DIPLOMARBEIT

Wireless Access in Vehicular Environments with Soft Output MIMO Enhancements

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Diplom-Ingenieurs

unter der Leitung von
Univ.Prof. Dipl.-Ing. Dr.techn. Christoph F. Mecklenbräuker
Univ.Ass. Dipl.-Ing. Dr.techn. Veronika Shivaldova
Institute of Telecommunications

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von
Admir Eminić
Am Eisteich 33/5
2440 Reisenberg

Wien, im September 2015 _____

To Dženana & Vedad ♡.

Abstract

The IEEE 802.11p standard defines the physical and medium access control layers for vehicular communications. Performance evaluation has shown that vehicular environments set high demands on communication technologies since the channel conditions are rapidly changing. Creating a robust and reliable communication standard is a challenging task.

This thesis describes an Orthogonal Frequency Division Multiplexing (OFDM) based physical layer implementation, designed according to the IEEE 802.11p standard. Further it evaluates possible technologies that promise system improvements such as Multiple-Input Multiple-Output (MIMO), Space-Time Block Codes (STBCs) and the use of soft information. In the first chapter a brief overview of the standards for vehicular communication is given. In the second chapter we focus on the physical layer of the standard and its implementation in Matlab. Components of the system are introduced and an insight into the principles of their functionalities is given. As this implementation is realized as a Single-Input Single-Output (SISO) system it represents the base for further extensions. The first extension that is described is MIMO technology investigated in [1], which improves the system in terms of reliability and a higher throughput by exploiting the diversity. Objective of this thesis is to evaluate the possible system improvement by implementing a soft Sphere Decoder (SD) for MIMO systems. The meaning of soft information is described in the first part of Chapter 3, while the second part considers SD algorithms. The resulting implementation of a simulator with MIMO in combination with SD is used to evaluate the performance in various channel models. In Chapter 4 results of these simulations and their interpretation is presented.

Zusammenfassung

Der IEEE 802.11p Standard definiert die physikalische und die Sicherungsschicht für Fahrzeugkommunikation. Auswertungen der Leistungsfähigkeit haben gezeigt dass Fahrzeugumgebungen hohe Anforderungen an Kommunikationstechnologien stellen da sich die Zustände des Kommunikationskanals rapide ändern. Einen robusten und zuverlässigen Kommunikationsstandard zu schaffen ist eine herausfordernde Aufgabe.

Diese Diplomarbeit beschreibt eine auf OFDM basierende Implementierung der physikalischen Schicht, entworfen entsprechend dem IEEE 802.11p Standard. Des Weiteren werden mögliche Technologien wie MIMO, STBCs und die Ausnutzung der soft Information, von denen man sich Verbesserungen des Systems erhofft, evaluiert. Im ersten Kapitel ist ein kurzer Überblick der Standards für Fahrzeugkommunikation gegeben. Im zweiten Kapitel fokussieren wir uns auf die physikalische Schicht des Standards, d.h. IEEE 802.11p und dessen Implementierung in Matlab. Die einzelnen Komponenten des Systems werden vorgestellt und ein Einblick in die Prinzipien derer Funktionalität wird gegeben. Da diese Implementierung als SISO System realisiert ist, bildet sie die Basis für Erweiterungen. Die erste Erweiterung die beschrieben wird ist die MIMO Technologie, die in [1] untersucht wurde, welche das System im Sinne von Zuverlässigkeit und einem höheren Datendurchsatz verbessert indem sie die Diversität ausschöpft. Dass Ziel dieser Diplomarbeit ist eine eventuelle Verbesserung des Systems durch die Implementierung eines SD für MIMO Systeme zu untersuchen. Die Bedeutung von Soft Information ist im erten Teil von Kapitel 3 beschrieben, während der zweite Teil sich mit SD Algorithmen beschäftigt. Die resultierende Implementierung eines Simulators mit MIMO in Kombination mit SD wird benutzt um die Leistungsfähigkeit zu untersuchen. In Kapitel 4 werden die Resultate dieser Simulationen und deren Interpretierung präsentiert.

Acknowledgements

I would like to thank Professor Christoph Mecklenbräuker for giving me the opportunity to work on my diploma thesis.

Especially, I would like to thank Veronika Shivaldova for supervising me during my work and for proofreading this thesis. Her suggestions, encouragement and patience helped me to finish my thesis.

Furthermore, I am grateful for the support of my parents Nesima and Enes and their encouragement during all the hard times. I would like to give special thanks to the family Behremović for their financial and emotional support which made my studies possible at all and for giving me a warm home far away from home. I'd like to thank my wife Dženana and my son Vedad for their support, patience and sacrifice during the writing of this thesis.

Contents

1

Introduction

Wireless communications offer many possibilities and, regarding the past years evolution of mobile communications, it is a matter of course that they gain more and more ground in vehicular environment. Although promising, communication at high velocities and dynamic conditions entails also demanding challenges. Standardization organizations already have released some standards addressing vehicular communications. One of them is called Wireless Access in Vehicular Environments (WAVE) and represents an amendment to the IEEE 802.11 standard. This thesis considers a possible extension of this standard with soft output MIMO enhancements.

1.1 Vehicular Communication

Communication between vehicles opens a wide range of possible applications. Most important of them are traffic safety and traffic efficiency. Intelligent Transport Systems (ITS) aim to provide traffic participants with important information. By this way it is possible to avoid accidents by alerting drivers in dangerous situations and provide information about road conditions, everything in order to save human life. This is the traffic safety field which requires high reliability and low latency and is the most promising one. The traffic efficiency field, not less important, gives the opportunity to manage traffic and increase the use of traffic networks. According to a study of the German automobile club (ADAC) in 2014 in Germany there were about 475,000 congestions with an overall length of circa 960,000 kilometers, thus wasting billions of liters of fuel, not to mention the air pollution as a result of such scenarios. While the radio traffic service is usually broadcasted every half hour and is not sufficient enough, modern ITS could provide real-time data about traffic load on the route of a user and possible alternative routes to avoid traffic bottlenecks. As this example shows, efficient traffic management has a global impact. Another possible application field includes value added services. These are e.g. Points of Interest (POI) notification, local electronic commerce, parking management, high speed Internet access etc. [2]. By implementing such commercial services, which are attractive for users, the costly deployment of infrastructure could be financed.

When talking about car communication there are two approaches:

- Vehicle-to-Vehicle (V2V) communication
- Vehicle-to-Infrastructure (V2I) communication

In V2V communications, also known as Vehicular Ad Hoc Networks (VANET), vehicles establish connections between each other and create a decentralized network providing each other with important information. The advantage of this approach is that no infrastructure is needed. A disadvantage is that when a vehicle is in a low traffic phase of the road or in a low traffic area, there are no other vehicles to obtain information from them. V2I communications assume that some infrastructure is present, like road side units (RSUs), from which the vehicles retrieve information. The advantage of V2I is that also in an area with low traffic, access to the network is still possible, so that, e.g. warnings about road conditions can still be obtained. Its disadvantage is that the infrastructure deployment is costly. Considering these facts, a combination of both approaches gives a suitable solution. This combination is called V2X.

1.2 Legal Background

In order to turn the vision of ITS into reality, there are two basic steps to be done. The first of them is a frequency band allocation for this system, the second one is its standardization.

1.2.1 Frequency Band Allocation

To avoid interferences with or disturbances by other systems, a frequency band has to be reserved, i.e. allocated, only for this system. Although this is regulated by governments, radio propagation doesn't stop at national boundaries. There are some bodies that work on standards for frequency allocation and cover whole regions which span over many countries. The International Telecommunication Union (ITU) acts globally and its member states are organized into six regional groups. There are also organizations that are independent of ITU. One of them is the United States (U.S.) Federal Communications Commission (FCC) which regulates communication in the U.S.. They were the first one who have allocated frequency for vehicular communications. Thus, in North America a 75 MHz wide band around 5.9 GHz is reserved for V2V and V2I communication. In August 2008 the European Commission allocated also a band around 5.9 GHz for V2V and V2I communications. It is not exactly the same as in the U.S. but as long as the applications comply with certain emission limits, the whole 75 MHz may be used for ITS. By this way the spectrum is harmonized with other regions so that equipment will work in all of them. Benefits are a wider market for equipment producers and technical compatibility. Systems built in Europe can be used in the U.S. and vice versa. The European Telecommunications Standards Institute (ETSI) defines seven channels in the 75 MHz frequency band which are 10 MHz wide in its Harmonized Standard EN 302 571 from 2007. In comparison to the IEEE 802.11a standard, from which 802.11p derives; these channels are narrowed from 20 MHz channels to meet the requirements of a vehicular environment. A wider frequency band has shorter transmission times; a shorter frequency band has longer transmission times for preambles and makes the communication more reliable. The ETSI standard also divides the channels according to their purpose. While the fourth is used for channel control, channels 3, 5, 6, and 7 are used for safety related communication and channels 1 and 2 for non-safety related communication. These channels can be merged to obtain higher data rates. Allowed are 20 MHz and 30 MHz channels.

1.2.2 Standardization

Before building up communication systems, their requirements have to be analyzed, possible technologies have to be evaluated and the ones that fulfill the requirements in the best way should find their way to get implemented for these systems. Standards define the way how

to implement systems or parts of it. They also make it possible that different equipment developers can provide solutions for the same system and devices from different suppliers can work together. A communication system bases on rules. The sum of all this rules is called a protocol. Usually a protocol is divided into several modules to make its development easier and to make the protocol more dynamic. Because these modules are layered upon each other, the implementation of a protocol is called a protocol stack. The International Organization for Standardization (ISO) developed a reference protocol stack, the Open Systems Interconnection (OSI) model, and most protocols base on it. The lowest level is the physical layer which interacts with the hardware and the physical channel with its conditions. Above the physical layer there are data link, network, transport, session, presentation and at the top the application layer. IEEE additionally divides the second layer, the data link layer, into two sublayers. These are the medium access control (MAC) layer and the Logical Link Control (LLC) layer. The standard that addresses vehicular communication is called WAVE. It consists of the underlying standards IEEE 802.11p and IEEE 1609 family of standards. An overview is given in Figure 1.1. While the IEEE 802.11p standard defines the lower protocol layers the IEEE 1609 family of standards defines higher protocol layers. IEEE 1609 contains the following sub-standards:

- IEEE 1609.1 - 2006, Standard for Wireless Access in Vehicular Environments (WAVE) — Resource Manager
- IEEE 1609.2 - 2006, Standard for Wireless Access in Vehicular Environments (WAVE) — Security Services for Applications and Management Messages
- IEEE 1609.3 - 2007, Standard for Wireless Access in Vehicular Environments (WAVE) — Networking Services
- IEEE 1609.4 - 2006, Standard for Wireless Access in Vehicular Environments (WAVE) — Multi-Channel Operation

The Resource Manager (RM) is designed to manage the communication of multiple remote applications to multiple onboard units (OBUs). In the context of vehicular applications there exist two types of communication entities. On the one hand the RSUs, which are permanently mounted on the roadside and stationary, and on the other hand OBUs, which are mobile and mounted on vehicles [3]. Remote applications provide services. They communicate through RSUs to OBUs. OBUs host peer applications which consume the services provided over the RSUs. The peer of the RM is the resource command processor (RCP) which resides on OBUs and provides access to resources like memory, user interfaces and other onboard equipment to the resource manager applications (RMAs). Figure 1.2 illustrates the

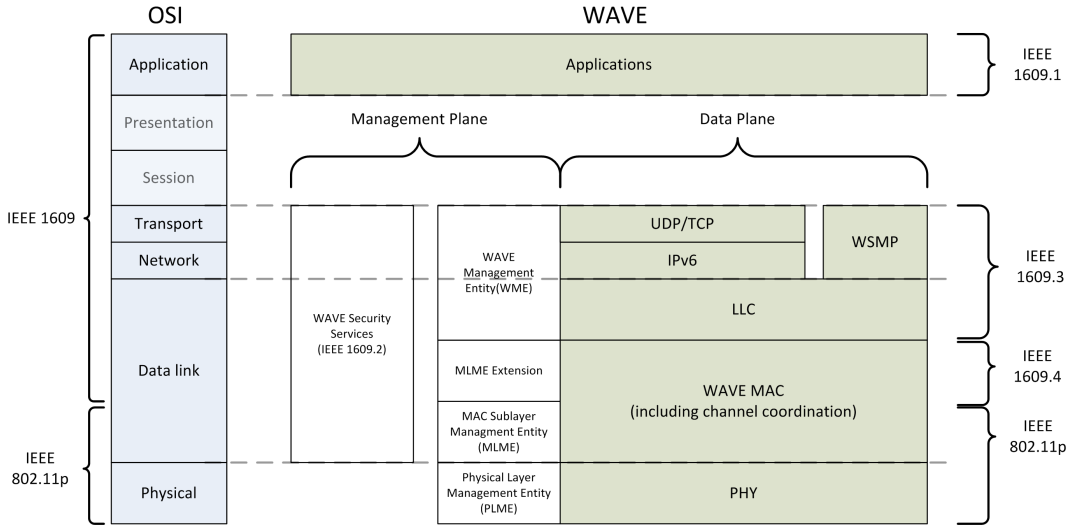


Figure 1.1: *OSI reference and WAVE protocol stack with standards coverage*

relation between OSI reference model and WAVE protocol stack (redrawn from [3]). The IEEE 1609.1 standard also defines commands, responses, status requests and message formats.

The IEEE 1609.2 standard defines the security services in WAVE. It describes the authentication methods, security processing services, security management services, certification management, cryptographic operations, data structures and secure data exchange [4].

The IEEE 1609.3 standard describes the network and the transport layer (OSI layers 3 and 4) protocols and services. WAVE networking services consist of data plane layers and the associated management plane entity (WAVE Management Entity (WME)). The data plane consists of LLC, Internet Protocol version 6 (IPv6), the WAVE Short Messages Protocol (WSMP) and optional Internet Engineering Task Force (IETF) protocols such as User Datagram Protocol (UDP) or Transmission Control Protocol (TCP). The LLC analyzes headers and forwards the incoming packets either to the IPv6 (IP-based data transfer) or to the WSMP (non-IP-based data transfer). The WSMP delivers the incoming message to higher layer entities according to the Provider Service Identifier (PSID). The PSID is a unique number which identifies an application and needs to be registered. The WME (management plane) accepts service requests and provides access to service channels (SCHs) [5].

