Spring 2011

Multiple-input multiple-output system simulation for spinning vehicles

Samuel James Petersen

MULTIPLE-INPUT MULTIPLE-OUTPUT SYSTEM SIMULATION FOR SPINNING VEHICLES

by

SAMUEL J PETERSEN

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

2011

Approved by

Dr. Kurt Kosbar, Advisor
Dr. Randy Moss
Dr. Steven Grant

ABSTRACT

This paper investigates the performance of a multiple-input multiple-output (MIMO) wireless communication system, when the transmitter is located on a spinning vehicle. In particular, a 2x2 MIMO system is used, with Alamouti coding at the transmitter. Both Rayleigh and Rician statistical models are considered in this paper and are combined with a deterministic spinning model to simulate the channel. The spinning of the transmitting vehicle, relative to the stationary receive antennas causes additional signal fading, and complicates the decoding and channel estimation. The simulated bit error rate is the primary performance metric used. A 4x2 asymmetric Alamouti coding scheme is proposed to increase the performance of the space-time code in this configuration. The 2x2 Alamouti channel code is shown to perform better than the Maximal Ratio Receiver Combining (MRRC) and single receiver (2x1) system in some circumstances and performs similarly to the MRRC in the broadside case. The unsymmetric 4x2 Alamouti code is shown to improve the performance of the 2x2Alamouti code in the broadside case.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the support and guidance of my advisor Dr. Kosbar. I would also like to thank the International Foundation for Telemetering and the International Telemetry Conference for their financial support and providing the impetus for the work. My thanks also go to the Department of Electrical and Computer Engineering, my Advising Committee, the Office of Graduate Studies and all those involved with the Chancellor’s Fellowship. Last but not least, I would also like to thank my parents John and Emily Petersen for their support and encouragement throughout my undergraduate and graduate studies.

TABLE OF CONTENTS

Page

SECTION

APPENDIX

LIST OF ILLUSTRATIONS

1. INTRODUCTION

The problem considered in this paper is the performance of a 2x2 Alamouti space-time channel encoding scheme when the transmitter or receiver antennas are located on a spinning vehicle. The geometry of the proposed problem is illustrated in Figure 1.1. The system consists of two transmit antennas mounted on opposite sides of a spinning, cylindrical vehicle and two fixed receive antennas near the ground.

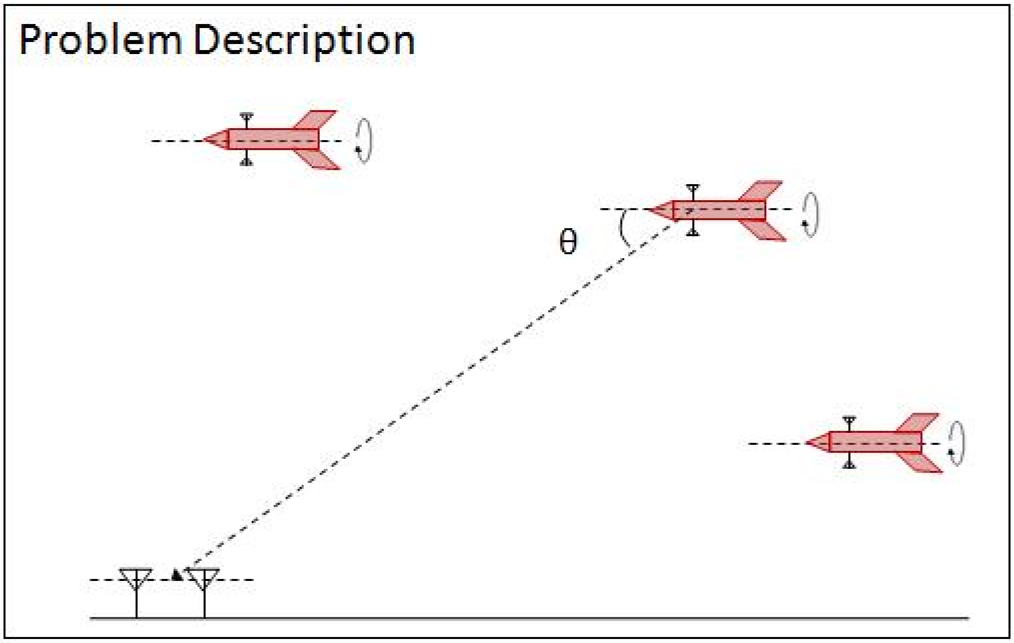

Figure 1.1. Problem Geometry

To maintain channel independence the transmit antennas are assumed to be separated by several carrier wavelengths. The same is also assumed for the receiver antennas. For this experiment the antenna separation at both the transmitter and receiver is fixed at $2\lambda_c$, where λ_c is the wavelength of the carrier. An integer number of wavelengths were chosen for convenience and small variations in separation do not have an appreciable effect on the results. The spinning frequency of the vehicle, relative to the spin axis, is held constant. The line-of-sight (LOS) angle is measured from the vehicle's spin axis, which for the purposes of this paper, is fixed in the horizontal orientation. We consider several cases where the line-of-sight angle is varied along an arc of constant range at 0°, 45° and 90°.

The primary channel considered for this experiment is a Rayleigh fading model that simulates the multi-path propagation of a wireless channel. The Rayleigh channel model assumes a rich scattering environment, where no single path dominates the rest. A Ricean channel model is also considered which adds a dominant line-of-sight component to the Rayleigh model. The expected effects of the vehicle rotation on the time-varying channel were incorporated into the statistical Rayleigh and Ricean models. The bit-error rate is the main metric of system performance and is calculated through computer simulation for various signal-to-noise ratios (SNR).

The results of the simulation show that the bit-error performance of all considered scheme changes as a function of the LOS angle. As expected the overall performance of all systems decreases as the LOS angle increases from 0° to 90°. For all LOS angles the Alamouti code is shown to outperform or match the Maximal Ratio Receive Combining (MRRC) and 2x1 schemes. As the LOS angle approaches the broadside case the BER of the Alamouti code converges to the MRRC. An asymmetric 4x2 Alamouti scheme is proposed and shown to improve the performance of the Alamouti code in the broadside case.

2. BACKGROUND INFORMATION

2.1. DIGITAL WIRELESS COMMUNICATIONS

Digital wireless communications is a broad field that is constantly evolving and covers a wide variety of topics. This section is devoted to providing a brief overview of the major topics in wireless communications that are referenced in the simulation description later in the paper.

2.2. BINARY PHASE SHIFT KEYING

Binary Phase Shift Keying (BPSK) is a digital modulation technique that is commonly used in wireless communications. In BPSK the binary bits are mapped to complex symbols with equal magnitude and a phase difference of π. In the baseband that means that the binary 0's and 1's bits are mapped to $-\sqrt{E_b}$ and $\sqrt{E_b}$, where E_b is the average energy per bit. In the pass-band the equation for BPSK can be seen in Equation 1 and in the base-band Equation 2. [Ziemer]

$$s_{pb}(t) = \sqrt{E_b}\cos\left(2\pi f_c t + (x-1)\pi\right) \tag{1}$$

$$s_{bb}(t) = \sqrt{E_b}(2x-1) \tag{2}$$

The signal-space constellation for BPSK is given in Figure 2.1 with basis functions defined as $\Phi_1(\mathrm{t}) = \cos(2\pi f_c t)$ and $\Phi_2(\mathrm{t}) = \sin(2\pi f_c t)$.

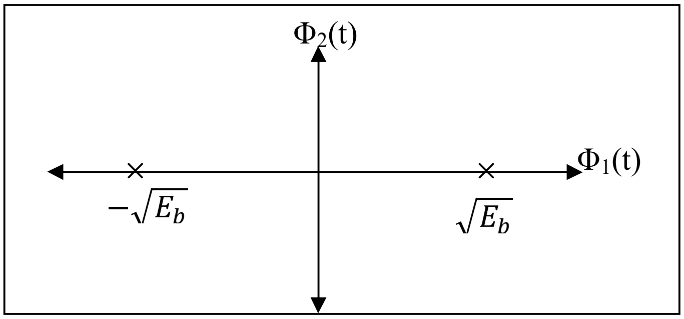

Figure 2.1. BPSK Signal Constellation

2.3. MULTIPATH

A common problem in wireless communications is the existence of a multipath propagation. Multipath propagation results from the transmitted signal reaching the receiver by two or more paths by either reflection or refraction in the environment. The multipath signals at the receiver can arrive with differing phases and amplitudes caused by the different path lengths. A graphical representation of a multipath environment can be seen in Figure 2.2.

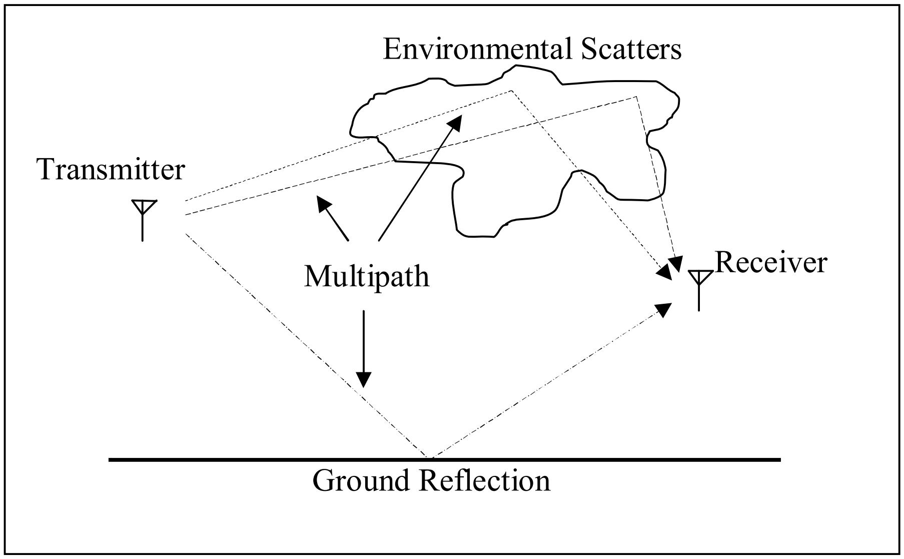

Figure 2.2. Multipath Propagation

The superposition of these signals at the receive antenna can lead to constructive or destructive interference which causes either an increase or decrease in the signal power at the receiver. This fluctuation in the strength of the received signal referred to as small-scale fading since it is often caused by reflections off the minutia of the environment. The superposition of many reflections at the receiver is often modeled as a stochastic process. This seemingly random fluctuation of the received signal strength on a relatively fast time scale can cause serious disruptions in a wireless communications system. [Rappaport]

2.3.1. Rayleigh Scattering Model. In wireless communications the Rayleigh Fading model is a commonly used multi-path model. The Rayleigh Fading model assumes uniformly distributed scattering objects around the receiver. The reflections of these scatterer objects are assumed to have equal magnitude and uniformly distributed relative phase. When the number of scattering objects becomes large enough the summation of N uniformly distributed complex random variables creates a Gaussian Complex random variable by the Central Limit Theorem which has a complex envelope, or magnitude, with a pdf that has the familiar Rayleigh curve given by the equation in Eq. (1). [Hogg]

$$F(x) = \frac{x^2}{\sigma^2} e^{-x^2 / 2\sigma^2} \tag{3}$$

A common and effective way of producing channel coefficients with the desired statistics of a Rayleigh distribution is the Sum of Sinusoids method (SoS). One example of which is the Jake's Model which uses a sum of sinusoids with uniformly distributed phases (-π,π]. An implementation of Jakes' Model can be seen in Equations (4-7). [Xiao]

$$X(t) = X_c(t) + jX_s(t) \tag{4}$$

$$X_c(t) = \sqrt{\frac{2}{M}} \sum_{n=1}^{M} \cos(\varphi) \cos(\omega_d t \cos(\alpha_n) + \gamma) \tag{5}$$

$$X_s(t) = \sqrt{\frac{2}{M}} \sum_{n=1}^{M} \sin(\varphi) \cos(\omega_d t \sin(\alpha_n) + \gamma) \tag{6}$$

$$\alpha_n = \frac{2\pi n - \pi + \theta}{4M}, \; n = 1,2, \dots, M \tag{7}$$

Where θ, γ, φ are independent random variables with uniform distribution from [-π π) for all values of n. A graphical representation of the Jakes' model can be seen in Figure 2.3.

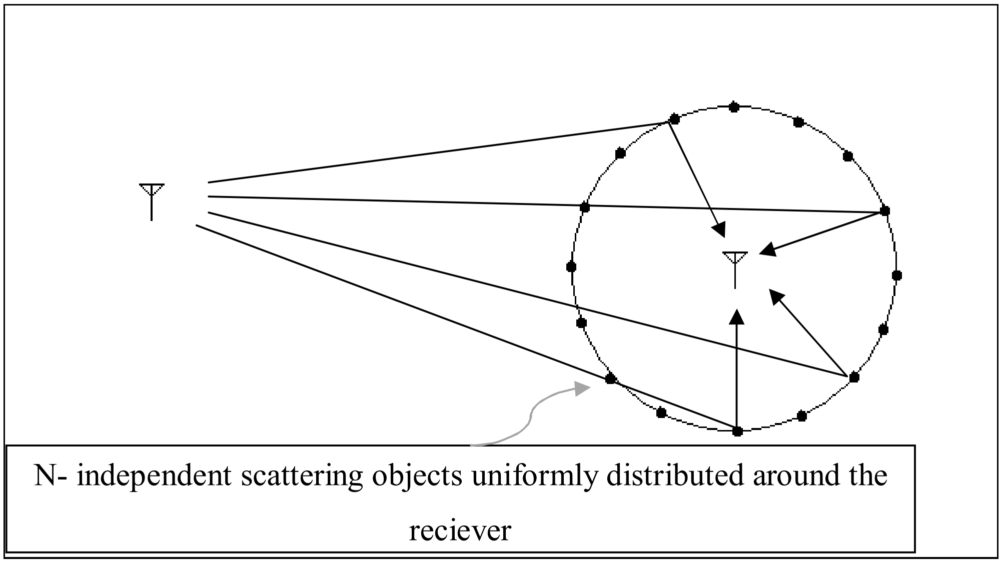
N- independent scattering objects uniformly distributed around the reciever

Figure 2.3. Jakes' Rayleigh Scattering Model

The results of a Jakes' model simulation of a Rayleigh distribution can be seen in Figure 2.4.

