet. In addition, the signal-to-noise ratio reading would correspond to where the vehicle was three to four seconds prior. Based on these two factors, the probability of getting an accurate reading of signal strength is low. The solution for this iteration in the signal optimization research was to run the site survey at a slow speed

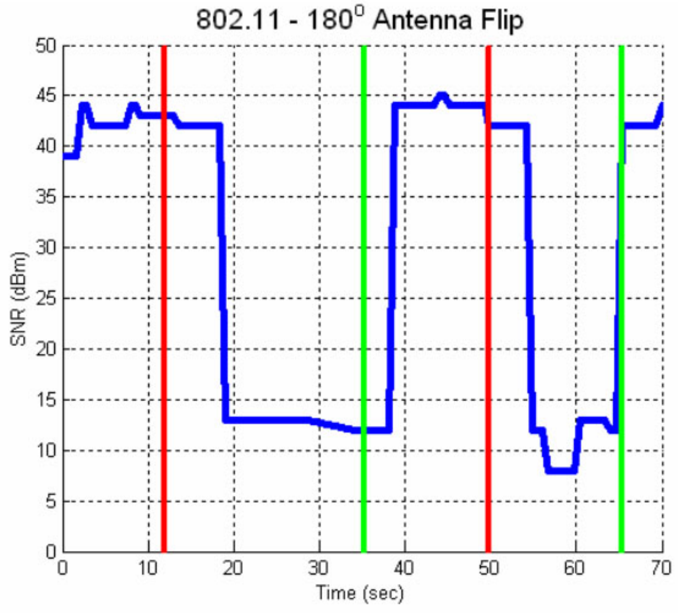


Figure 4.4: 180^o Rotation of a Directional Antenna for an 802.11 Access Point. The blue line is the signal-to-noise ratio, the red line indicates a instantaneous rotation away from the access point, and the green lines displays when the antenna was instantaneously rotated back toward the access point.

of approximately two miles per hour. Therefore, the software gets a signal-to-noise ratio reading every three feet. This reduction in vehicle speed provided the simplest way to eliminate the update rate limitation and minimize the effect of the lag.

Site Survey Optimization

Given a path defined by GPS waypoints, the unmanned ground vehicle can autonomously create a signal-to-noise ratio (SNR) map of the area. The map is created using the signal-to-noise ratio values from NetStumbler and the current position from the vehicle's Inertial Navigation System. Once the map is complete, the optimal location for communication is determined based on three different quantities. The first quantity, deemed the multi-link quality, is used to maximize the minimum communication link. The second quantity used to determine the optimal vehicle location is the sum of the signal-to-noise ratio values. This variable is uses when there are multiple entries with the same maximum multi-link quality. The maximum summation value is also desired for optimal communication. The largest value ensures that the maximum SNR of both access points is achieved. The third quantity used in optimization is vehicle altitude. If there is more than one entry with equal communication strength, the one with the highest altitude is chosen. Locating the communication subsystem at a higher altitude will, in general, provide better line-of-sight to the subsystems in the network. Figure 4.5 shows the software flow for the NetStumbler Site Survey.

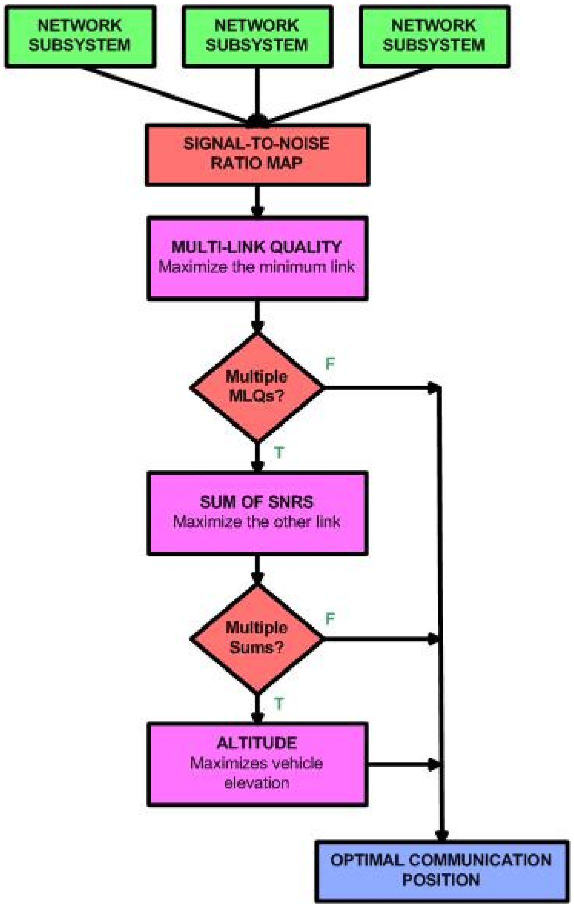


Figure 4.5: NetStumbler Site Survey Software Flow

Navigation Algorithm

The vehicle used a slightly modified version of Dynamic Expanding Zones (DEZ) for autonomous navigation [15]. This reactive navigation algorithm provided waypoint navigation and obstacle avoidance during the site survey. A predetermined file of waypoints was loaded into the DEZ navigation software to create the signal-to-noise ratio map of the test field. However, the program was modified to input new target waypoints once the optimal communication location was determined by the Site Survey Optimization software. This allowed the vehicle to reposition itself autonomously after it had completed the predetermined path. A software flow diagram of the modified DEZ algorithm is displayed in Figure 4.6. The modified DEZ algorithm was also used for the experiments presented in Chapter 5.

4.2.3 Performance

Site Survey Results

The NetStumbler site survey and position optimization algorithm were a success. The unmanned ground vehicle, Rocky, was commanded to autonomously perform the site survey at approximately two miles per hour. Figure 4.7 shows the signal-to-noise ratio maps generated for each access point. Acquiring the GPS positions of landmarks on an aerial photograph can be used to convert a GPS position to pixel location in an image. The conversion takes into account both the scaling from meters to pixels and the rotation of the image. With this GPS to pixel conversion, the vehicle path and corresponding signal-to-noise ratio value is overlaid onto an aerial photograph of the test field. Though the experiments produce similar maps, there are still noteworthy variations in signal-to-noise ratios. These differences could be the result of the slight discrepancy in vehicle path, changes in 2.4 GHz interference in the twenty-five minutes between experiments, or

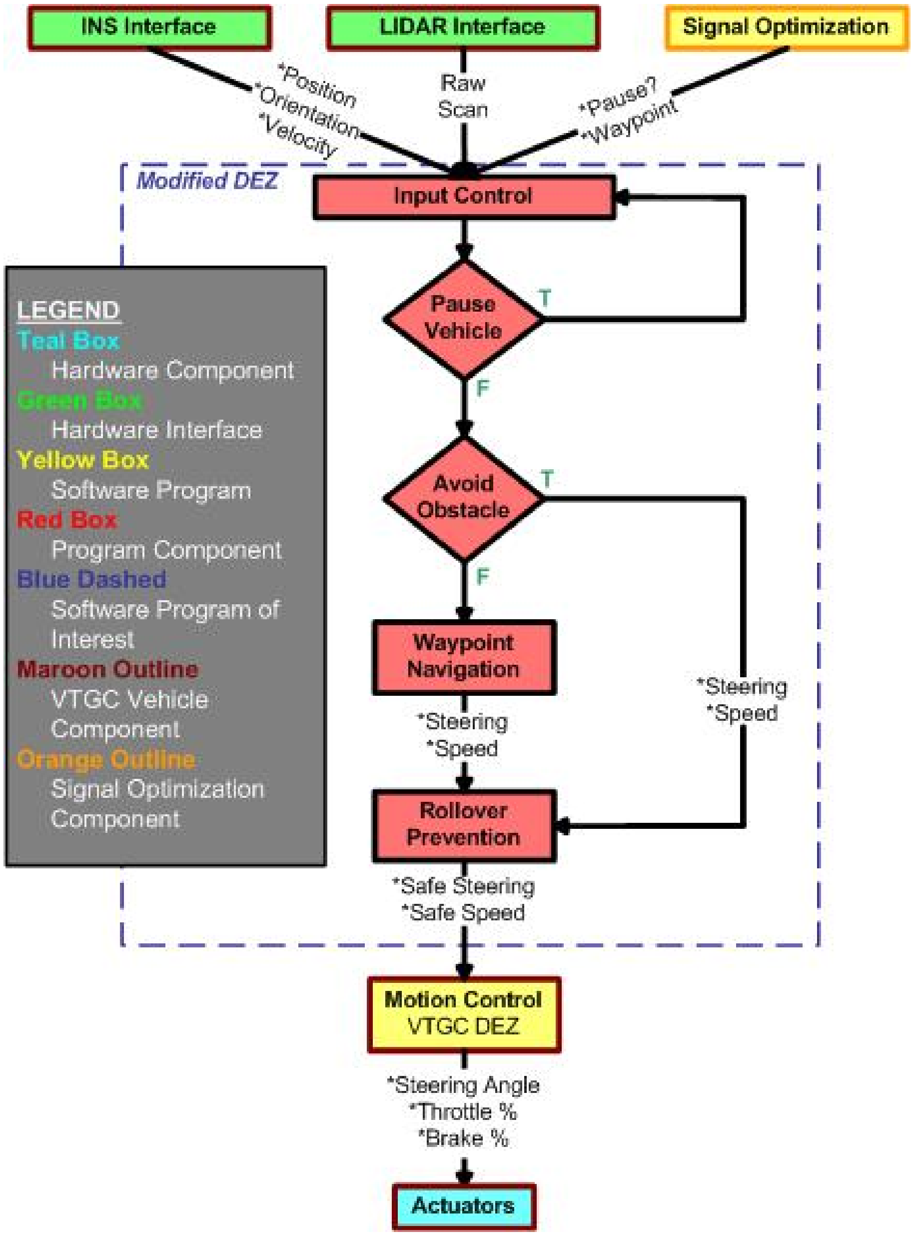


Figure 4.6: Software Level Data Flow Diagram of Dynamic Expanding Zones Modified for Signal Optimization

changes in RF propagation (multipathing) due to variations in the environment.