The IEEE 1609.4 standard provides MAC-layer enhancements to the capabilities standardized in IEEE 802.11p [6]. The data services of the MAC Sublayer Management Entity (MLME) comprise channel coordination, channel routing and user priority. Management plane services comprise multi-channel synchronization (aligning of channel intervals - time synchronization), channel access (managing of requests for communication coming from the WME, Vendor Specific Action (VSA) frames passing to the WME, access to other

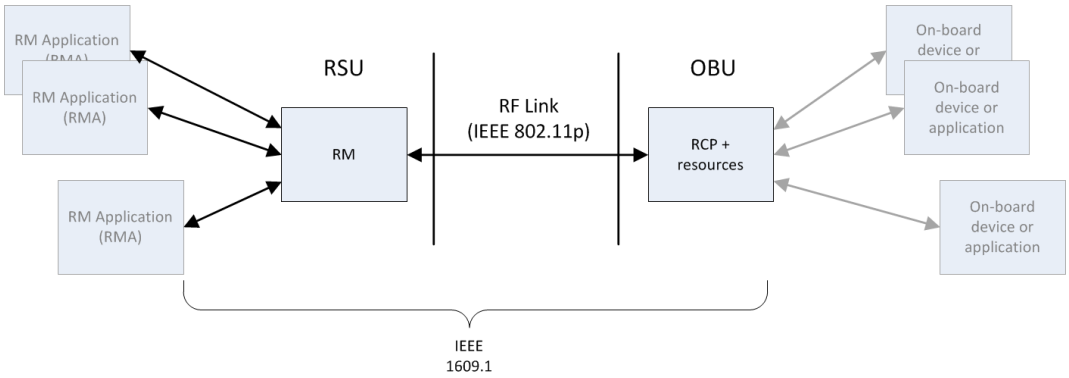


Figure 1.2: *Components addressed by IEEE 1609.1.*

IEEE 802.11p services, maintenance of the Management Information Base (MIB) and readdressing.

The Physical Layer (PHY) and the MAC are defined in IEEE 802.11p. In contrast to the IEEE 802.11a standard, from which IEEE 802.11p derives, some changes have been made to adapt for vehicular environments with very high mobility of users and rapidly changing channel conditions. The authentication process is simplified to make it faster. While in IEEE 802.11a this process can last up to few seconds, in vehicular environments, where fast movement of at least one of the participant is present, this has to be done in much shorter time and is limited to 50 to 100 milliseconds. The MAC layer of IEEE 802.11p establishes connections to networks and is responsible for fragmentation, packet retransmission and acknowledgments. The PHY layer consists of two sublayers. The upper sublayer is called Physical Layer Convergence Protocol (PLCP) and is responsible for preparing packets for different PHYs coming from the MAC layer in a common format. This common format is the PLCP Service Data Unit (PSDU). When sending data, the PSDU is wrapped into the PLCP Protocol Data Unit (PPDU) which contains a preamble and signal information. This is then forwarded to the lowest part of the protocol, the Physical Medium Dependent (PMD) layer which is responsible for modulation, channel coding and transition from digital to analog form of the signal. The focus of this thesis lies on the lowest part (PMD) of the above described protocol stack. The transmission is done using OFDM with 64 subcarriers whereas only the inner 52 are used for payload transmission. The remaining 4 subcarriers are used to transmit pilots required for channel estimation. The IEEE 802.11p standard supports 4 different modulation schemes. While the signal field is always modulated using Binary Phase Shift Keying (BPSK), payload can be modulated using Quadrature Phase Shift Keying (QPSK), 16 Quadrature Amplitude Modulation (QAM) or 64 QAM. A convolutional encoder is used to encode the payload data. Puncturing is also used to adapt the coding rate. Different combinations of modulation and code rate are grouped into regimes. The standard [7] defines

eight regimes. Three of them are compulsory: regime 1 (BPSK with code rate 1/2), regime 3 (QPSK with code rate 1/2) and regime 5 (16 QAM with code rate 1/2).

An extension to the SISO existing scheme with MIMO schemes has been investigated in [1]. MIMO techniques utilize 2 (or more) transmit and 2 (or more) receive antennas. They promise improvements such as increased diversity and enhanced transmission quality. Previous work [1] has shown that MIMO schemes provide a significant reliability improvement, increased throughput and better capability of handling outdated Channel State Information (CSI). In this work the Alamouti and the Golden Code, both in a 2x2 constellation, i.e. with 2 transmit and 2 receive antennas were analyzed. The decoding techniques for Alamouti STBC was Zero-Forcing (ZF) and for the Golden Code the brute force Maximum Likelihood (ML) method. In this thesis, an enhancement to the MIMO scheme will be investigated in form of a soft sphere decoder which provides soft information to the convolutional decoder and by doing so, reduces error rate.

2

IEEE 802.11p Simulator and MIMO Extension

In this chapter an overview of the IEEE 802.11p PHY simulator is given. By describing the simulator, purpose and functionality of the different system components are explained. We first focus on the SISO simulator developed by Forschungszentrum Telekommunikation Wien (FTW) within the REALSAFE project. This description is mainly oriented on [2] and gives a system overview of introductory character. The second part presents MIMO extensions in combination with STBCs.

2.1 Simulator Overview

The focus of the simulation lies on the physical layer of the WAVE protocol, i.e. the part which is covered by the IEEE 802.11p standard. Thus, the simulation starts at the point where the MAC layer delivers the PSDU to the upper PHY layer, the PLCP. These data are generated randomly and passed through the blocks of the simulator. At a first glance the simulator can be divided into three main blocks: the transmitter, the channel and the receiver, as depicted in Figure 2.1.

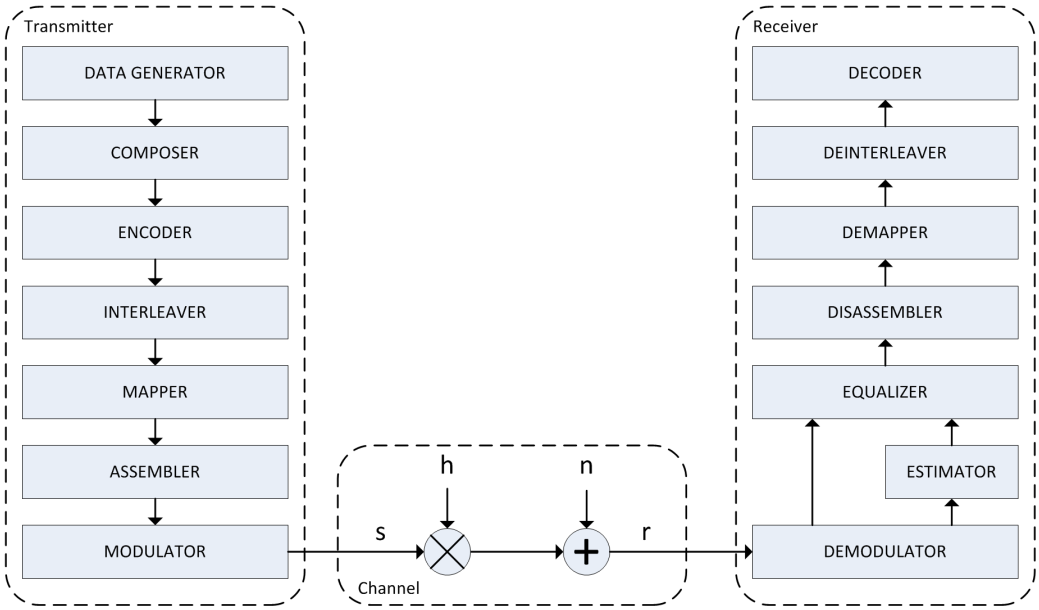


Figure 2.1: *SISO simulator overview*

2.1.1 Transmitter

Before the simulation of the transmitter starts, random data are generated by a random numbers generator. Its outcome is a set of uniformly distributed pseudorandom values between 0 and 1, which are then rounded to 0 or to 1. These data represent the PSDU, i.e. the payload of the PPDU. The transmitter, as a first step, wraps the randomly generated data PSDU into the PPDU by adding the preamble, signal and postamble to the data. In the simulator this is done in the part called ”composer”. Figure 2.2 shows the PPDU generation. Depending on the selected parameters for the simulation, the signal values are set. The rate field gives information about the data rate which is in fact representing one of the regimes defined in the standard. Each regime is characterized by a combination of modulation and code rate. Size of the payload is given in octets and is placed in the length field of the signal. For transmission of real application data a scrambler is used to randomize the data pattern, because application data can contain long sequences of 1s or 0s. Such sequences influence the signal's power spectrum. The scrambler is not implemented in the simulator as we already have a randomized data pattern resulting from the uniform distribution of the random numbers generator.

Forward Error Correction (FEC) Coding is performed by a convolutional encoder. The standard defines code rates of 1/2, 2/3 and 3/4. For rate 1/2 each data bit is encoded into 2 code bits. By adding some redundancy into the data stream, it is possible to combat errors introduced by the channel. The mother code is the 1/2 rate code, while the 2/3 and the 3/4

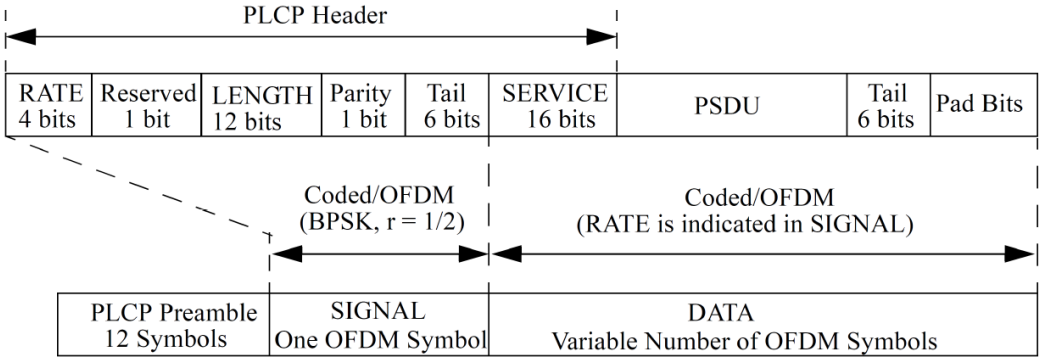


Figure 2.2: *Construction of the PPDU frame [7]*

rate codes are obtained by puncturing. Puncturing is a technique to increase the rate of a code without changing the coder or the decoder. This is done by repeatedly removing some bits of the coded sequence at the transmitter according to a predefined pattern. At receiver side depuncturing can be done as the code rate of the transmission and the puncturing patterns are known.

Encoding is followed by interleaving. The interleaver rearranges the data bits in such a way that adjacent bits are spread over non adjacent subcarriers. That means it takes the modulation scheme into account, because the block of bits that is interleaved has to be of the same size as the number of coded bits in one OFDM symbol. This size is determined by the used modulation scheme. The influence of burst errors or fading dips can be reduced by this technique. A burst error occurs when a sequence of bits is erroneous. If the length of such a sequence exceeds the error correcting code's capability, the original information cannot be recovered. By rearranging the bits, erroneous bits are distributed and are not adjacent anymore, allowing the FEC to correct the errors. A fading dip occurs when on one frequency the attenuation is high and data sent at this frequency is susceptible for errors by noise. Distributing adjacent bits among non-adjacent subcarriers avoids that all adjacent bits are sent over the strongly attenuated frequency.

The next part of the transmitter is the mapper. In order to achieve higher throughput, adjacent bits are mapped to higher order non-binary symbols. If symbol $a[k]$ carries l bits, it can take $M_a = 2^l$ different values. The set of all possible values of the symbols for a modulation format is called symbol-alphabet and is denoted as

$$\mathcal{A} = \{a^{(1)}, a^{(2)}, ...a^{(M_a)}\} \quad \text{with} \quad M_a = 2^l. \tag{2.1}$$

The symbol-alphabet can be represented as a signal constellation which shows the symbols in a complex plane. Each of these symbols is being sent with the symbol rate that is

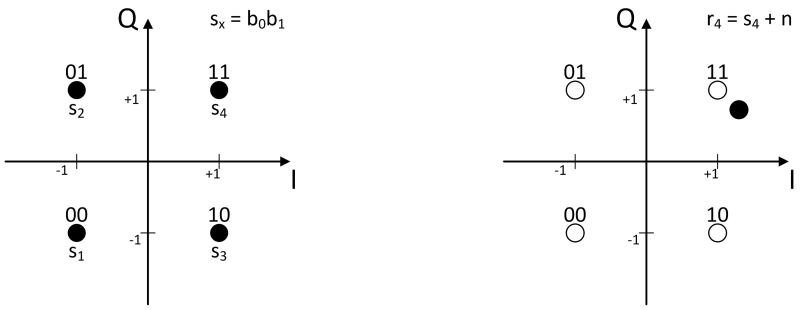


Figure 2.3: *QPSK signal constellation and error introduced by additive noise*

equal to the pulse rate and limited by the channel bandwidth. By mapping l bits to one symbol, the transmission rate is increased by l. With increasing modulation order, the number of bits mapped to one symbol is increased as well and a higher transmission rate is obtained. In practice it is not possible to increase the throughput arbitrary as there are power and bandwidth constraints. A higher order modulation at the same transmit power means that the distance between the symbols is decreased, resulting in larger number of detection errors. The channel adds noise to the sent symbol so that the received symbol is shifted in the constellation plane, see Figure 2.3 on the right. When this shift by noise is large enough the sent symbol may be detected as one of its neighbors and a symbol error occurs. To reduce bit error probability in case of such a wrong detection, Gray code is used so that neighbor symbols differentiate always in exactly one bit. The IEEE 802.11p standard supports four signal-constellations: BPSK, QPSK, 16-QAM and 64-QAM. While signaling is always modulated using BPSK, see Figure 2.2, payload data is modulated using either BPSK or one of the other constellations. The symbols are normalized in order to achieve equal average symbol power. Assuming equally likely symbols, the mean power for a signal-constellation is given by

$$P_A = \frac{1}{M_a} \sum_{m=1}^{M_a} |a^{(m)}|^2 \tag{2.2}$$

so the amplitude needs to be normalized by a factor $\frac{1}{P_A}$.

