Angle of Arrival Estimation using Received Signal Strength with Directional Antennas

An Honor Thesis

Presented in Partial Fulfillment of the Requirements for

the Degree Bachelor of Science with Distinction in Electrical and Computer

Engineering of The Ohio State University

By
Sean Winfree
****

The Ohio State University
2007

Examination Committee:

Dr. Lee C. Potter, Adviser
Dr. Randolph L. Moses

Abstract— This paper presents experimental results of angle of arrival (AOA) estimation for sensor node localization of wireless networks via received signal strength measurements from directional antennas. Field experiments were conduct with a Hex Dipole Parasitic Antenna (HDPA) board using an IEEE 802.11 2.4 GHz radio. A number of beam patterns of the HDPA are measured and stored as data sets organized by time and distance. The AOA estimates had large error for a single training data set; however when multiple data sets taken over longer time periods were averaged together the AOA error decreased. Intelligently choosing directional beam patterns is shown to improve the AOA performance while lowering complexity.

Table of Contents

1 Introduction

The localization of nodes in a sensor network is required in many applications of sensor networks. Directional antennas can enhance performance of a wireless network by providing a direct path from one node to another with minimal interference to other nodes. The utilization of directional antennas for routing requires the known location of the nodes in a network.

This paper presents field tests as proof of concept for the localization methods proposed in [2,3,4] specifically using directional antennas to compute AOA estimates. Node locations can be determined from the angle of arrivals (AOA) which are calculated from the ratios of received signal strength (RSS) of the radio signal using a switching directional antenna array. The switched directional antenna array allows for multiple beam pattern comparisons to evaluate the RSS ratios. This approach demonstrates a low cost and low complexity solution to node localization.

The RSS measurements are used for localization by calculating angle of arrival (AOA). Previous methods have used RSS to estimate distances for use in localization. However, distance measurements are difficult because of a lack of required processor speed to calculate the time of flight. Also the signal strength of a transmitter does not always decay at a rate of 1/(d^2) where d is the distance from transmitter to receiver.

RSS measurements used for AOA estimates are susceptible to a large variability due to propagation losses. Despite this the AOA estimates we consider will rely only on the relative RSS measured at the HDPA for different beam patterns, thus providing invariance to propagation loss

2 Theory

The directional antennas used in the field tests will be the Hex Dipole Parasitic Antenna Board (HDPA), a switched parasitic antenna design. The directional antenna array will be used to provide accurate AOA estimation. The array, in principle, should allow accurate AOA estimation and can be combined with the known localization algorithms to achieve sub-meter localization accuracy.

2.1 Localization Using AOA

The locations of the wireless nodes can be calculated by first estimating the AOAs, as described in [4], and then using th AOA to location algorithm in [3]. Both algorithms are low complexity; a requirement in most wireless networks. [1] This paper focuses on the AOA estimation leaving the localization aspect to [1, 2, 3].

An example of a wireless sensor network with HDPA arrays is present in **Figure 1**. This example shows how the individual nodes correspond to real world coordinates. The $\theta_M = 0^o$ of node M is reference to the world coordinate system (East = 0^o) by $\phi_{M|World}$.

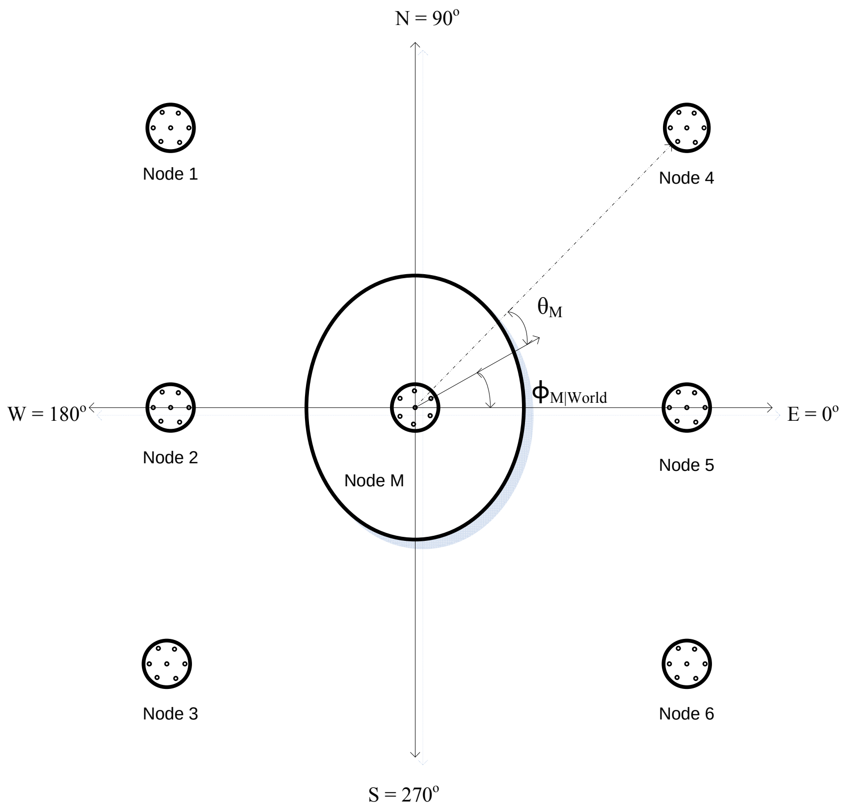


Figure 1: Localization example with seven HDPA arrays as nodes; Node M is used as the Main node to align the system to the World coordinates (East = 0°)

2.2 AOA with Directional Antennas

Directional antenna arrays allow for contrasting beam patterns over specific angles for use in AOA estimation. An example of two directional beam patterns can be seen in **Figure 2**. The antenna beam patterns, g1 and g2, are functions of θ which is the AOA such that the angles 0^o to 360^o are counter clockwise and $\theta=0^o$ is a constant reference point. The scalar U is the signal power (dBm) incident upon the antenna array from node i. The RSS measurements at angle θ of the directional antenna array are y1 and y2 where their respective beam patterns are $g_1(\theta)$ and $g_2(\theta)$.

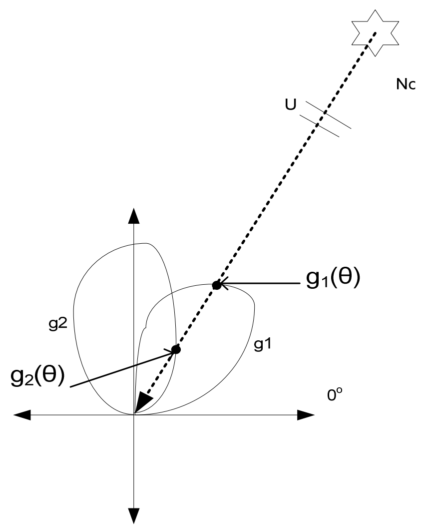


Figure 2: Directional beam AOA estimation example

The previous example can be easily expanded from two directional beam patterns to N beam patterns such that g_j (for j = 1:N) are functions of θ describing the beam patterns. This leads to N RSS measurements y_j (for j = 1:N). The vector n_j (for j = 1:N) is the noise associated with each RSS measurement. Thus we have the measurement model (1).

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{bmatrix} = U \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_N \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_N \end{bmatrix} \qquad (1)$$

The equation (1) is solved using a nonlinear least squares approach [4]. The solution is the minimum of the squared, normalized equation (2) solving for θ and U. G is a vector of g_j (for j = 1:N) and □ is a vector of y_j (for j = 1:N). The value y_j is the average of RSS measurements at an unknown θ. U is a single nuisance parameter returned by the nonlinear least squares estimator. The Matlab [8] function lsqnonlin was used to estimate the local minimums of expression (2).

$$\arg_{\theta,U} \min \left\| \overline{Y} - (U1 + G) \right\|^2 \qquad (2)$$

The model in equation (1) assumes additive white Gaussian noise, common incident power, U, for all antenna patterns, known antenna patterns, and a logarithmic scale for RSS measurements and antenna gain.

3 Hex Dipole Parasitic Antenna (HDPA)

The HDPA was designed with a single ¼ wave center antenna connected to the 802.11 radio surrounded by six evenly spaced parasitic antennas that can either be grounded or parasitic (floating). The six parasitic antennas are controlled via I/O lines to the switches. An example of the HDPA is discussed further in [2].

3.1 HDPA and AOA

The HDPA has six parasitic antennas which allows for N = 2^6 possible beam patterns. The spacing between antennas, Δ, is uniform for all six parasitic antennas. These patterns can be thought of as a six bit binary number from 000000b to 111111b. An example is shown in **Figure 4**, where the value 110101b describes the parasitic antenna combination used to form a beam pattern. The 1's (solid) represent an antenna that is grounded while the 0's (open) represent a parasitic/floating antenna.

There are 14 different binary combinations available with a six bit value and the rest are shifts around the antenna board of the other patterns. For example, 110101b and 111010b are considered the same patterns that are single bit shifts of one another. Most of the N antenna patterns should be 180^o, 120^o or 60^o shifts of other beam patterns.

The HDPA is aligned such that θ is the AOA of node i where angle $\theta = 0^o$ is antenna 1 and follows counter-clockwise to 360^o.

