Low complexity maximum-likelihood detector for DSTTD architecture based on the QRD-M algorithm

Joaquín Cortez¹ · Miguel Bazdresch² · Julio Ramírez³ · Ramón Palacio¹ · Erica Ruiz¹

Published online: 22 May 2018
© The Author(s) 2018

Abstract

This paper presents a new decoder algorithm for the double space–time transmit diversity (DSTTD) system. The decoder is based on the QRD-M algorithm, which performs a breadth-first search of possible solutions tree. The search is simplified by skipping unlikely candidates, and it is stopped when no promising candidates are left. Furthermore, the search is divided into three concurrent iterations, making possible a fast, parallel implementation either in hardware or software. After presenting an analysis of the capacity and diversity of DSTTD, we present performance results showing that the proposed decoder is capable of achieving near maximum likelihood performance. We also show that the proposed algorithm exhibits lower computational complexity than other existing maximum likelihood detectors.

Keywords Space–time codes · Double space–time transmit diversity · Maximum-likelihood detection · QRD-M algorithm

1 Introduction

It is known that, when transmitting and receiving over multiple antennas, the rich-scattering wireless channel has enormous capacity [1]. Furthermore, this capacity can be exploited to obtain increased data rates, or an increase in reliability; a tradeoff between these two properties can also be achieved [2].

Over the past two decades, techniques known as space–time codes have been developed to exploit these gains. Some

space–time codes focus on spatial multiplexing gain; independent symbols are transmitted over different antennas at each channel use, increasing the data rate, but also the interference at the receiver. The Bell Labs layered space–time (V-BLAST) architecture [3] is an example. Space–time block codes (STBC), such as the Alamouti scheme [4] and other orthogonal designs [5], aim to provide transmitter diversity; crucially, this is achieved without requiring channel knowledge at the transmit side. General space–time frameworks have also been proposed, which enable the design of codes that provide a mix of spatial multiplexing and diversity gain. One example is linear dispersion codes [6], which use the maximization of the mutual information between transmitter and receiver as a design criterion.

Hybrid space–time codes present a simple way to achieve both spatial multiplexing and transmit diversity gain [7–10]. These codes operate in layers, like V-BLAST; however, at least some of the layers consist of a set of antennas transmitting an STBC code. In this paper, we focus on the double space–time transmit diversity (DSTTD) scheme [11], which consists of two layers, each of which is an Alamouti STBC. This architecture provides an increase in data rate, as the two layers transmit in parallel, while still offering the transmit diversity advantage of each underlying Alamouti code. DSTTD has been adopted by the IEEE 802.11n [12] and 802.16e [13] WLAN standards.

✉ Miguel Bazdresch
mxbiee@rit.edu

Joaquín Cortez
joaquin.cortez@itson.edu.mx

Julio Ramírez
jramirez@ucaribe.edu.mx

Ramón Palacio
ramon.palacio@itson.edu.mx

Erica Ruiz
erica.ruiz@itson.edu.mx

¹ Instituto Tecnológico de Sonora, 5 de Febrero 818, Ciudad Obregón, Mexico

² Rochester Institute of Technology, 1 Lomb Memorial Drive, Rochester 14623, NY, USA

³ Universidad del Caribe, SM. 78, Mza. 1, Lote 1, Esq. Frac. Tabachines, 77528 Cancún, Mexico

The practical feasibility of a space–time code depends on the complexity of the decoder. The maximum-likelihood (ML) decoder is optimum, but its complexity increases exponentially with the constellation size and the number of antennas. On the other hand, the DSTTD code is a linear dispersion code and, as such, can be decoded using the same low-complexity, ordered decision feedback algorithm developed for V-BLAST [10,14]; however, its performance is quite suboptimal. A number of detectors have been proposed, with less complexity than ML but better performance than ordered decision feedback equalizers. Some of these algorithms were designed for V-BLAST [15,16]. Others are general near-ML algorithms, such as the sphere decoder [17]. All of them can be easily adapted to DSTTD.

Recently, tree-search algorithms have been applied to STBC decoding [18]; these have enabled near-ML decoding performance with reduced complexity compared to the sphere decoder [8,19–22]. Good results have been obtained with decoders based on the M -algorithm combined with the QR decomposition of the channel matrix. The reduction in complexity is obtained by reducing the number of distances calculated [19] and by exploiting the structure of the channel matrix [23]. These algorithms have been successfully applied to different STBC architectures, including DSTTD [24,25].

In this paper, we propose a new decoding algorithm for DSTTD that achieves near-ML performance and exhibits lower complexity than other known decoders. The new algorithm builds on ideas presented in the past. Like the decoders inspired by [23], we exploit the structure of the QR decomposition of the channel matrix to simplify the ML problem. We perform a tree search similar to that of [19] and [22], with an improved search order that allows the decoder to find the optimum solution in fewer iterations. Furthermore, the size of the search performed by the proposed decoder can be easily constrained to impose a maximum limit to the complexity, in many cases with negligible impact on its error performance. Finally, we divide the candidate search into three independent searches that can be executed concurrently, which enables a fast hardware implementation.

The algorithm, as presented, is adapted to work exclusively in the detection of DSTTD. However, with a slight modification it can also detect a two-layer hybrid code where the second layer consists of a spatial antenna (for a total of three transmitter antennas) [26].

The paper is organized as follows. In Sect. 2, we present an analysis of the DSTTD space–time code and show that its capacity is just slightly below that of the underlying MIMO channel. In Sect. 3, we present an overview of existing decoding algorithms for DSTTD. In Sect. 4 we make a detailed presentation of the proposed algorithm. Simulation results, including error rates and complexity, as well as a comparison with other algorithms, are presented in Sect. 5. Finally, we present our conclusions in Sect. 6.

