Low Complexity MIMO Detection Algorithms and Implementations

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Lian Huai

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Professor Gerald E. Sobelman, Advisor

December, 2014

Acknowledgements

First and foremost, I would like to give my deepest thanks to my advisor Professor Gerald E. Sobelman. Over the past years, I have received a lot of guidance, support and encouragement from him. I appreciate all his contributions of time, help, and caring to make my Ph.D. experience stimulating.

I would also like to express my acknowledgements to my dissertation committee of Professor Keshab K. Parhi, Professor Marc Riedel, and Professor William L. Cooper from Industrial & Systems Engineering Department. Thanks for their helpful comments, suggestions and support to my oral exam, final defense and dissertation.

In my later work of MIMO detection algorithm design, I am particularly indebted to Samer Hijazi and Raul Casas for their inspiration and support. The discussion with them have always been enlightening and helpful.

I gratefully acknowledge the Department of Electrical & Computer Engineering for the financial support for my Ph.D. study. The last five years here is an invaluable experience for me.

My time at Minnesota was made enjoyable in large part due to the many friends that became a part of my life. I am grateful for the help of friends who shared the happiness with me and also inspired my research. I would like to give the thanks to Yuma Han, Peng Li.

Lastly, I would like to thank my family for all their love and encouragement. For my parents who raised me with love and supported me in all my pursuits.

Abstract

Currently, wireless communication is playing a very important role in everyone's daily life. People can no longer live without internet, smart phones, Wi-Fi, cellular data service and wireless sensor networks. As a result of Moore's law, the downsizing of silicon feature sizes has made the capabilities of mobile devices ever more powerful. In addition, the demand for ubiquitous and higher speed connection to the internet has resulted in the rapid growth of wireless communication systems. Wireless communications must provide higher data rates and wider coverage with limited spectrum resources.

As one of the major innovative technologies in the physical layer of communications systems, multiple-input multiple-output (MIMO) is playing an important role in current industry practice. MIMO techniques use multiple antennas at both the transmitter and receiver sides to achieve diversity gain, multiplexing gain, or both. It has been incorporated into modern wireless communication standards such as IEEE 802.11n (WiFi), 3GPP long term evolution (LTE), Worldwide Interoperability for Microwave Access (WiMAX) and Evolved High-Speed Packet Access (HSPA+) due to its high spectral efficiency. MIMO continues to be a promising technology for future wireless communication systems, as it can be used to enhance both throughput and reliability.

MIMO detection methods may be divided into two categories, i.e., hard- and soft-detection. The basic idea of hard-detection is to directly search the feasible set for the symbol vector that is closest to the transmitted signal vector and to generate the detected

symbol vector as an output. On the other hand, a soft-detector outputs log-likelihood ratio (LLR) values for each bit in every symbol of the symbol vector.

One of the key challenges in exploiting the potential of MIMO systems is to design high-throughput, low-complexity detection algorithms while achieving near-optimal performance. In this thesis, we design and optimize algorithms for MIMO detection and investigate the associated performance and field-programmable gate array (FPGA) implementation aspects.

First, we study and optimize a detection algorithm developed by Shabany and Gulak for a K-Best based high throughput and low energy hard output MIMO detection. To this end, we expand this algorithm to the complex domain. The new method is based on the use of simple lookup tables, and it is fully scalable for a wide range of K-values and constellation sizes. This technique reduces the computational complexity, without sacrificing performance. The complexity scales only sub-linearly with the constellation size. As a result, the implementation complexity of a K-best detector can be reduced by using this method without any significant performance loss.

Secondly, we apply the bidirectional technique to trellis search and propose a high performance soft output bidirectional path preserving trellis search (PPTS) detector for MIMO systems. The comparative error analysis between single direction and bidirectional PPTS detectors is given. We demonstrate that the bidirectional PPTS detector can minimize the detection error.

Next, we design a novel bidirectional processing algorithm for soft-output MIMO systems. It combines features from several types of fixed complexity tree search procedures. The algorithm obtains a list of candidates used to calculate likelihood information in parallel and it includes two stages for tree searching, where each stage corresponds to one direction of the path selection process. The proposed approach achieves a higher performance than previously proposed algorithms while, at the same time, having a comparable computational cost. The algorithm can also make tradeoffs among different performance and complexity objectives. Moreover, its parallel nature and fixed throughput characteristics make it attractive for very large scale integration (VLSI) implementation.

Following that, we present a novel low-complexity hard output MIMO detection algorithm for LTE and WiFi applications. Our method provides a well-defined tradeoff between computational complexity and performance. The proposed algorithm is an enhancement of the zero-forcing (ZF) method. It does not require QR decomposition as a preprocessing step, as is commonly needed in sphere detection (SD) based algorithms such as K-best detection and lattice reduction. This results in significant complexity savings and provides designs over a large performance/complexity tradeoff region. Simulations have been performed for 3x3 and 4x4 MIMO configurations, with both 16- and 64-Quadrature Amplitude Modulation (QAM) constellations. The proposed algorithm uses a much smaller number of Euclidean distance (ED) calculations while attaining only a 0.5dB loss compared to maximum likelihood detection (MLD). A 3x3

MIMO system with a 16QAM detector architecture is designed, and the latency and hardware costs are estimated.

Finally, we present a stochastic computing implementation of trigonometric and hyperbolic functions which can be used for QR decomposition and other wireless communications and signal processing applications.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Along with the emergence and rapid development of modern communication technologies, people's lives have been enormously changed by several novel concepts such as streaming media, cloud storage, smart phones, and so on. The related market opportunities are very large and of increasing potential. The requirements for wireless communications systems are growing in terms of spectrum efficiency, power efficiency and error rate performance. Multiple-input multiple-output (MIMO) technology increases link capacity and spectral efficiency without adding bandwidth and improves performance by reducing the error [1-3]. As a result, it has drawn much attention in both academia and industry.

The remainder of this chapter is organized as follows. Section 1.2 provides a brief introduction to the various gains enabled by MIMO systems. Section 1.3 introduces MIMO detection concepts. It includes the MIMO system model, hard and soft detection, and a classification of MIMO detection algorithms. Difficulties in hardware design and

QR based sphere detection (SD) algorithms are discussed in section 1.4. A summary of the research contributions in this thesis is given in section 1.5, and section 1.6 provides an outline of the remaining chapters.

1.2 MIMO Gain

As its name indicates, a MIMO system uses multiple antennas at both the transmitter and the receiver to improve communication performance. MIMO technology has become the air interface for many wireless communication standards, such as IEEE 802.11n, WiMAX, 3GPP long term evolution (LTE) and many other radio technologies [4]. Compared to a traditional single input and single output (SISO) system which has only one antenna at both sides, MIMO has several important advantages. In particular, it provides a significant increase in data throughput and link range without requiring additional bandwidth or transmit power. It increases the transmission rate by introducing multiplexing gain [5-7] and its diversity gain [8-10] improves link reliability.

Multiplexing gain arises from the multiple parallel channels between the transmitter and receiver. The error rate performance in a fading channel is dominated by the worst case, which occurs in deep fading. The channel might have a significant probability of deep fading which will lead to a failure at the receiver side. The different channels between different antennas in a MIMO system have independent fading characteristics so that the reliability of the communication can be increased. Furthermore, by multiplexing the transmitted data streams among different antennas, an increase in the

overall data rate can be attained. This rate increase is proportional to the minimum of numbers of transmitter and receiver antennas, i.e. min{T_X antennas, R_X antennas}.

In a SISO system, the bit error rate (BER) performance is dominated by the worst case of fading. The larger the variance of signal to noise ratio (SNR), the higher the BER. The basic idea of diversity is to obtain multiple independent observations of the same information to lower the BER, since multiple paths are unlikely to fade simultaneously. Receive diversity in MIMO systems refers to the combination of independently faded signals from different receive antennas. It creates multiple independent faded branches so that the link reliability is improved significantly. The diversity order is a multiple of the number of transmitter antennas and the number of receiver antennas. Well-known spatial coding techniques include space-time trellis code (STTC) [11], space-time block code (STBC) [12], space-frequency block code (SFBC) [13] and space-time-frequency block code (STFBC) [14].

Also, since multiple copies of the signals are obtained at the receiver side with more than one antenna, the signals can be combined coherently to achieve gain in the SNR. Methods such as equal-gain combining (EGC) [15] and maximal ratio combing (MRC) [16] are very popular. With pre-coding, the signals will arrive at the receiving antenna coherently.

1.3 MIMO Detection Basics

1.3.1 System model and notation

Consider a spatial multiplexing $N_r \times N_t$ MIMO system with N_r receive antennas and N_t transmit antennas as shown in Figure 1.1. Let $\mathbf{s} = [s_1, s_2, ..., s_{N_t}]^T$ denote the transmitted symbol vector after the channel encoder. The equivalent baseband system can be modeled as

$$\mathbf{y}=\mathbf{H}\mathbf{s}+\boldsymbol{\omega} \tag{1.1}$$

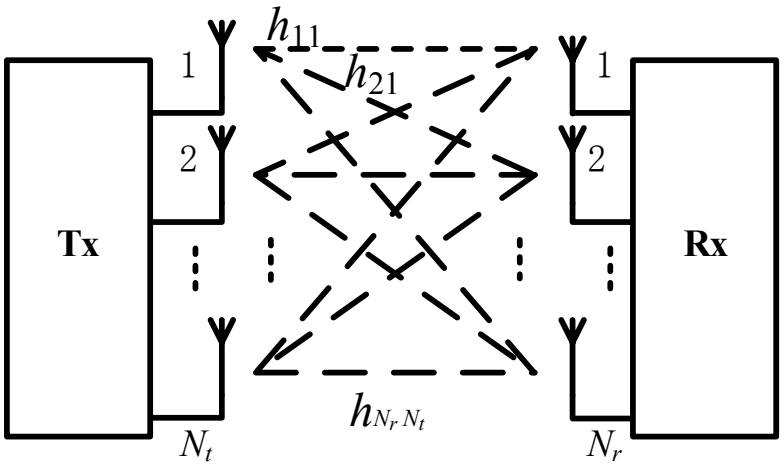


Figure 1.1 A spatial-multiplexing MIMO system model has $N_t \times N_r$ antennas

where $\mathbf{y} = [y_1, y_2, ..., y_{N_r}]^T$ and $\boldsymbol{\omega} = [\omega_1, \omega_2, ..., \omega_{N_r}]^T$ are the received symbol vector and the identically independently distributed (IID) additive white Gaussian noise (AWGN) vector with zero mean and variance $\delta_\omega^2 = N_0$, respectively.

The $N_r \times N_t$ channel information matrix $\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & \\ h_{21} & h_{22} & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_r 1} & h_{N_r 1} & \cdots & \end{bmatrix}$ is the equivalent baseband channel model between the transmitter and the receiver for each subcarrier. Here, h_{ij} (i= 1,2, … N_r, j= 1, 2, … N_t) is the channel coefficient from j-th transmit antenna to i-th receive antenna. Note that the vectors and matrices used here are all composed of complex values. Let Ω be the set of all constellation nodes. In a quadrature amplitude modulation (QAM) system with a constellation size of M, each element in Ω is represented by a $\log_2 M$-bit symbol.

The objective of the MIMO detector is to estimate the transmitted symbol vector from the received symbol vector which contains the channel fading as well as the noise interference. The optimal detection is the maximum likelihood (ML) detection, which maximizes the probability of **y** on condition of the correct symbol **s** is sent. The ML detector is as follows:

$$\mathbf{s}^{ML} = \arg\max_{\mathbf{s}\in\Omega^{N_t}} P(\mathbf{y} \mid \mathbf{s}) \tag{1.2}$$

where Ω^{Nt} denotes the set of all the possible symbol vectors from the M QAM symbols δ. Since it is an AWGN system, the ML detection rule is simplified to a minimum distance rule, which gives the best symbol error rate (SER) by finding the closest estimate $\hat{\mathbf{s}}$ given the received signal vector **y**:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}\in\Omega^{N_t}} \left\|\mathbf{y} - \mathbf{H}\mathbf{s}\right\|^2 \tag{1.3}$$

where $\|\cdot\|^2$ stands for Euclidean distance (ED). Hence, ML detection will exhaustively search for the minimum squared Euclidean distance for **y**-**Hs** from all possible symbol vector combinations with symbols from the transmitted modulation constellation. Therefore, the search complexity will significantly increase with increased QAM constellation size. ML detection is impractical for hardware implementation, especially for the large QAM constellations of interest here.

1.3.2 Hard detection and soft detection

Detection methods may be divided into two categories, i.e., hard- and soft-detection [17, 18]. The basic idea of hard-detection is to directly search the feasible set for the symbol vector that is closest to the transmitted signal vector and to generate the detected symbol vector as an output, as shown in equation (1.3).

On the other hand, a soft-detector outputs log-likelihood ratio (LLR) [19] values for each bit in every symbol of the symbol vector. Usually, the maximum a posteriori (MAP) probability is utilized. The soft output log-likelihood ratio values are then sent to a soft decoder.

The system can also be rewritten in matrix format as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N_r} \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & \cdots & \\ h_{21} & h_{22} & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_r 1} & h_{N_r 1} & \cdots & \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \\ s_{N_t} \end{bmatrix} + \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{N_r} \end{bmatrix} \tag{1.4}$$

After QAM mapping, for the j-th transmit antenna, a real bit data stream $\mathbf{x}_k=[x_{k,1}\ x_{k,2,}\ ...,\ x_{k,L}]^T$ of length L will be mapped to s_k by the QAM modulator where $L= \log_2 M$ and the b-

th bit of $\mathbf{x}_k$ is denoted as $x_{k,b,}$ $b = 0,1,\ldots, L\text{-}1$.

The LLR value for the a posteriori probability (APP) of each transmitted bit is computed by the MAX-Log MAP detector, which can be represented as:

$$\begin{aligned} \mathrm{LLR}(x_{k,b}|\mathbf{y}) &= \ln \frac{p(\mathbf{y}|x_{k,b}=+1)}{p(\mathbf{y}|x_{k,b}=-1)} \\ &\cong \frac{1}{2\sigma^2}\left(\min_{x\in x_{k,b}(-1)} \|\mathbf{y}-\mathbf{H}\cdot\mathbf{s}(x)\|^2 - \min_{x\in x_{k,b}(+1)} \|\mathbf{y}-\mathbf{H}\cdot\mathbf{s}(x)\|^2 \right) \end{aligned} \tag{1.5}$$

where $x_{k,b}^{(-1)}$ and $x_{k,b}^{(+1)}$ stands for the sets of vector symbols having bit b in symbol j equal to -1 (i.e. logical '0') and +1 (i.e. logical '1'), respectively.

A soft detector sends the soft information for each bit to indicate the probability of each bit to be equal to '1' or '0', which will give more information to the decoder. It is usually used iteratively with a decoder to improve the detection performance but at the cost of larger hardware consumption. Hard detection sends hard decisions for the symbol vector directly to the decoder and is much simpler in its hardware implementation.

1.3.3 MIMO detection classification

It can been seen from section 1.3.2 that both hard and soft detection have the objective to find the minimum distance between received the symbols and transmitted symbols by searching combinations from the signal space. According to the antenna size, the combination of symbols and bits can be very large.

The enumerative search for the minimum squared ED for **y**-**Hs** from all possible symbol vector combinations in ML and MAP detection make them too complicated and impractical for hardware implementation. Therefore, many sub-optimal methods have been developed to reduce the complexity while at the same time attaining sufficient

performance. Figure 1.2 shows a taxonomy of MIMO detection algorithms.

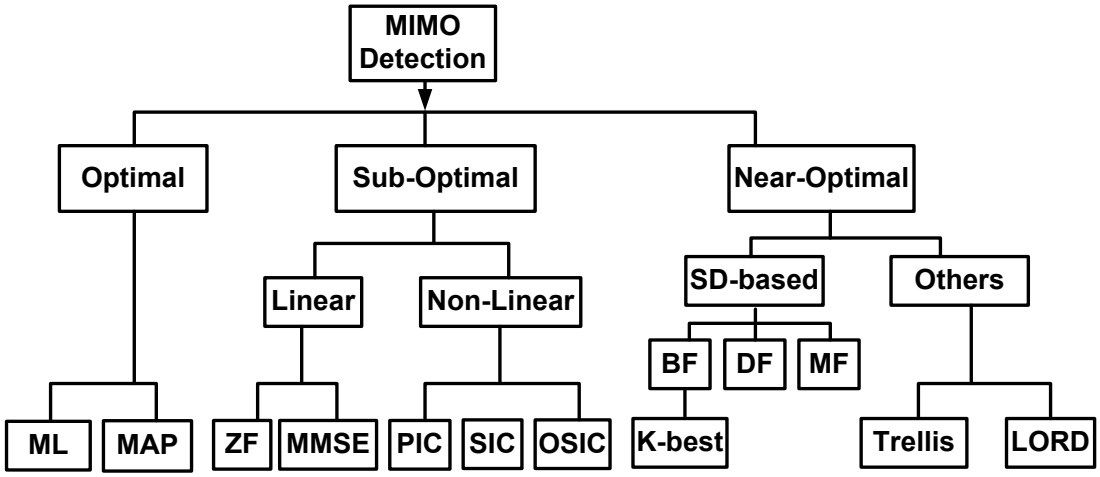


Figure 1.2 Taxonomy of MIMO Detection Algorithms

Sub-optimal detection algorithms include both linear and non-linear detections. Zero Forcing (ZF) and Minimum Mean Square Error (MMSE) are common linear detection methods. ZF has low complexity and assumes that the channel information is known at receiver side. However, it causes noise enhancement so that it will decrease the SNR. MMSE reduces the noise enhancement with respect to ZF by taking into account the noise variance, but it can still have poor performance in a fading channel. Non-linear algorithms include parallel interference cancellation (PIC) [20], which needs high SNR to obtain better performance. Other methods such as Successive Interference Cancellation (SIC) [21] and Ordered SIC (OSIC) both have the problem of error propagation.

A number of near-optimal detection algorithms have been proposed which have lower complexity and near to optimal performance as well. The most common near optimal methods include sphere detection [22-25], trellis search [26] and layered

orthogonal lattice detector (LORD) [27]. SD-based algorithms are divided into three categories: depth first (DF), metric first (MF) and breadth first (BF). MF has the disadvantage that it does not have constant throughput. BF detection methods, such as K-best, have the advantage of constant throughput and fixed complexity, making them friendly for hardware implementation.

1.3.4 Zero-forcing detection

The operation of the ZF algorithm [28] is quite simple. As a linear detector, the basic idea is to obtain the linear weighting of the received symbol vector $\mathbf{y}$ such that the interference generated from the superposition of signals from different antennas can be eliminated. According to (1.1), the linear weighting matrix **G** can be expressed as:

$$\mathbf{G} = (\mathbf{H}^{H}\mathbf{H})^{-1}\mathbf{H}^{H} \tag{1.6}$$

where $\mathbf{H}^{H}$ is the Hermitian conjugate of the matrix **H**. In some cases **H** might not be an invertible matrix or even a square matrix. Thus, the matrix **G**, which is the pseudo-inverse of **H**, is used.

At the receiver side, **G** is directly multiplied with the received symbol vector **y**, so that:

$$\mathbf{Gy} = \mathbf{G}(\mathbf{Hs}+\boldsymbol{\omega}) = \mathbf{GHs}+\mathbf{G}\boldsymbol{\omega} = \mathbf{s}+\mathbf{G}\boldsymbol{\omega} \tag{1.7}$$

It is evident that the ZF algorithm can eliminate the interference between antennas. However, it can also significantly increase the noise component. This is a weakness of ZF algorithm. Hence, it is normally not used as the sole detection method.

1.3.5 Minimum mean square error detection

The basic criterion of MMSE [29] is to minimize the error between the estimated and the transmitted vectors. Similar to ZF detection, MMSE is also a linear detection algorithm with low complexity. The weighting matrix of MMSE is:

$$\mathbf{P} = (\mathbf{H}^{\mathbf{H}}\mathbf{H} + \frac{\sigma_{\omega}^2}{\sigma_s^2}\mathbf{I})^{-1}\mathbf{H}^{\mathbf{H}} \tag{1.8}$$

where σ_{ω}^2 and σ_s^2 are the variances of the noise and the signal, respectively. MMSE takes into consideration the suppression of noise. It has superior performance compared to the ZF algorithm and has a complexity similar to that of the ZF algorithm.

1.4 Hardware Implementation Difficulties and QR Decomposition Based Sphere Detection

In general, the most computationally intensive part of MIMO detection is the computation of the ED values. Usually, near-optimal searching methods have a search space that is a finite subset of the large search set. QR decomposition based SD has drawn much attention due to its reduction of the search space [30].

The preprocessing for SD is QR decomposition. The channel matrix **H** can be decomposed as into a product of two matrixes. Specifically, **H**=**QR**, where **Q** is a unitary matrix of size $N_r \times N_t$ with $\mathbf{Q}^{\mathrm{H}}\mathbf{Q}=\mathbf{I}$ and **R** is an $N_t \times N_t$ upper triangular matrix. After multiplying by $\mathbf{Q}^{\mathrm{H}}$ on the left in equation (1.1), the equation becomes

$$\mathbf{z} = \mathbf{Q}^{\mathbf{H}}\mathbf{Q}\mathbf{R}\mathbf{s} + \mathbf{Q}^{\mathbf{H}}\mathbf{n} = \mathbf{R}\mathbf{s} + \mathbf{w} \tag{1.9}$$

where $\mathbf{z} = \mathbf{Q}^{\mathbf{H}}\mathbf{y}$.

After QR decomposition is applied, $\|\mathbf{y} - \mathbf{H}\cdot\mathbf{s}(x)\|$ in equation (1.3) can be written as $\|\mathbf{y} - \mathbf{QR}\cdot\mathbf{s}(x)\|$. Since $\left\|\mathbf{Q}^{\mathbf{H}}\mathbf{y} - \mathbf{Q}^{\mathbf{H}}\mathbf{QR}\cdot\mathbf{s}(x)\right\| = \left\|\mathbf{Q}^{\mathbf{H}}\mathbf{y} - \mathbf{R}\cdot\mathbf{s}(x)\right\|$, the equivalent problem of finding the minimum ED value in equation (1.3) becomes

$$\|\mathbf{y} - \mathbf{H}\cdot\mathbf{s}(x)\| = \|\mathbf{z} - \mathbf{R}\cdot\mathbf{s}(x)\| = \sum_{k=0}^{N_t-1} \left|(\mathbf{z})_k - (\mathbf{R}\cdot\mathbf{s})_k\right|^2 \tag{1.10}$$

Due to the upper triangular nature of the matrix **R**, equation (1.3) can be rewritten as:

$$\hat{\mathbf{s}} = \arg\min_{s\in\Omega^{N_t}} \sum_{i=N_t}^{1} \left| z_i - \sum_{j=i}^{N_t} R_{ij}s_j \right|^2 \tag{1.11}$$

Therefore, the search of the received symbol vector can be considered as a tree-search problem having N_t levels. The search proceeds from level $i = N_t$ to level $i = 1$, with level N_t+1 as a pseudo root node. At level i, the partial Euclidean distance (PED) between two successive levels is expressed as:

$$\left|e_i(s^{(i)})\right|^2 = \left| z_i - \sum_{j=i}^{N_t} R_{ij}s_j \right|^2 \tag{1.12}$$

Thus, the search for the closet $\hat{\mathbf{s}}$ in (1.3) can be calculated in an iterative manner as follows:

$$T_i(s^{(i)}) = T_{i+1}(s^{(i+1)}) + \left|e_i(s^{(i)})\right|^2 \tag{1.13}$$

and

$$e_i(s^{(i)}) = z_i - \sum_{j=i}^{N_t} R_{ij}s_j = L_i(s^{(i)}) - R_{ii}s_i \tag{1.14}$$

where

$$L_i(s^{(i)}) = z_i - \sum_{j=i+1}^{N_t} R_{ij}s_j \tag{1.15}$$

The quantity $T_i(s^{(i)})$ is the cumulative PED from level N_t to level i, where $T_{Nt+1}(s^{(Nt+1)}) = 0$. Also, $s^{(i)} = [s_i, s_{i+1}, \ldots, s_{Nt}]^T$ is the partial received symbol vector from level N_t to level i. The nodes at the last search level (i.e., level 1) of the tree are the leaf nodes. The path having the lowest cumulative PED from the root node to a leaf node is the hard-decision output of the MIMO detector

1.5 Summary of Thesis Contributions

The goal of this thesis is to design and optimize algorithms for MIMO communications. To this end, we analyze and improve two hard-output and two soft-output MIMO detection algorithms and novel low-complexity solutions are developed. In addition, we compare the performance and complexity for different antenna numbers and QAM sizes in a detailed analysis. Moreover, we describe corresponding hardware architectures for one of the low complexity algorithms and provide implementation results. After that, we introduce stochastic computing from the perspective of trigonometric and hyperbolic function implementations, which can be used in QR decomposition and MIMO detection.