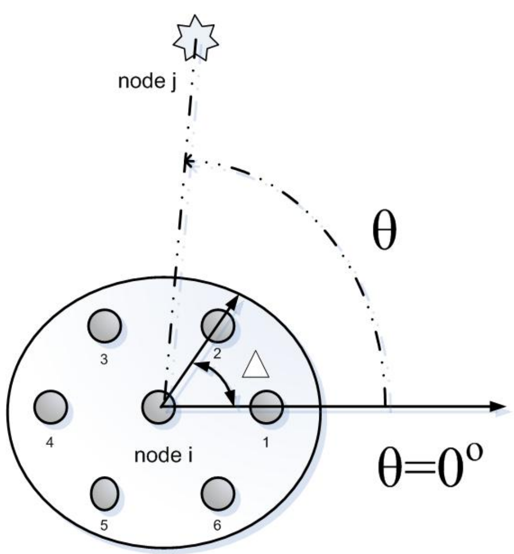


Figure 3: HDPA array; Center antenna is radiator; Antennas 1 through 6 are either parasitic(0) or grounded(1);

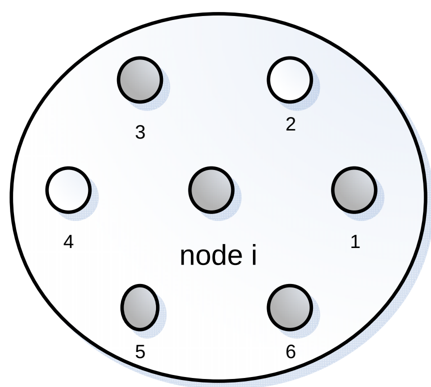


Figure 4: Example beam combination of HDPA board

3.2 HDPA Simulations

Simulations of the HDPA were run using ESP Workbench V1.1.18 [9]. The simulation file (*.esp) is located in section 8.8 in the appendix. These simulations demonstrated the HDPA's directional antenna beam patterns. All 64 beam patterns were simulated in order to choose a variety of different beam patterns for the field test. Examples are shown in **Figure 3**.

Notice the beam pattern shifting as the grounding of antennas change. Starting with 111110 where 1's are grounded and 0 are parasitic the simulation shows the bit shifting of a combination of parasitic and grounded antennas.

{111110} << 1 → {111101} << 1 → {111011} …

Notice the antenna beam pattern follows with the parasitic antennas in simulation. This may be because the grounded antennas are acting as reflectors and the parasitic antenna/s are left to direct the beam from the center radiator.

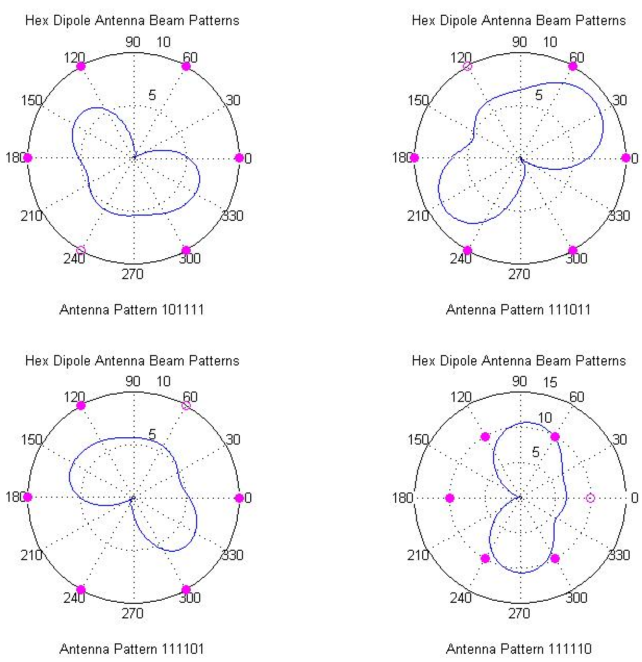


Figure 3: Simulations of HDPA run in ESP Workbench

3.3 Debugging

The HDPA was designed to be controlled with an up-counter, however, due to faulty design specifications this feature was not operational. To circumvent the counter the antenna I/O lines were controlled directly from the Stargate [6] as shown in Appendix 8.7. The scripts to control the I/O lines are seen in Appendix 8.1.

4 Experiments

Two tests were be run before AOA estimation: the reflection coefficient of the hex dipoles antenna board and a beam pattern/ RSS gathering with a single HDPA board.

4.1 HDPA Reflection Coefficient

The reflection coefficient of the antenna board was computed by using a network analyzer and measuring the S11 reflection coefficient in Log magnitude. The reason for obtaining the reflection coefficients is to guarantee the proper functionality of the antenna board before connecting to a transmitter. If the reflection coefficient is to large at the transmitting frequency the transmitter card can be damaged by too much reflected power. The reflection coefficients for the HDPA are shown in the Appendix 8.4.

4.2 HDPA Field Experiment Setup

The HDPA field experiment will measure the RSS over a 360^o rotation to find the switch-able parasitic antenna directive beam patterns. The Stargate I/O lines were used to electronically switch the parasitic elements, thereby selecting one of 64 beam patterns. This was done by placing the HDPA on a flat rotating surveyor tripod with single degree markings. A transmitter was placed at a known distance and the RSS is measured from the receiver as it is rotated in constant degree increments.

This test shows the antenna beam patterns. This is especially important for the HDPA antenna board because the beam patterns of the antenna board vary drastically based on the arrangement of the switching parasitic antennas. We chose N=12 different directive beam patterns for comparison of the RSS during the localization algorithm. It is very important that the antenna beam patterns be somewhat directional. **Figure 6** shows the HDPA board and **Figure 7** shows the HDPA as a receiver on a rotating platform used in the field test.

Figure 6: Hex Dipole Parasitic Antenna (HDPA) Board

Figure 7: Receiver Setup used in Field Test

4.3 HDPA Field Experiment

The HDPA board used for the tests had a defect such that antenna 2 was always a parasitic antenna. This limited the total number of possible antenna patterns to N = 2^5 and also limited the ability for complete set of rotated beam patterns of 60^o shifts to be evaluated. This will present problems down the road when the goal is to provide contrasting beam patterns for as many arrival angles as possible.

The Stargate [8] is connected to the HDPA board via the connector seen in the Appendix section 8.7. The 802.11b High Power Wireless PC card from SMC was used with the Stargate as the radio for the transmitter and receiver in the field test. The transmitter used only the SMC wireless card with its standard integrated antenna, but the receiver used the card in conjunction with the HDPA board. The HDPA was connected to the wireless card via a MMCXRARP-RG316/12-SMBSP cable through. The mmcx connection was made with antenna port 1 on the SMC card which is directly next to the card's LEDs.

The receiver setup comprised of the connector, cables, SMC wireless card, Stargate and HDPA were placed on a rotating platform that was marked 0^o to 360^o. The receiver remained in the same location throughout the experiment. The rotation platform was 1m from the ground and the field was relatively flat.

For reference to the plots in section 5, antenna 1 is set to 0^o. Three separate rotations were performed in 5^o increments. The transceiver was placed at a distance of 35m (Far) and 22m (Close). These three rotations are referenced as Far (35m), Far 2 (35m) and Close (22m) with each set taken 45 minutes apart. Twelve antenna patterns each with 1000 packets were collected during the field test. The 1000 RSS measurements recorded for single beam patterns were acquired in less than 100ms. The entire 12 beam patterns recording took approximately 25s per angle. The patterns were unfortunately limited in variety due to the defect to antenna 2.

The RSS measurements collected from the SMC wireless card were integer values. It was found to be best if those values remained between 50 and 90 because of the mapping of the wireless card's RSS integer values to dBm. This was accomplished by shutting off the automatic power adjust of the wireless card and setting the power accordingly. The script used in the transmitter for power control in Appendix section 8.1 as 'wipwr.'

A detailed instruction to the field test along with the scripts used in the Stargate can be found in the Appendix in sections 8.5 and 8.1 respectively. The **Figure 8** shows in detail the layout of the transmitter and receiver along with any significant geographical features.