Table 1 DSTTD space–time mapping

Time	Antenna 1	Antenna 2	Antenna 3	Antenna 4
T	s_1	s_2	s_3	s_4
$T + 1$	s_2^*	$-s_1^*$	s_4^*	$-s_3^*$

2 The double space–time transmit diversity linear dispersion code

A space–time block code (STBC) is a mapping from a vector of n_s information-bearing symbols s_i , $i = 1, 2, \dots, n_s$, to a $n_t \times T$ space–time code matrix $\mathbf{S}$, that specifies how symbols are spread over n_t antennas and T time intervals. The double space–time transmit diversity (DSTTD) linear space–time block code transmits $n_s = 4$ complex symbols over $T = 2$ symbol intervals and $n_t = 4$ transmit antennas [11]. The DSTTD space–time code matrix $\mathbf{S}$ is given by

$$\mathbf{S} = \sum_{n=1}^{n_s} (\text{Re}(s_n) A_n + j\text{Im}(s_n) B_n), \quad (1)$$

where the dispersion matrices $A_1, \dots, A_4$ and $B_1, \dots, B_4$ (of size $n_t \times T$) are defined as

$$\begin{aligned} A_1 &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}; & A_2 &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}; \\ A_3 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & -1 \end{bmatrix}; & A_4 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}; \\ B_1 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}; & B_2 &= \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}; \\ B_3 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}; & B_4 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & -1 \\ 1 & 0 \end{bmatrix}. \end{aligned}$$

Note that, since we assume that the transmitter has no channel knowledge, this code allocates the same average power to each transmitter antenna and each symbol. The DSTTD time–space mapping is summarized in Table 1.

We assume a rich-scattering, Rayleigh wireless channel with flat and slow fading, where the channel between transmitter antenna j and receiver antenna i can be modeled as a complex Gaussian gain $h_{ij} \sim \mathcal{C}(0, 1)$ of zero mean and variance 0.5 per dimension. This gain remains constant for several symbol intervals, after which it changes to a new

Fig. 1 Block diagram representation of a DSTTD system

independent realization. The overall channel can be modeled as a random matrix $\mathbf{H}$ of size $n_r \times n_t$. The receiver is assumed to have perfect channel state information, obtained using techniques such as those described in [27].

Further assumptions are that all the antennas transmit information symbols from the same M -QAM constellation, that the receiver is perfectly synchronized to the transmitter, that each receiver antenna is also subject to additive white Gaussian noise of zero mean and power spectral density $N_0/2$ per dimension. A block diagram of a DSTTD system is shown in Fig. 1.

2.1 Analysis of DSTTD: mutual information and diversity order

The DSTTD code is equivalent to two “stacked” Alamouti 2×1 codes. Each Alamouti code can be interpreted as a separate layer in a spatial multiplexing code; as such, it belongs to the category of hybrid space–time codes [10]. Hybrid codes are ad-hoc combinations of layered space–time codes, which may potentially achieve maximum spatial multiplexing gain, and (quasi) orthogonal codes, which achieve maximum diversity gain. These codes are interesting because, under certain conditions, they offer larger diversity gain than spatial multiplexing codes, and larger transmission rates than orthogonal codes. In this sense, hybrid codes’ diversity gain and rate can be designed to lie on intermediate points of the diversity–multiplexing trade-off curve described in [2]. At the same time, their structure allows for low complexity decoding (see e.g. [26]).

In the rest of this section we describe some properties of the DSTTD code, with the aim of showing that it offers a diversity–multiplexing trade-off that is close to optimal. Receiver algorithms are studied in subsequent sections.

Note that the DSTTD code has rate $R = n_s/T = 2$. It is not orthogonal, since $\mathbf{S}\mathbf{S}^H \neq \sum |s_n|^2 \mathbf{I}$. In contrast, a 2×1 Alamouti code has rate $R = 1$ and is orthogonal.

We now calculate the mutual information of DSTTD. The ergodic capacity of a 4×2 MIMO system is given by

$$C(\mathbf{H}) = E_{\mathbf{H}} \left[\log_2 \det \left(\frac{\text{SNR}}{4} \mathbf{H}\mathbf{H}^H \right) \right], \quad (2)$$

Fig. 2 Capacity of a 4×2 MIMO channel compared to the mutual information of DSTTD. $\text{SNR} = P/(\sigma^2 n_t)$, where P is total radiated power, n_t is the number of transmitter antennas, and σ^2 is the noise power per receiver antenna. The information rate is measured by channel use; DSTTD uses the channel two times per STBC symbol

where $E_{\mathbf{H}}(\cdot)$ is the expectation over $\mathbf{H}$ and $\mathbf{H}^H$ is the Hermitian conjugate of $\mathbf{H}$.

The mutual information $M(\mathbf{H})$ of DSTTD can be expressed in terms of the channel matrix $\mathbf{H}$ and the linear dispersion matrices. Let

$$\begin{aligned} \mathbf{F}_a &= [\text{vec}(\mathbf{H}\mathbf{A}_1) \cdots \text{vec}(\mathbf{H}\mathbf{A}_{n_s})] \\ \mathbf{F}_b &= [\text{vec}(\mathbf{H}\mathbf{B}_1) \cdots \text{vec}(\mathbf{H}\mathbf{B}_{n_s})] \\ \mathbf{F} &= [\mathbf{F}_a \quad \mathbf{F}_b]. \end{aligned}$$

Then, the mutual information can then be expressed as

$$M(\mathbf{H}) = \frac{1}{4} E_{\mathbf{H}} \left[\log_2 \det \left(\mathbf{I} + \frac{\text{SNR}}{4} \text{Re}(\mathbf{F}^H \mathbf{F}) \right) \right], \quad (3)$$

where the factor $1/4$ normalizes for two channel uses.

The capacity of a 4×2 channel is compared to the mutual information of DSTTD in Fig. 2. It can be seen that, for a signal to noise ratio of around 15 dB, the mutual information is around 1 dB below the capacity. The difference between them increases for larger SNR, but it can be concluded that DSTTD achieves a large fraction of the channel capacity.

Regarding the diversity gain of DSTTD, it can be calculated as follows. Let $\mathbf{S}$ and $\mathbf{W}$ be two different code matrices, and let $\mathbf{D} = \mathbf{S} - \mathbf{W}$. The diversity gain of DSTTD is equal to the rank of $\mathbf{D}$ times the number of receiver antennas. The difference matrix $\mathbf{D}$ is equal to

$$\mathbf{S} - \mathbf{W} = \begin{bmatrix} s_1 - w_1 & s'_2 - w'_2 \\ s_2 - w_2 & -s'_1 - w'_1 \\ s_2 - w_3 & s'_4 - w'_4 \\ s_4 - w_4 & -s'_3 - w'_4 \end{bmatrix} \quad (4)$$

and its rank is equal to 2. The diversity gain of the code is then equal to 4. This is double that of the Alamouti 2×1 code.