After the bits are mapped onto symbols, these are now inserted into an OFDM structure. This structure represents an OFDM frame in form of a matrix as shown in Figure 2.4. Rows of the matrix represent the subcarriers while each column represents an OFDM symbol. As already mentioned, out of the 64 subcarriers, 52 are used for pilot and payload transmission: 4 of them are pilot sequences and 48 are payload. Subsequent symbols are arranged on the 48 subcarriers from ”top” to ”bottom”. These 48 symbols create one OFDM symbol. The next 48 symbols are put in the next column representing the second OFDM symbol. This procedure is done until all symbols are inserted. In case that the generated data results in a

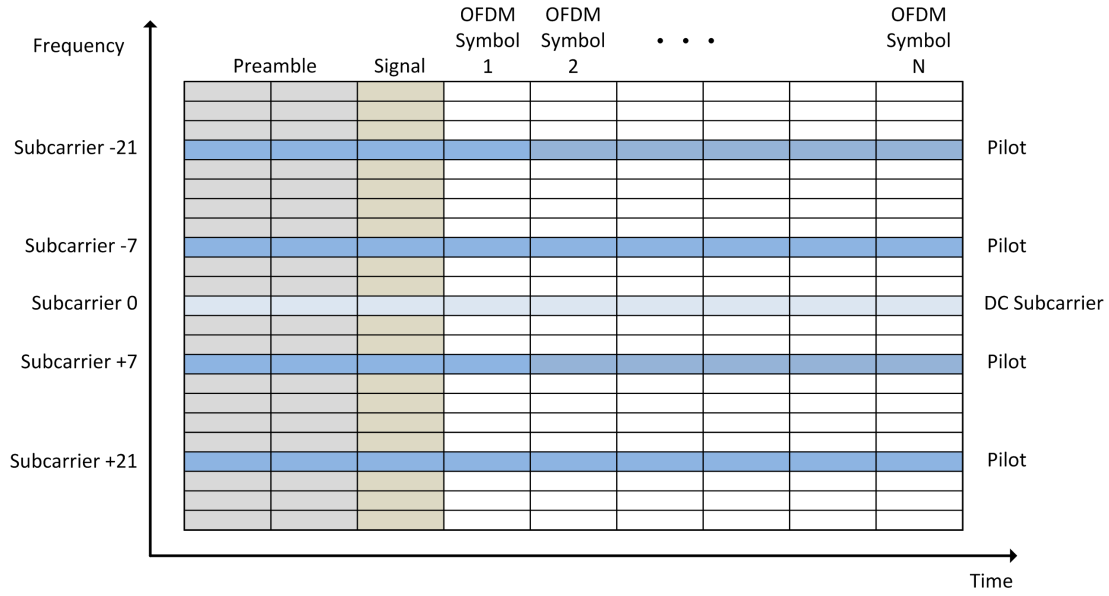


Figure 2.4: *Schematical representation of OFDM symbol assembly*

symbols count that is not a multiple of 48, zero filling is done at the beginning at bit level to get a modulo 48 count of alphabet symbols. In the simulator part called ”assembler”, this arrangement of symbols into a matrix is performed. The OFDM structure itself contains 53 rows. Row 1 represents subcarrier -26, row 27 represents the DC subcarrier and row 53 represents subcarrier 26. When filling the symbols, subcarrier -21, -7, 0, 7 and 21 are skipped. These subcarriers are filled with the pilot sequences except the DC subcarrier; this one is filled with all zeros. The first two columns are preamble, while the third one is the signal as described in Figure 2.2. Because the signal is 24 bits long and always modulated using BPSK and coded with a coding rate of 1/2, its size results always in 48 coded bits, i.e. BPSK symbols, that match exactly to one OFDM symbol.

Finally the OFDM frame is being modulated in the simulator part called ”modulator”. At this part, the prepared OFDM structure becomes an OFDM frame by converting digital data into analogue signals. In OFDM, subcarriers are orthogonal to each other and can be spaced closer together as this is the case with Frequency Division Multiple Access (FDMA). The spectrum of each subcarrier has a null at the center frequencies of all the other subcarriers. An OFDM symbol is generated by having K orthogonal pulses $g_k(t)$, one for each subcarrier, multiplied by the according amplitude factors, represented by the symbols $a[k]$. Now, let us denote the symbols according to their position inside the OFDM matrix, with k representing the subcarrier index and n representing the OFDM symbol index. The amplitude factor for subcarrier k of the n^{th} OFDM symbol is $a_k[n]$ and the signal component for one OFDM

frame is

$$s_n(t) = \sum_{k=0}^{K-1} a_k[n] g_k(t - nT_s) e^{j2\pi kFt}. \tag{2.3}$$

The overall transmit signal is the sum of all signal components

$$s(t) = \sum_{k=-\infty}^{\infty} s_n(t). \tag{2.4}$$

Sampling the signal at time instances mT_s for one OFDM symbol and thereby performing the transition into the discrete-time processing results in

$$s[m] = \frac{1}{\sqrt{K}} \sum_{k=-\infty}^{\infty} \sum_{k=0}^{K-1} a_k[n] e^{j2\pi kFt}. \tag{2.5}$$

This is an inverse Discrete Fourier Transform (DFT) which is usually realized by an Inverse Fast Fourier Transformation (IFFT). In contrast to an analog implementation that would require multiple local oscillators (LOs), which is not a practical solution, above described digital implementation allows much simpler and cheaper transceiver design. In order to eliminate delay dispersion and the resulting Inter Carrier Interference (ICI) a special guard interval is used. This guard interval, also known as cyclic prefix (CP), is realized by prepending the 64 subcarrier symbol with the last 16 subcarriers of the symbol in the guard interval in front of that symbol. By doing this, the linear convolution of the transmitted signal with the channel impulse response turns into a cyclic convolution and as a result, the frequency-selective channel is converted into parallel flat-fading channels [8].

2.1.2 Channel

The channel is the physical media between the transmitter output with the receiver input. For wireless channels, multipath propagation is typical since the signal travels different paths from the transmitter to the receiver. On these paths, the signal interferes with objects, which are called Interacting Objects (IOs), resulting in reflections, diffractions and scattering of the signal. A description of all this processes that define the multipath components is very complicated. Simplified models are often used instead as they reflect the most important properties of the channel. Following channel models can be chosen in the simulator and are described below:

- Additive White Gaussian Noise (AWGN) Channel Model

- Block-Fading Channel Model
- Jakes Channel Model
- Geometry-Based Stochastic Channel Model (GSCM)

The AWGN channel is the simplest model of the channel. It attenuates the signal, causes phase rotation and adds Gaussian-distributed noise [8]. The Gaussian distribution of noise suits best for real world measurements as, according to the central limit theorem, a superposition of n random variables is Gaussian distributed if $n \to \infty$, no matter how the distribution of the individual variables looks like. As attenuation and phase rotation are temporally constant, they can easily be taken into account [8], so the simulator only adds complex noise to the signal. The AWGN channel produces a receive signal $r[n]$ which is modeled as

$$r[n] = \alpha s[n] + z[n]. \tag{2.6}$$

$s[n]$ is the transmitted signal, α is the complex channel gain $|\alpha|e^{(j\varphi)}$, and $z[n]$ is the noise component at time instance n. It can be shown that $z[n]$ is a stationary, zero-mean, circularly symmetric random process [9]. Its real and imaginary parts are statistically independent and normally distributed with identical variances equal to $\sigma_Z^2/2$ [9]. The probability density function (pdf) of $z[n]$ is therefore

$$f_z(z) = f_{Z_R,Z_I}(z_R, z_I) = f_{Z_R}(z_R) f_{Z_I}(z_I) = \frac{1}{\pi\sigma_Z^2} \exp\left(- \frac{|z|^2}{\sigma_Z^2} \right). \tag{2.7}$$

Above separation of the joint pdf into a product of the pdfs of the real and imaginary part can be done due to their statistical independence. As the noise is zero-mean its variance σ_Z^2 equals the noise mean power.

With the above described channel model, multipath propagation is not taken into account. As the signal arrives to the receiver along different paths, each of these arriving copies has different amplitude, phase and delay. The receiver obtains a superposition of these copies which create constructive and destructive interferences and as a result of this, location-dependent fading dips occur. Such channels are said to be time-dispersive or also frequency-selective. The influences of multipath propagation can be modeled by the channel transfer function. If all components are assumed static, i.e. the transmitter, the receiver and the IOs, the channel is time invariant and can be described by its impulse response $h(t)$ and treated as a linear filter. As the power of the first arriving copy of the signal is usually the strongest and that of the later arriving copies is less strong due to a longer path, a monotonically decaying Power-Delay Profile (PDP) is used to model this effect. The PDP is sampled at time points mT_s and

is described as a tapped delay line, where m denotes the propagation path. The gain of the m^{th} path is

$$p_h[m] = \gamma\sqrt{\exp(-\frac{mT_s}{\sigma})} \quad \text{with} \quad \gamma = \frac{1}{\sqrt{\sum_{m=0}^{L-1}|p[m]|^2}}. \tag{2.8}$$

The above calculated gains are multiplied by complex Gaussian noise for each instance of a channel realization. Now, the impulse response of the channel is created so that the PDP is equal for the whole OFDM frame and therefore the name block-fading channel model. The output of the channel is a convolution of the Rayleigh distributed channel coefficients $h[m]$ with the transmitted sequence and an addition of Gaussian noise as in the AWGN channel

$$r[n] = (h[m] * s[n]) + z[n]. \tag{2.9}$$

Block-fading channel assumes constant channel coefficients for the same subcarrier during the whole OFDM frame. The channel realization for the following OFDM frame is calculated again and differs from the previous one.

In practice, block-fading channel assumptions are rarely fulfilled. Time-variant channel models are used to simulate channels with time selectivity in addition with frequency selectivity. Above mentioned assumption of static components is now discarded. Movement of transmitter, receiver or any of the IOs causes frequency shifts called Doppler Shifts. If the transmitter moves away from the receiver, the received frequency f_c is decreased, i.e. the Doppler Shift has to be added

$$\nu = -f_c\frac{v}{c_0}, \tag{2.10}$$

with c_0 as speed of light. Note that the above speed is the speed in direction of the wave propagation. If the moving direction of the transmitter is not the same as the wave propagation direction, then the component in wave propagation direction has to be taken into account, which is $v \cdot cos(\phi)$, i.e. the magnitude of the projection of $\vec{v}$ on the direction of wave propagation $\vec{e}_{wave}$. The classical model of time-variant channels is the Jakes simulation model for Rayleigh fading channels with Non-Line-of-Sight (NLOS). Here, the frequency selective part is implemented with monotonically decaying PDP and the time selective with relative speed between transmitter and receiver by applying a Doppler Shift to each delay-tab of the PDP. In this simulation model, the impulse response of one channel realization varies for each OFDM symbol with time. The channel coefficients are formed by simple multiplication

$$h[m,n] = h[m] \cdot h[n]. \tag{2.11}$$

Multipath propagation is introduced by Rayleigh distributed channel coefficients $h[m]$ and time variation of each multipath component due to movement is introduced by multiplication with independent Jakes spectrum $h[n]$. Channel output is given by a convolution of the transmitted signal with the frequency-selective, time-variant impulse response and addition of Gaussian noise as in the AWGN channel

$$r[n] = (h[m, n] * s[n]) + z[n]. \tag{2.12}$$

As the simulation should represent a communication system in vehicular environments, the Jakes channel model is not optimal. A stochastic ray-based model is implemented which simulates V2I communication on a four lane highway, two in each direction. That means that there is a moving transmitter, a static receiver and static as well as moving scatterers. Input parameters to the model are positions of transmit and receive antennas and the transmitter velocity. The time-variant impulse response is modeled as follows

$$h(t, \tau) = h_{LOS}(t, \tau) + \sum_{p=1}^{P} h_{MD}(t, \tau_p) + \sum_{q=1}^{Q} h_{SD}(t, \tau_q) + \sum_{r=1}^{R} h_{DI}(t, \tau_r), \tag{2.13}$$

where the first term represents a Line-of-Sight (LOS) component with ground reflections, the second one representing mobile discrete (MD), the third one static discrete (SD) and the fourth one diffuse scatterers (DS). Mobile discrete scatterers simulate cars driving in same or opposite direction, static discrete scatterers simulate road signs and diffuse scatterers simulate buildings. Mobile scatterers move with a randomly assigned speed with mean of 90 km/h and standard deviation of 2 km/h. A more detailed description of this channel model can be found in [10].

2.1.3 Receiver

The digital data has been prepared and transformed into analog form by the transmitter, sent through the channel which corrupts the signal and is now received at the receiver. The received signal needs to be processed in order to determine the sent information. A block diagram of the receiver is given in Figure 2.1 on the right.

The first step of the processing is to parallelize the signal, i.e. transform it from the time domain into the frequency domain. The serial to parallel (S/P) conversion is done in the demodulator by using the inverse transform of the IFFT: the Fast Fourier Transformation (FFT), see Equation 2.14. Before that can be done, the cyclic prefix has to be removed by simply ignoring the first 16 samples of each OFDM symbol. From the demodulated data, the

preamble is extracted and used for channel estimation.

$$S[k] = \sum_{n=0}^{K-1} s[n] e^{-j2\pi \frac{nk}{K}}. \tag{2.14}$$

At the estimator block, the preamble is used to estimate the coefficients of the channel impulse response. As one alphabet symbol, for example on the k^{th} subcarrier at time point n is given by

$$r_k[n] = H_k[n] s_k[n] + z_k[n], \tag{2.15}$$

the preamble, which is known at the receiver, can be used to estimate the coefficients $H_k[n]$. This estimation can be done by following calculation

$$\hat{H}_k[n] = \frac{r_k[n]}{s_k[n]} = H_k[n] + \frac{z_k[n]}{s_k[n]}, \tag{2.16}$$

with $H_k[n]$ the true channel coefficient and $z_k[n]/s_k[n]$ the residual noise. This is the Least Squares (LS) estimation and has a disadvantage of high mean-square error. The method that is implemented in the simulator uses the long preamble and calculates the coefficients for two OFDM symbols by above method. For each subcarrier there are now two estimates. The final estimate for that specific subcarrier is simply the average value of both of them. That method is called *averaged LS estimation*.