The specific contributions of this thesis are summarized as follows:

Efficient Complex-Valued Enumeration for K-Best MIMO Detection Using Lookup Tables

In Chapter 2, we begin with the detection algorithm developed by Shabany and Gulak [22] for a K-Best based high throughput and low energy hard output MIMO detection. Then, we expand this algorithm to the complex domain. In this way, an efficient

complex-valued enumeration scheme for K-best MIMO detection is proposed. This novel enumeration scheme is based on using lookup tables. The proposed scheme is fully scalable for a wide range of K-values and constellation sizes. Simulation results indicate that the proposed technique achieves a close-to-optional performance, with a complexity that scales sub-linearly with the constellation size.

High Performance MIMO Detector Based on Bidirectional Path Preserving Trellis Search

In Chapter 3, we describe enhancements to a path-preserving trellis-search (PPTS) algorithm for MIMO detection which was proposed in [31] and [26]. The algorithm is optimized by using a bidirectional search method. A high performance bidirectional PPTS detector for MIMO systems is then proposed. It is shown that the performance of the L-best bidirectional PPTS is better than that of the $2L$-best single direction PPTS. Moreover, the bidirectional PPTS has much lower hardware cost than the traditional PPTS since only half of the paths are searched. The error analysis between the single direction and the bidirectional PPTS detector is given. We demonstrate that the bidirectional PPTS detector can minimize the detection error. The bidirectional PPTS requires that the QR decomposition be performed twice. However, since the channel is constant over the duration of the training sequence [32], the QR decomposition unit can be reused. Moreover, the cost of the QR decomposition unit is as low as 1% of that of the bidirectional PPTS detector [33].

A Novel Fixed-Complexity Soft-Output MIMO Detector using Parallel Bidirectional Scheme

In Chapter 4, a novel detection algorithm for soft-output MIMO systems is presented. The algorithm obtains a list of candidates used to calculate likelihood information in parallel and it includes two stages for tree searching, where each stage corresponds to one direction of the path selection process. The fixed-complexity sphere decoder and its variations such as the soft-output K-Best MIMO detector [34], the soft fixed complexity sphere decoder (SFSD) [35] and the soft-output LORD [27] can be viewed as extensions of the breadth-first tree search algorithm. The key idea of the proposed algorithm is to process the tree in a bidirectional fashion by combining several aspects of the above algorithms. The bidirectional processing leads to a parallel method with fixed-complexity and constant throughput, making it suitable for hardware implementation. Finally, the simulation results indicate that the proposed bidirectional scheme outperforms the previously proposed methods. The simulation results indicate that the algorithm achieves better performance with lower complexity than a list-based fixed-complexity soft-output sphere decoder. Moreover, due to its parallel nature, it is well suited for hardware implementation.

A Fixed and Low Complexity MIMO Detection Algorithm without QR Decomposition

Chapter 5 addresses the need for efficient hardware implementation of LTE and WiFi applications. A novel MIMO detection method is presented which does not require QR

decomposition as a preprocessing step. Moreover, a variety of techniques at the algorithmic and architectural levels are employed in order to attain a low-complexity and high-performance architecture of the optimized algorithm. The hard output MIMO detection algorithm provides a well-defined tradeoff between computational complexity and performance. The proposed algorithm is an enhancement of the ZF method. Simulations have been performed for 3x3 and 4x4 MIMO configurations, with both 16 and 64 QAM constellations. Results show that the proposed algorithm uses a much smaller number of ED calculations than competing techniques while attaining only a 0.5dB loss compared to maximum likelihood detection (MLD). Finally, an efficient semi-parallel architecture for a 3x3 MIMO system with 16QAM is designed.

Stochastic Computing Implementation of Trigonometric and Hyperbolic Functions for QR decomposition

In Chapter 6, we describe the stochastic computing implementation of trigonometric and hyperbolic functions that can be used in QR decomposition. Stochastic computing has drawn much attention in recent years due to its efficiency in hardware cost. Stochastic computing also moderates the output errors caused by bit-flips. By only using registers and multiplexers, the state machine architecture of stochastic computing can be applied for implementing various trigonometric and hyperbolic functions. We present a field-programmable gate array (FPGA) implementation of trigonometric and hyperbolic functions using the stochastic computing methodology. The results are compared to the well-known coordinate rotation digital computer (CORDIC) approach. Both designs are

synthesized and implemented on a Xilinx Virtex-5 FPGA. The results are compared in terms of delay and area for various input data widths. The results show that the proposed design method has advantages over the traditional CORDIC algorithm. Moreover, the same circuit can be used for all functions with only a change in the coefficient values. It is also more tolerant of soft errors (bit flips) than CORDIC implementations.

1.6 Thesis Outline

The remainder of this thesis is organized as follows. In Chapter 2, we expand the enumeration scheme for K-best MIMO detection this algorithm to the complex domain. The proposed technique achieves a close-to-optional performance, with a complexity that scales sub-linearly with the constellation size. In Chapter 3, the PPTS algorithm for MIMO detection proposed in [31] and [26] is further optimized by using of a bidirectional search method. We demonstrate that the performance of an L-best bidirectional PPTS is even better than the $2L$-best single direction PPTS with a much lower hardware cost since only half as many paths are searched. A novel bidirectional detection algorithm for soft-output MIMO systems is proposed in Chapter 4. It processes the tree in a bidirectional way by combining several aspects of a soft-output K-Best MIMO detector [34], the soft fixed complexity sphere decoder [35] and the soft-output layered orthogonal lattice detector [27]. Chapter 5 describes a novel detection method which does not requires QR decomposition and which only needs a small number of calculations of the Euclidean distance, without affecting the algorithm's performance. In

Chapter 6, we discuss stochastic computing implementations of trigonometric and hyperbolic functions for application to QR decomposition. Finally, the conclusions of the thesis are given in Chapter 7.

Chapter 2

Efficient Complex-Valued Enumeration for K-Best Detection Using Lookup Tables

2.1 Introduction

MIMO systems have attracted significant attention due to their high spectral efficiency. MIMO techniques have been adopted in many recent standards such as IEEE802.11n, IEEE802.11e and the LTE project. Thus, it is particularly important to design low-complexity high-performance MIMO detection schemes with near maximum-likelihood performance to satisfy the requirements of modern wireless systems.

The ML detection scheme achieves optimal performance by means of an exhaustive search. However, the complexity of this scheme grows exponentially with the number of transmit antennas and the constellation size. As a result, sub-optimal non-linear detection schemes have been proposed in the literature. Sphere decoding is a typical example of a search technique. However, the throughput of depth-first SD is variable and its performance depends on the signal-to-noise-ratio. The K-best algorithm is a breadth-first search method that guarantees a fixed throughout, independent of the

SNR. With a suitable K value, the performance of the K-best algorithm is close to that of ML detection. The standard K-best algorithm exhaustively searches all child nodes, which results in a large computational complexity. The Schnorr-Euchner (SE) enumeration technique proposed in [36] has been introduced to improve the search efficiency. A distributed K-best algorithm based on a real domain SE enumeration technique has been proposed in [22] and [37], and an SE enumeration method in the complex domain was used in [23].

Compared to the real-valued enumeration scheme, the complex-valued enumeration scheme visits a smaller number of nodes due to a reduction in the tree depth by a factor of 2, which leads to lower area, latency and complexity. The child node expansion and sorting schemes of the distributed K-best algorithm reported in [22] are efficient in the real domain and only 2K-1 nodes are visited at each level of the tree. However, the current literature lacks an efficient expansion scheme in the complex domain. The expansion method proposed in [23] visits the same number of nodes as in the real-valued scheme. In this chapter, we propose a novel and efficient complex-valued enumeration scheme that is based on using lookup tables.

2.2 Efficient Complex-Valued Enumeration for K-Best MIMO Detection Using Lookup Tables

2.2.1 Proposed complex-valued enumeration scheme

The enumeration of the K-best nodes in the complex domain may be obtained by utilizing fixed look-up tables. In order to describe this process, we first briefly review the one-dimensional SE enumeration method [24]. Consider the detection of one symbol in the symbol vector $\boldsymbol{s}$ at a given level. As shown in Figure. 2.1(a), when enumerating the K-best nodes having the smallest PEDs in the real domain for 16-QAM, the order of the constellation node coordinates having increasing distances to the point $\boldsymbol{s_i}$ is {1, -1, 3, -3}, which we label as $\{P_{(1)}, P_{(2)}, P_{(3)}, P_{(4)}\}$. $P_{(1)}$ is the first child (i.e. the best child) of the current layer from the parent node in the previous layer, while the other $P_{(i)}$ are the sibling nodes of $P_{(1)}$. $P_{(1)}$ is obtained by rounding s_i to the nearest point in the constellation, while the ordering of the other sibling nodes is found using the zigzag rule [24]:

$$P_{(n)} = P_{(1)} + (-1)^n Sb \cdot 2 \cdot fix(n/2), \quad where\, Sb = \begin{cases} 1 & if\ s_i > P_{(1)} \\ -1 & otherwise \end{cases} \tag{2.1}$$

In equation (2.1), *Sb* is a sign bit and the function *fix* returns the integer part of the input value. However, if s_i is located in a different region, the later portion of the ordering may violate the zigzag rule. As an example, for the case shown in Figure 2.1(b), equation (9) is not correct for the calculation of nodes $P_{(3)}$ and $P_{(4)}$.

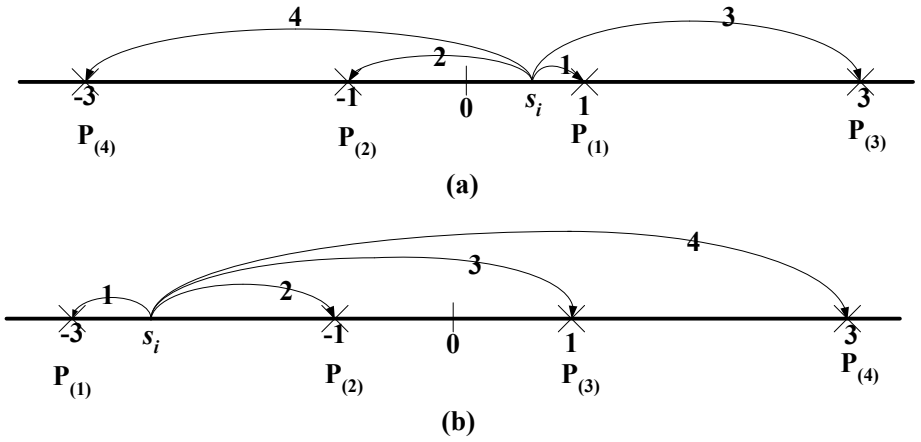


Figure 2.1 Order of the SE enumeration in the real domain for two different situations

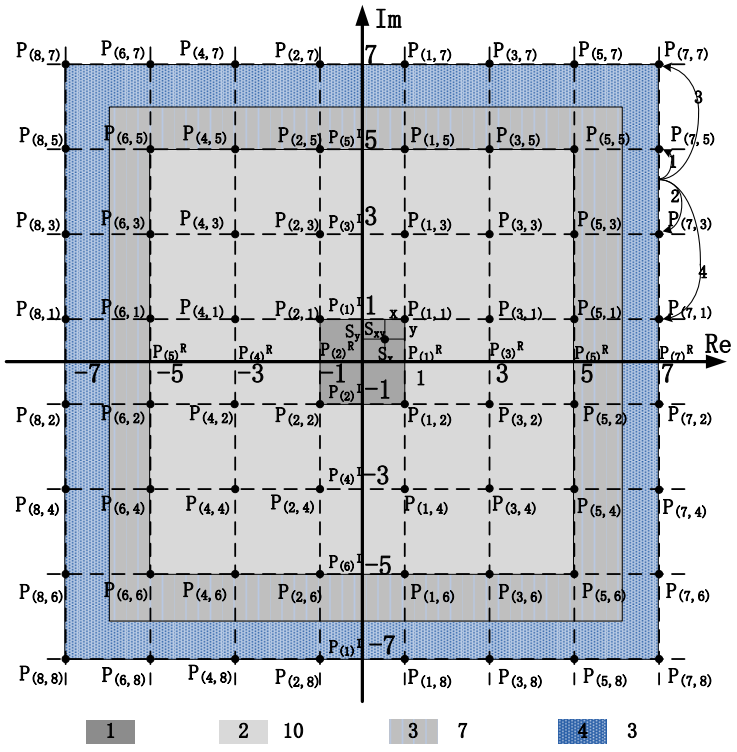


Figure 2.2 Four zones of the 64-QAM constellation, and the applicable number of table entries for zones 2, 3 and 4

Now consider the extension of the SE enumeration procedure to the complex domain. As an example, consider the case of 64-QAM. We partition the constellation into four different zones, as shown in Figure 2.2.

First assume that s_{xy} is a randomly received signal symbol located in zone 1 of Figure 2.2. Each of the 64 points in the constellation can be labeled by $P_{(re,\,im)}$, where re, im = 1, 2,…,8. The re and im indices are used to represent the increasing sequence order of SE enumeration along the real and imaginary axes, respectively. The point $P_{(1,1)}$ is called the first child node and the other $P_{(re,\,im)}$ are its sibling nodes. We extend the zigzag rule to the complex domain as follows:

$$\begin{aligned} &P_{(re,im)} = P_{(1,1)} + (-1)^{re} Sb^R \cdot 2 \cdot fix(re/2) + j(-1)^{im} Sb^I \cdot 2 \cdot fix(im/2) \\ &where\ Sb^R = \begin{cases} -1, & s_x < \mathrm{Re}\{P_{(1,1)}\} \\ 1, & s_x \geq \mathrm{Re}\{P_{(1,1)}\} \end{cases} Sb^I = \begin{cases} -1, & s_y < \mathrm{Im}\{P_{(1,1)}\} \\ 1, & s_y \geq \mathrm{Im}\{P_{(1,1)}\} \end{cases} \end{aligned} \tag{2.2}$$

In equation (1.12), the complex domain PED $|e_i(s^{(i)})|^2$ was calculated using the ℓ^2-norm. Here, however, we will use the Manhattan distance, i.e. the ℓ^1-norm. Simulations show that the difference in the BER performances from using these two norms is negligible [38].

Our goal is to find the ℓ^1-norm metric increment (LMI) order of the sibling nodes for a given point s_{xy}. Let x and y denote the distances between $P_{(1,1)}$ and s_{xy} along the real and imaginary axes, respectively. Also, let $e_i(P_{(re,\,im)})$ and $T_i(P_{(re,\,im)})$ denote the current-level partial LMI and the cumulative LMI, respectively. The cumulative LMI of a point $P_{(re,im)}$ can be expressed as :

$$T_i(P_{(re,im)}) = T_{i+1}(S) + e_i(P_{(re,im)}) \tag{2.3}$$

where $T_{i+1}(S)$ is the cumulative LMI value of its parent node. The distance increment $e_i(P_{(re, im)})$ can be calculated as:

$$e_i(P_{(re, im)}) = 2*\text{fix}(re/2) - (-1)^{(re)} * x + 2*\text{fix}(im/2)) - (-1)^{(im)} * y \quad (2.4)$$

By sorting the child node $e_i(P_{(re, im)})$ values, the 64 constellation points $P_{(re, im)}$ can be ordered with increasing $e_i(P_{(re, im)})$ for four different situations:

1) if ($x \geq y$ and $x+y<1$):

   $\{P_{(1,1)}, P_{(2,1)}, P_{(1.2)}, P_{(1,3)}, P_{(3,1)}, P_{(2,2)}, P_{(2,3)}, P_{(4,1)}, P_{(3,2)}, P_{(1,4)}, P_{(3,3)}, P_{(1,5)}, P_{(5,1)}, P_{(4,2)}, P_{(2,4)} \ldots.\}$

2) if ($x \geq y$ and $x+y \geq 1$):

   $\{P_{(1,1)}, P_{(2,1)}, P_{(1,2)}, P_{(2,2)}, P_{(1,3)}, P_{(3,1)}, P_{(2,3)}, P_{(4,1)}, P_{(3,2)}, P_{(1,4)}, P_{(4,2)}, P_{(2,4)}, P_{(3,3)}, P_{(1,5)}, P_{(5,1)} \ldots.\}$

3) if($x<y$ and $x+y<1$):

   $\{P_{(1,1)}, P_{(1,2)}, P_{(2,1)}, P_{(3,1)}, P_{(1,3)}, P_{(2,2)}, P_{(3,2)}, P_{(1,4)}, P_{(2,3)}, P_{(4,1)}, P_{(3,3)}, P_{(5,1)}, P_{(1,5)}, P_{(2,4)}, P_{(4,2)} \ldots.\}$

4) if($x<y$ and $x+y \geq 1$):

   $\{P_{(1,1)}, P_{(1,2)}, P_{(2,1)}, P_{(2,2)}, P_{(3,1)}, P_{(1,3)}, P_{(3,2)}, P_{(1,4)}, P_{(2,3)}, P_{(4,1)}, P_{(2,4)}, P_{(4,2)}, P_{(3,3)}, P_{(5,1)}, P_{(1,5)} \ldots.\}$

The above orderings of $P_{(re, im)}$ can be used as lookup tables when expanding the K-best nodes. According to [22], K = 10 is a reasonable value to use for 64-QAM, which means that only the first 10 elements of these tables are actually used.

When the first child node and the corresponding x, y values are known, the lookup tables will be used in the detection algorithm. Assume that $P_{(1)}^{R}$ is the point of the first

child node along the real axis. Let k denote the sequence number of node $P^{R}_{(1)}$ in the real axis ordering. The searching of node $P^{R}_{(k)}$ can be described as follows:

A. Turning Point

$$T_P = \sqrt{M} - abs(P^{R}_{(1)}) + \frac{1 - sign(P^{R}_{(1)})Sb^{R}}{2}$$

B. Node $P^{R}_{(k)}$

if $k <= T_p$

$$P^{R}_{(k)} = P^{R}_{(1)} + (-1)^{k} Sb^{R} \cdot 2 \cdot fix(k/2)$$

else

$$P^{R}_{(k)} = P^{R}_{(1)} - sign(P^{R}_{(1)}) \cdot \left(k - fix\left(\frac{\sqrt{M} - abs(P^{R}_{(1)})}{2} \right) - 1 \right) \cdot 2$$

end

The quantity T_p is the "Turning point." The node $P^{R}_{(k)}$ will violate the zigzag rule if k is larger than T_p. Also, the function *sign* returns the sign bit of the input value and *abs* returns the absolute value. Note that the above pseudo-code is applied in the real domain. However, by using the same operation twice, i.e. for both the real and imaginary axes, it can also be applied in the complex domain.

Note that the lookup table entries were obtained under the assumption that s_{xy} is located in zone 1 of Figure 2.2. If s_{xy} is located outside of zone 1, equations (2.2) and (2.4) cannot be used for all of the points in the constellation. However, if s_{xy} is located in

zone 2 then at least the first 4 elements of both the real and imaginary axes SE enumeration orders will obey the zigzag rule. This, in turn, means that the first 10 elements of the lookup tables are still applicable. Similarly, the first 7 elements and first 3 elements of the lookup tables are applicable for symbols located in areas 3 and area 4, respectively. Moreover, while the subsequent elements may not be in the correct order, it is still reasonable to use the table ordering in all of these cases. The reason for this is discussed in the following section.

2.2.2 Modified K-best algorithm with complex-valued enumeration

According to the distributed K-best algorithm as described in [22], after expanding the K first child nodes from each parent node, complex-valued enumeration can be applied to expand the sibling nodes until the K best nodes have been found. During this process, zero, one or more than one child node are chosen as the K-best nodes from the same parent at the current level. This means that, for the case of complex-valued enumeration, the number of elements used in the lookup tables is variable. Therefore, the modified K-best algorithm can be described as follows:

Step 1) Set one path at level N_t+1 with LMI=0;

Step 2) Find the first child node at level N_t and calculate its LMI.

Select it as the surviving node of level N_t;

Step 3) **for** l=2: K

a. Perform the complex-valued enumeration on the current node and obtain the next best sibling node;

b. Calculate the corresponding LMI, select it as the surviving node of

level N_t;

end

Step 4) **for** $i = N_t$ -1: -1: 2

Find all the first child nodes and their LMIs of all surviving nodes, and save them in a new set ς.

for k=1: K

a. Select the child node in ς with the lowest LMI and select it as the surviving node of level i;

b. Perform the complex-valued enumeration on the current node; obtain the next best sibling node;

c. Calculate the corresponding LMI, and save them in ς;

end

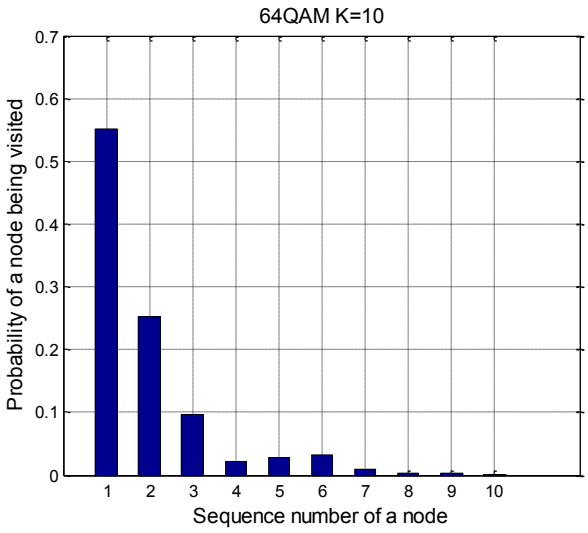
end

Step 5) Find the first child of all surviving nodes in the last level.

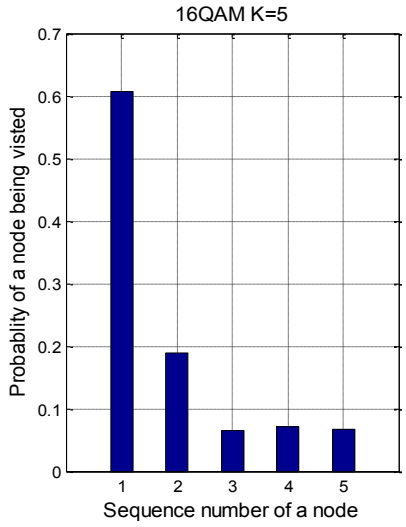
The one with the lowest LMI is the hard-decision output of the detector.

As mentioned earlier, not all of the elements in the lookup tables will be used in the complex-valued enumeration. Figure. 2.3 shows the probability of usage of the lookup table elements for 64-QAM and 16-QAM. The horizontal axis is the sequence number of the elements in the lookup tables, while the vertical axis is the probability of visiting that element during the enumeration process. Figure. 2.3(a) indicates that the first three elements in the tables are used nearly 90% of the time. A similar phenomenon is

observed in the case of 16-QAM, as shown in Figure. 2.3(b). Thus, even though the ordering of the later elements in the tables is not exact, it will not have a significant impact on the detection performance since those elements are only rarely visited. Therefore, it is reasonable to use the above lookup tables for our proposed scheme no matter where the estimated node s_{xy} is located.



(a) 64-QAM



(b) 16-QAM

Figure 2.3 Simulation results for the distribution of child nodes

2.2.3 Simulation results and complexity comparisons

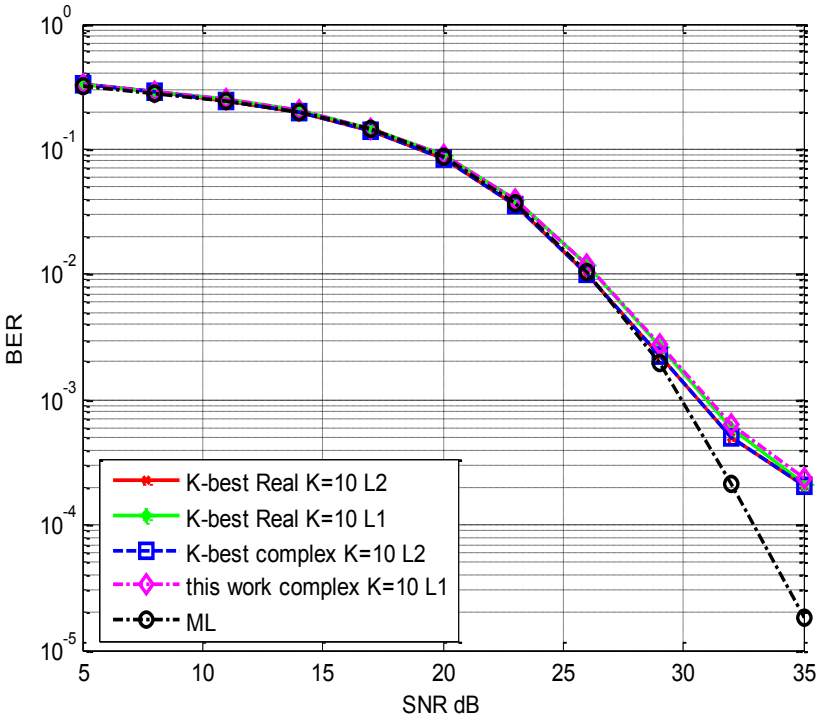
In this section, we first compare the complexity of the proposed K-best algorithm with other K-best schemes. Then, our simulation results are shown.