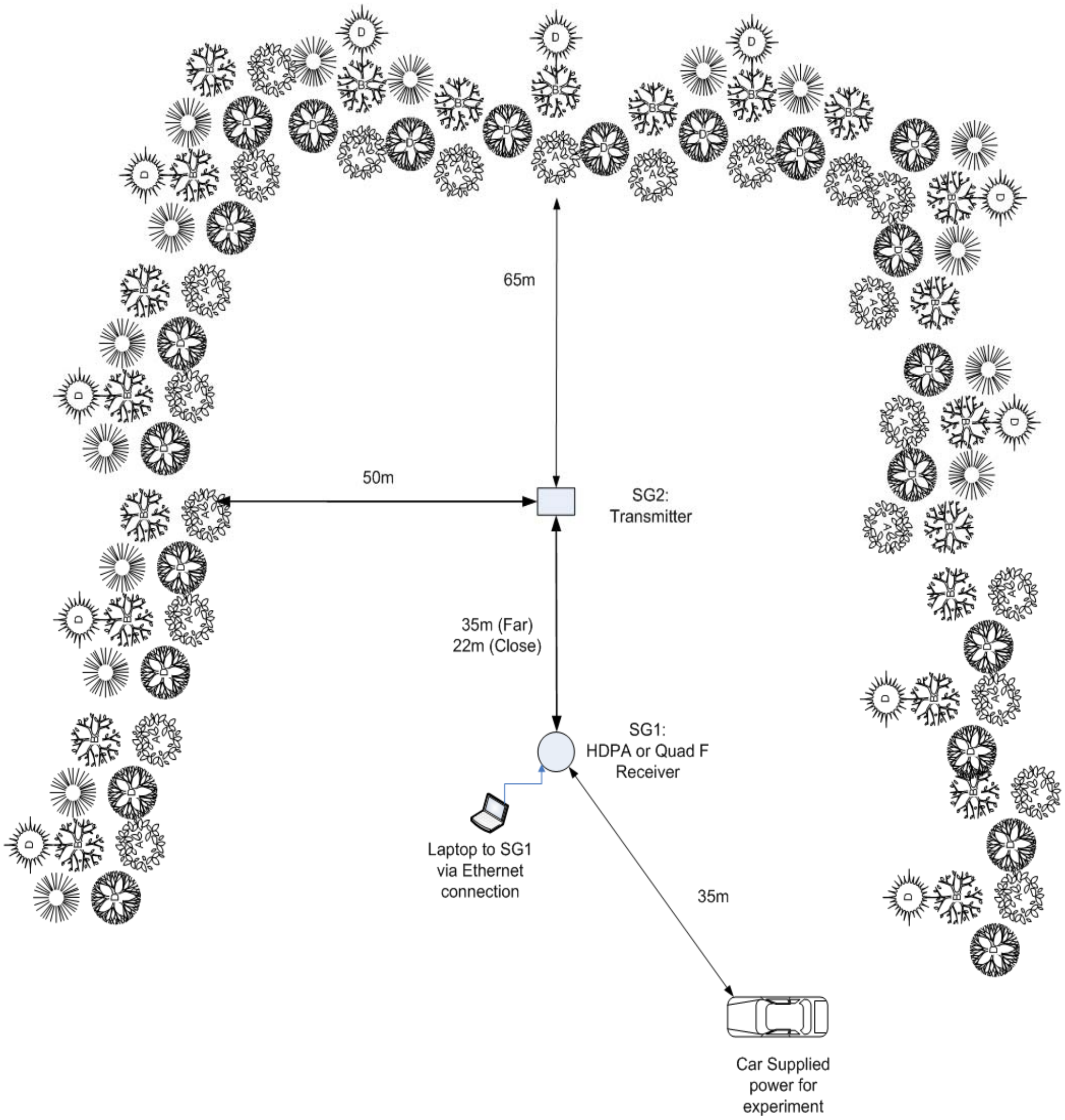


Figure 8: Field Test Layout for gathering data sets: Far (35m), Far 2 (35m), and Close (22m)

5 Results

5.1 HDPA Beam Patterns

The RSS measurements for the 12 antenna pattern combination collected during the field test are plotted in **Figure 9**. These twelve patterns are listed and shown in Appendix section 8.3. Notice that there is little variation of RSS in the beam patterns in the 150^{o} to 200^{o} region. The best case scenario for AOA estimation is to have two or more contrasting patterns with sharp, opposite sloped RSS measurements for all angles. When the beam patterns being evaluated for AOA estimation have similar characteristics the estimation error increases. The lack of contrasting patterns over certain angles will present a problem throughout the AOA estimation.

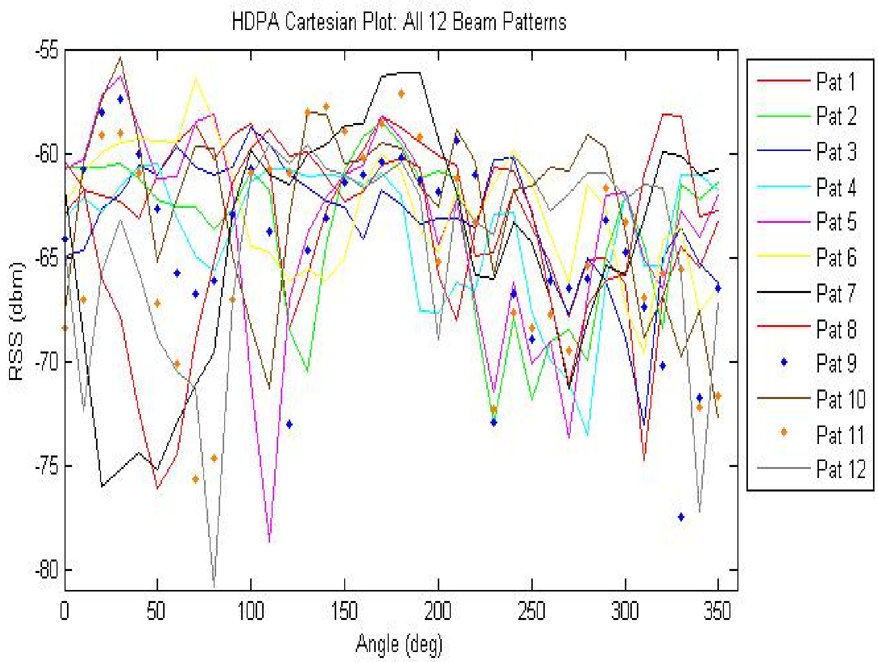
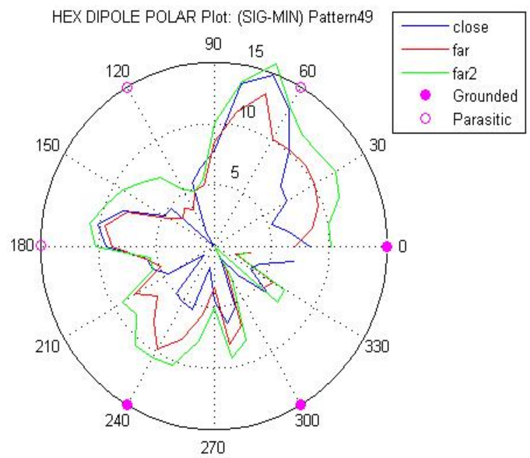


Figure 9: Twelve beam patterns plotted from data set Far (35m)

The two beam patterns shown in **Figure 10** and **Figure 11** are examples of the 12 beam patterns evaluated during the field test. The two antenna pattern combinations in question are 111000b and 110001b which are single bit shifts of one another. All three data sets are plotted in the figures; however the patterns are self normalized to the individual minimuma to fit on a polar plot. The plots are to show the variations in RSS measurements between the three data sets.



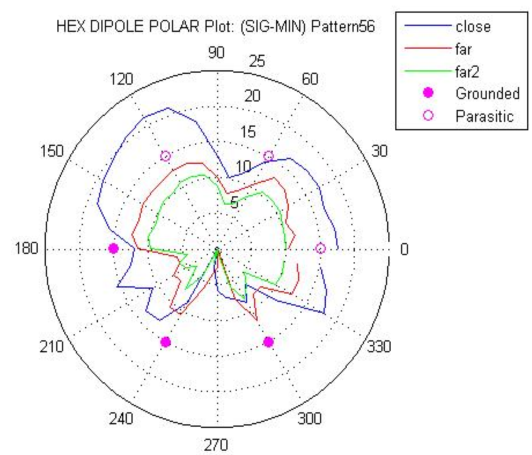


Figure 10: Three data set plots of antenna pattern 111000b

Figure 11: Three data set plots of antenna pattern 100011b

An approximate 60^o shift can be seen in comparing **Figure 10** and **Figure 11** as the combination pattern shifts from 110001b to 111000b. The beam patterns are not exact shifted copies but they are similar. This shifting effect is not assumed to occur for all antenna combinations and as such throughout the estimation each pattern is thought of as independent from one another.

5.2 Spline Fitting of Beam Pattern

During the field test in section 4.2 data points were collected for each beam pattern at increments of 5^o. To allow estimation at sub-degree increments the data set is interpolated using a fitting algorithm the Matlab [8] function, Spline, to fit the training data sets to function $g(\theta)$.

Spline was chosen because it is a simple to use piecewise polynomial fitting algorithm. The **Figure 12** shows the seventy two data points collected in data set Far for a particular antenna pattern combination. These data points are fit using the Spline function and the resulting function is plotted. The fitted Spline function is stored as $g_j(\theta)$ which is a part of the cost function in (2). The $g_j(\theta)$ functions are created from the data set referred to as "train" for all N antenna beam patterns evaluated.

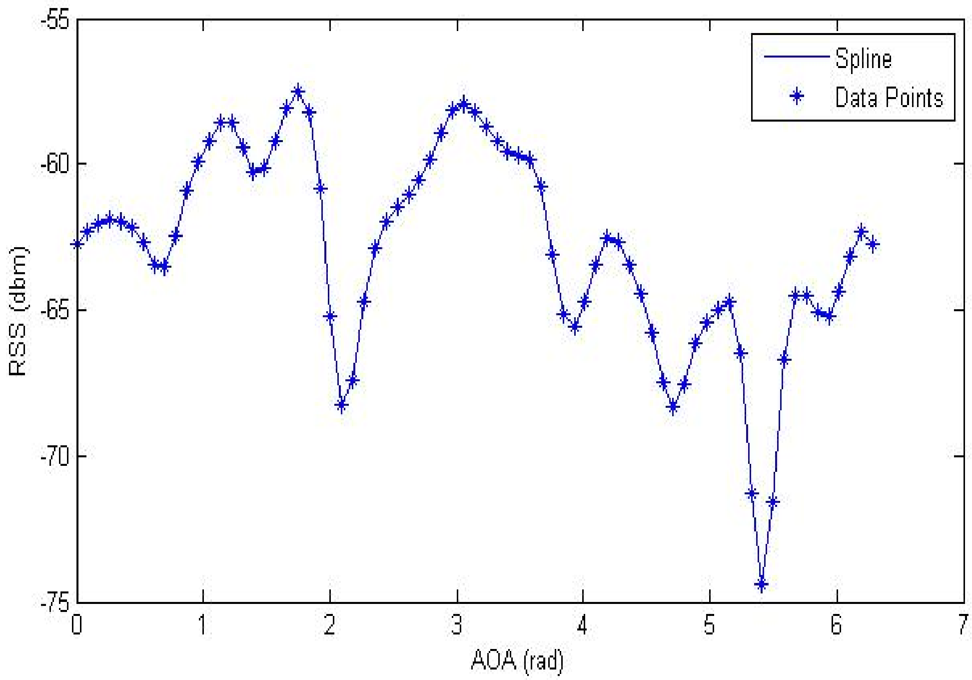


Figure 12: Spline of 72 data points describing a beam pattern from data set Far.

5.3 AOA Estimation

The AOA estimation was done using the algorithm presented in [4]. The training pattns ($g(\theta)$ functions) were computed with the data set Far unless otherwise specified.