Once the signal-to-noise ratio survey was complete, Rocky analyzed the data using the three step process illustrated in Figure 4.8. The image overlays display the evaluation parameters for the entire path traveled by the autonomous ground vehicle. Again, the

Figure 4.7: Signal-to-noise ratio maps for each of the two access points during the NetStumbler Site Surveys. Right: maps from the first survey. Left: maps from the second survey.

vehicle first looks at the multi-link quality that maximizes the minimum signal. The areas with the maximum multi-link quality have been circled in red. These are the sections that are further evaluated to find the optimal signal. The sum of SNR values is analyzed next to maximize both signals. The maximum altitude is then used to help ensure line-of-sight to the access points. The altitude plots are a relative elevation with respect to the minimum altitude along the path of the vehicle.

The position for optimal communication was then given to the vehicle for autonomous repositioning. Figure 4.9 shows the site survey path in red and the signal optimization path in yellow for each experiment. Though the vehicle chose different optimal locations for each survey, both make sense. Looking at the altitude overlays in Figure 4.8, the highest part of the field is along the road on the right side of the image. This altitude is also the approximate altitude of both of the access points. The rest of the path traveled is in a valley where the left access point, GC Cliff, does not always have line-of-sight communication with the communication vehicle. As shown in Table 4.1, the two chosen points have similar multi-link qualities, SNR sums, and altitudes. It is reasonable to expect different local optima to be selected in different surveys.

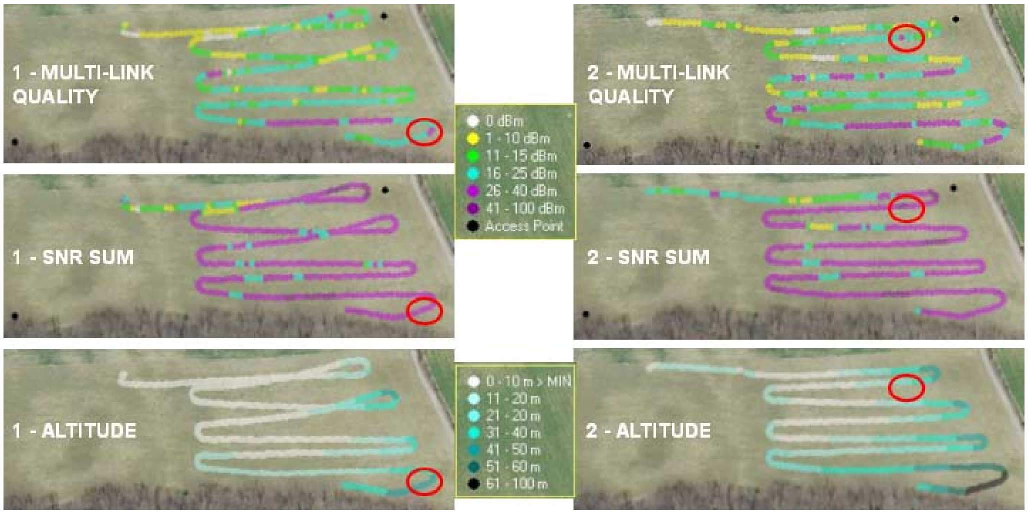
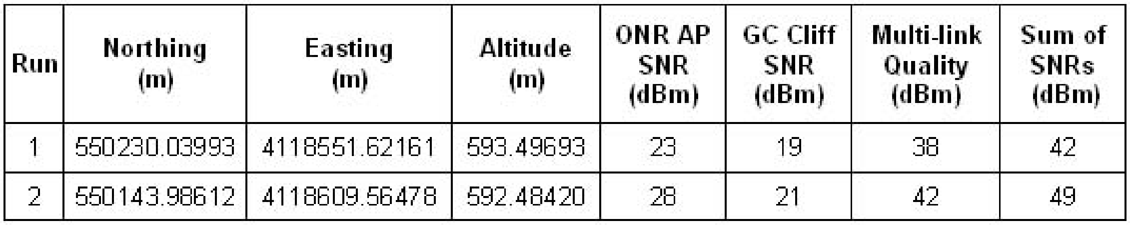


Figure 4.8: Plots of the three step evaluation process to determine best location for communication: Mutli-Link Quality (top), SNR Sum (middle), and Relative Altitude (bottom) of the Survey. The area of interest in the selection process in circled in red. Right: maps from the first survey. Left: maps from the second survey.

Figure 4.9: NetStumbler Site Survey Results. The red line indicates the site survey path taken by the vehicle. The access point locations are shown as black dots. The yellow line represents the path to the point with best communication. Right: maps from the first survey. Left: maps from the second survey



Run	Northing (m)	Easting (m)	Altitude (m)	ONR AP SNR (dBm)	GC Cliff SNR (dBm)	Multi-link Quality (dBm)	Sum of SNRs (dBm)
1	550230.03993	4118551.62161	593.49693	23	19	38	42
2	550143.98612	4118609.56478	592.48420	28	21	42	49

Table 4.1: Table of the Optimal Communication Locations chosen during two NetStumbler Site Surveys.

Chapter 5

Signal Optimization

Once the successful autonomous site surveys were completed, the next steps were to solve the signal-to-noise ratio latency problem, address mobile ad-hoc networking issues, and finalize a plan for real-time signal optimization.

5.1 Hardware Overview

5.1.1 Wireless Mesh Cards

Research and discussion with Sean Kragelund at the Naval Postgraduate School led to the discovery of a mobile ad-hoc networking solution that could also help reduce SNR latency. ITT Technologies and Motorola produce wireless mesh cards (WMC6300) that provide a self-forming, self-healing, self-configuring mobile ad-hoc network [9] [13]. Using their mesh enabled architecture, a flexible wireless network is create that maximize performance and bandwidth efficiency. Every card acts as a repeater, and data can hop up to fifteen nodes (less than five is preferable) to its destination. The cards operate at 2.4GHz and minimize collision by using "listen before talk" and acknowledgment algorithms to control network access.

As opposed to 802.11, the wireless mesh cards use a Quadrature Division Multiple Access (QDMA) radio platform that was designed and optimized for mobile ad-hoc broadband networking. The 802.11 system was designed to be a cost effective solution for wireless communication between relatively stationary computers. These systems require high data rates over short distances. On the other hand, QDMA was optimized for wide area, mobile environments. They have high performance RF capabilities with real-time equalization algorithms to compensate for rapidly varying RF conditions. QDMA radios have a range of 1 mile with line-of-sight, while 802.11 only has a typical maximum of 300 feet. The QDMA-based networks are also rated for speeds between transmitter and receiver of up to 250 mph compared to 802.11 that drops the link around 20 mph.

The cards, however, do not support multicasting, which can be a useful tool with inter-vehicle communication. Due to their long range capabilities, they can also only transfer data at a maximum of 6 Mbps. Transferring a large amount of data, such as real-time video, can slow down the entire network. For the signal optimization research, small amounts of data are being sent to a single subsystem. Therefore, the QDMA radios provide an ideal solution.

The mesh cards were integrated into the existing signal-to-noise ratio monitoring system to investigate the Signal-to-Noise Ratio (SNR) lag. The same directional antenna experiments that were performed with the 802.11 system were repeated for the mesh system. A directional antenna was pointed at the access point approximately 800 feet away in an open field until the signal-to-noise ratio value stabilized at its maximum value. The antenna was then instantaneously rotated 180 degrees to point directly away from the access point (red line). Once the signal-to-noise ratio stabilized at its minimum value, the antenna was quickly rotated back toward the access point (green line). The results of both the 802.11 and mesh systems are displayed in Figure 5.1. The mesh system shows great improvement with an average lag of one second as opposed to three to five seconds in the 802.11 system.

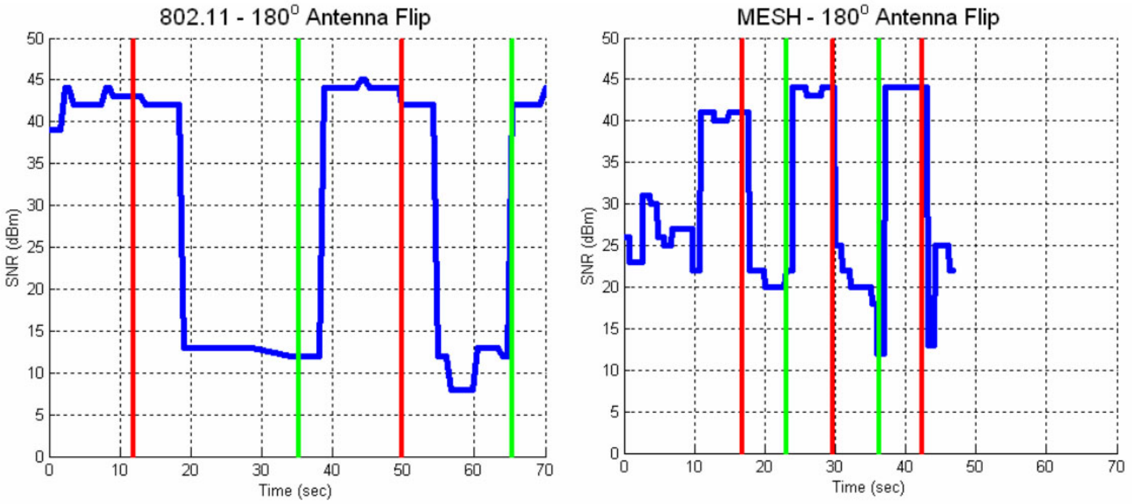


Figure 5.1: 180^o Rotation of a Directional Antenna for an 802.11 Access Point (left) and WMC6300 Network Card (right). The blue line is the signal-to-noise ratio, the red line indicates a instantaneous rotation away from the access point, and the green lines displays when the antenna was instantaneously rotated back toward the access point.

5.1.2 Network Subsystems

Communication Subsystem

Cliff, the other Virginia Tech entry to the 2005 DARPA Grand Challenge, was used as the unmanned ground vehicle for this experiment. The vehicle was equipped with an ITT mesh card in order to monitor the other network subsystems. Figure 5.2 shows the communication subsystem hardware.