Table 2.1 Complexity comparisons of K-best algorithms

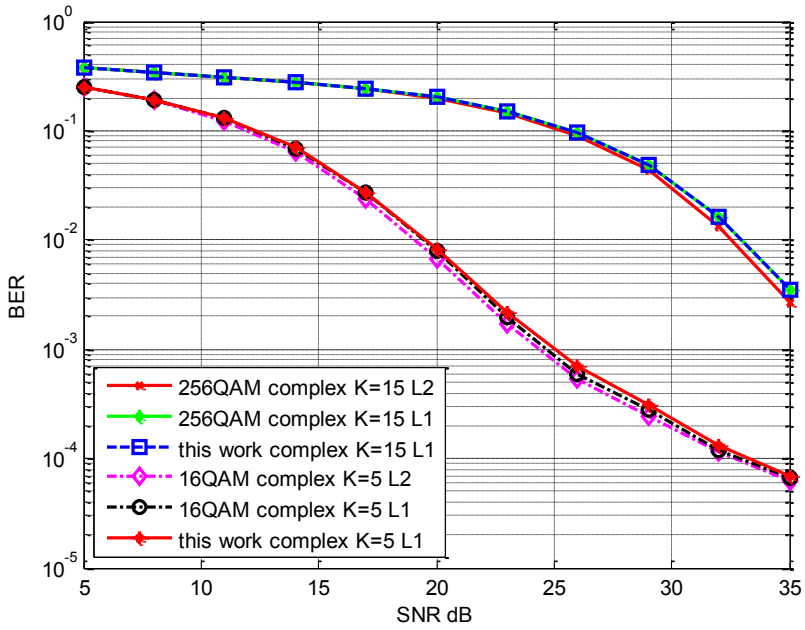
	Operation	[24] (real)	[25] (real)	[22] (real)	[23] (complex)	This work (complex)
4×4 16-QAM K=5	expand	140	140	72	44	32
	multiply	273	696	287	304	232
	add	413	712	346	379	287
	compare	560	140	139	81	47
4×4 64-QAM K=10	expand	552	552	152	84	67
	multiply	818	2776	601	590	482
	add	1370	3320	725	735	597
	compare	4610	560	461	240	150
4×4 256-QAM K=15	expand	1696	1696	232	140	102
	multiply	2123	8448	919	966	732
	add	3819	8704	1108	1206	907
	compare	21210	1680	1184	579	367

Table 2.1 lists the complexity comparisons for various approaches. Note that these comparisons only include the detection portion, i.e. excluding the QR-decomposition and channel preprocessing since these operations are common to all these schemes. The number of visited child nodes is referred as “expand” in the table. Multiply operations, add operations and the required number of compare operations in sorting are given for the detection of one transmit symbol vector. The value listed in the “expand” row refers to the number of PED or LMI values that must be calculated. For the same value of K, the BER performance of our proposed approach is the same as that of other

approaches using the ℓ^1-norm. The complexity differences arise from the manner in which the K-best surviving nodes are found. From Table 2.1, it can be seen that the complex domain algorithms have a significant complexity savings compared to that of the real domain. This is due to the fact that the tree depth is halved and the same value of K results in the same performance. Thus, the proposed complex-valued enumeration scheme has a significant complexity reduction compared to the enumeration scheme of [23].



(a) 64-QAM



(b) 16-QAM and 256-QAM

Figure 2.4 BER performance for various constellation sizes

Simulation results for the bit error rate are shown in Figure. 2.4. In the figure, L2 indicates the ℓ^2-norm, while the ℓ^1-norm is labeled as L1. Figure. 2.4(a) compares the BER performance for 64-QAM for the ℓ^2-norm and ℓ^1-norm in both real and complex domains. Over a wide range of interest (SNR in the range from 5-30 dB), the performance loss with our proposed scheme is small compared to the ML method. Furthermore, the difference in the BER performance between the ℓ^2-norm and the ℓ^1-norm is negligible. The curves labeled by L2 depict the performance of the original K-best algorithm, and the other curves display the performance of the modified algorithm proposed in this chapter. It can be seen that the performance curves in the complex domain are nearly overlapped with those in the real domain for both the ℓ^2-norm and the

ℓ^1-norm, which indicates that the proposed modified K-best algorithm does not have any significant performance loss compared to the previously proposed algorithms. Also, the performance curves for 16-QAM and 256-QAM are plotted in Figure. 2.4(b), which shows similar results. This demonstrates that the proposed complex-valued enumeration scheme can be used for a wide range of constellation sizes.

2.3 Conclusions

An efficient and flexible complex-valued enumeration scheme for K-best MIMO detection has been proposed. The new method is based on the use of simple lookup tables, and it is fully scalable for a wide range of K-values and constellation sizes. Simulation results indicate that the proposed technique achieves a close-to-optional performance, with a complexity that scales sub-linearly with the constellation size. As a result, the implementation complexity of a K-best detector can be reduced by using this method without any significant performance loss.

Chapter 3

Bidirectional Path Preserving Trellis Search

In this chapter we propose a high performance bidirectional path preserving trellis search detector for MIMO systems. The error analyses for single direction and bidirectional PPTS detectors are given. We show that the bidirectional PPTS detector can minimize the detection error. The proposed detector has a 0.1dB Frame Error Rate (FER) gain compared with traditional PPTS detectors.

3.1 Introduction

MIMO systems have been adopted in many wireless standards such as 3GPP LTE, IEEE 802.16e, and IEEE 802.11 due to its high spectral efficiency [5]. Many research efforts focus on the high throughput and low cost MIMO detector design. The sphere detection algorithm proposed in [39-41] is a depth-first algorithm which finds the nearest lattice point. However the depth-first algorithm has limited throughput when the SNR is low. The *K*-best algorithm based on the breadth-first tree-search algorithm is a fixed-complexity algorithm which requires a high sorting complexity to find the best candidates [24-25, 42].

In [31] and [26], a path-preserving trellis-search algorithm for MIMO detection was proposed. Different from the tree-search algorithm, the PPTS detector searches the MIMO signal with an unconstrained trellis. It maps each node of the trellis in a stage k to a possible complex-valued symbol from antenna k. Then, the soft-output MIMO detection problem is converted into a multiple shortest paths problem. A path reduction method is applied to reduce the algorithm complexity. The path extension method is also used to guarantee that every node in stage k is included in the selected path. The complex search algorithm in a traditional MIMO detector becomes a simple individual node-by-node search problem in a PPTS detector.

The PPTS algorithm can be further optimized by using a bidirectional search method. The searching process of PPTS operates from antenna Txk to the Tx1, which maps on the trellis from source node to sink node. However, if noise affects the first stage of the trellis graph search signal, the result will have large errors. In this chapter, we demonstrate that the performance of L-best bidirectional PPTS is better than the 2L-best single direction PPTS. Moreover, the bidirectional PPTS has lower hardware cost than traditional PPTS since only half as many paths are searched. The bidirectional PPTS requires that the QR decomposition be performed twice. However, since the channel is constant over the duration of the training sequence [32], the QR decomposition unit can be reused.

The rest of this chapter is organized as follows: Section 3.2 gives a brief PPTS algorithm review. The bidirectional PPTS is proposed in Section 3.3, and section 3.4 gives the conclusions.

3.2 PPTS Algorithm Review

The main idea of PPTS is computing the Euclidean distance recursively by using an unconstrained trellis. The stage k symbol s_k is transmitted by antenna k. The nodes are also ordered into N_t vertical stages. The search process begins from the root and ends with the virtual sink node. Each node in the trellis graph is mapped to a constellation point which belongs to a known alphabet. Each path from root to sink represents a transmitted symbol vector.

The weight $e_{k-1}(\mathbf{q}^{(k-1)})$ between nodes $\langle k,q\rangle$ and $\langle k-1,q'\rangle$ is given by

$$e_{k-1}\left(\mathrm{q}^{(k-1)}\right)=\left|\hat{y}_{k-1}-\sum_{j=k-1}^{N_R-1}R_{k-1,j}\cdot s_j\right|^2 \tag{3.1}$$

where $\mathbf{q}^{(k-1)}$ is the partial symbol vector $[q_{k-1}\ q_k\cdots$, and s_j is the complex-valued symbol mapped by q_j (q=0,1,…2^M-1). The sum of edge weights along the search path is obtained by a partial path metric d_k, which can be computed recursively as

$$d_{k-1}(q')=d_k(q)+e_{k-1}\left(\mathbf{q}^{(k-1)}\right) \tag{3.2}$$

The path weight can be obtained at the sink node with $d_0(\cdot)$. Then, the soft MIMO detection problem is transformed into a multiple shortest paths problem. As stated in [31], "For each node $\langle k,q\rangle$ in the trellis graph, find a shortest path from root to sink that must visit this node $\langle k,q\rangle$." To reduce the search complexity, a path reduction method is proposed in which each node evaluates all its incoming paths and only preserves a predefined number (L) of them [31]. However, it is difficult to guarantee every node in the trellis has a shortest path through the node after the path reduction process. Hence, a

path extension processing is required which fills in missing paths for each trellis node, in which every node in each stage is included in a path from root to sink. According to the simulation result, the L=2 PPTS MIMO detector shows only 0.3dB performance degradation compared to the full search method.

As shown in [26], each node in stage 2 has two outgoing paths after path extension due to the path extension processing. When the L shortest paths of each node are selected, the LLR can be computed as:

$$\mathrm{LLR}(x_{k,b}) = \frac{1}{2\sigma^2}\left(\min_{i:b=-1}\gamma^{(L)}(k,i,t) - \min_{i:b=+1}\gamma^{(L)}(k,i,t)\right) \tag{3.3}$$

where $\gamma^{(L)}(k,i,t)$ are the selected L shortest paths.

3.3 High Performance MIMO Detector Based on Bidirectional Path Preserving Trellis Search

3.3.1 Error analysis

The PPTS operates from the root trellis stage to the sink stage in the trellis graph as mentioned in the previous section. We find that the search path in the beginning stage largely determines the accuracy of the search result. In other words, if the received signal is corrupted by noise which causes the PPTS detector to select a wrong path in the first stage, the error between the search result and correct value is large. We provide a search error analysis for the PPTS algorithm to support this point.

In the full search Max-Log MAP (MLM) detection algorithm, the minimal Euclidian distance min(d(**s**)) between the received signal and **R**·**s** is required, where

$$d(\mathbf{s}) = \sum_{k=0}^{N_t-1} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s})_k \right|^2 \tag{3.4}$$

When symbol vector $\mathbf{s}$ equals to $\mathbf{s}^{\min}$, the minimal value of $d(\mathbf{s})$ is obtained as

$$\min(d(\mathbf{s})) = d(\mathbf{s}^{\min}) = \sum_{k=0}^{N_t-1} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s}^{\min})_k \right|^2 \tag{3.5}$$

The full search MLM detector has the best performance but with a search complexity of $Q^{N_t}/2$. That is to say, the full search MLM can be used as a benchmark for our proposed MIMO detection method. We define a variable δ to measure the error magnitude between full search MLM and the PPTS detector:

$$\delta(\mathbf{s}_i) = \left| d(\mathbf{s}^{\min}) - d(\mathbf{s}'_i) \right| \quad i \in 0,1,\cdots \tag{3.6}$$

Suppose that, in $\mathbf{s}'_i$, the i-th signal is different from the minimal search symbol vector $\mathbf{s}^{\min}$. This means that the search error happened at $\mathbf{s}_i$ with bias value $\delta(\mathbf{s}_i)$. The search process starts from stage N_t -1 to stage 0. Since $\mathbf{R}$ is an $N_t \times N_t$ upper triangular matrix, $\mathbf{R}(k,i)\cdot\mathbf{s}_k = 0\ k = 0,1,2,\ \cdots$. The bias value $\delta(\mathbf{s}_i)$ can be rewritten as

$$\delta(\mathbf{s}_i) = \left| \sum_{k=0}^{i} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s}^{\min})_k \right|^2 - \sum_{k=0}^{i} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s})_k \right|^2 \right| \tag{3.7}$$

If the searching error happened at the j-th stage with $\mathbf{s}'_j$ where $j > i$ when $k \le i$ then $\delta(\mathbf{s}_j) < \delta(\mathbf{s}_i)$, since

$$\begin{aligned} \delta(\mathbf{s}_i) - \delta(\mathbf{s}_j) &= \sum_{k=0}^{j} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s}'_j)_k \right|^2 - \sum_{k=0}^{i} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s}'_i)_k \right|^2 \\ &\ge \sum_{k=i+1}^{j} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}\cdot\mathbf{s}'_i)_k \right|^2 > 0 \end{aligned} \tag{3.8}$$

That is, if the search error occurs at the *j*-th stage it will be transferred and amplified to the *i*-th stage on the search path. Thus, the first search stage will dominate the search error at the last search stage 0 as

$$\delta(\mathbf{s}'_j) << \delta(\mathbf{s}'_i) \quad (j = N_t - 1, i = 0) \tag{3.9}$$

Fortunately, the order of search stages can be changed by rearranging the channel matrix. The first search stage will become the last search stage. If we combine the search results from two directions, the search error can be reduced significantly.

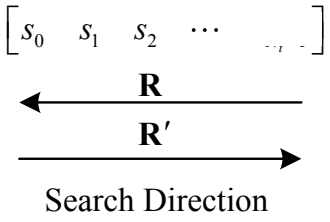
3.3.2 Bidirectional PPTS

Now, we rearrange the channel matrix as $\mathbf{H}'=\mathbf{H}(:,N_t\text{-}1{:}0)$. Then, perform the QR decomposition again as $\mathbf{H}' = \mathbf{Q}'\mathbf{R}'$. According to equation (3.9), the search direction of $\mathbf{H}' = \mathbf{Q}'\mathbf{R}'$ (referred to backward search) is the reverse direction of $\mathbf{H} = \mathbf{QR}$ (referred to forward search), as shown in Figure 3.1.

$$\begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_{N_t-1} \end{bmatrix} = \begin{bmatrix} R_{0,0} & R_{0,1} & R_{0,2} & \cdots & \\ 0 & R_{1,1} & R_{1,2} & \cdots & \\ 0 & 0 & R_{2,2} & \cdots & \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & R_{N_t-1,N_r-1} \end{bmatrix} \cdot \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ \cdot \\ s_{N_t-1} \end{bmatrix} \quad \text{Search Direction}$$

$$\begin{bmatrix} \hat{y}_{N_t-1} \\ \vdots \\ \hat{y}_2 \\ \hat{y}_1 \\ \hat{y}_0 \end{bmatrix} = \begin{bmatrix} R'_{0,0} & R'_{0,1} & R'_{0,2} & \cdots & \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & R'_{N_t-3,2} & \cdots & \\ 0 & 0 & 0 & \cdots & \\ 0 & 0 & 0 & \cdots & \end{bmatrix} \cdot \begin{bmatrix} s_{N_t-1} \\ \vdots \\ s_2 \\ s_1 \\ s_0 \end{bmatrix} \quad \text{Search Direction} \qquad \text{A-1}$$

(a)



(b)

Figure 3.1 Search direction of the bidirectional MIMO detector

We define the minimal Euclidian distance in the forward search as $d(\mathbf{s}^{\mathrm{F}})$ and for the backward search as $d(\mathbf{s}^{\mathrm{B}})$. Then, we select the smaller value between $d(\mathbf{s}^{\mathrm{F}})$ and $d(\mathbf{s}^{\mathrm{B}})$ after obtaining both the forward and backward search minimal Euclidian distances as

$$d_{\mathrm{opt}}(\mathbf{s}) = \min(d(\mathbf{s}^{\mathrm{F}}), d(\mathbf{s}^{\mathrm{B}})) \tag{3.10}$$

Then, the LLR of bidirectional PPTS is given by

$$\mathrm{LLR}(s_{k,b}) \approx \frac{1}{2\sigma^2}\left(\min_{\mathrm{x}:s_{k,b}=-1} d_{\mathrm{opt}}(\mathbf{x}) - \min_{\mathrm{x}:s_{k,b}=+1} d_{\mathrm{opt}}(\mathbf{x}) \right) \tag{3.11}$$

In the following equations, it is shown that the bidirectional search result $d_{\mathrm{opt}}(\mathbf{s})$ has a smaller search bias value than the single direction search.

The bias value of backward search is given by

$$\delta(\mathbf{s}_i^{\mathrm{B}}) = \left| \sum_{k=0}^{N_t-i+1} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}' \cdot \mathbf{s}_{\min})_k \right|^2 - \sum_{k=0}^{N_t-i+1} \left| (\hat{\mathbf{y}})_k - (\mathbf{R}' \cdot \mathbf{s}_i^{\mathrm{B}})_k \right|^2 \right| \tag{3.12}$$

Then, the relationship between $\delta(\mathbf{s}_i^{\mathrm{F}})$ and $\delta(\mathbf{s}_i^{\mathrm{B}})$ is given by

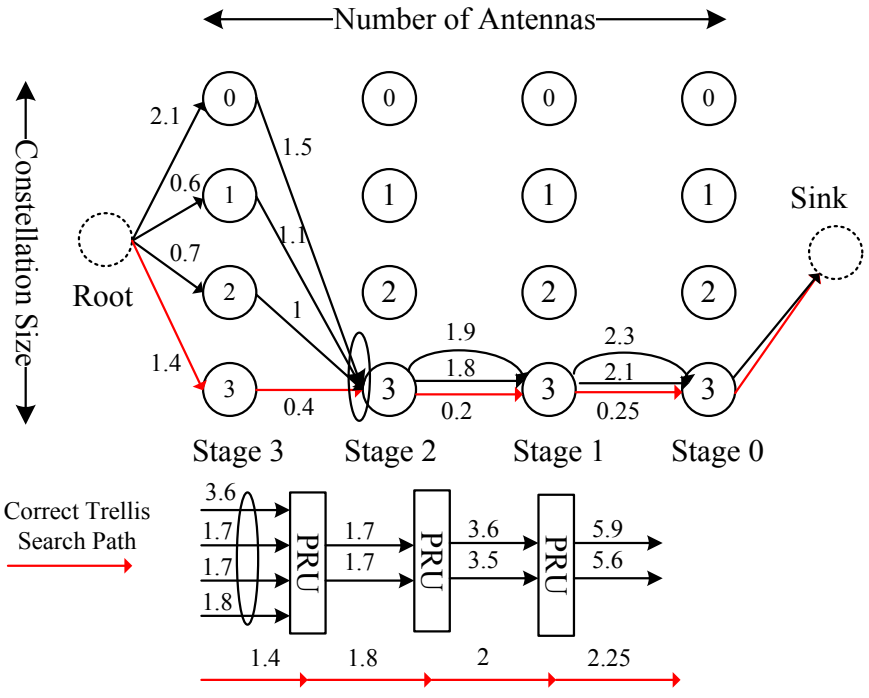
$$\begin{aligned} \delta(\mathbf{s}_i^{\mathrm{F}}) &\geq \delta(\mathbf{s}_i^{\mathrm{B}}) \quad i \geq N_t/2 \\ \delta(\mathbf{s}_i^{\mathrm{F}}) &\leq \delta(\mathbf{s}_i^{\mathrm{B}}) \quad i \leq N_t/2 \end{aligned} \tag{3.13}$$

The bias of the bidirectional search result is the minimal one between the forward

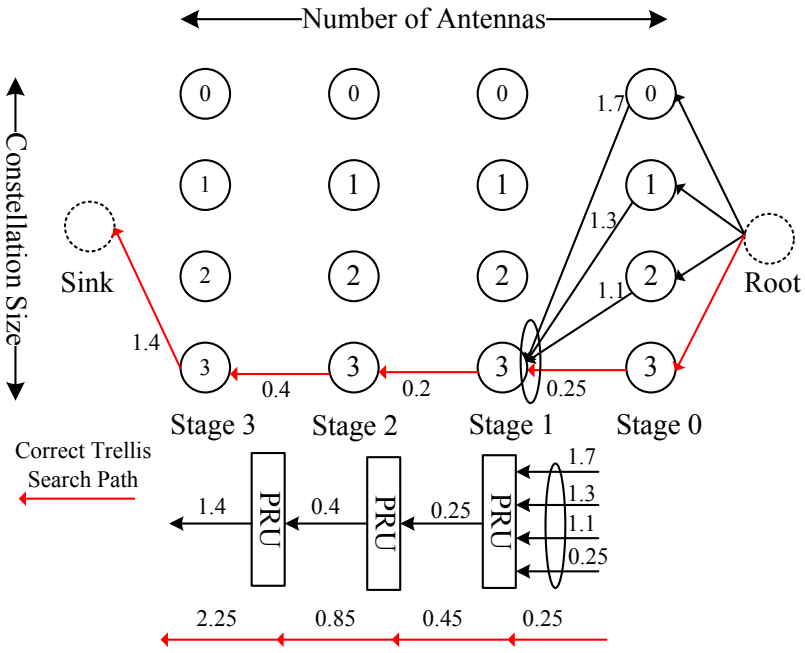
and backward search results:

$$\delta_{\text{opt}}(\mathbf{s}_i) = \min\left(\delta(\mathbf{s}_i^{\text{F}}), \delta(\mathbf{s}_i^{\text{B}})\right) \tag{3.14}$$

To illustrate the principle of the bidirectional PPTS search method, we give an example in Figure 3.2 which has failed to select the correct path at L=2 (L is the number of reserved paths in each stage) of the forward search but succeeds at the L=1 backward search. The red line is the correct transition path. As shown in Figure 3.2(a), at stage 2 after the L=2 PRU, the paths 1→3 and 2→3 are preserved but the correct transition path 3→3 is discarded. Hence, the error will be broadcast to the rest of the trellis stages, which leads to a large bias for node 3 in stage 0. However, the correct search path is selected after the backward L=1 search process, as shown in Figure 3.2(b). The Euclidian distance value of stage 0 node 3 will be replaced by the backward search result.



(a) Forward L=2 Search Trellis Path



(b) Backward L=1 Search Trellis Path

Figure 3.2 Bidirectional 4×4 QPSK MIMO detection method. (a) shows the forward L=2 search trellis path with error path selected. (b) shows the backward L=1 search trellis path with correct path selected

Table 3.1 Error simulation results based on 4 ×4 16-QAM MIMO with 10K transmitted symbols

	Single PPTS			Bidirectional PPTS	
	Bias δ			Bias δ	
	mean	variance		mean	variance
QPSK L=2	1.154	4.091	QPSK L=1	0.645	0.668
16QAM L=2	1.19	7.69	16QAM L=1	0.76	2.01
16QAM L=4	0.457	1.609	16QAM L=2	0.258	0.261
64QAM L=4	1.09	5.3767	64QAM L=2	0.477	0.580

Thus, the bidirectional search has better performance than the single direction search method. To verify this, we simulate the bias of PPTS based on a 4×4 16-QAM MIMO system with 10,000 transmitted symbols. The smaller mean and variance values in Table 3.1 represent a more accurate search result. According to the simulation results, the L based bidirectional search PPTS performs better than the $2L$ based single direction search PPTS system.

The FER performance comparison between the single direction and bidirectional search MIMO systems based on the MLM algorithm is shown in Figure 3.3. The channel matrices are assumed to have independent Gaussian distributions. The AWGN channel model is used in the simulation work. From the FER simulation, it is observed that the L based bidirectional search PPTS performs better than the 2L single direction search PPTS by about 0.1dB. That is to say, the numbers of reserved paths L in traditional PPTS can be reduced by half in the bidirectional search PPTS with a better performance. Also, the L=2 bidirectional PPTS method has nearly the same FER as the full search MLM detector.

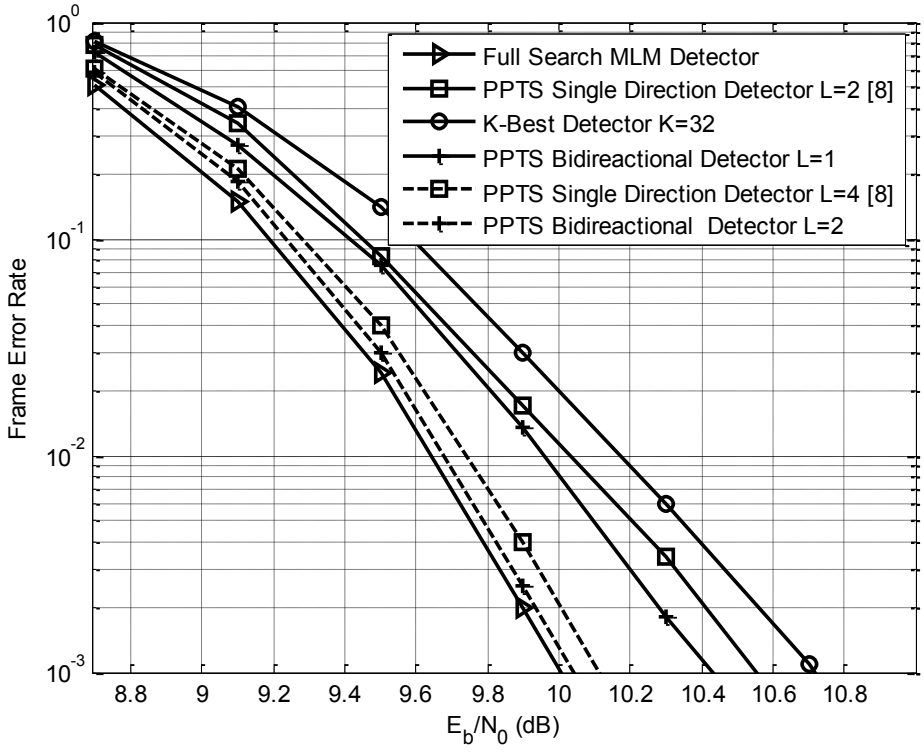


Figure 3.3 FER performance of a coded 4×4 16-QAM MIMO system using the proposed algorithm with sorted QRD. The outer channel code is a WiMax LDPC code with rate 1/2 and length 1296 MPA decoder

3.4 Conclusion

In this chapter, we have proposed a high performance PPTS MIMO detector based on a bidirectional search method, which has a 0.1dB performance gain compared to the single direction search architecture. The proposed bidirectional PPTS can be used in an iterative MIMO detection system to improve the performance even further.