Now, that the channel coefficients are estimated, the equalization block utilizes the information about the channel to remove the distortion of the signal introduced by the channel. The simplest way to do this is the ZF method. Under the assumption of perfect knowledge of the CSI, the received sample can be divided by the channel impulse response coefficient for that sample. Using Equation 2.15 gives

$$\hat{r}_k[n] = \frac{r_k[n]}{H_k[n]} = \frac{1}{H_k[n]}(H_k[n] s_k[n] + z_k[n]) = s_k[n] + \frac{z_k[n]}{H_k[n]} \tag{2.17}$$

for the equalized alphabet symbol on k^{th} subcarrier at sampling point n. As the CSI is not known perfectly, but is estimated, the equalization is done using the estimated channel coefficients $\hat{H}_k[n]$. Because the above described channel estimation provides one channel coefficient for all the time samples of one OFDM symbol for one subcarrier, $\hat{H}_k[n]$ is equal for all n during one symbol, but differs for the subcarriers k: $\hat{H}_k[n] = \hat{H}_k$. Equation 2.17

becomes

$$\hat{r}_k[n] = \frac{r_k[n]}{\hat{H}_k} = \frac{H_k[n]s_k[n]}{\hat{H}_k} + \frac{z_k[n]}{\hat{H}_k} = \tilde{s}_k[n] + \tilde{z}_k[n]. \tag{2.18}$$

Zero Forcing equalization has the disadvantage that it amplifies the noise. For Block-Fading channels, the estimated channel coefficients remain equal and the equalizer has good performance, but for time-variant channels the coefficients change during one OFDM frame. That means the estimated $\hat{H}_k$ do not represent the correct coefficients at time points n and as a result $\tilde{s}_k[n]$ has a larger error the more $\hat{H}_k$ differs from the real coefficient $H_k[n]$. More advanced estimators can be used to get the CSI for time variant channels, but will not be presented here, as the resulting MIMO simulator implements only the *averaged LS estimator* described above.

The outputs of the equalizer are complex valued numbers and represent symbols in the constellation plane. These are now reordered by the "disassembler" block from the OFDM matrix structure into a sequence of symbols again and represent the input to the decoder and demapper block. Due to noise shifts and distortion introduced by the channel, the received symbols usually do not lie on the predefined positions of the symbols of the given alphabet $\mathcal{A}$. The demapper block of the simulator has the purpose to detect the symbols $\hat{a} \in \mathcal{A}$ from the received values $\tilde{r}_k[n]$ either by hard or soft detection and to demap these detected symbols to the coded bits c_i according to the used modulation format. Hard demapping is based on the minimum Euclidean distance of the received symbol and all the possible symbols of the alphabet [9].

$$\hat{a}[n] = \arg\min_{a \in \mathcal{A}} \sqrt{|r[n] - a|}. \tag{2.19}$$

Soft demapping is based on the A-Posteriori Probability (APP) $P(c_i = 0|r)$, i.e. the probability that the detected code bit c_i has the value 0 given the received symbol r. This probability is a measure of the reliability of the information about the code bit c_i. Soft decoders work with Log-Likelihood Ratios (LLRs) that are defined as

$$LLR(c_i) = \log \frac{P(c_i = 1|r)}{P(c_i = 0|r)}. \tag{2.20}$$

When using LLRs, the sign of the LLR contains the information about the value of code bit $c_i \in 0, 1$ and the magnitude is a measure for the reliability of c_i being the correct value. Soft decoders utilize that information for decoding.

After the demapper, the code bits are rearranged by inverting the interleaving which is done at the receiver. This is done by the deinterleaver block.

Decoding is done at the last block of the receiver. Before decoding can be done, the puncturing, if some has been done, has to be reversed. This is done by inserting zeros at the positions where code bits were removed at the transmitter and it has no influence on the decoder. Decoding can be done using either the Viterbi algorithm or the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm. In the presented simulations in Chapter 4, the BCJR algorithm was used to utilize the soft information from the Soft-Sphere decoder.

2.2 Space-Time Block Codes

STBCs describe how information is distributed over antennas (space) and symbol periods (time), thus STBCs are always used in context of a system that utilizes multiple transmit and receive antennas. Such systems are called MIMO systems and will be shortly described before proceeding to the STBCs Alamouti and Golden Code.

2.2.1 MIMO Basics

MIMO systems use multiple antennas at the transmitter and at the receiver side. Simultaneous transmission of multiple data streams increases the system capacity. MIMO systems can also improve reliability and coverage range. These streams travel different paths and a superposition of them is received at the receiver side. To make use of this spatial multiplexing, the number of receive antennas has to be at least that of transmit antennas. If M_T is the number of transmit antennas and M_R the number of receive antennas, a maximum of $M_T M_R$ links is available and diversity order is $M_T M_R$. The data rate can thereby be increased by the factor $min(M_T, M_R)$. Some disadvantages of MIMO are that in order to separate the combinations of multiple data streams, appropriate signal processing on the receiver side is required. Especially for mobile devices, this results in a shorter battery lifetime. Another issue is that a radio frequency chain is needed for every additional antenna, which makes the system more expensive. The links of a MIMO system are characterized by the channel impulse response matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} h_{1,1} & \dots & h_{1,M_T} \\ \vdots & \ddots & \vdots \\ h_{M_R,1} & \dots & h_{M_R,M_T} \end{bmatrix}. \tag{2.21}$$

The elements $h_{i,j}$ of $\mathbf{H}$ represent the link from transmit antenna j to receive antenna i at a given time point. In a MIMO system, each antenna transmits one symbol, usually from a modulation alphabet $\mathcal{A}$ as described in Equation 2.1 and the system is described by

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{M_R} \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} & \dots & h_{1,M_T} \\ h_{2,1} & h_{2,2} & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ h_{M_R,1} & h_{M_R,2} & \dots & h_{M_R,M_T} \end{bmatrix} \begin{bmatrix} c_{1,1} & \cdots & c_{1,N} \\ c_{2,1} & \cdots & c_{2,N} \\ \vdots & \ddots & \vdots \\ c_{M_T,1} & \cdots & c_{M_T,N} \end{bmatrix} + \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_{M_R} \end{bmatrix}, \quad (2.22)$$

with y_i being the signal that is received on the i^{th} receive antenna, $c_{j,n}$ is the symbol that is transmitted over the j^{th} transmit antenna and during the n^{th} time slot and z_i is the Gaussian noise introduced by the channel at the i^{th} receive antenna. In a more compact form, Equation 2.22 can be written as

$$\mathbf{y} = \mathbf{HS} + \mathbf{z}. \quad (2.23)$$

A combination of space-time coding with OFDM will be described later, now the STBCs in their simplest form will be described, i.e. for one frequency (no subcarriers).

This work analyzes the Alamouti and the Golden Code STBCs, both in a 2x2 constellation, i.e. with 2 transmit and 2 receive antennas. Both codes are mapping alphabet symbols to codeword matrices, whose rows and columns represent antennas and time slots, respectively. In a 2x2 constellation, the codeword matrices have following form:

$$\mathbf{X} = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}, \quad (2.24)$$

where the $c_{i,j}$ represent the transformed alphabet symbols which are sent from antenna i in time slot j. The transformation of the alphabet symbols is specific to the underlying code and is explained for Alamouti and Golden Code in the next two sections.

2.2.2 Alamouti

The first STBC is the Alamouti code, named after S.M. Alamouti, described in [11]. STBCs are designed to exploit the full diversity of MIMO systems. Alamouti's code is designed for a two branch transmit diversity scheme and transmits two symbols of an alphabet using two time slots under the assumption that the channel doesn't change during these two time slots. It improves quality of reception with low complexity at the receiver. Utilizing two antennas is suitable for vehicular communication as mounting of more than two antennas on a vehicle is a financial and implementation issue. The code is constructed by sending the two consecutive symbols s_1 and s_2 over the two transmit antennas at the first time slot. At the

second time slot the conjugates of these symbols are sent each over the other antenna, i.e. antenna 1 sends $-s_2^*$ and antenna 2 sends s_1^*. In case of the Alamouti Code, the codeword matrix from Equation 2.24 is created as:

$$\mathbf{X} = \begin{bmatrix} s_1 & -s_2^* \\ s_2 & s_1^* \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}, \tag{2.25}$$

where $c_{i,j}$ is sent from antenna i in time slot j. The codeword matrix $\mathbf{X}$ for this code is always orthogonal: $\mathbf{X}\mathbf{X}^H = \mathbf{I}(|s_1|^2 + |s_2|^2)$. The Alamouti code obtains always the maximum diversity order as its codeword difference matrix has full rank [12].

2.2.3 Golden Code

The Golden Code is a full rate, full diversity STBC with a nonzero lower bound on its coding gain, which is $1/5$ and comes from the key role of the Golden number $\frac{1+\sqrt{5}}{2}$ from which the code's name originates [14]. Its main structure bases on a cyclic division algebra. This code was designed between 2003 and 2005 by authors of [13] and [14] independently of each other. In the case of two transmit and two receive antennas, the code is constructed according to Equation 2.26. Compared to the Alamouti 2x2 code, instead of two symbols, now four symbols are used to create a 2x2 codeword matrix: let s_1, s_2, s_3 and s_4 be four consecutive symbols, the codeword matrix constructed from these symbols is

$$\mathbf{X} = \frac{1}{\sqrt{5}} \begin{bmatrix} \alpha[s_1 + s_2\theta] & \alpha[s_3 + s_4\theta] \\ j\sigma(\alpha)[s_3 + s_4\sigma(\theta)] & \sigma(\alpha)[s_1 + s_2\sigma(\theta)] \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}, \tag{2.26}$$

$$\begin{aligned} &\text{with} \qquad j = \sqrt{-1} \qquad \theta = \frac{1+\sqrt{5}}{2} \qquad \sigma(\theta) = \frac{1-\sqrt{5}}{2} = 1 - \theta \\ &\alpha = 1 + j - j\theta = 1 + j\sigma(\theta) \qquad \sigma(\alpha) = 1 + j - j\sigma(\theta) = 1 + j\theta. \end{aligned}$$

As in the case of the Alamouti code, the channel is assumed not to change during the transmission of one codeword matrix, i.e. for the two time slots.

Encoding in the simulator is done in such a way that a block ob subsequent symbols can be put into a vector and multiplied by a matrix $\mathbf{G}$ to get the codeword matrix in form of a vector. In order to do that, the entries of the codeword matrix (2.26) have to be reformulated. Assuming $c_{i,j} = \tilde{c}_{i,j}/\sqrt{5}$, one can see that the entries of the codeword matrix have the same

forms:

$$\alpha[a + \theta b] \quad \text{for} \quad \tilde{c}_{1,1}, \tilde{c}_{1,2} \tag{2.27}$$

$$\sigma(\alpha)[a + \sigma(\theta)b] \quad \text{for} \quad \tilde{c}_{2,1}, \tilde{c}_{2,2}. \tag{2.28}$$

By using the expressions for α, $\sigma(\theta)$ and $\sigma(\alpha)$ from Equation 2.26 and additionally the fact that $\theta\sigma(\theta) = -1$ and $\theta + \sigma(\theta) = 1$, expression 2.27 can be reformulated. In the following a, b are placeholders for one of the four symbols s_1, s_2, s_3 or s_4.

$$\begin{aligned} \alpha[a + \theta b] &= [1 + j\sigma(\theta)]a + [1 + j\sigma(\theta)]\theta b \\ &= [1 + j\sigma(\theta)]a + [\theta + j\sigma(\theta)\theta]b \\ &= [1 + j\sigma(\theta)]a + [\theta + j(-1)]b \\ &= [1 + j\sigma(\theta)]a + [\theta - j]b. \end{aligned} \tag{2.29}$$

Expression 2.28 can be reformulated as:

$$\begin{aligned} \sigma(\alpha)[a + \sigma(\theta)b] &= [1 + j\theta]a + [1 + j\theta]\sigma(\theta)b \\ &= [1 + j\theta]a + [\sigma(\theta) + j\theta\sigma(\theta)]b \\ &= [1 + j\theta]a + [\sigma(\theta) - j]b. \end{aligned} \tag{2.30}$$

For $\tilde{c}_{1,1}$ we have $a = s_1$ and $b = s_2$, for $\tilde{c}_{1,2}$ $a = s_3$ and $b = s_4$, resulting in:

$$\begin{aligned} \tilde{c}_{1,1} &= \alpha[s_1 + \theta s_2] = [1 + j\sigma(\theta)]s_1 + [\theta - j]s_2 \\ \tilde{c}_{1,2} &= \alpha[s_3 + \theta s_4] = [1 + j\sigma(\theta)]s_3 + [\theta - j]s_4. \end{aligned} \tag{2.31}$$

For $\tilde{c}_{2,1}$ we have $a = s_3$ and $b = s_4$, for $\tilde{c}_{2,2}$ $a = s_1$ and $b = s_2$. Note that for $\tilde{c}_{2,1}$ a multiplication with i has to be performed.

$$\begin{aligned} \tilde{c}_{2,1} &= j\sigma(\alpha)[s_3 + \sigma(\theta)s_4] = [j - \theta]s_3 + [1 + j\sigma(\theta)]s_4 \\ \tilde{c}_{2,2} &= \sigma(\alpha)[s_1 + \sigma(\theta)s_2] = [1 + j\theta]s_1 + [\sigma(\theta) - j]s_2. \end{aligned} \tag{2.32}$$

The codeword matrix has now the following form

$$\mathbf{X} = \frac{1}{\sqrt{5}} \begin{bmatrix} [1 + j\sigma(\theta)]s_1 + [\theta - j]s_2 & [1 + j\sigma(\theta)]s_3 + [\theta - j]s_4 \\ [j - \theta]s_3 + [1 + j\sigma(\theta)]s_4 & [1 + j\theta]s_1 + [\sigma(\theta) - j]s_2 \end{bmatrix} \tag{2.33}$$

and can be rewritten in a vector form:

$$\mathbf{x} = \frac{1}{\sqrt{5}} \begin{bmatrix} [1 + j\sigma(\theta)]s_1 + [\theta - j]s_2 \\ [j - \theta]s_3 + [1 + j\sigma(\theta)]s_4 \\ [1 + j\sigma(\theta)]s_3 + [\theta - j]s_4 \\ [1 + j\theta]s_1 + [\sigma(\theta) - j]s_2 \end{bmatrix} = \begin{bmatrix} c_{1,1} \\ c_{2,1} \\ c_{1,2} \\ c_{2,2} \end{bmatrix}. \tag{2.34}$$

