Hindawi Publishing Corporation
EURASIP Journal on Applied Signal Processing
Volume 2006, Article ID 97676, Pages 1–14
DOI 10.1155/ASP/2006/97676

A Fast Sphere Decoding Algorithm for Space-Frequency Block Codes

Zoltan Safar,[1, 2] Weifeng Su,[1, 3] and K. J. Ray Liu[1]

[1] *Department of Electrical and Computer Engineering, A. James Clark School of Engineering, and Institute for Systems Research, University of Maryland, College Park, MD 20742, USA*
[2] *Modem System Design, Technology Platforms, Nokia, Copenhagen 1790, Denmark*
[3] *Department of Electrical Engineering, School of Engineering and Applied Sciences, State University of New York, Buffalo, NY 14260, USA*

Received 10 November 2004; Revised 8 August 2005; Accepted 22 August 2005

The recently proposed space-frequency-coded MIMO-OFDM systems have promised considerable performance improvement over single-antenna systems. However, in order to make multiantenna OFDM systems an attractive choice for practical applications, implementation issues such as decoding complexity must be addressed successfully. In this paper, we propose a computationally efficient decoding algorithm for space-frequency block codes. The central part of the algorithm is a modulation-independent sphere decoding framework formulated in the complex domain. We develop three decoding approaches: a modulation-independent approach applicable to any memoryless modulation method, a QAM-specific and a PSK-specific fast decoding algorithm performing nearest-neighbor signal point search. The computational complexity of the algorithms is investigated via both analysis and simulation. The simulation results demonstrate that the proposed algorithm can significantly reduce the decoding complexity. We observe up to 75% reduction in the required FLOP count per code block compared to previously existing methods without noticeable performance degradation.

1. INTRODUCTION

Multiple-input multiple-output (MIMO) systems employing multiple transmit and receive antennas have the potential to play a significant role in the development of future wireless communication systems. By exploiting the larger number of propagation paths between the transmitter and the receiver, the adverse effects of the channel fading can be significantly reduced. To take advantage of both the MIMO systems and the OFDM modulation, MIMO-OFDM systems have been proposed recently [1–6], where space-frequency (SF) coding is applied as the channel code. However, before such a system becomes an attractive choice for practical applications, implementation issues such as decoding complexity must be addressed successfully.

Computationally efficient decoding algorithms have only been proposed for decoding space-time (ST) block codes in quasistatic, flat (frequency nonselective) fading environment [7, 8]. For ST block codes transmitted over temporally evolving channels and for SF block-coded MIMO-OFDM systems, where the channel changes along the frequency axis, low-complexity decoding algorithms still do not exist in the literature.

The sphere decoding algorithm was introduced in [9] assuming a single-antenna, real-valued fading channel model. Later results [10–13] (and the references therein) generalized the algorithm to complex-valued MIMO channels. A reduced complexity algorithm was proposed in [10], where the signal coordinates were sorted according to their partial metrics and explored in this order. The authors of [11] achieved complexity reduction by exploring the signal coordinates in a zig-zag order. This approach was further refined and improved in [12]. In [13], the sphere decoding algorithm was applied to equalize frequency-selective MIMO channels. All of these works considered uncoded MIMO systems and assumed quasistatic, flat fading channels. Moreover, they formulated the sphere decoding problem in the real domain, so the resulting algorithms can only be used with modulation methods that can be decomposed into the product of two real constellations (e.g., square QAM).

A complex-domain sphere decoding algorithm was described in [14]. This work considered iterative (turbo) decoding in a MIMO system where linear ST mapping was combined with an outer channel code, and a sphere detector was used to approximate the log-likelihood ratio in a computationally efficient way. This approach was specific to modulation methods that can be decomposed into PSK constellations, and its objective was to identify a set of candidate solutions, as opposed to finding the best candidate solution with the maximum possible efficiency.

Most previous works considered frequency-flat quasistatic MIMO channels, where the channel stays constant over many code blocks. As a consequence, they assumed that most of the decoding complexity comes from the searching stage of the sphere decoding algorithm, and the complexity of the preprocessing stage, which calculates the Cholesky or QR decomposition of the squared channel matrix, is negligible. However, in case of MIMO-OFDM systems transmitting over frequency-selective channels, the channel changes in the frequency domain, so the Cholesky or QR decomposition of the squared channel matrix has to be calculated for each code block. Therefore, when optimizing the decoder performance, the decoding complexity of both the preprocessing stage and the searching stage has to be taken into account.

In this paper, we propose a fast algorithm for decoding SF block codes constructed from orthogonal and quasi-orthogonal designs based on the idea of sphere decoding. We formulate the decoding problem in the complex domain, which allows us to fully exploit the distance structure of complex signal constellations. We propose a modulation-independent sphere decoding framework by interpreting the sphere decoding algorithm as a greedy, constrained depth-first search algorithm. Due to the modular structure of the framework, it can be used to construct a decoding algorithm that can be used with any memoryless modulation, and it can also be tailored to a particular modulation method. The latter possibility will be explored by developing QAM-specific and PSK-specific decoding algorithms that take advantage of the special structure of the constellations by performing nearest-neighbor signal point search.

When comparing the computational complexity of different approaches, we will only consider algorithmic complexity. That is, we will not take advantage of any specific implementation-dependent or architecture-dependent optimization. The motivation behind this is that we would like to preserve the generality of our results. Given a specific architecture or optimization technique, the analysis can be easily modified based on the results provided in this paper. This is in contrast, for example, with the results reported in [15–18], where different parallel VLSI architectures have been proposed for the implementation of the sphere decoding algorithm. Exploiting parallelism in case of a specific signal processing architecture such as pipelining or a systolic array can further decrease the decoding delay, but it would make the results applicable only to that particular architecture.

The paper is organized as follows. Section 2 will introduce the system model and the notation. Section 3 will describe the equivalent representation of the received signal used in this paper. The proposed decoding framework will be explained in Section 4, and the proposed fast signal search method will be described in Section 5. Section 6 will provide the simulation results, and some conclusions will be drawn in the last section.

2. SYSTEM MODEL AND NOTATION

Consider an SF-coded MIMO-OFDM system having K transmit antennas, L receive antennas, and S subcarriers, with S being a multiple of the code block length G. Suppose that the frequency-selective fading channels between each pair of transmit and receive antennas have P independent delay paths and the same power delay profile. We consider a medium-mobility scenario, where the MIMO channel changes from OFDM block to OFDM block, but it can be assumed to be constant over one OFDM block period. During each OFDM block period, the channel impulse response from transmit antenna k to receive antenna l at time delay τ is modeled as

$$h_{k,l}(\tau) = \sum_{p=0}^{P-1} \beta_{k,l}(p)\delta(\tau - \tau_p), \tag{1}$$

where τ_p is the delay and $\beta_{k,l}(p)$ is the complex amplitude of the pth path between transmit antenna k and receive antenna l. The $\beta_{k,l}(p)$'s are zero-mean, complex Gaussian random variables with variances $E[|\beta_{k,l}(p)|^2] = \delta_p^2$. The powers of the P paths are normalized such that $\sum_{p=0}^{P-1} \delta_p^2 = 1$. The frequency response of the channel is given by

$$H_{k,l}(f) = \sum_{p=0}^{P-1} \beta_{k,l}(p)e^{-j2\pi f \tau_p}. \tag{2}$$

We assume that the MIMO channel is spatially uncorrelated, that is, the $\beta_{k,l}(p)$'s are independent for different indices (k,l).

The input bit stream is divided into b bit long segments, creating B-ary ($B = 2^b$) source symbols. The encoder forms S/G source symbol blocks, each containing N source symbols. Source symbol $s_i \in \{0,1,\dots,B-1\}$, $i = 0,1,\dots,N-1$, is mapped onto a complex channel symbol (or constellation point) x_i according to $x_i = \Omega(s_i)$, where the function $\Omega(\cdot)$ represents the modulation operation. The average energy of the constellation will be denoted by E_{avg}. Then, the SF encoder forms two-dimensional codewords from the channel symbols. The SF codeword corresponding to the tth ($t = 0,1,\dots,S/G-1$) source symbol block can be expressed as a G by K matrix $\mathbf{C}$:

$$\mathbf{C} = \begin{bmatrix} c_0[Gt] & c_1[Gt] & \cdots & c_{K-1}[Gt] \\ c_0[Gt+1] & c_1[Gt+1] & \cdots & c_{K-1}[Gt+1] \\ \vdots & \vdots & \ddots & \vdots \\ c_0[Gt+G-1] & c_1[Gt+G-1] & \cdots & c_{K-1}[Gt+G-1] \end{bmatrix}, \tag{3}$$

where $c_k[i]$ denotes the channel symbol transmitted over the ith subcarrier by transmit antenna k. The symbol rate of the code is N/G.

At the receiver side, after matched filtering, removing the cyclic prefix, and applying FFT, the received signal corresponding to the tth source symbol block at subcarrier

$Gt+g$ $(g=0,1,\ldots,G-1)$ and receive antenna l is given by

$$y_l[Gt+g]=\sum_{k=0}^{K-1}H_{k,l}[Gt+g]c_k[Gt+g]+n_l[Gt+g], \quad (4)$$

where $H_{k,l}[i]=H_{k,l}(i\Delta f)$ is the channel frequency response at the ith subcarrier between transmit antenna k and receive antenna l, $\Delta f=1/T$ is the subcarrier separation in the frequency domain, and T is the OFDM symbol period. We assume that the channel state information $H_{k,l}[i]$ is known at the receiver, but not at the transmitter. In (4), $n_l[i]$ denotes the complex, zero-mean, additive white Gaussian noise component at the ith subcarrier at receive antenna l. The variance of the noise samples is assumed to be $1/(\rho\gamma)$, where the scaling factor γ is defined as $\gamma=b/(KGE_{\text{avg}})$, so ρ is the signal to noise ratio per bit at each subcarrier at each receive antenna. In the sequel, we will focus our attention on decoding a single code block, so a simplified notation will be used by dropping the block index t:

$$y_l[g]=\sum_{k=0}^{K-1}H_{k,l}[g]c_k[g]+n_l[g], \quad (5)$$

for $g=0,1,\ldots,G-1$.