Chapter 4

Fixed-Complexity Soft-Output MIMO Detector Using Parallel Bidirectional Scheme

This chapter presents a novel detection algorithm for soft-output, MIMO systems. The algorithm obtains a list of candidates used to calculate likelihood information in parallel and it includes two stages for tree searching, where each stage corresponds to one direction of the path selection process. The simulation results indicate that the algorithm achieves better performance with lower complexity than a list-based fixed-complexity soft-output sphere decoder. Moreover, due to its parallel nature, it is well suited for hardware implementation.

4.1 Introduction

In recent years, MIMO technology has been put into practice in several standards such as IEEE 802.11n and WiMax. Furthermore, soft-output MIMO signal detection is usually employed since most practical wireless systems use error correcting codes such as turbo

codes or low-density parity check (LDPC) codes which require soft information to perform the decoding [43]. One of the most challenging design aspects is to obtain high performance MIMO signal detection with reasonable complexity in order to make the approach feasible for hardware implementation.

The detection procedure can be transformed into a tree-search problem and the soft information can be efficiently computed using a sphere decoding algorithm. Among various kinds of soft-output MIMO detectors, the ones having fixed complexity are especially attractive due to their constant throughput, independent of the channel conditions and the noise level. The fixed-complexity sphere decoder and its variations such as the soft-output K-Best MIMO detector [34], the soft fixed complexity sphere decoder [35] and the soft-output layered orthogonal lattice detector [27] can be viewed as extensions of the breadth-first tree search algorithm.

However, the soft-output K-Best detector may not achieve superior performance because it is simply an extension of the hard-output detector, which means the LLR values obtained may not be accurate enough. The LORD algorithm is capable of achieving good performance, but it requires many additional matrix permutations, which leads to a heavy computational load. The SFSD, which is based on the list fixed-complexity sphere decoder (LFSD) [44], creates a new subset in order to obtain an accurate approximation of the LLR values, resulting in better performance than LFSD. The key idea of the new algorithm proposed in this chapter is to process the tree in a bidirectional fashion by combining several aspects of the above algorithms. The bidirectional processing leads to a parallel method with fixed-complexity and constant

throughput, making it suitable for hardware implementation. Finally, the simulation results indicate that the proposed bidirectional scheme outperforms the previously proposed methods.

4.2 Complex to Real

Consider a MIMO system with N_t transmit and N_r receive antennas. The coded bit-stream is mapped to N_t - dimensional transmit symbol vectors $\mathbf{s} \in \Omega^{N_t}$ with an M-QAM constellation. The complex-valued system can be transformed to an equivalent real-valued model based on real-value decomposition (RVD) as follows:

$$\begin{pmatrix} \mathrm{Re}(\mathbf{y}_1) \\ \mathrm{Im}(\mathbf{y}_1) \\ \vdots \\ \mathrm{Re}(\mathbf{y}_{N_r}) \\ \mathrm{Im}(\mathbf{y}_{N_r}) \end{pmatrix} \begin{pmatrix} \mathrm{Re}(\mathbf{H}_{1,1}) & -\mathrm{Im}(\mathbf{H}_{1,1}) & \cdots & \mathbf{H}_{1,N_t}) & -\mathrm{Im}(\mathbf{H}_{1,N_t}) \\ \mathrm{Im}(\mathbf{H}_{1,1}) & \mathrm{Re}(\mathbf{H}_{1,1}) & \cdots & \mathbf{H}_{1,N_t}) & \mathrm{Re}(\mathbf{H}_{1,N_t}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathrm{Re}(\mathbf{H}_{N_r,1}) & -\mathrm{Im}(\mathbf{H}_{N_r,1}) & \cdots & {}_{N_r,N_t}) & -\mathrm{Im}(\mathbf{H}_{N_r,N_t}) \\ \mathrm{Im}(\mathbf{H}_{N_r,1}) & \mathrm{Re}(\mathbf{H}_{N_r,1}) & \cdots & {}_{N_r,N_t}) & \mathrm{Re}(\mathbf{H}_{N_r,N_t}) \end{pmatrix} \cdot \begin{pmatrix} \mathrm{Re}(\mathbf{s}_1) \\ \mathrm{Im}(\mathbf{s}_1) \\ \vdots \\ \mathrm{Re}(\mathbf{s}_{N_t}) \\ \mathrm{Im}(\mathbf{s}_{N_t}) \end{pmatrix} \begin{pmatrix} \mathrm{Re}(\mathbf{n}_1) \\ \mathrm{Im}(\mathbf{n}_1) \\ \vdots \\ \mathrm{Re}(\mathbf{n}_{N_t}) \\ \mathrm{Im}(\mathbf{n}_{N_t}) \end{pmatrix}$$

In this case, the number of layers of the tree search will be doubled.

4.3 Proposed Algorithm

4.3.1 The motivation of the algorithm

This work is motivated by several previously proposed fixed-complexity detectors including SFSD [35], LORD [27] and the Bidirectional Partial Tree Search technique [43]. The SFSD provides a simple way to generate a candidate list by identifying the distribution of expanding nodes at each layer. It can be seen that full expansion followed

by single expansion brings certain implementation advantages, e.g. low complexity, parallelism and pipelining. The LORD algorithm searches the tree using different layer orderings in such a way that each layer becomes the reference once [27]. It achieves more reliable diversity in bit values for the LLR calculation compared to similar methods without multiple permutations and it guarantees that a counter-hypothesis can be found. The basic idea of the proposed bidirectional technique is to perform a partial tree search in both the forward and backward directions. During the search process in each direction, only half of the list is generated. This means that only half of the levels of the tree need to be searched in one direction, which results a lower complexity compared to a full tree search. The aforementioned aspects will be utilized to advantage in the proposed algorithm.

4.3.2 The bidirectional candidates adding algorithm

The pseudo code for the proposed algorithm is shown in Figure 4.1. The notation $\hat{p}$ refers to the node having the minimum PED in the current i-th layer and $q = \log_2 M$, which is the number of the bits each node contains. The detection operates in two directions. In the forward tree search stage, there is a full expansion at the root node of the tree, while half of the layers in the tree use the same candidate adding method [45-46] as in SFSD. First, the node having the minimum PED is selected. Then, its sibling nodes are added into ς via bit flipping. After the forward stage, the m best nodes in the last layer are kept in ς to become the top layer nodes in the backward stage, which processes the node expanding and candidate adding in an identical way. By the end of the procedure, every bit's LLR value will have been estimated.

1: **Forward:**
2: ς = all child nodes of root
3: for $i = 2$ to N_t do
4: $\varsigma = \cup_{node \in \varsigma}$ {the best children of node}
5: if $i \leq N_t / 2$ then
6: for $j = (i - 1) \cdot q + 1$ to $i \cdot q$ do
7: Flip jth of $\hat{p}$ and add to ς
8: end for
9: end if
10: end for
11: Permute **H**, sort and keep m nodes in ς
12: **Backward:**
13: $\varsigma = \varsigma \cup \{ q$ flipped nodes$\}$
14: Repeat step 3 to 10

Figure 4.1 Pseudo code of the bidirectional algorithm

The bidirectional technique leads to simplification since a zero-forcing (ZF) solution is used in half of the levels. In addition, the list used for LLR computation has better diversity for $\chi'^{(-1)}_{j,b}$ and $\chi'^{(+1)}_{j,b}$, which also ensures that there is at least one counter-hypothesis for each bit. In addition, the layer ordering technique in [44] is adopted to improve the performance. The layer with largest noise amplification is detected first in the forward stage, and the rest of the layers use a strongest-first signal detection order.

As an example, Figure 4.2 shows the node expansion of the tree search for the case of 4QAM modulation. It uses m = 1 in a 4×4 MIMO configuration. The horizontal solid line stands for Zero-Forcing computing, and the nodes with vertical solid line are $\hat{p}$ with minimum PED. The dashed line indicates the new added partial solution based on flipping $\hat{p}$'s bit. In this 4QAM case, there's two added nodes for each $\hat{p}$. The bit flipping happens at the layer of antenna 2 in the forward stage, and antenna 3, 4 in backward stage. The dash-dot line indicates the sorting and pruning process. In this case m = 1, so only one node is kept for the backward case. The final set ς contains 11 solutions produced in two stages.

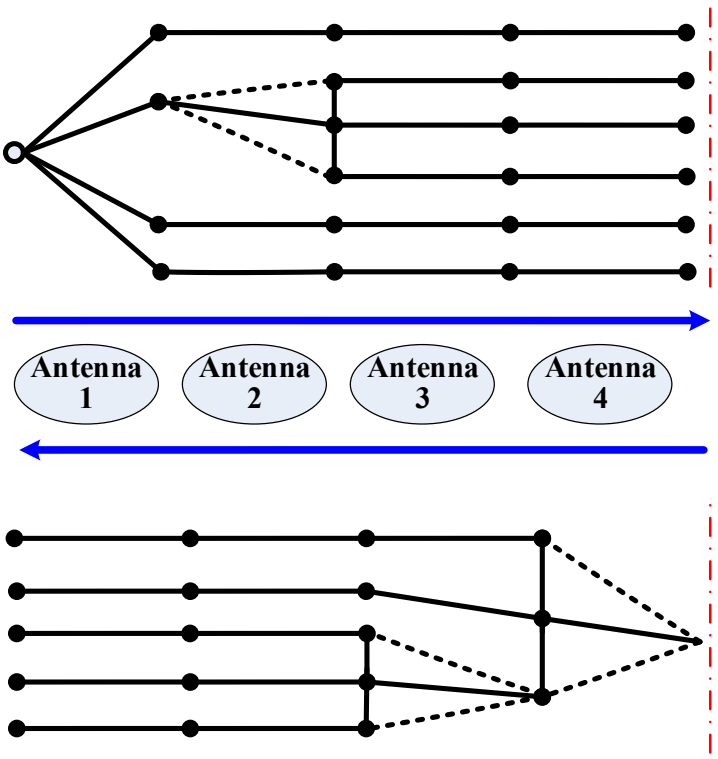


Figure 4.2 The bidirectional search diagram for the case of 4QAM, where the dashed lines indicate the nodes introduced by the candidate adding method

4.3.3 The techniques used in the algorithm

This bidirectional algorithm introduced a combination of proposed techniques which improve detection performance. There are several reasons why the proposed algorithm produces good results based on the following.

1) *Full Expansion*: Note that a ZF solution does not achieve good performance because of error propagation, since only one child node is expanded in the tree search. Thus, in the proposed algorithm the forward processing uses a full expansion of the root node and candidate adding in the top-half layers, so that a performance loss can be alleviated.

2) *Bit Flipping:* Although only the node with minimum PED is chosen for bit-flipping in the candidate adding procedure, the bit-flipped symbols are still likely to have small metrics in the case of Gray code mapping and QAM alphabets because the flipped sibling nodes are nearest neighbors of the transmitted symbol estimate. In addition, bit flipping is simple and implementation friendly.

3) *Zero-Forcing on Lower Layers*: At the lower layers in the decomposition $\mathbf{H} = \mathbf{QR}$, the diagonal elements $r_{i,i}$ of the upper triangular matrix $\mathbf{R}$ are distributed independently according to a standard Gamma distribution with $n = N+1-i$. This means that the probability that $r^2_{i,i}$ takes on small values increases with i. In other words, the lower the layer is, the higher its SNR value will be [47]. Therefore, it is reasonable to use the ZF

solution in the lower half layers to achieve a better overall complexity/performance tradeoff.

4) *Reference Layer and Pruning*: The bidirectional technique introduces an extra reference layer in detection, which has proven to be effective for soft detection in the LORD algorithm [27]. Furthermore, after the forward processing has been finished, a sorting and pruning procedure can be adopted to lower the backward processing complexity considering that the nodes in the last layer already have adequate reliability.

In contrast to sphere detectors, two QR decompositions must be performed for each sub-channel, but this pre-processing is done only once per channel estimate for orthogonal frequency-division multiplexing (OFDM) symbols at the receiver. Normally, the channel is assumed to be constant over a duration of several packets, so that this additional preprocessing cost would be fairly low.

4.4 Simulation and Complexity Analysis

All of the simulations use a rate 2/3 convolutional code in a MIMO-OFDM system with $N_t = N_r = 4$, 64QAM, and soft-input Viterbi decoding. Each subcarrier is subject to flat fading, i.e., all the entries in the MIMO channel matrix are independent random Gaussian variables. We also assume that the receiver has perfect channel knowledge. We compared the proposed algorithm with soft-output K-best, LORD and SFSD, using the FER as the performance metric, with the results shown in Figure 4.3. The parameter $K = 25$ in the K-best algorithm is used. It can be seen that the bidirectional algorithm with $m = 12$ achieves a better FER than the other algorithms.

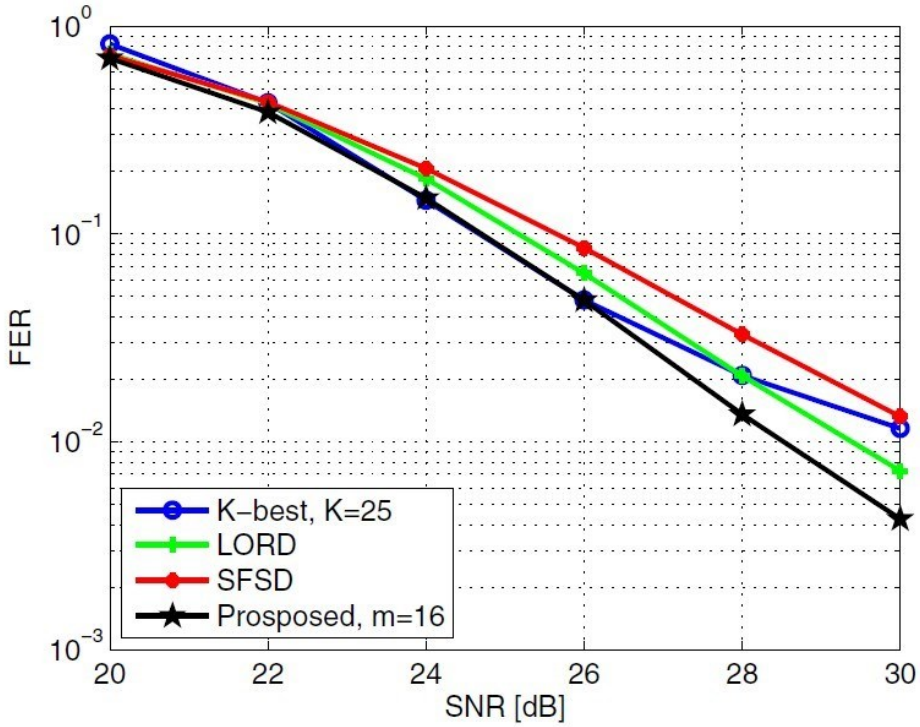


Figure 4.3 Performance simulation: 64QAM, 4 × 4 antennas

In addition, it can be inferred that the value of *m* has an impact on both complexity and performance. So the performance attained with different *m* values was also considered. Figure 4.4 shows that the performance improves as *m* increases. But the improvement become negligible when *m* is larger than a certain value. So a given implementation choice can be selected as a performance/complexity trade-off depending on the circumstances.

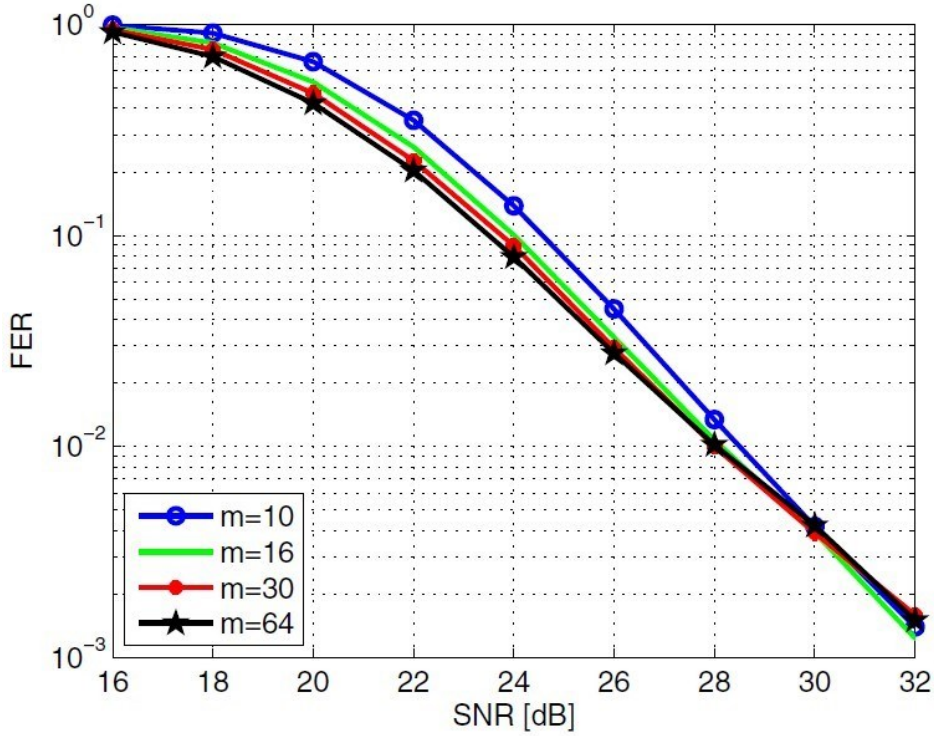


Figure 4.4 Performance simulation for different *m*, 64QAM, 4 × 4 antennas

For the specific parameters chosen in Figure 4.3, the complexity results of the 64QAM, 4×4 system are given in Table 4.1. The first column in the table shows the complexity in terms of checked symbols, which is the key metric [48]. The other columns list the required number of real multiplications, additions and comparisons for node expanding and the LLR computation, without the pre-processing stage. As can be seen, the K-best method requires a large number of comparisons in order to sort and prune the nodes in each layer, while the bidirectional algorithm only needs to sort once in order to choose the nodes for the backward processing. Compared to LORD, the proposed algorithm inherits the ZF solution's computational characteristics and thereby saves more than half of the operations because only two reference layers are considered instead of

four. While SFSD has fewer nodes expanded so that it has fewer multiplication and addition operations, the proposed algorithm gives better performance than SFSD.

Table 4.1 Number of real operations

Detector	Symbols	Multiplications	Additions	Comparisons
K-best (K=25)	364	2450	2462	10306
LORD	1024	9792	9240	1488
SFSD	292	2520	2504	1944
Proposed (m=12)	346	3168	3134	1176

4.5 Conclusion

A novel bidirectional processing algorithm for soft-output MIMO systems has been proposed that combines features from several types of fixed complexity tree search procedures. The algorithm obtains a list of candidates used to calculate likelihood information in parallel and it includes two stages for tree searching, where each stage corresponds to one direction of the path selection process. The simulation results demonstrate that the proposed approach achieves a higher performance than previously proposed algorithms while, at the same time, having a comparable computational cost. The algorithm can also easily tradeoff for different performance and complexity objectives. Moreover, its parallel nature and fixed throughput characteristics make it attractive for VLSI implementation.

Chapter 5

Fixed and Low Complexity MIMO Detection Algorithm without QR Decomposition

This chapter presents a novel low-complexity hard output MIMO detection algorithm for LTE and WiFi applications. Our method provides a well-defined tradeoff between computational complexity and performance. The proposed algorithm is an enhancement of the ZF method. It does not require QR decomposition as a preprocessing step, as is commonly needed in SD based algorithms such as K-best detection and lattice reduction. Simulations have been performed for 3x3 and 4x4 MIMO configurations, with both 16 and 64 QAM constellations. Results show that the proposed algorithm uses a much smaller number of ED calculations while attaining only a 0.5dB loss compared to MLD.

5.1 Introduction

MIMO techniques have been incorporated into modern wireless communication standards such as IEEE 802.11n (WiFi), 3GPP LTE, WiMAX and HSPA+ due to their

high spectral efficiency. One of the key challenges in exploiting the potential of MIMO systems is to design high-throughput, low-complexity detection algorithms while achieving near-optimal performance. Unfortunately, most algorithms have a complexity that grows rapidly with the number of antennas and with the constellation size.

There are many detection algorithms for MIMO at the receiver side. The full search MLD [49-50] has the optimal bit error rate; however, its computational complexity makes it impractical for hardware implementation, especially for large QAM constellations. Other algorithms such as ZF and MMSE detection have reduced complexity but at the cost of poorer performance. Some other methods like sphere detection [51-52] appear to be promising alternatives, but their computational complexity is significant. Near-optimal detection algorithms such as K-best detection [27, 22] have constant complexity for a given K value. However, their performance is not acceptable for small values of K and the complexity increases with K. Moreover, most of these algorithms require QR decomposition [37, 24], which is computationally intensive. Here, we are interested in a system that can be used for both LTE and WiFi applications. That is to say, it must be able to function under worst-case conditions for both of these standards. Accordingly, the channel matrix **H** cannot be assumed to be flat fading or frequency-invariant and it must be treated as an independent matrix for each transmitted subcarrier.

This chapter introduces a low complexity MIMO detection algorithm which can be used for both LTE and WiFi systems. The key contribution is a methodology that combines ZF detection with obtaining the first neighbors of a constellation node while

eliminating the need for QR decomposition. The complexity of the initial phase of the algorithm depends on the constellation size but in the subsequent phase it is a function of number of neighbor nodes in the constellation and K, which is the number of s_1,s_2 pairs we kept in the first step.

The remainder of this chapter is organized as follows: Section 5.2 describes our proposed algorithm in detail. Section 5.3 presents the simulation environment and gives our results. In section 5.4, the architecture of 3x3 MIMO system for 16QAM are given. Lastly, we give our conclusions in Section 5.5.

5.2 Proposed Algorithm

5.2.1 Motivation

Our goal is to create a modem design for both LTE and WiFi having acceptable detection performance and hardware cost. We seek to develop an algorithm that has performance close to that of ML and which has relatively low complexity. Specifically, QR decomposition will not be employed as it is computationally intensive. This leads to substantial savings in hardware costs. Specifically, we need to support 64 QAM for LTE and 256 QAM for WiFi.

In general, the most computationally intensive part of MIMO detection is the computation of the ED values. In order to reduce the computational complexity, we propose a novel approach combined with the well-known low complexity ZF method.

5.2.2 The proposed algorithm

We will use a 4x4 MIMO system as an example to explain our algorithm. In this case, the baseband system model is expressed as:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ h_{41} & h_{42} & h_{43} & h_{44} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} + \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{5.1}$$

First, we reorder the columns of **H** (as well as the corresponding elements of vectors **s** and **y**) in increasing order of the power of the columns. That is, after the reordering, column four will have the largest power. On average, then, more reliable information will be carried on symbol s_4 and less reliable information will be contained in s_1. For simplicity, we keep the same notation for the reordered channel and vectors.

So as to provide greater options for the least-reliable symbol s_1, we first move all terms containing s_1 to the left side of (5.1).

$$\mathbf{r} = \mathbf{y} - \mathbf{h}_1 \mathbf{s}_1 = \mathbf{H}_{234} \begin{bmatrix} s_2 \\ s_3 \\ s_4 \end{bmatrix} + \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}, \textit{where } \mathbf{H}_{234} = \begin{bmatrix} h_{12} & h_{13} & h_{14} \\ h_{22} & h_{23} & h_{24} \\ h_{32} & h_{33} & h_{34} \\ h_{42} & h_{43} & h_{44} \end{bmatrix} \tag{5.2}$$

If the QAM constellation size is M, there are M possible s_1 values. Afterwards, for each possible symbol of s_1 we solve the ZF solution for $[s_2, s_3, s_4]^T$ which gives us s_{234}^{ZF} .

$$\mathbf{s}_{234}^{ZF} = (\mathbf{H}_{234}^{H} \mathbf{H}_{234})^{-1} \mathbf{H}_{234}^{H} \mathbf{r} \tag{5.3}$$

By slicing them into constellation nodes, we get M different $[s_1, s_2, s_3, s_4]^T$ values. After that, we calculate all ED values from the M different $[s_1, s_2, s_3, s_4]^T$ vectors and keep only the K best pairs of s_1, s_2.