5.3.1 AOA Performance for Three Data Sets

Train equal test is when the data set to create the $g_j(\theta)$ functions is the same as the data set with the RSS measurements y_j. The case were the same data used to Train is used to test can be thought of as a best case scenario. **Figure 13** is an example of a best case scenario where train and test are calculated from the Far data set. Notice that the figure shows the maximum AOA estimation error to be less then 1.2 degrees.

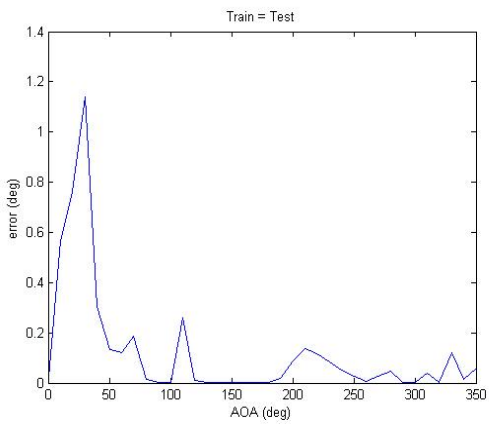


Figure 13: Train = Far; Test = Far

The three data sets recorded from the field test are used to evaluate AOA estimation performance as shown in **Figure 14**. The training data set Far is compared against Far, Far 2, and Close as test data. Observer the large error in AOA estimation for a data set at a different time or different distance between the HDPA and transmitter.

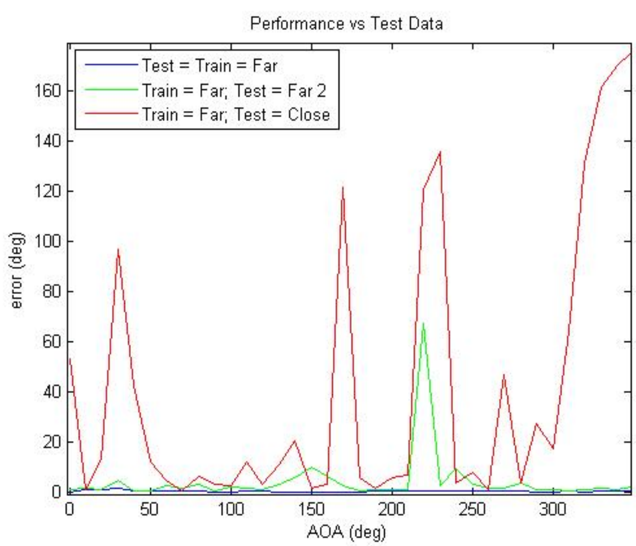


Figure 14: AOA estimation for Train = Far compared to all three data sets (Far, Far 2, Close)

5.3.2 AOA Performance vs. Number of Antenna Beam Patterns

AOA estimation theory suggests that as the number of patterns, N, increases the error goes down. [4] This assumes that added beam patterns offer different RSS measurements over many angles. If the added patterns are too similar then AOA performance will plateau or decrease with added complexity. To illustrate consider the case of AOA estimation using two antenna beam patterns versus 12 antenna beam patterns as shown in **Figure 15**. Notice that the AOA estimation error is much less when 12 beam patterns are used as compared with only 2 beam patterns.

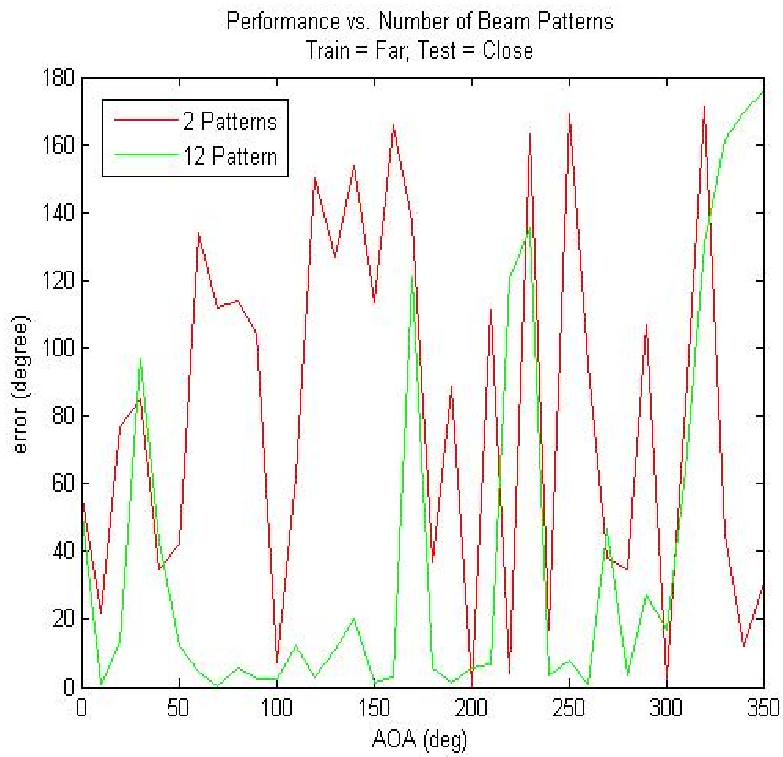


Figure 15: AOA estimation performance vs. Number of beam patterns

5.3.3 AOA Performance vs. Number of RSS Packets

Theory suggests that as the number of packets increase so should the AOA estimation performance [4]. This assumes that the training beam pattern is known and variations in RSS measurement from packet to packet are due to zero mean additive noise.

However during the field test the RSS strength measurements recorded rarely varied more than a single integer from the average. This could be the effect of averaging by the SMC wireless card. It is also possible that the RSS measurements were recorded too quickly. For the field test, the 1000 RSS measurement were recorded for a single beam pattern in less than 100ms. The entire 12 beam patterns recording took approximately 25s per angle.

A comparison of 200 packets to 1000 RSS measurement is computed for Train = Far and Test = Close using all twelve beam patterns can be seen in **Figure 16**. The performance is almost exactly the same for all AOA estimates. This shows that there is little variance within the 1000 RSS measurement data set taken in < 100ms.

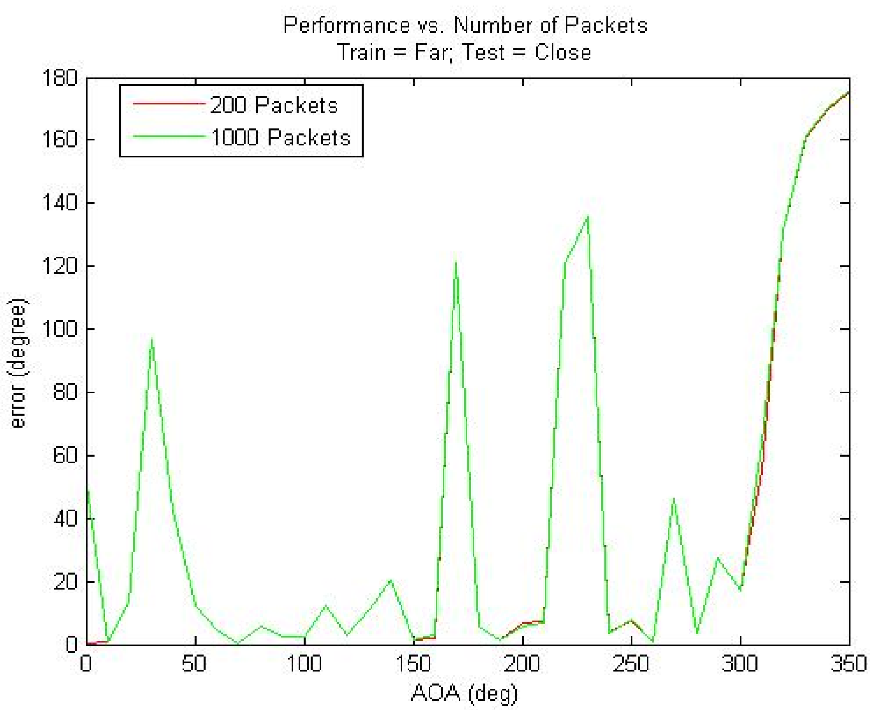


Figure 16: Comparison of AOA estimation performance for 200 RSS measurements vs 1000 RSS measurements.

5.3.4 AOA Performance with True Beam Pattern

The previous performance evaluations were based on the assumption that the data set Far was sufficient to use for training. This training data set however produced large error in the AOA estimation when compared to the other data sets. A possible cause for this large error could be that the data set used for Train does not properly describe the 12 beam patterns being evaluated.

The RSS measurement from the two data sets, Far and Far 2, were averaged together. The average 1000 RSS measurements of data set Far are averaged with the mean of data set Far 2. These two data sets were taken with the same transmitter/receiver placement as described in section 4.2 but 45 minutes apart. The averaging of these two data sets will reduce the likelihood that a single set of 1000 RSS measurement set being averaged far from the true beam pattern. These averaged sets are then used for the train, using Spline to find g(θ).

The performance of the training data created from the average of the two data sets compared against the train from a single data set can be seen in **Figure 17**. All 12 beam patterns were used for this performance evaluation. It can be seen that the averaging of the data sets allowed for greater performance of AOA estimation. This suggests that the data set Far had outlying RSS measurements for the entire 100ms sampling time per beam pattern. This suggests that the average of 1000 RSS measurements over a short time can be an outlying value on the antenna beam pattern and not correlate to the true gain of the antenna at that angle. Another thing to note is that an averaging of RSS measurements over a longer time period then 100ms should result in a truer representation of the direction antenna beam patterns.