3. EQUIVALENT REPRESENTATION

In general, SF coding introduces spatial and frequency-domain dependence among the code symbols $c_k[i]$ within a code block $\mathbf{C}$. For example, in case of the 2×2 orthogonal design [7]:

$$\mathbf{C}=\begin{bmatrix} x_0 & x_1\\ -x_1^* & x_0^*\end{bmatrix}, \quad (6)$$

the channel symbols transmitted from different transmit antennas and through different subcarriers are clearly related. Note that in this case, the channel is not quasistatic, so the efficient decoding methods described in [7, 8] cannot be applied. To be able to use a sphere decoder (to be able to make sequential decisions on the sent signal coordinates), it is necessary to transform the received signal to an equivalent signal representation, where the coordinates of the sent signal vector are independent. We have constructed the SF block codes from orthogonal and quasi-orthogonal designs, as the signal transformation can be carried out easily in this case. We consider both full-rate SF codes and full-diversity SF codes for 2, 3, and 4 transmit antennas.

3.1. *Full-rate SF codes: two transmit antennas*

For 2 transmit antennas $(K=2)$, the 2×2 orthogonal design (6) with symbol rate 1 $(N=2, G=2)$ is used to construct SF codes. From (5), the received signal at receive antenna l can be expressed as

$$\begin{aligned} y_l[0]&=H_{0,l}[0]x_0+H_{1,l}[0]x_1+n_l[0],\\ y_l[1]&=-H_{0,l}[1]x_1^*+H_{1,l}[1]x_0^*+n_l[1].\end{aligned} \quad (7)$$

By taking the complex conjugate of the second line of (7), the transformed equivalent received signal vector $\mathbf{y}_l=[y_l[0], y_l^*[1]]^T$ for receive antenna l can be rewritten in matrix-vector form as

$$\mathbf{y}_l=\mathbf{H}_l\mathbf{x}+\mathbf{n}_l, \quad (8)$$

where $\mathbf{x}=[x_0,x_1]^T$ is the $N\times1$ transmitted channel symbol vector, $\mathbf{n}_l=[n_l[0],n_l^*[1]]^T$ is the equivalent noise component, and $\mathbf{H}_l$ is defined as

$$\mathbf{H}_l=\begin{bmatrix} H_{0,l}[0] & H_{1,l}[0]\\ H_{1,l}^*[1] & -H_{0,l}^*[1]\end{bmatrix}. \quad (9)$$

By collecting the received signal and noise components corresponding to different receive antennas in $KL\times1$ vectors as $\mathbf{y}=[\mathbf{y}_0^T,\ldots,\mathbf{y}_{L-1}^T]^T$ and $\mathbf{n}=[\mathbf{n}_0^T,\ldots,\mathbf{n}_{L-1}^T]^T$, the equivalent received signal can be expressed as

$$\mathbf{y}=\mathbf{H}\mathbf{x}+\mathbf{n}, \quad (10)$$

where the $KL\times N$ matrix $\mathbf{H}$ is the equivalent channel matrix defined as

$$\mathbf{H}=\begin{bmatrix}\mathbf{H}_0\\ \vdots\\ \mathbf{H}_{L-1}\end{bmatrix}. \quad (11)$$

Note that the above described equivalent representation has the following properties that are important from the viewpoint of the sphere decoding algorithm. First, the coordinates of the noise vector $\mathbf{n}$ are independent, zero mean, complex Gaussian random variables with variance $1/(\rho\gamma)$. Second, the coordinates of the $\mathbf{x}$ vector are independent. Third, the matrix $\mathbf{H}$ has at least as many rows as columns, independently of the number of receive antennas. Fourth, since the entries in the matrix $\mathbf{H}$ are complex, zero mean, Gaussian random variables and we have assumed that the MIMO channel is spatially uncorrelated, the matrix $\mathbf{H}$ has full rank with very high probability.

3.2. *Full-rate SF codes: four transmit antennas*

In case of 4 transmit antennas $(K=4)$, we have adopted the quasi-orthogonal design [19]

$$\mathbf{C}=\begin{bmatrix} x_0 & x_1 & x_2e^{j\phi} & x_3e^{j\phi}\\ -x_1^* & x_0^* & -x_3^*e^{-j\phi} & x_2^*e^{-j\phi}\\ x_2e^{j\phi} & x_3e^{j\phi} & x_0 & x_1\\ -x_3^*e^{-j\phi} & x_2^*e^{-j\phi} & -x_1^* & x_0^*\end{bmatrix}, \quad (12)$$

which provides a symbol rate 1 $(N=4, G=4)$. In (12), the channel symbols x_i are taken from the same constellation, and the rotation angle ϕ is chosen in such a way that it can ensure the full rank of the code difference matrix for any two distinct code matrices. For example, if the x_i's are taken from a square QAM constellation, the best rotation angle is $\phi=\pi/4$. Proceeding similarly to Section 3.1, the equivalent received signal vector for receive antenna l can be obtained as $\mathbf{y}_l = [y_l[0], y_l^*[1], y_l[2], y_l^*[3]]^T$, and the equivalent noise vector becomes $\mathbf{n}_l = [n_l[0], n_l^*[1], n_l[2], n_l^*[3]]^T$. The result is the matrix equation $\mathbf{y}_l = \mathbf{H}_l\mathbf{x}+\mathbf{n}_l$, with $\mathbf{x} = [x_0, x_1, x_2, x_3]^T$, and the matrix $\mathbf{H}_l$ defined as

$$\mathbf{H}_l = \begin{bmatrix} H_{0,l}[0] & H_{1,l}[0] & H_{2,l}[0]e^{j\phi} & H_{3,l}[0]e^{j\phi} \\ H_{1,l}^*[1] & -H_{0,l}^*[1] & H_{3,l}^*[1]e^{j\phi} & -H_{2,l}^*[1]e^{j\phi} \\ H_{2,l}[2] & H_{3,l}[2] & H_{0,l}[2]e^{j\phi} & H_{1,l}[2]e^{j\phi} \\ H_{3,l}^*[3] & -H_{2,l}^*[3] & H_{1,l}^*[3]e^{j\phi} & -H_{0,l}^*[3]e^{j\phi} \end{bmatrix}. \quad (13)$$

The $\mathbf{y}$ and $\mathbf{n}$ vectors are formed similarly to the two transmit antenna case, and the equivalent received signal is given by (10), with the $KL \times N$ equivalent channel matrix $\mathbf{H}$ formatted according to (11). All the properties of $\mathbf{H}$ described in Section 3.1 also hold. Note that to make the decoding process easier, the effect of the constellation rotation has been included in the equivalent channel matrix. The full-rate SF code for three transmit antennas ($K = 3$) can be easily obtained from the 4×4 quasi-orthogonal design by deleting one column from the code matrix $\mathbf{C}$.

3.3. *Full-diversity SF codes*

The SF codes described in Sections 3.1 and 3.2 can achieve full spatial diversity (a diversity order of KL), but not full spatial and frequency diversity (a diversity order of KLP). In [6], a method was proposed to construct full-diversity SF codes from full-rank ST codes via a repetition mapping, trading off data rate for performance. It was shown that by repeating each row of the ST code matrix p times ($1 \le p \le P$), a diversity order of KLp can be achieved. For instance, for MIMO channels with at least two delay paths ($P \ge 2$), the SF code (6) achieves a diversity order of $2L$, while the SF code obtained by repeating each row of (6) two times as

$$\mathbf{C} = \begin{bmatrix} x_0 & x_1 \\ x_0 & x_1 \\ -x_1^* & x_0^* \\ -x_1^* & x_0^* \end{bmatrix} \quad (14)$$

can achieve a diversity order of $4L$. Now we demonstrate that this construction can also be transformed into an equivalent representation that is convenient for sphere decoding through the simple $K = 2$, $L = 1$ example with repetition 2. In this case, the equivalent received signal vector becomes $\mathbf{y} = [y_0[0], y_0[1], y_0^*[2], y_0^*[3]]^T$, and the equivalent noise component is $\mathbf{n} = [n_0[0], n_0[1], n_0^*[2], n_0^*[3]]^T$. Then, the equivalent received signal vector $\mathbf{y}$ can be expressed as in (10), with the channel symbol vector $\mathbf{x} = [x_0, x_1]^T$ and the equivalent channel matrix

$$\mathbf{H} = \begin{bmatrix} H_{0,0}[0] & H_{1,0}[0] \\ H_{0,0}[1] & H_{1,0}[1] \\ H_{1,0}^*[2] & -H_{0,0}^*[2] \\ H_{1,0}^*[3] & -H_{0,0}^*[3] \end{bmatrix}. \quad (15)$$

The generalization of this approach to three and four transmit antennas, more receive antennas, and more repetitions is straightforward.