Figure 5.2: Communication Subsystem Hardware: Cliff (left), Dell Inspiron Laptop (center), ITT WMC6300 - mesh card (right)

Other Subsystems

The other subsystems in the network could include manned and unmanned aerial, surface, underwater, and ground vehicles aw well as ground troops. These subsystems must broadcast their current GPS position over the mesh network to be included in the network optimization procedure. An immediately available solution for providing this capability was the micro wireless instrument payload (μWIP) developed by researchers at Virginia Tech as an instrument to help evaluate the performance of an unmanned vehicle [8]. This data acquisition system operates independent of the control system of the vehicle. Its purpose is to record and wirelessly transmit the position and orientation of the vehicle. The μWIP (Figure 5.3) was designed for smaller unmanned vehicles where the size and weight of a payload is crucial. This system provides a highly mobile GPS for additional network subsystems. In addition to wireless transmission, the μWIP can also can provide information over ethernet. It can, therefore, be interfaced to provide GPS data over the mesh network.

Figure 5.3: Network Subsystem Hardware: μWIP (left), Dell Latitude Laptop (center), ITT WMC6300 - mesh card (right)

5.2 Navigation Algorithm

Research, previous experiments, and the goal of a mobile network led to the development of a navigation algorithm based on current subsystem position and current signal-to-noise ratio values. In order to succeed, two conditions must be met. First, when the autonomous ground vehicle begins, it has communication with all subsystems in the network. The second condition is that the location of all subsystems is known. In other words, the subsystems must broadcast their current location. A system flow diagram is illustrated in Figure 5.4.

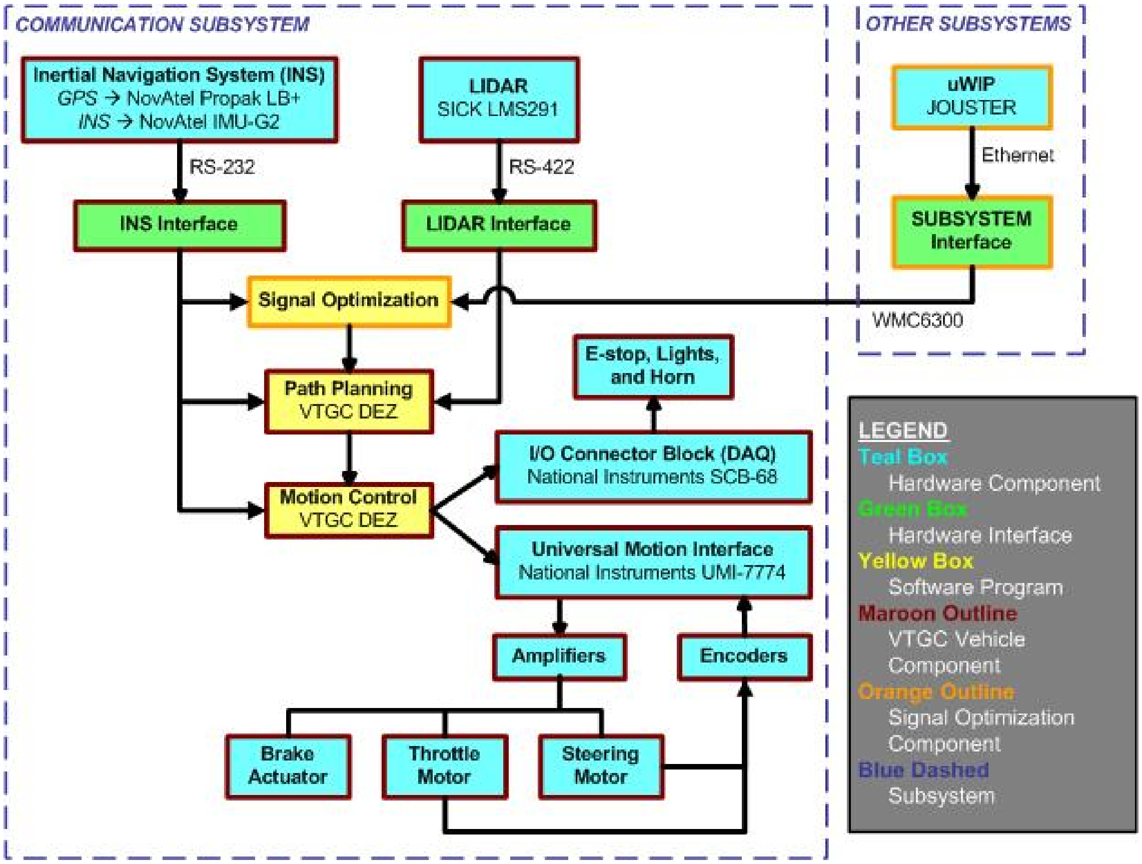


Figure 5.4: System Level Data Flow Diagram for the Signal Optimization Experiment

5.2.1 Software Development

Optimization consists of a two part process that moves a vehicle in the direction that will maximize the rate of change of the desired metric. The first step is to determine the direction of the maximum rate of change, typically referred to as steepest descent. The next step is to move the vehicle some distance in the determined direction. This method was used to navigate the vehicle for optimizing the communication link quality.

Subsystem Attraction

The subsystem attraction provides the direction of steepest ascent which is the first step in optimization. In other words, it is used to calculate the desired heading for the communication vehicle that will maximize rate of change for improving signal-to-noise ratios of the network. Referring to the example in Figure 5.5, as the autonomous ground vehicle (P_0) begins, it looks at the current signal-to-noise ratio and position of each subsystem (P_1, P_2, P_3) in the network. The signal-to-noise ratio is used to calculate an attraction value, A, for each subsystem, as follows:

$$A = 100 - SNR \tag{5.1}$$

This attraction value is used as a vector magnitude in subsequent calculations to determine the desired vehicle heading. Next, the GPS position information is used to find the unit vector from the current vehicle position, P_0, to each of the network subsystems. Each subsystem attraction value is applied as the magnitude of the corresponding subsystem unit vector. The following vector addition then determines the desired vehicle heading:

$$\theta_D = atan2\left(\frac{Y}{X}\right) = atan2\left(\frac{A_1 sin\theta_1 + A_2 sin\theta_2 + ... + A_N sin\theta_N}{A_1 cos\theta_1 + A_2 cos\theta_2 + ... + A_N cos\theta_N}\right) \tag{5.2}$$

where A_N is the signal-to-noise ratio attraction for network subsystem N and θ_N is the angle of the vector to network subsystem N.

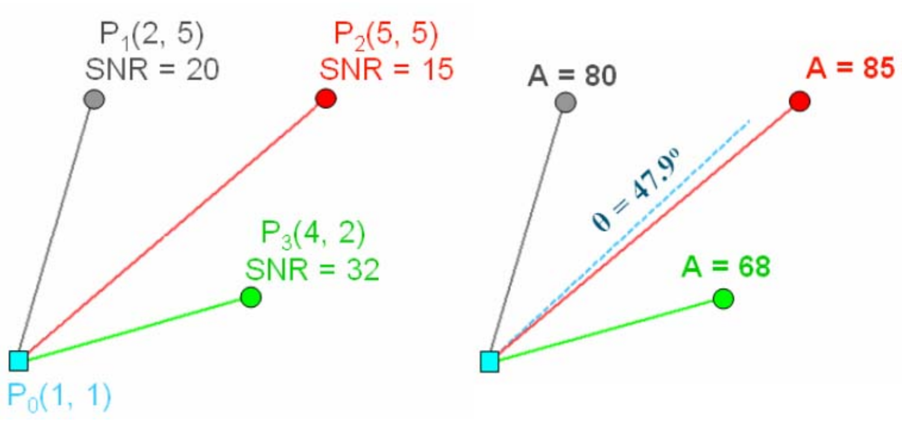


Figure 5.5: Subsystem Attraction Example. P_0 represents the autonomous communication subsystem. P_1, P_2, and P_3 are other subsystems in the network. Using current location and SNR, P_0 is attracted to lower SNR values.

For the second step of an optimization-based approach, using a constant distance to move is typical. Therefore, the desired vehicle waypoint for the communication vehicle is placed five meters ahead in the direction of the desired vehicle heading. The angle and range to the waypoint are calculated using this waypoint and the current vehicle heading. The communication vehicle then generates a desired speed and steering angle to navigate to this waypoint. This process is repeated at approximately ten hertz to ensure the communication subsystem is always making its decision based on the current position of all network subsystems.

Vehicle Pauses

There are three conditions that will autonomously pause the motion of the communication subsystem. The first two pauses result from finding a local maximum. Constantly moving to new locations could potentially worsen network communication. If the communication subsystem is at a local maximum based on signal-to-noise ratio and a vector threshold, it will be commanded to remain stationary. The third pause is a safety feature added to prevent the communication subsystem from running into another subsystem. This condition will pause the vehicle until the desired waypoint moves the vehicle away from the network subsystem.

A signal-to-noise ratio threshold is included in the algorithm to keep the vehicle at its current location if the signal-to-noise ratios from all other network subsystems are above a certain value. Signal strength is continually monitored, and the vehicle will resume autonomous signal optimization when needed. The vehicle also keeps track of other subsystem position in the event that communication with a subsystem is lost. The communication subsystem can move in the direction of the last known position to regain the communication link.

The vector threshold results in a pause if the other subsystems are approximately equiangular to one another and the signal-to-noise ratios are above a certain threshold. Without this, the vehicle has the potential of driving in a circular pattern around this local maximum. Small changes in the network would significantly change the desired direction for the communication vehicle. Figure 5.6 illustrates how a small change in signal-to-noise ratio could greatly effect the desired travel direction designated by a dashed teal line.