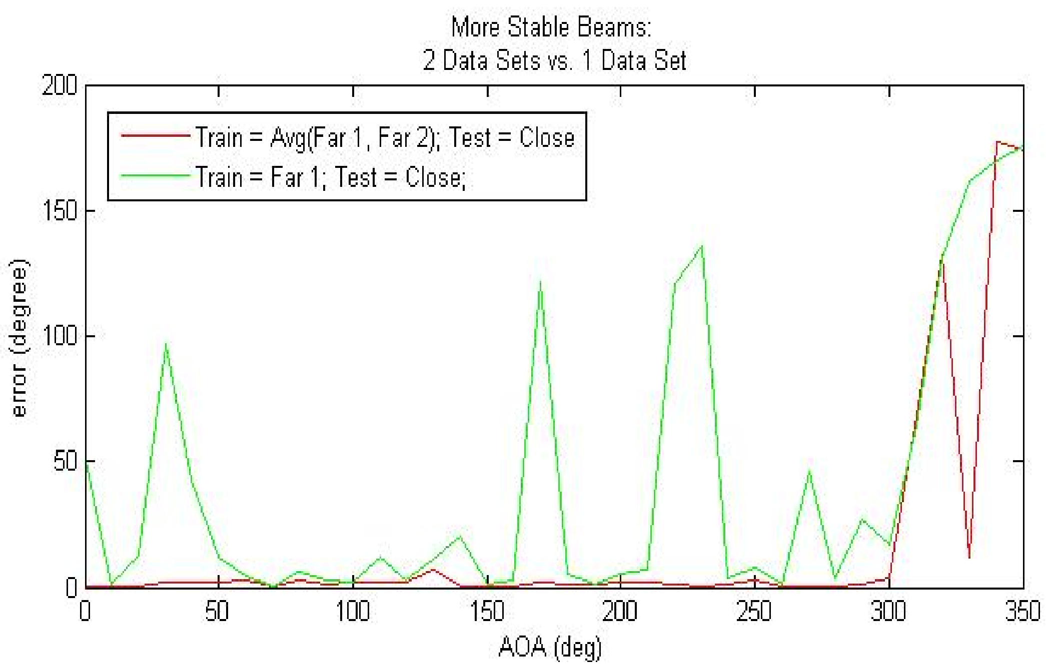


Figure 17: Combined data sets for Train vs. Single data set for Train

5.3.5 AOA Performance with Intelligently Chosen Beam Patterns

The HDPA provides 64 possible antenna beam patterns. Many of these beam patterns are similar to each other. To maximize AOA estimation performance it is important to have contrasting beam patterns for all AOA. The field test was limited to twelve beam patterns and due to a faulty antenna 2 these 12 beams could not be chosen to ensure complete 360^o coverage by contrasting beam patterns as discussed in section 5.1.

Eight beam patterns from the 12 gathered in the field test were chosen by visual inspection on the condition of least redundancy. These eight patterns were compared to eight randomly chosen beam patterns and compared in **Figure 18**. The average error for intelligently chosen beam patterns was 50.5904 degrees compared against randomly chosen beam patterns average error of 76.6543 degrees.

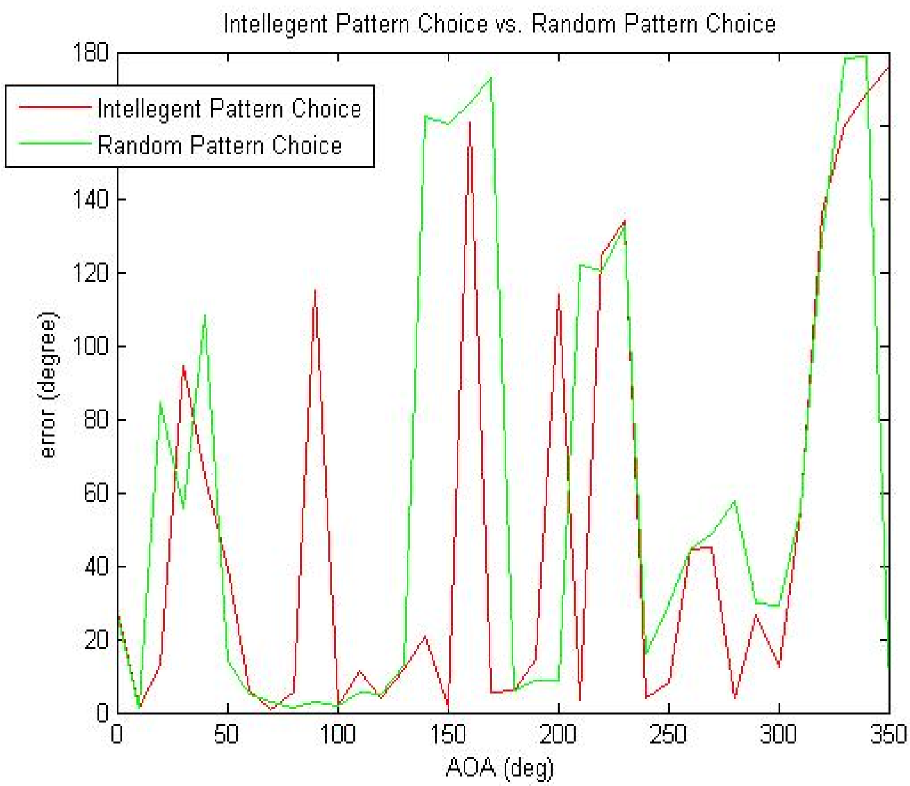


Figure 18: Compare intellegently chosen beam patterns vs. randomly chosen beam patterns for Train = Far and Test = Close

6 Conclusions

The field testing of the Hex Dipole allowed for RSS measurements from directional antennas over a 360 degree radius. This data was used as proof-of-concept to show that the RSS can easily be converted to AOA using the algorithm in [4]. Also the directional antennas offered a varied enough beam pattern to compute accurate AOA estimation. The AOA estimation can then be used with the algorithm in [3] to calculate localization. These two algorithms in conjunction with directional antenna board designs will offer low cost and low complexity solutions to localization.

The HDPA antenna boards used are large and costly. This paper presents a proof of concept for AOA estimation with directional antennas and leaves the low cost solution to future work.

This paper shows the importance of finding a true representation of the directional antenna combination beam patterns. Using a longer time period for averaging of the RSS measurements should give a truer representation of the directional antenna patterns. Future work will be done to see if gathering the RSS measurement over longer time periods yields a better training data set. The AOA performance impact can then be measured using in test data sets.

Intelligently chosen antenna beam pattern combinations impacts the performance of AOA estimation. Intuitively it is known that computation of redundant beam patterns add getter complexity with minimal benefit. Finding a base for beam pattern comparison and their resulting contribution to AOA estimation would be a great benefit to using directional antennas.

7 References

[1] N. Patwari, J. Ash, S. Kyperountas, A. Hero III, R. Moses, and N. Correal, Locating the Nodes, *IEEE Signal Processing Magazine*, July 2005, pp 54-69.

[2] E. Taillefer, A. Hirata, T. Ohira, Direction-of-Arrival Estimation using Radiation Power Pattern with an ESPAR Antenna, *IEEE Transactions On Antennas and Propagation*, VOL 53, No. 2, Feburary 2005, pp 678-684.

[3] Joshua Ash, Lee Potter, Robust System Multiangulation Using Subspace Methods, In Submission*:* Information Processing in Sensor Networks. 2007

[4] Joshua Ash, Lee Potter, Sensor Network Localization via Received Signal Strength Measurements with Directional Antennas, (*Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*), pp 1861-1870, Monticello, IL, Sep. 2004.

[5] L.Potter, J. Ash, H. Romero, Sensor Network Localization using a Printed Circuit Antenna Array, Unpublished Manuscript, July 2005.

[6] Crossbow Technology Inc. "Stargate Data Sheet", May 2005, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/6020-0049-01_B_STARGATE.pdf.

[7] SMC Networks, "802.11b High Power Wireless PC card", May 2005, http://www.smc.com/index.cfm?event=viewProduct&localeCode=EN_USA&pid=346

[8] Mathworks. MATLAB version 7 (R2007a) ; with DSP, statistics, and optimization toolboxes. www.mathworks.com

[9] Ed Newman and Frank Payeter

ESP Workbench V1.1.18, Ohio State University

8 Appendix

8.1 Bash Scripts

Control_Hex

```
#!/bin/bash
# Author: Sean Winfree and Josh Ash
# Pre:  When calling fnct provide <num_packets\> and \<angle\>
# Example of use:  ./control_hex 600  180

if [ -z "$2" ]; then
   echo usage: $0 \<filename\> \<num_packets\>
   echo usage: $0      2 arguments total
   exit
fi

./ant_pattern 61
FILENAME=/mnt/cfcard/antpat61angle$2.txt
NUM_PACKETS=$1
echo $NUM_PACKETS
ANTENNA=2
IP=10.0.1.22
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 56
FILENAME=/mnt/cfcard/antpat56angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 60
FILENAME=/mnt/cfcard/antpat60angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 28
FILENAME=/mnt/cfcard/antpat28angle$2.txt
NUM_PACKETS=$1
echo $NUM_PACKETS
ANTENNA=2
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
```

```
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 49
FILENAME=/mnt/cfcard/antpat49angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 1
FILENAME=/mnt/cfcard/antpat1angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 4
FILENAME=/mnt/cfcard/antpat4angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 41
FILENAME=/mnt/cfcard/antpat41angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 52
FILENAME=/mnt/cfcard/antpat52angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 57
FILENAME=/mnt/cfcard/antpat57angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME
```

```
./ant_pattern 21
FILENAME=/mnt/cfcard/antpat21angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./ant_pattern 45
FILENAME=/mnt/cfcard/antpat45angle$2.txt
echo $NUM_PACKETS
echo $ANTENNA
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME
echo DONE
```