4. THE PROPOSED DECODING FRAMEWORK

For communication systems where the received signal vector can be expressed as in (10), to decode the sent signal vector $\mathbf{x}$ with the maximum-likelihood (ML) algorithm, the task is to find a valid signal vector $\mathbf{x}$ that minimizes the metric $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2$. Unfortunately, in some cases this can only be performed by exhaustive search over all valid signal vectors, which may result in prohibitively high computational complexity. To alleviate this computational burden, sphere decoding was proposed [9], where the decoder searches over only a subset of $\mathbf{x}$ vectors that lie within a hypersphere of radius r centered around the received signal vector, that is, $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2 \le r^2$.

Despite the recent advances in sphere decoding [10–14], the existing methods have some disadvantages. Most decoding approaches were formulated in the real domain, limiting their use for cases where the complex constellation can be decomposed into the product of two real constellations. Moreover, all of these methods are tied to a particular constellation structure: square QAM or PSK. Motivated by the above observations, we have devised a general decoding framework for SF codes constructed from orthogonal and quasi-orthogonal designs. The framework is formulated in the complex domain and can be used with any memoryless modulation method. Moreover, its flexible modular structure allows for extra implementation freedom. We have divided the description of the framework in two parts: the preprocessing stage and the searching stage.

4.1. *Preprocessing stage*

The purpose of this stage is to transform the expression $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2$ in such a form that the decisions on the coordinates of $\mathbf{x}$ can be made sequentially. In the sequel, we assume that the equivalent channel matrix $\mathbf{H}$ is $M\times N$. In case of the full-rate SF codes, we have $M = KL$, and in case of the full-diversity SF codes, M can be calculated as $M = KLp$.

First, the complex Cholesky factorization of the matrix $\mathbf{H}^{\mathcal{H}}\mathbf{H}$ is calculated, obtaining an $N \times N$ complex, upper triangular matrix $\mathbf{R} = \{R_{k,l}\}$ with positive and real diagonal elements. We used a simple extension of the "Gaxpy" version of the real Cholesky decomposition algorithm [20]. Then, the zero forcing solution $\mathbf{z} = \mathbf{H}^+\mathbf{y}$ is determined, where $\mathbf{H}^+$ denotes the pseudoinverse of $\mathbf{H}$, defined as $\mathbf{H}^+ = (\mathbf{H}^{\mathcal{H}}\mathbf{H})^{-1}\mathbf{H}^{\mathcal{H}}$. This can be done most efficiently by solving the lower triangular system $\mathbf{R}^{\mathcal{H}}\mathbf{w} = \mathbf{H}^{\mathcal{H}}\mathbf{y}$ for $\mathbf{w}$ and solving the upper triangular system $\mathbf{R}\mathbf{z} = \mathbf{w}$ for $\mathbf{z}$. By taking advantage of the equalities $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2 = \|\mathbf{H}(\mathbf{z} - \mathbf{x})\|^2 + \|\mathbf{y}\|^2 - \|\mathbf{H}\mathbf{z}\|^2$ and $\|\mathbf{H}(\mathbf{z} - \mathbf{x})\|^2 = \|\mathbf{R}(\mathbf{z} - \mathbf{x})\|^2$, the sphere decoding problem can be expressed in the following way [13]: find only those $\mathbf{x}$ signal vectors that satisfy $\|\mathbf{R}(\mathbf{z} - \mathbf{x})\|^2 \le r'^2$, where r' is the modified radius given by $r'^2 = r^2 - \|\mathbf{y}\|^2 + \|\mathbf{H}\mathbf{z}\|^2$. Finally, the elements of the $N \times N$ matrix $\mathbf{Q} = \{Q_{k,l}\}$ are calculated

Table 1: Number of real operations for the complex implementation.

Algorithm component	Number of additions	Number of multiplications	Number of divisions	Number of square roots
Calculating the $\mathbf{H}^{\mathcal{H}}\mathbf{H}$ matrix	$(2M-1)N^2+(2M-1)N$	$2MN^2+2MN$	0	0
Calculating the Cholesky fact.	$(1/3)(2N^3+3N^2+N)$	$(1/3)(2N^3+6N^2+4N)$	N	N
Calculating $\mathbf{H}^{\mathcal{H}}\mathbf{y}$	$4MN-2N$	$4MN$	0	0
Solving two $N\times N$ triang. systems	$4N^2-4N$	$4N^2-4N$	$4N$	0
Calculating the radius r'^2	$4MN+2M$	$4MN+4M$	0	0
Calculating the $\mathbf{Q}$ matrix	0	N	N^2-N	0

Table 2: Number of real operations for the real implementation.

Algorithm component	Number of additions	Number of multiplications	Number of divisions	Number of square roots
Calculating the $\mathbf{H}^T\mathbf{H}$ matrix	$2(2M-1)N^2+(2M-1)N$	$4MN^2+2MN$	0	0
Calculating the Cholesky fact.	$(4/3)(N^3-N)$	$(1/3)(4N^3+6N^2+2N)$	$2N$	$2N$
Calculating $\mathbf{H}^T\mathbf{y}$	$4MN-2N$	$4MN$	0	0
Solving two $2N\times 2N$ triang. systems	$4N^2-2N$	$4N^2-2N$	$4N$	0
Calculating the radius r'^2	$4MN+2M$	$4MN+4M$	0	0
Calculating the $\mathbf{Q}$ matrix	0	$2N$	$2N^2-N$	0

as $Q_{k,k} = R_{k,k}^2$ and $Q_{k,l} = R_{k,l}/R_{k,k}$ for $k < l$. The matrix $\mathbf{Q}$ is an auxiliary quantity [10] used by the sphere decoding algorithm instead of $\mathbf{R}$.

We now provide a detailed complexity analysis for the preprocessing stages using Cholesky decomposition. Tables 1 and 2 compare the computational complexities of the complex and real implementations of the preprocessing stage in terms of the number of real additions (and subtractions), multiplications, divisions, and square root operations per code block. Table 1 summarizes the complexity of the complex implementation. One complex addition was counted as two real additions, and one complex multiplication was counted as four real multiplications and two real additions. Since in the preprocessing stage, complex quantities are divided only by real quantities, one complex division was counted as two real divisions. The square root operation was applied only to real quantities. One squared magnitude operation was counted as two real multiplications and one real addition.

The computational complexity of the real implementation was determined similarly. In this case, all operations are real, including the Cholesky decomposition, but the $\mathbf{H}$ matrix has $2M$ rows and $2N$ columns. The number of real operations for the real implementation is given in Table 2.

Comparing the operation counts in Tables 1 and 2, it is apparent that the complex implementation of the preprocessing stage has lower computational complexity. Taking into account only the dominant terms MN^2 and N^3, the complex version requires about half of the number of real additions and multiplications. The complex implementation needs N^2+4N real divisions per code block, while the real implementation needs $2N^2+5N$, approximately twice as many. Finally, the number of square root operations for the complex version is exactly the half of that for the real version. In summary, by implementing the preprocessing stage in the complex domain, for each operation, the operation count can be approximately halved compared to the real implementation.

4.2. Searching stage

The searching stage generates the signal vectors $\mathbf{x}$ satisfying the sphere constraint and selects the decoded signal vector. Using the quantities produced by the preprocessing stage, the sphere equation can be expressed as

$$||\mathbf{R}(\mathbf{z}-\mathbf{x})||^2 = \sum_{k=0}^{N-1} Q_{k,k}\left| z_k - x_k + \sum_{l=k+1}^{N-1} Q_{k,l}(z_l - x_l)\right|^2, \tag{16}$$

where x_k and z_k denote the kth ($k = 0,1,\dots,N-1$) coordinates of the vectors $\mathbf{x}$ and $\mathbf{z}$, respectively.

To be able to make decisions on the signal vector $\mathbf{x}$ coordinate by coordinate, the sphere constraint is expressed in a recursive manner by defining the quantities α_k and T_k as follows:

$$\alpha_k = z_k + \sum_{l=k+1}^{N-1} Q_{k,l}(z_l - x_l) = z_k + \sum_{l=k+1}^{N-1} Q_{k,l}\Delta_l \tag{17}$$

for $k = N-1, N-2, \dots, 0$, and $\Delta_k = z_k - x_k$ is defined for computational convenience: $T_{N-1} = r'^2$, and

$$T_k = T_{k+1} - Q_{k+1,k+1}\left|\alpha_{k+1} - x_{k+1}\right|^2 \tag{18}$$

for $k = N-2, N-3, \dots, 0$. The quantity T_k can be thought of as the remaining squared distance between the partial solution $x_{N-1}, x_{N-2}, \dots, x_{k+1}$ and the surface of the sphere, and