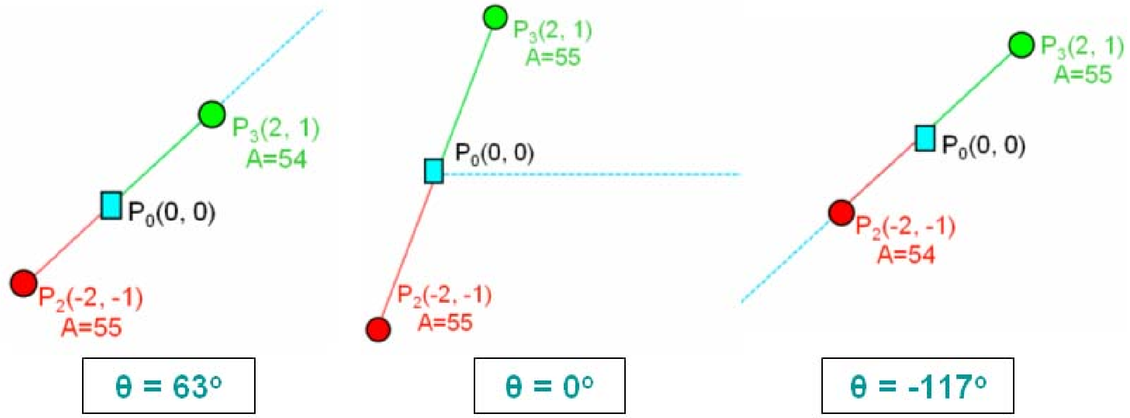


Figure 5.6: Vector Threshold Example. Each of the three plots above have a significantly different desired direction (teal dashed line) with only a 1 dB change in SNR/Attraction.

The third pause prevents the communication vehicle from running into another subsystem. This is a backup, safety feature that commands zero speed and steering angle if the communication subsystem is within ten meters of another subsystem. However, this should never occur. As the vehicle moves toward this specific network subsystem, this SNR value will increase and the others will decrease. Therefore, the vehicle would have a stronger attraction to the other subsystems.

5.2.2 Performance

The algorithm developed to attract the communication vehicle to other subsystem based on signal-to-noise ratio performed as expected in the three-subsystem network. The vehicle determined the desired waypoint and moved to position itself approximately halfway between the other two network subsystems. Although testing has been conducted only for the three-subsystem system, the algorithm can run with as many subsystems as the computer processor can handle.

Static Experiment

As an initial test, two static network subsystems were placed in the test field. The communication subsystem was placed at an arbitrary location in the test field and set in autonomous mode. The vehicle used the signal optimization algorithm to determine and reposition itself in the location with optimal communication. The test was repeated multiple times at various starting location to compare results. The signal-to-noise ratio threshold was also adjusted to evaluate the outcome. Figure 5.7 shows the path, in red, taken by the communication vehicle from one of its starting locations. The other two network subsystems are displayed as yellow and blue dots. For this experiment, the threshold was set at 25 dB - a fairly low expectation for the typical range of 5 to 65 dB. Therefore, the vehicle did not have to travel directly in between the other subsystems to achieve the desired signal-to-noise ratio. The vehicle paused based on the SNR threshold.

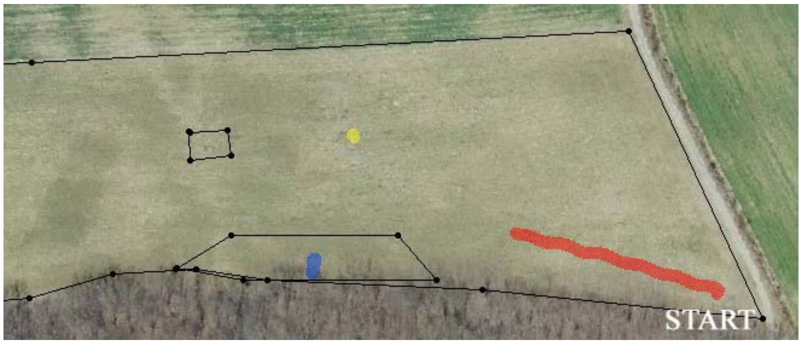


Figure 5.7: Static SNR Attraction Navigation results using an SNR threshold of 25 dB. The path of the communication subsystem is displayed in red. The other two network subsystems appear in yellow and blue.

The experiment was also repeated for higher signal-to-noise ratio threshold values. Figure 5.8 shows the results from a different starting position for the vehicle and using a signal-to-noise threshold of 45 dB. The vehicle drove in between the two subsystems and achieved the vector threshold. However, the threshold had not yet been met for the blue subsystem. Therefore, the vehicle continued to move toward that subsystem and paused when the signal-to-noise ratio and vector thresholds were met. The vehicle had to travel farther to meet the requirements, but came to a rest almost directly between the two other subsystems.

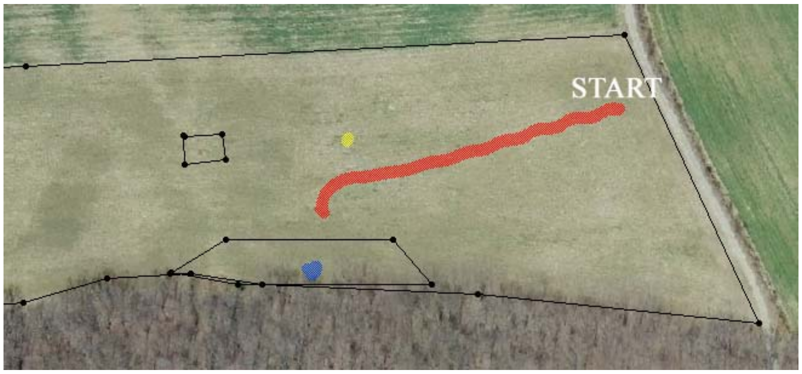


Figure 5.8: Static SNR Attraction Navigation results using an SNR threshold of 45 dB. The path of the communication subsystem is displayed in red. The other two network subsystems appear in yellow and blue.

Dynamic Experiment

The algorithm also worked in a dynamic environment. This was tested by observing the communication subsystems ability to track a moving subsystem. Both moving subsystems were placed in a truck that drove between five and ten miles per hour through the test field. The signal-to-noise ratio threshold was set low (25 dB) in order to keep the communication subsystem moving. The results of this experiment are displayed in Figure 5.9. The path of the communication subsystem is shown in red while the truck carrying the other two network subsystems is shown in yellow and blue. The autonomous communication vehicle follows the path of the truck exactly when it was driving straight. When the truck starts driving the curves path, the angle to waypoint calculation causes the communication vehicle to start turning as the truck turns. Therefore, the vehicle takes a turning path inside that of the truck.

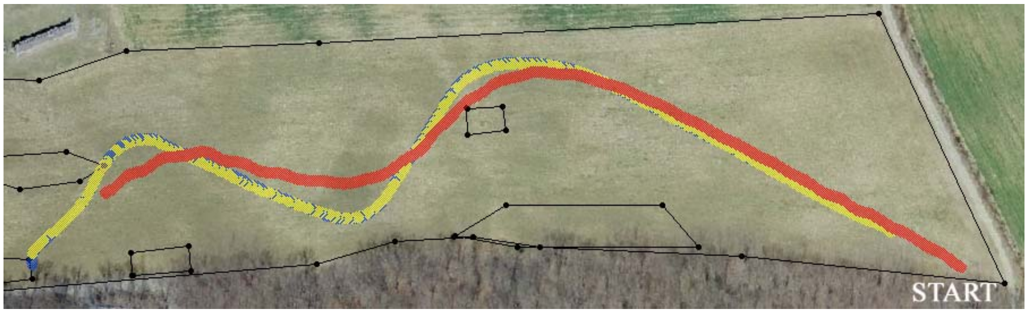


Figure 5.9: SNR Attraction Navigation Results - 25 dB Threshold. The path of the communication subsystem is displayed in red. The other two network subsystems appear in yellow and blue. For this test, the network subsystems were both placed in a single truck and driven around the field.

One thing that was noticed during these experiments was the tendency for some overshoot in steering. It was not always apparent, but the wavy paths of the vehicle in Figures 5.7 and 5.8 provide a good illustration of this phenomenon. After analyzing data logged by the signal optimization software, the desired vehicle heading used to calculate the desired waypoint is commanding the correct direction to the other network subsystems. The vehicle also accurately follows the desired waypoint that is set based on this desired direction. Plotting all of the desired waypoints from an experiment shows that the oscillatory behavior is generated by the desired waypoints. The underlying cause remains unknown; however, a waypoint radius minimizes the oscillation. The first desired waypoint is stored as a reference waypoint. As new desired waypoints are calculated, they are compared to the reference waypoint. If the waypoint is within a certain radius, the reference waypoint is commanded. If not, the new desired waypoint becomes the new reference waypoint and is commanded to the vehicle. Figure 5.10 shows the original desired waypoints (in black) and the straighter path (in red) that results from using a waypoint radius on logged experimental data. The original desired waypoint path was smoothed using a ten meter waypoint radius. Experimentation with this waypoint radius should equal or improved results for correcting the oscillation.

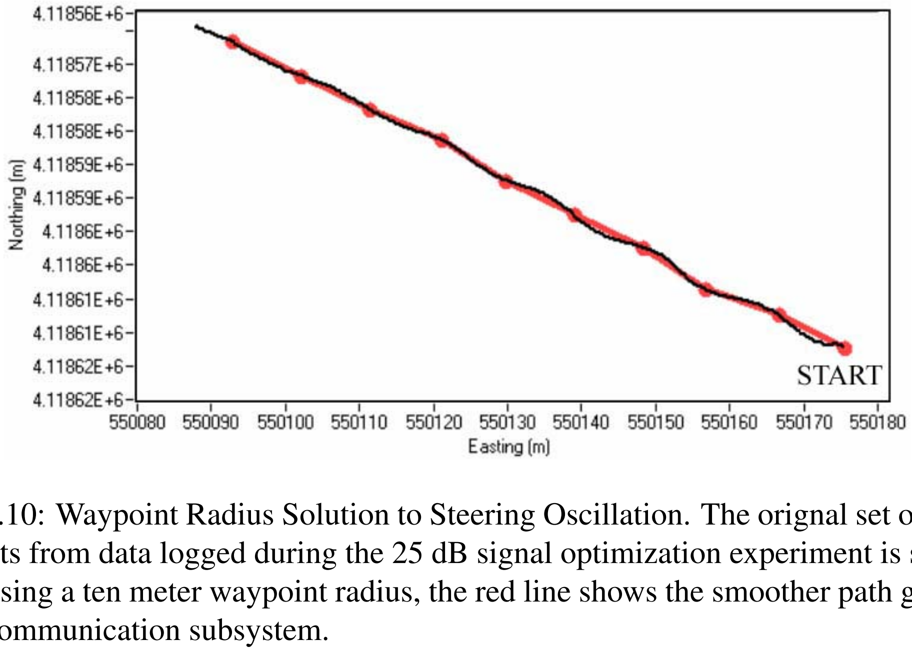


Figure 5.10: Waypoint Radius Solution to Steering Oscillation. The orignal set of desired waypoints from data logged during the 25 dB signal optimization experiment is shown in black. Using a ten meter waypoint radius, the red line shows the smoother path generated for the communication subsystem.