Control_quad

```
#!/bin/bash
# Author: Sean Winfree and Josh Ash
# Pre:  When calling fnct provide <num_packets\> and \<angle\>
# Example of use:  ./control_quad 600  180

if [ -z "$2" ]; then
   echo usage: $0 \<filename\> \<num_packets\>
   echo usage: $0      2 arguments total
   exit
fi

./quad_ant_pattern 0
FILENAME=/mnt/cfcard/antpat0angle$2.txt
NUM_PACKETS=$1
echo $NUM_PACKETS
ANTENNA=2
IP=10.0.1.22
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

./quad_ant_pattern 1
FILENAME=/mnt/cfcard/antpat1angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME
```

```
./quad_ant_pattern 2
FILENAME=/mnt/cfcard/antpat22angle$2.txt
echo $NUM_PACKETS
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME


./quad_ant_pattern 3
FILENAME=/mnt/cfcard/antpat3angle$2.txt
NUM_PACKETS=$1
echo $NUM_PACKETS
ANTENNA=2
prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
sleep 1
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME

echo DONE
```

Hex_ant_pattern

```
####################################################
## Script: hex_ant_pattern
## Author: Sean Winfree
## DATE  : Jan 21, 2007
#
## Description: Bash script used to control the Hex
#               Dipole parasitic antenna array.
#      MUST BE IN STARGATE
#
#    Uses $1
####################################################
#!/bin/bash
########### SET ANTENNA PATTERN #####################
Logic=(1 2 4 8 16 32 ) # all possible values need to logic and
####################################################
echo "SETTING UP ANTENNAS."
pant=65
VL=1
CNT=0
# allow for 6 itterations
while [ $CNT -lt 6 ]; do

      # create a gpio for each entry
      echo r$pant > /proc/platx/gpio/GPCTL

      # value is 1 equals set line
      if [ $(($1 & ${Logic[$CNT]})) == ${Logic[$CNT]} ]; then
            echo s$pant > /proc/platx/gpio/GPCTL
            #set the gpio line
      else              # value is 0 equals clear  line
            echo c$pant > /proc/platx/gpio/GPCTL    #set the gpio line
      fi                # endif
      let pant=pant+1      #increment parasitic antenna # (change GPIO
LINE)
      let CNT=CNT+1        #inc itteration counter
      let VL=VL+1    #increment for next antenna value
done                 # end if
###### END ant_pattern ##########################################
```

quad_ant_pattern

```
## script: quad_ant_pattern
## Author: Sean Winfree
## DATE  : April 1, 2007
#
## Description: Bash script used to control the Quad F
#   antenna array.
#       MUST BE IN STARGATE used as receiver
#
#   Uses $1  possible values to pass are 0,1,2,3 to select
#         one of the four antennas
#####################################################
#!/bin/bash
########### SET ANTENNA PATTERN #####################
Logic=(1 2 4 8 16 32 ) # all possible values need to logic and
#####################################################
echo "SETTING UP ANTENNAS."
pant=65
VL=1
CNT=0
echo r67 > /proc/platx/gpio/GPCTL   # create a gpio for each entry
echo c67 > /proc/platx/gpio/GPCTL    #set the gpio line 67 which is enable
# allow for 6 itterations
while [ $CNT -lt 6 ]; do
        # create a gpio for each entry
        echo r$pant > /proc/platx/gpio/GPCTL
        # value is 1 equals set line
        if [ $(($1 & ${Logic[$CNT]})) == ${Logic[$CNT]} ];
        then
                echo s$pant > /proc/platx/gpio/GPCTL    #set the gpio line
            cat /proc/platx/gpio/gpio$pant
        else                                    # value is 0 equals clea
                echo c$pant > /proc/platx/gpio/GPCTL    #set the gpio line
               cat /proc/platx/gpio/gpio$pant                  #confirm setting
        fi                                    # endif

        let pant=pant+1    #increment parasitic antenna # (change GPIO LINE)
        let CNT=CNT+1      #inc itteration counter
        let VL=VL+1        #increment for next antenna value
done                    # end if
echo    “DONE”
###### END ant_pattern ############################################
```

takemeas_fast

```
# Author: Josh ash
# Pre:  When calling fnct provide \<filename\>  and <num_packets\>
# Example of use:  ./takemeas_fast /mnt/cfcard/yagi2angle180 600

#!/bin/bash
if [ -z "$2" ]; then
   echo usage: $0 \<filename\> \<num_packets\>
   echo usage: $0      2 arguments total
   exit
fi
## Set Constants
FILENAME=$1.txt
NUM_PACKETS=$2
echo $NUM_PACKETS
ANTENNA=2
IP=10.0.1.22              ## Must be transmitter IP

prism2_param wlan0 antsel_rx $ANTENNA
/root/rsslog wlan0 --reset
ping -i 0 -l 300 -c $NUM_PACKETS -w 2 $IP > /dev/null
/root/rsslog wlan0 --dump > $FILENAME
```

Wipwr

```
# code to turn of power control on transmitter and set it to a set level
# pre: <pwr_setting>  integer number from 0 to 255
# example ./wipwr 120
prism2_param wlan0 alc 0
iwpriv wlan0 writemif 62 $1
```

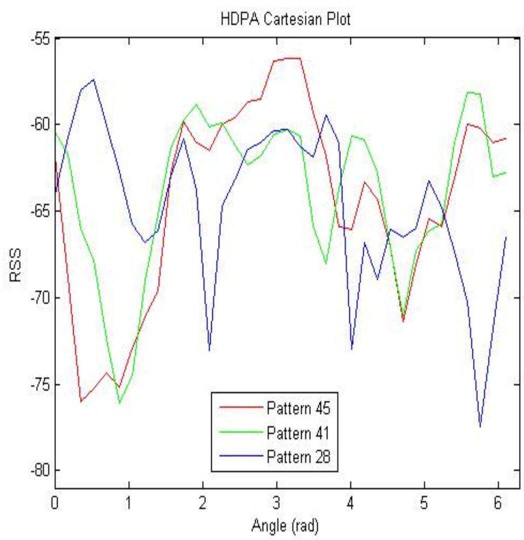
8.2 Matlab Code

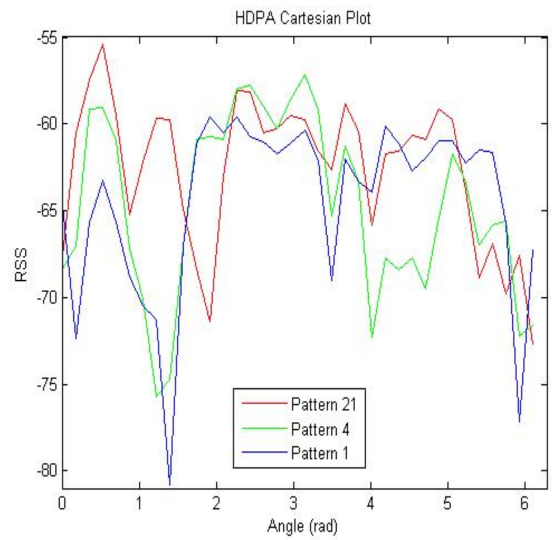
make_beam_pattern.m

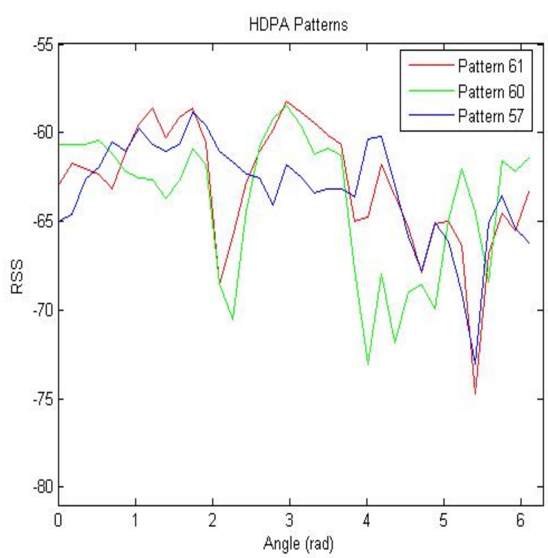
```
%% make_beam_pattern.m
% Author: Sean Winfree
% Date: March 28, 2007
%
% Description: inports text files of RSS for the antenna
% beam pattern using columns2mat.m
% naming convention is
angle$((ANGLE_CNTR))antpattern$((pat$ANT_CNTR)).txt
% where ANGLE_CNTR= 0:350 and can be ANT_CNTR = 1:20

%% Antenna Array file numbers (adjust for chosen patterns)
ANT_NUM = [63,32,16,8,4,2,1,48,24,12,6,3,33,50,25,45,22,11,37,0];
pic=1;
for ANT_CNTR = 1:20;    %for all values in array ANT_NUM
    ant = num2str(ANT_NUM(ANT_CNTR));     %convert to string
    i=1;                         % array counter
    for ANGLE_CNTR = 0:10:350;    % for 0->350 every 10deg
        angle = num2str(ANGLE_CNTR);    %convert to string
        filenm = ['angle' angle 'antpattern' ant '.txt'];
        [src_addr, rssi] = columns2mat(filenm);
        signal(i) = mean(rssi);
        i=i+1;
    end     % end angles
    figure(pic);
    pic=pic+1;
    %t = 0:.0174:2*pi;  % for 360 angles in array
    t = 0:.175:2*pi;  % for 36 angles in array
    polar(t,signal,'--r')
    title(['RSS vs Angle for ' filenm]);
    xlabel('Angle (deg)');
end     %end antenna counter
%%
```