Now, the real and imaginary parts are separated in the following way:

$$\mathbf{x} = \begin{bmatrix} \Re(c_{1,1}) \\ \Im(c_{1,1}) \\ \Re(c_{2,1}) \\ \Im(c_{2,1}) \\ \Re(c_{1,2}) \\ \Im(c_{1,2}) \\ \Re(c_{2,2}) \\ \Im(c_{2,2}) \end{bmatrix} = \frac{1}{\sqrt{5}} \begin{bmatrix} \Re(s_1) - \sigma(\theta)\Im(s_1) + \theta\Re(s_2) + \Im(s_2) \\ \sigma(\theta)\Re(s_1) + \Im(s_1) - \Re(s_2) + \theta\Im(s_2) \\ -\theta\Re(s_3) - \Im(s_3) + \Re(s_4) - \sigma(\theta)\Im(s_4) \\ \Re(s_3) - \theta\Im(s_3) + \sigma(\theta)\Re(s_4) + \Im(s_4) \\ \Re(s_3) - \sigma(\theta)\Im(s_3) + \theta\Re(s_4) + \Im(s_4) \\ \sigma(\theta)\Re(s_3) + \Im(s_3) - \Re(s_4) + \theta\Im(s_4) \\ \Re(s_1) - \theta\Im(s_1) + \sigma(\theta)\Re(s_2) + \Im(s_2) \\ \theta\Re(s_1) + \Im(s_1) - \Re(s_2) + \sigma(\theta)\Im(s_2) \end{bmatrix}, \tag{2.35}$$

and can be factorized

$$\mathbf{x} = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & -\sigma(\theta) & \theta & 1 & 0 & 0 & 0 & 0 \\ \sigma(\theta) & 1 & -1 & \theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\theta & -1 & 1 & -\sigma(\theta) \\ 0 & 0 & 0 & 0 & 1 & -\theta & \sigma(\theta) & 1 \\ 0 & 0 & 0 & 0 & 1 & \sigma(\theta) & \theta & 1 \\ 0 & 0 & 0 & 0 & \sigma(\theta) & 1 & -1 & \theta \\ 1 & -\theta & \sigma(\theta) & 1 & 0 & 0 & 0 & 0 \\ \theta & 1 & -1 & \sigma(\theta) & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Re(s_1) \\ \Im(s_1) \\ \Re(s_2) \\ \Im(s_2) \\ \Re(s_3) \\ \Im(s_3) \\ \Re(s_4) \\ \Im(s_4) \end{bmatrix}, \tag{2.36}$$

and finally written in a more compact form:

$$\mathbf{x} = \mathbf{G}\mathbf{s}. \tag{2.37}$$

Equation 2.37 makes it possible to map a block of four consecutive symbols to a code block $\mathbf{x}$ by simple matrix multiplication. The symbols just need to be split into their real and imaginary parts, stacked into a vector $\mathbf{s}$ and left-side multiplied by the generator matrix $\mathbf{G}$. To get the codeword matrix $\mathbf{X}$, the code symbols $c_{i,j}$ are obtained from $\mathbf{x}$ by adding the real and imaginary part of each code symbol.

2.3 Decoding Techniques

There are several decoding techniques for STBCs. The most common decoding techniques and their basics will be mentioned here. The detailed implementation will be presented in the next section.

2.3.1 Maximum Likelihood

The ML detection is the optimum decoding technique in sense of minimum codeword error probability. The detection is based on the Euclidean distance of the received codeword $\mathbf{Y}$ to codewords from the set of possible codewords $\mathcal{C}$. The valid codeword with minimum distance from the received $\mathbf{Y}$ is taken as the detected one

$$\hat{\mathbf{X}}_{ML} = \underset{\mathbf{X}\in\mathcal{C}}{\arg\min} ||\mathbf{Y} - \mathbf{H}\mathbf{X}||^2. \tag{2.38}$$

A disadvantage of the ML detection is that the Euclidean distance to each valid codeword has to be calculated, which is processing intensive and the processing demand grows with increasing symbol alphabet size.

2.3.2 Zero Forcing

This technique bases on the calculation of the pseudoinverse of the estimated channel coefficients matrix $\mathbf{H}$. It has low computational complexity, but its disadvantage is the noise amplification. As $\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{Z}$ is the received code matrix, with $\mathbf{Z}$ representing the noise, we can multiply it with the pseudoinverse of $\mathbf{H}$, which results in:

$$\hat{\mathbf{X}}_{ZF} = \mathcal{Q}\{(\mathbf{H}^{\mathbf{H}}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{H}}\mathbf{Y}\} = \mathcal{Q}\{\mathbf{X} + (\mathbf{H}^{\mathbf{H}}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{H}}\mathbf{Z}\} \tag{2.39}$$

with $\mathcal{Q}$ being a nonlinear function, describing the quantization introduced by a slicer.

2.3.3 Sphere Decoding

Sphere decoding is a detection technique for finding the nearest neighbor of the received vector by searching only lattice points that lie in a certain sphere of radius d around the received vector $\mathbf{Y}$, see Figure 2.5. The promise of sphere detection is to reduce the search space and the computation. The difficulty is to find the lattice points that lie inside a sphere without comparing all lattice points, which would omit the sense of sphere decoding. There is an efficient way to do this. A sphere of dimension one is simply an interval and the desired lattice points lie in this interval. By moving from an arbitrary dimension k, of which we

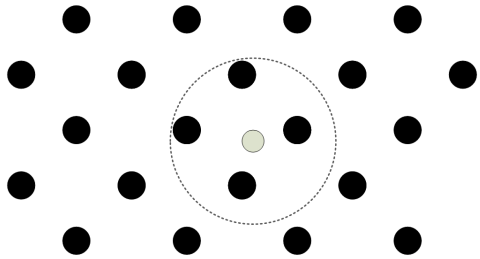


Figure 2.5: *Sphere detection concept*

know all points inside the sphere of radius d, to a dimension $k+1$, the detection of the points that lie in the same sphere with radius d in the $k+1$-th dimension breaks down to detection of points inside an interval of the $k+1$-th dimensional coordinate. This means that one can determine the lattice points inside a sphere of dimension m by successively detecting the lattice points of lower dimensions $1, 2, ..., m$ as proposed in [15]. Such search algorithms result in a tree search which will be described in detail in Chapter 3. There are two approaches on detection in general. Hard detection calculates an estimated symbol which contains a sequence of bits. Soft detection estimates bits that are mapped to a symbol and additionally provides a measure of reliability for each detected bit. These reliability measures, also called soft information, can be exploited by an appropriate channel decoder to improve decoding performance. The described soft sphere decoder in Chapter 3 provides such soft information by reducing computation complexity to the lattice points lying inside a sphere.

2.4 MIMO Simulator

Until now, the SISO IEEE 802.11p simulator and additionally basics of MIMO systems and STBCs were described. In this subsection, the implementation of MIMO in combination with STBCs into the simulator will be presented. The IEEE 802.11p standard defines only a SISO scheme. Previous work has been done to implement MIMO into the simulator [1]. This implementation adapts the IEEE 802.11n standard, which supports MIMO, to the IEEE 802.11p standard. Both standards have the same origin, namely IEEE 802.11a. Adapting a standard which has the same origin and is already in use has the advantage of a high reuse factor as hardware and software implementations of the mentioned standard already exist.

2.4.1 Conversion from IEEE 802.11n to IEEE 802.11p Standard

The preamble of IEEE 802.11n is adapted to IEEE 802.11p by reducing the bandwidth to 10 MHz and the corresponding discrete time tap period of 100 ns. In IEEE 802.11n it has two modes: mixed mode and greenfield-mode, but only the mixed mode suits the IEEE 802.11p

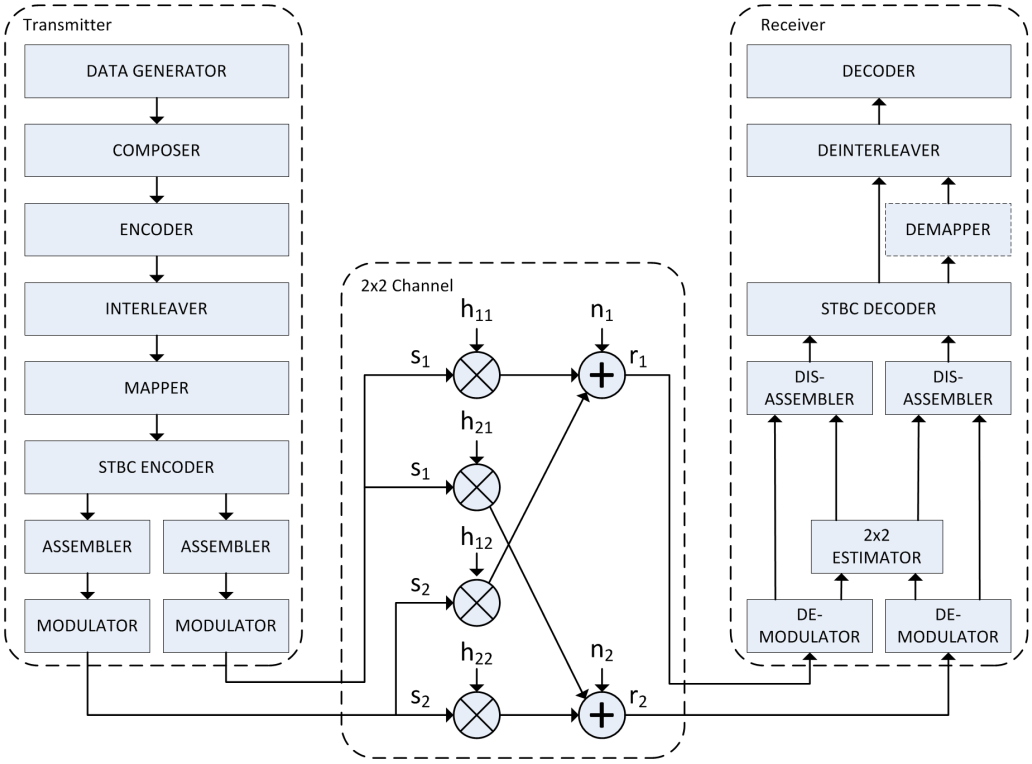


Figure 2.6: *MIMO simulator overview*

implementation. The adapted preamble is constructed in the following way: at the beginning the same two training fields and one signal field are sent as in the SISO case over one antenna and, with a delay δ, over the second antenna. After the training fields and signal, as shown in Figure 2.2 a new signal filed, called high throughput signal field (HT-SIG) follows. The HT-SIG contains information about the signal such as modulation, detection scheme, STBC, guard interval length etc. This new signal field is followed by a high throughput short training field (HT-STF), used for retraining the automatic gain control (AGC) unit, and two high throughput long training fields (HT-LTFs) used for MIMO channel estimation. As in the case of the SISO system, these fields are BPSK modulated. The delay δ on the second antenna is applied because the preambles contain the same information and thus sending same signals over two antennas leads to unintended beamforming. As the receiver can interpret the delayed signal from the second antenna as a multipath component, it might be treated as Inter Symbol Interference (ISI). To avoid this effect a diversity technique called cyclic delay diversity (CDD) is used. It cyclically shifts the first part of the symbol, with length δ, to the end of the symbol in the time domain, resulting in a phase shift in the frequency domain, transforming spatial diversity into frequency diversity.

2.4.2 Mapping of Space-Time Block Codes to OFDM Symbols

The implementation of the MIMO simulator is identical to that of SISO simulator for all entities including the mapper. As can be seen in Figure 2.6, after the mapper, the stream of symbols $\mathbf{s} = [s_1, s_2 ... s_N]$ is entering the STBC encoder, which, depending on the selected STBC, maps the symbols into a codeword matrix. Equations 2.25 and 2.26 describe the mapping for Alamouti and Golden Code, respectively. The mapping of the whole sequence of symbols results in a sequence of codeword matrices. For Alamouti STBC these matrices have the form:

$$s_1 s_2 \rightarrow \mathbf{X}_1, s_3 s_4 \rightarrow \mathbf{X}_2, \quad ... \quad s_{N-1} s_N \rightarrow \mathbf{X}_{N/2}, \tag{2.40}$$

and for Golden Code STBC:

$$s_1 s_2 s_3 s_4 \rightarrow \mathbf{X}_1, s_5 s_6 s_7 s_8 \rightarrow \mathbf{X}_2, \quad ... \quad s_{N-3} s_{N-2} s_{N-1} s_N \rightarrow \mathbf{X}_{N/4}. \tag{2.41}$$

The entries of the codeword matrices are represented by $c^n_{i,j}$ where i, j and n denote the transmit antenna, the time slot and the codeword matrix number in the sequence of all generated codeword matrices, respectively:

$$\mathbf{X}_n = \begin{bmatrix} c^n_{1,1} & c^n_{1,2} \\ c^n_{2,1} & c^n_{2,2} \end{bmatrix}. \tag{2.42}$$

Note that the time slot number refers to the considered codeword matrix. The first codeword matrix is sent as described in Equations 2.25 and 2.26. In OFDM only 48 subcarriers contain payload. In Figure 2.7 the rows represent the subcarriers and the columns represent the time slots. The structure of one STBC codeword matrix requires that its entries are spread over two time slots and two antennas. This means that 48 codeword matrices are arranged into two OFDM symbols at the first and two OFDM symbols at the second antenna. For antenna 1, which sends only entries $c^n_{1,1}$ in one time slot and $c^n_{1,2}$ in the next time slot, and antenna 2, which sends only entries $c^n_{2,1}$ in one time slot and $c^n_{2,2}$ in the next time slot, the sequence of codeword matrices is arranged as depicted in Figure 2.7. As each codeword matrix is spread over two timeslots, the number of OFDM symbols has to be an even number. This has to be taken into account at the data generator block. In case of Alamouti and Golden Code the final number of codeword matrices is therefore a multiple of 48. Alamouti codeword matrices contain information of 2 alphabet symbols, therefore the number of alphabet symbols is a multiple of 96, and for Golden Code a multiple of 192, as the Golden Code codeword matrix contains information of 4 alphabet symbols, see Table 2.1. The STBC encoder block

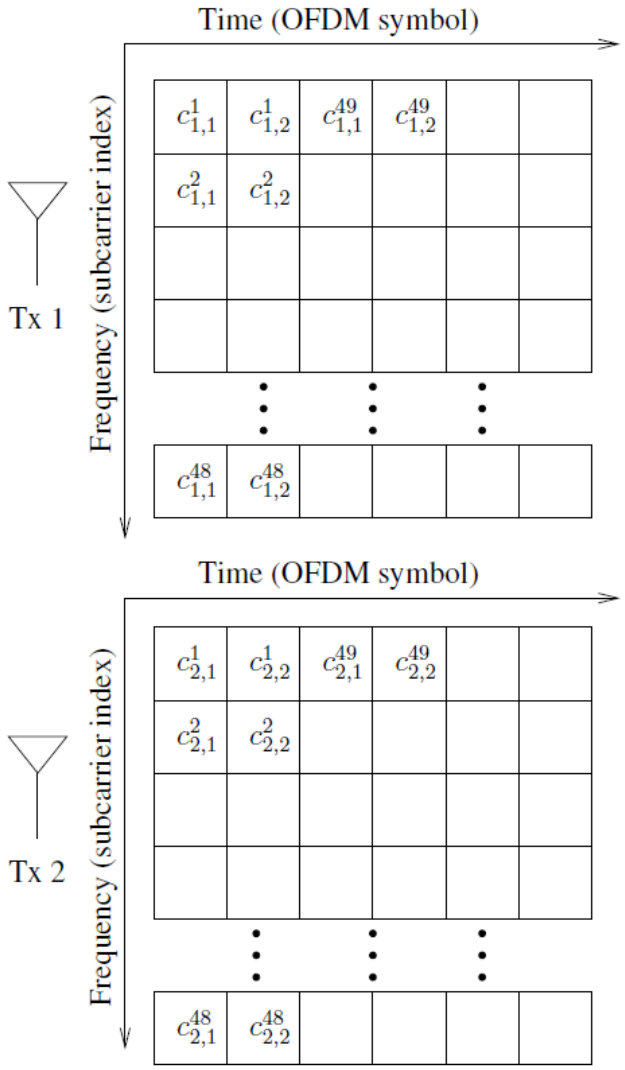


Figure 2.7: *Mapping of codeword matrix sequence to OFDM symbols [1]*

Minimum value of	**Alamouti**	**Golden Code**
OFDM symbols	2	2
Codeword matrices	48	48
Alphabet symbols	96	192

Table 2.1: *Mappings of alphabet symbols to OFDM symbols*

prepares the encoded data for the assembling into an OFDM structure by reordering the encoded symbols and splitting them into 2 streams, one for each antenna. The assembler block inserts the signal, preambles and the pilots as described in the SISO case. After that, each OFDM frame is being modulated and transmitted over its antenna. This part is equal to the SISO case, except that there are 2 radio frequency chains now.