Next, for each pair of s_1 and s_2, we obtain the first neighbors of s_2. "Neighbors" means the closest surrounding nodes of the given node in the QAM constellation. We keep s_2 and its first neighbors as a new set $s_{2\_1nei\_set}$. As in Figure 5.1, for an s_2 located at any of the solid points, all of the nodes inside its enclosing dashed rectangle are the new sets $s_{2\_1nei\_set}$.

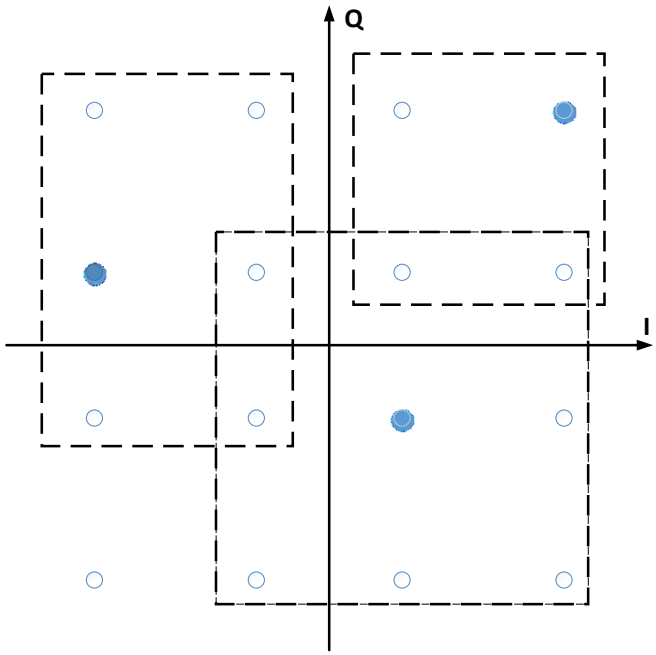


Figure 5.1 First neighbors in 16 QAM

We only keep the first neighbors because simulation results show that this will have a favorable balance of performance vs. complexity. As can be seen from the figure, the new set $s_{2\_1nei\_set}$ will contain up to 9 nodes. After that, for each symbol inside $s_{2\_1nei\_set}$, move the terms containing s_1, s_2 to the left of (5.1):

$$\mathbf{r}' = \mathbf{y} - \mathbf{h}_1\mathbf{s}_1 - \mathbf{h}_2\mathbf{s}_2 = \mathbf{H}_{34}\begin{bmatrix} s_3 \\ s_4 \end{bmatrix} + \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}, where\ \mathbf{H}_{34} = \begin{bmatrix} h_{13} & h_{14} \\ h_{23} & h_{24} \\ h_{33} & h_{34} \\ h_{43} & h_{44} \end{bmatrix} \tag{5.4}$$

Then, obtain the ZF solution for s_3, s_4. Thus we have:

$$\mathbf{s}_{34}^{ZF} = (\mathbf{H}_{34}^{H}\mathbf{H}_{34})^{-1}\mathbf{H}_{34}^{H}\mathbf{r}' \tag{5.5}$$

Assuming there are b symbols in each $s_{2\_1nei\_set}$, after quantization we obtain b of $[s_1, s_2, s_3, s_4]^T$ vectors. Then we calculate all ED values for the b vectors and keep the best one. After K iterations of this operation, we will have K of the best symbol vectors. At the last stage, we find the best solution which yields the minimum squared ED value. The algorithm can be summarized as shown in the following table:

Table 5.1 Proposed algorithm for a 4x4 MIMO system

Step 1) Initialization: Reorder **H**, **y** and **s**

Step 2) Move s_1 to the left-hand side and find M s_{234}^{ZF}

Step 3) Calculate EDs of all M ***s*** vectors;
Keep the K best pairs of s_1, s_2

Step 4)
for i =1 to K do
$s_{2\_1nei} \leftarrow$ Obtain b 1st neighbors of s_2
for j =1 to b do
Move s_1, s_2 to the left-hand side and find s_{34}^{ZF}
end
Calculate EDs of all b ***s*** vectors
Keep the best ***s*** vector
end

The proposed algorithm has the following features:

- Aside from Step 2, all other steps have a complexity independent of the constellation size.
- We only need to calculate the ED $K \cdot b$ times, which is the main cost of the computation.
- The complexity does not depend on the SNR.
- It has a fixed complexity for a given K and neighbor size, which makes its hardware implementation simpler.
- The computational complexity scales up only moderately with an increase in constellation size.
- It can be extended in a straightforward manner to more antennas and/or larger constellations.
- There is no need to compute QR decompositions, as is required in most other fast search algorithms.

For a 3x3 system, the algorithm is similar to that of 4x4 system. The only difference is in step 4) above. Instead of s_{34}^{ZF} , we find s_{3}^{ZF} . Aside from this, the other steps remain the same.

For a larger constellation size, we can still achieve good performance by slightly increasing the K value or by increasing the neighbor region to include 2nd neighbors or 3rd neighbors. This would still have a relatively low complexity compared to other algorithms.

5.3 Simulation and Complexity Analysis

In this section the performance results of our proposed algorithm are reported. We tested 3x3 and 4x4 MIMO systems with both 16 QAM and 64 QAM sizes. 10,000 simulation trials were used to obtain the probabilistic results. We used SNR values such that the SER is in the range near 10^{-3}. We also compared our proposed algorithm with ML, MMSE, and the K-best algorithm with different K values. The results for a 3x3 system and are shown in Figures 5.2 and 5.3. The results for a 4x4 system are shown in Figures 5.4 and 5.5.

It is evident from Figures 5.2 through 5.5 that our proposed algorithm outperforms the MMSE and K-best algorithms. Even when the K value of K-best is as high as 15 for 16QAM modulation in a 4x4 system, the performance is similar to our performance. The proposed algorithm has only a 0.5dB performance loss at high SNR compared with ML detection.

The complexity comparisons are given in Table 5.2 in terms of the total number of times that the ED needs to be calculated. As can be seen, the K-best method requires a large number of ED calculations in order to sort and prune the nodes in each layer, whereas the proposed algorithm only needs $M+K \cdot b$ ED calculations, where M is the QAM constellation size, K is the number of s_1, s_2 pairs saved, and b is the number of symbols in the set of neighbor nodes. In particular, it only depends on M initially and subsequently it depends on the product of K and b.

It is also noteworthy that the proposed algorithm is accomplished without the need for QR decomposition in the pre-processing part. In LTE, we have to allow for a

new channel matrix for each subcarrier and for each OFDM symbol. If QR decomposition had been required in the algorithm, this would have represented a significant additional computational cost. The K-best or K-best based methods in [22] need QR decomposition, which will greatly increase their actual complexity.

Table 5.2 Number of times ED is calculated for different algorithms

Antennas & QAM size	Proposed alg. (Avg. No.)	Ref. [22]	K-best
3x3, 16QAM	35	--	144
3x3, 64QAM	79	--	448
4x4, 16QAM	54	72	736
4x4, 64QAM	117	152	2944

To determine the complexity, we used the average value of *b*, weighted by the number of occurrences of each possible value. In the case of 16QAM, *b* is a value in the set {4, 6, 9}. In the constellation, there will be 4 occurrences of the value 4, 8 occurrences of the value 6 and 4 occurrences of the value 9, so that the weighted average value of *b* is 6.25. For 64QAM, a similar calculation yields a weighted average value of b equal to 7.56. The complexity is compared with the advanced detection method in [22] and also the K-best algorithm.

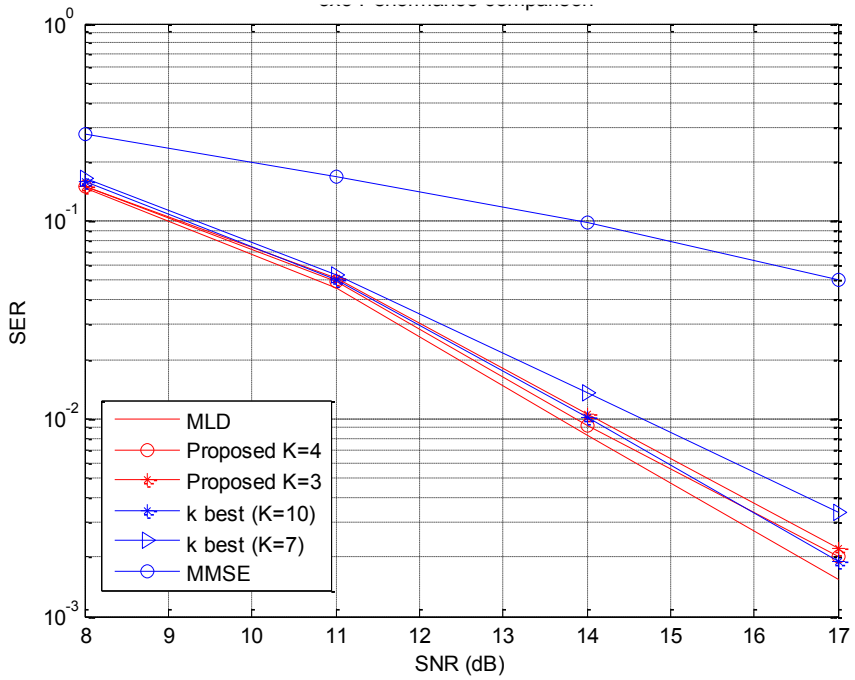


Figure 5.2 SER vs. SNR performance of a 3x3 MIMO system using 16 QAM

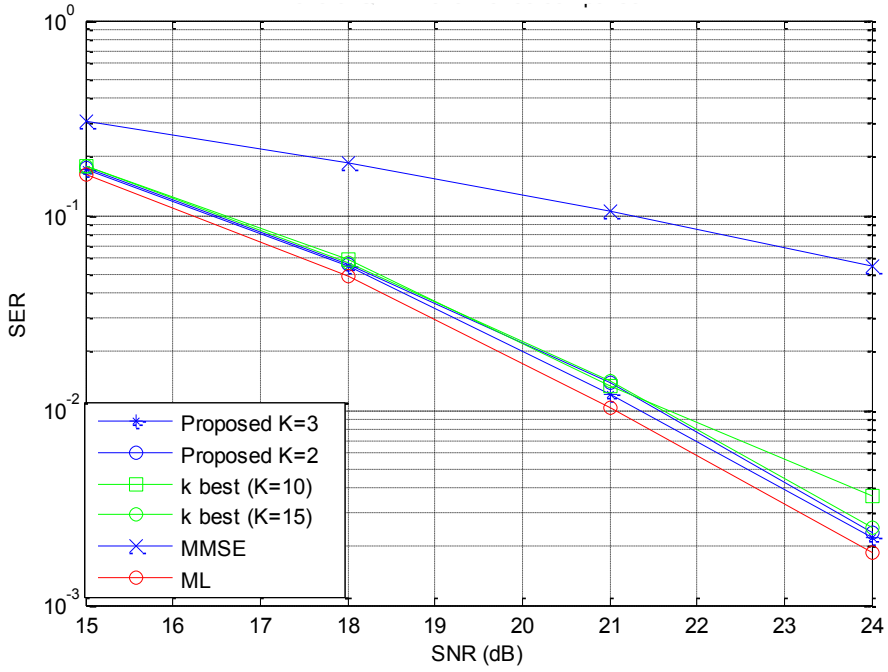


Figure 5.3 SER vs. SNR performance of a 3x3 MIMO system using 64 QAM

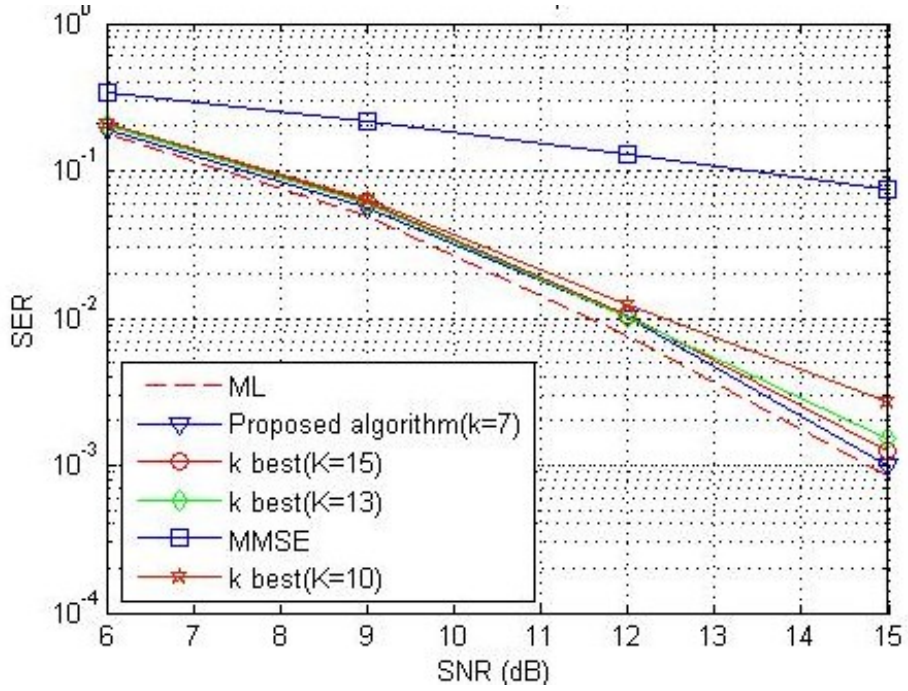


Figure 5.4 SER vs. SNR performance of a 4x4 MIMO system using 16 QAM

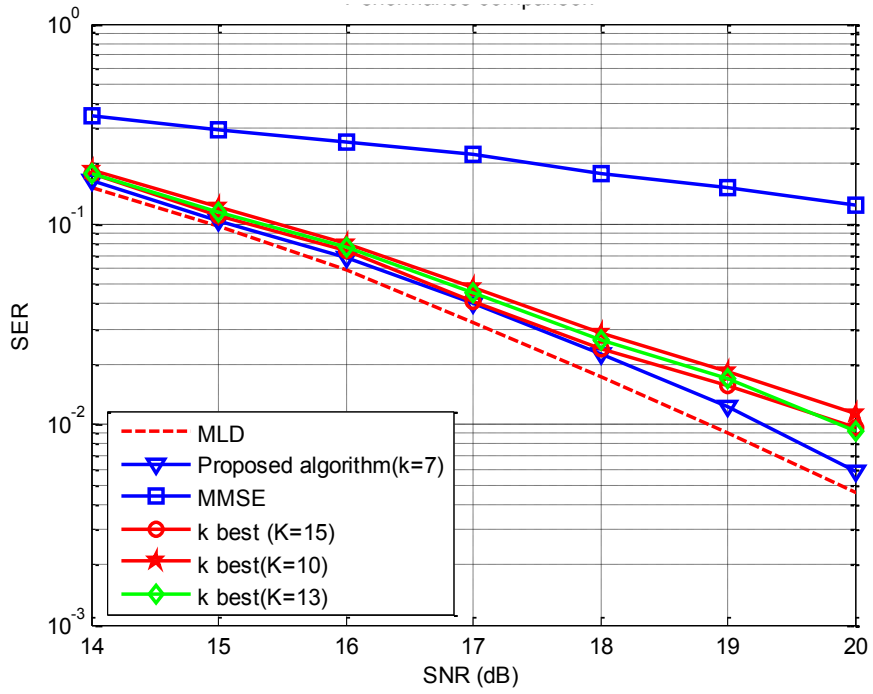


Figure 5.5 SER vs. SNR performance of a 4x4 MIMO system using 64 QAM

5.4 Low-Complexity High-Throughput 3x3 MIMO Detector Architecture

While the performance of the proposed algorithm is close to that of ML detection, its efficient implementation is challenging. In particular, the proper tradeoff between the hardware requirements and throughput is a critical issue.

5.4.1 Overall architecture of MIMO detector

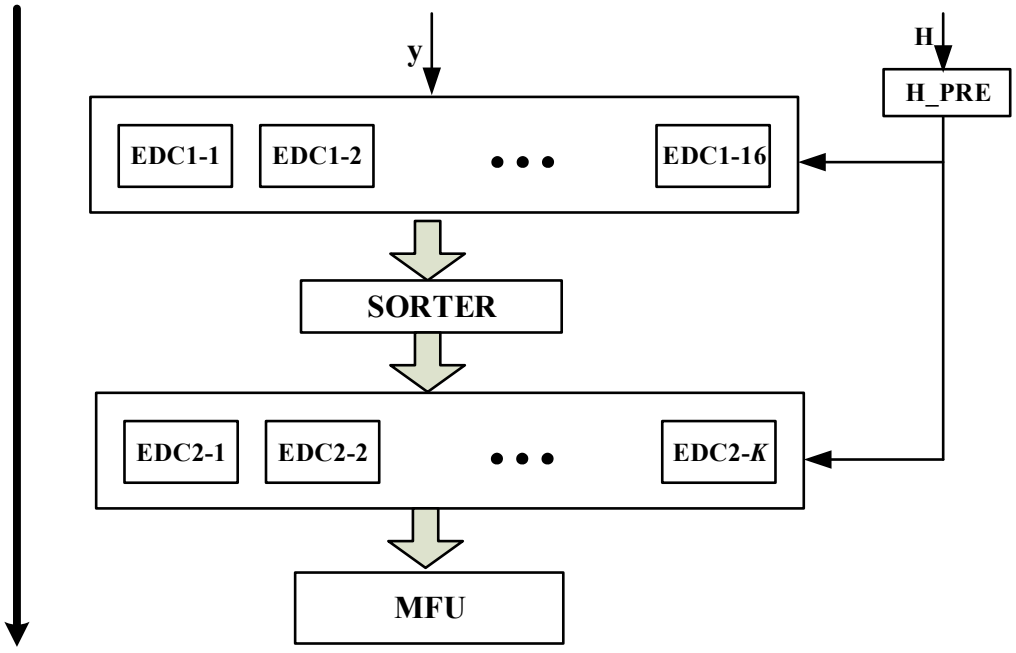


Figure 5.6 Block diagram of the proposed semi-parallel MIMO detector

As an example of a 3-by-3 MIMO system implementation, we use a pipeline structure to design the MIMO detector following the steps in Table 5.1. For the calculation of the

EDCs (which stands for Euclidean distance calculation units), we use a parallel architecture to increase the output data rate. The overall semi-parallel architecture of the proposed detector is shown in Figure 5.6.

The inputs **y** and **H** are a complex-valued vector and matrix, respectively. **H** is fed into a preprocessing unit H_PRE. This unit calculates $(\mathbf{H}_3^H\mathbf{H}_3)^{-1}$ and $(\mathbf{H}_{23}^H\mathbf{H}_{23})^{-1}$, and sorts the columns of **H**. Note that $\mathbf{H}_3^H\mathbf{H}_3$ and $\mathbf{H}_{23}^H\mathbf{H}_{23}$ are a scalar and a 2-by-2 Hermitian matrix, respectively. These calculations will be discussed in more detail in the following section.

y is input to *M* (i.e., the QAM size) EDC1 blocks, each of which computes the ED value of **y-Hs**. The EDC1 block consists of the ZF calculation, constellation node quantization and the ED calculation. After these calculations, the ED values are sorted. We only keep the *K* best of the nodes having the minimum ED values.

In the next pipeline layer, each of the s_2 is expanded to its neighbor set $s_{2\_1nei\_set}$ and the ED values are calculated serially using one EDC2 block. There are a total of *K* instances of the EDC2 unit so that the calculations can be performed in parallel. Finally, the minimum finding unit (MFU) is used to obtain the minimum among the *K* best nodes and produces the final output.

5.4.2 H_PRE architecture

The block diagram of H_PRE is shown in Figure 5.7. As noted above, the function of the H_PRE block is to sort the columns of **H** and to generate $(\mathbf{H}_3^H\mathbf{H}_3)^{-1}$ and $(\mathbf{H}_{23}^H\mathbf{H}_{23})^{-1}$, which requires matrix inversion. However, since this is followed by the QUAN block we

do not need to perform the inversion here. Instead, it can effectively be performed in the QUAN block through a scaling operation.

We define $p = \mathbf{H}_3^{\mathrm{H}}\mathbf{H}_3$, $\mathbf{P} = \mathbf{H}_{23}^{\mathrm{H}}\mathbf{H}_{23}$. Note that p is a scalar and $\mathbf{P}$ is of size 2-by-2, which will largely reduce the computational complexity. Also, $p_{22} = p$. Then, $\mathbf{P}^{-1} = \mathbf{C}/d$, where $\mathbf{C}$ is the adjugate matrix of $\mathbf{P}$ and d is the determinant of $\mathbf{P}$. Since $\mathbf{P}$ and $\mathbf{C}$ are 2-by-2 Hermitian matrices, not all of their elements have to be directly calculated. For $\mathbf{P}$ and p, we only need to calculate 3 terms, i.e., $p_{11} = h_{12}^* h_{12} + h_{22}^* h_{22} + h_{32}^* h_{32}$, $p_{22} = h_{13}^* h_{13} + h_{23}^* h_{23} + h_{33}^* h_{33} = p$, $p_{12} = h_{12}^* h_{13} + h_{22}^* h_{23} + h_{32}^* h_{33}$. The remaining element, $p_{21} = p_{12}^*$, is obtained by complex conjugation. The C_MAC (complex multiplier accumulator) block in Figure 5.9 is used to calculate these values.

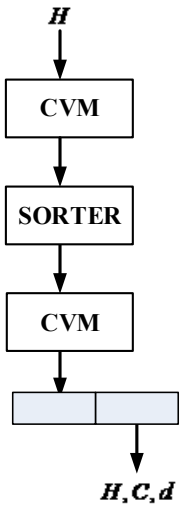


Figure 5.7 Block diagram of H_PRE

First, the 9 elements of **H** are fed into the CVM (complex vector multiplier) block in Figure 5.7. This block is used to calculate the power of the columns of **H**. The outputs of the CVM are passed to a SORTER, which sorts the columns in increasing order. Then, the subsequent CVM block is used to calculate **P**, **C** and p.

The CVM block diagram is shown in Figure 5.8. The values of **H**, **C** and d are stored in registers. Control logic is used to activate reading input values, sending values to the C_MAC block and for writing output values.

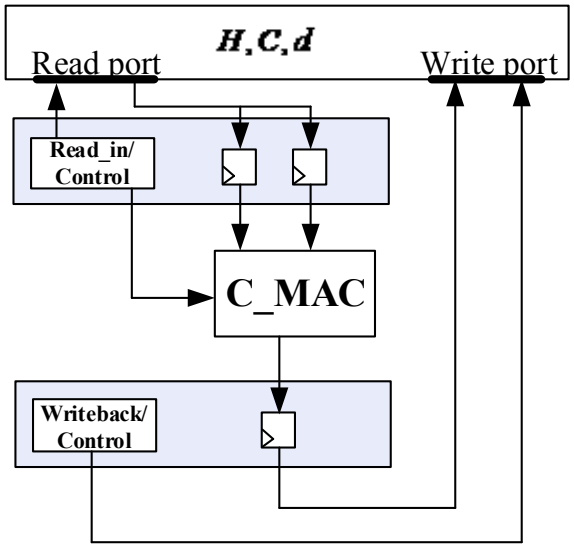


Figure 5.8 CVM block architecture

Since it is costly to compute inverses in hardware, an alternative procedure is used. First, the conjugate of the determinant, d^*, is multiplied with **C**. Then, the scalar $d^* \cdot d = |d|^2$ will be used as a scale factor in the QUAN block. Similarly, for the inversion of p, $1/p = p^*/(p^* \cdot p) = p^*/|p|^2$. We will use p^* instead and send $|p|^2$ as scale factor to the QUAN block.

The detailed architecture of the C_MAC is shown in Figure 5.9. Two complex values are input to the multiplier Complex_mu and accumulated to obtain a final output.