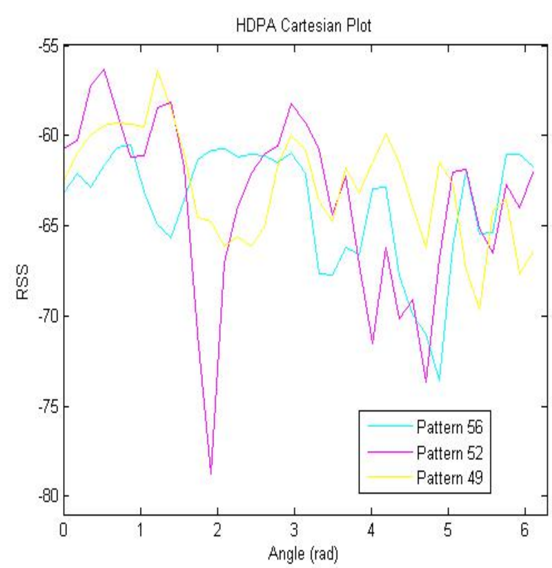
8.3 Beam Pattern Plots

Hex Dipole Beam Pattern Cartesian Plots



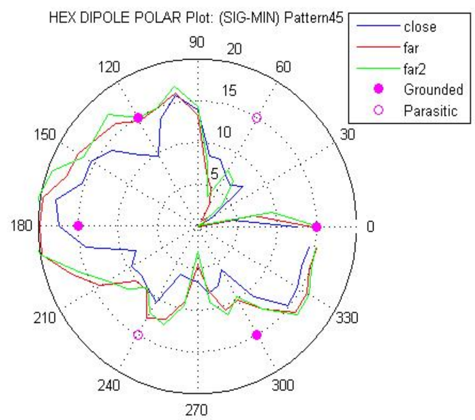


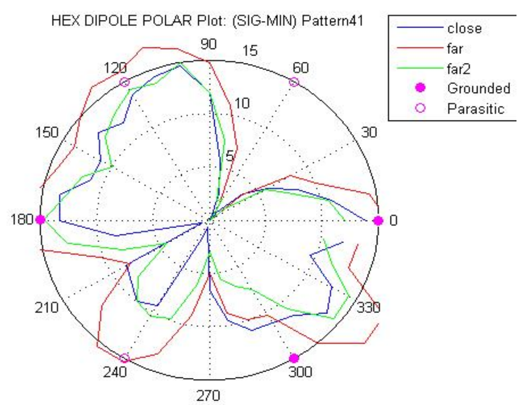


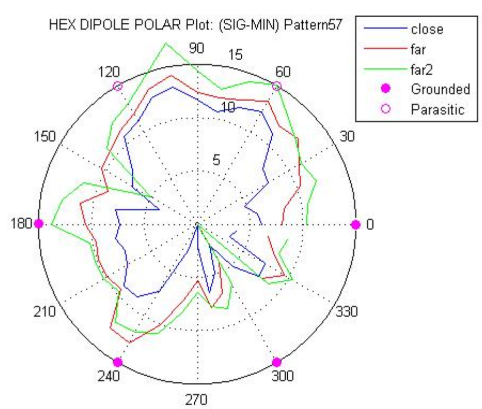


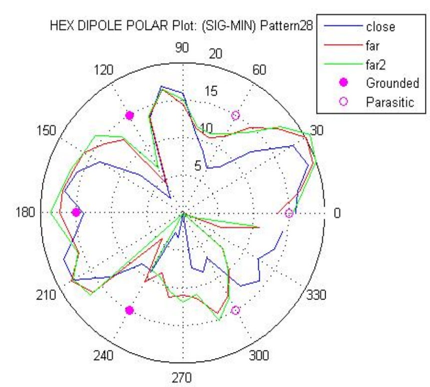
Hex Dipole Beam Pattern Polar Plots

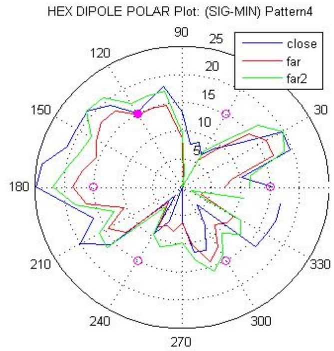
The signal normalized to its own minimum and plotted in polar form in relation to it beam pattern.

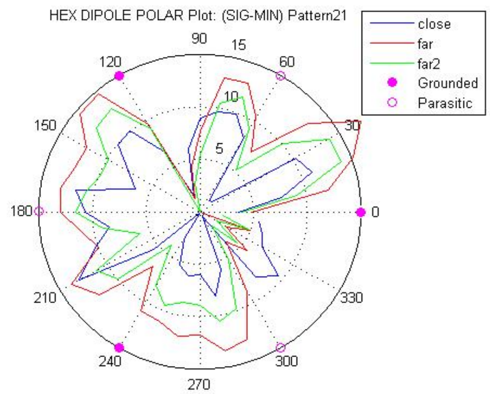


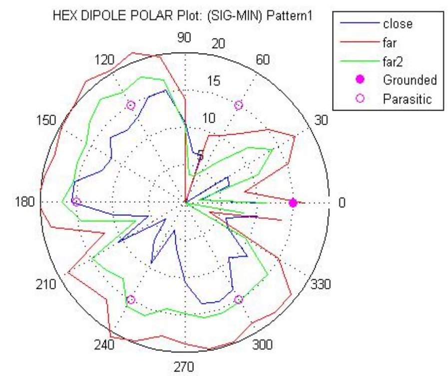


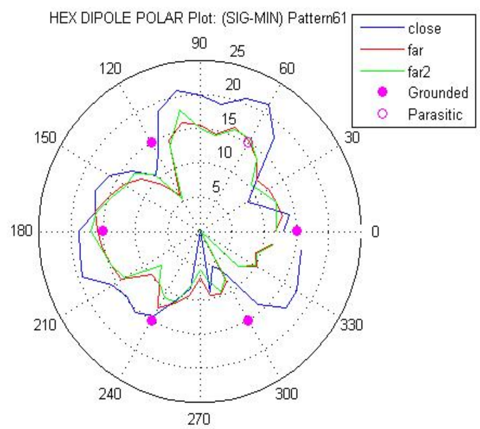


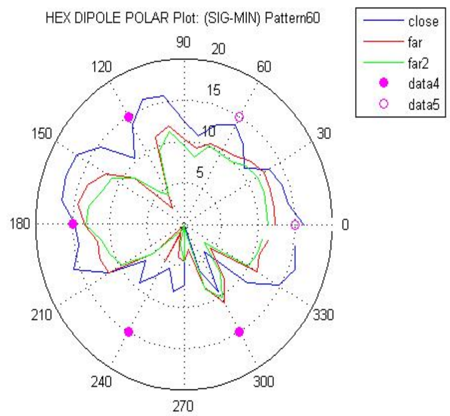


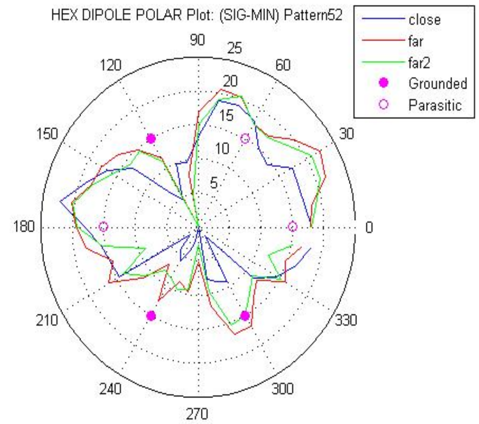

HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern4
close
far
far2

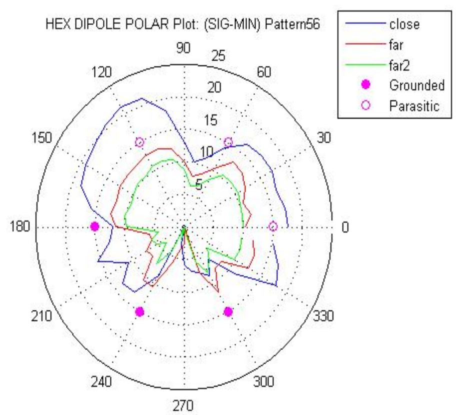

HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern21
close
far
far2
Grounded
Parasitic

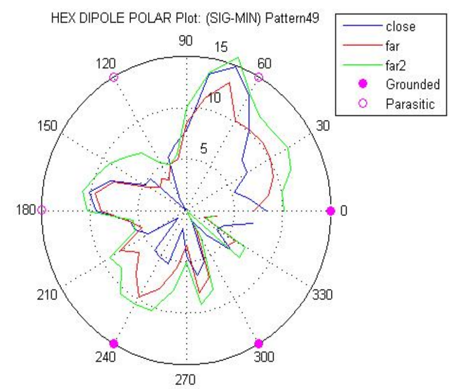

HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern1
close
far
far2
Grounded
Parasitic


HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern61
close
far
far2
Grounded
Parasitic


HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern60
close
far
far2
data4
data5

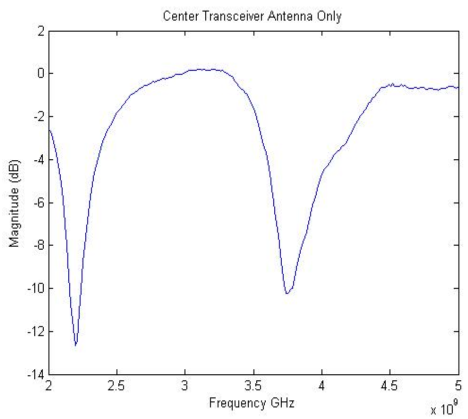

HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern52
close
far
far2
Grounded
Parasitic

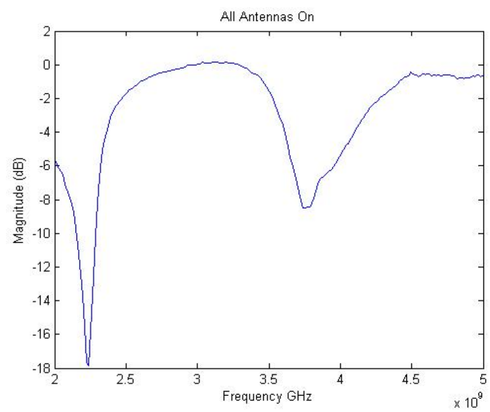

HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern56
close
far
far2
Grounded
Parasitic


HEX DIPOLE POLAR Plot: (SIG-MIN) Pattern49
close
far
far2
Grounded
Parasitic

8.4 Reflection Coefficients

Hex Dipole Antenna Board Reflection Coefficients





8.5 Beam Pattern Field Test

Instructions:

The following instructions will provided a step by step guide to reproduce to field test done to obtain the beam pattern of the directional antenna arrays. It will be assumed that the reader has access to all the equipment and software listed. It is also assumed a basic understanding of Cygwin (though not significant) and Matlab.