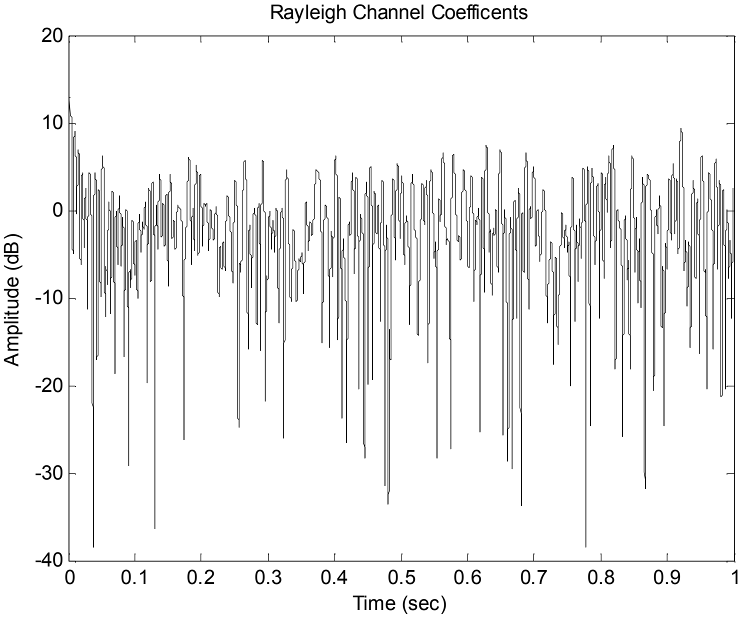

Figure 2.4. Rayleigh Channel Coefficients (Jakes' Model)

The amplitude of the channel coefficients shown in Figure 2.4 clearly illustrates the time-varying nature of the channel. At some times the multipath signals add constructively and the received power increases. At other times the multipath signals add destructively and the received power is diminished even approaching zero. When the received power is reduced significantly the receiver is said to be in a deep fade, where the communication link is greatly diminished.

In Figure 2.5 the PDF of the Jakes' Model channel coefficients can be seen compared to the theoretical Rayleigh distribution given by Equation (3).

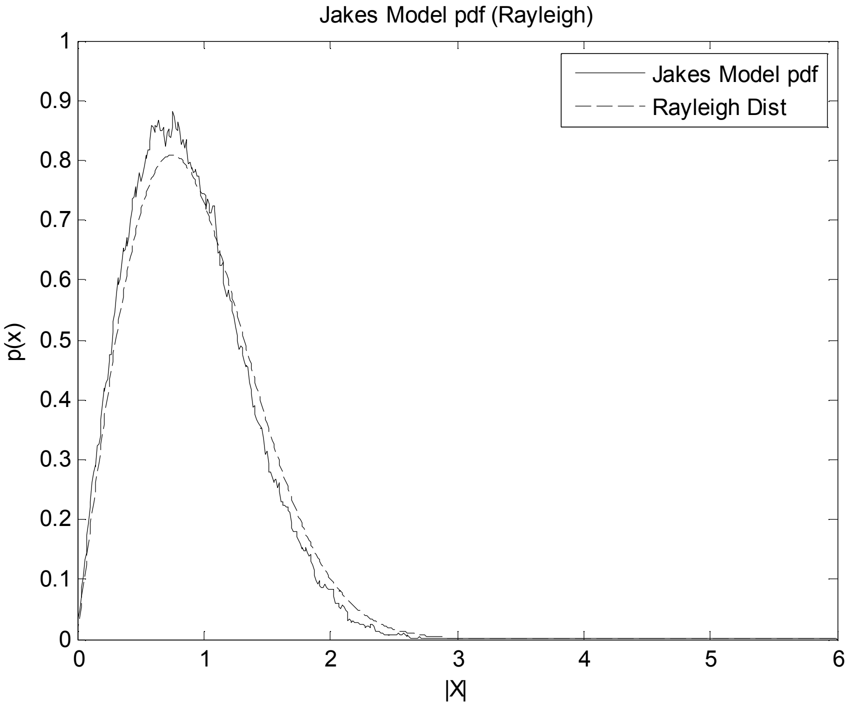

Figure 2.5. Jakes' Model PDF

2.3.2. Ricean Channel Model. Another commonly used description of a frequency flat fading channel is the Ricean distribution. The Ricean distribution includes a stationary, non-varying, line-of-sight component to the Rayleigh distribution. The coefficients of a Ricean distributed channel have a similar time-varying nature to their Rayleigh counterparts. Where the differences between the Rayleigh and Ricean distributions become apparent are in the shape of their PDF's. A comparison of a typical Ricean distribution to a Rayleigh distribution can be seen in Figure 2.6. [Rappaport]

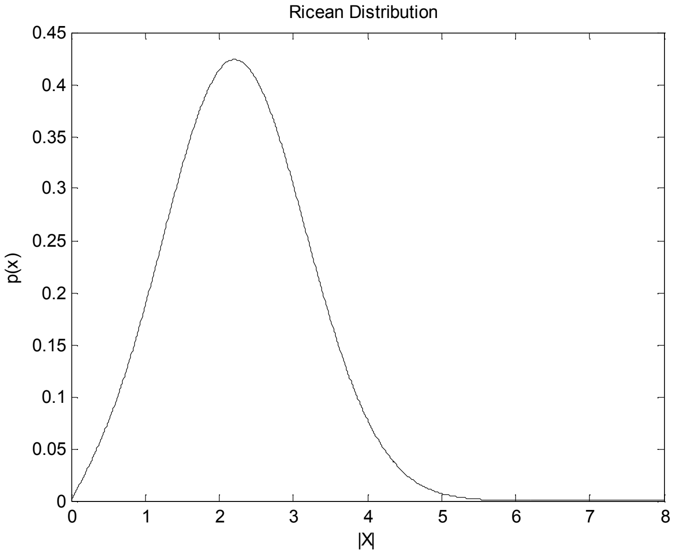

Figure 2.6. Ricean Channel PDF

2.4. 2X2 CHANNEL

One method of mitigating the effects of a fast fading channel is to take advantage of spatial, frequency and coding diversity. Spatial diversity is created by using several uncorrelated channels either by adding multiple transmitters (MISO), adding multiple receivers (SIMO) or adding both multiple transmitters and receivers (MIMO). The simplest MIMO case is a 2x2 system with two transmit antennas and two receive antennas spaced sufficiently apart to create 4 uncorrelated fading channels. Since the channels are uncorrelated it is unlikely that all four are in a deep-fade at the same time. A graphical representation of a 2x2 MIMO system can be seen in Figure 2.7. [Jankiraman]

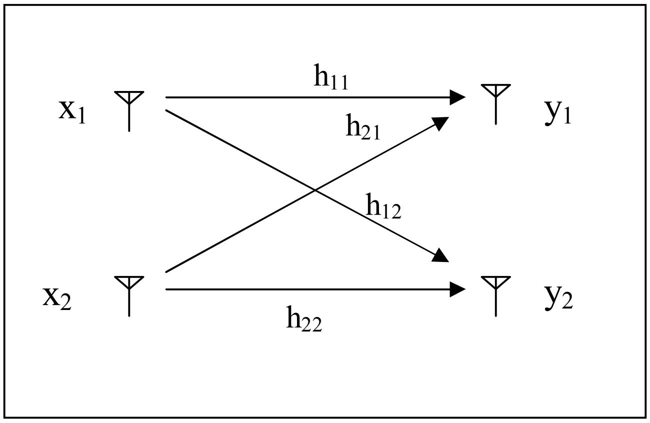

Figure 2.7. MIMO Communication System

A 2x2 MIMO system can also be written in matrix form as seen in Equation (8).

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad (8)$$

2.5. MAXIMAL RATIO RECEIVER COMBINING

There are several methods of combining the received signals from two or more receive antennas. The method that gives the best statistical reduction of fading through combining the multiple received signals is known as the Maximal Ratio Receiver Combining (MRRC) method. In this scheme the signals from all receive branches are first weighted by their signal voltage to noise ratio and then summed. In this way the maximal signal power is preserved without increasing the noise level through the summation. In this method each receiver branch needs to be co-phased before being summed in contrast to the selection method which only needs one receiver branch. This adds complexity to the receiver implementation. In cases where variable weighting coefficients may not be convenient, the receiver branches may be combined with unity weights. This however is not optimal and inferior to the MRRC but still better then the selection method [Rappaport]. A block diagram representation of the Maximal Ratio Combining scheme can be seen in Figure 2.8.

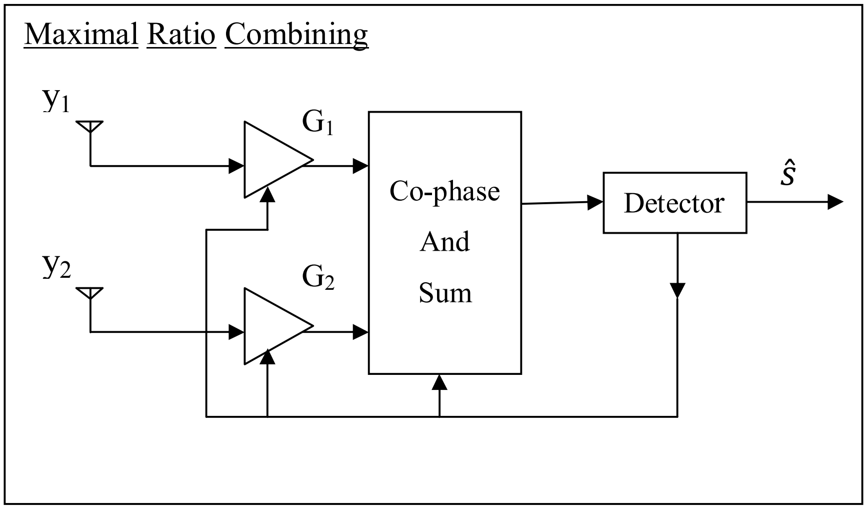

Figure 2.8. Maximal Ratio Combining Block Diagram

2.6. ALAMOUTI SPACE-TIME BLOCK CODE

In Siavash Alamouti's landmark 1998 paper [Alamouti] a novel method was presented that can create a two branch diversity scheme with two transmitters and a single receive antenna. By transmitting two orthogonally encoded symbols from the spatially separated transmit antennas, a single receiver can separate and decode the received signal and achieve the same diversity order as a 2x2 MRC scheme. Alamouti also generalizes the procedure for two transmit antennas and M receive antennas. Using this generalized version a 2x2 Alamouti space-time code scheme can achieve the same diversity order as a 2x4 MRC system. In Figure 2.9. a block diagram representation of a 2x2 Alamouti space-time code scheme can be seen.

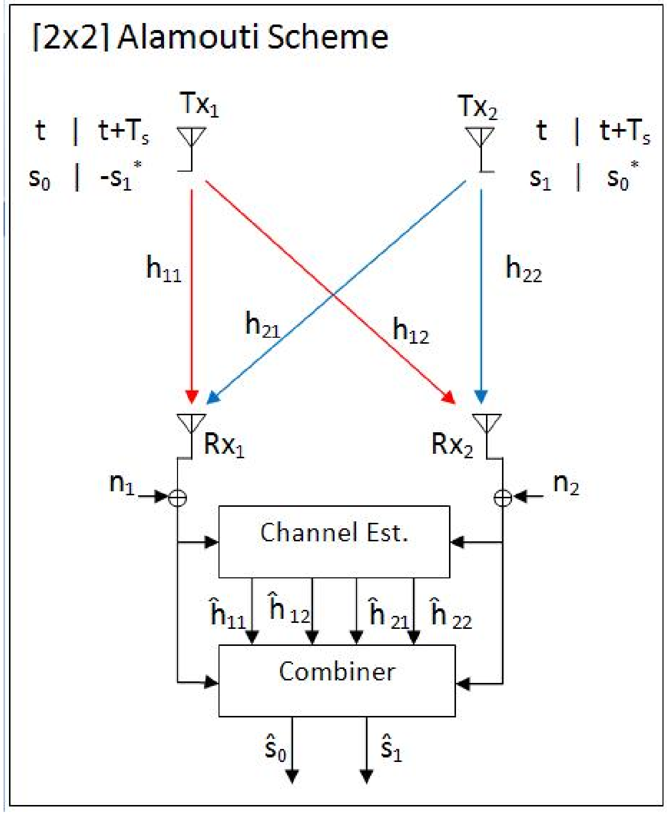

Figure 2.9. Alamouti Space-Time Coding Scheme

The Alamouti space-time code works by simultaneously transmitting the first symbol (s_o) from the first transmitter (Tx_1) and the second symbol (s_1) from the second transmitter (Tx_2). Then, at $t = t_0 + T_s$, Tx_1 transmits the negative conjugate of the second symbol ($-s_1*$) and Tx_2 transmits the conjugate of the first symbol (s_0*). The scheme then follows the same pattern for all subsequent symbols, so that received signal can be represented by the equations in Equations (9-12).

$$r_1(t) = h_{11}s_0 + h_{21}s_1 + n_1(\mathrm{t}) \tag{9}$$

$$r_1(t+T_s) = -h_{11}{s_1}^* + h_{21}{s_0}^* + n_1(t+T_s) \tag{10}$$

$$r_2(t) = h_{12}s_0 + h_{22}s_1 + n_2(t) \tag{11}$$

$$r_2(t+T_s) = -h_{12}{s_1}^* + h_{22}{s_0}^* + n_2(t+T_s) \tag{12}$$

Then at the receiver the Alamouti decoder recovers the transmitted data, by estimating the channel coefficients and combining received signals as shown in Equations (13,14).

$$\hat{s}_0 = \hat{h}^*{}_{11} r_1(t) + \hat{h}_{21} r_1{}^*(t+T_s) + \hat{h}^*{}_{12} r_2(t) + \hat{h}_{22} r_2{}^*(t+T_s) \qquad (13)$$

$$\hat{s}_1 = \hat{h}^*{}_{11} r_1(t) - \hat{h}_{21} r_1{}^*(t+T_s) + \hat{h}^*{}_{12} r_2(t) - \hat{h}_{22} r_2{}^*(t+T_s) \qquad (14)$$

The Alamouti space-time code significantly outperforms the bit-error rate (BER) of a MRRC receiver with the same number of receive antennas. Both the Alamouti code and a MRRC receiver significantly outperform a SISO wireless link in a fading environment. Verification of this through simulation in a Rayleigh fading channel can be seen in Figure 2.10.