2.2 System equation

The received signal can be represented as a matrix $\mathbf{Y}$ of size $n_r \times T$ given by

$$\mathbf{Y} = \mathbf{H}\mathbf{S} + \mathbf{N}, \quad (5)$$

where $\mathbf{N}$ is a matrix of noise samples. Following the conventional analysis for STBC, we can vectorize the expression for the received symbols as

$$\begin{bmatrix} y_{11} \\ y_{12}^* \\ y_{21} \\ y_{22}^* \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ -h_{12}^* & h_{11}^* & -h_{14}^* & h_{13}^* \\ h_{21} & h_{22} & h_{23} & h_{24} \\ -h_{22}^* & h_{21}^* & -h_{24}^* & h_{23}^* \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} + \begin{bmatrix} n_{11} \\ n_{12}^* \\ n_{21} \\ n_{22}^* \end{bmatrix}. \quad (6)$$

Note that the 4×2 DSTTD system is equivalent to a 4×1 system where the channel matrix has a specific structure. By defining $\mathbf{H}_a$ as the channel matrix in Eq. 6, and $\mathbf{s} = [s_1 \ s_2 \ s_3 \ s_4]^T$, we can write the system's equation as

$$\mathbf{y} = \mathbf{H}_a \mathbf{s} + \mathbf{n}. \quad (7)$$

The system equation with this redefined channel matrix plays an important role in the development of receiver algorithms described in the next section.

3 Decoding algorithms for DSTTD

In the previous section we showed that the DSTTD space-time code has the potential for large spatial multiplexing and diversity gains compared to the underlying Alamouti 2×1 layers. In this section, we present several decoding algorithms that are tailored to the DSTTD code. First, we present the optimal maximum-likelihood decoder. Then, we describe several sub-optimal decoders that have low complexity, making them attractive for hardware or software implementation.

The algorithms presented below all follow three common steps: first, the system equations are rewritten to obtain a more convenient system representation; second, the QR decomposition of the channel matrix is calculated, and finally the actual decoding is performed. The structure of the matrix $\mathbf{R}$ is exploited to reduce the number of operations performed.

3.1 Maximum likelihood decoder

Assume that the vector $\mathbf{y}$ is received over two symbol periods as described in Eq. (7). The maximum-likelihood detection of the transmitted symbol $\mathbf{s}$ is given by

$$\hat{\mathbf{s}} = \underset{\mathbf{x} \in \Omega^4}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{H}_a \mathbf{x}\|^2 \quad (8)$$

where Ω is the signal constellation and Ω^4 is the set of all possible transmitted vectors. We may exploit the structure of the channel matrix to simplify this problem. Let $\mathbf{H}_a = \mathbf{Q}\mathbf{R}$ be the QR decomposition of $\mathbf{H}_a$, where $\mathbf{Q}$ is a unitary matrix and $\mathbf{R}$ is upper triangular. Then, if we multiply the received vector $\mathbf{y}$ by $\mathbf{Q}^H$, we obtain the modified vector

$$\begin{aligned} \tilde{\mathbf{y}} &= \mathbf{Q}^H \mathbf{y} \\ &= \mathbf{R}\mathbf{s} + \tilde{\mathbf{n}}. \end{aligned} \quad (9)$$

The statistical properties of the noise don't change, because $\mathbf{Q}$ is unitary. The structure of the channel matrix $\mathbf{H}_a$ results in a matrix $\mathbf{R}$ with the following structure [10]:

$$\mathbf{R} = \begin{bmatrix} R_{11} & 0 & R_{13} & R_{14} \\ 0 & R_{11} & -R_{14}^* & R_{13}^* \\ 0 & 0 & R_{33} & 0 \\ 0 & 0 & 0 & R_{33} \end{bmatrix}. \quad (10)$$

The ML detector can then be stated as:

$$\hat{\mathbf{s}} = \underset{\mathbf{x} \in \Omega^4}{\operatorname{argmin}} \sum_{i=1}^4 D_i^2 \quad (11)$$

where D_i^2 , $i = 1, 2, 3, 4$, are given by:

$$\begin{aligned} D_1^2 &= |\tilde{y}_1 - R_{11}x_1 - R_{13}x_3 - R_{14}x_4|^2 \\ D_2^2 &= |\tilde{y}_2 - R_{11}x_2 + R_{14}^*x_3 - R_{13}^*x_4|^2 \\ D_3^2 &= |\tilde{y}_3 - R_{33}x_3|^2 \\ D_4^2 &= |\tilde{y}_4 - R_{33}x_4|^2. \end{aligned} \quad (12)$$

In general, a brute-force approach to solving Eq. (11) requires the calculation and comparison of $|\Omega|^4$ metrics. However, careful analysis of Eq. (12) reveals that only $|\Omega|^2$ metric calculations are needed. The reason is that fixing the values of x_3 and x_4 allows estimation of x_1 and x_2 ; this means that iteration over other values of x_1 and x_2 is not necessary [19,25]. The complete process required by the ML detector is summarized in Algorithm (1). In this algorithm, we denote the i -th symbol in the constellation by $\Omega(i)$, and $\mathcal{Q}[\cdot]$ denotes a hard decision on a symbol.

This complexity reduction is intuitively satisfactory; the orthogonality of each of the underlying layers results in

a simplified detection problem. However, as it could be expected, the fact that the code is not truly orthogonal results in off-diagonal elements in $\mathbf{R}$ that do not allow the extreme decoding simplicity of orthogonal codes.

Algorithm 1 Maximum likelihood detection

Input: $\mathbf{R}, \tilde{\mathbf{y}}, \Omega$

Output: Optimum $\hat{\mathbf{s}}$

```

1: Let  $D = \infty$ 
2: for  $i = 1$  to  $|\Omega|$  do
3:   Let  $x_3 = \Omega(i)$ 
4:   for  $j = 1$  to  $|\Omega|$  do
5:     Let  $x_4 = \Omega(j)$ 
6:     Let  $x_1 = \mathcal{Q} \left[ \frac{\tilde{y}_1 - R_{13}x_3 - R_{14}x_4}{R_{11}} \right]$ 
7:     Let  $x_2 = \mathcal{Q} \left[ \frac{\tilde{y}_2 + R_{14}^*x_3 - R_{13}^*x_4}{R_{11}} \right]$ 
8:     Calculate  $D_T^2 = \sum_{i=1}^4 D_i^2$  using Eq. (12).
9:     if  $D_T^2 < D$  then
10:       Let  $D = D_T^2, \hat{\mathbf{s}} = \mathbf{x}$ 
11:     end if
12:   end for
13: end for
14: Return  $\hat{\mathbf{s}}$ .
```

3.2 OSIC detector using sorted QR decomposition

The ordered, successive interference cancellation (OSIC) detector, coupled with the sorted QR decomposition, results in a very low-complexity but sub-optimal detector. The sorted QR decomposition calculates a triangular matrix $\mathbf{R}$, a unitary matrix $\mathbf{Q}$ and a permutation vector $\mathbf{p}$, such that $\mathbf{H}_p = \mathbf{QR}$, where $\mathbf{H}_p$ is the channel matrix $\mathbf{H}_a$ with its columns reordered according to $\mathbf{p}$.