5.3 GIS Evaluation Software

Geographic Information System (GIS) data was also investigated to evaluate its ability to assist the navigation algorithm aboard the autonomous communication vehicle. Elevation, infrastructure, and ground cover GIS data are required for this process. The GIS data can be used to evaluate the potential vehicle waypoint before it is sent to the vehicle for navigation. In other words, it can be used as preventative measure to help plan the next move for the vehicle.

5.3.1 Software Development

Line-of-Sight

Line-of-sight is determined based on the viewshed and infrastructure. Using a grayscale elevation image and the vision toolkit from LabVIEW, the viewshed is analyzed. The viewshed operates solely on the elevations from the image and ground vehicle heights. Acquiring the GPS positions of landmarks on an aerial photograph can be used to convert the GPS position to pixel location in the image. The conversion takes into account both the scaling from meters to pixels and the rotation of the image. After this GPS to pixel conversion, the communication and other subsystem are located on the image and the line between them is evaluated. The image elevation is compared to the calculated elevation of the line between the vehicles. If the actual elevation is ever greater than a calculated elevation, the software outputs that the line-of-sight between the vehicle is blocked. Buildings are another possible occlusion; therefore, the GIS infrastructure data is also analyzed for line-of-sight. If the path between the communication subsystem and other network subsystem passes through a building, then line-of-sight is blocked. Figure 5.11 illustrates blocked line-of-sight evaluation examples for both viewshed (left) and infrastructure (right).

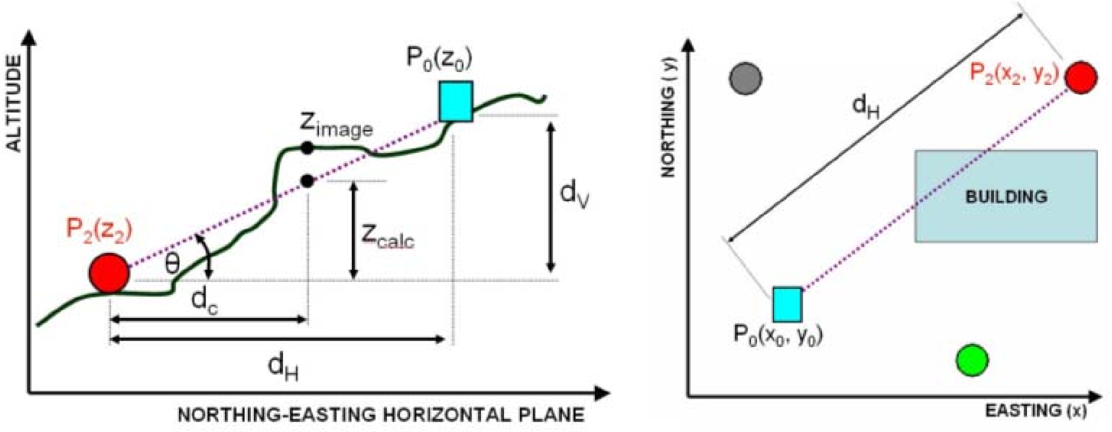


Figure 5.11: Illustration of Line-of-Sight Viewshed (left) and Line-of=Sight Infrastructure (right). Both diagrams show examples of blocked LOS between the communication subsystem (teal) and a network subsystem (red).

Boundary and Obstacles

The boundary is the perimeter of the user-defined area of operation of the vehicle. Obstacles are defined as any area the vehicle cannot drive, which includes buildings, bodies of water, and woods. Both boundaries and obstacles can be detected with the same software if their perimeter is defined as a closed set of waypoints. Figure 5.12 displays black lines that represent the boundary and obstacles on the test field. Once defined, a waypoint is checked for the number of perimeter lines that appear to the east and north. If both produce an odd number, then the point is within the defined perimeter.

Figure 5.12: Vehicle boundary (perimeter of the field) and obstacles overlaid on the test field.

5.3.2 Performance

Viewshed Evaluation

The results of the viewshed experiment is limited by the accuracy of the GIS data used. Unfortunately for this experiment, the best GIS data that could be obtained and easily converted to a grayscale image was one-third arc second, which is equivalent to approximately ten meter resolution [18]. The data is input into the viewshed program as a grayscale image (Figure 5.13 - left). The resulting 3D image is displayed on the right in Figure 5.13. Though better data would have been preferred, in general, this GIS data provided good representation of the test field.

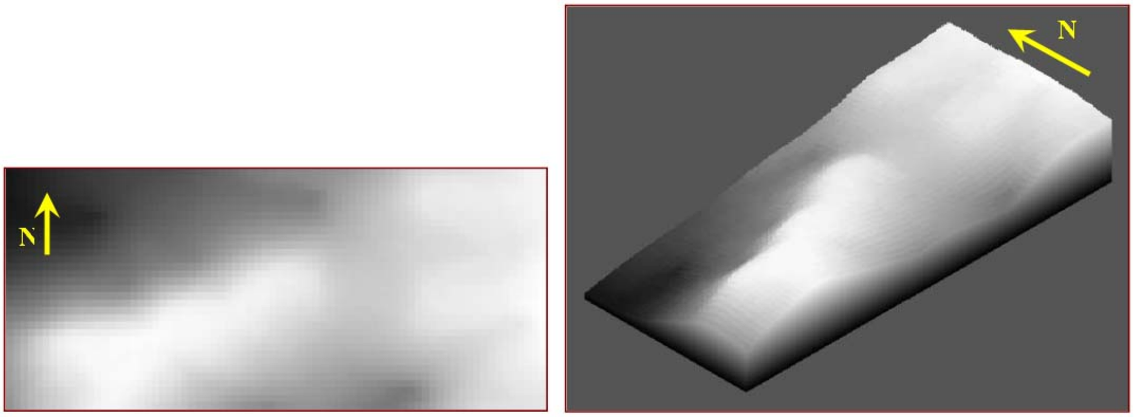


Figure 5.13: GIS Data - Elevation (one-third arc second) Image of Plantation Road Test Field. Left: Grayscale Elevation Image. Right: 3D Elevation Image

The evaluation of the viewshed detection consisted of a single subsystem placed at an arbitrary location in the test field (the yellow dot in Figure 5.14). The communication subsystem then drove a path around the field to investigate the results. When the path is blue, the two subsystems have line-of-sight. When the path is red, they do not. The software occasionally indicated that line-of-sight was blocked when it was not. This is due to the low grid resolution for the elevation data that creates steps of elevation instead of a smooth contour of the ground surface. An example of how the GIS data could produce incorrect line-of-sight results is illustrated in Figure 5.15. The dotted purple line represents the line-of-sight between the two vehicles. The image on the left shows the steps in elevation generated from GIS data with low grid resolution. The image on the right shows the actual contour of the ground surface.

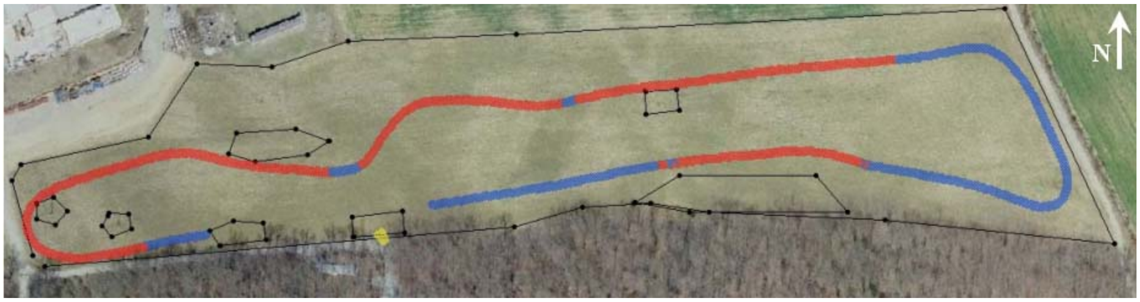


Figure 5.14: Viewshed Evaluation Results. A network subsystem (yellow) was arbitrarily placed in the test field. The communication vehicle drove the path shown. Red indicates that line-of-sight is blocked between the two vehicles, and blue represents when there was line-of-sight.

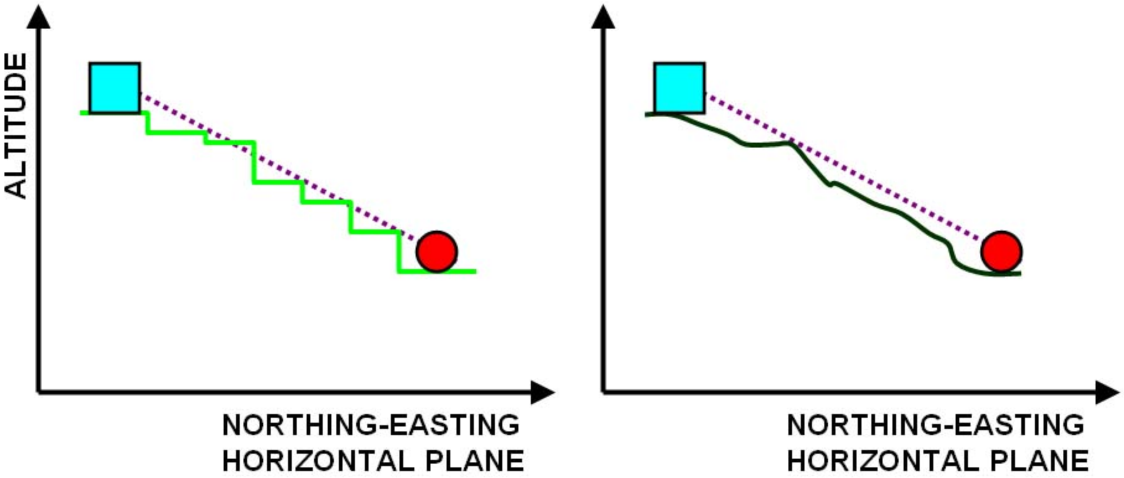


Figure 5.15: Illustration of Blocked Line-of-Sight due to Low Resolution GIS Data. The LOS between the two vehicles is sketched as a dotted purple line. The left image shows that the LOS line is blocked by the step changes in elevation generated by low resolution GIS data. The image on the right displays the clear LOS for the actual contour of the ground.