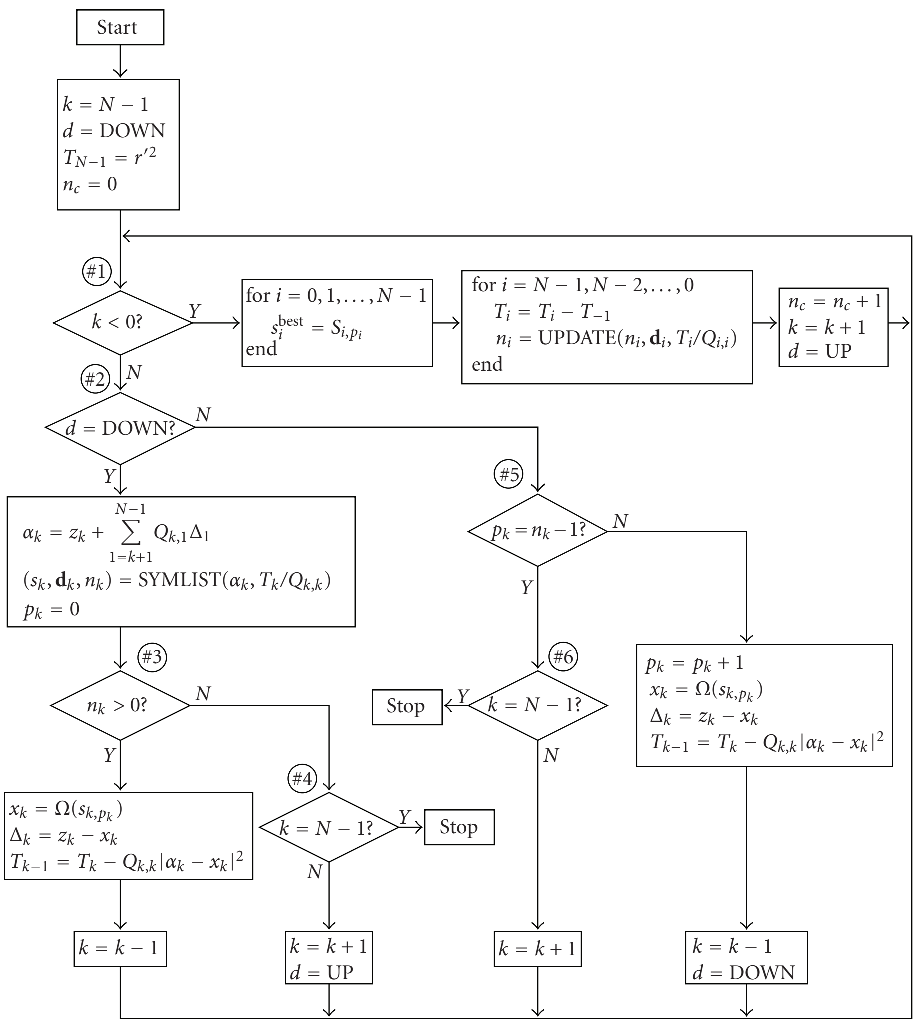
Start
$k = N-1$
$d = \text{DOWN}$
$T_{N-1} = r'^2$
$n_c = 0$
#1
$k < 0?$
Y
N
for $i = 0, 1, \ldots, N-1$
$s_i^{\text{best}} = S_{i,p_i}$
end
for $i = N-1, N-2, \ldots, 0$
$T_i = T_i - T_{-1}$
$n_i = \text{UPDATE}(n_i, \mathbf{d}_i, T_i/Q_{i,i})$
end
$n_c = n_c + 1$
$k = k + 1$
$d = \text{UP}$
#2
$d = \text{DOWN}?$
$\alpha_k = z_k + \sum_{1=k+1}^{N-1} Q_{k,1}\Delta_1$
$(s_k, \mathbf{d}_k, n_k) = \text{SYMLIST}(\alpha_k, T_k/Q_{k,k})$
$p_k = 0$
#3
$n_k > 0?$
$x_k = \Omega(s_{k,p_k})$
$\Delta_k = z_k - x_k$
$T_{k-1} = T_k - Q_{k,k}|\alpha_k - x_k|^2$
$k = k - 1$
#4
$k = N - 1?$
Stop
$k = k + 1$
$d = \text{UP}$
#5
$p_k = n_k - 1?$
#6
$k = N - 1?$
Stop
$k = k + 1$
$p_k = p_k + 1$
$x_k = \Omega(s_{k,p_k})$
$\Delta_k = z_k - x_k$
$T_{k-1} = T_k - Q_{k,k}|\alpha_k - x_k|^2$
$k = k - 1$
$d = \text{DOWN}$

Figure 1: The flowchart of the decoding algorithm.

α_k can be interpreted as the kth received signal component given $x_{N-1}, x_{N-2}, \ldots, x_{k+1}$. The sphere decoding problem can be solved by going over the coordinates of $\mathbf{x}$ starting from $k = N-1$ and continuing down to $k = 0$. For each k value, we need to calculate α_k and T_k and determine those valid x_k values that satisfy

$$|\alpha_k - x_k|^2 \le \frac{T_k}{Q_{k,k}}. \tag{19}$$

Then, for each valid x_k value, we decrease the value of k by one, and repeat the procedure. If the $k = 0$ level is completed, we have found an $\mathbf{x}$ vector inside the sphere.

The original sphere decoding algorithm [9] was based on an algorithm developed for finding the shortest vector in a lattice. Our objective is to devise a decoding framework for any memoryless modulation method, so the signal coordinates x_k may take on any complex value, and the set of vectors $\{\mathbf{Hx}\}$ does not form a lattice. To extend the sphere decoding to arbitrary constellations, we provide an alternative interpretation. We abandon the lattice concept and look at the sphere decoding problem from a different angle. For each k value, all x_k values satisfying the partial constraint (19) are enumerated and explored, so the search space can be represented by a tree. The sphere constraint limits the number of branches emanating from the tree nodes, constraining the search space. Furthermore, finding a valid candidate $\mathbf{x}$ (reaching one of the leaf nodes) may also result in search space reduction, as the radius of the sphere can be decreased at this point [9]. This suggests that the search tree should be explored in a depth-first manner. Finally, as suggested in [10], the valid x_k values are sorted according to the metric $d_k = |\alpha_k - x_k|^2$ in increasing order, and they are explored in this order, which corresponds to a "greedy" strategy. As a consequence, the proposed sphere decoding framework can be interpreted as a greedy, constrained depth-first tree search algorithm.

The flowchart of the proposed algorithm can be observed in Figure 1. The main differences between this algorithm and the one described in [10] are that (a) we formulated the decoding problem in the complex domain and not in the real domain, (b) our algorithm can be used with any constellation, not only with square QAM, and (c) we provide a flexible and modular structure, as opposed to the fixed structure of the algorithm in [10].

The algorithm starts at the root of the tree and initializes the necessary variables. The variable d indicates whether a node is reached from above ("DOWN"); that is, it is visited for the first time, or from below ("UP"), that is, it has been visited before. The variable n_c counts the number of valid candidate solutions that have been found. At the beginning of the main loop (point (#1) in Figure 1), the index k is checked, and if it is nonnegative, the bottom level of the tree has not been reached yet. At point (#2), the algorithm checks whether the current node is visited for the first time. If so (d = "DOWN"), the current value of α_k is calculated and the function SYMLIST generates the list of possible source symbols $s_{k,i}$, whose corresponding channel symbols $x_{k,i} = \Omega(s_{k,i})$ satisfy the normalized partial constraint $|\alpha_k - x_{k,i}|^2 \leq T_k/Q_{k,k}$. The SYMLIST function takes α_k and the normalized partial constraint $T_k/Q_{k,k}$ as inputs and produces 3 outputs. The first output, n_k $(0 \leq n_k \leq B)$, is the number of symbols satisfying the current partial constraint, so there will be n_k branches emanating from the current node. The second output is the symbol list $\mathbf{s}_k = [s_{k,0}, s_{k,1}, \ldots, s_{k,n_k-1}]^T$ and the vector $\mathbf{d}_k = [d_{k,0}, d_{k,1}, \ldots, d_{k,n_k-1}]^T$ whose coordinates are the metrics $d_{k,i} = |\alpha_k - x_{k,i}|^2$. The symbols in $\mathbf{s}_k$ are ordered according to increasing $d_{k,i}$ values. The symbol pointer p_k is initialized to point to the first element of $\mathbf{s}_k$. At point (#3), the algorithm checks whether there are any symbols on the list. If there is at least one, we take the first source symbol from the list, calculate the corresponding channel symbol, calculate the current Δ_k value, determine the partial constraint for the next tree level, and move one level down on the tree. If there are no symbols on the list, we reach to point (#4), where we check the current tree level. If it is the top level ($k = N - 1$), the algorithm terminates without solution. If it is not the top level, we change directions and move back up to the parent node of the current node to continue the search of the tree.

If the current node has been visited before (point (#5)), we check whether there are symbols left on the list. If so, we increment the symbol pointer p_k to point to the next symbol, take the next source symbol from the list, calculate the corresponding channel symbol, calculate the current Δ_k value, determine the partial constraint for the next level, and move one level down on the tree. If there are no source symbols left on the list, we get to point (#6) to check whether we are at the top level ($k = N - 1$). If so, the algorithm terminates. If the value of n_c is zero, no solution was found; otherwise the algorithm was able to find at least one $\mathbf{x}$ vector satisfying the sphere constraint. If the algorithm has not reached the top level yet, it moves one level up to the parent node to continue the search.

If the value of k becomes negative (point (#1)), we have reached the bottom level, so a valid candidate solution $\mathbf{x}$ has been found. This $\mathbf{x}$ vector is the best solution found so far, so the corresponding source symbols are saved in the $\{s_i^{\text{best}}\}$ variables by overwriting the previous solution. Then, the radius of the sphere is reduced to further decrease the decoding complexity by adjusting all partial constraint values such that the last solution satisfies the constraints with equality (the "surplus" partial constraint T_{-1} is subtracted from each partial constraint). The source symbol lists are also modified by the UPDATE function, which keeps only those source symbols on the list whose corresponding d_k metric values satisfy the new partial constraints. Because the symbols are ordered according to the corresponding d_k values, this can be done simply by changing the value of n_k for each k. Finally, we move back up to the parent node to continue the search.