The reordering of the channel matrix results in matrix $\mathbf{R}$ with rows are ordered from higher to lower signal-to-noise ratio [15]. Then, symbols are estimated in sequence, from lower stream to higher stream; in each layer, the interference from previously-estimated symbols is subtracted. Assuming that all previous decisions are correct, the interference of previous symbols can be perfectly canceled at each step. For DSTTD, the OSIC detector calculates the symbol estimates $\hat{\mathbf{x}} = [\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4]^T$ as [10]:

$$\begin{aligned}
 \hat{x}_4 &= \mathcal{Q} \left[\frac{\tilde{y}_4}{R_{33}} \right] \\
 \hat{x}_3 &= \mathcal{Q} \left[\frac{\tilde{y}_3}{R_{33}} \right] \\
 \hat{x}_2 &= \mathcal{Q} \left[\frac{\tilde{y}_2 + R_{14}^* \hat{x}_3 - R_{13}^* \hat{x}_4}{R_{11}} \right] \\
 \hat{x}_1 &= \mathcal{Q} \left[\frac{\tilde{y}_1 - R_{13} \hat{x}_3 - R_{14} \hat{x}_4}{R_{11}} \right].
 \end{aligned} \tag{13}$$

Finally, the estimate $\hat{\mathbf{s}}$ is obtained by reordering $\hat{\mathbf{x}}$ according to the permutation vector $\mathbf{p}$. A similar algorithm, also based on reordering the channel matrix according to the norm of its columns, is presented in [14].

3.3 Near-ML detector based on improved OSIC algorithm

The OSIC detection algorithm for DSTTD scheme has very low complexity, but sequential instead of joint detection means that, in many cases, the optimum symbol vector is discarded during the vector nulling process. In [19] an efficient scheme based on OSIC detection was proposed to improve upon its error-rate performance by finding better starting points for further searches. In particular, it defines the metric $D_\theta^2 = f(D_1^2, D_2^2) + D_3^2 + D_4^2$, and explores vectors that were discarded by the OSIC detector but may have, in fact, a better metric. The function f may be chosen among $\max()$, $\min()$ and a weighted average; each one has slightly different performance and complexity properties. In general, though, this detector achieves a significant reduction in complexity compared to optimal ML algorithms such as the sphere decoder, because in practice few additional candidate vectors are examined, but the search usually includes the optimum solution.

3.4 Decoders based on the M -algorithm

The ML solution to Eq. (11) may be expressed as a search in a tree. Symbol s_4 sits at the root of the tree, and it branches to each possible value of s_3 , and so on successively to s_1 . Each branch is assigned a distance metric, and the symbols with smallest overall distance are selected as the optimum solution [28].

The M -algorithm is a breadth-first, sorted tree search algorithm that may be adapted for MIMO detection [16,18]. The algorithm reduces the search complexity by storing only the best M branches at a time. Small values for M result in low complexity, but quite sub-optimal performance; as M increases, the complexity also increases but the algorithm's performance gets closer to the ML decoder.

This algorithm has been adapted to the DSTTD code [24]. The main idea is to choose the best M candidates in the estimation of the symbols $\hat{s}_3$ and $\hat{s}_4$; then, the search for $\hat{s}_2$ and $\hat{s}_1$ is limited to M^2 candidates. This results in a marked reduction in complexity without a large sacrifice in optimality. The decoding process is presented in Algorithm (2).

3.5 The LC maximum likelihood detector

A quasi-orthogonal space-time block code (QSTBC) is one for which the code matrix product $\mathbf{S}\mathbf{S}^H$ has a small number of off-diagonal elements. In general, this results in a maximum-

Algorithm 2 DSTTD decoding based on the QRD-M algorithm

Input: $\mathbf{R}$, M , Ω , $\tilde{\mathbf{y}}$

Output: Symbol estimate $\hat{\mathbf{s}}$

- 1: Calculate and sort the metrics $|\tilde{y}_4 - R_{33}x_4|^2$, for all possible values of $x_4 \in \Omega$. Store the values of x_4 producing the best M metrics in $\mathcal{X}_4$.
- 2: Calculate and sort the metrics $|\tilde{y}_3 - R_{33}x_3|^2$, for all possible values of $x_3 \in \Omega$. Store the values of x_3 producing the best M metrics in $\mathcal{X}_3$.
- 3: Let $q = 1$ and $D = \infty$.
- 4: **for** $i = 1$ **to** M **do**
- 5: Let $x_3 = \mathcal{X}_3(i)$
- 6: **for** $j = 1$ **to** M **do**
- 7: Let $x_4 = \mathcal{X}_4(i)$
- 8: $x_1 = \mathcal{Q} \left[\frac{\tilde{y}_1 - R_{13}x_3 - R_{14}x_4}{R_{11}} \right]$
- 9: $x_2 = \mathcal{Q} \left[\frac{\tilde{y}_2 + R_{14}^*x_3 - R_{13}^*x_4}{R_{11}} \right]$
- 10: Calculate $D_T^2 = \sum_{i=1}^4 D_i^2$ using Eq. (12).
- 11: **if** $D_T^2 < D$ **then**
- 12: Let $D = D_T^2$ and $\hat{\mathbf{s}} = \mathbf{x}$.
- 13: **end if**
- 14: **end for**
- 15: **end for**
- 16: Return $\hat{\mathbf{s}}$.

likelihood decoder with much more complexity than that of an OSTBC; however, in some cases, low-complexity decoders can be found. Consider the QSTBC code specified by the following code matrix:

$$\mathbf{S} = \begin{bmatrix} s_1 & -s_2^* & s_3 & -s_4^* \\ s_2 & s_1^* & s_4 & s_3^* \\ s_3 & -s_4^* & s_1 & -s_2^* \\ s_4 & s_3^* & s_2 & s_1^* \end{bmatrix} \quad (14)$$

Assuming $n_R = 1$, the QR decomposition of the equivalent channel matrix $\mathbf{H} = \mathbf{QR}$ results in $\mathbf{Q}$ equal to the identity matrix and

$$\mathbf{R} = \begin{bmatrix} R_{11} & 0 & R_{13} & 0 \\ 0 & R_{11} & 0 & R_{13} \\ 0 & 0 & R_{33} & 0 \\ 0 & 0 & 0 & R_{33} \end{bmatrix}. \quad (15)$$