2.4.3 2x2 Channel

The possible channel simulation models are the same as in the SISO case and are listed in Section 2.1.2. Because now 2 transmit and 2 receive antennas are present, the impulse response matrix which is describing the channel is not a single matrix anymore. The channel can rather be described by a matrix containing 4 submatrices, each one representing the influence of each transmit antenna to each receive antenna.

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{1,1} & \mathbf{H}_{1,2} \\ \mathbf{H}_{2,1} & \mathbf{H}_{2,2} \end{bmatrix} \quad \text{with} \quad \mathbf{H}_{x,y} \in \mathbb{C}^{M \times N}. \tag{2.43}$$

The submatrix $\mathbf{H}_{x,y}$ contains the influence of transmit antenna y to receive antenna x and the matrix' entries $h_{x,y}[m,n]$ denote the impulse response at m^{th} subcarrier in the n^{th} time slot. The received signals at antennas 1 and 2 at subcarrier m in time slot n, assuming transmit antenna 1 sends the signal s_1 and transmit antenna 2 sends the signal s_2, are then characterized by:

$$\begin{aligned} r_1[m,n] &= h_{1,1}[m,n]s_1 + h_{1,2}[m,n]s_2 + z_1[m,n] \\ r_2[m,n] &= h_{2,1}[m,n]s_1 + h_{2,2}[m,n]s_2 + z_1[m,n]. \end{aligned} \tag{2.44}$$

2.4.4 MIMO Receiver

The receiver block is depicted on the right side of Figure 2.6. Demodulation is done in the same way as for the SISO case for each receive antenna in parallel. To obtain the channel impulse response matrix described by Equation 2.43, estimation is done by the 2x2 estimator, which is using OFDM frames received on both antennas. In the disassembler block, the pilots, preamble and signal are removed from the OFDM symbols and also from the impulse response submatrices. By doing so only received data symbols are contained in the OFDM structures and their corresponding impulse responses in the impulse response matrices.

The STBC decoder extracts the received codeword matrices from the OFDM structures and decodes the STBC. The Alamouti code is decoded using the ZF decoding, described in Section 2.3.2, and the Golden Code is decoded using ML detection technique described in Section 2.3.1. Alternatively the SD with soft output or hard output can be used for the STBC

decoding. In case of soft output, the outcome of the decoder is a LLR sequence which contains information about reliability of each bit and is directly forwarded to the interleaver to restore the original order of the bitstream. When the hard output option is selected, the LLRs resulting form the SD are mapped to the symbols of the used symbol alphabet, similarly as at the output of the equalizer in the SISO case. These symbols need to be demapped to bits before entering the interleaver, so for the hard output option a demapper is still required. Note that the decoder is a Soft-Sphere decoder which is used for calculating the soft information, the hard output option is just implemented for comparison to the ML decoder, as the complexity of a hard output SD decoder is lower. The part from the interleaver is equal to the SISO receiver block.

2.4.5 STBC Decoding

The orthogonality of the codeword matrix for Alamouti Code boils the ML decoding technique down to ZF [12]. The symbols are transmitted according to Equation 2.25. The received symbols at the first time slot are:

$$\begin{aligned} y_{1,1} &= h_{1,1}s_1 + h_{1,2}s_2 + n_{1,1} \\ y_{1,2} &= h_{2,1}s_1 + h_{2,2}s_2 + n_{1,2}, \end{aligned} \tag{2.45}$$

and at the second time slot:

$$\begin{aligned} y_{2,1} &= h_{1,1}(-s_2^*) + h_{1,2}s_1^* + n_{2,1} \\ y_{2,2} &= h_{2,1}(-s_2^*) + h_{2,2}s_1^* + n_{2,2}. \end{aligned} \tag{2.46}$$

In order to factorize s_1 and s_2 out of the equations for the second time slot, both need to be conjugated, which results in:

$$\begin{aligned} y_{2,1}^* &= -h_{1,1}^*s_2 + h_{1,2}^*s_1 + n_{2,1}^* \\ y_{2,2}^* &= -h_{2,1}^*s_2 + h_{2,2}^*s_1 + n_{2,2}^*. \end{aligned} \tag{2.47}$$

Written in matrix form the subsequent receive symbols can be arranged in a vector, resulting in:

$$\begin{bmatrix} y_{1,1} \\ y_{1,2} \\ y_{2,1}^* \\ y_{2,2}^* \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} \\ h_{2,1} & h_{2,2} \\ h_{1,2}^* & -h_{1,1}^* \\ h_{2,2}^* & -h_{2,1}^* \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} n_{1,1} \\ n_{1,2} \\ n_{2,1} \\ n_{2,2} \end{bmatrix}, \tag{2.48}$$

Modulation	**Alphabet Size**	**Size of $\mathcal{C}$ (Alamouti)**	**Size of $\mathcal{C}$ (Golden Code)**
BPSK	2	4	16
QPSK	4	16	256
16-QAM	16	256	65536
64-QAM	64	4096	16777216 (2^{24})

Table 2.2: *Codeword matrix set sizes for defined modulation formats*

or, in a more comapct form:

$$\tilde{\mathbf{y}} = \tilde{\mathbf{H}}\mathbf{s} + \tilde{\mathbf{n}} \tag{2.49}$$

where $\tilde{\mathbf{H}}$ represents the virtual channel matrix, which is orthogonal and its pseudoinverse can be calculated. That pseudoinverse can be viewed as an equalizer

$$pinv(\tilde{\mathbf{H}}) = (\tilde{\mathbf{H}}^{\mathbf{H}}\tilde{\mathbf{H}})^{-1}\tilde{\mathbf{H}}^{\mathbf{H}} = \bar{\mathbf{H}} \quad \text{with} \quad \bar{\mathbf{H}} \in \mathbb{C}^{2\times 4}. \tag{2.50}$$

By left side multiplication of Equation 2.49 with $\bar{\mathbf{H}}$, $\hat{s}_1$ and $\hat{s}_2$ are obtained

$$\begin{bmatrix} \hat{s}_1 \\ \hat{s}_2 \end{bmatrix} = \bar{\mathbf{H}}\tilde{\mathbf{y}}. \tag{2.51}$$

These symbols need to be passed to a slicer to get valid detected symbols from the alphabet.

The Golden Code is decoded using the ML brute force technique, already mentioned in Section 2.3.1. The minimum Euclidean distance search performance depends on the alphabet size M_a. Equation 2.38 describes a minimum metric search over all $\mathbf{X} \in \mathcal{C}$. The set $\mathcal{C}$ is the set of all possible codeword matrices created by mapping 4 symbols of the alphabet to the Golden codeword matrix as described in Equation 2.26, therefore, the size of $\mathcal{C}$ is:

$$M_g = M_a^4. \tag{2.52}$$

In Table 2.2 the count of possible combinations of symbols for the Alamouti and for the Golden Code can be seen in dependence on the used modulation format. For the Golden Code in combination with 64-QAM there are 2^{24} possible combinations of the 4 symbols creating one codeword matrix and the ML decoding technique gets impractical at this point. To solve this problem the sphere decoder can be used. It can decode both, Alamouti and Golden Code and is described in the next chapter.

Transmission of one codeword matrix is spread over two time slots, in the first one we

receive:

$$\begin{aligned} y_{1,1} &= h_{1,1}c_{1,1} + h_{1,2}c_{2,1} + n_{1,1} \\ y_{2,1} &= h_{2,1}c_{1,1} + h_{2,2}c_{2,1} + n_{2,1}, \end{aligned} \tag{2.53}$$

and in the second time slot:

$$\begin{aligned} y_{1,2} &= h_{1,1}c_{1,2} + h_{1,2}c_{2,2} + n_{1,2} \\ y_{2,2} &= h_{2,1}c_{1,2} + h_{2,2}c_{2,2} + n_{2,2}. \end{aligned} \tag{2.54}$$

Stacking the matrices into a vector, so that the first time slot is represented by the upper part, and the second by the lower part of the reception, gives following equation:

$$\begin{bmatrix} y_{1,1} \\ y_{2,1} \\ y_{1,2} \\ y_{2,2} \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} & 0 & 0 \\ h_{2,1} & h_{2,2} & 0 & 0 \\ 0 & 0 & h_{1,2} & h_{1,1} \\ 0 & 0 & h_{2,2} & h_{2,1} \end{bmatrix} \begin{bmatrix} c_{1,1} \\ c_{2,1} \\ c_{1,2} \\ c_{2,2} \end{bmatrix} + \begin{bmatrix} n_{1,1} \\ n_{2,2} \\ n_{1,2} \\ n_{2,2} \end{bmatrix}. \tag{2.55}$$

The ML decoder calculates all possible combinations of input matrix $\mathbf{X}$ and uses the estimated channel matrix $\mathbf{H}$ to calculate $\mathbf{HX}$ and the minimum distance from the received $\mathbf{Y}$.

3

Soft Sphere Decoding

In the previous chapter transmission of data in a MIMO system was described and its implementation into the IEEE 802.11p PHY simulator in combination with STBC was introduced. Despite the broad number of advantages brought by MIMO systems and STBCs, they introduce higher processing demands. When considering the mapping of alphabet symbols to multiple antenna schemes, without focusing on a special STBC, it can be realized that the set of valid combinations, i.e. codeword matrices, that can be sent increases. Therefore a larger set of valid codeword matrices requires more computational power, especially if the optimum ML decoder is used. In Section 2.3.3 the sphere decoder as a solution to find nearest neighbors in an effective manner was introduced. The performance increase resulting from the soft information at the receiver motivates the implementation of a sphere decoder with soft output. The promise is a reduced processing demand combined with improved performance in comparison to hard detection.

3.1 2x2 MIMO Reception

In this section the input data for the soft sphere decoder is introduced. Both, Alamouti and Golden Code implementations produce a codeword matrix $\mathbf{X}$ which is sent through a 2x2 MIMO channel over two time slots. The channel introduces distortion which is described by the impulse response and adds Gaussian noise.

In case of the Golden Code the received vector is described by Equation 2.55 and can be rewritten as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}. \tag{3.1}$$

The coded vector $\mathbf{x}$ from Equation 2.34 can be factorized now without separating real and imaginary parts:

$$\begin{aligned}
\mathbf{x} = \begin{bmatrix} c_{1,1} \\ c_{2,1} \\ c_{1,2} \\ c_{2,2} \end{bmatrix} &= \frac{1}{\sqrt{5}} \begin{bmatrix} [1+j\sigma(\theta)]s_1 + [\theta - j]s_2 \\ [j-\theta]s_3 + [1+j\sigma(\theta)]s_4 \\ [1+j\sigma(\theta)]s_3 + [\theta - j]s_4 \\ [1+j\theta]s_1 + [\sigma(\theta) - j]s_2 \end{bmatrix} \\
&= \frac{1}{\sqrt{5}} \begin{bmatrix} 1+j\sigma(\theta) & \theta - j & 0 & 0 \\ 0 & 0 & j-\theta & 1+j\sigma(\theta) \\ 0 & 0 & 1+\sigma(\theta) & \theta - 1 \\ 1+j\theta & \sigma(\theta) & 0 & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} \\
&= \mathbf{G}\mathbf{s}
\end{aligned} \tag{3.2}$$

Inserting this result into Equation 3.1 gives

$$\mathbf{y} = \mathbf{H}\mathbf{G}\mathbf{s} + \mathbf{n} = \mathbf{H}_{\text{eff}}\mathbf{s} + \mathbf{n} \tag{3.3}$$

with $\mathbf{H}_{\text{eff}} = \mathbf{H}\mathbf{G}$ now containing the influence of the channel impulse response and the shifts of the alphabet symbols introduced by the Golden Code STBC. By using $\mathbf{H}_{\text{eff}}$ instead of $\mathbf{H}$ the decoder directly decodes the vector of alphabet symbols $\mathbf{s}$, i.e. their soft information without need for knowledge of the used STBC.

Similarly for the Alamouti Code with received vector given in Equation 2.51, we have the

following input-output vector relationship:

$$\begin{bmatrix} y_{1,1} \\ y_{2,1} \\ y_{1,2}^* \\ y_{2,2}^* \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} & 0 & 0 \\ h_{2,1} & h_{2,2} & 0 & 0 \\ 0 & 0 & h_{1,2}^* & -h_{1,1}^* \\ 0 & 0 & h_{2,2}^* & -h_{2,1}^* \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} n_{1,1} \\ n_{2,1} \\ n_{1,2} \\ n_{2,2} \end{bmatrix}, \tag{3.4}$$

or in compact form:

$$\tilde{\mathbf{y}} = \mathbf{H}_{\text{eff}}\mathbf{s} + \mathbf{n}. \tag{3.5}$$

As we receive the vector $\mathbf{y} = [y_{1,1} \quad y_{2,1} \quad y_{1,2} \quad y_{2,2}]^T$ we need to take the conjugates of the last two entries to get $\tilde{\mathbf{y}}$ and pass it together with $\mathbf{H}_{\text{eff}}$ to the decoder which can now detect the symbols in vector $\mathbf{s}$, i.e. the soft information of the receive vector.