Since the algorithm makes decisions based on variables that are functions of random quantities and previous decisions, it is very hard to calculate its computational complexity accurately. Other researchers could only determine approximate asymptotic results [21] on the order of the number of operations or provide formulas for the approximate complexity that can only be evaluated numerically [22]. Both works assumed lattice structure, that is, the coordinates of $\mathbf{x}$ were integers, which is not true in our case. Moreover, asymptotic results on the order of the operations are not appropriate for the comparison of two sphere decoding algorithms, as they hide any constant factors, and the dimensionality of the problem (the values M and N) never becomes large. Therefore, we have chosen simulation-based complexity comparison by counting the number of operations and averaging them over a large number of experiments.

5. THE PROPOSED SEARCH METHODS

The algorithm of Figure 1 is only a general framework that performs a greedy, constrained depth-first tree search. The heart of the decoding algorithm is the function SYMLIST, which creates the ordered list of source symbols satisfying the partial constraint at the given tree level. This section proposes several ways to implement the SYMLIST function.

The header of the function is described as $(\mathbf{s}, \mathbf{d}, n) =$ SYMLIST(α, T), where the inputs are the value of α and the partial constraint T and the outputs are n, the number of symbols on the list, the symbol list $\mathbf{s} = [s_0, s_1, \ldots, s_{n-1}]^T$, and the corresponding metric list $\mathbf{d} = [d_0, d_1, \ldots, d_{n-1}]^T$.

5.1. *The modulation-independent search*

First, we assume that there is no a priori information available on the used constellation, so we have to develop an algorithm that would work with any memoryless modulation method. In this case, to create the source symbol list, we need to enumerate all possible source symbols, and sort the ones that satisfy the partial constraint, and put them onto the symbol list.

The algorithm goes over all possible source symbols $s = 0, 1, \ldots, B - 1$, and for each source symbol, it calculates the corresponding channel symbol $x = \Omega(s)$ and the metric $d = |\alpha - x|^2$. Then, it checks whether the partial constraint $d \leq T$ is satisfied. If not, the execution continues with the next s value. If the constraint is satisfied, the source symbol s is inserted into the list $\mathbf{s}$, the metric d is inserted into the metric list $\mathbf{d}$, and the number of items on the list gets incremented. At the end, the metric list and the symbol list are sorted according to the metric values such that the symbol with the smallest metric will be the first on the list.

5.2. *The fast search method*

Most of the previously proposed sphere decoding approaches enumerate all signal points x satisfying the partial constraint at all tree levels. However, it was observed via simulations that most of the time, the first candidate solution $\mathbf{x}$ found by the greedy tree search (i.e., the greedy solution) is actually the ML solution, so only one leaf node is explored during the decoding process with high probability. This means that enumerating (and possibly sorting [10]) all signal points that satisfy the partial constraints is redundant because most source symbols on the symbol lists get eliminated via radius reduction and the subtrees corresponding to their values will never be explored.

To further improve the computational efficiency of the sphere decoding algorithm, we propose a fast search approach by making the most probable case more efficient. The proposed fast search algorithm performs a nearest-neighbor signal point search. The symbol list generation algorithm SYMLIST enumerates only a few nearest signal points x to α and ignores the signal points that are further away, so the unnecessary enumeration and sorting operations can be avoided.

5.2.1. *Square QAM*

If the used constellation is B-ary square QAM, the source symbol s can be expressed as the concatenation of its real and imaginary parts: $s = (s^R, s^I)$, $s^R, s^I \in \{0, 1, \ldots, \sqrt{B} - 1\}$, and the complex channel symbol corresponding to this source symbol is

$$x = \Omega(s) = \left(s^R - \frac{\sqrt{B}-1}{2}\right) + j\left(s^I - \frac{\sqrt{B}-1}{2}\right). \quad (20)$$

For the simplicity of the explanation, we do not consider Gray bit-mapping and neglect the edge effects, assuming that the point α falls in a region where it has four neighboring signal points, as shown in Figure 2. The basic idea of the fast search method is to identify these four constellation points efficiently by exploiting the geometrical properties of the QAM constellation. The first step is to express the value of α in the coordinate system of the real and imaginary parts of the source symbols as

$$s_\alpha^R = \mathrm{Re}\{\alpha\} + \frac{\sqrt{B}-1}{2}, \qquad s_\alpha^I = \mathrm{Im}\{\alpha\} + \frac{\sqrt{B}-1}{2}. \quad (21)$$

Then, the source symbols corresponding to the four neighboring constellation points can be identified easily by rounding the values of s_α^R and s_α^I up or down. For example, in Figure 2, the source symbol $c_0 = (c_0^R, c_0^I)$ corresponding to the upper right signal point can be obtained as $c_0^R = \lceil s_\alpha^R \rceil$ and $c_0^I = \lceil s_\alpha^I \rceil$. After determining the 4 source symbols $c_i = (c_i^R, c_i^I)$, $i = 0, 1, 2, 3$, that correspond to the 4 nearest neighbors of α, we need to determine the order in which they should be put on the list. The space between the neighboring QAM constellation points is divided into 4 quadrants, as shown by the dashed lines in Figure 2, and the fractional parts of s_α^R and s_α^I are calculated to determine which quadrant the value s_α falls in by comparing the fractional parts to 0.5. Once the index of the quadrant is known, the source symbol corresponding to the signal point closest to α (c_0 in Figure 2) can be easily determined. The signal point furthest away from α (c_2 in Figure 2) can also be identified. Based on this approach, the order between the other two source symbols (c_1 and c_3 in Figure 2) cannot be decided, but our simulations have shown that the actual order of the symbols is not important as long as the source symbol corresponding to the closest signal point is the first on the list, so we set arbitrary (e.g., random) order for those two signal points.

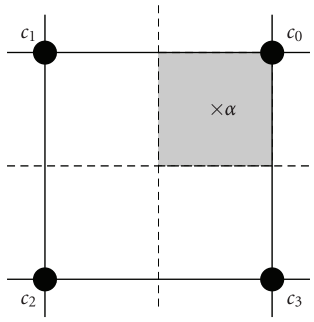

FIGURE 2: The fast QAM-search algorithm.

Finally, for each of the 4 source symbols, the algorithm calculates the corresponding channel symbol and checks whether the partial constraint is satisfied. If so, the symbol and the corresponding metric value are put on the lists. The remaining $B - 4$ source symbols are ignored. Note that the symbols on the list will not be perfectly ordered some of the time. However, the algorithm ensures that the source symbol corresponding to the closest constellation point to α will always be the first. As can be seen, the algorithm avoids the enumeration of all channel symbols satisfying the partial constraint and avoids the sorting operation altogether.

The edge effects are handled by projecting the values of s_α^R and s_α^I onto the boundaries of the constellation if necessary. That is, if s_α^R is smaller than 0, it is set to 0, and if it is greater than $\sqrt{B} - 1$, it is set to $\sqrt{B} - 1$. The imaginary part s_α^I is transformed similarly. With this transformation, the transformed α point will fall on the boundary of a region with four neighboring constellation points, so the previously described algorithm can work properly. Since the original signal point α was outside the constellation, this perpendicular projection will not affect the identity of the closest constellation point.

5.2.2. *PSK*

In case of B-ary PSK, the modulated complex channel symbol is determined by

$$x = \Omega(s) = Re^{j((2\pi/B)s + \Phi)}, \quad s \in \{0, 1, \ldots, B - 1\}. \quad (22)$$

Table 3: Number of real operations for the preprocessing stages.

Implementation	Number of additions	Number of multiplications	Number of divisions	Number of square roots	Total number of FLOPS
Complex	128.00	154.00	12.00	2.00	296.00
Real	158.00	196.00	18.00	4.00	376.00
Preprocessing stage of [12]	646.00	916.00	27.96	4.00	1593.96

In (22), R is the radius of the constellation (usually unity), and Φ is the rotation angle of the constellation (usually zero). The idea is similar to the square QAM case: we identify the source symbols corresponding to the two closest signal points to α without enumerating all signal points that satisfy the partial constraint. As the first step, we express α in the coordinate system of the source symbols, which is an angular coordinate system:

$$s_\alpha = \frac{B}{2\pi}(\Phi_\alpha - \Phi), \tag{23}$$

where Φ_α is the angle of α, given by $\Phi_\alpha = \arctan(\mathrm{Im}\{\alpha\}, \mathrm{Re}\{\alpha\})$, and $\arctan(\cdot,\cdot)$ is the four-quadrant arcus tangent function. Now we can easily determine the two source symbols c_0 and c_1 corresponding to the nearest neighbors of s_α by rounding up and down its value. For easy explanation, the edge effect will not be considered, so c_0 and c_1 are assumed to be in the set $\{0, 1, \ldots, B-1\}$. If they fall outside, we can easily map them back by adding or subtracting B to/from their values. Then, the fractional part of s_α is calculated to determine which neighbor is the closest by comparing it to 0.5. The source symbol corresponding to the closest signal point is put on the list first, and the other source symbol will be the second, provided that they satisfy the partial constraint. The rest of the source symbols are ignored.