In [23], a very low-complexity decoder with near ML performance was proposed. This decoder selects the estimated symbol vector $\hat{\mathbf{s}}$ that satisfies

$$\hat{\mathbf{s}} = \underset{\mathbf{x} \in \Omega^4}{\operatorname{argmin}} \sum_{i=1}^4 D_i^2, \quad (16)$$

where $D_T^2 = \sum_{i=1}^4 D_i^2$ and the D_i^2 are given by:

$$\begin{aligned} D_1^2 &= |\tilde{y}_1 - R_{11}x_1 - R_{13}x_3|^2 \\ D_2^2 &= |\tilde{y}_2 - R_{11}x_2 - R_{13}x_4|^2 \\ D_3^2 &= |\tilde{y}_3 - R_{33}x_3|^2 \\ D_4^2 &= |\tilde{y}_4 - R_{33}x_4|^2. \end{aligned} \quad (17)$$

The decoding algorithm has two interesting complexity-reducing properties. One is that the detection of s_1 and s_3 can be done concurrently with that of s_2 and s_4 , since these two detection steps are completely independent. The second is that the symbols $\mathbf{x} \in \Omega$ are sorted in a specific way so that not all of them need to be tested using Eq. (16).

Note the similarity of the matrix $\mathbf{R}$ in Eq. (15) with the corresponding DSTTD matrix in Eq. (10); likewise, compare the DSTTD decoding metrics in Eq. (12) and the LC-ML decoding procedure in Eq. (17). This would suggest that the strategies presented in [23] might also be applicable to DSTTD decoding. This is explored in the next section.

4 Proposed near-ML decoding algorithm

In this section, we present a new decoding algorithm for DSTTD. The aim of the decoder is to find the optimum solution to the maximum likelihood equation (11), using the distances calculated in Eq. (12). Consider a breadth-first, tree search decoder that operates by executing the following steps:

1. Sort the elements of Ω in order of increasing distance D_3^2 and store them in vector $\mathbf{x}_3$. Repeat for distance D_4^2 , storing the result in $\mathbf{x}_4$.
2. Using $\hat{x}_3 = \mathbf{x}_3[1]$ and $\hat{x}_4 = \mathbf{x}_4[1]$, find symbols $\hat{x}_1, \hat{x}_2 \in \Omega$ that minimize D_1^2 and D_2^2 . Store the current total distance $D_T^2 = \sum_i D_i^2$.
3. Iterate over all remaining elements of $\mathbf{x}_3$ and $\mathbf{x}_4$. For each pair $\hat{x}_3 \in \mathbf{x}_3, \hat{x}_4 \in \mathbf{x}_4$, find the pair $\hat{x}_1$ and $\hat{x}_2$ that minimizes D_T^2 .
4. Return the symbols that produce the smallest total distance.

Note that this procedure can be improved in several ways. First, the iteration in step 3 does not need to be over all symbols in $\mathbf{x}_3$ and $\mathbf{x}_4$. Since the symbols are ordered in order of increasing distance, the likelihood of a pair of symbols $\hat{x}_3, \hat{x}_4$ being in the optimal solution decreases as the algorithm progresses. This suggests that some pairs of symbols may be skipped, and that the search can be stopped early, according to some criterion, resulting in a significant reduction in complexity. The criterion should maximize the probability of the optimum solution being included in the search, while minimizing the number of symbol pairs examined.

Note, as well, that the iteration in step 3 can be divided into a number of independent, concurrent iterations. This means that the decoder is amenable to fast implementations, either in hardware or in software, using multiple processors. While execution in a single processor requires a roughly similar amount of memory as other proposed decoders, concurrent execution of the algorithm may involve a small memory usage penalty, because each process requires its own local metric storage.

The proposed algorithm builds on the decoders described in Sect. 3, which were first presented in [19], [23] and [24]. The decoder in [23] is designed for an ABBA code, which is similar to DSTTD but requires $T = 4$. In [19], a pool of candidate symbols is examined per iteration, whereas our proposal examines only one. In addition, [19] requires fine-tuning of a distance weighting function; no such adjustments are required in our proposal. Finally, the algorithm proposed in [24] has fixed complexity, and does not employ any heuristics for stopping the search early.

We present a more detailed description of the proposed algorithm in the next sub-sections; we also propose specific criteria for skipping symbols and for stopping the search. For clarity, we have divided the algorithm into three different stages: pre-processing, parallel candidate search, and post-processing.

4.1 Pre-processing

It can be seen in Eq. (12) that the SNR of $\hat{x}_3$ and $\hat{x}_4$ depends on R_{33} . Since a correct initial estimate of these two symbols reduces the search complexity, their SNR should be maximized. This is accomplished by re-ordering the columns of the channel matrix as described in Algorithm (3). Note that the column re-ordering does not affect the structure of matrix R .

Algorithm 3 Pre-processing algorithm

Input: Channel matrix $\mathbf{H}_a$

Output: Re-ordered channel matrix, Boolean flag *reverse*

```

1: Let reverse := false
2: if  $\|\mathbf{H}_a[:, 3]\|^2 < \|\mathbf{H}_a[:, 1]\|^2$  then
3:   Let reverse := true
4:   Exchange columns 1 and 3, and columns 2 and 4, of
      $\mathbf{H}_a$ 
5: end if

```

4.2 Candidate search

This is the main portion of the algorithm, where candidate solutions are explored. It is divided into two parts. The first part, presented in Algorithm (4), calculates required quanti-

ties and obtains an initial estimate. The following conventions are used:

1. Variables in bold represent either vectors (lowercase) or matrices (uppercase). $\boldsymbol{\Omega}$ is a vector whose elements are the constellation symbols.
2. Arithmetic on vectors is performed element by element.
3. Vector indexing is indicated using square brackets.
4. The function $\text{findmin}(\mathbf{x})$ returns a tuple of the smallest element in vector $\mathbf{x}$ and its corresponding index.
5. The function $\text{sortperm}(\mathbf{x})$ returns a vector of indices to the elements of $\mathbf{x}$ in increasing order.
6. The instruction “break” exits all nested loops.

Algorithm 4 Candidate search – Initialization

Input: (Possibly re-ordered) channel matrix $\mathbf{H}_a$, received vector $\mathbf{y}$, constellation vector $\boldsymbol{\Omega}$, search limiter N_c .