Boundary and Obstacle Evaluation

The vehicle must be able to remain in the desired field of operation and avoid obstacles such as buildings, water, and difficult-to-traverse ground cover. A single algorithm was developed to check whether a waypoint is within a specific perimeter. For this experiment, the boundary was set as the perimeter of the field. The areas containing telephone poles, bushes, buildings, and other obstacles have also been outlined. Both the boundaries and obstacles are displayed as black waypoints and perimeter lines in all of the field overlay images in this section. Figure 5.16 shows the path of the vehicle as it was driven from the lab to the test field. The path is red when the vehicle is out of bounds, and blue when it is in bounds. Figure 5.17 displays the results of the obstacle test with red signifying the vehicle is in an obstacle. This test was performed with a human driver, and the obstacle perimeters were set with a factor of safety around them. This allowed the vehicle to pass through a designated obstacle for the experiment.

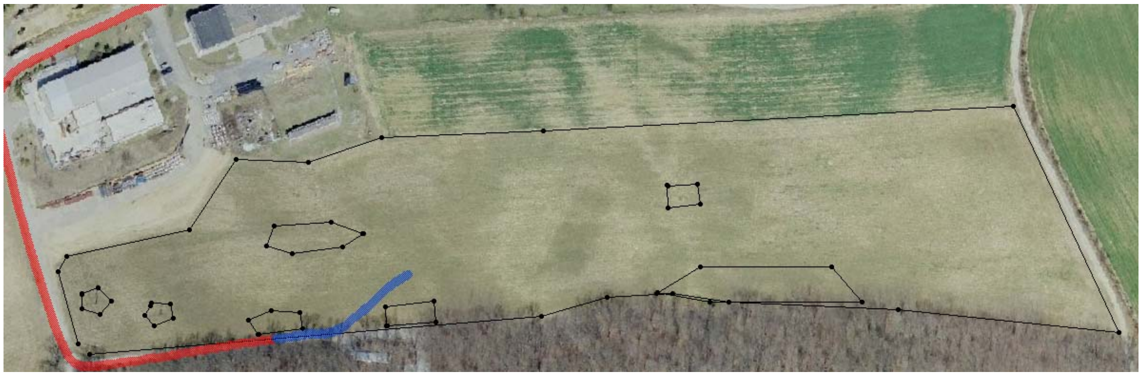


Figure 5.16: Vehicle Boundary Evaluation Results. As the path of the vehicle crosses the boundary line, it changes from red (out of bounds) to blue (in bounds).

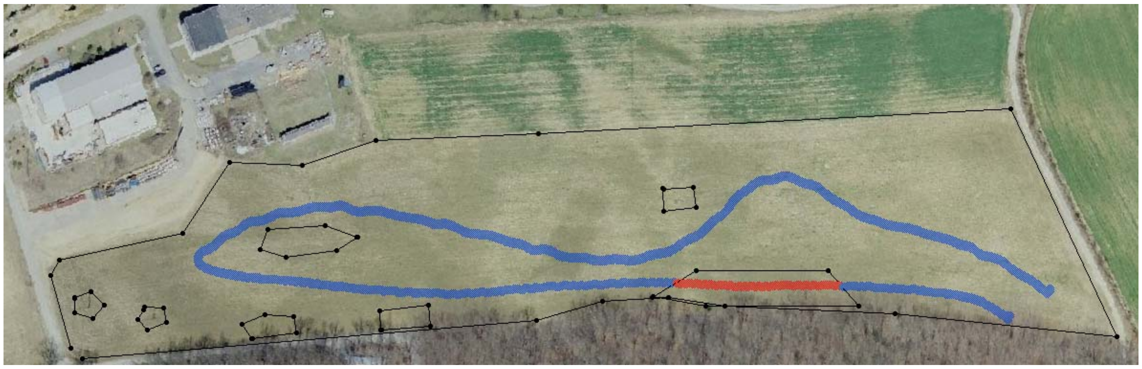


Figure 5.17: Obstacle Evaluation Results. As the path of the vehicle crosses the obstacle line, it changes from blue (traversable) to red (in an obstacle) and back.

5.4 GIS Cost Function

The signal optimization experiments using signal-to-noise ratio attraction successfully repositioned the ground vehicle to improve communication. The GIS data experiments also demonstrated that using GIS data can assist in navigation planning. The cost-based algorithm described below provides an approach to implementing the GIS data into the signal-to-noise ratio attraction navigation. Once the desired vehicle heading is established, multiple potential waypoints can be chosen for evaluation. A cost can then be calculated for each of these points to decide the next best position for the unmanned ground vehicle. The cost is a function of the following factors: elevation, line-of-sight, GIS data, vehicle heading, data rate, and deviation from desired path. A software data flow diagram of this cost function is displayed in Figure 5.18.

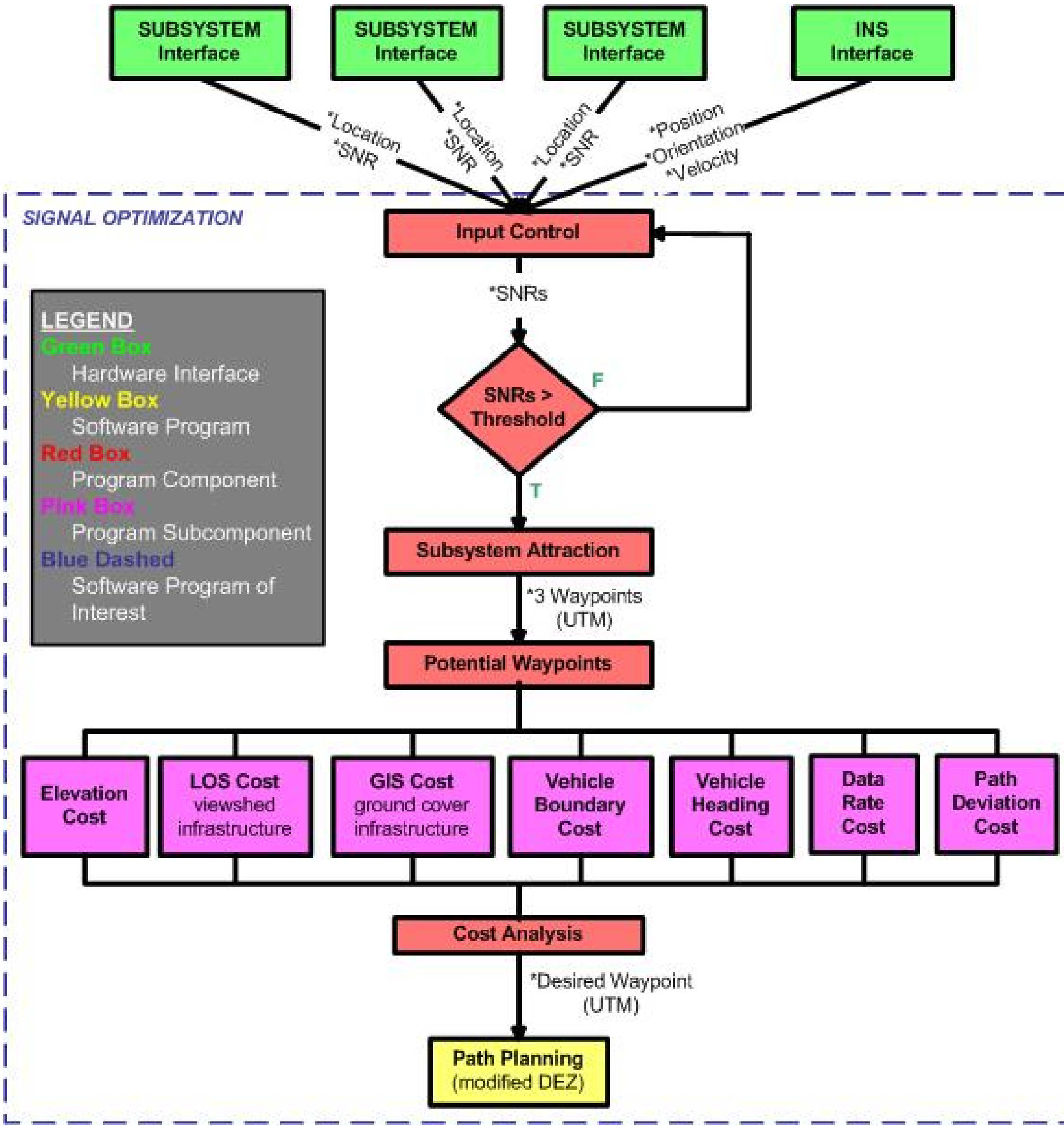


Figure 5.18: Software Level Data Flow Diagram of Signal Optimization

5.4.1 Elevation Cost

Typically, the communication subsystem will have a better view of the surrounding area at higher elevations. The elevation cost provides an attraction to higher elevation

$$ElevationCost = Gain * (WaypointElevation - VehicleElevation) \tag{5.3}$$

where *WaypointElevation* is the elevation of the waypoint being evaluated, and *VehicleElevation* is the elevation of the communication subsystem. This analysis should also monitor for elevation changes that the unmanned ground vehicle cannot accomplish. For example, if the elevation changes drastically between two points (a drop off or a cliff) it is probably not safe for the unmanned ground vehicle to proceed to that point.

5.4.2 Line-of-Sight Cost

The second attraction is to the waypoint with the line-of-sight to the most subsystems.

$$LOScost = Gain * \frac{NumberOfVisibleSubsystems}{TotalNumberOfSubsystems} \tag{5.4}$$

where the *NumberOfVisibleSubsystems* is the number of subsystems that have line-of-sight with the potential waypoint for the communication subsystem. Therefore, the highest value will occur only when there is line-of-sight with all of the other network subsystems.