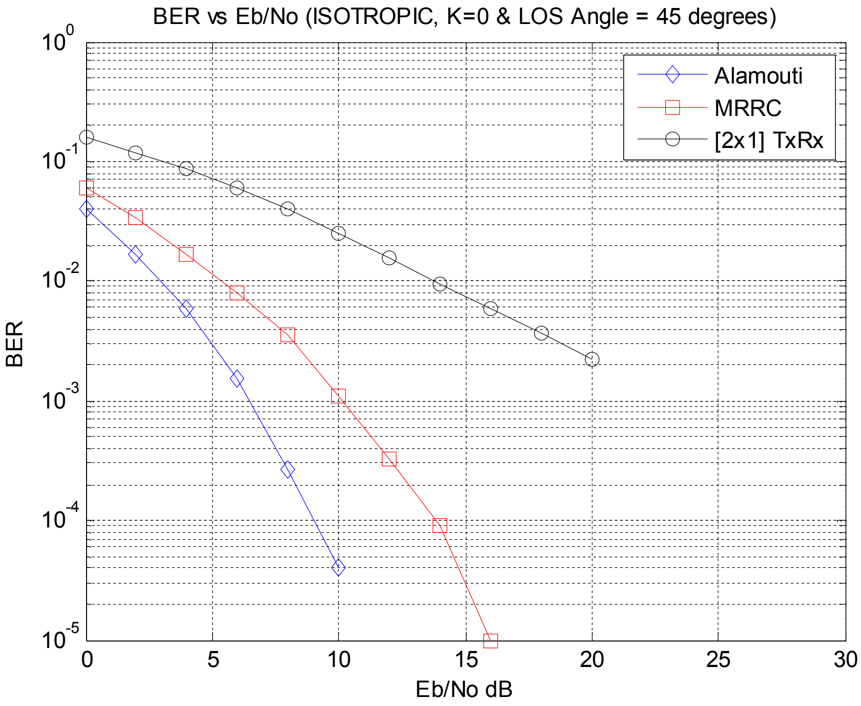

Figure 2.10. Alamouti BER Performance Verification

2.7. ANTENNA PATTERNS

A common graphical representation of the spatial properties of an antenna is its radiation pattern, or antenna pattern. An antenna pattern is a mathematical or graphical representation of the radiation properties of an antenna as a function of space. In most cases antenna patterns are based on measurements made in the far-field. Although antennas exist in three dimensions often antenna patterns are created as slices along orthogonal plains in two dimensions. This is sufficient to describe the three dimensional propagation of an antenna. Also antenna patterns are usually plotted in polar coordinates on a logarithmic scale, normalized with peak amplitude and scaled in decibels (dB). Sometimes the pattern is scaled based on an ideal omni-directional antenna, or isotropic antenna that radiates equally in all directions, these patterns are given in dBi. In both cases the antenna pattern is independent of radial distance [Balanis]. An example of an Antenna Radiation Pattern for a dipole antenna can be seen in Figure 2.11.

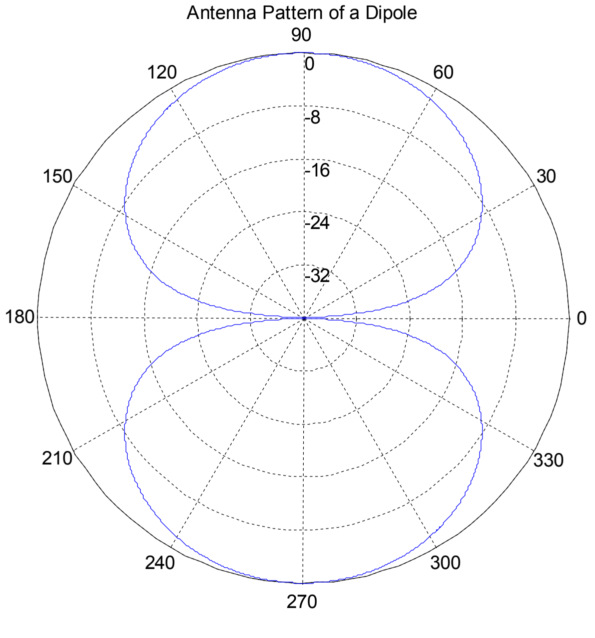

Figure 2.11. Antenna Pattern of a Dipole Antenna

2.7.1. Patch-Type Antenna Pattern. The antenna type considered in this experiment is a micro-strip or patch antenna. These antennas are commonly used when the physical profile of the antenna is of concern. They can be made very small and light and can be formed on curved surfaces. Patch antennas are constructed similarly to a printed circuit board with a thin copper sheet on a dielectric substrate and a large ground plane. The shape of the patch antenna can vary greatly depending on the application but a common configuration is a rectangular patch antenna. The rectangular patch antenna is a directional antenna that radiates maximally in a direction normal to the patch antenna. Ideally a patch antenna like this has no radiation in the backward direction, but this model assumes an infinite ground plane. In a practical antenna there is some backward propagation but is significantly reduced. An example of an antenna pattern for a rectangular patch antenna is shown in Figure 2.12. For a square antenna patch antenna the radiation is assumed to be symmetric about an axis perpendicular to the plane of the patch [Balanis].

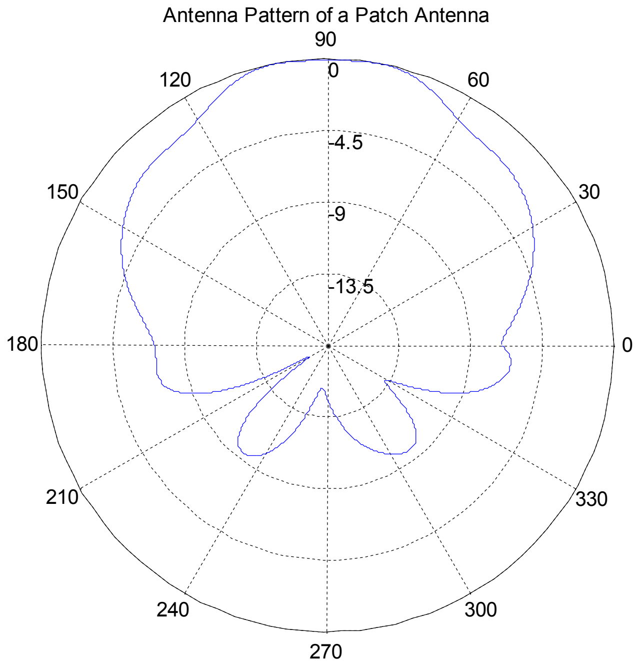

Figure 2.12. Patch-Type Antenna Pattern

2.7.2. Array Factor. To maintain spatial diversity between the receivers and transmit antennas they need to be separated by several carrier wavelengths. If the multiple transmit antennas in a MIMO system are considered like an array antenna this can cause a few problems. Because of the separation between the antennas can be in excess of λ_c, grating lobes will appear in the array factor of the array. This means that the array pattern will have side-lobes with significant magnitude and unintended nulls. To model this interaction between antennas in this array configuration the array factor is calculated in the simulation and combined with the individual antenna patterns, through multiplication to find the overall antenna pattern [Balanis].

The antenna Array Factor (A_F) was calculated using the far-field approximation as described in Figure 2.13. The far-field approximation states that the antennas are far enough away from the receiver, or observer so as to appear that the incident wave-front is flat. This means that the distance travelled in terms of magnitude from each transmitter and the angle are the same as shown in Figure 2.13. The angle γ in three dimensions can be written as

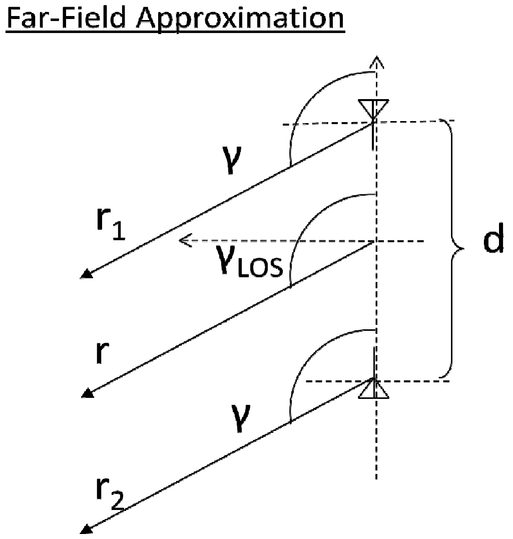

Figure 2.13 Antenna Far-Field Approximation

Using this far-field approximation the general equation for the antenna array pattern can be written like in equation (15a). Then the derivation of a convenient form of the antenna pattern is shown resulting in the final antenna A_F given in equation (15e).

$$A_F = e^{-jkr_1} + e^{-jkr_2} \tag{15a}$$

$$= e^{-jk\left(r-\frac{d(t)}{2}\cos(\gamma)\right)} + e^{-jk\left(r+\frac{d(t)}{2}\cos(\gamma)\right)} \tag{15b}$$

$$= e^{-jkr}\left(e^{jk\frac{d(t)}{2}\cos(\gamma)} + e^{-jk\frac{d(t)}{2}\cos(\gamma)}\right) \tag{15c}$$

$$= 2e^{-jkr}\left(\cos\left(\mathrm{k}\frac{d(t)}{2}\cos(\gamma)\right)\right), \tag{15d}$$

$$where\ \gamma = \cos^{-1}(\cos\emptyset(\mathrm{t})\sin\theta)$$

$$A_F = \cos\left(\frac{1}{2}kd(t)\cos\emptyset(\mathrm{t})\sin\theta\right) \tag{15e}$$

The antenna Array Factor can be plotted similarly to an antenna pattern of its elements. The Array Factor calculated in equation (13e) can be seen graphically in Figure 2.14.

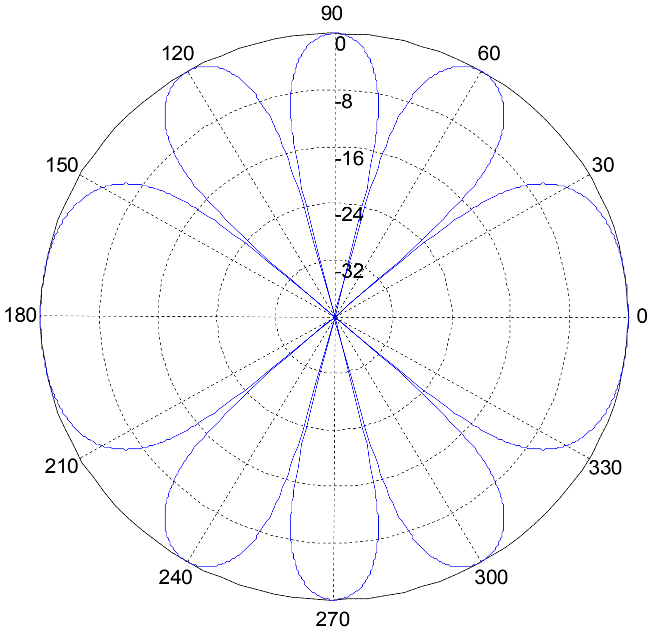

Figure 2.14 Two Element Antenna Array Factor d = 2λ_c

The antenna Array Factor and element antenna patterns are considered independent. The antenna pattern solely derived from the individual elements that construct the array, and the A_F derived solely from the geometry of the array. So the total antenna pattern of an array is the product of the element factor and array factor.

3. SIMULATION DESCRIPTION

3.1. MATLAB SIMULATION

Computer simulation is a commonly used tool to evaluate the anticipated performance of a wireless communications system in a wide variety of scenarios. In this way algorithms and configurations can be tested and compared without necessarily implementing them in hardware or risking expensive prototypes in every experiment. Computer simulations can be made to have varying degrees of fidelity and can be invaluable at all stages of development. Lower fidelity simulations may be useful in the early development stages to test feasibility. As the system designs mature, assumptions can be replaced with actual data and model parameters can be adjusted to match measured results, thus increasing the fidelity of the simulation. The simulation discussed in this paper is a preliminary feasibility study in the early developmental stages of a low-cost, experimental MIMO receiver and transmitter for the telemetry community. As such it was necessary to model the transmitter, receiver and channel with broad strokes, which will hopefully be updated as the system design progresses and experimental data is recorded. The block diagram in Figure 3.1 shows the major components of the simulation that will be discussed in greater detail later in this section.

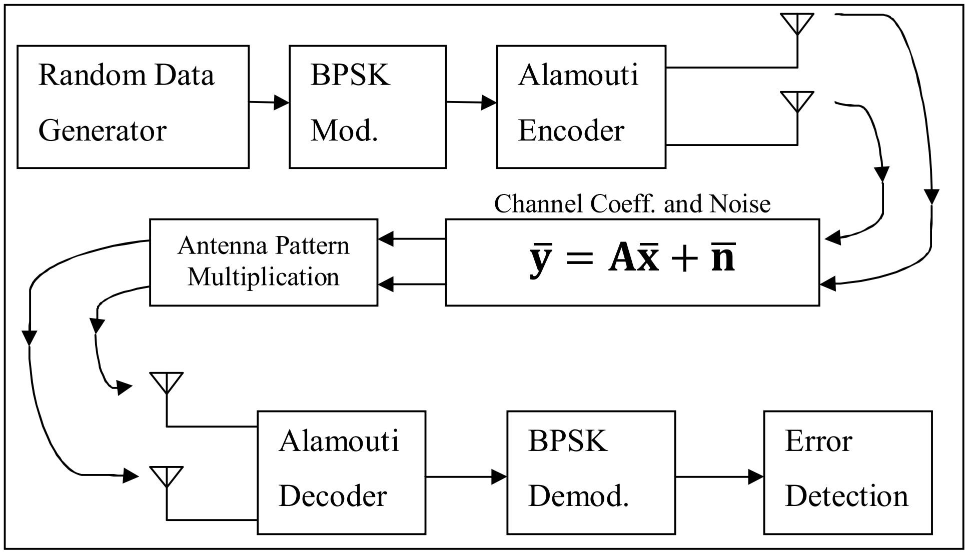

Figure 3.1. Simulation Block Diagram

3.2. RANDOM NUMBER GENERATOR

The source data for this simulation is assumed uniform and a pseudo-random number generator was used to create a string of uniformly distributed binary bits. The x=rand(1,N) function native in MATLAB^{TM} was used to produce a 1xN vector of uniformly distributed bits from 0 to 1. Then the vector is logically compared to 0.5 to produce equally likely binary bits [0,1] as seen in equation (16).

```
x = rand(1,N)  > 0.5;
```

(16)

3.3. BPSK MODULATION

The type of digital modulation used in this experiment is Binary Phase Shift Keying or BPSK. Higher order phase shift keying schemes were considered like Quadrature Phase Shift Keying (QPSK) and 8-ary PSK, but BPSK was selected because of its simplicity. The robustness of BPSK and relative simplicity of decoding was important because the purpose of this experiment is to evaluate the performance of the Alamouti Space-time code not the modulation scheme. Implementation of BPSK modulation is straight forward. The random binary data [0,1] must be mapped to [-1,1]. This can be accomplished with the equation (17).

$$s = 2x - 1; \qquad \text{where } x=[0,1] \tag{17}$$

3.4. AWGN CHANNEL

The implementation of the AWGN channel was done without using the awgn() function built into MATLAB. There are two main reasons for doing this. The first is to increase the transparency of the process. The second is to ensure that the AWGN channel has the intended signal-to-noise ratio (SNR). The energy per bit (E_b) is first set nominally to 1, but could practically be any reasonable value. Then using the defined SNR which is one of the simulation parameters the noise power (N_o) is then calculated. The AWGN channel is then calculated by creating a vector of complex random variables with a Gaussian distribution. The AWGN channel is then scaled to the appropriate power level for the given SNR. This process can be seen in the code segment in equation (18).

```
AWGN = sqrt(No)*[randn(2,Ns) + 1i*randn(2,Ns)];          (18)
```

3.5. RAYLEIGH CHANNEL SIMULATION

The Rayleigh channel model in this simulation is a modified Jakes' model. The Rayleigh channel coefficients, like the AWGN channel, are a vector of complex random variables. The easiest way of generating a Rayleigh distributed random vector would be to use the MATLAB function randn(1,N) to create a vector of complex Gaussian distributed random variables. This method lacks some of the physical relations that are given by the Jakes' model.