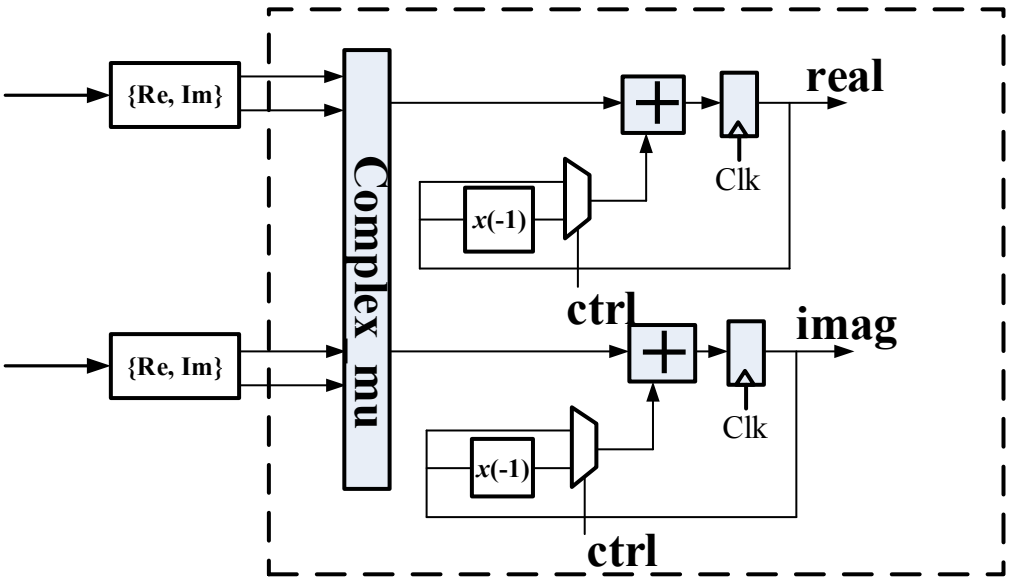


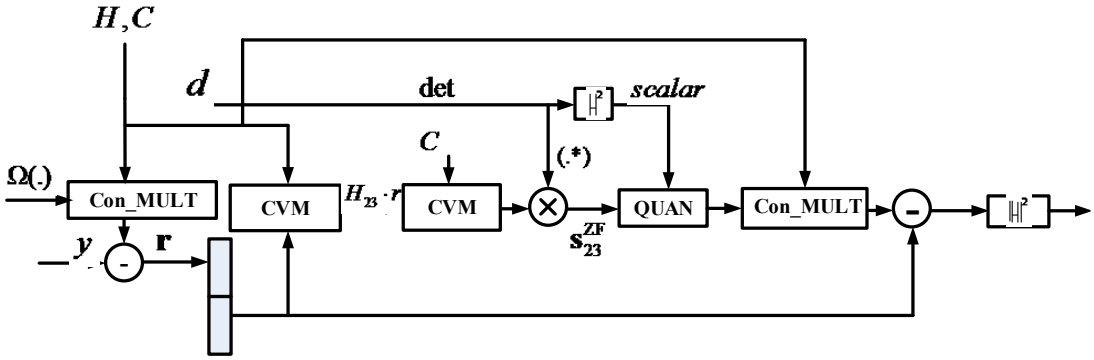
Figure 5.9 C_MAC block diagram

5.4.3 EDC1 and EDC2 block architectures

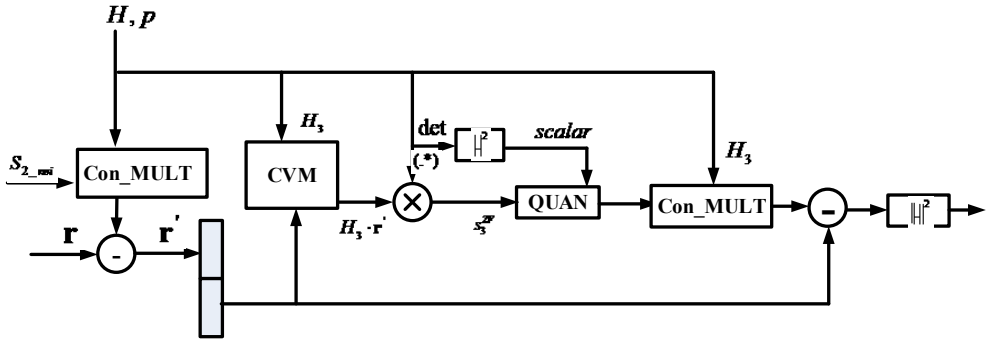
The structures of the EDC1 and EDC2 blocks are shown in Figure 5.10 (a) and (b), respectively. As mentioned in the previous section, the EDC blocks consist of constellation node multiplier (Con_MULT) units, the CVM, the constellation node quantization block QUAN, and the ED calculation.

In EDC1, the **H** and constellation values are fed in to a constellation node multiplier Con_MULT. In EDC2, **H** values and the s_2 neighbors are sent to Con_MULT.

The outputs are used to compute **r** or **r′** as in equations (5.2) and (5.4), respectively. In EDC1, the calculated **r** values are stored in a register. The subsequent CVM is used to generate $\mathbf{H}_{23}\cdot\mathbf{r}$. Next, **C** and d^* are multiplied to $\mathbf{H}_{23}\cdot\mathbf{r}$. A quantizer QUAN maps the s symbols to the constellation nodes. The quantized s_{23}^{ZF} or s_3^{ZF} is sent to a Con_MULT to complete the calculation of $\mathbf{H}_{23}\cdot\mathbf{s}_{23}$ or $\mathbf{H}_3\cdot\mathbf{s}_3$ and is then subtracted from **r** or **r′** in order to obtain **y-Hs**. Afterward, there is an ED calculator to obtain its Euclidean distance value. The structure of EDC2 is similar; the only difference is that the output of CVM does not multiply the **C** matrix but only multiplies p^*.



(a)



(b)

Figure 5.10 EDC block diagrams: (a) EDC1, (b) EDC2

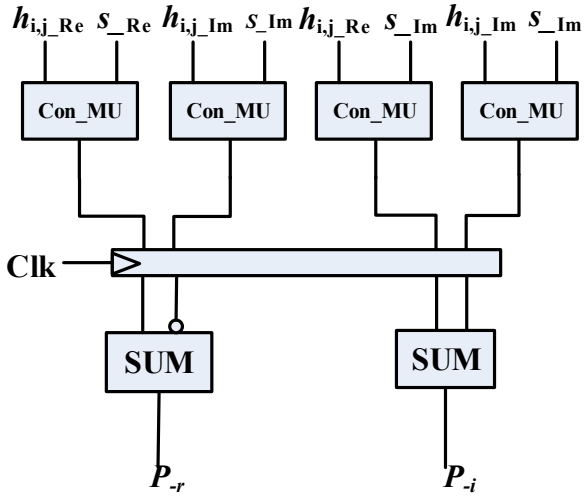


Figure 5.11 Complex multiplier architecture for constellation nodes

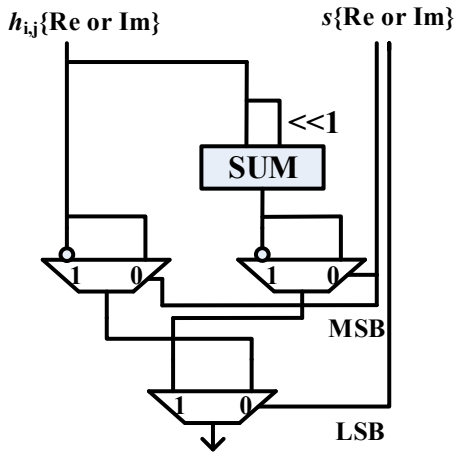


Figure 5.12 Con_MU architecture

Figure 5.11 shows the complex multiplier for the constellation nodes and matrix elements. Since the constellation node values are restricted to being symmetric odd

integers, the multiplier architecture is much simpler than for a general multiplier. For a 16QAM constellation, the real and imaginary parts of symbol s can only be from the set {-3, -1, 1, 3}. Thus, the Con_MU in Figure 5.11 is implemented as shown in Figure 5.12. First, h_re/im is shifted left by 1 bit and added to h to obtain {-3·h_re/im, -1·h_re/im, 1·h_re/im, 3·h_re/im}. Then, the least significant bit (LSB) and most significant bit (MSB) of s_re/im are used to select the correct value.

Since $\mathbf{r}$ is calculated in EDC1, there is no need to calculate $\mathbf{y}\text{-}\mathbf{h}_1 s_1$ again in EDC2. It can simply be passed from the EDC1 blocks to the EDC2 blocks and reused. Similarly, the $\{s_1, s_2\}$ pairs used in EDC2 can be obtained from EDC1 level after obtaining the quantized s_{23}^{ZF}. Between the stages of EDC1 and EDC2, there is a SORTER block to sort the M ED values and to find the K best $\{s_1, s_2\}$ pairs.

After passing the $\{s_1, s_2\}$ pairs to the EDC2 stage, s_2 is input to a Get_neighbors block to generate $s_{2\_1nei\_set.}$ Then, EDC2 is used to calculate the ED values. This process is similar to tree expansion from a parent node to b child nodes. Since there are K parent nodes, there are a total of K parallel EDC2 blocks at this stage. And after getting every b ED values, there is a MIN block to find the minimum among them. The detailed architecture is show in Figure 5.13.

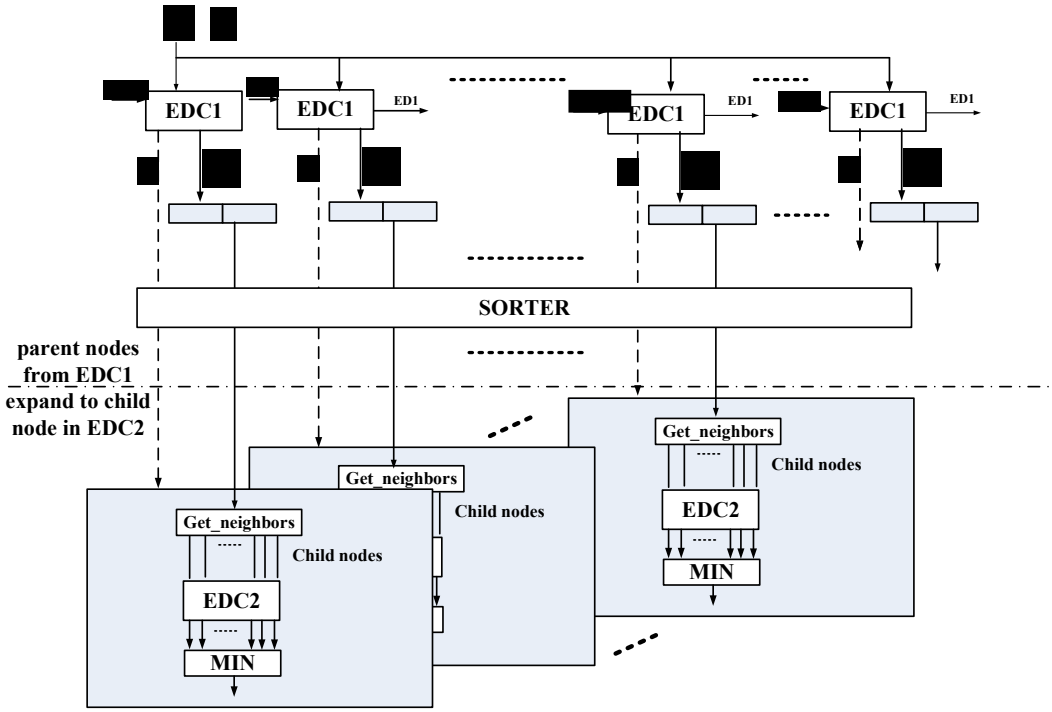


Figure 5.13 Parallel architecture of EDC1 and EDC2

The structure of the quantization block diagram QUAN is shown in Figure 5.14. For a 16QAM constellation, the values for either the real and imaginary parts of symbol *s* come from the set {-3, -1, 1, 3}, and the slicing boundary is {2, -2}. Accordingly, the scalar obtained from the H_PRE block is shifted left by 1 bit position prior to being input to the comparator.

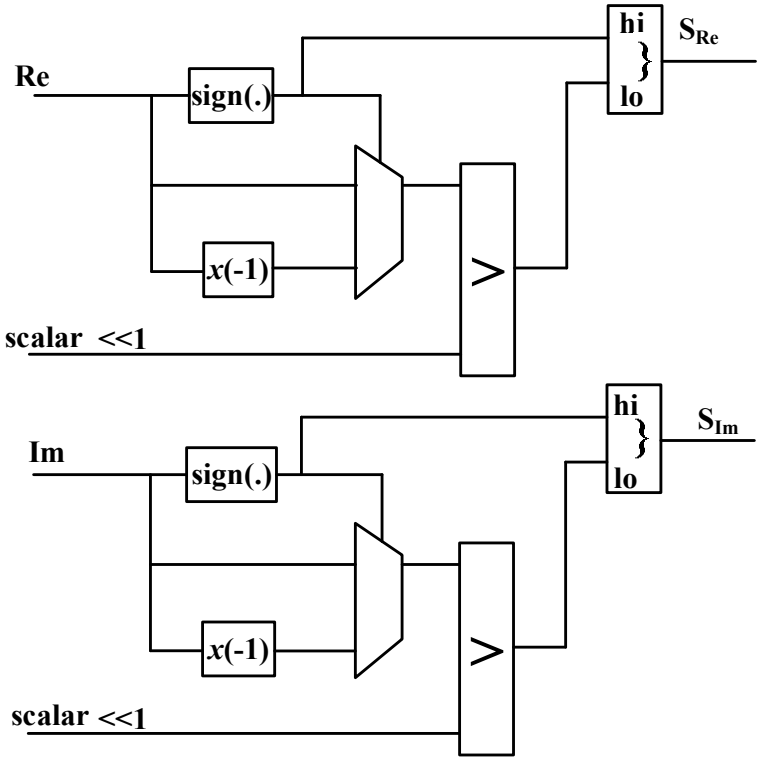


Figure 5.14 Block diagram of QUAN

5.4.4 SORTER and MFU blocks architectures

Between the two stages of EDC1 and EDC2, there is a SORTER block to sort the M ED values and to find the K best $\{s_1, s_2\}$ pairs. The detailed architecture of this block is shown in Figure 5.15. Block "max" and "min" are used to find the max and min value between two values.

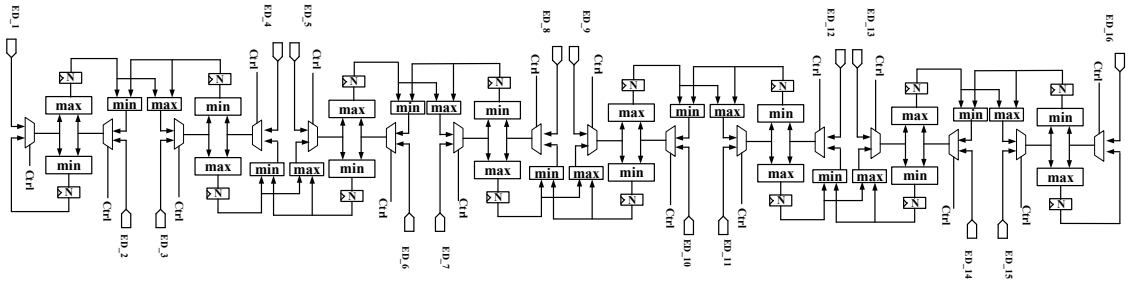


Figure 5.15 SORTER architecture for 16QAM

On the first clock cycle, 16 ED values are grouped into pairs and are compared simultaneously. Then, the adjacent max output and min output values are compared. In this way, the min values are gradually shifted left. As a result, over 8 clock cycles the 16 values will be sorted and stored in increasing order in the 16 N-bit registers, from left to right. In general, it will take $K/2$ clock cycles to sort K values using this structure.

The MFU block is shown in Figure 5.16. The figures illustrates an example of finding the minimum of 9 values.

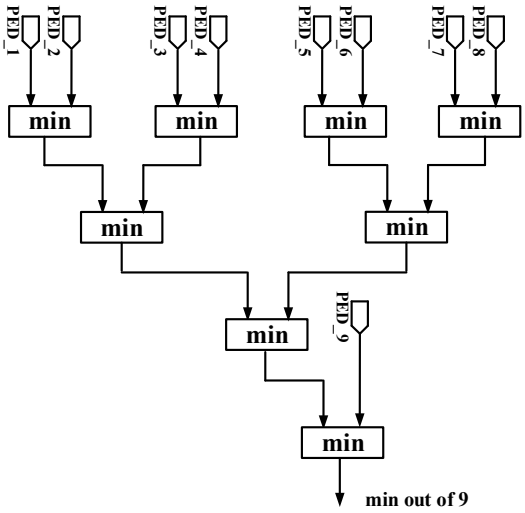


Figure 5.16 Block diagram for MFU

5.4.5 FPGA results

A Xilinx Virtex 5 FPGA was used for estimating the latency and hardware cost of the 3x3 MIMO detector architecture. The xc5vsx95t-2ff1136, 65-nm CMOS technology was chosen. The particular device contains 58880 FF/LUTs and 640 DSP48 blocks.

The CVM block consumes a considerable amount of resources, since the C_MAC portion of this unit by itself requires 3 DSP48 blocks. The EDC1 and EDC2 are similar to each other in hardware consumption. The EDC1 block contains 2 CVM, 1 complex multipliers, 2 complex adders, 2 ED calculators, 2 constellation node multipliers and the QUAN block. The EDC2 unit has only 1 complex multipliers but has one additional MIN block after it. A complex multiplier uses 3 DSP48, and a complex adder requires 1 DSP48.

We use the notation Qm.n to represent a fixed point number having m integer bits and n fractional bits so that the total word length is m+n. The signed numbers will have one additional sign bit. Table 5.3 summarizes the fixed point design parameters for the scaled **H**, received **y** and ED values.

Table 5.3 Fixed point design parameters for the 3x3 16-QAM system

Received **y** (signed fixed point)	Scaled **H** (signed fixed point)	ED (unsigned fixed point)
Q3.6	Q0.9	Q4.4

For the 16QAM 3x3 system, the estimated latency results are listed in Table 5.4.

Table 5.4 Latency of various blocks for 16QAM 3x3 MIMO system (K=3)

Unit	Latency (clock cycles)
H_PRE	34
EDC1	26
SORTER-16	8
EDC2	16*9=144
MFU	1

Using the fixed point numbers in Table 5.3, the hardware cost results for the 16QAM 3x3 MIMO detection system are given in Table 5.5. The estimated value for the total design was obtained by adding the values for each of the components that are used; it does not include some additional control and associated "glue logic" that would also be needed in the full design.

Table 5.5 Hardware cost values for a 16QAM 3x3 MIMO detection system

Unit	Resource utilization		
	LUT	FF	DSP48
H_PRE	1102	984	6
EDC1	1488x16=23808	890x16=14240	13x16=208
SORTER-16	414	358	0
EDC2	1375x3=4125	740x3=2220	10x3=30
MFU	52	24	0
Estimated total (percent utilized)	29501 (50.1%)	17826 (30.3%)	244 (38.1%)

Of course, one can always make a trade-off between throughput and hardware cost. Here we use 16 EDC1 blocks in parallel in the first stage and *K* EDC2 blocks at the later stage. Each EDC2 block is used serially to calculate ED values. If lower latency is required, we can use a parallel set of C_MAC blocks in the CVM unit.

5.5 Conclusion

In this chapter, a novel MIMO detection algorithm with low complexity has been proposed. This algorithm does not need to perform QR decomposition as a pre-processing step. It fulfills the requirements of LTE and WiFi even in worst-case situations in which the channel matrix is varying for all the sub-carriers. Thus, it overcomes a major disadvantage present in SD algorithms that are based on QR decomposition. The simulation results show that the proposed algorithm only has a 0.5dB SER performance loss compared to ML detection and it is achieved with a significantly reduced computational complexity. The algorithm is scalable to large QAM constellation sizes. In addition, we have designed the architecture for a 16QAM 3x3 MIMO detector based on this algorithm, and the estimated resource costs and latency values have been determined on a Xilinx Virtex 5 FPGA.

Chapter 6

QR Decomposition and Stochastic Computing

6.1 Implementation Methods of QR Decomposition and Stochastic Computing

QR decomposition is widely used in MIMO detection systems to reduce the search complexity. The most commonly used QR decomposition [53-55] algorithms are the Gram-Schmidt process, the Householder transformation and Givens rotations. Since Givens rotations do not require inversion or square root calculation, they are often used for hardware implementation. The matrix elements involved in Givens rotations consist of trigonometric functions. More generally, trigonometric and hyperbolic functions are required in many computational applications.

This chapter presents a FPGA implementation of trigonometric and hyperbolic functions using the stochastic computing methodology. The results are compared to the well-known CORDIC approach. Both designs are synthesized and implemented on a Xilinx Virtex-5 FPGA. The results are compared in terms of delay and area for various input data widths. The results show that the proposed design method has advantages over

the traditional CORDIC algorithm. Moreover, the same circuit can be used for all functions with only a change in coefficient values. In addition, it is more tolerant of soft errors (bit flips) than CORDIC implementations.

6.2 Stochastic Computing Implementation of Trigonometric and Hyperbolic Functions

There is a high demand for reducing the area of VLSI hardware designs so that the implementation cost can be reduced. Trigonometric and hyperbolic functions, which are used in a wide array of applications, are among the most area-consuming portions of some designs. The most convenient way to implement these functions is to use a look-up table [56]. However, such a table needs considerable memory to store the table values, which accounts for the majority of the hardware overhead. The CORDIC algorithm, invented by Jack E. Volder more than fifty years ago [57], has drawn considerable attention due to its easy implementation in hardware using only adders and shifters [58]. Therefore, CORDIC units are commonly used in systems that require a low hardware cost. However, in some applications even the CORDIC method may not be able to satisfy the area requirements.

Stochastic computing has drawn much attention recently due to its efficiency in hardware cost. Also, with the downsizing of semiconductor feature sizes due to Moore's Law, soft errors caused by ionizing radiation have become important, especially for circuits operating in harsh environments. Stochastic computing moderates the output errors caused by bit-flips. It has been used in applications such as image processing [59]

and LDPC decoding [60]. Recently, a novel structure used to compute mathematical functions has been proposed by Li et al. [59]. By using registers and multiplexers, the state machine architecture of stochastic computing can be applied for implementing various trigonometric and hyperbolic functions. A specific function is obtained by using an appropriate set of coefficient values. Thus, the same design structure can be reused for many different applications. In this paper, several different trigonometric and hyperbolic functions are implemented and compared using the stochastic computing and CORDIC methods. The superior tolerance to soft errors by the stochastic computing technique is also demonstrated.

6.3 Proposed Stochastic Computing Method

The Stochastic Computational Element (SCE) makes use of conventional digital logic gates to perform computations on stochastic bits streams, i.e., random bits [59]. There are two coding formats for stochastic computing. One is unipolar and the other one is bipolar. Here we use unipolar coding format. In a unipolar format, the range of a real number x is $0 \leq x \leq 1$. The probability that each bit in the stream is one is $P(X = 1) = x$ and, consequently, $P(X=0) = 1-x$. In a stochastic bit stream, if there are k ones out of a length N, then the real value represented by the bit stream is k/N. Li et al. proposed a general approach to synthesize a linear finite state machine (FSM) based SCE for a target function and applied it to image processing [59].

In this paper, we adopt the linear FSM structure in [59] to construct the trigonometric functions sine, cosine, arctangent, and their hyperbolic counterparts. The

functions are implemented in sequential logic with the FSM shown in Figure 6.1. Here, in order to achieve appropriate precision, we use 16 states. There are two inputs to the FSM, namely X and K. Each of them takes the form of a unipolar stochastic bit stream. Their actual values are the probabilities of 1s among all the bits and are each in the range of [0, 1]. There is one output t indicating the current state number S_t ($0 < t < 15$). The FSM has 16 possible output values, encoded as a 4-bit binary number t. In Figure 6.1, the number below each state symbol S_t is the decimal output value when the current state is S_t. The numbers along the arcs represent the transition condition, with the first value corresponding to the input X and the second value corresponding to the input K.

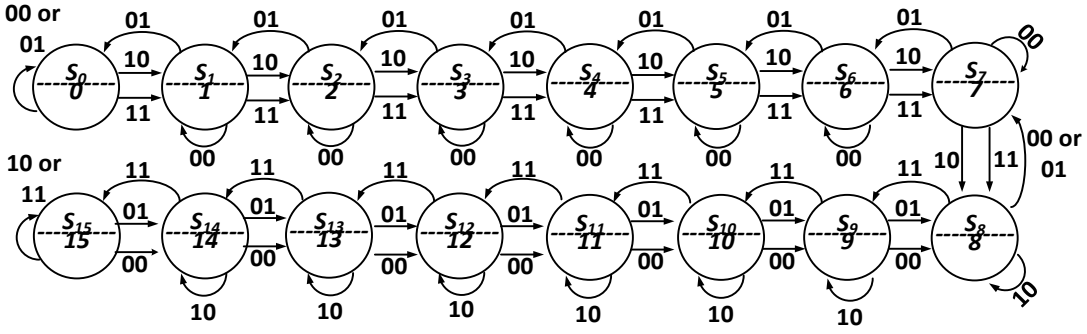


Figure 6.1 FSM transition diagram

Since the state transition is a special case of a Markov chain [61-62], the output of the FSM is the encoded output S_t. It depends on and only on the current state.