5.4.3 GIS Cost

The Geographic Information System (GIS) cost influences the vehicle by avoiding buildings and difficult-to-traverse ground cover.

$$GIScost = Gain * (LevelOfTraversability) \tag{5.5}$$

To find the *LevelOfTraversability*, the software looks at the GIS data for the potential waypoints to determine if they are drivable goals. High traversability is equivalent to points that are not in water, in the woods, or in a building. In addition to checking the GIS data, the algorithm also ensures that the vehicle does not drive outside of its operation area designated by a predetermined vehicle boundary. This cost will also have to consider how to handle the situation where all the potential waypoints may be out of bounds or within an obstacle. An aerial photograph that has been overlaid with sample GIS data and vehicle boundaries is displayed in Figure 5.19. The drivable area has been roughly highlighted in orange.

Figure 5.19: Illustration of Example GIS Data. The aerial photo of the test field has been overlaid with building perimeters, tree line of the woods, difficult-to-traverse brush, bodies of water, and roads. Also shown is the predetermined vehicle boundary that the communication subsystem must operate.

5.4.4 Vehicle Heading Cost

The vehicle heading cost provides an attraction to the waypoint in the original desired direction for the communication subsystem. This cost refers back to Subsystem Attraction (Section 5.2.1), where vector addition was used to calculate the desired vehicle heading. The vehicle should only deviate from this desired direction if the other potential waypoints provide a significantly better position for communication.

5.4.5 Data Rate Cost

As previously discussed, signal-to-noise ratio was chosen as the metric for signal optimization navigation. However, the amount of data that is being sent and received is ultimately the most important factor. This cost is a way to influence the movement of the vehicle based on data rate. It will attract the vehicle to potential waypoints that have higher data rates to increase network communication.

5.4.6 Path Deviation Cost

Finally, the path deviation cost is an attraction to the desired path. This path refers to an overall goal that must be achieved by the autonomous ground vehicle. For example, the communication subsystem may have the goal of traveling along a beach. The vehicle should remain on this desired path unless it is detrimental to network communication. It should then take the necessary action to maintain communication until it can return to the overall goal.

Chapter 6

Conclusion

6.1 Conclusion

Wireless technologies have become a part of every day life for many people throughout the world. In the United States, towers have been built to provide coverage to most of the country. This is an ideal solution for the general public, but it may not be an option for military applications operating overseas. Combat systems and personnel must remain in contact to carry out missions. To address this need, a solution using an autonomous ground vehicle was investigated. A navigation algorithm was researched and developed for optimizing communication in a dynamic network environment. This autonomous solution monitors and reacts to current signal-to-noise ratio conditions allowing it to adapt to the changing network. Using a vector sum of the current conditions, a goal waypoint is calculated to attract the autonomous vehicle toward subsystems with lower signal-to-noise ratios. The vehicle follows these waypoints until a local maximum is found and pauses the motion of the vehicle.

GIS data was also explored to help plan movement instead of simply reacting to the current environment. Based on this data, software was developed and tested. One program analyzed the line-of-sight based on viewshed by comparing the GIS elevation and projected line-of-sight elevation between two GPS points. Another program was written for determining level of traversability. Based on predetermined area perimeters, the software indicated whether a desired point was within that designated area or not. It was used for vehicle boundary and large obstacle detection, such as buildings and difficult-to-traverse ground cover.

From theory to experimentation, the signal-optimizing navigation algorithm and GIS data evaluation software proved to be a success. The autonomous ground vehicle repositioned itself for optimal communication with the other network subsystem. It also could detect line-of-sight between subsystems and whether the vehicle was within the vehicle boundary and obstacles. A GIS cost function suggests a solution for implementing the GIS elevation into the signal-to-noise ratio attraction algorithm. Further development and testing would be required to implement this cost-based algorithm.

6.2 Future Work

In addition to combining the signal-to-noise ratio attraction and the GIS data, real world implementation of the algorithm would require more research and development. This section describes potential future work for this signal optimization research.

6.2.1 GIS Cost Function

Implementation of the GIS Cost Function would require incorporating software to generate the elevation cost and the GIS cost. With the proper GIS data available, this should be a fairly straight forward task. The gains would also have to be set to produce the correct balance of the five different costs. The algorithm would have to be testing in both a static and dynamic environment. It has been developed for a dynamic network, but only further experimentation will prove if all aspects of a dynamic environment have been properly considered.

6.2.2 Mobile Ad-hoc Networks

The navigation approaches discussed in this thesis are based on 802.11 network characteristics. The wireless mesh cards provided a solution for a mobile ad-hoc network; however, much of the development was already in place once the mesh cards were implemented. Additional work should be completed to adjust navigation planning based on the characteristics of a mobile ad-hoc network. In a mobile ad-hoc network, each node acts as a repeater. Therefore, it is not likely that the communication subsystem will need to position itself as the central subsystem in the network. The navigation software should look at the positions of all subsystems and position the vehicle to maintain the links to the furthest subsystems from the overall network location. This will greatly increase the range of the network and the required size of the test field.

6.2.3 Additional Suggestions

Three recommended improvements include adding reactive behaviors for when communication is lost with one or more network subsystems, a way to learn signal-to-noise ratio patterns for the current environment, and using the trajectories of the other network subsystems to aid in repositioning. This section provides a brief explanation of these improvements.

Communication Link Recovery

The current state of the signal optimization algorithm assumes that the communication subsystem always has a link with all other network subsystems. It operates based on the current position that these subsystems send and their current signal-to-noise ratio. If a subsystem signal is lost, the values will stop updating. Essentially, the communication subsystem will think that the other subsystem has become static and continue to process the incorrect information. A more robust algorithm would be able to detect and react to a lost network subsystem.

Learning Signal-to-Noise Ratio Patterns

Another useful feature would be the ability to learn signal-to-noise ratio patterns. As mentioned, predicting signal-to-noise ratios is a difficult task and highly dependent on the environment. If the vehicle could learn patterns of SNR propagation based on distance and other variables, it could probably obtain a fairly good signal-to-noise ratio estimate for the desired location. Either that, or it could also be used as a flag for areas or conditions that would greater reduce the communication strength. For example, it could prevent the vehicle from traveling too far from a node after it learns that x meters away results in an extremely low signal.

Network Subsystem Path Projection

The final addition would be to incorporate the projected paths of the other network subsystems. Monitoring the paths could help the vehicle determine where to position itself. For example if a network subsystem is moving behind a building, its signal-to-noise ratio value is about to suddenly drop. Therefore, the vehicle could adjust for this subsystem movement before the SNR value drops below the signal-to-noise ratio threshold. Looking into the future of the network would provide better signal optimization efficiency.

References

[1] Cisco Systems, Inc. Cisco Aironet 802.11G 1100 and 1200 Series Access Point Upgrade Kit. Data Sheet, 2006. Available Online: http://www.cisco.com/

[2] L. Chaimowicz, A. Cowley, D. Gomez-Ibanez, B. Grocholsky, M.A. Hsieh, H. Hsu, J.F. Keller,V. Kumar, R. Swaminathan, and C.J. Taylor. Deploying Air-Ground Multirobot Teams in Urban Environments. Multirobot Workshop, Washington,DC, 2005.

[3] Defense Advanced Research Projects Agency. DARPA Grand Challenge Congressional Mandate, 2004. Available Online: http://www.darpa.mil/

[4] Ekahau, Inc. Ekahau Site Survey. 2006. Available Online: http://www.ekahau.com/

[5] Jim Geier. Minimizing 802.11 Interference Issues. Wi-Fi Planet Tutorials, January 11, 2002. Available Online: http://www.wi-fiplanet.com/

[6] J. Geier. Multipath a Potential WLAN Problem. Wi-Fi Planet Tutorials, May 14, 2002. Available Online: http://www.wi-fiplanet.com/

[7] Brett Gombar. Design and Evaluation of a Mobile Instrumentation Platform for Unmanned Vehicle Testing. Master's thesis, Virginia Tech, 2006.

[8] Ben Hastings. Design of a Micro Wireless Instrumented Payload for Unmanned Vehicle Testing. Master's thesis, Virginia Tech, 2006.

[9] ITT Industries Inc. ITT WMC - an Ad Hoc Networking Modem Card. Data Sheet, 2006. Available Online: http://www.acd.itt.com/

[10] B. Leedy, J. Putney, C. Bauman, S. Cacciola, J. Webster, and C. Reinholtz. Virginia tech's twin contenders: A comparative study of reactive and deliberative navigation. *Journal of Field Robotics*, 23.9, September 2006, 709-727.

[11] Brett Leedy. Two Minds for One Vehicle: A Case Study in Deliberative and Reactive Navigation. Master's thesis, Virginia Tech, 2006.

[12] Mobile ad-hoc network Wikipedia, The Free Encyclopedia. October 15, 2006. Available Online: http://en.wikipedia.org/wiki/MANET

[13] Motorola Inc. Mesh Network Technology. Motorola-North America-Government and Enterprise, 2004. Available Online: http://www.motorola.com/mesh/

[14] NetStumbler 2006. Available Online: http://www.netstumbler.com/

[15] Joseph Putney. Reactive Navigation of an Autonomous Ground Vehicle using Dynamic Expanding Zones. Master's thesis, Virginia Tech, 2006.

[16] Eric Schnell. Wireless Networking Technical FAQ. Information Systems, The Ohio State University, December 17, 2003. Available Online: http://is.med.ohio-state.edu/

[17] P. Ulam, and R.C. Arkin. When Good Comms Go Bad: Communications Recovery for Multi-Robot Teams. *Proc. IEEE Conference on Robotics and Automation*, April/May 2004, pp. 3727-3734.

[18] U.S. Geological Survey. Seamless Data Distribution System, Earth Resources Observation and Science. October 28, 2004. Available Online: http://seamless.usgs.gov/

[19] Virginia Center for Autonomous Systems. Heterogeneous Teams of Autonomous Vehicles - Advanced Sensing and Control. Proposal to the Office of Naval Research, Virginia Tech, 2005.