Equipment:

1) Stargate Gateway (Xbow) (2 required)
2) 1.0GB CompactFlash SD
3) 2.4GHz 802.11b High Power Wireless PC Card
4) Ethernet Crossover cable
5) Hex Dipole Parasitic Antenna Array
6) Quad F Antenna Board
7) GPIO connector Stargate -> HDPA (see schematic)
8) PC w/ SSH and Cygwin
9) Rotating tripod with degree markings for 0 to 360 degrees.
10) Antenna cable: MMCXRARP – RG316/12 – SMBSP

SSH client can be used to communicate with SG1 and SG2. Set up a laptop/PC to have a IP address of 10.0.0.1 with subnet 255.255.255.0. Set SG1 and SG2 to IP address 10.0.0.21 and 10.0.0.22 for Ethernet and 10.0.1.21 and 10.0.1.22 for wireless. Then the SG1 and SG2 can be log onto with an account.

Steps:

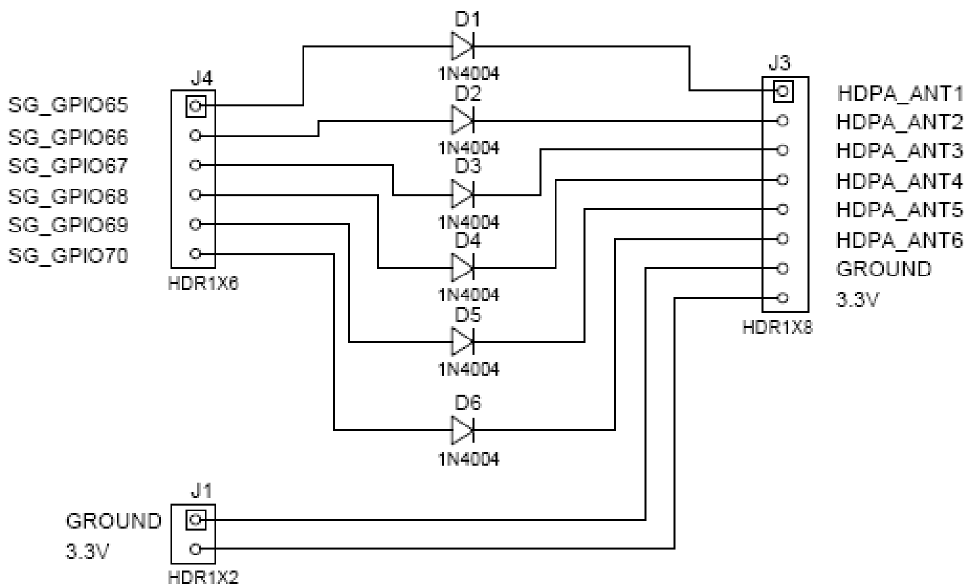
1) Store files ant_pattern and control_hex to Stargate Gateway 1 (SG1).
2) Store control_quad
3) Mount the memory card on SG1 using SSH
   a. stargate2:/root# mount /mnt/cfcard // mount the card
   b. stargate2:/root# cd /mnt/cfcard // change directory to card
   c. stargate2:cfcard# ls // this shows the files on card
4) Store file wipwr to Stargate Gateway 2 (SG2).
   a. Wipwr takes value 0 to 255 and sets driver transmit power
   b. Important to keep rss signals between 50 and 80.
5) Store files make_beam_pattern.m in project folder on PC
6) Place HDPA board securely on rotating tripod and center it.
7) Start at angle 0
8) Run the script control_hex or control_quad
   d. ./control_hex `<num_packets> <angle_reading>`
   e. An example is: `./control_hex 1000 255`
      i. for 1000 packets at angle 255°
9) Turn the tripod 1 degree (or whatever degree increment desired)
10) Run control script again
    f. An example is: `./control_hex 1000 256`

g. `Notice the first number is the number of packets (1000) and last number is the angle (350).`

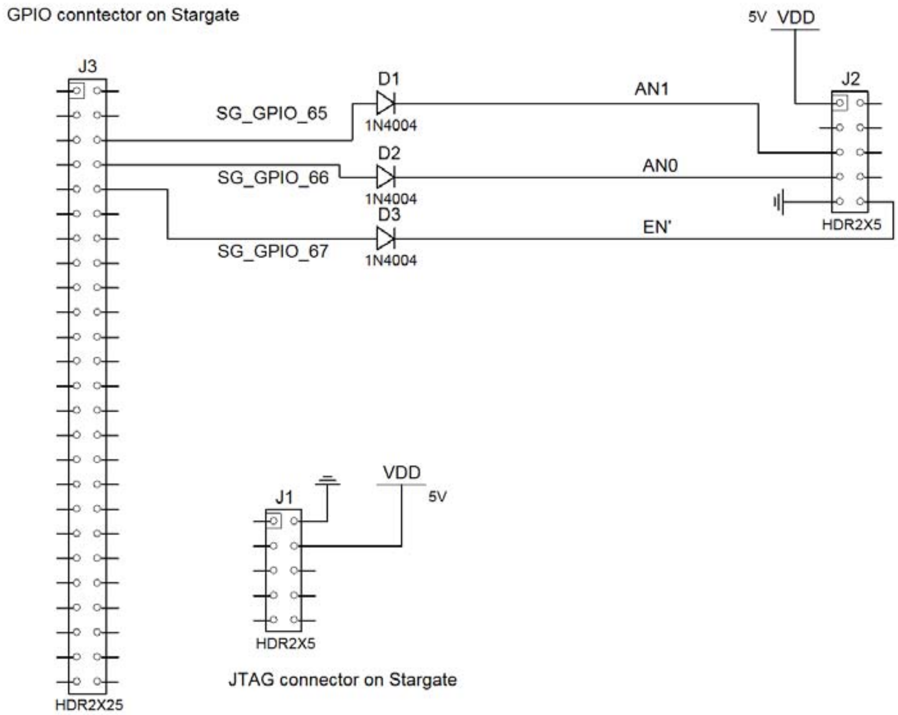
11) Continue with steps 7 and 8 until all 360 degrees are obtained.
12) Enter command in Cygwin to transfer all .text files to current directory (Project Folder)
    h. scp root@10.0.0.21:/mnt/cfcard/*.text .
    i. if the files are not stored on the card use appropriate location
        i. scp root@10.0.0.21:/root/example/*.text .
13) Open Matlab and run make_beam_patterns.m
    a. Ensure that you file names are consistent with ‘filenm’
14) Beam Patterns for each antenna pattern will be created
15) Store as many data sets as desired then run the AOA_main code
    a. Ensure that the folder calls throughout AOA estimation code are correct to you data set folders

8.6 Schematic

Hex Dipole to Stargate connector:



Quad-F to Stargate connector



8.7 ESP Workbench Simulation File

```
REM: ESPAR, Hex parasitic array
REM: version v1
REM: scale = all dimensions in lambda
REM: ... so, enter frequency 300MHz
REM:------------------------------------

RAD: Azimuth scan; fix theta and vary phi
300.000, 300.000, 1
90.0000, 90.0000, 0.000000
-180.000, 180.000, 1.00000

PLC: ground plane=finite disk PEC
20
0.583300, 0.000000, 0.000000
0.554751, 0.180250, 0.000000
0.471900, 0.342855, 0.000000
0.342855, 0.471900, 0.000000
0.180250, 0.554751, 0.000000
0.000000, 0.583300, 0.000000
-0.180250, 0.554751, 0.000000
-0.342855, 0.471900, 0.000000
-0.471900, 0.342855, 0.000000
-0.554751, 0.180250, 0.000000
-0.583300, 0.000000, 0.000000
-0.554751, -0.180250, 0.000000
-0.471900, -0.342855, 0.000000
-0.342855, -0.471900, 0.000000
-0.180250, -0.554751, 0.000000
0.000000, -0.583300, 0.000000
0.180250, -0.554751, 0.000000
0.342855, -0.471900, 0.000000
0.471900, -0.342855, 0.000000
0.554751, -0.180250, 0.000000
PLZ:
0, 0
WRR:
0.00400000, 38.0000
WRC: Driven element at origin
0.000000, 0.000000, 0.000000
0.000000, 0.000000, 0.250000
WRG:
1, 1.00000, 0.000000
WRA:
1
```

```
MUT: feed line & S matrix
50.0000, 0.000000
0, 3, 0

WRC: 1
0.333300, 0.000000, 0.000000
0.333300, 0.000000, 0.250000

WRC: 2
0.166700, 0.288600, 0.000000
0.166700, 0.288600, 0.250000

WRC: 3
-0.166700, 0.288600, 0.000000
-0.166700, 0.288600, 0.250000
WRA:
1

WRC: 4
-0.333300, 0.000000, 0.000000
-0.333300, 0.000000, 0.250000

WRC: 5
-0.166700, -0.288600, 0.000000
-0.166700, -0.288600, 0.250000
WRA:
1

WRC: 6
0.166700, -0.288600, 0.000000
0.166700, -0.288600, 0.250000
WRA:
1
RUN:
CLS:
END:
```