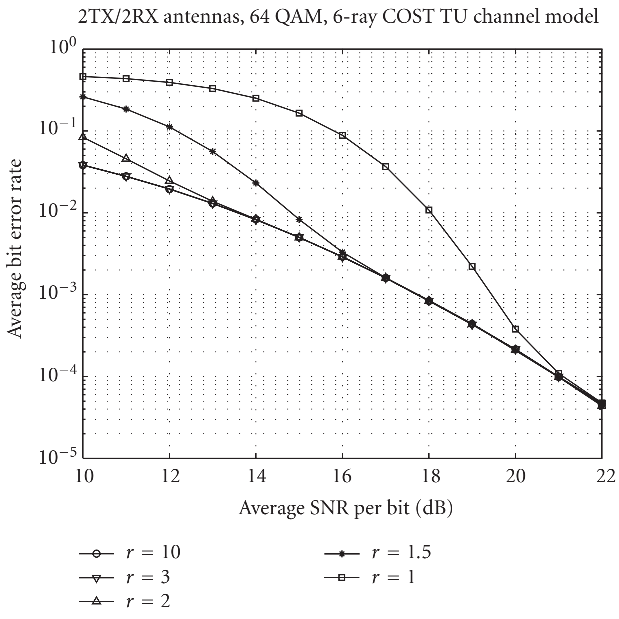

Figure 3: Average bit error rate of the decoding algorithm with 64 QAM.

6. SIMULATION RESULTS

To illustrate the performance of the proposed sphere decoding algorithm, we provide some simulation results. The simulated communication system had 2 transmit antennas ($K = 2$), 2 receive antennas ($L = 2$), and the 2×2 orthogonal design (6) was used as the SF code ($M = KL = 4$, $N = 2$, $G = 2$). The OFDM modulation had $S = 128$ subcarriers with an OFDM symbol period of $T = 128\,\mu$s. The frequency-selective MIMO channel was modeled by the COST 207 Typical Urban 6-ray power delay profile [23].

Since the computational complexity of the preprocessing stage is independent of the applied modulation method and the sphere radius, it is discussed separately from the searching stage. Table 3 shows the number of real operations executed by the complex and real preprocessing stages for each decoded code block. The entries in the first two rows of the table were obtained by substituting $M = 4$ and $N = 2$ into the formulas of Tables 1 and 2. The last row of the table contains the operation counts for the preprocessing stage of the sphere decoding algorithm in [12], which uses real QR decomposition. The QR decomposition was implemented according to the description in [20], and the values were obtained via simulations. The operation count values show that considerable complexity reduction can be achieved by implementing the preprocessing stage in the complex domain, even for small M and N values. Particularly the number of necessary multiplications and square root operations can be reduced significantly. The computationally most expensive part was the calculation of the matrix $\mathbf{H}^{\mathcal{H}}\mathbf{H}$ (or $\mathbf{H}^T\mathbf{H}$), and the complexity of the Cholesky decomposition was negligible due to the small size (2×2) of the matrix to be decomposed. It can also be observed that the preprocessing stage of [12] has considerably higher computational complexity, mainly due to the complexity of the QR decomposition.

We compared the complexity of the searching stages of different decoding algorithms by counting the number of real operations and averaging them over a large number ($10^5 - 10^6$) of experiments. The complex operations were counted according to the method described earlier. Moreover, one floating-point comparison was counted as one addition. Some algorithms also used transcendent function evaluations, such as arcus tangent and arcus cosine; these operations are counted separately. The floating-point operation (FLOP) count includes all operations.

Figure 3 shows the bit error rate (BER) curves of the proposed sphere decoding algorithm with 64 QAM modulation, and the average FLOP counts per code block of the searching stage of the proposed algorithm is shown in Figure 4, both as a function of the average signal to noise ratio (SNR) in case of different radius values. The figure shows how the performance improves at the price of increased computational complexity. If the radius is small and the SNR is low, no solution vector will fall inside the sphere some of the time, which reduces the average decoding complexity, but results in considerable performance degradation. It can also be observed that increasing the radius beyond $r = 3$ has almost no effect on the decoding complexity of the searching stage. Therefore, the radius was set to a large enough value, $r = 10$, to ensure ML performance in the SNR range of interest in case of all simulations.

6.1. 64 QAM

In case of 64 QAM modulation, we compared the complexity of the searching stages of five different decoding algorithms: the ML decoding algorithm (performing exhaustive search), the real-domain sphere decoding algorithm described in [10] (preprocessing stage: real Cholesky decomposition), the real-domain sphere decoding algorithm described in [12] (preprocessing stage: real QR decomposition), and the proposed decoding framework (preprocessing stage: complex Cholesky decomposition) with two symbol list generating method: the modulation-independent search and the fast QAM-specific search. Figure 5 depicts the BER curves as functions of the average SNR. It can be observed that all sphere decoding algorithms have the same performance as the ML decoding algorithm in the SNR range of interest. Table 4 provides a detailed break-down of the number of operations for the searching stages, including the average value of n_c. Since the variations of the operation counts is very small over the different SNR values, we only present the operation counts averaged over the simulated SNR values. In Figure 6, we also provide an example histogram of n_c, which is the number of valid signal vectors found in the sphere, at SNR of 16 dB.

Based on the above results, we can make several observations. First, using sphere decoding, the computational complexity of the SF decoder can be reduced by orders of magnitude without perceptible performance loss in the meaningful BER range. Second, the computational cost of the QAM-specific sphere decoding algorithms is dominated by the preprocessing stage. This is due to the combined effects of sphere radius reduction, the greedy tree search, and the fast symbol list generation method. Third, the complexity of the searching stages do not change considerably as the SNR increases, contradictory to the results of [13]. The reason is that in [13], the initial radius was reduced as the SNR increased (while we kept it constant), and the greedy tree search was not implemented. Fourth, n_c takes the value of 1 with overwhelming relative frequency, and its average value is very close to 1, indicating that the first solution found by the greedy search was actually the ML solution most of the time. Finally, the searching stage of the algorithm described in [12] requires the least number of FLOPs, followed by the proposed fast

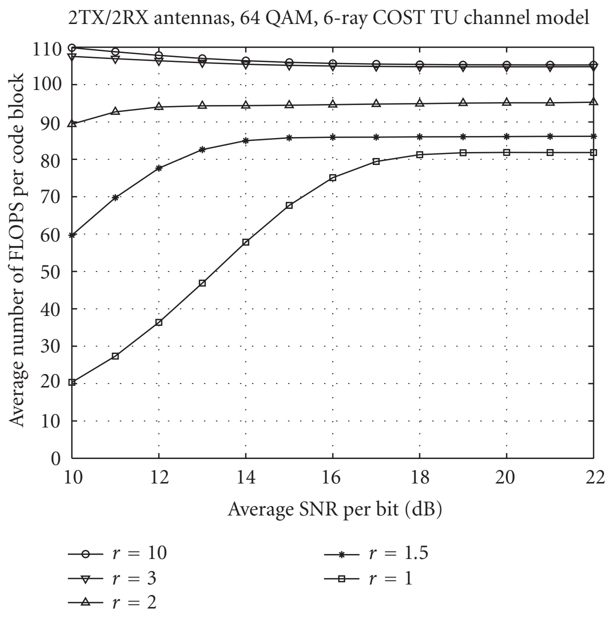

Figure 4: Average FLOP counts of the searching stage of the decoding algorithms with 64 QAM.

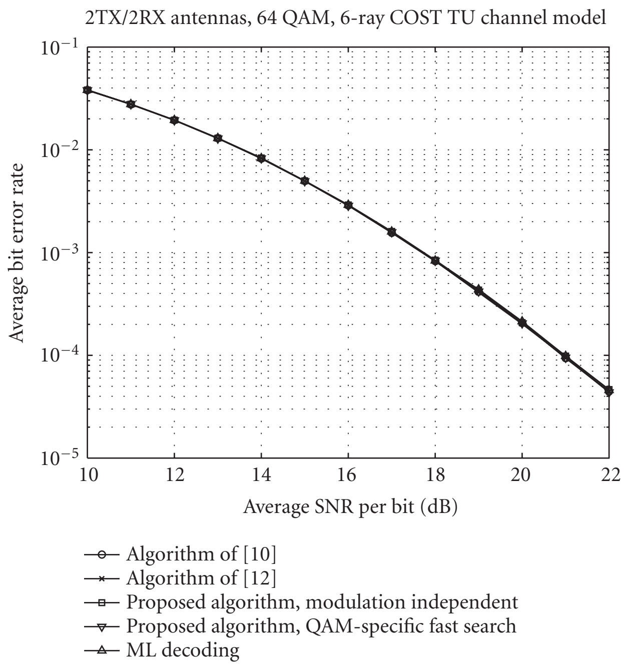

Figure 5: Average bit error rate of the decoding algorithms with 64 QAM.

search method, and the searching stage of [10] is the computationally most expensive. However, taking into account the complexity of both the preprocessing and the searching stages, the proposed QAM-specific fast search algorithm is the most efficient. The total number of FLOPs per decoded code block, including both stages, as a function of the SNR is

Table 4: The average umber of real operations for the searching stages, 64 QAM.