The Jakes' model is implemented as a function with parameters, **t** the simulation time vector, **Ts** the simulation sample period, **fd** the maximum Doppler frequency shift, and **M** the number of reflectors. Then three random vectors are generated phi, theta, and psi that are uniformly distributed (0,2π]. The reflector parameter alpha is calculated using theta with the equation alpha= (2*pi*n – pi + theta) / (4*M). Then the coefficients for the M reflectors are calculated and summed in a FOR loop shown below.

```
for n=1:M
    Xc_tn=cos(wd*t*cos(alpha(n))+phi(n));
    Xs_tn=sin(wd*t*sin(alpha(n))+psi(n));

    Xc_t=Xc_t+Xc_tn;
    Xs_t=Xs_t+Xs_tn;
end
```

The final coefficient vector is created by combining the real and imaginary parts and scaling the magnitude by X_t=sqrt(2/M)[Xc_t + jXs_t]. Four independent coefficient vectors are calculated for the 2x2 MIMO case. For the Alamouti Block Encoding a block size of two bits is required and the channel is required to be constant for the block duration. Therefore for an N bit length simulation N/2 channel coefficients are calculated and each value repeated once to get the required length of N coefficients.

3.6. RICEAN CHANNEL SIMULATION

The Ricean Channel model uses a Rayleigh distribution as its base with an additional line of sight component. Typically this line of sight component is held constant or given some time varying characteristics. In this simulation the Ricean line of sight component has a time-varying nature that is calculated based on the path length difference between the two transmitting antennas to a given receive antenna. The path length difference is a function of the time-varying antenna separation at the transmitter, the line-of-sight angle and the relative location of the receive antenna. Using the equation for the Array Factor derived previously in equation(15) the LOS interference from the two transmit antennas at a given receiver antenna can be easily determined. The LOS angle for a given scenario is calculated for a given receive element and the calculation then repeated for the remaining receiver locations. To simulate the spinning of the array the time-varying angle is progressed from $(0,2\pi]$ for one period of vehicle rotation. A visual representation can be seen in Figure 3.2.

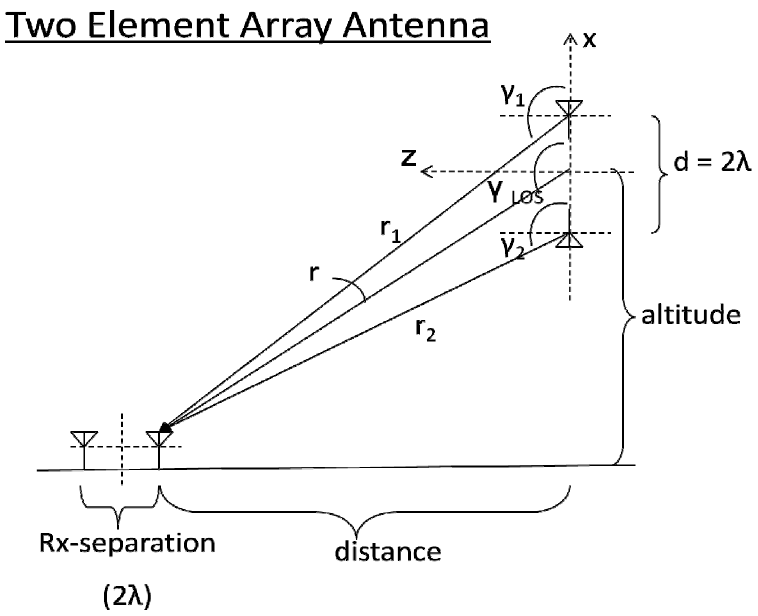

Figure 3.2 Time Varying Array Factor Calculation

The power ratio between the line of sight and the Rayleigh scatterers, K is a constant used to scale the power of the line of sight component vs. the overall power of the Rayleigh scatterers. The equations used to combine of the deterministic line of sight component with the Rayleigh scattering model are given in equations (22-24).

$$Zc = (Xc + sqrt(A_F*K))/sqrt(1+A_F*K); \tag{22}$$

$$Zs = (Xs + sqrt(A_F*K))/sqrt(1+A_F*K); \tag{23}$$

$$Z =sqrt(E_F)*(Zc + j*Zs); \tag{24}$$

The coefficients for the time varying line of sight component can be seen for line of sight angles 0°, 45°, 90° in Figures 3.3-3.5.

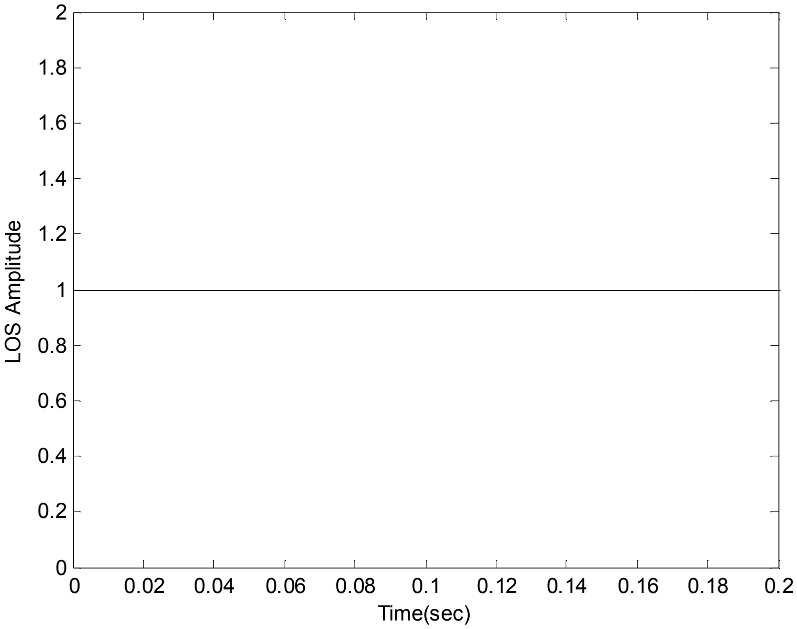

Figure 3.3. Ricean Channel Deterministic Line-of-Sight (0°)

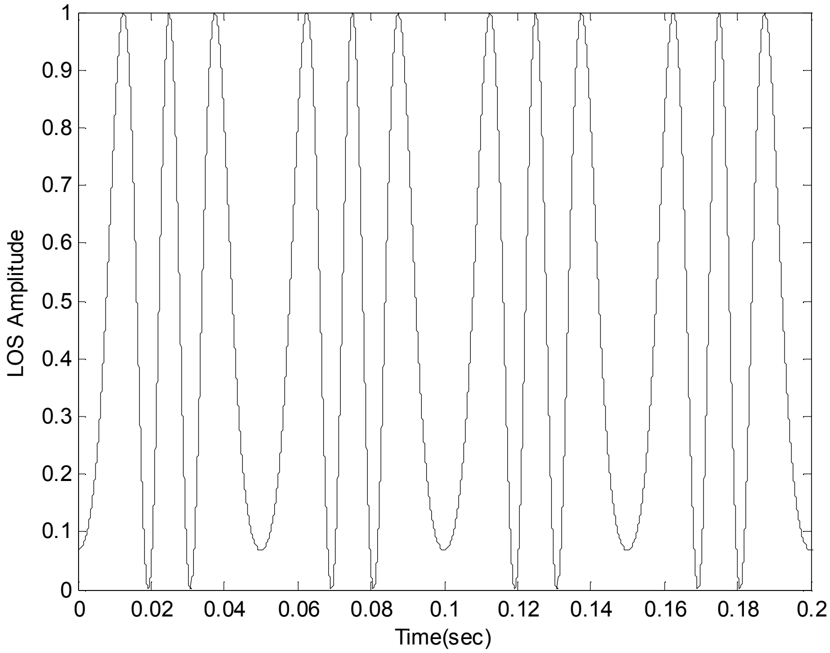

Figure 3.4. Ricean Channel Deterministic Line-of-Sight (45°)

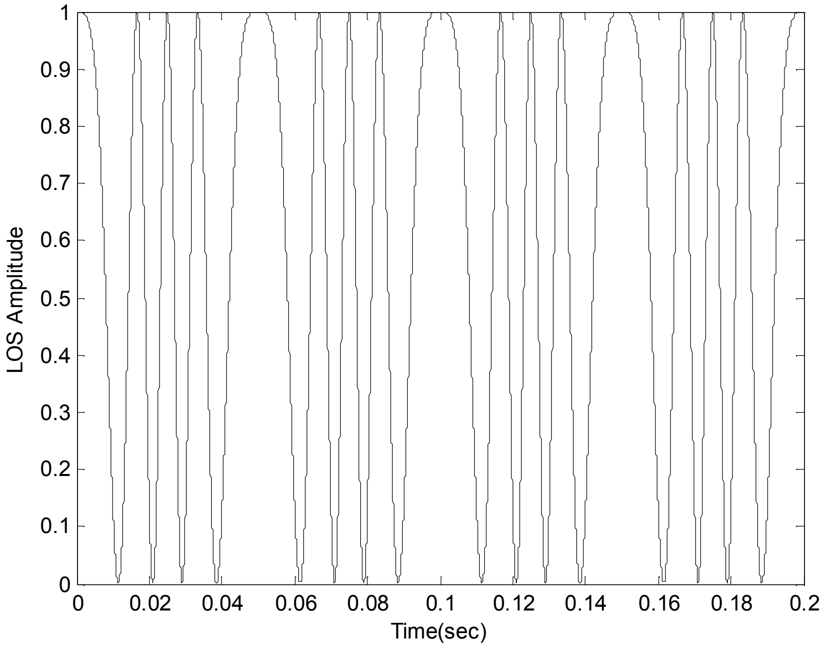

Figure 3.5. Ricean Channel Deterministic Line-of-Sight (90°)

3.7. ANTENNA PATTERN AND ROTATION

For this simulation a typical radiation pattern of a square patch antenna was chosen. A measured antenna pattern was chosen rather than the closed form solution of the theoretical pattern since the models used to calculate the theoretical solution assume an infinite ground plane which does not represent adequately of a practical antenna. To digitize the antenna pattern 25 equally spaced points were measured. This is a sufficient resolution to preserve the main features of the pattern. Then the using the interpl() function the points in between the measured values were interpolated to the desired resolution. Then the angle between the line-of-sight and the normal vector of the antenna is used to calculate the antenna radiation intensity in the direction of line-of-sight. This angle changes as a function of time as the vehicle rotates relative to the line-of-sight. In this way a vector of radiation intensity verse time is created as can be seen plotted in Figure 3.6.

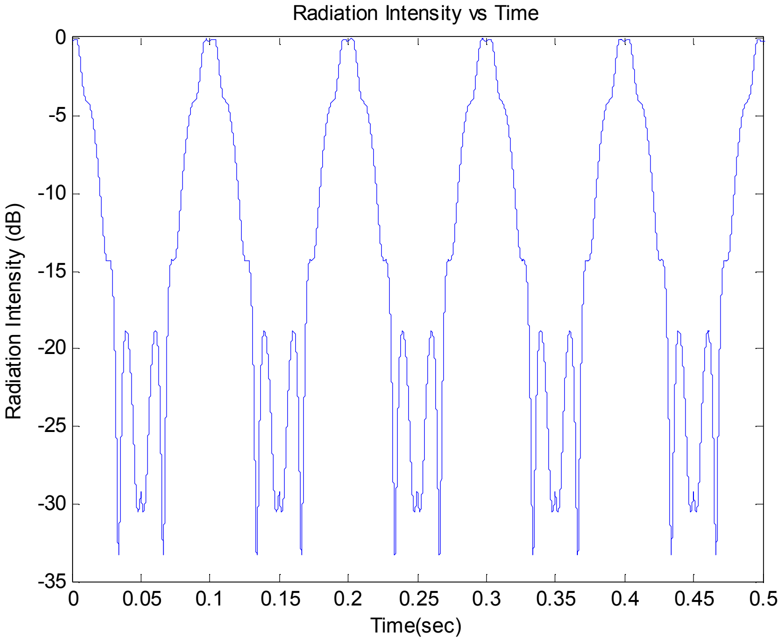

Figure 3.6. Antenna Radiation Intensity as a Function of Time

The radiation intensity is normalized relative to its maximum radiation intensity, and so ranges between [0,1]. This vector of normalized radiation intensities is then applied as channel coefficients.

3.8. CHANNEL MULTIPLICATION AND AWGN

The received signals are calculated separately at the two receivers. The radiation intensity **A** and Rayleigh/Ricean channel coefficients **h** multiply at the receiver and the white Gaussian noise adds. The received signals are calculated as shown in equations (25-28).

$$r_1(n) = h_{11}(n)s_1 + h_{21}(n)s_2 \ + wgn_1(n) \tag{25}$$

$$r_1(n+1) = (h_{21}(n)^*)s_1 + (-h_{11}(n)^*)s_2 \ + wgn_1(n+1) \tag{26}$$

$$r_2(n) = h_{12}(n)s_1 + h_{22}(n)s_2 \ + wgn_2(n) \tag{27}$$

$$r_2(n+1) = (h_{22}(n)^*)s_1 + (-h_{12}(n)^*)s_2 \ + wgn_2(n+1) \tag{28}$$

3.9. 2x2 ALAMOUTI ENCODING

The Alamouti Space-time block encoding scheme presented in Figure 2.9 was implemented in the MATLAB simulation with a few modifications. Instead of coding the transmitted symbols the channel coefficient was encoded which has the same overall effect. The Alamouti code was implemented in this way for convenience of simulation, specifically in the Alamouti decoding.