3.2 Soft Information

Soft information is defined as the reliability of one bit detection. This information can be utilized by channel decoders, e.g. decoders using the BCJR algorithm. The soft information is usually expressed in form of LLRs, i.e. the ratio of the a posteriori probabilities of a detected bit being either 1 or 0, given in Equation 2.20. We now rewrite this equation as

$$L(b_j|\mathbf{y}) = \ln \frac{P(b_j = 1|\mathbf{y})}{P(b_j = 0|\mathbf{y})}. \tag{3.6}$$

The LLRs are now denoted with L and the coded bits are denoted as b_j, where j is the index inside the detected codeword, given the receive vector $\mathbf{y}$. The sign of the LLR value $sign(L(b_j|\mathbf{y}))$ gives information about the bit being 1 or 0 and its magnitude $|L(b_j|\mathbf{y})|$ is a measure for the reliability. Note that the mentioned soft information in Section 2.1.3 is based on the received alphabet symbols in case of the SISO transmission scheme. For IEEE 802.11p the received alphabets include BPSK, QPSK, 16-QAM or 64-QAM modulation schemes. That means the lattice is represented by the symbol constellation. Now, we need to calculate LLRs of codewords that are created by mapping multiple alphabet symbols to STBC codeword matrices, i.e.LLRs of the vectors $\mathbf{s}$ from Equations 3.3 and 3.5.

Applying Bayes' theorem, Equation 3.6 becomes

$$L(b_j|\mathbf{y}) = \ln \frac{p(\mathbf{y}|b_j = 1)\frac{P(b_j=1)}{p(\mathbf{y})}}{p(\mathbf{y}|b_j = 0)\frac{P(b_j=0)}{p(\mathbf{y})}} = \ln \frac{p(\mathbf{y}|b_j = 1)P(b_j = 1)}{p(\mathbf{y}|b_j = 0)P(b_j = 0)} \tag{3.7}$$

with $p(\mathbf{y}|b_j)$ being the conditional probability density function of receive vector $\mathbf{y}$ conditioned on b_j and $P(b_j = 1)$ and $P(b_j = 0)$ the probabilites of b_j being either 1 or 0. By transforming the conditional pdfs $p(\mathbf{y}|b_j = 1)$ and $p(\mathbf{y}|b_j = 0)$ and assuming that both cases $b_j = 1$ and $b_j = 0$ are equally likely, we get [16]:

$$L(b_j|\mathbf{y}) = \ln \frac{\sum\limits_{\mathbf{b}\in\chi_j^{(1)}} p(\mathbf{y}|\mathbf{b})P(\mathbf{b})}{\sum\limits_{\mathbf{b}\in\chi_j^{(0)}} p(\mathbf{y}|\mathbf{b})P(\mathbf{b})} = \ln \frac{\sum\limits_{\mathbf{b}\in\chi_j^{(1)}} p(\mathbf{y}|\mathbf{b})}{\sum\limits_{\mathbf{b}\in\chi_j^{(0)}} p(\mathbf{y}|\mathbf{b})}. \tag{3.8}$$

In above equation, $\chi_j^{(1)}$ is the set of all possible transmit vectors with a 1 at the j^{th} bit position and $\chi_j^{(0)}$ the set of all possible transmit vectors with a 0 at the j^{th} bit position. Further, due to the noise statistics, we assume that the pdf of $\mathbf{y}$ corresponds to a joint pdf of a multi-variate complex Gaussian random variable with independent and identically distributed (i.i.d.) circularly symmetric components, each with variance σ_n^2 and mean $\mathbf{H_{eff}s}$

$$p(\mathbf{y}|\mathbf{b}) = p(\mathbf{y}|\mathbf{s} = \mathbf{s}(\mathbf{b})) = \frac{1}{(\pi\sigma_n^2)^{M_R}} e^{-\frac{1}{\sigma_n^2}||\mathbf{y}-\mathbf{H_{eff}s}(\mathbf{b})||^2}. \tag{3.9}$$

Inserting Equation 3.9 into Equation 3.8 results in logarithms of a sum of exponentials

$$\begin{aligned} L(b_j|\mathbf{y}) &= \ln \frac{\sum\limits_{\mathbf{b}\in\chi_j^{(1)}} e^{-\frac{1}{\sigma_n^2}||\mathbf{y}-\mathbf{H_{eff}s}(\mathbf{b})||^2}}{\sum\limits_{\mathbf{b}\in\chi_j^{(0)}} e^{-\frac{1}{\sigma_n^2}||\mathbf{y}-\mathbf{H_{eff}s}(\mathbf{b})||^2}} \\ &= \ln \sum_{\mathbf{b}\in\chi_j^{(1)}} e^{-\frac{1}{\sigma_n^2}||\mathbf{y}-\mathbf{H_{eff}s}(\mathbf{b})||^2} - \ln \sum_{\mathbf{b}\in\chi_j^{(0)}} e^{-\frac{1}{\sigma_n^2}||\mathbf{y}-\mathbf{H_{eff}s}(\mathbf{b})||^2}. \end{aligned} \tag{3.10}$$

For simpler notation, we introduce the Jacobian logarithm which is defined as:

$$\text{jacln}\,(a_1, a_2, ..., a_N) \triangleq \ln\,(e^{a_1} + e^{a_2} + ... + e^{a_N}), \tag{3.11}$$

and insert it in Equation 3.10:

$$L(b_j|\mathbf{y}) = \underset{\mathbf{b}\in\chi_j^{(1)}}{\text{jacln}}\,\big(-\frac{1}{\sigma_n^2}||\mathbf{y} - \mathbf{H_{eff}s}(\mathbf{b})||^2\big) - \underset{\mathbf{b}\in\chi_j^{(0)}}{\text{jacln}}\,\big(-\frac{1}{\sigma_n^2}||\mathbf{y} - \mathbf{H_{eff}s}(\mathbf{b})||^2\big). \tag{3.12}$$

In order to make the calculation more suitable for digital signal processors (DSPs), some

reformulations can be done. For an arbitrary number of summands, we have:

$$\begin{aligned}
\underset{i}{\text{jacln}}(x_i) &= \ln\big(\sum_i e^{x_i}\big) \quad \text{with} \quad m = \max_i (x_i) \\
&= \ln\big(\sum_i \frac{e^m}{e^m} e^{x_i}\big) \\
&= \ln\big(e^m \sum_i \frac{e^{x_i}}{e^m}\big) \\
&= \ln\big(e^m\big) + \ln\big(\sum_i e^{x_i - m}\big) \\
&= m + \ln\big(\sum_i e^{x_i - m}\big).
\end{aligned}$$

This results in the reformulated Jacobian logarithm

$$\underset{i}{\text{jacln}}(x_i) = \max_i (x_i) + \ln\big(\sum_i e^{x_i - m}\big), \quad \text{with} \quad e^{x_i - m} < 1. \tag{3.13}$$

For DSPs, this simplification can be implemented as a maximum search over the x_i's with a correction factor $\ln\big(\sum_i e^{x_i - m}\big)$. As m is the maximum among the x_i's, the correction factor is small and in DSPs without hardware units for logarithm or exponential function calculation it can be omitted, which results in the max-log approximation:

$$\underset{i}{\text{jacln}}(x_i) \approx \max_i (x_i) \tag{3.14}$$

This approximation can be used to calculate approximated LLR values by inserting into Equation 3.12

$$L(b_j|\mathbf{y}) \approx \max_{\mathbf{b} \in \chi_j^{(1)}} (-\frac{1}{\sigma_n^2} ||\mathbf{y} - \mathbf{H_{eff}}\mathbf{s}(\mathbf{b})||^2) - \max_{\mathbf{b} \in \chi_j^{(0)}} (-\frac{1}{\sigma_n^2} ||\mathbf{y} - \mathbf{H_{eff}}\mathbf{s}(\mathbf{b})||^2) \tag{3.15}$$

and finally, omitting the noise variance, we get

$$L(b_j|\mathbf{y}) \approx \min_{\mathbf{b} \in \chi_j^{(0)}} (||\mathbf{y} - \mathbf{H_{eff}}\mathbf{s}(\mathbf{b})||^2) - \min_{\mathbf{b} \in \chi_j^{(1)}} (||\mathbf{y} - \mathbf{H_{eff}}\mathbf{s}(\mathbf{b})||^2). \tag{3.16}$$

The performance degradation of max-log approximated LLRs is often very small [17]. Another important fact is that for each bit, one of these minima always corresponds to the ML solution

$$\lambda^{\text{ML}} = \min_{\mathbf{s} \in \mathcal{A}^{M_T}} ||\mathbf{y} - \mathbf{H_{eff}}\mathbf{s}||^2 = ||\mathbf{y} - \mathbf{H_{eff}}\hat{\mathbf{s}}_{\text{ML}}||^2. \tag{3.17}$$

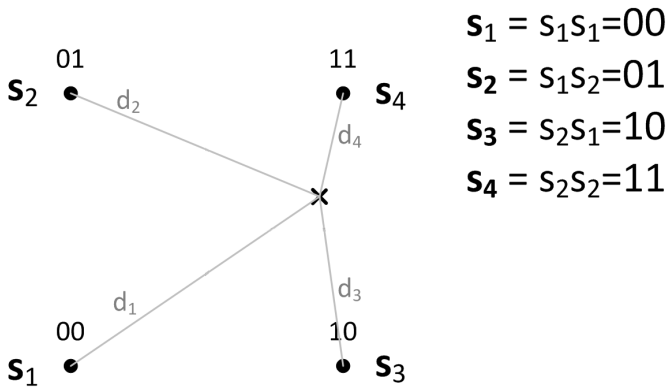


Figure 3.1: *Soft information calculation*

Here $\mathcal{A}^{M_T}$ is the set of all possible symbol vectors $\mathbf{s}$ and due to the unique correspondence of each symbol vector $\mathbf{s}$ to a bit vector $\mathbf{b}$, we can write:

$$\mathcal{A}^{M_T} = \chi_j^{(1)} \cup \chi_j^{(0)}, \tag{3.18}$$

i.e. the ML solution is searched over the set of all possible symbol (bit) vectors. The ML solution corresponds to a bit label $\mathbf{s_{ML}} \equiv \mathbf{b_{ML}}$. If in the ML bit label the j^{th} bit is a 0, i.e. from the set $\chi_j^{(0)}$, the other minima in Equation 3.16 is found in the *counter-set* $\chi_j^{(1)}$ and vice versa. In literature, the *counter-set* is usually denoted with $\chi_j^{(\overline{b_j^{\mathrm{ML}}})}$ and the corresponding metric with $\lambda^{\overline{\mathrm{ML}}}$. We summarize, that the max-log approximated LLR value calculation narrows down to calculation of the ML metric and the metrics of the (bit wise) counter hypothesis, which can be written in following form

$$L(b_j|\mathbf{y}) = \begin{cases} \lambda^{\mathrm{ML}} - \lambda_j^{\overline{\mathrm{ML}}} & \text{if} \quad b_j^{\mathrm{ML}} = 0 \\ \lambda_j^{\overline{\mathrm{ML}}} - \lambda^{\mathrm{ML}} & \text{if} \quad b_j^{\mathrm{ML}} = 1 \end{cases} \tag{3.19}$$

For the ease of notation, we will further write $\chi_j^{\overline{\mathrm{ML}}}$ instead of $\chi_j^{(\overline{b_j^{\mathrm{ML}}})}$ to denote the *counter-set*.

The above result can be exemplified by considering a simple 2x2 MIMO system with BPSK modulation. As the BPSK modulation is used, the symbols s_1 and s_2 correspond to bits 0 and 1, i.e. $\mathcal{A} = \{s_1, s_2\} = \{0, 1\}$. The 2x2 MIMO system sends symbol vectors of 2 BPSK symbols, that means we have a set $\mathcal{A}^{M_T} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4\}$ which are all possible combinations of the BPSK symbols and are defined as in Figure 3.1. Further, we assume an AWGN channel with channel impulse response matrix being $\mathbf{I}$, so $\mathbf{Hs} = \mathbf{s}$. In case of a received vector $\mathbf{y}$ as depicted in Figure 3.1, the ML solution is obviously $\mathbf{s}_4$ with $\lambda^{\mathrm{ML}} = d_4$. For the calculation of the LLR for the first bit we have the set $\chi_0^{(0)} = \{\mathbf{s}_1, \mathbf{s}_2\}$ and $\chi_0^{(1)} = \{\mathbf{s}_3, \mathbf{s}_4\}$ and the search

for the two minima in Equation 3.16 results in

$$L(b_1|\mathbf{y}) = d_2 - d_4 = \lambda_1^{\overline{\mathrm{ML}}} - \lambda^{\mathrm{ML}} > 0 \Rightarrow b_1 = 1. \tag{3.20}$$

For the second bit we have sets $\chi_1^{(0)} = \{\mathbf{s}_1, \mathbf{s}_3\}$ and $\chi_1^{(1)} = \{\mathbf{s}_2, \mathbf{s}_4\}$ and Equation 3.16 gives

$$L(b_2|\mathbf{y}) = d_3 - d_4 = \lambda_0^{\overline{\mathrm{ML}}} - \lambda^{\mathrm{ML}} > 0 \Rightarrow b_2 = 1 \tag{3.21}$$

for the LLR value for the second bit. According to the distances, we can see that $d_2 - d_4 > d_3 - d_4$, i.e. $L(b_0|\mathbf{y}) > L(b_1|\mathbf{y})$, which means that the reliability of the first bit decision for 1 is larger than the reliability of the second bit decision for 1. This is visualized in Figure 3.1 as the received vector $\mathbf{y}$ is pretty far on the right side where the symbols have the first bit equal 1, but for the second bit the vector is not far from the threshold (the I-axis) between the second bit being 1 or 0.

Above example can be expanded to different modulation formats and MIMO schemes. Although the LLR calculation is simplified, the calculation of all LLRs is still expensive as for the λ^{ML} metric a search over $|\mathcal{A}^{M_T}| = 2^{l \cdot M_T}$ (with l from Equation 2.1) different symbol vectors is required and for each of the $\lambda_j^{\overline{\mathrm{ML}}}$ metrics with $j = 1, \cdots, l \cdot M_T$ a search over $2^{l \cdot M_T - 1}$ symbol vectors is required.