Algorithm	Number of additions	Number of multiplications	Number of divisions	Number of square roots	Number of FLOPS	n_c
ML decoding	167936	163840	0	0	331776	4096
Searching stage of [10]	183.6	48.5	8.1	8.1	248.3	1.00
Searching stage of [12]	57.8	20.8	4.1	0	82.7	1.00
Proposed method, modulation-indep.	1707.6	273.1	4.1	0	1984.8	1.00
Proposed method, fast search	76.1	26.3	4.0	0	106.4	1.00

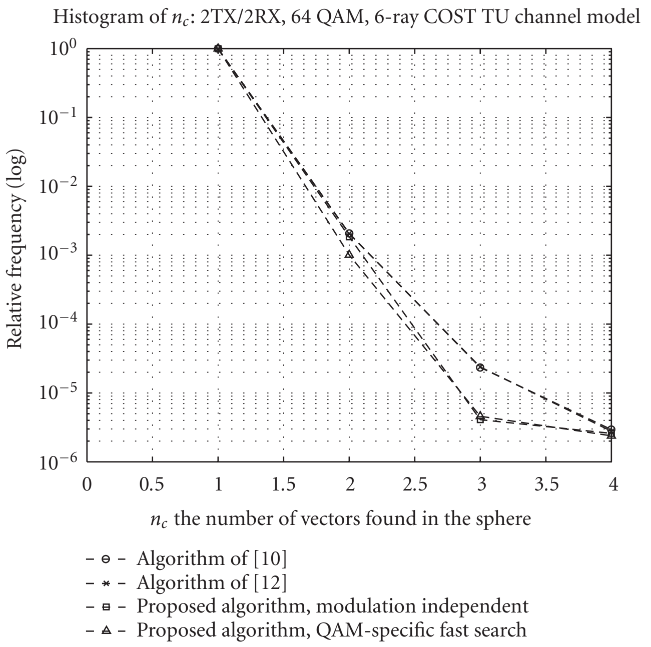

Figure 6: The histogram of n_c for the decoding algorithms with 64 QAM at SNR = 16 dB.

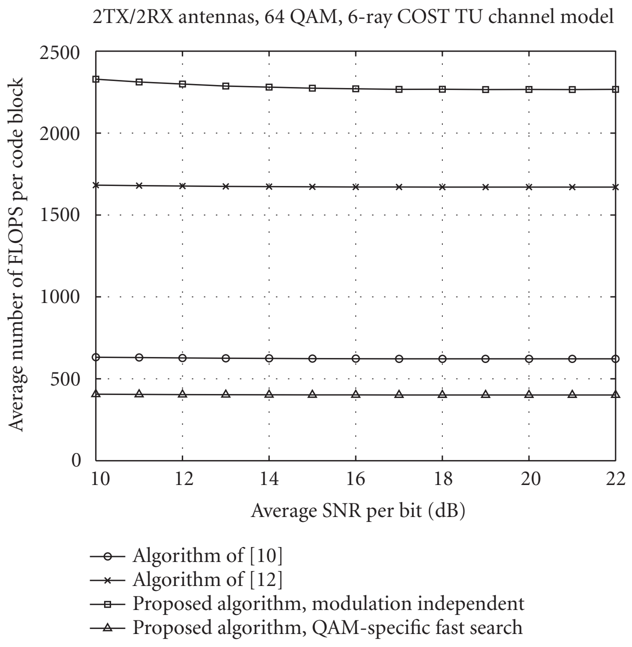

Figure 7: Average FLOP counts of the decoding algorithms with 64 QAM.

depicted in Figure 7. Combining the FLOP counts of Tables 3 and 4, we observe that the total number of FLOPs per code block was reduced to about 64% of the FLOP count of the algorithm in [10] and to about 25% of the FLOP count of the algorithm in [12].

6.2. 16 PSK-QAM

The last experiment was conducted with 16 PSK-QAM modulation. The constellation corresponding to this modulation method is shown in Figure 8. The 16 PSK-QAM constellation is made up of two PSK constellations. In case of natural bit mapping, the inner PSK signal points can be generated by (22) with $R = 1$, $\Phi = 0$, and $B = 8$. The outer constellation points are determined by the same equation with $R = \cos(\pi/8) + \sqrt{3}\sin(\pi/8) = 1.5867$, $\Phi = \pi/8$, and $B = 8$.

The searching stages of four different decoding approaches were compared: the ML decoding algorithm, the algorithm of [14], the proposed sphere decoding algorithm with modulation-independent symbol list generation, and the proposed sphere decoding algorithm with fast, PSK-specific nearest-neighbor search. For all sphere decoding

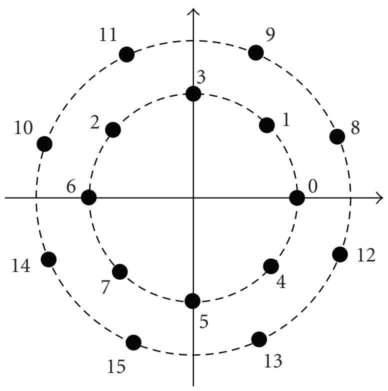

Figure 8: The bit mapping for the 16 PSK-QAM constellation.

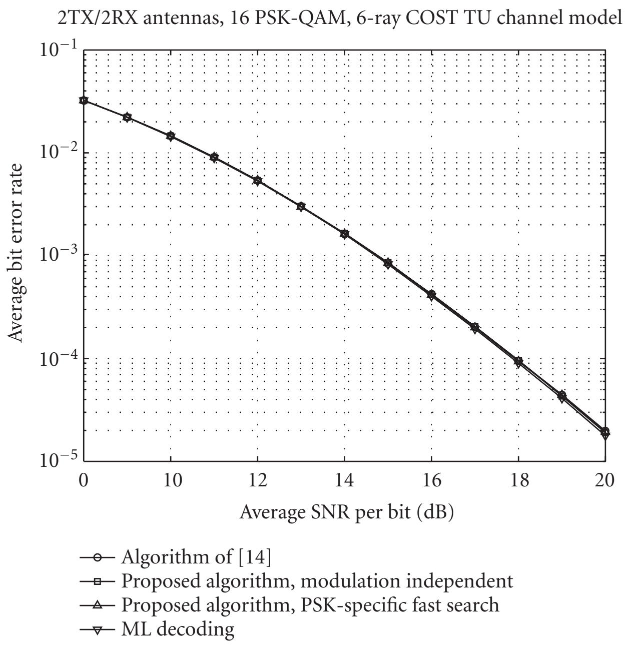

FIGURE 9: Bit error rate of the decoding algorithms with 16 PSK-QAM.

TABLE 5: The average number of real operations for the searching stages, 16 PSK-QAM.

Algorithm	Number of additions	Number of multiplications	Number of divisions	Number of square roots	Number of transc. op.	Number of FLOPS	n_c
ML decoding	10496	10240	0	0	0	20736	256
Searching stage of [14]	310.8	86.3	8.1	2.0	2.1	409.3	1.00
Proposed method, modulation-indep.	326.0	75.0	4.0	0	0	405.0	1.00
Proposed method, fast search	73.9	30.2	8.0	0	2.0	114.1	1.00

algorithms, the same complex Cholesky decomposition-based preprocessing stage was used, and the PSK-specific decoding methods were modified for the 16 PSK-QAM modulation.

The average bit error rate curves of the above mentioned decoding methods are depicted in Figure 9. It is apparent that all sphere decoding approaches perform as well as the ML decoding algorithm. The detailed operation counts averaged over the simulated SNR range are given in Table 5. The average number of transcendent function evaluations was added to the table, as some of the decoding algorithms calculate arcus tangent and arcus cosine values. The histogram of the n_c values at SNR = 14 dB can be observed in Figure 10. Finally, for overall comparison, the total number of FLOPS per code block, including both the preprocessing and the searching stages, is shown in Figure 11.

There are two important observations to be made. First, the constellation-independent search performs slightly better than the search method in [14]. However, for larger constellation sizes, built from PSK constellations, the method of [14] is expected to have lower computational complexity than the modulation-independent decoding algorithm. Second, the proposed fast nearest-neighbor search algorithm achieved significant complexity reduction compared to the search stage of [14]. Taking into account both decoding stages, the total number of FLOPs was reduced by the proposed decoding algorithm to about 58% of the FLOPs of the decoding method described in [14].

7. CONCLUSION

We proposed a computationally efficient SF block code decoding algorithm based on the principles of sphere decoding. We formulated the decoding problem in the complex domain and developed a modulation-independent decoding framework by interpreting sphere decoding as a greedy, constrained depth-first tree search algorithm. We combined this flexible decoding framework with a modulation-independent symbol list generation algorithm, and two modulation-specific symbol list generation algorithms that perform nearest-neighbor signal point search. The simulation results showed that the proposed decoding algorithm further reduced the decoding complexity compared to previously existing approaches. If the MIMO channel is quasistatic (i.e., the preprocessing stage has to be executed only once for many code blocks) and QAM modulation is used,

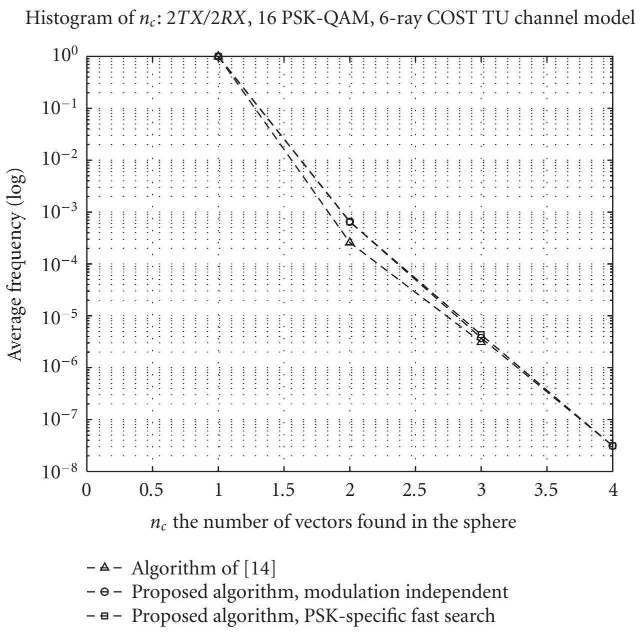

FIGURE 10: The histogram of n_c for the decoding algorithms with 16 PSK-QAM at SNR = 14 dB.