The first step in the Alamouti encoding is to reshape the data stream from a 1xN vector s = [s(n), s(n+1) …, s(N)], of BPSK modulated symbols to a 2xN matrix. This is accomplished using the reshape() function in MATLAB$^{\text{TM}}$ which moves the even indexed values of the 1xN vector to a new row creating a 1x$\left(\frac{N}{2}\right)$ matrix. Then the Kronecker tensor product function is used, kron(), to multiply the 1x$\left(\frac{N}{2}\right)$ vector by a 2x1 vector of ones which creates a copy of the values in each row creating a 2xN vector of the form in equation (29).

$$S = \begin{bmatrix} s_1 & s_1 & s_3 & s_3 & \cdots \\ s_2 & s_2 & s_4 & s_4 & \cdots \end{bmatrix} \tag{29}$$

The next part is the channel encoding, where the channel coefficients are modified for the Alamouti code. The channel coefficients are split in two matrices, H_1 and H_2 which represent the channels as seen by the individual receivers Rx_1 and Rx_2 respectively. The four independent vectors of channel coefficients are combined into channel matrices described in equations (30) and (31).

$$H_1 = \begin{bmatrix} h_{11}(n) & h_{21}(n)^* & h_{11}(n+1) & h_{21}(n+1)^* & \ldots \\ h_{21}(n) & -h_{11}(n)^* & h_{21}(n+1) & -h_{11}(n+1)^* & \ldots \end{bmatrix} \quad (30)$$

$$H_2 = \begin{bmatrix} h_{12}(n) & h_{22}(n)^* & h_{12}(n+1) & h_{22}(n+1)^* & \ldots \\ h_{22}(n) & -h_{12}(n)^* & h_{22}(n+1) & -h_{12}(n+1)^* & \ldots \end{bmatrix} \quad (31)$$

3.10. 2x2 ALAMOUTI DECODING

The Alamouti Decoding scheme relies on channel estimation to effectively undo the effects of the channel. For this simulation we assume that the receiver is capable of perfect channel estimation. The channel estimation matrix is constructed and then conjugated as shown in equation (29).

$$H_{eq} = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}^* = \begin{bmatrix} h_{11}(n)^* & (h_{21}(n)^*)^* & \ldots \\ h_{21}(n)^* & (-h_{11}(n)^*)^* & \ldots \\ h_{12}(n)^* & (h_{22}(n)^*)^* & \ldots \\ h_{22}(n)^* & (-h_{12}(n)^*)^* & \ldots \end{bmatrix} \quad (32)$$

Then the conjugated channel estimation matrix multiplies the received signal as shown in equation (33).

$$\begin{bmatrix} \hat{s}(n) \\ \hat{s}(n+1) \end{bmatrix} = \begin{bmatrix} r_1(n) & r_1(n+1) & r_2(n) & r_2(n+1) \\ r_1(n) & r_1(n+1) & r_2(n) & r_2(n+1) \end{bmatrix} \begin{bmatrix} h_{11}(n)^* & (h_{21}(n)^*)^* \\ h_{21}(n)^* & (-h_{11}(n)^*)^* \\ h_{12}(n)^* & (h_{22}(n)^*)^* \\ h_{22}(n)^* & (-h_{12}(n)^*)^* \end{bmatrix} \quad (33)$$

The product is then normalized by dividing by the power of the channel estimation matrix. This ensures that power of the estimate symbols is not affected by the channel estimation.

Now all that is left is the estimates of the transmitted symbols at the receiver and the symbols are arranged back into a vector as shown in equation (34).

$$\hat{S} = [\hat{s}(n) \quad (\hat{s}(n+1))^{*} \quad \ldots \quad \hat{s}(N)\,] \tag{34}$$

3.11. BPSK DEMODULATION

The BPSK demodulation is as straight forward as the modulation. The real component of the received symbol is compared to zero. If greater than zero the symbol demodulates into a binary 1 and if less than zero the symbol demodulates to a binary 0. This process can be seen in equation (35).

$$\hat{x} = real(\hat{s}) > 0; \tag{35}$$

This method is possible since we use channel estimation. Without channel estimation Differential BPSK or DBPSK would have to be used which uses a phase change of π to encode binary state changes.

3.12. BIT ERROR DETECTION

The Bit Error Detection is also straight forward. The absolute value of the estimated bit is subtracted from the original bit. This creates a vector that marks the errors as binary 1's. This vector is then summed to get the total number of bit errors. The Bit Error Rate is then calculated by dividing by the total number of bits (N). The Bit Error Rate calculation can be seen in equation (36).

$$BER = \frac{\sum|(x-\hat{x})|}{N}; \tag{36}$$

3.13. 4x2 ALAMOUTI ENCODING AND DECODING

In an attempt to increase the BER performance of the 2x2 Alamouti code in the broadside case a 4x2 Alamouti coding scheme was developed. The 4x2 encoding scheme considered is not an extension of the 2x2 code to a 4x4 code. This approach has the same anticipated problems as the 2x2 code. The 4x2 scheme simply adds two additional transmit antennas fixed opposing each other on the cylindrical vehicle and perpendicular to the original two antennas, Thus creating a cruciform configuration if viewed down the long axis of the cylinder. This now allows the same data or symbol to be transmitted through antennas opposite one another, and their orthogonal Alamouti counterpart to be transmitted through the remaining two. This insures that both Alamouti signals will be visible at the receiver even in the broadside case.

The 4x2 Alamouti decoding is conducted nearly identical to the 2x2 decoding. This is advantageous since the expansion to the 4x2 case requires little if any additional complexity in the receiver.

4. SIMULATION RESULTS

4.1. BASELINE RESULTS

Baseline Bit Error Curves were generated to verify the channel models and implementation of the 2x2 Alamouti, MRRC and 2x1 systems. The baseline BER curves were created in a non-spinning case to eliminate the effects of the vehicle spin on the performance of the system. The results of the baseline experiment can be seen in Figure 4.1.

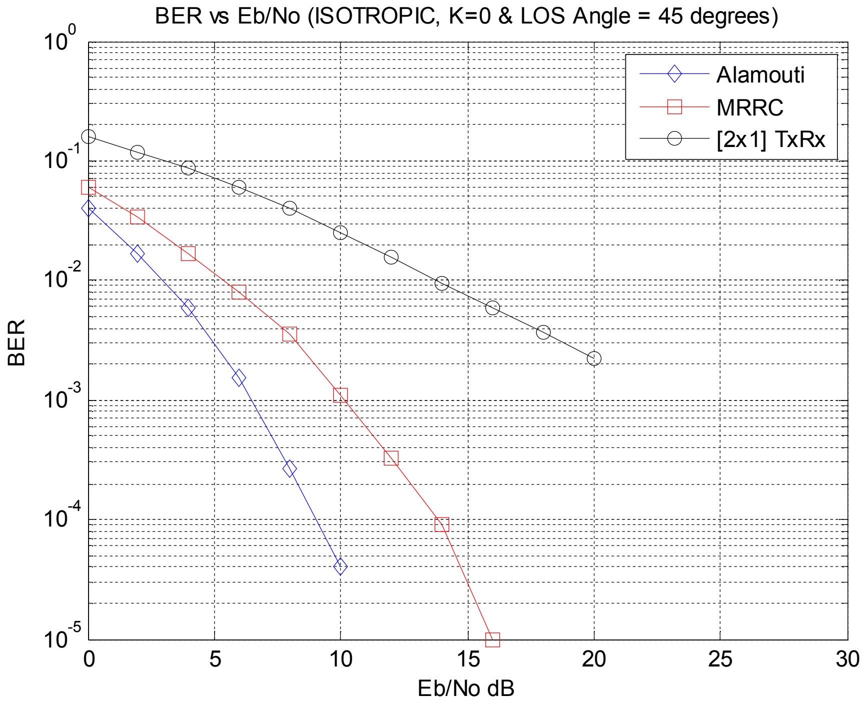

Figure 4.1. Alamouti Code Base-line Results

The baseline BER curves match the results reported by Alamouti for a 2x2 Alamouti encoding scheme. This confirms the performance accuracy of the Alamouti scheme implementation.

4.2. EFFECT OF VEHICLE SPIN AT VARIOUS ANGLES

The Bit Error Rate performance was evaluated for several different line-of-sight angles along an arc of constant range. The LOS angles considered were 0°, 45° and 90° from the horizontal. The results for the 0° LOS case or head-on case with a LOS power coefficient of K=0, are shown in Figure 4.2.

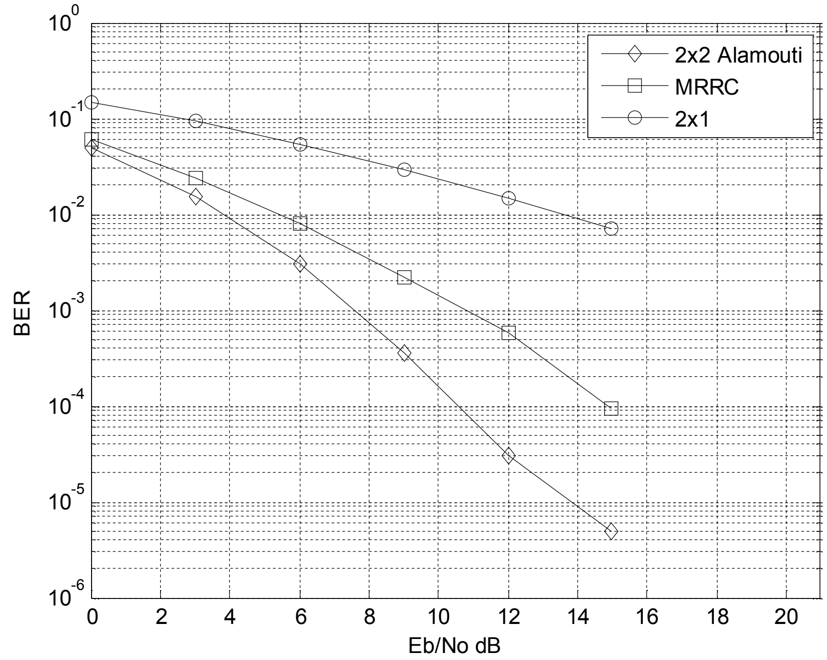

Figure 4.2. BER Curve for LOS Angle = 0°, K = 0

As expected the overall BER performance is decreased by only a small amount. This is because in the head-on case the effect of the vehicle body in obscuring the signal is minimal. In addition, the variation caused by the antenna pattern is small, so for a given SNR the fading effects are dominated by the Rayleigh fading channel. Therefore the results at 0° LOS are consistent with the baseline case and the Alamouti coding outperforms the MRRC receiver by about 2 dB.

The next case considered is the 45° case. The results for the 45° case with a LOS power coefficient of K=0 can be seen in Figure 4.3.

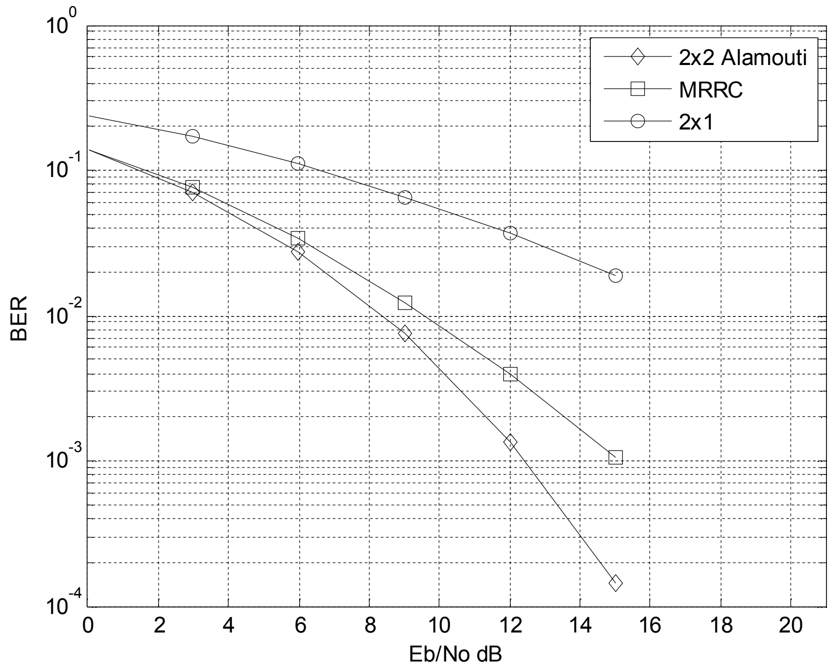

Figure 4.3. BER Curve for LOS Angle = 45°, K = 0

Again as expected the overall BER system performance is decreased by a factor of several dB due to the additional fading caused by the vehicle body obscuring while it spins and the variation caused by the antenna element pattern. The Alamouti coding still outperforms the MRRC receiver, even if by a reduced amount.

The last case presented in this section is the 90° LOS angle or broadside case, with a LOS power ratio of K=0. The results for this case can be seen in Figure 4.4. The overall BER performance is again reduced since the vehicle body blocking and antenna pattern variation is most extreme in the broadside case. The Alamouti still outperforms the MRRC receiver even if only by 1 dB or less.

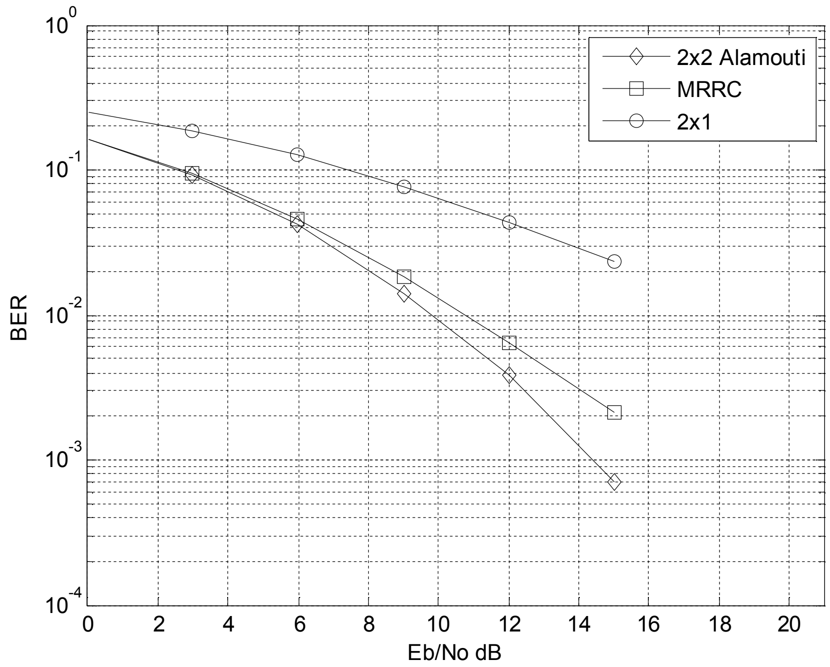

Figure 4.4. BER Curve for LOS Angle = 90°, K = 0

The convergence of the 2x2 Alamouti BER curve with the MRRC receiver BER curve was expected. As the vehicle body begins to block the signal from the far transmitter sufficiently, the Alamouti coding scheme devolves into the MRRC case. The coding gain from the Alamouti code is eliminated and the result is that the receiver only sees one transmitter and the two receive antennas add that signal coherently which leaves only the spatial diversity gain from the MRRC receiver. This result is important since it shows that the2x2 Alamouti code works even when one transmitting antenna is eliminated and still receives the benefits of spatial diversity at the receiver.