[20] A.R. Wagner, and R.C. Arkin. Internalized Plans for Communication-Sensitive Robot Team Behaviors. *Proc. IEEE Conference on Intelligent Robotics and Systems*, October 2003, pp. 2480-2487.

Appendix A

Acronyms

AGV Autonomous Ground Vehicle

AP Access Point

CEP Circular Error Probable

DARPA Defense Advanced Research Projects Agency

DEZ Dynamic Expanding Zones

GIS Geographic Information System

GPS Global Positioning System

IMU Inertial Measurement Unit

INS Inertial Navigation System

LIDAR Light Detecting and Ranging

LOS Line-of-Sight

MANET Mobile Ad-hoc Network

MARS Mobile Autonomous Robot Software

μWIP micro wireless instrument payload

MIP Mobile Instrumentation Platform

MLQ Multi-Link Quality

MOUT Military Operations in Urban Terrain

NQE National Qualifying Event

ONR Office of Naval Research

PID Proportional Integral Differential

QDMA Quadrature Division Multiple Access

RDDF Route Description Data File

RF Radio Frequency

SNR Signal-to-Noise Ratio

SSID Service Set Identifier

UAV Unmanned Aerial Vehicle

UDP User Datagram Protocol

UGV Unmanned Ground Vehicle

VaCAS Virginia Center for Autonomous Systems

VI Virtual Instrument

WLANs Wireless Local Area Networks

Appendix B

Signal-to-Noise Ratio Acquisition

B.1 802.11 SNR Acquisition

Steps to acquire SNR values from NetStumbler for real-time processing:

1. Create UDPsender DLL
   (a) Visual Studio - Language: C#
   (b) Enter the following code:

```
using System;
using System.Text;
using System.Net;
using System.Net.Sockets;

namespace UDPSender
{
  public class SendClass
  {
    public SendClass(){}
    public void Transmit(string hostname, int port, string message){
        UdpClient udpClient = new UdpClient();
          udpClient.Connect(hostname, port);
          Byte[] sendBytes = Encoding.ASCII.GetBytes(message);
          udpClient.Send(sendBytes, sendBytes.Length);
          udpClient.Close();
    }
  }
}
```

    (c) Build the DLL

2. Register UDPsender.dll File in Windows
    (a) Windows.NET framework version 1.1 or greater required
    (b) Copy UDPsender.dll to the Systems32 directory (C:\WINDOWS\Systems32)
    (c) Open Windows DOS prompt
    (d) Navigate to .NET framework directory (C:\WINDOWS\Microsoft.NET\Framework)
    (e) In this directory, run the command REGASM C:\WINDOWS\Systems32\UDPsender.dll, wait for completion confirmation message
    (f) Run the command GACUTIL -if C:\WINDOWS\Systems32\UDPsender.dll, wait for completion confirmation message
    (g) UDPsender.dll is now successfully registered
3. Create a VBScript Script File with the following code:

```
Function OnScanResult(SSID, BSSID, CapFlags, Signal, Noise, LastSeen)
        set UdpSend = CreateObject(“UDPsender.SendClass”)
        sndstr = ”ScanResult:&̈ SSID & “,” & BSSID & “,” & Signal & “,” & Noise
        UdpSend.Transmit “localhost”,61557, sndstr
End Function
```

4. In NetStumbler [14]
    (a) Install and open NetStumbler
    (b) Go to: View>>Options...
    (c) On the Scripting tab, set Type to ‘External Script’
    (d) Select the correct file
    (e) Click Apply
    (f) Check the Status box - it should say ‘Running’
    (g) Run NetStumbler
5. Setup program (LabVIEW Virtual Instrument (VI)) to input and parse User Datagram Protocol (UDP) message with SNR data

B.2 WMC630 SNR Acquisition

1. Install WMC6300 Software/Drivers (Mesh Installation CD)
2. Create meshAPI DLL
   (a) Visual Studio - Language: C++
   (b) Using Microsoft Foundation Class
   (c) Header File - meshAPI.h

```
// meshAPI.h : main header file for the meshAPI DLL
#pragma once

#ifndef __AFXWIN_H__
   #error include ’stdafx.h’ before including this file for PCH
#endif

#include “resource.h” // main symbols

#include <atlbase.h>
#import “meshAPI.tlb” no_namespace named_guids

// CmeshAPIApp
// See meshAPI.cpp for the implementation of this class
//
class CmeshAPIApp : public CWinApp
{
public:
   CmeshAPIApp();

   __declspec(dllexport) void GetMACAddress(char *mac);
   __declspec(dllexport) void GetNeighborTable(char* MAC, char* SignalStrength, char* Rate);

// Overrides
public:
   virtual BOOL InitInstance();

   DECLARE_MESSAGE_MAP()
};
```

(d) C++ File - meshAPI.cpp

```
// meshAPI.cpp : Defines the initialization routines for the DLL.
//
#include “stdafx.h”
#include “meshAPI.h”

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// data structures for raw neighbor table
typedef union
{
  unsigned char hw_addr[6];
  struct COMPARE_ADDR
  {
    unsigned short addr_1_2;
    unsigned short addr_3_4;
    unsigned short addr_5_6;
  } compare_addr;
} MAC_ADDR_TYPE;

typedef struct
{
  MAC_ADDR_TYPE dst_addr; // Full MAC address
  unsigned short dev_type; // Device Type: 100 = Subscriber Device,
             // 101 = Wireless Router,
             // 102 = IAP,
             // 104 = Non-Routing
  unsigned char lastRate; // 22.10 last data rate, from ATP data
  char signalLevel; // signalLevel from ATP data
} NT_INFO;

NT_INFO* pNT;

// variables for getting neighbor table data
char ntMAC[18];
unsigned char* ntMACbytes;
char sigLevel;
unsigned char lastRate;
unsigned long numNeighbors;

// CmeshAPIApp

BEGIN_MESSAGE_MAP(CmeshAPIApp, CWinApp)
END_MESSAGE_MAP()
```

```
// CmeshAPIApp construction
CmeshAPIApp::CmeshAPIApp()
{
}

// The one and only CmeshAPIApp object
CmeshAPIApp theApp;

// CmeshAPIApp initialization
BOOL CmeshAPIApp::InitInstance()
{
  CWinApp::InitInstance();

  return TRUE;
}

__declspec(dllexport) void GetMACAddress(char *mac)
{
  AFX_MANAGE_STATE(AfxGetStaticModuleState());

  CComBSTR bstrMyMAC; //No need to call SafeArrayDestroy, the CComVariant class
             // will clean itself up once it goes out of scope.

IMeshNetPtr pMeshNet; //Declare a smart pointer

USES_CONVERSION; // enables OLE string convertions

  // Initialize COM
  HRESULT hr = CoInitialize(NULL);
  if (FAILED(hr))
  {
  }
  if( SUCCEEDED(hr) )
  {
    // Create MeshNet
    HRESULT hr1 = pMeshNet.CreateInstance(“MeshAPI.MeshNet”);
    if( FAILED(hr1) )
    {
    }
    if( SUCCEEDED(hr1) )
    {
      // Get MAC Address for local Mesh Card
      HRESULT hr2 = pMeshNet->GetMAC( &bstrMyMAC );
      if (FAILED(hr2))
      {
```

```
        mac=“failed to get mac”;
      }
      if (SUCCEEDED(hr2))
      {
        char *tempstring = OLE2T(bstrMyMAC);
        int k;
           for( k=0; k ¡ strlen(tempstring); k++)
           {
             mac[k] = tempstring[k];
           }
      }
    }
  }
}
__declspec(dllexport) void GetNeighborTable(char* MAC, char* SignalStrength, char* Rate)
{
  AFX_MANAGE_STATE(AfxGetStaticModuleState());

  CComBSTR bstrMyMAC; //No need to call SafeArrayDestroy, the CComVariant class
              // will clean itself up once it goes out of scope.
  IMeshNetPtr pMeshNet; //Declare a smart pointer

  CComVariant vtRawNT;

  USES_CONVERSION; // enables OLE string conversions

  // Initialize COM
  HRESULT hr = CoInitialize(NULL);
  if (FAILED(hr))
  {
    MAC=“Failed COM Init”;
  }
  if( SUCCEEDED(hr) )
  {
    // Create MeshNet
    HRESULT hr1 = pMeshNet.CreateInstance(“MeshAPI.MeshNet”);
    if( FAILED(hr1) )
    {
      MAC=“Failed MeshNet Create”;
    }
    if( SUCCEEDED(hr1) )
    {
      // get neighbor table data
      HRESULT hr3 = pMeshNet-¿GetRawNeighborTable(&vtRawNT);
      if (FAILED(hr3))
      {
```

```
        MAC=“Failed GetRawNeighborTable”;
      }
      if (SUCCEEDED(hr3))
      {
        // per API documentation, get the number of table entries this way
        numNeighbors = vtRawNT.parray->rgsabound->cElements / sizeof(NT_INFO);

        // copy address of first array element to an NT_INFO pointer
        pNT = (NT_INFO*)vtRawNT.parray->pvData;

        // collect data for each neighbor in table and write to file
        for (UINT i=0; i¡numNeighbors; i++, pNT++)
        {
            // generate MAC address string
            ntMACbytes = pNT->dst_addr.hw_addr;
            sprintf(ntMAC,“%02X-%02X-%02X-%02X-%02X-%02X\0”,
               ntMACbytes[0],
               ntMACbytes[1],
               ntMACbytes[2],
               ntMACbytes[3],
               ntMACbytes[4],
               ntMACbytes[5]);

            sigLevel = pNT->signalLevel;
            lastRate = pNT->lastRate;

            for(UINT j=0; j¡18;j++)
            {
               MAC[i*17+j]=ntMAC[j];
            }

            SignalStrength[i]=sigLevel;

            Rate[i]=lastRate;
        }
      }
    }
  }
}
```

   (e) Need meshAPI.tlb - Mesh Installation CD>>Administration>>Documentation
   (f) Build the DLL

3. Download mfc71d.dll and msvcr71d.dll

4. Write software (LabVIEW VI) to call meshAPI.dll and parse message