Let P_X, P_K, P_t represent the probability of ones in X, K, and t, respectively. P_t represents the probability of FSM output is t. It is derived that [59]

$$P_t = \begin{cases} \dfrac{(\dfrac{P_X}{1-P_X})^t P_K^{-t}}{\alpha}, 0 \le t \le 7 \\ \dfrac{(\dfrac{P_X}{1-P_X})^t P_K^{t-15}}{\alpha}, 8 \le t \le 15 \end{cases} \quad where \quad \alpha = \sum_{t=0}^{7} (\frac{P_X}{1-P_X})^t \cdot P_K^{-t} + \sum_{t=8}^{15} (\frac{P_X}{1-P_X})^t \cdot P_K^{t-15} \qquad (6.1)$$

The basic structure of the stochastic computing circuit is shown in Figure 6.2 [59]. As shown in Figure 6.2, it has two inputs X and K and one output Y. X and K are the inputs of the FSM. The current state number is the output t and it is used as the select inputs to the multiplexer. The multiplexer has 16 data inputs (ω_0, ω_1, …, ω_{15}). If the FSM has a current state S_t ($0 \le t \le 15$), then the output Y of the multiplexer will be selected as ω_t.

X, K, ω_t and Y are all stochastic bit streams. Let $P_{\omega t}$ and P_Y denote the probability of ones in ω_t and the probability of ones in Y, respectively. As seen from Figure 6.2, the probability input ω_t to the multiplexer is selected when the current state output is S_t. It is derived that [59]

$$P_Y = \sum_{t=0}^{15} P_{\omega t} \cdot P_t \qquad (6.2)$$

The method for calculating P_K and $P_{\omega t}$ for a given function is given in [59]. After we obtain the optimal P_K and $P_{\omega t}$, the stochastic bit streams K and ω_t are generated to implement the target function T(P_X) stochastically.

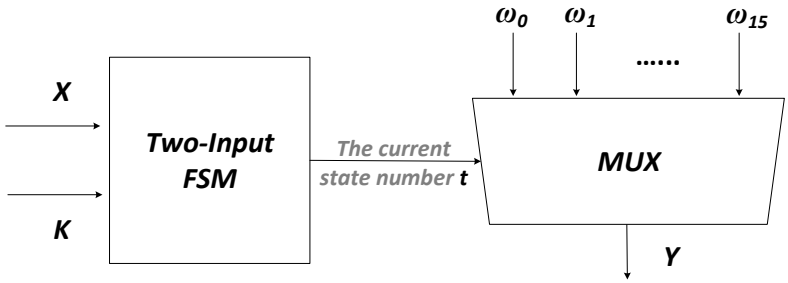


Figure 6.2 The circuit for synthesizing target function

The coefficient values of $P\omega_0$, $P\omega_1$, ..., $P\omega_{15}$ and P_K for the trigonometric functions are pre-calculated and are given in Table 6.1. We have calculated these values using the quadratic programming method described in [59]. The coefficients can be stored in a look-up-table if flexibility is needed, or they can be hardwired if desired.

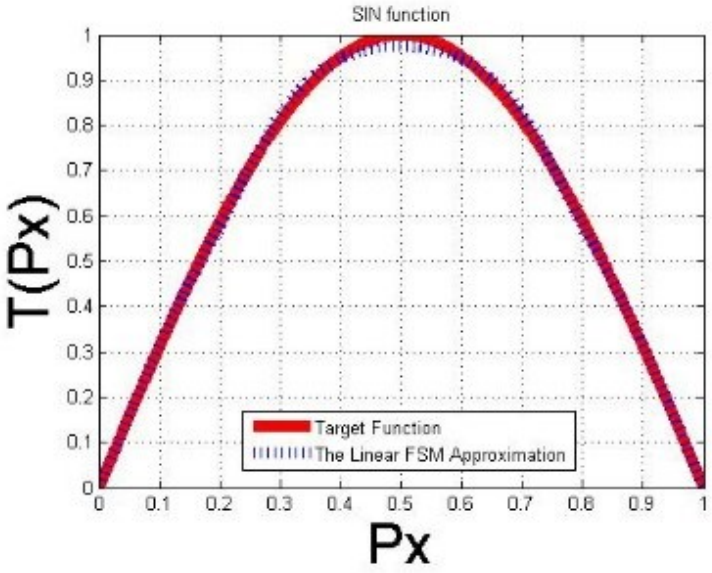
Table 6.1 Coefficient values for various functions

Func.	sin	cos	atanh	sinh	cosh	arctanh
$P\omega_0$	0	1	0	0	0.16	0
$P\omega_1$	0.95	1	0.79	0.12	0.16	0.02
$P\omega_2$	1	0.76	0.66	0.08	0.23	0.02
$P\omega_3$	0.35	0.97	0.22	0.10	0.17	0.05
$P\omega_4$	0	0.83	0.78	0.11	0.24	0
$P\omega_5$	1	0.95	0.44	0.07	0.15	0
$P\omega_6$	1	0.79	0.39	0.11	0.25	0.05
$P\omega_7$	1	1	0.88	0.09	0.16	0.01
$P\omega_8$	1	0.53	0.96	0.28	0.32	0.35
$P\omega_9$	1	0.37	0.98	0.72	0.68	0.98
$P\omega_{10}$	1	0.53	0.93	0.31	0.34	0.37
$P\omega_{11}$	0	0.43	0.99	0.63	0.69	0.44
$P\omega_{12}$	0.35	0.48	0.95	0.35	0.26	0.75
$P\omega_{13}$	1	0.38	0.97	0.69	0.85	1
$P\omega_{14}$	0.95	0.69	0.96	0.19	0.07	0.12
$P\omega_{15}$	0	0	0.98	0.91	0.98	0.97
P_K	0.3125	0.4375	0.1875	0.3125	0.3750	0.0625

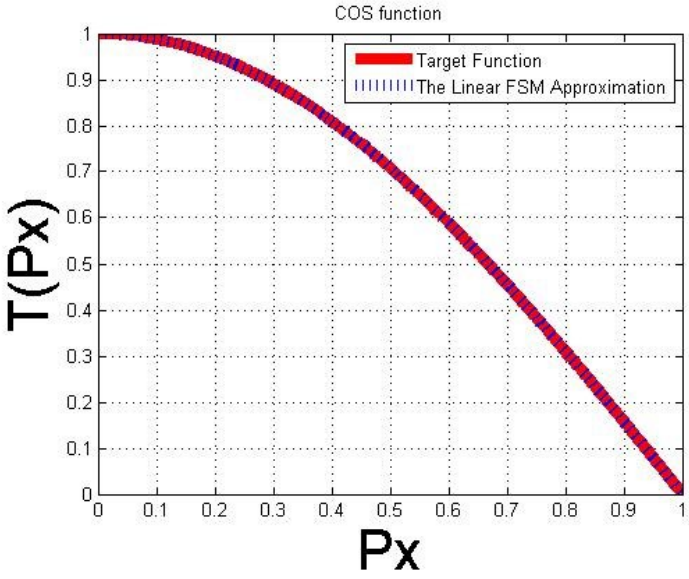
6.4 Simulation Results

The simulations are based on the state machine in Figure 6.2. Simulation results of all the functions and their comparisons to the target functions are shown in Figure 6.3.

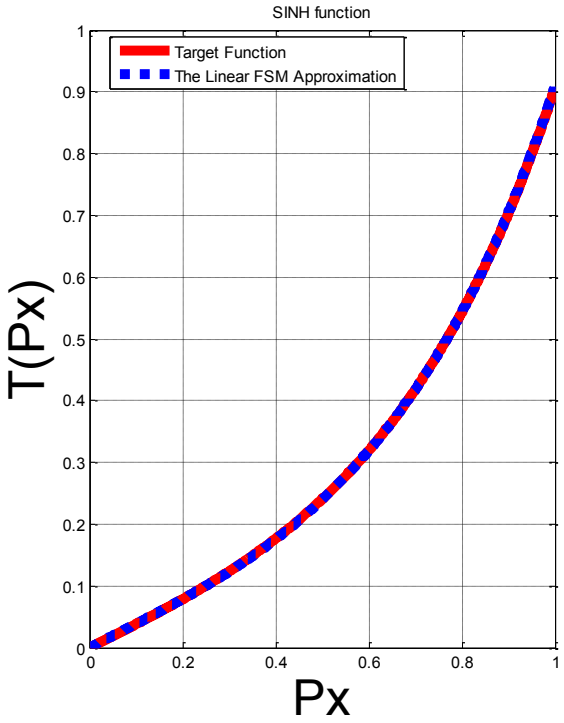
Due to the fact that the input and output must each be within the range of [0,1] in stochastic computing, we multiplied both axes by scaling factors in order to ensure the coordinate values are within the required range.


SIN function
T(Px)
Px
Target Function
The Linear FSM Approximation

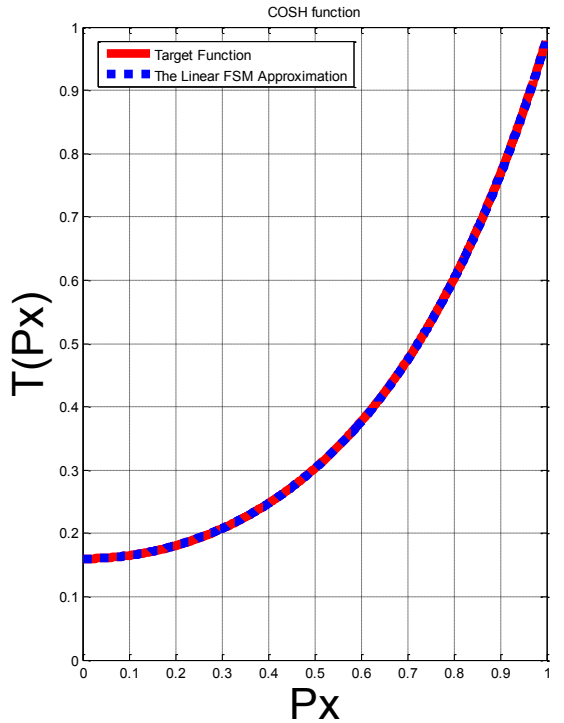
(a)


COS function
T(Px)
Px
Target Function
The Linear FSM Approximation

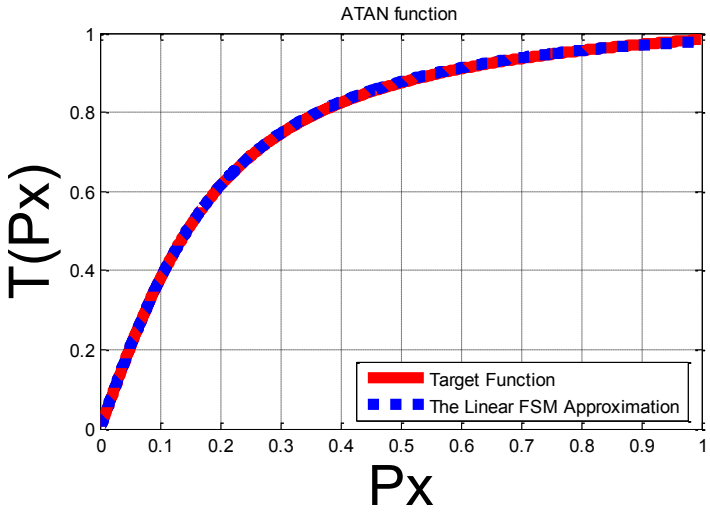
(b)


SINH function
Target Function
The Linear FSM Approximation
T(Px)
Px
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.2
0.4
0.6
0.8

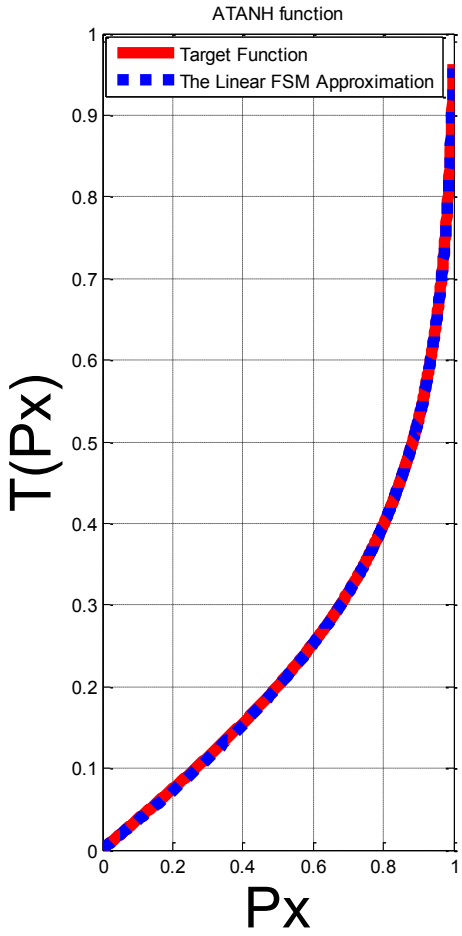
(c)


COSH function
Target Function
The Linear FSM Approximation
T(Px)
Px
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

(d)



(e)



(f)

Figure 6.3 Simulation results compared with target functions: (a) sin, (b) cos, (c) sinh, (d) cosh, (e) atan, (f) atanh

The trigonometric functions have periodicity and symmetry properties, such as even symmetric or odd symmetric, etc. We only plot curves within part of a period (for periodic functions such as sin and cos) or plot a portion of the curve for the aperiodic functions. By using different scale factors, we can determine the curves over a larger span of the horizontal axis. From the figure, it can be seen that, all of simulation curves nearly overlap with the target curves. The approximation error for the stochastic logic computation is 1.79e-11. This small error is due to the relatively large 16-state structure of the state machine, since the error is reduced as more states are used.

All of our designs have been implemented on a Xilinx Virtex-5 Xc5clx50t FPGA. The Xilinx Core Generator [61] was used to synthesize separate CORDIC cores for sin, cos, arctangent and their hyperbolic counterparts, and the stochastic design was implemented using the structure of Figure 6.1.

We have implemented all of these functions in several different data widths, namely 10 bits, 9 bits, and 8 bits. The results for delay, number of slices, maximum clock cycles, and the delay-area product are given in Tables 6.2, 6.3 and 6.4, for the cases of 10-bit data, 9-bit data and 8-bit data, respectively.

The second column in each of the tables gives the system delay, in clock cycles. The third and fourth columns show the number of FPGA slices and the maximum clock periods at which the circuits can run, respectively. The fifth column gives the product of delay and area, which is the metric used to compare circuit effectiveness. The final column gives the normalized product of delay and area expressed as a percentage, with the stochastic implementation normalized to 100%. In the majority of cases, the

stochastic computing method outperforms the CORDIC algorithm in terms of the delay-area product, particularly for the 8-bit and 9-bit data widths.

Table 6.2 Delay and area comparison for 10-bit inputs

Function	No. of clk (cycles)	Area (slices)	Clk cycle time (ns)	delay×area	Percentage
sin or cos	13	123	3.699	5914.701	124.27
sinh or cosh	13	79	3.648	3746.496	78.72
arctangent	13	103	3.646	4881.994	102.57
hyperbolic arctan	13	77	3.730	3733.73	78.45
stochastic computing	1024	4	1.162	4759.552	100

Table 6.3 Delay and area comparison for 9-bit inputs

Function	No. of clk (cycles)	Area (slices)	Clk cycle time (ns)	delay×area	Percentage
sin or cos	13	118	3.699	5674.266	238.44
sinh or cosh	13	80	3.648	3793.92	159.423
arctangent	13	107	3.646	5071.586	213.11
hyperbolic arctan	13	85	3.730	4121.65	173.19
stochastic computing	512	4	1.162	2379.776	100

Table 6.4 Delay and area comparison for 8-bit inputs

Function	No. of clk (cycles)	Area (slices)	Clk cycle time (ns)	delay×area	Percentage
sin or cos	13	112	3.699	5385.744	452.63
sinh or cosh	13	75	3.648	3556.8	298.92
arctangent	13	108	3.646	5118.984	430.21
hyperbolic arctan	13	77	3.652	3733.73	313.79
stochastic computing	256	4	1.162	1189.888	100

6.5 Soft Error Tolerance

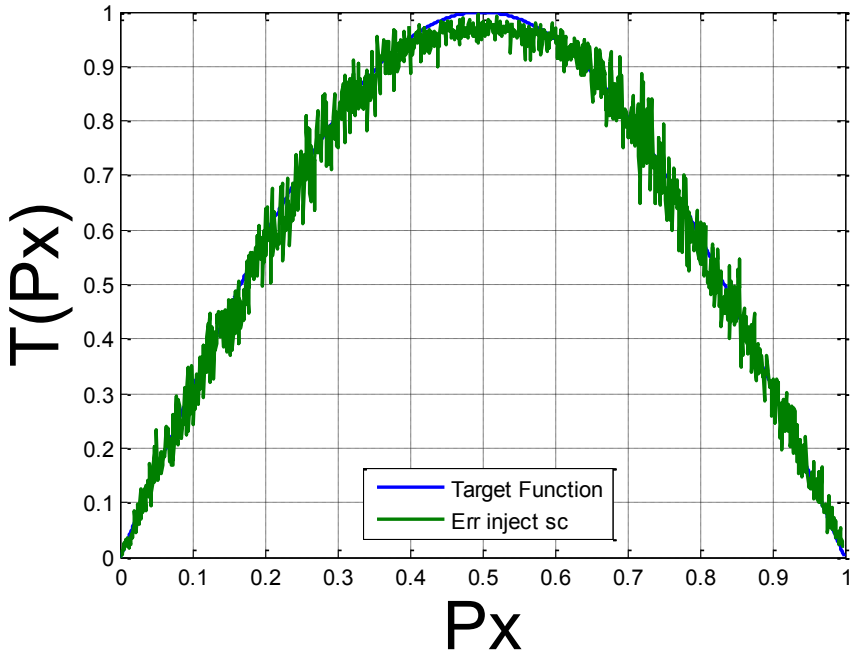
According to [63], there are three kinds of errors that are introduced by the nature of stochastic logic. They are state machine approximation error, quantization error and random fluctuation error. Among these errors, the random fluctuation error is the most significant. As mentioned earlier, soft errors are caused by bits flipping due to natural disturbances in the environment. We tested the output under various error conditions, and compared the results with those of a CORDIC implementation.

We define the soft error ratio to be the percentage of unwanted bit flipping. For example, a 5% error rate means 5% of the bits are flipped during process. For 10-bit data, in the stochastic computing case 1024 bits will represent a number, so with a 5% error rate there are 0.05*1024=51 bits are flipped to their opposite values.

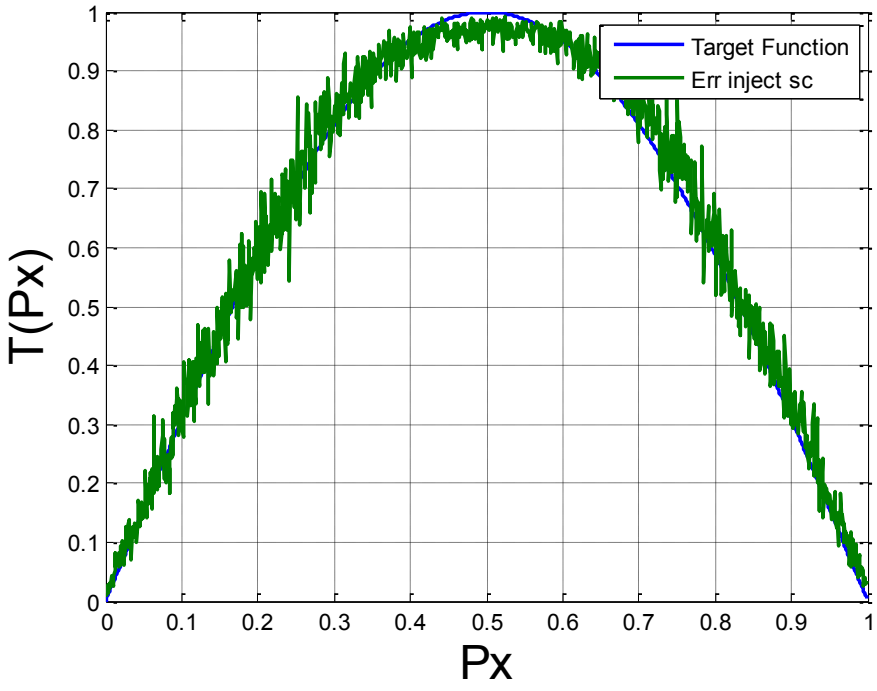
In Figure 6.4, the 10-bit stochastic computing implementation results for the sine function are compared to the ideal curves. The curves are for 0%, 0.5%, 1%, 5%, 10% and 15% soft errors, respectively. For the input on the x-axis, we uniformly obtain 1024 points from 0 to π (normalized to 0 to 1). For each input point, 1000 error trials were performed, which means each of the 1000 1024-bit numbers are randomly flipped with a certain error rate. Then, the root mean square (RMS) errors are obtained from the 1000 trials results by comparing the outputs to the desired results.

In Figure 6.4 (a), the error injection rate is 0, which mean there is no soft error injection. However, the implementation result still has a small difference with the optimal target curve. These errors are caused by the other previously mentioned error sources. It is evident from Figure 6.4 that with increasing rates of soft errors, the agreement becomes worse. However, compared to CORDIC, this deterioration is much less.

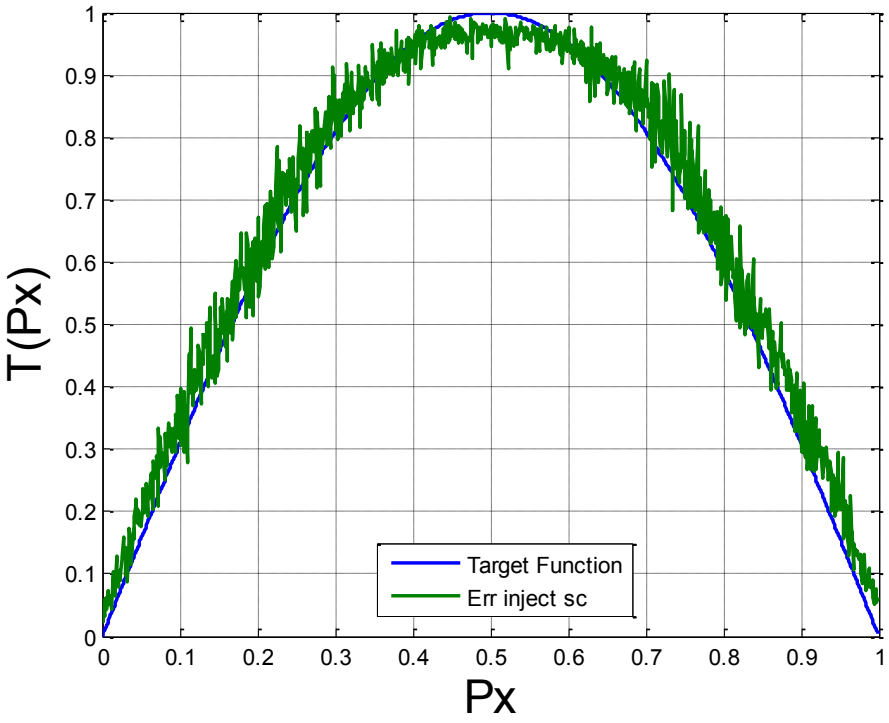
Tables 6.5, 6.6 and 6.7 present the error results for Stochastic Computing and CORDIC implementations for several different soft error rates.