4.3. EFFECT OF LINE-OF-SIGHT

The effect of the deterministic line-of-sight model combined with the Rayleigh Scattering model into a modified Ricean Scattering model is evaluated in this section. The same LOS angles were considered as the previous section which used solely the Rayleigh Scattering model. The LOS angle is varied along an arc of constant range at fixed points at 0°, 45° and 90°. For the following results the LOS power ratio K is fixed at 2, which gives us a fairly dominant LOS component without completely overwhelming the Rayleigh fading effect. The initial head-on case with a LOS angle equal to 0° and a LOS power ratio K = 2 can be seen in Figure 4.5.

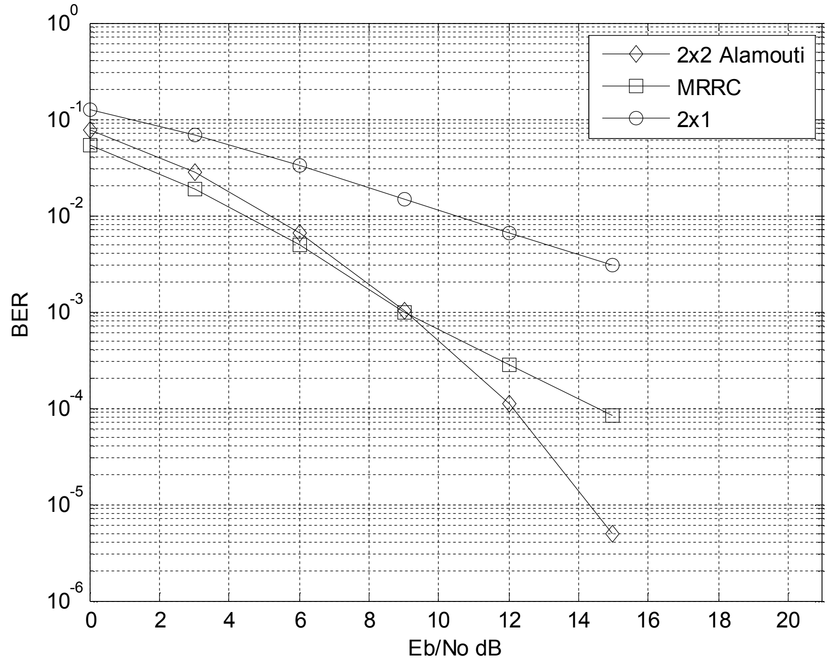

Figure 4.5. BER Curve for LOS Angle = 0°, K = 2

In the results given in Figure 4.5 it is obvious that the 2x2 Alamouti and MRRC receivers perform nearly identically. This is due to the effect of the strong line-of-sight component in the channel. The Alamouti code was designed to provide a gain in SNR in fast-fading Rayleigh distributed channels, and the addition of a strong LOS component removes that gain over the MRRC receiver. This is confirmed for the remaining LOS angles 45° and 90° in Figure 4.6 and Figure 4.7 respectively.

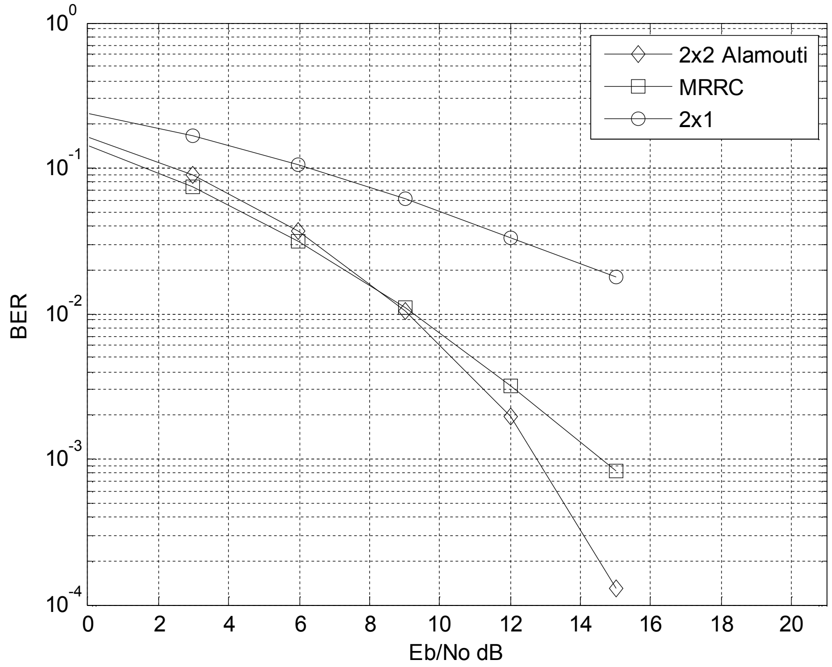

Figure 4.6. BER Curve for LOS Angle = 45°, K =2

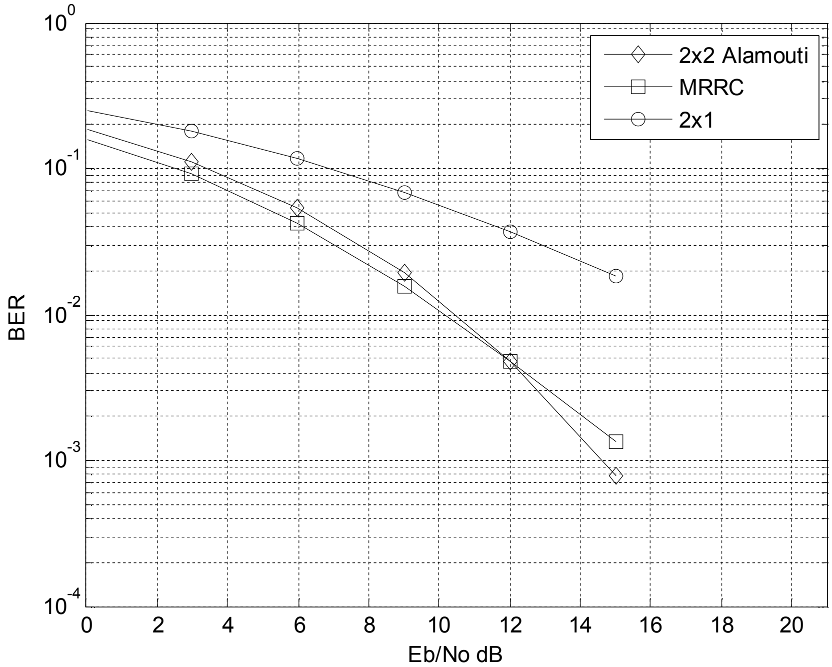

Figure 4.7. BER Curve for LOS Angle = 90°, K = 2

4.4. 4x2 RESULTS

A proposed solution to increase the BER of the Alamouti encoding scheme when the vehicle is at a LOS angle equal to 90°, or the broadside case is to add an additional two antennas. These additional two antennas are located on opposite sides of the cylindrical vehicle, perpendicular to the other set of antennas. The Alamouti code used is not a 4x4 extension of the 2x2 Alamouti code, but instead the same 2x2 code transmitted out of two antennas simultaneously. Using this method the space-time coding gain is maintained since a particular transmitted Alamouti symbol is never fully obscured at the receiver. Another advantage of using this 4x2 Alamouti code is only two antennas are required and there is almost no additional computational complexity at the receiver for the Alamouti decoding.

The 4x2 Alamouti case is compared to the 2x2 Alamouti case at the same LOS angles previously considered. Figure 4.8 shows the BER curves at 0° LOS and K equal to 0.

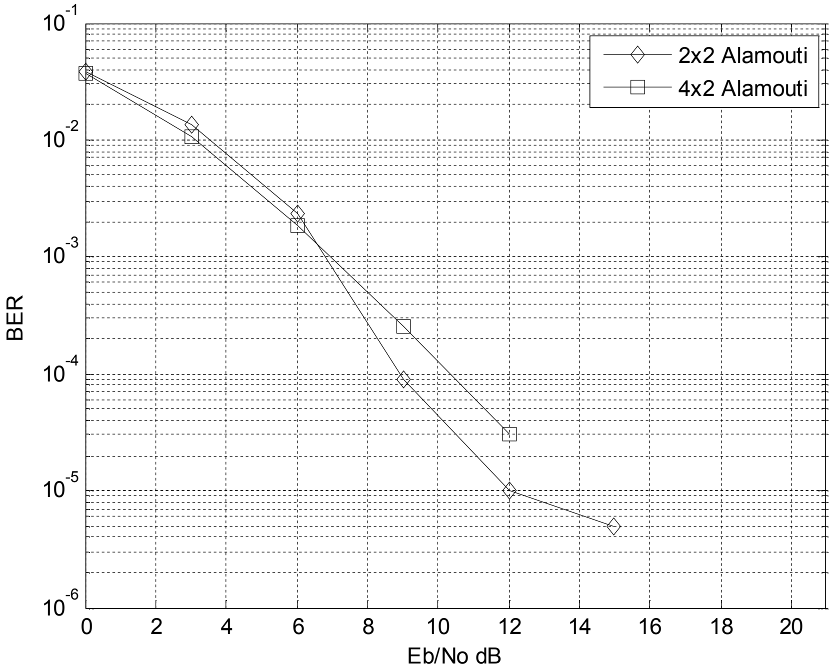

Figure 4.8. Alamouti Comparison BER Curve for LOS Angle = 0°, K = 0

The 4x2 Alamouti case performs consistently with the 2x2 Alamouti case as expected. At low LOS angles there is almost no signal blocking by the body and little variation in antenna pattern. Figure 4.9 shows the results for LOS angle equal to 45° and K equal to 0.

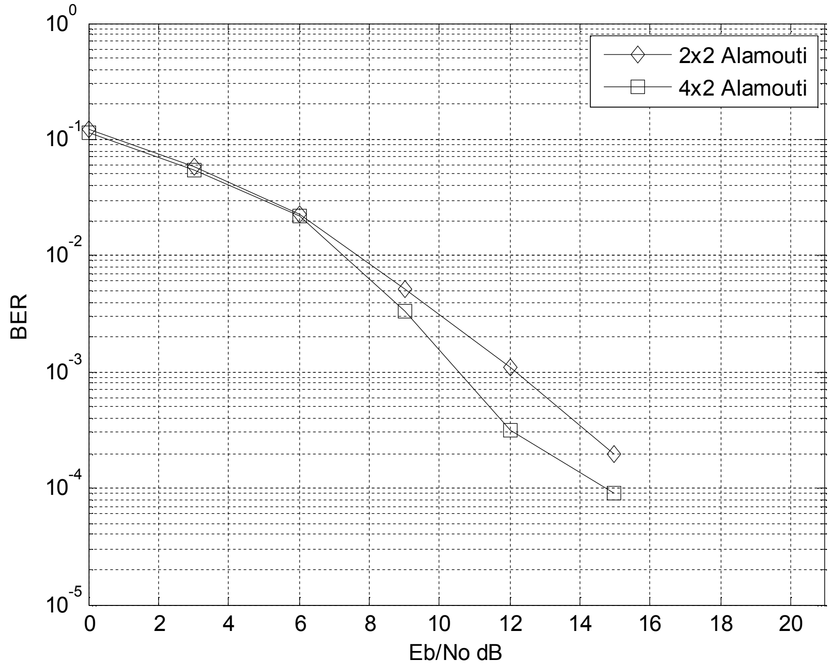

Figure 4.9. Alamouti Comparison BER Curve for LOS Angle = 45°, K = 0

Again as expected the 4x2 Alamouti case offers little improvement over the 2x2 Alamouti case. In Figure 4.10 the BER curve for the broadside case, LOS angle equal to 90° and K equal to 0 can be seen

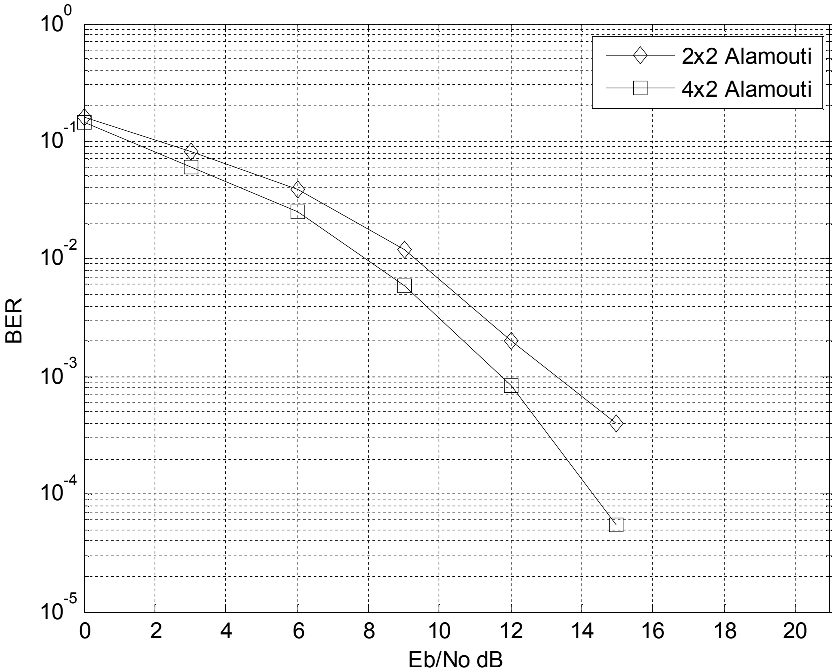

Figure 4.10. Alamouti Comparison BER Curve for LOS Angle = 90°, K = 0

In the broadside case significant improvement in 4x2 Alamouti performances over the 2x2 Alamouti case is demonstrated. The 4x2 case outperforms the 2x2 case by a factor of several dB.