Output: (Possibly re-ordered) estimate vector $\hat{\mathbf{s}}$.

```

1: Let  $\mathbf{Q}, \mathbf{R}$  be the QR decomposition of  $\mathbf{H}_a$ 
2: Let  $\mathbf{v} = \mathbf{Q}^H \mathbf{y}$ 
3: Let  $\mathbf{r}_1 \mathbf{x} = R_{11} \boldsymbol{\Omega}$ 
4: Let  $\mathbf{r}_3 \mathbf{x} = R_{33} \boldsymbol{\Omega}$ 
5: Let  $\mathbf{d} = \|\mathbf{v}[3] - \mathbf{r}_3 \mathbf{x}\|^2$ 
6: Let  $\mathbf{i}_3 = \text{sortperm}(\mathbf{d})$ 
7: Let  $\mathbf{x}_3 = \boldsymbol{\Omega}[\mathbf{i}_3]$ 
8: Let  $\mathbf{d}_3 = \mathbf{d}[\mathbf{i}_3]$ 
9: Let  $\hat{x}_3 = \mathbf{x}_3[1]$ 
10: Let  $d_3 = \mathbf{d}_3[1]$ 
11: Let  $\mathbf{d} = \|\mathbf{v}[4] - \mathbf{r}_3 \mathbf{x}\|^2$ 
12: Let  $\mathbf{i}_4 = \text{sortperm}(\mathbf{d})$ 
13: Let  $\mathbf{x}_4 = \boldsymbol{\Omega}[\mathbf{i}_4]$ 
14: Let  $\mathbf{d}_4 = \mathbf{d}[\mathbf{i}_4]$ 
15: Let  $\hat{x}_4 = \mathbf{x}_4[1]$ 
16: Let  $d_4 = \mathbf{d}_4[1]$ 
17:  $[d_1, i_1] = \text{findmin}(\|\mathbf{v}[1] - \mathbf{r}_1 \mathbf{x} - R_{13} \hat{x}_3 - R_{14} \hat{x}_4\|^2)$ 
18: Let  $\hat{x}_1 = \boldsymbol{\Omega}[i_1]$ 
19:  $[d_2, i_2] = \text{findmin}(\|\mathbf{v}[2] - \mathbf{r}_1 \mathbf{x} + R_{14}^* \hat{x}_3 - R_{13}^* \hat{x}_4\|^2)$ 
20: Let  $\hat{x}_2 = \boldsymbol{\Omega}[i_2]$ 
21: Let  $d = d_1 + d_2 + d_3 + d_4$ .
22: Let  $d_{m1} = d_{m2} = d_{m3} = d$ .

```

The initialization process is very similar to the OSIC decoder: the initial symbol estimates are those that individually minimize the metrics D_i^2 in Eq. (11). After initialization, the estimate is refined in three iterative processes than can run concurrently. Each iteration examines a different subset of the available candidates, as defined in Table 2. The purpose of this partition is to share work as equally as possible between the three iterations, while allowing each iteration to examine its assigned symbols from most to least promis-

Table 2 Index ordering of the three search iterations. N_c is a user-supplied parameter to constrain the search size

Iteration 1		Iteration 2		Iteration 3	
$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_3$	$\mathbf{x}_4$
2	1	1	2	2	2
3	1	1	3	3	3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
N_c	1	1	N_c	N_c	N_c
3	2	2	3		
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
N_c	2	2	N_c		
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
N_c	$N_c - 1$	$N_c - 1$	N_c		

ing. Furthermore, note that the search size is constrained by parameter N_c . If $N_c = |\Omega|$, then the search can occur over the entire symbol set.

Iteration 1 is described in detail in Algorithm (5). Note that the other two iterations are identical except for the different indexing order. Each iteration produces a symbol estimate $\hat{s}_i$ with distance d_{mi} , for $i = 1, 2, 3$. Symbol pairs $\hat{x}_3$ and $\hat{x}_4$ whose distances D_3^2 and D_4^2 are not better than previous ones are skipped, as specified in line 9. Furthermore, the search is stopped early if the condition in lines 9 and 19 are both met. This is the main reason for the algorithm's reduced complexity.

4.3 Post-processing

After all three iterations have finished, the estimate $\hat{s}_i$ with least distance d_i , $i = 1, 2, 3$ is selected. If the Boolean flag *reverse* is true, then elements 1 and 2 of $\hat{s}$ are swapped with elements 3 and 4. This concludes the decoding process.

5 Simulation results and analysis

To demonstrate the advantages of the proposed detector, we compare the bit error rate and complexity of several different detectors for DSTTD with 4×2 (4 transmit and 2 receive antennas). We present results with QPSK and 32-QAM modulation. In all cases, the channel block length was fixed to $L = 2$, and simulations were run until 1000 symbol errors were found. The BER is represented as a function of the per-bit signal-to-noise ratio E_b/N_0 .

The detectors used in the comparison are the following. The OSIC detector (Sect. 3.2, [15]) has the worst error rate, but it is included as the baseline for low complexity. The DSTTD-adapted QR M -algorithm with $M = 2$ (Sect. 3.4,

Algorithm 5 Candidate search – Iteration

```

1: skip := false
2: for  $n = 1 : N_c$  do
3:   Let  $\hat{x}_4 = \mathbf{x}_4[n]$ 
4:   Let  $d_4 = \mathbf{d}_4[n]$ 
5:   for  $m = (n + 1) : N_c$  do
6:     Let  $\hat{x}_3 = \mathbf{x}_3[m]$ 
7:     Let  $d_3 = \mathbf{d}_3[m]$ 
8:     Let  $d_{34} = d_3 + d_4$ 
9:     if  $d_{34} < d_{m1}$  then
10:      Let  $[d_1, i_1] = \text{findmin}(\|\mathbf{v}[1] - \mathbf{r}_1\mathbf{x} - R_{13}\hat{x}_3 - R_{14}\hat{x}_4\|^2)$ 
11:      Let  $\hat{x}_1 = \Omega[i_1]$ 
12:      Let  $[d_2, i_2] = \text{findmin}(\|\mathbf{v}[2] - \mathbf{r}_1\mathbf{x} + R_{14}^*\hat{x}_3 - R_{13}^*\hat{x}_4\|^2)$ 
13:      Let  $\hat{x}_2 = \Omega[i_2]$ 
14:      Let  $d = d_1 + d_2 + d_{34}$ 
15:      if  $d < d_{m1}$  then
16:        Let  $d_{m1} = d$ 
17:        Let  $\hat{s}_1 = [\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4]$ 
18:      end if
19:    else if  $n > 1$  and  $m == n + 1$  then
20:      break
21:    end if
22:  end for
23: end for

```

Fig. 3 Bit error rate performance of several DSTTD decoders compared to the proposed algorithm, using QPSK

[24] is also included. Next is the near-ML, improved OSIC detector described in Sect. 3.3 [19]; we have chosen the metric function $f()$ to be equal to $\max()$, since it offers the best error performance (with a slight computational complexity increase). Finally, for QPSK, we include the brute-force ML

Fig. 4 Bit error rate performance of several DSTTD decoders compared to the proposed algorithm, using 32-QAM

Fig. 5 Decoding complexity of existing DSTTD tree-search decoders compared to the proposed algorithm, using QPSK

detector (Algorithm 1) to bracket the optimum error performance.