T(Px)
Px
Target Function
Err inject sc
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

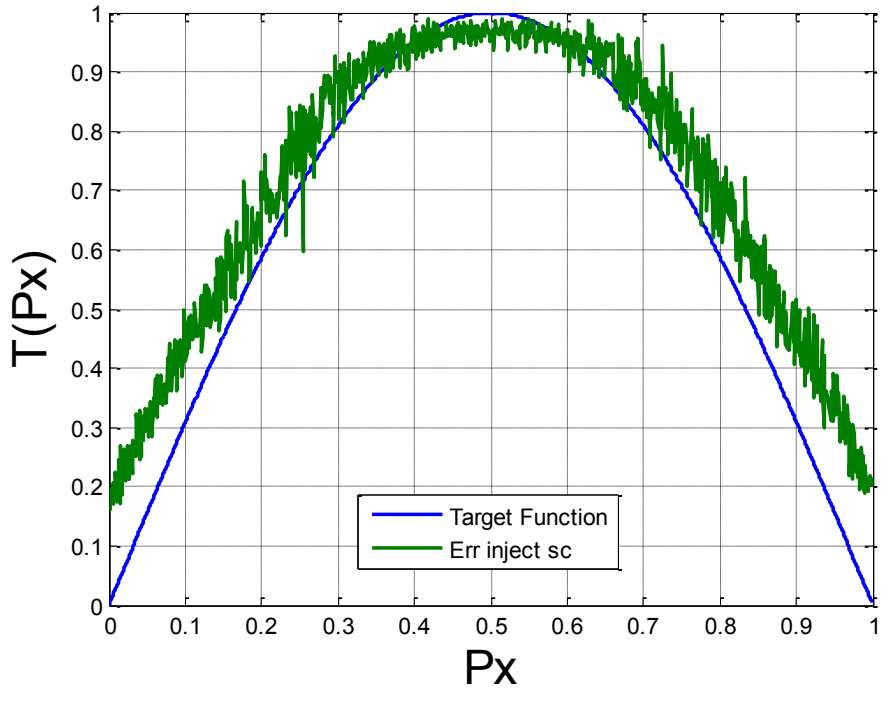
(a)


T(Px)
Px
Target Function
Err inject sc
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

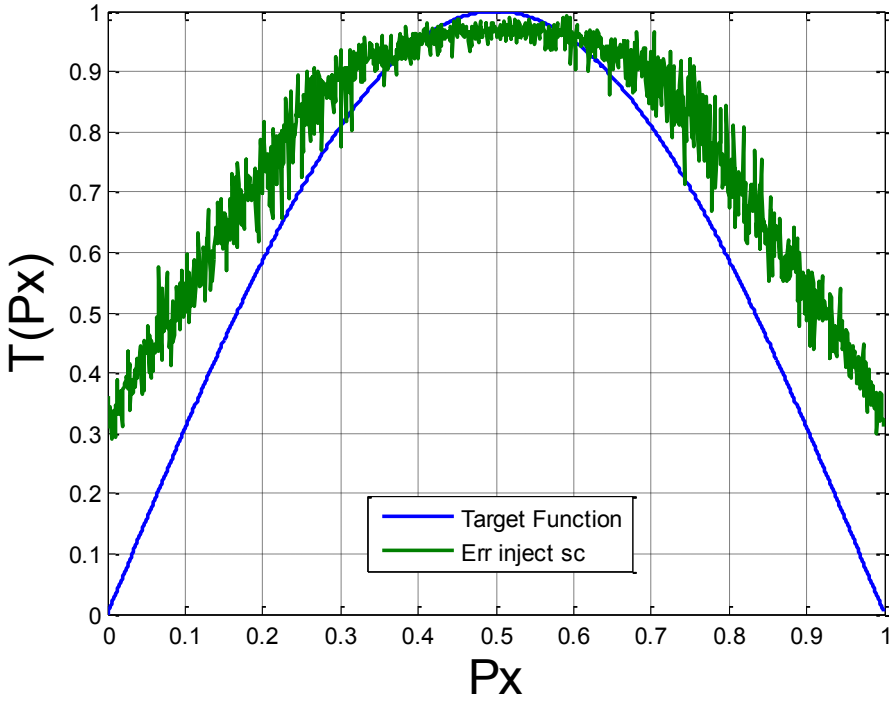
(b)


T(Px)
Px
Target Function
Err inject sc
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

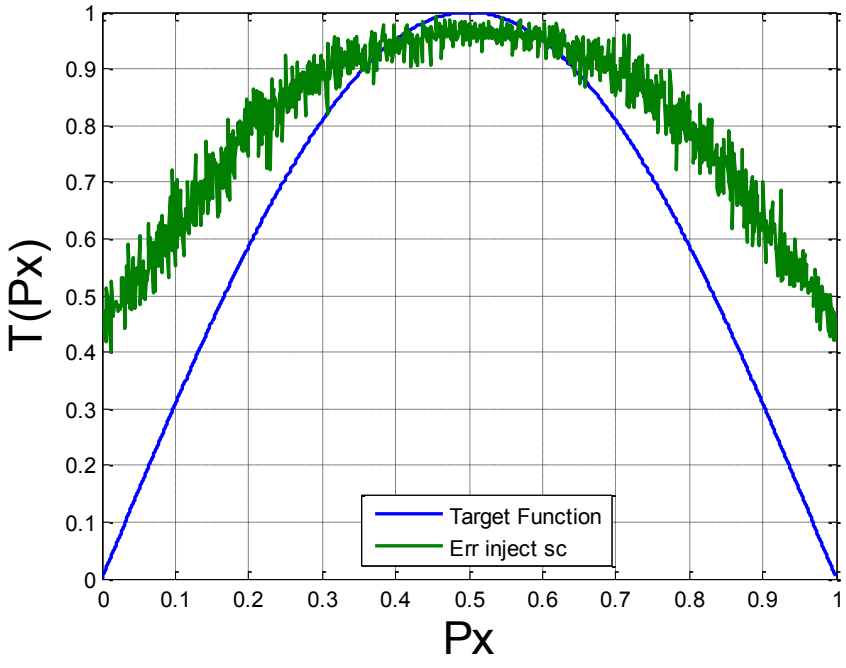
(c)


T(Px)
Px
Target Function
Err inject sc
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

(d)



(e)



(f)

Figure 6.4 Sin function implementation results compared with target function with soft error ratio (%): (a) 0, (b) 0.5, (c) 1, (d) 5, (e) 10, (f) 15

It can be seen from Tables 6.5, 6.6 and 6.7 that the stochastic computing technique has better error tolerance than CORDIC when the soft error rate is greater than 1%. Also, as the soft error increases, the error tolerance characteristics become better on a relative basis compared to CORDIC. This is because for stochastic computing, each bit has the same weight, so that the effect of any single bit flip will not significantly affect the value. On the other hand, in the case of binary weighting used in CORDIC, there is a possibility that the flipped bit is one of the most significant bits of a number. If that should occur, then the effect on the error would be large.

Table 6.5 The average output error (%) of proposed implementation and CORDIC implementation for functions with 10-bit inputs

Function	Injected Error					
	1%		2%		10%	
	S.C.	Cordic	S.C.	Cordic	S.C.	Cordic
atan	2.70	11.50	2.30	15.40	7.80	33.90
atanh	3.48	10.60	3.50	14.80	5.90	32.70
cos	2.80	12.70	2.80	8.50	6.90	32.10
sin	4.40	6.50	5.20	12.60	15.60	26.70
cosh	3.04	6.80	2.93	5.00	7.65	11.90
sinh	2.30	9.90	3.57	7.30	7.52	13.80
Average	3.12	9.70	3.38	10.60	8.56	25.20

Table 6.6 The average output error (%) of proposed implementation and CORDIC implementation for functions with 9-bit inputs

Function	Injected Error					
	1%		2%		10%	
	S.C.	Cordic	S.C.	Cordic	S.C.	Cordic
atan	3.16	4.10	3.76	10.23	5.67	17.80
atanh	4.71	8.10	4.93	12.48	8.01	30.50
cos	4.35	5.90	4.43	14.53	7.53	29.77
sin	6.68	9.62	6.88	10.38	12.12	28.77
cosh	4.16	3.69	3.67	6.89	9.35	11.17
sinh	3.81	4.33	6.89	7.42	6.35	14.72
Average	4.48	5.96	5.09	10.32	8.17	22.12

Table 6.7 The average output error (%) of proposed implementation and CORDIC implementation for functions with 8-bit inputs

Function	Injected Error					
	1%		2%		10%	
	S.C.	Cordic	S.C.	Cordic	S.C.	Cordic
atan	4.97	8.90	4.85	9.13	7.05	28.62
atanh	6.87	8.72	6.75	17.33	7.70	24.98
cos	4.91	11.28	5.74	18.78	9.23	35.03
sin	9.01	14.36	8.62	14.54	17.20	25.66
cosh	5.00	5.18	5.20	4.10	7.14	10.35
sinh	5.15	6.48	4.86	7.41	10.28	15.95
Average	5.99	9.15	6.00	11.88	9.77	23.43

6.6 Conclusion

We have applied the stochastic computing approach to efficiently implement several different trigonometric and hyperbolic functions. Our stochastic computing implementation and the CORDIC cores have all been implemented on a Virtex-5 FPGA. The results show that the proposed stochastic computing approach achieves considerably lower delay-area products than the traditional CORDIC method, especially for smaller data width applications. In addition, the results are more robust in the presence of soft errors.

Chapter 7

Conclusions

By using multiple antennas at both the transmitter and receiver sides, MIMO systems obtain diversity gain or multiplexing gain without requiring additional bandwidth. However, with the increasing requirements on the data transmission rate, reliability and implementation cost for handset devices, it is a challenging problem to design MIMO detectors that are both hardware-friendly and of high performance. This thesis has described several approaches for designing high-performance, low-complexity MIMO detection algorithms for modern communication systems.

Performance and hardware complexity are the two main metrics in the design of a MIMO detector. Previous work on implementation aspects of MIMO detection mainly focused on techniques that require low implementation complexity and hence, achieve relatively poor performance. In this thesis, we investigated efficient MIMO detection schemes that are able to approach near optimal performance.

In Chapter 2, we developed an efficient and flexible complex-valued enumeration scheme for K-best MIMO detection. The new method is based on the use of simple lookup tables, and it is fully scalable for a wide range of K-values and constellation sizes.

It has a complexity that scales sub-linearly with the constellation size and which at the same time attains performance that is close to optional.

A high performance PPTS detector for MIMO systems was proposed in Chapter 3. The bidirectional PPTS detector can effectively reduce detection errors. The error analysis between single direction and bidirectional PPTS detectors was given. Additionally, the proposed detector has a 0.1dB FER gain compared with traditional PPTS detectors.

In Chapter 4, we designed a novel bidirectional processing soft-output MIMO detection algorithm. It adopts features from several different types of fixed complexity tree search procedures. The candidates used to calculate soft likelihood information are determined in parallel from two stages of tree searching. Each stage corresponds to one direction of the path selection process. The proposed approach achieves a higher performance than previously proposed algorithms while, at the same time, having a comparable computational cost. This algorithm has a good tradeoff between performance and complexity objectives. Furthermore, its parallel and fixed throughput characteristics make it friendly for hardware implementation.

Chapter 5 introduced a novel MIMO detection algorithm with low complexity and without requiring QR decomposition. It fulfills the requirements of LTE and WiFi even in worst-case situations when the environment causes the channel information to change for all sub-carriers. The proposed algorithm only has a 0.5dB SER performance loss compared to ML detection and it is achieved with a greatly reduced computational complexity. Moreover, the algorithm is scalable to large QAM constellation sizes. After

that, a 3x3 MIMO system with a 16QAM detector architecture is designed, and the latency and hardware costs are estimated.

In Chapter 6, we introduced a stochastic computing technique, which has drawn much attention in recent years because of its efficiency in hardware cost as well as its ability to moderate errors caused by bit-flips. Specifically, we investigated the stochastic computing implementation of trigonometric and hyperbolic functions, of which the former may be applied in QR decomposition. Using only simple registers and multiplexers, the state machine architecture of stochastic computing was adapted to the computation of various trigonometric and hyperbolic functions. An FPGA implementation of trigonometric and hyperbolic functions using the stochastic computing was compared to the well-known CORDIC approach. Both designs were synthesized and implemented on a Xilinx Virtex-5 FPGA. The proposed method has advantages over the traditional CORDIC algorithm in terms of delay and area for various input data widths. Moreover, the same circuit can be used for all functions with only a change in coefficient values. Furthermore, it is more tolerant of soft errors (bit flips) than a CORDIC implementation. This fault tolerance advantage becomes more significant as the number of soft errors increases. Future work in this area may include the combination of stochastic computing with MIMO detection.

Bibliography

[1] S.N. Diggavi, N. Al-Dhahir, A. Stamoulis and A.R. Calderbank, "Great expectations: the value of spatial diversity in wireless networks," *Proc. of IEEE* vol. 92, no. 2, pp. 219-270, February 2004.

[2] D. Gesbert, M. Shafi, D.-S. Shiu; P.J. Smith and A. Naguib, "From theory to practice: an overview of MIMO space-time coded wireless systems," *IEEE J. Select Area Comm.*, vol. 21, issue 3, pp. 281-302, April 2003.

[3] A.J. Paulraj, D.A. Gore, R.U. Nabar and H. Bolcskei, "An overview of MIMO communications - a key to gigabit wireless," Proc. of IEEE vol. 92, no. 2, pp. 198 – 218, February 2004.

[4] X. Li, A. Gani, R. Salleh, and O. Zakaria, "The future of mobile wireless communication networks," Proc. International Conference on Communication Software and Networks (ICCSN), pp. 554-557, 2009.

[5] G. J. Foschini, Jr., "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," Bell Labs Tech. Journal, vol. 1, pp. 41-59, Autumn 1996.

[6] M. Sellathurai and S. Haykin, “Turbo-BLAST for wireless communicaitons: Theory and experiments,” IEEE Trans. on Signal Processing, vol. 50, pp. 2538-2545, October 2002.

[7] G. J. Foschini, D. Chizhik, M. J. Gans, C. Papadias, and R. A. Valenzuela, “Analysis and performance of some basic space time architectures,” IEEE Journal on Selected Areas in Communications, vol. 21, pp. 303-320, April 2003.

[8] S. M. Alamouti, “A simple trasmit diversity technique for wireless communications,” IEEE Journal on Selected Areas in Communications, vol. 16, pp. 1451-1458, October 1998.

[9] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, “Space-time block codes from orthogonal designs,” IEEE Trans. on Information Theory, vol. 45, pp. 1456-1467, July 1999.

[10] V. Tarokh, A. Naguib, N. Seshadri, and A. R. Calderbank, “Combined array processing and space-time coding,” IEEE Trans. on Information Theory, vol. 47, pp. 199-207, Feb. 1999.

[11] B. Badic, A. Linduska and H. Weinrichter, “Impact of spatial channel correlation on super quasi-orthogonal space-time trellis codes,” IEEE 6th Workshop on Signal Processing Advances in Wireless Communications, pp. 91-95, 2005.

[12] G. Wang, J.K Zhang and M. Amin, “Space-Time Block Code Designs Based on Quadratic Field Extension for Two-Transmitter Antennas,” IEEE Trans. on Information Theory, Vol. 58, Issue 6, pp. 4005-4013, 2012.

[13] B. Rani and G. Saini, “PAPR reduction using Cooperative PTS for space frequency block code MIMO-OFDM signal,” IEEE Int. Conf. on Parallel Distributed and Grid Computing (PDGC), pp. 132-134, 2012.

[14] W. Liu and L. Wang, “BER-Minimized Space-Time-Frequency Codes for MIMO Highly Frequency-Selective Block-Fading Channels,” IEEE Int. Conf. on Communications, pp. 5113-5117, 2008.

[15] N. Ekanayake, “Equal-gain combining diversity reception of M-ary cpsk signals in nakagami fading,” IEEE Communications Letters, Vol. 14, Issue 4, pp. 285-287, 2010.

[16] X. Zhao and A.H. Sayed, “Single-link diffusion strategies over adaptive networks,” IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 3749-3752, 2012.

[17] M.M. Saad, K.A. Abd Rashid, M. Senon and J.S. Hamidon, “A review on compact slot antenna for wireless MIMO communication system,” Int. Symp. on Technology Management and Emerging Technologies (ISTMET), pp. 344-348, 2014.

[18] M.A. Jensen and J.W. Wallace, “A review of antennas and propagation for MIMO wireless communications,” IEEE Trans. on Antennas and Propagation, Vol. 52, Issue 11, pp. 2810-2824, 2004.

[19] C. Leroux, I. Tal, A. Vardy, and W. J. Gross, "Hardware architectures for successive cancellation decoding of polar codes," in Proc. IEEE International

Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1665-1668, 2011.

[20] C. Studer, S. Fateh, and D. Seethaler, "ASIC Implementation of Soft-Input Soft-Output MIMO Detection Using Parallel Interference Cancellation," IEEE Journal of Solid-State Circuits, Vol. 46, No. 7, pp. 1754–1765, July 2011.

[21] P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela, V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel. In Proc. of URSI ISSSE, pp. 295–300, 1998.

[22] M. Shabany and P. G. Gulak, "A 675 Mbps, 44 64-QAM K-Best MIMO Detector in 0.13umCMOS", IEEE Trans. on Very Large Scale Integration (VLSI) System. pp. 135-147, Jan 2012.

[23] M. Mahdavi and M. Shabany, "Novel MIMO Detection Algorithm for High-Order Constellations in the Complex Domain", IEEE Trans. on Very Large Scale Integration (VLSI) System, pp.1-4, May 2012.

[24] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," IEEE J. Sel. Areas Commun.., vol. 24, no. 3, pp. 491–503, Mar. 2006.

[25] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "K-best MIMO detection VLSI architectures achieving up to 424 Mbps," in *Proc. IEEE Int. Symp. Circuits Syst*, pp. 1151–1154. May. 2006

[26] Y. Sun and J.R. Cavallaro, "Trellis-Search Based Soft-Input Soft-Output MIMO Detector: Algorithm and VLSI Architecture," IEEE Trans. on Signal Processing, vol.60, no.5, pp.2617-2627, May 2012.

[27] M. Siti and M. Fitz, “A Novel Soft-output Layered Orthogonal Lattice Detector for Multiple Antenna Communications,” IEEE Int. Conf. on Communications, Vol. 4, pp. 1686 –1691, June 2006.

[28] C. Siriteanu, Y. Miyanaga, S.D. Blostein, S. Kuriki and X. Shi, “MIMO Zero-Forcing Detection Analysis for Correlated and Estimated Rician Fading,” IEEE Trans. on Vehicular Technology, Vol. 61, Issue 7, pp. 3087-3099, 2012.

[29] A. Salari, S.M. Fakhraie and A. Abbasfar, “Algorithm and FPGA implementation of interpolation-based soft output mmse mimo detector for 3GPP LTE,” IET Communications, Vol. 8, Issue 4, pp. 492-499, 2014.

[30] Y. Yang, “A Review of Sphere Decoding for MIMO Systems and Its Improvement,” Int. Conf. on Wireless Communications Networking and Mobile Computing (WiCOM), pp. 1-4, Sept. 2010.

[31] Y. Sun; J.R. Cavallaro, “High-Throughput Soft-Output MIMO Detector Based on Path-Preserving Trellis-Search Algorithm,” IEEE Trans. on Very Large Scale Integration (VLSI) Systems, vol.20, no.7, pp.1235-1247, July 2012.

[32] A.F. Naguib, V.Tarokh, N. Seshadri and A.R. Calderbank, “A space-time coding modem for high-data-rate wireless communications,” IEEE J. on Selected Areas in Communications, vol. 16, no. 8, pp. 1459-1478, Oct 1998.

[33] P. Salmela, A. Burian, H. Sorokin, and J. Takala, “Complex-valued QR decomposition implementation for MIMO receivers,” Proc. Int. Conf. Acoustics, Speech, and Signal Process., pp. 1433–1436, 2008.

[34] D. Patel, V. Smolyakov, M. Shabany, and P. Gulak, “Vlsi implementation of a wimax/lte compliant low-complexity high-throughput soft-output kbest mimo detector,” IEEE Proc. International Symposium on Circuits and Systems (ISCAS), pp. 593 –596, June 2010.

[35] L. Barbero, T. Ratnarajah, and C. Cowan, “A low-complexity soft-mimo detector based on the fixed-complexity sphere decoder,” IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 2669 –2672. April 2008.

[36] C. P. Schnorr and M. Euchner, “Lattice basis reduction: Improved practical algorithms and solving subset sum problems,” Math. Programming, vol. 66, pp. 181–191, Sept. 1994.

[37] M. Shabany, K. Su, and P. G. Gulak, “A pipelined high-throughput implementation of near-optimal complex K-Best lattice decoders,” Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Proc., pp. 3173–3176, 2008

[38] M. Mahdavi and M. Shabany, “A Modified Complex K-Best Scheme for High-Speed Hard-Output MIMO Detectors,” IEEE Int. Mid. Symp. Circuits and Systems (MWSCAS), pp. 845-848, Aug., 2010.

[39] M.O. Damen, H.E. Gamal, and G. Caire, “On maximum-likelihood detection and the search for the closest lattice point,” IEEE Trans. Inf. Theory, vol. 49, no. 10, pp. 2389–2402, Oct. 2003.

[40] H. Vikalo and B. Hassibi, “On the sphere-decoding algorithm II. Generalizations, second-order statistics, and applications to communications,” IEEE Trans. on Signal Process., vol. 53, no. 8-1, pp. 2819–2834, Aug. 2005.

[41] B. Hassibi and H. Vikalo, “On the sphere-decoding algorithm I. Expected complexity,” IEEE Trans. on Signal Process., vol. 53, no. 8-1, pp. 2806–2818, Aug. 2005.

[42] Q. Li and Z. Wang, “Improved K-best sphere decoding algorithms for MIMO systems,” Proc. IEEE Int. Symp. Circuits Syst., pp. 1159–1162, 2006.

[43] S. Chen and T. Zhang, “Low power soft-output signal detector design for wireless mimo communication systems,” ACM/IEEE Int. Symp. on Low Power Electronicsand Design (ISLPED), pp. 232 –237, Aug. 2007.

[44] L. Barbero and J. Thompson, “Extending a fixed-complexity sphere decoder to obtain likelihood information for turbo-mimo systems,” IEEE Trans. on Vehicular Technology, vol. 57, no. 5, pp. 2804 –2814, Sept. 2008.

[45] P. Marsch, E. Zimmermann, and G. Fettweis, “Smart candidate adding: A new low-complexity approach towards near-capacity,” European Signal Processing Conference (EUSIPCO05) on MIMO Detection, 2005.

[46] D. Milliner, E. Zimmermann, J. Barry, and G. Fettweis, “A fixed complexity smart candidate adding algorithm for soft-output mimo detection,” IEEE J. of Selected Topics in Signal Processing, , vol. 3, no. 6, pp. 1016 –1025, Dec. 2009.

[47] L. Liu, X. Ma, F. Ye, and J. Ren, “Design of highly-parallel, 2.2gbps throughput signal detector for MIMO systems,” IEEE Int. Conf. on Communications, pp. 742 –745, May 2008.

[48] H.-S. Kim, J. Lee, and S.-C. Park, “Complexity evaluation for MIMO sphere decoder with various tree-searching algorithms,” Int. Conf. on Communication Technology (ICCT ’06), pp. 1–4, Nov. 2006.

[49] Y. Teng, K. Naito, K. Mori, and H. Kobayashi, “Complexity reduced maximum likelihood detection for SDM-OFDM system”, Int. Conf. on Wireless Networks, Commun. and Mobile Computing, Vol. 1, pp. 256-261, Jun. 2005.

[50] T. Onizawa, D. Uchida, W. Jiang, T. Sugiyama, T. Fujita and A. Ohta, “Limiting the constellations subjected to ML detection in OFDM/SDM systems”, Vehicular Technology Conference(VTC), Vol. 5, pp. 3368-3372, Fall 2004.

[51] R.S.Yazdi and T. Kwasniewski, “A Low Complexity VLSI Architecture for MIMO Sphere Decoding Algorithm”, IEEE North-East Workshop on Circuits and Systems and TAISA (NEWCAS-TAISA), pp. 1-4, Jul. 2009.

[52] L.G. Barbero, and J. S. Thompson, “Fixing the Complexity of the Sphere Decoder for MIMO Detection”, IEEE Trans. on Wireless Commun., Vol. 7, no. 6, pp. 2131–2142, Jun. 2008.

[53] N.A. Chisty, “Matrix Inversion Using QR Decomposition by Parabolic Synthesis,” M.S. thesis, Abbrev. Dept. of Elec. and Info. Tech., Lund Univ., Lund, Sweden, 2012.

[54] D. Patel, M. Shabany and P.G. Gulak, "A low-complexity high-speed QR decomposition implementation for MIMO receivers," IEEE Int. Symp. on Circuits and Systems(ISCAS), pp. 33-36, May 2009.

[55] K.H. Lin, R.C. Chang, C.L. Huang, F.C. Chen and S.C Lin, "Implementation of QR decomposition for MIMO-OFDM detection systems," IEEE Int. Conf. on Elec., Cir. and Sys., pp. 57-60, Aug.-Sept. 2008.

[56] P. T. P. Tang, "Table-lookup algorithms for elementary functions and their error analysis", IEEE Symp. On Comp. Arithmetic, Grenoble, France, pp. 232- 236, Jun. 1991.

[57] P. K. Meher, J. Valls, T.B. Juang, K. Sridharan, and K. Maharatna, "50 Years of CORDIC: Algorithms, architectures, and applications", IEEE Trans. on Circuits and Syst., Vol. 56, no. 9, pp. 1893-1907, Sept. 2009.

[58] J. E. Volder, "The CORDIC trigonometric computing technique", IRE Trans. on Electronic Computers, Vol. EC-8, no. 3, pp. 330–334, Sept. 1959.

[59] P. Li, W. Qian, D. J. Lilja, "A stochastic reconfigurable architecture for fault tolerant computation with sequential logic", IEEE Int. Conf. on Computer Design, Montreal, QC, Canada, pp. 303-308, Sept.-Oct. 2012.

[60] G. Sarkis, and, W. J. Gross, "Efficient stochastic decoding of non-binary LDPC codes with degree-two variable nodes", IEEE Commun. Letters, Vol. 16, no. 3, pp. 389-391, Mar. 2012.

[61] Xilinx Corp., “LogiCORE IP CORDIC v4.0 Product Specification”, 2011. Available:http://www.xilinx.com/support/documentation/ip_documentation/cordic_ds249.pdf

[62] P. Li, D.J. Lilja, W. Qian, K. Bazargan, and M. Riedel, “The synthesis of complex arithmetic computation on stochastic bit streams using sequential logic”, IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD), pp. 480–487, 2012.

[63] W. Qian, X. Li, M. Riedel and K. Bazargan, “An Architecture for Fault-Tolerant Computation with Stochastic Logic”, IEEE Trans. on Computers, Vol. 60, no. 1, pp.93-105, Jan. 2011.