5. CONCLUSION

The performance of a Multiple-Input Multiple-Output (MIMO) wireless communication system, when the transmitter is located on a spinning vehicle was investigated. A 2x2 MIMO system was used, with Alamouti coding at the transmitter. The channel models used in the simulation were a Rayleigh and Ricean flat fading models with added deterministic effects from the vehicle spin. By simulating the bit-error rate the 2x2 Alamouti channel code is shown to perform better than the Maximal Ratio Receiver Combining (MRRC) and the single receiver (2x1) system in some circumstances and performs similarly to the MRRC in the broadside case. The asymmetric 4x2 Alamouti code is shown to improve the performance of the 2x2 Alamouti code in the broadside case.

APPENDIX A

2X2 ALAMOUTI CODE SIMULATION

```
Function [BitErr,CumBitErr]
=AlamoutiSpin_wRician(Tsym,Ns,Eb_No_dB,K,THETA,MAX_SEP_TX,RotatFreq)
%===========================================================================
% Alamouti Coding w/ Ricean and AWGN Channel
% Author: Samuel Petersen
% Ver. Date: 05/17/2010
%===========================================================================

% --------------------- PARAMS: -------------------------------------------
%   Tsym = Symbol Duration (Symbol Period)
%   Ns = Number of Symbols for experiment
%   Eb_No_dB = Signal-to-Noise Ration in dB
%   K = Power ratio of LOS to Scatterer
%   THETA = LOS angle relative to vehicle
%   RotatFreq = Frequency of vehicle rotation
%---------------------------------------------------------------------------

% Reset Random Data
stream = RandStream.getDefaultStream;
% reset(stream);
Ts = Tsym;
Eb_No = 10.^(Eb_No_dB./10);     % Conv from dB
Eb = 1;                         % Constant Eb
No = Eb / Eb_No;                % Calculate No

% Calculate Random Data
data = rand(stream,1,Ns) > 0.5;
% BPSK Modulation
S = 2*data-1;
% Reshape for Alamouti Coding
Sx = sqrt(Eb/2).*kron(reshape(S,2,Ns/2),ones(1,2));

fd = 50;
nReflect = 100;
Nh = Ns/2;

% Time vector for RAYLEIGH func and Ricean Channel
t = [0:2:Ns-1].*Ts;

% Vehicle Rotation
if RotatFreq ~= 0
    bits_rot = (1/(Tsym*RotatFreq));  % number of bits per 1-rotation
    dTh =(THETA+THETA)/(bits_rot/2);          % step size for 1/2-rotation

    if THETA ~= 0
        temp = [pi-THETA:dTh:pi+THETA-dTh];
        theta_rot = [temp fliplr(temp)];
    else
        theta_rot = zeros(1,bits_rot);
    end
    R = ceil(Nh/bits_rot);
    theta = [];
    for n = 1:R
        if n == R
            % Account for an non integer number of rotations in simulation
            theta = [theta theta_rot(1:round(bits_rot - bits_rot*(R -
Nh/bits_rot)))];
        else
            theta = [theta theta_rot];
        end
    end
```

```
else
    theta = THETA.*ones(1,Nh);
end

% Patch Antenna
% Pattern Definition
PHI = [0:(2*pi)/24:2*pi];
Pattern_dB = -[14 10 6 4 3 0 0 0 3 4 6 10 14 15 32 20 20 28 29 24 20 20 27 15
14];
% Convert from dB
Pat1 = 10.^(Pattern_dB/20);
Pat2 = fliplr(Pat1);            % Flip Pattern 180*
% Interpolate to the required resolution
Pattern1 = interp1(PHI,Pat1,theta,'spline');
Pattern2 = interp1(PHI,Pat2,theta,'spline');

P1 = Pattern1/max(Pattern1);
P2 = Pattern2/max(Pattern2);

% Add in effect of spinning on LOS
phi_block = [0:(2*pi)/bits_rot:2*pi-(1/bits_rot)];
R = ceil(Nh/bits_rot);
phi_t = [];
for n = 1:R
    if n == R
        % Account for an non integer number of rotations in simulation
        phi_t = [phi_t phi_block(1:round(bits_rot - bits_rot*(R -
Nh/bits_rot)))];
    else
        phi_t = [phi_t phi_block];
    end
end

Range = 5e2;
Rx_sep = 2;         % Separation in meters

d_t = MAX_SEP_TX;

THETA_1 = atan2((Range*sin(THETA)),((Range*cos(THETA)) - (Rx_sep/2)));
THETA_2 = atan2((Range*sin(THETA)),((Range*cos(THETA)) + (Rx_sep/2)));

% Considering this as an antenna array
gamma = acos(cos(phi_t).*sin(THETA_1));
AF1 = cos(pi*d_t*cos(gamma)).^2;

gamma = acos(cos(phi_t).*sin(THETA_2));
AF2 = cos(pi*d_t*cos(gamma)).^2;

% --- Calculate Ricean Channel -----------------------------------------
X1 = RAYLEIGH(t,Ts,fd,nReflect);
% Add Line-of-Sight to Rayleigh Scatters
% Real Component
Zc = (real(X1) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X1) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z1 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z1(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z1(2:2:end) = sqrt(P1).*(Zc+1i*Zs);
```

```
% --- 2nd independent channel ----------------------------------------------
X2 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X2) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X2) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z2 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z2(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z2(2:2:end) = sqrt(P2).*(Zc+1i*Zs);
H1 = [Z1;Z2];

% --- 3rd independent channel ----------------------------------------------
X3 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X3) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X3) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z3 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z3(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z3(2:2:end) = sqrt(P1).*(Zc+1i*Zs);

% --- 4th independent channel ----------------------------------------------
X4 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X4) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X4) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z4 = zeros(1,Ns);
% Create Complex Coefficents and Repeat for Alamouti Coding
Z4(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z4(2:2:end) = sqrt(P2).*(Zc+1i*Zs);
H2 = [Z3;Z4];

t = [0:Ns-1]*Ts;

% Reshape Channel Matrix Rx1 and Code using Alamouti
temp = H1;
H1(1,[2:2:end]) = conj(temp(2,[2:2:end]));
H1(2,[2:2:end]) = -conj(temp(1,[2:2:end]));

% Reshape Channel Matrix Rx2
temp = H2;
H2(1,[2:2:end]) = conj(temp(2,[2:2:end]));
H2(2,[2:2:end]) = -conj(temp(1,[2:2:end]));

% Calculate White Gaussian Noise
% Create White Gaussian Noise
nWGN = sqrt(No)*[randn(2,Ns) + 1i*randn(2,Ns)];

Rx = zeros(2,Ns);
Heq = zeros(4,Ns);
% Calculate Received Signal
Rx(1,:) = sum(H1.*Sx,1) + nWGN(1,:);
Rx(2,:) = sum(H2.*Sx,1) + nWGN(2,:);

% Create Equalizer Matrix
Heq([1:2],:) = H1; Heq([3:4],:) = H2;
Heq(1,[1:2:end]) = conj(Heq(1,[1:2:end]));
Heq(2,[2:2:end]) = conj(Heq(2,[2:2:end]));
Heq(3,[1:2:end]) = conj(Heq(3,[1:2:end]));
Heq(4,[2:2:end]) = conj(Heq(4,[2:2:end]));

% Reshape for Receiver Equalizer
y_eq = zeros(4,Ns);
y_eq([1:2],:) = kron(reshape(Rx(1,:),2,Ns/2),ones(1,2));
y_eq([3:4],:) = kron(reshape(Rx(2,:),2,Ns/2),ones(1,2));

% Calculate Equalizer Power
hEqPower = sum(Heq.*conj(Heq),1);
% Estimate
y_hat = sum(Heq.*y_eq,1)./hEqPower;
y_hat(2:2:end) = conj(y_hat(2:2:end));

% Decode Equalized Signal
y_est = real(y_hat)>0;

% Calculate Errors
CumBitErr = cumsum(abs(y_est-data))./([1:Ns]);
BitErr = sum(abs(y_est-data))/Ns;
```

APPENDIX B

MRRC SIMULATION

```
function [BitErr,CumBitErr] =
MRRC_wSpin(Tsym,Ns,Eb_No_dB,K,THETA,MAX_SEP_TX,RotatFreq)
%==========================================================================
% MRRC w/ a spinning vehicle in a Rician/AWGN channel     [2x2]
% Author: Samuel Petersen
% Ver. Date: 05/16/2010
%==========================================================================

% Reset Random Data
stream = RandStream.getDefaultStream;
reset(stream);
Ts = Tsym;
Eb_No = 10.^(Eb_No_dB./10);   % Conv from dB
Eb = 1;                     % Constant Eb
No = Eb / Eb_No;            % Calculate No

% Calculate Random Data
data = rand(stream,1,Ns) > 0.5;
% BPSK Modulation
S = 2*data-1;
% Reshape
Sx = zeros(2,Ns);
Sx(1,:) = (sqrt(1/2)*S);
Sx(2,:) = (sqrt(1/2)*S);

% PwerSx = (mean(abs(Sx(1,:)).^2)).*(Ns-1).*Ts
% PwerS = (mean(abs(S).^2)).*(Ns-1).*Ts

fd = 50;
nReflect = 100;
Nh = Ns/2;
% Calculate the Channel Coefficients
% K = 5;        % power ratio of the LOS to scatters (input of func)

% Time vector for RAYLEIGH func and Ricean Channel
t = [0:2:Ns-1].*Ts;

% Vehicle Rotation
if RotatFreq ~= 0
    bits_rot = (1/(Tsym*RotatFreq));  % number of bits per 1-rotation
    dTh =(THETA+THETA)/(bits_rot/2);          % step size for 1/2-rotation

    if THETA ~= 0
        temp = [pi-THETA:dTh:pi+THETA-dTh];
        theta_rot = [temp fliplr(temp)];
    else
        theta_rot = zeros(1,bits_rot);
    end
    R = ceil(Nh/bits_rot);
    theta = [];
    for n = 1:R
        if n == R
            % Account for an non integer number of rotations in simulation
            theta = [theta theta_rot(1:round(bits_rot - bits_rot*(R -
Nh/bits_rot)))];
        else
            theta = [theta theta_rot];
        end
    end
```

```
else
    theta = THETA.*ones(1,Nh);
end

 % Patch Antenna
% Pattern Definition
PHI = [0:(2*pi)/24:2*pi];
Pattern_dB = -[14 10 6 4 3 0 0 0 3 4 6 10 14 15 32 20 20 28 29 24 20 20 27 15
14];
% Convert from dB
Pat1 = 10.^(Pattern_dB/20);
Pat2 = fliplr(Pat1);            % Flip Pattern 180*
% Interpolate to the required resolution
Pattern1 = interp1(PHI,Pat1,theta,'spline');
Pattern2 = interp1(PHI,Pat2,theta,'spline');

P1 = Pattern1/max(Pattern1);
P2 = Pattern2/max(Pattern2);

% Add in effect of spinning on LOS
phi_block = [0:(2*pi)/bits_rot:2*pi-(1/bits_rot)];
R = ceil(Nh/bits_rot);
phi_t = [];
for n = 1:R
        if n == R
            % Account for an non integer number of rotations in simulation
            phi_t = [phi_t phi_block(1:round(bits_rot - bits_rot*(R -
Nh/bits_rot)))];
        else
            phi_t = [phi_t phi_block];
        end
end

Range = 5e2;
Rx_sep = 2;         % Separation in meters
d_t = MAX_SEP_TX;

THETA_1 = atan2((Range*sin(THETA)),((Range*cos(THETA)) - (Rx_sep/2)));
THETA_2 = atan2((Range*sin(THETA)),((Range*cos(THETA)) + (Rx_sep/2)));

% Considering this as an antenna array
gamma = acos(cos(phi_t).*sin(THETA_1));
AF1 = cos(pi*d_t*cos(gamma)).^2;

gamma = acos(cos(phi_t).*sin(THETA_2));
AF2 = cos(pi*d_t*cos(gamma)).^2;

% --- Calculate Ricean Channel ------------------------------------------
X1 = RAYLEIGH(t,Ts,fd,nReflect);
% Add Line-of-Sight to Rayleigh Scatters
% Real Component
Zc = (real(X1) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X1) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z1 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z1(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z1(2:2:end) = sqrt(P1).*(Zc+1i*Zs);

% --- 2nd independent channel --------------------------------------------
```

```
X2 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X2) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X2) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z2 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z2(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z2(2:2:end) = sqrt(P2).*(Zc+1i*Zs);
H1 = [Z1;Z2];

% --- 3rd independent channel ----------------------------------------------
X3 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X3) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X3) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z3 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z3(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z3(2:2:end) = sqrt(P1).*(Zc+1i*Zs);

% --- 4th independent channel ----------------------------------------------
X4 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X4) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X4) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z4 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z4(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z4(2:2:end) = sqrt(P2).*(Zc+1i*Zs);
H2 = [Z3;Z4];

t = [0:Ns-1]*Ts;

% Reshape Channel Matrix Rx1
temp = H1;
H1(1,[2:2:end]) = temp(2,[2:2:end]);
H1(2,[2:2:end]) = temp(1,[2:2:end]);

% Reshape Channel Matrix Rx2
temp = H2;
H2(1,[2:2:end]) = temp(2,[2:2:end]);
H2(2,[2:2:end]) = temp(1,[2:2:end]);

% Calculate White Gaussian Noise
% Create White Gaussian Noise
nWGN = sqrt(No)*[randn(2,Ns) + 1i*randn(2,Ns)];

Rx = zeros(2,Ns);
Heq = zeros(2,Ns);
% Cacluate Recieved Signal
Rx(1,:) = sum(H1.*Sx,1) + nWGN(1,:);
Rx(2,:) = sum(H2.*Sx,1) + nWGN(2,:);

% Create Equalizer Matrix
Heq(1,:) = conj(H1(1,:)+H1(2,:));
Heq(2,:) = conj(H2(1,:)+H2(2,:));
```

```
% Calculate Equalizer Power
hEqPower = sum(Heq.*conj(Heq),1);
% Estimate
y_hat = sum(Heq.*Rx,1)./hEqPower;

% Decode Equalized Signal
y_est = real(y_hat)>0;

% Calculate Errors
CumBitErr = cumsum(abs(y_est-data))./([1:Ns]);
BitErr = sum(abs(y_est-data))/Ns;
```