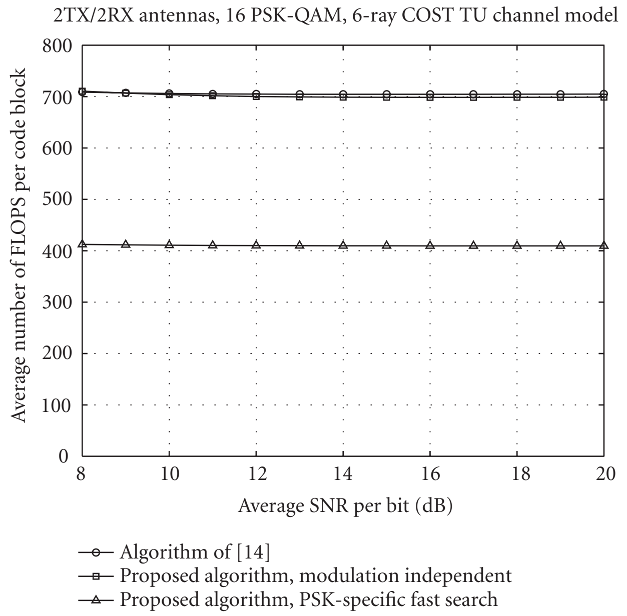

FIGURE 11: Average FLOP counts of the decoding algorithms with 16 PSK-QAM.

the algorithm described in [12] seems to be the right choice, but in case of broadband MIMO-OFDM systems, where the preprocessing stage has to be executed for every code block, our algorithm offers the lowest decoding complexity without any perceptible performance loss.

Note

The work described in this manuscript was in part presented at the European Wireless Conference in February 2004 [24] and at the IEEE International Conference on Communications in January 2004 [25].

REFERENCES

[1] D. Agrawal, V. Tarokh, A. Naguib, and N. Seshadri, "Space-time coded OFDM for high data-rate wireless communication over wideband channels," in *Proceedings of 48th IEEE Vehicular Technology Conference (VTC '98)*, vol. 3, pp. 2232–2236, Ottawa, Ontario, Canada, May 1998.

[2] H. Bölcskei and A. J. Paulraj, "Space-frequency coded broadband OFDM systems," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '00)*, vol. 1, pp. 1–6, Chicago, Ill, USA, September 2000.

[3] Y. Gong and K. B. Letaief, "An efficient space-frequency coded wideband OFDM system for wireless communications," in *Proceedings of IEEE International Conference on Communications (ICC '02)*, vol. 1, pp. 475–479, New York, NY, USA, April–May 2002.

[4] Z. Hong and B. L. Hughes, "Robust space-time codes for broadband OFDM systems," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '02)*, vol. 1, pp. 105–108, Orlando, Fla, USA, March 2002.

[5] Z. Liu, Y. Xin, and G. B. Giannakis, "Space-time-frequency coded OFDM over frequency-selective fading channels," *IEEE Transactions on Signal Processing*, vol. 50, no. 10, pp. 2465–2476, 2002.

[6] W. Su, Z. Safar, M. Olfat, and K. J. R. Liu, "Obtaining full-diversity space-frequency codes from space-time codes via mapping," *IEEE Transactions on Signal Processing*, vol. 51, no. 11, pp. 2905–2916, 2003.

[7] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1451–1458, 1998.

[8] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[9] E. Viterbo and J. Bouros, "A universal lattice code decoder for fading channels," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1639–1642, 1999.

[10] A. M. Chan and I. Lee, "A new reduced-complexity sphere decoder for multiple antenna systems," in *Proceedings of IEEE International Conference on Communications (ICC '02)*, vol. 1, pp. 460–464, New York, NY, USA, April–May 2002.

[11] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Transactions on Information Theory*, vol. 48, no. 8, pp. 2201–2214, 2002.

[12] M. O. Damen, H. El Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Transactions on Information Theory*, vol. 49, no. 10, pp. 2389–2402, 2003.

[13] H. Vikalo and B. Hassibi, "Maximum likelihood sequence detection of multiple antenna systems over dispersive channels via sphere decoding," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 5, pp. 525–531, 2002.

[14] B. M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Transactions on Communications*, vol. 51, no. 3, pp. 389–399, 2003.

[15] K.-W. Wong, C.-Y. Tsui, R. S.-K. Cheng, and W.-H. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '02)*, vol. 3, pp. 273–276, Scottsdale, Ariz, USA, May 2002.

[16] B. Widdup, G. Woodward, and G. Knagge, "A highly-parallel VLSI architecture for a list sphere detector," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 5, pp. 2720–2725, Paris, France, June 2004.

[17] A. Burg, M. Wenk, M. Zellweger, M. Wegmueller, N. Felber, and W. Fichtner, "VLSI implementation of the sphere decoding algorithm," in *Proceedings of 30th European Solid-State Circuits Conference (ESSCIRC '04)*, pp. 303–306, Leuven, Belgium, September 2004.

[18] A. Burg, M. Borgmann, C. Simon, M. Wenk, M. Zellweger, and W. Fichtner, "Performance tradeoffs in the VLSI implementation of the sphere decoding algorithm," in *Proceedings of 5th IEE International Conference on 3G Mobile Communication Technologies (IEE 3G '04)*, pp. 92–96, London, UK, October 2004.

[19] W. Su and X.-G. Xia, "Signal constellations for quasi-orthogonal space-time block codes with full diversity," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2331–2347, 2004.

[20] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Md, USA, 1996.

[21] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Mathematics of Computation*, vol. 44, no. 170, pp. 463–471, 1985.

[22] B. Hassibi and H. Vikalo, "On the expected complexity of integer least-squares problems," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, vol. 2, pp. 1497–1500, Orlando, Fla, USA, May 2002.

[23] G. L. Stüber, *Principles of Mobile Communication*, Kluwer Academic, Boston, Mass, USA, 2001.

[24] Z. Safar, W. Su, and K. J. R. Liu, "Fast sphere decoding of space-frequency block codes via nearest neighbor signal point search," in *Proceedings of 5th European Wireless Conference, Mobile and Wireless Systems beyond 3G (EW '04)*, Barcelona, Spain, February 2004.

[25] Z. Safar, W. Su, and K. J. R. Liu, "A fast sphere decoding framework for space-frequency block codes," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 5, pp. 2591–2595, Paris, France, June 2004.

Zoltan Safar received the University Diploma in electrical engineering from the Technical University of Budapest, Budapest, Hungary, in 1996, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, Maryland, USA, in 2001 and 2003, respectively. From September 2003 to March 2005, he was an Assistant Professor at the Department of Innovation, IT University of Copenhagen, Copenhagen, Denmark. Currently, he is a Senior Design Engineer at Modem System Design, Nokia, in Copenhagen, Denmark. His research interests include wireless communications and multimedia signal processing, with particular focus on multiple-input multiple-output (MIMO) wireless communication systems and space-time and space-frequency coding. Dr. Safar received the Outstanding Systems Engineering Graduate Student Award from the Institute for Systems Research, University of Maryland, in 2003, and the Invention of the Year Award (together with W. Su and K. J. R. Liu) from the University of Maryland in 2005.

Weifeng Su received the Ph.D. degree in electrical engineering from the University of Delaware, Newark, in 2002. He received his B.S. and Ph.D. degrees in mathematics from Nankai University, Tianjin, China, in 1994 and 1999, respectively. His research interests span a broad range of areas from signal processing to wireless communications and networking, including space-time coding and modulation for MIMO wireless communications, MIMO-OFDM systems, cooperative communications for wireless networks, and ultra-wideband (UWB) communications. He has published more than 20 journal papers and 30 conference papers in the related areas. Dr. Su is an Assistant Professor in the Department of Electrical Engineering, the State University of New York (SUNY) at Buffalo. From June 2002 to March 2005, he was a Postdoctoral Research Associate with the Department of Electrical and Computer Engineering and the Institute for Systems Research (ISR), University of Maryland, College Park. Dr. Su received the Signal Processing and Communications Faculty Award from the University of Delaware in 2002 as an outstanding graduate student in the field of signal processing and communications. In 2005, he received the Invention of the Year Award from the University of Maryland. He won the first prize of the Chinese Olympic Mathematics Competition in 1990.

K. J. Ray Liu received the B.S. degree from the National Taiwan University in 1983, and the Ph.D. degree from UCLA in 1990, both in electrical engineering. He is a Professor and Director of Communications and Signal Processing Laboratories of Electrical and Computer Engineering Department and Institute for Systems Research, University of Maryland, College Park. His research contributions encompass broad aspects of wireless communications and networking, information forensics and security, multimedia communications and signal processing, bioinformatics and biomedical imaging, and signal processing algorithms and architectures. Dr. Liu is the recipient of numerous honors and awards including Best Paper Awards from the IEEE Signal Processing Society, the IEEE Vehicular Technology Society, and the EURASIP; IEEE Signal Processing Society Distinguished Lecturer, the EURASIP Meritorious Service Award, and National Science Foundation Young Investigator Award. He also received Poole and Kent Company Senior Faculty Teaching Award from A. James Clark School of Engineering, and Invention of the Year Award, both from the University of Maryland. Dr. Liu is a Fellow of the IEEE. Dr. Liu is the Vice President-Publication and on the Board of Governor of the IEEE Signal Processing Society. He was the Editor-in-Chief of the IEEE Signal Processing Magazine, the prime proposer and architect of the new IEEE Transactions on Information Forensics and Security, and was the founding Editor-in-Chief of EURASIP Journal on Applied Signal Processing.