In Figs. 3 and 4, we compare the BER performance of our proposal to that of the detectors listed above, for QPSK and 32-QAM modulations. It can be seen that both our proposed algorithm and the improved OSIC detector essentially achieve the optimum error performance. For QPSK, the M -algorithm detector is approximately 1 dB worse at $\text{BER} = 10^{-3}$ and it diverges for increasing SNR; the OSIC detector is 4 dB worse at the same point. A similar margin exists for 32-QAM.

In Figs. 5 and 6, we compare the computational complexity of the different detectors. This complexity is measured in terms of the average number of $\hat{x}_3, \hat{x}_4$ pairs examined per

Fig. 6 Decoding complexity of existing DSTTD tree-search decoders compared to the proposed algorithm, using 32-QAM

each decoded space-time symbol. In particular, for the proposed algorithm this is equivalent to averaging how many times lines 10–18 of Algorithm (5) are executed in the decoding of a single space-time symbol. This complexity measure is useful because the improved OSIC detector, the M -algorithm and the proposed algorithm perform a similar number of arithmetic and logic operations when examining a candidate pair. Note that since Algorithm (4) is always executed, the minimum number of examined pairs is one.

It can be seen that the proposed algorithm shows significantly reduced complexity compared to the improved OSIC detector, at essentially the same BER performance. The difference between the algorithms increases as the constellation order increases. It is interesting to note that, for 32-QAM, the complexity of the improved OSIC detector does not always decrease smoothly with increasing SNR. We attribute this behavior to a large variance in the number of examined pairs at low SNR. The proposed algorithm does not seem to exhibit this variance and its complexity is more easily predictable.

One feature of the proposed algorithm is that the number of examined pairs can be limited by the parameter N_c ; this can be seen in lines 2 and 5 in Algorithm (5). Limiting the search like this has the effect of reducing the detector's complexity, at the cost of possibly omitting the optimum solution from the search. In Figs. 7 and 8, we show the bit error rate for $N_c = 2, 3, 4$ in the case of QPSK, and $N_c = 4, 6, 8, 32$ for 32-QAM. In Figs. 9 and 10, we present the complexity for different values of N_c (recall that $N_c \leq |\Omega|$).

For QPSK modulation, limiting the search size has a measurable effect in the bit error rate. For low SNR, setting $N_c = 2$ or $N_c = 3$ results in a reduction in complexity; however, for high SNR, the complexity in all three cases is very similar and converges to the OSIC complexity. This result is

Fig. 7 Bit error rate performance of the proposed algorithm when the search size N_c is constrained to different values, using QPSK

Fig. 8 Bit error rate performance of the proposed algorithm when the search size N_c is constrained to different values, using 32-QAM

interesting, because it implies that, for high SNR, the algorithm initialization phase almost always finds the optimum solution; however, in some cases, it needs to consider further candidates to reach the ML performance.

For 32-QAM, results indicate that setting $N_c = 8$ is enough to achieve ML performance. We again see that, as SNR increases, the complexity tends to 1, albeit more slowly than for QPSK.

6 Conclusions

We have presented a receiver algorithm for the DSTTD hybrid space–time code with near-optimal bit error rate and substantially less complexity than other, existing near-ML

Fig. 9 Complexity of the proposed algorithm when the search size N_c is constrained to different values, using QPSK

Fig. 10 Complexity of the proposed algorithm when the search size N_c is constrained to different values, using 32-QAM

decoders. The presented decoder is an adaptation of the QRD-M tree search algorithm. After an initial estimate, three concurrent iterations search the tree of candidates and store the solution with least metric. The low complexity results from the decoder skipping over candidates that do not meet a specified criterion, as well as the search being stopped as soon as no promising candidates are left. The concurrent nature of the three iterations suggests the algorithm can be executed on multiple processors, increasing its operating speed.

Acknowledgements The present article was jointly funded by PFCE Projects and PROFAPI 2018.