APPENDIX C

4X2 ALAMOUTI CODE SIMULATION

```
function [BitErr,CumBitErr] =
Alamouti4x2_wRician(Tsym,Ns,Eb_No_dB,K,THETA,MAX_SEP_TX,RotatFreq)
%==========================================================================
% Alamouti Coding w/ Ricean and AWGN Channel with 4x2
% Author: Samuel Petersen
% Ver. Date: 11/07/2010
%==========================================================================

% --------------------- PARAMS: -------------------------------------------
%   Tsym = Symbol Duration (Symbol Period)
%   Ns = Number of Symbols for experiment
%   Eb_No_dB = Signal-to-Noise Ration in dB
%   K = Power ratio of LOS to Scatterer
%   THETA = LOS angle relative to vehicle
%   RotatFreq = Frequency of vehicle rotation
%--------------------------------------------------------------------------

% Reset Random Data
stream = RandStream.getDefaultStream;
% reset(stream);
Ts = Tsym;
Eb_No = 10.^(Eb_No_dB./10);     % Conv from dB
Eb = 1;                         % Constant Eb
No = Eb / Eb_No;                % Calculate No

% Calculate Random Data
data = rand(stream,1,Ns) > 0.5;
% BPSK Modulation
S = 2*data-1;
% Reshape for Alamouti Coding
temp = sqrt(Eb/4).*kron(reshape(S,2,Ns/2),ones(1,2));
Sx = zeros(4,Ns);
Sx(1,:) = temp(1,:);
Sx(2,:) = temp(1,:);
Sx(3,:) = temp(2,:);
Sx(4,:) = temp(2,:);

fd = 50;
nReflect = 100;
Nh = Ns/2;

% Time vector for RAYLEIGH func and Ricean Channel
t = [0:2:Ns-1].*Ts;

%--- Vehicle Rotation -----------------------------------------------------
if RotatFreq ~= 0
    bits_rot = (1/(Tsym*RotatFreq));  % number of bits per 1-rotation
    dTh =(2*THETA)/(bits_rot/2);          % step size for 1/2-rotation

    if THETA ~= 0
        temp = [pi-THETA:dTh:pi+THETA-dTh];
        theta_rot = [temp fliplr(temp)];
    else
        theta_rot = zeros(1,bits_rot);
    end
    R = ceil(Nh/bits_rot);
    theta12 = [];
    for n = 1:R
        if n == R
            % Account for an non integer number of rotations in simulation
            theta12 = [theta12 theta_rot(1:round(bits_rot - bits_rot*(R - 
Nh/bits_rot)))];
        else
            theta12 = [theta12 theta_rot];
        end
    end
    % for Antenna's 3 and 4
    dTh = (2*THETA)/(bits_rot/2);           % step size for 1/4-rotation
    if THETA ~= 0
        temp1 = [pi:dTh:pi+THETA-dTh];
        temp2 = [pi-THETA+dTh:dTh:pi];
        theta_rot = [temp1 fliplr(temp1) fliplr(temp2) temp2];
    else
        theta_rot = zeros(1,bits_rot);
    end
    R = ceil(Nh/bits_rot);
    theta34 = [];
    for n = 1:R
        if n == R
            % Account for an non integer number of rotations in simulation
            theta34 = [theta34 theta_rot(1:round(bits_rot - bits_rot*(R - 
Nh/bits_rot)))];
        else
            theta34 = [theta34 theta_rot];
        end
    end
else
    theta12 = THETA.*ones(1,Nh);
    theta34 = THETA.*ones(1,Nh);
end

% Patch Antenna
% Pattern Definition
PHI = [0:(2*pi)/24:2*pi];
Pattern_dB = -[14 10 6 4 3 0 0 0 3 4 6 10 14 15 32 20 20 28 29 24 20 20 27 15 
14];
% Convert from dB
Pat1 = 10.^(Pattern_dB/20);
Pat2 = fliplr(Pat1);            % Flip Pattern 180*
% Interpolate to the required resolution
Pattern1 = interp1(PHI,Pat1,theta12,'spline');
Pattern2 = interp1(PHI,Pat2,theta12,'spline');
Pattern3 = interp1(PHI,Pat2,theta34,'spline');
Pattern4 = interp1(PHI,Pat1,theta34,'spline');

P1 = Pattern1/max(Pattern1);
P2 = Pattern2/max(Pattern2);
P3 = Pattern3/max(Pattern3);
P4 = Pattern4/max(Pattern4);

%--- Add in effect of spinning on LOS -------------------------------------
phi_block = [0:(2*pi)/bits_rot:2*pi-(1/bits_rot)];
R = ceil(Nh/bits_rot);
phi = [];
for n = 1:R
    if n == R
        % Account for an non integer number of rotations in simulation
        phi = [phi phi_block(1:round(bits_rot - bits_rot*(R - Nh/bits_rot)))];
    else
        phi = [phi phi_block];
    end
end
```

```
Range = 5e2;
Rx_sep = 2;         % Separation in meters

d_t = MAX_SEP_TX;

THETA_1 = atan2((Range*sin(THETA)),((Range*cos(THETA)) - (Rx_sep/2)));
THETA_2 = atan2((Range*sin(THETA)),((Range*cos(THETA)) + (Rx_sep/2)));

phi12 = phi;
phi34 = phi+pi/2;
% Considering this as an antenna array
gamma12 = acos(cos(phi12).*sin(THETA_1));
gamma34 = acos(cos(phi34).*sin(THETA_1));
AF1 = cos(pi*d_t*cos(gamma12)).^2 + cos(pi*d_t*cos(gamma34)).^2;

gamma12 = acos(cos(phi12).*sin(THETA_2));
gamma34 = acos(cos(phi34).*sin(THETA_2));
AF2 = cos(pi*d_t*cos(gamma12)).^2 + cos(pi*d_t*cos(gamma34)).^2;

% --- Calculate Ricean Channel ------------------------------------------
X1 = RAYLEIGH(t,Ts,fd,nReflect);
% Add Line-of-Sight to Rayleigh Scatters
% Real Component
Zc = (real(X1) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X1) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z1 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z1(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z1(2:2:end) = sqrt(P1).*(Zc+1i*Zs);

% --- 2nd independent channel ------------------------------------------
X2 = RAYLEIGH(t,Ts,fd,nReflect);
% Add Line-of-Sight to Rayleigh Scatters
% Real Component
Zc = (real(X2) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X2) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z2 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z2(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z2(2:2:end) = sqrt(P2).*(Zc+1i*Zs);

% --- 3nd independent channel ------------------------------------------
X3 = RAYLEIGH(t,Ts,fd,nReflect);
% Add Line-of-Sight to Rayleigh Scatters
% Real Component
Zc = (real(X3) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
% Imaginary Component
Zs = (imag(X3) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z3 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z3(1:2:end) = sqrt(P3).*(Zc+1i*Zs);
Z3(2:2:end) = sqrt(P3).*(Zc+1i*Zs);

% --- 4nd independent channel ---------------------------------------------
X4 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X4) + sqrt(AF1.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
```

```
% Imaginary Component
Zs = (imag(X4) + sqrt(AF1.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF1.*K);
Z4 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z4(1:2:end) = sqrt(P4).*(Zc+1i*Zs);
Z4(2:2:end) = sqrt(P4).*(Zc+1i*Zs);
H1 = [Z1;Z2;Z3;Z4];

% --- 5rd independent channel ---------------------------------------------
X5 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X5) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X5) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z5 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z5(1:2:end) = sqrt(P1).*(Zc+1i*Zs);
Z5(2:2:end) = sqrt(P1).*(Zc+1i*Zs);

% --- 6rd independent channel ---------------------------------------------
X6 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X6) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X6) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z6 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z6(1:2:end) = sqrt(P2).*(Zc+1i*Zs);
Z6(2:2:end) = sqrt(P2).*(Zc+1i*Zs);

% --- 7rd independent channel ---------------------------------------------
X7 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X7) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X7) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z7 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z7(1:2:end) = sqrt(P3).*(Zc+1i*Zs);
Z7(2:2:end) = sqrt(P3).*(Zc+1i*Zs);

% --- 8th independent channel ---------------------------------------------
X8 = RAYLEIGH(t,Ts,fd,nReflect);
% Real Component
Zc = (real(X8) + sqrt(AF2.*K).*cos(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
% Imaginary Component
Zs = (imag(X8) + sqrt(AF2.*K).*sin(2*pi*fd*t*cos(THETA)))./sqrt(1+AF2.*K);
Z8 = zeros(1,Ns);
% Create Complex Coefficients and Repeat for Alamouti Coding
Z8(1:2:end) = sqrt(P4).*(Zc+1i*Zs);
Z8(2:2:end) = sqrt(P4).*(Zc+1i*Zs);
H2 = [Z5;Z6;Z7;Z8];

t = [0:Ns-1]*Ts;

% Reshape Channel Matrix Rx1 and Code using Alamouti
temp = H1;
H1(1,[2:2:end]) = conj(temp(3,[2:2:end]));
H1(2,[2:2:end]) = conj(temp(4,[2:2:end]));
H1(3,[2:2:end]) = -conj(temp(1,[2:2:end]));
H1(4,[2:2:end]) = -conj(temp(2,[2:2:end]));
```

```
% Reshape Channel Matrix Rx2
temp = H2;
H2(1,[2:2:end]) = conj(temp(3,[2:2:end]));
H2(2,[2:2:end]) = conj(temp(4,[2:2:end]));
H2(3,[2:2:end]) = -conj(temp(1,[2:2:end]));
H2(4,[2:2:end]) = -conj(temp(2,[2:2:end]));

% Calculate White Gaussian Noise
% Create White Gaussian Noise
nWGN = sqrt(No)*[randn(4,Ns) + 1i*randn(4,Ns)];

Rx = zeros(2,Ns);
Heq = zeros(4,Ns);
% Calculate Received Signal
Rx(1,:) = sum(H1.*Sx,1) + nWGN(1,:);
Rx(2,:) = sum(H2.*Sx,1) + nWGN(2,:);

% Create Equalizer Matrix
Heq(1,:) = H1(1,:)+H1(2,:); Heq(2,:) = H1(3,:)+H1(4,:);
Heq(3,:) = H2(1,:)+H2(2,:); Heq(4,:) = H2(3,:)+H2(4,:);

Heq(1,[1:2:end]) = conj(Heq(1,[1:2:end]));
Heq(2,[2:2:end]) = conj(Heq(2,[2:2:end]));
Heq(3,[1:2:end]) = conj(Heq(3,[1:2:end]));
Heq(4,[2:2:end]) = conj(Heq(4,[2:2:end]));

% Reshape for Receiver Equalizer
y_eq = zeros(4,Ns);
y_eq([1:2],:) = kron(reshape(Rx(1,:),2,Ns/2),ones(1,2));
y_eq([3:4],:) = kron(reshape(Rx(2,:),2,Ns/2),ones(1,2));

% Calculate Equalizer Power
hEqPower = sum(Heq.*conj(Heq),1);
% Estimate
y_hat = sum(Heq.*y_eq,1)./hEqPower;
y_hat(2:2:end) = conj(y_hat(2:2:end));

% Decode Equalized Signal
y_est = real(y_hat)>0;

% Calculate Errors
CumBitErr = cumsum(abs(y_est-data))./([1:Ns]);
BitErr = sum(abs(y_est-data))/Ns;
```

APPENDIX D

RAYLEIGH CHANNEL SIMULATION

```
%==========================================================================
% Rayleigh Channel Simulation (Improved Jakes' Model - Xiao & Zheng)
% EE 401 Wireless Communications
% =========================================================================

function [X_t]=RAYLEIGH(t,Ts,fd,M)

%------ Parameters --------------------------------------------------------
%   M = Number of reflectors
%   fd = Maximum Doppler frequency
%   t = time vector
%--------------------------------------------------------------------------

% create index vector n
n=[1:M];

wd=2*pi*fd;
% Create random vectors uniformly dist from [0,2*pi]
phi= 2*pi*(rand(1,M)-0.5);
theta= 2*pi*(rand(1,M)-0.5);
psi = 2*pi*(rand(1,M)-0.5);

alpha=(2*pi*n-pi+theta)/(4*M);

Xc_t=zeros(1,length(t));Xs_t=zeros(1,length(t));
for n=1:M
    Xc_tn=cos(wd*t*cos(alpha(n))+phi(n));
    Xs_tn=sin(wd*t*sin(alpha(n))+psi(n));

    Xc_t=Xc_t+Xc_tn;
    Xs_t=Xs_t+Xs_tn;
end

X_t= sqrt(2/M)*(Xc_t + 1i*Xs_t);
```

BIBLIOGRAPHY

Alamouti, Siavash M. "A Simple Transmit Diversity Technique for Wireless Communications." *IEEE Journal on Select Areas in Communications* 16.8 (1998). Print.

Balanis, Constantine A. *Antenna Theory Analysis and Design*. 3rd ed. Hoboken, NJ: Wiley-Interscience, 2005. Print.

Chengshan, Xiao, Y. R. Zheng, and N. C. Beaulieu. "Statistical Simulation Models for Rayleigh and Rician Fading." *Communications, 2003. ICC '03 IEEE International Conference on* 5 (2003). Print.

Hogg, Robert V., and Elliot A. Tanis. *Probability and Statistical Inference*. 7th ed. Upper Saddle River, NJ: Prentice Hall, 2006. Print.

Jankiraman, Mohinder. *Space-time Codes and MIMO Systems*. Boston: Artech House, 2004. Print.

Oestges, Claude, and Bruno Clerckx. *MIMO Wireless Communications: From Real-World Propagation to Space-Time Code Design.* Amsterdam: Elsevier, 2007. Print.

Pop, M. F., and N. C. Beaulieu. "Design of Wide-sense Stationary Sum-of-sinusoids Fading Channel Simulators." *Communications, 2002 IEEE International Conference on* 2 (2002). Print.

Rappaport, Theodore S. *Wireless Communications: Principles and Practice. Rappaport.* 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1996. Print.

Ziemer, Rodger E., and Roger L. Peterson. *Introduction to Digital Communication*. 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2001. Print.

VITA

Samuel James Petersen was born in Springfield, MO on October 6, 1986. On graduating from Rolla High School in 2005, he enrolled in the University of Missouri – Rolla, which is now the Missouri University of Science and Technology. In 2009 he completed a Bachelor of Science in Electrical Engineering. He then began work towards a Master's of Science in Electrical Engineering at Missouri University of Science and Technology, and was awarded the Chancellor's Fellowship. He completed his degree requirements in 2010. On finishing the course work he began work at Dynetics, Inc in Huntsville, Alabama and will receive his Masters in Electrical Engineering in May 2011.