Compliance with ethical standards

Conflicts of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Telatar, I. E. (1999). Capacity of multiple-antenna Gaussian channels. *European Transactions on Telecommunications*, 10(6), 585–595.
2. Zheng, L., & Tse, D. (2003). Diversity and multiplexing: A fundamental tradeoff in multiple-antenna channels. *IEEE Transactions on Information Theory*, 49(5), 1073–1096.
3. Foschini, G. J. (1996). Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas. *Bell Labs Technical Journal*, 1(2), 41–59.
4. Alamouti, S. M. (1998). A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16(8), 1451–1458.
5. Tarokh, V., Jafarkhani, H., & Calderbank, A. R. (2002). Space-time block codes from orthogonal designs. *IEEE Transactions on Information Theory*, 45(5), 1456–1467.
6. Hassibi, B., & Hochwald, B. M. (2002). High-rate codes that are linear in space and time. *IEEE Transactions on Information Theory*, 48(7), 1804–1824.
7. Cortez, J., Bazdresch, M., Torres, D., & Parra-Michel, R. (2007). An efficient detector for non-orthogonal Space-Time Block Codes with Receiver Antenna Selection. In: *IEEE 18th international symposium on personal, indoor and mobile radio communications (PIMRC)*. <https://doi.org/10.1109/PIMRC.2007.4394046>.
8. Yu, S.-J., Lee, E.-Y., & Song, H.-K. (2015). Combination of STBC and SM scheme with iterative detection in LTE systems. *Wireless Personal Communications*, 83(2), 1203–1211.
9. Soria, F., Garcia, J., & Barboza, F. (2015). Improved detection of SM-SMux signals for MIMO channels. *IEEE Latin America Transactions*, 13(1), 43–47.
10. Bazdresch, M., Cortez, J., Longoria, O., & Parra-Michel, R. (2012). A family of hybrid space-time codes for MIMO wireless communications. *Journal of Applied Research and Technology*, 19(2), 122–142.
11. Onggosanusi, E., Dabak, A., & Schmidl, T. (2002). High rate space-time block coded scheme: Performance and improvement in correlated fading channels. *IEEE Wireless Communications Networking Conference (WCNC)*. <https://doi.org/10.1109/WCNC.2002.993489>.
12. IEEE Standard for Local and Metropolitan Area Networks, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput, IEEE Std. 802.11n-2009 (2009).
13. IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems, Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1, IEEE Std. 802.16e-2005 (2006).
14. Kim, H., & Park, H. (2005). Efficient successive interference cancellation algorithms for the DSTTD system. In *16th IEEE international symposium on personal, indoor and mobile radio communications (PIMRC)*. <https://doi.org/10.1109/PIMRC.2005.1651399>.
15. Wübben, D., Böhnke, R., Rinas, J., Kühn, V., & Kammeyer, K. D. (2001). Efficient algorithm for decoding layered space-time codes. *Electronics Letters*, 37(22), 1348.
16. Wu, K., Sang, L., Xiong, C., Zhang, X., & Yang, D. (2009). Novel QRM-MLD algorithm for V-BLAST systems with permuted channel matrix. In *20th IEEE international symposium on personal, indoor and mobile radio communications (PIMRC)*. <https://doi.org/10.1109/PIMRC.2009.5449768>.
17. Hassibi, B., & Vikalo, H. (2005). On the sphere-decoding algorithm: I. Expected complexity. *IEEE Transactions on Signal Processing*, 53(8), 2806–2816.
18. de Jong, Y., & Willink, T. (2002). Iterative tree search detection for MIMO wireless systems. *IEEE Transactions on Communications*, 53(6), 930–935.
19. Kim, H., & Park, H. (2008). Approaching maximum-likelihood performance with reduced complexity for a double space-time transmit diversity system. *IET Communications*, 2(5), 682–689.
20. Kim, I., Park, Y. O., & Bang, Y. (2009). Very fast detection for rate-2 quasi-orthogonal STBCs. *IEEE Transactions on Wireless Communications*, 8(1), 95–101.
21. Lee, Y., & Shieh, H.-W. (2011). A simple layered space-time block nulling technique for DSTTD systems. *IEEE Communication Letters*, 15(12), 1323–1325.
22. Pai, H.-T. (2012). Adaptive detection for double STBCs based on QR decomposition. *IEEE Transactions on Vehicular Technology*, 61(3), 1182–1187.
23. Le, M.-T., Pham, V.-S., Mai, L., & Yoon, G. (2005). Low-complexity maximum-likelihood decoder for four-transmit-antenna quasi-orthogonal space-time block code. *IEEE Transactions on Communications*, 53(11), 1817–1821.
24. Cortez, J., Palacio, R., Ramirez-Pacheco, J. C., & Ruiz-Ibarra, E. (2015). A very low complexity near ML detector based on QRD-M algorithm for STBC-VBLAST architecture. In *7th IEEE Latin American conference on communications (LATINCOM)*. <https://doi.org/10.1109/LATINCOM.2015.7430123>.
25. Kim, H., & Park, H. (2007). New DSTTD transceiver architecture for low-complexity maximum-likelihood detection. In *IEEE wireless communications and networking conference (WCNC)*. <https://doi.org/10.1109/WCNC.2007.144>.
26. Cortez, J., Bazdresch, M., & Longoria-Gandara, O. (2017). A low-complexity near-ML detector for a $3 \times nR$ hybrid space-time code. In *IEEE 9th Latin-American conference on communications (LATINCOM)*. <https://doi.org/10.1109/LATINCOM.2017.8240182>.
27. Longoria-Gandara, O., & Parra-Michel, R. (2011). Estimation of correlated MIMO channels using partial channel state information and DPSS. *IEEE Transactions on Wireless Communications*, 10(11), 3711–3719. <https://doi.org/10.1109/TWC.2011.091411.101199>.
28. Anderson, J., & Mohan, S. (1984). Sequential coding algorithms: a survey and cost analysis. *IEEE Transactions on Communications*, 32(2), 169–176.

Joaquín Cortez received the M.Sc. and Ph.D. degrees in Electrical Engineering from CINVESTAV Guadalajara, Mexico, in 2001 and 2008 respectively. He is currently an electrical and electronics engineering professor at the Sonora Institute of Technology, in Mexico. His research interests include digital communications and digital signal processing.

Ramón Palacio is a professor of Software Engineering at the Sonora Institute of Technology, where he is a member of the Networks and Intelligent Systems research group. His research interests include Computer Networks, Human-Computer Interaction (HCI), and Software Engineering. With his research group, he conducts studies to gain a better understanding of current work practices and based on this understanding, designs, develops and evaluates appropriate technologies for diverse work environments. He holds a Ph.D. in Computer Science from Universidad Autónoma de Baja California (UABC), Mexico.

Miguel Bazdresch received a M.Sc. degree in electrical engineering from CINVESTAV, Mexico, in 1999, and a Ph.D. degree in electrical communications from the ENST, Paris, France, in 2004. He was with the Center for Semiconductor Technology, in Mexico, from 1993 to 1997, as a communications integrated circuits designer. He helped found TDCOM, a startup company that consulted in circuit design, in 1999. From 2005 to 2012, he was a Professor at the electrical engineering department

of the Western Technology and Higher Studies Institute (ITESO), in Guadalajara, Mexico. Since 2012, he has been as Assistant Professor in the electrical and telecommunications engineering technology department at the Rochester Institute of Technology, Rochester, New York. His research interests are in software-defined radio, algorithms and signal processing for wireless communications, and in electrical engineering education.

Erica Ruiz received M.Sc. Ph.D. degrees in Electronics and Telecommunications from CICESE Research Center, in Mexico, in 2000 and 2010 respectively. She is currently an electrical and electronics engineering professor at the Sonora Institute of Technology. Her research interests include sensor networks, design and performance evaluation of routing protocols, digital communications and human computer interaction.

Julio Ramírez is a researcher and full-time professor at Universidad del Caribe, Mexico. He received the B. Tech. degree in electronics engineering from Instituto Tecnológico de Mérida in 2002 and the M.Sc. and Ph.D. degrees in electrical engineering from CINVESTAV-IPN, Unidad Guadalajara, Mexico in 2005 and 2012. His research interests are in the study of scaling signals, fractal stochastic processes, non-extensive wavelet entropies, and telecommunications software design. He also

contributes in the areas of discrete mathematics, finance and economics, in which he does research on techniques to solve time-series analysis problems. Prof. Ramírez has served as guest editor and reviewer in many journals and conferences.