3.3 Tree Search

In order to transform the metric search into a tree search problem, we need to reformulate our input-output relation. This can be done with the QR decomposition of the channel impulse response matrix $\mathbf{H}$. For the search of the two minima from Equation 3.16 there are several soft sphere decoding algorithms that will be described in the remainder of this section.

3.3.1 Transformation into a Tree Search Problem

Applying the QR decomposition, also called QR factorization, to a matrix $\mathbf{A}$ decomposes the matrix into an orthogonal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$. There are several methods to do that. One of them is the Gram-Schmidt orthogonalization of the matrix that we wish to decompose, which results in a matrix with orthogonal columns, followed by normalization of these orthogonal colums in order to make them orthonormal. The resulting matrix $\mathbf{Q}$ represents the canonical basis spanning a vector space, while the upper triangular matrix $\mathbf{R}$ contains the linear coefficients for representing the decomposed matrix $\mathbf{A}$ in the

spanned vector space.

$$\mathbf{A} = \mathbf{QR} \quad \text{with} \quad \mathbf{A} \in \mathbb{C}^{n \times m}. \tag{3.22}$$

Furthermore if $n = m$, then $\mathbf{Q}$ is unitary, i.e. $\mathbf{Q^H Q} = \mathbf{Q Q^H} = \mathbf{I}$ and this is fulfilled in the case of a 2x2 MIMO system. As we wish to decompose the channel impulse response matrix $\mathbf{H}$, we can assume that its entries are random and therefore this matrix is always nonsingular ($\mathbf{Hx} = \mathbf{0}$ only for $\mathbf{x} = \mathbf{0}$) i.e. for randomly selected coefficients the probability of linear dependent rows is very small. For unitary $\mathbf{Q}$ and nonsingular $\mathbf{H}$, $\mathbf{R}$ may be chosen so that its entries are real valued and positive and $\mathbf{Q}$ and $\mathbf{R}$ are unique [16]. By applying the QR decomposition to the channel matrix and left side multiplying the input-output relation 3.1 (or equivalently 3.3) by $\mathbf{Q^H}$, the modified input-output relation is obtained:

$$\mathbf{Q^H y} = \tilde{\mathbf{y}} = \mathbf{Rs} + \mathbf{Q^H n}. \tag{3.23}$$

The effect of $\mathbf{Q^H}$ on the noise is negligible and we obtain following for the metrics from Equation 3.16:

$$\begin{aligned} \lambda^{\mathrm{ML}} &= \min_{\mathbf{b} \in \mathcal{A}^{M_T}} ||\tilde{\mathbf{y}} - \mathbf{Rs}||^2, \\ \lambda_j^{\overline{\mathrm{ML}}} &= \min_{\mathbf{b} \in \chi_j^{\overline{\mathrm{ML}}}} ||\tilde{\mathbf{y}} - \mathbf{Rs}||^2. \end{aligned} \tag{3.24}$$

The triangular form of $\mathbf{R}$ transforms the calculation of the Euclidean distance $\mathbf{d(s)} = ||\tilde{\mathbf{y}} - \mathbf{Rs}||^2$ into a more simplified form

$$\begin{aligned} \mathbf{d(s)} = ||\tilde{\mathbf{y}} - \mathbf{Rs}||^2 &= \underbrace{|\tilde{y}_{M_T} - R_{M_T,M_T} s_{M_T}|^2}_{|e_{M_T}|^2} \\ &+ \underbrace{|\tilde{y}_{M_T-1} - R_{M_T-1,M_T-1} s_{M_T-1} - R_{M_T-1,M_T} s_{M_T}|}_{|e_{M_T-1}|^2} \\ &+ \dots \\ &+ \underbrace{|\tilde{y}_1 - R_{1,1} s_1 - R_{1,2} s_2 - \dots - R_{1,M_T} s_{M_T}|^2}_{|e_1|^2} \\ &= \sum_{i=1}^{M_T} |e_i|^2. \end{aligned} \tag{3.25}$$

The summands $|e_i|^2$ of the Euclidean distance are called distance increments (DIs). By summing up the DIs it is possible to recursively calculate $\mathbf{d}(\mathbf{s})$:

$$|e_i|^2 = \left| \tilde{y}_i - \sum_{k=i}^{M_T} R_{i,k} s_k \right|^2 . \tag{3.26}$$

The partial sum of the Euclidean distances, e.g. from the index i to index M_T is called Partial Euclidean Distance (PED) and can be defined as the $(i+1)$-th PED plus the i-th distance increment:

$$d_i = d_{i+1} + |e_i|^2. \tag{3.27}$$

This allows us to introduce the Partial Symbol Vectors (PSVs) $\mathbf{s}^{(i)} = [s_i \dots s_{M_T}]$ which can be arranged into a tree as nodes with the tree having its root above level $i = M_T$ and its leaves at level $i = 1$. The dependence of the PEDs d_i on the symbol vector $\mathbf{s}$ is only through the PSVs $\mathbf{s}^{(i)}$. This turns the ML search into a tree search problem [18] where the PSVs and PEDs correspond to nodes and the DIs to branches. Such an arrangement can be seen in Figure 3.2. For the DI with index $M_T + 1$ we choose 0 as initialization value $d_{M_T+1} = 0$. The leaves of the tree represent one of the possible $2^{l \cdot M_T}$ symbol vectors $\mathbf{s} \in \mathcal{A}^{M_T}$. The PEDs of the according PSVs that correspond to nodes, can be calculated by summing up the DIs of the PSV from the root.

3.3.2 Sphere Decoding Algorithms

There are different methods to traverse the tree and search for the metric of the ML solution λ^{ML} and the metrics of the counter-hypothesis $\lambda_j^{\overline{\mathrm{ML}}}$. One of the first detection algorithms known as SD was developed in 1981 by Pohst [19]. Later it was further improved. Notable are the improvements by Viterbo and Biglieri [20], applying the algorithm in lattice decoding in 1993 and the improvement by Schnorr and Euchner in 1994 [21] which leads to reduced complexity.

Schnorr-Euchner Sphere Decoder

The Schnorr-Euchner Sphere Decoder (SESD) algorithm [21] starts traversing the tree with an initial radius of $r = \infty$. It uses a technique known as radius reduction. This technique handles the disadvantage of a fixed radius which, when chosen too small may lead to no result (no lattice point in sphere around $\tilde{\mathbf{y}}$) or, when chosen too large leads to high computational complexity as many nodes will be visited. Traversing is done in ascending order of the PEDs of each node, which leads to effective tree pruning. When the algorithm comes to the first

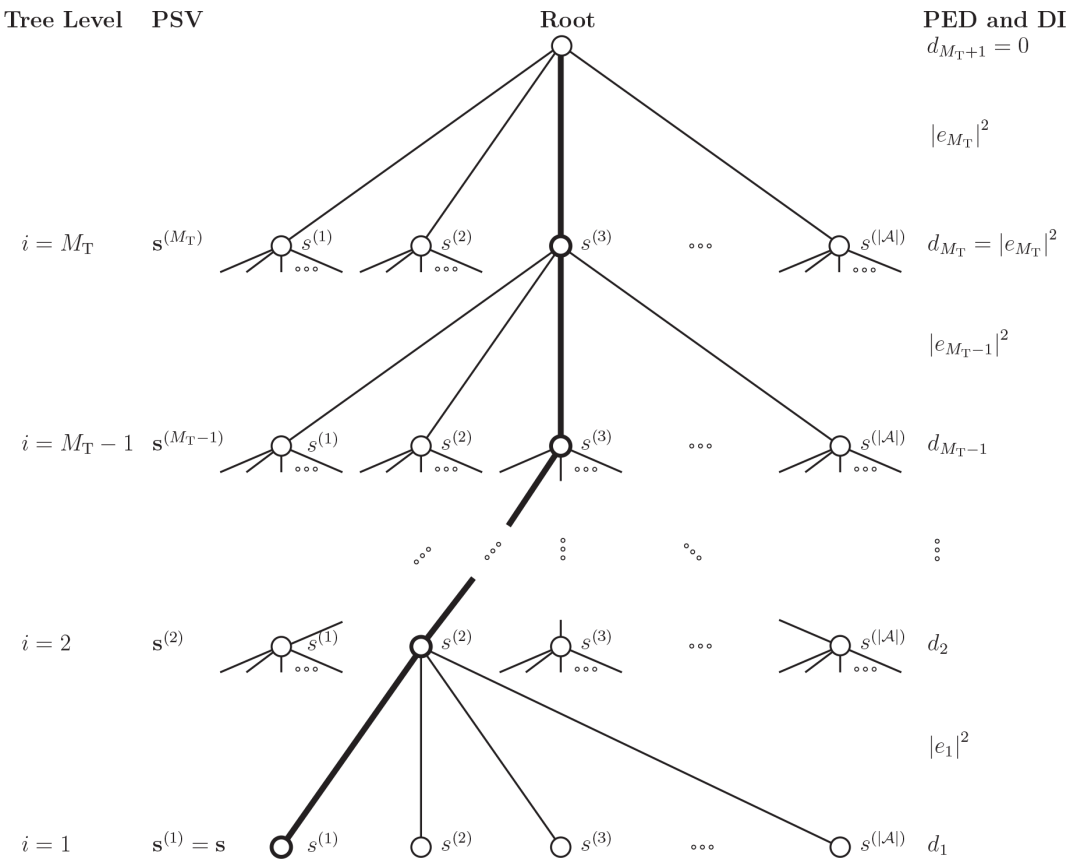


Figure 3.2: *Tree for a $M_T \times M_R$ MIMO system [16]*

leaf, the obtained metric is set to be the new radius as it is smaller than ∞, thereby it is ensured that at least one lattice point is found. Every time a leaf is reached that has smaller metric than the current radius it becomes the new radius, hence the name radius reduction. Each time a node is visited whose PED is larger than the actual radius, i.e. the sphere constraint (Equation 3.28) is violated, the children nodes of that node are not traversed, i.e. the tree is pruned, as the PED can only grow when traversing further children nodes

$$d(\mathbf{s}^{(i)}) < r^2. \tag{3.28}$$

However, the SESD algorithm finds the ML solution in an effective manner but does not provide a solution for finding the metrics of the counter hypothesis.

Repeated Tree Search

Another SD algorithm is the repeated tree search (RTS) [22]. It finds the ML solution using the SESD and also the metrics of the counter hypothesis. When the ML solution is found, the tree is prepruned by excluding all nodes that do not have leaves in the counter sets $\chi_j^{\overline{\text{ML}}}$ and SESD is reran to find the $\lambda_j^{\overline{\text{ML}}}$ metrics. Since the prepruning still results in a large tree and the SESD algorithm has to be reran for each $\lambda_j^{\overline{\text{ML}}}$, i.e. $l \cdot M_T$ times, the complexity of the RTS algorithm increases. There are many redundant computations and nodes are visited multiple times.

Single Tree Search

The Single Tree Search (STS) algorithm [18] visits each node of the tree only once. This can be achieved by searching the ML solution and the $\lambda_j^{\overline{\text{ML}}}$ metrics concurrently. Pruning is done when the subtree originating from a node can not lead to an update of either λ^{ML} or one of the $\lambda_j^{\overline{\text{ML}}}$. The algorithm itself administrates a list which contains the actual ML metric λ^{ML}, the bit label of the actual ML solution $\mathbf{b}_{\mathbf{ML}}$ and the metrics of the counter hypothesis $\lambda_j^{\overline{\text{ML}}}$. Note that for an arbitrary MIMO scheme and arbitrary modulation format the current bit label of the ML solution and the bit label of the counter hypothesis are given by, for example:

$$\mathbf{b}_{\mathbf{ML}} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \mathbf{b}_{\overline{\text{ML}}} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}. \tag{3.29}$$

The algorithm contains always preliminary values until the end of the search is reached. It starts by setting all metrics in the list to infinity: $\lambda^{\text{ML}} = \lambda_j^{\overline{\text{ML}}} = \infty$ for all j. When one leaf is reached, i.e. we have calculated the metric for one symbol vector in the lattice, the algorithm

does the following:

1. If the calculated metric $d(\mathbf{b}) < \lambda^{\mathrm{ML}}$, which means that we have found a new ML solution, the metrics for the counter hypothesis get updated by the former ML metric for all bits that change in the new ML bit label. For example, if the former bit label was given by $\mathbf{b}_{\mathbf{ML,old}}$ and should be updated to the new bit label $\mathbf{b}_{\mathbf{ML,new}}$:

$$\mathbf{b}_{\mathbf{ML,old}} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \mathbf{b}_{\mathbf{ML,new}} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \tag{3.30}$$

   then only the second bit has changed in the update process and the counter hypothesis metric for that bit is getting updated by the value of the former ML metric: $\lambda_2^{\overline{\mathrm{ML}}} \leftarrow \lambda^{\mathrm{ML}}$. After the counter hypothesis metrics got updated, the new ML metric is set $\lambda^{\mathrm{ML}} \leftarrow d(\mathbf{b})$ and the bit label is updated $\mathbf{b}_{\mathbf{ML}} \leftarrow \mathbf{b}_{\mathbf{ML,new}}$.

2. If $d(\mathbf{b}) > \lambda^{\mathrm{ML}}$, i.e. no new ML solution is found, only the metrics of the counter hypothesis for which the checked label $\mathbf{b}$ has bits opposite to the actual ML bit label, need to be checked. For all j with $b_j = b_j^{\overline{\mathrm{ML}}}$ and the metric $d(\mathbf{b}) < \lambda_j^{\overline{\mathrm{ML}}}$, this metric gets updated $\lambda_j^{\overline{\mathrm{ML}}} \leftarrow d(\mathbf{b})$. Let us, for example, assume that the actual ML bit label $\mathbf{b}_{\mathbf{ML}}$ and the bit label of the reached leaf $\mathbf{b}$ are given by:

$$\mathbf{b}_{\mathbf{ML}} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \tag{3.31}$$

   then we have to check only the counter hypothesis metrics of the last two bits ($\lambda_3^{\overline{\mathrm{ML}}}$ and $\lambda_4^{\overline{\mathrm{ML}}}$) as they differ from the bits in the ML label. Should one of the metrics be larger than the metric of the reached leaf, it gets updated, i.e. if $\lambda_3^{\overline{\mathrm{ML}}} > d(\mathbf{b})$, then $\lambda_3^{\overline{\mathrm{ML}}} \leftarrow d(\mathbf{b})$, if $\lambda_4^{\overline{\mathrm{ML}}} > d(\mathbf{b})$, then $\lambda_4^{\overline{\mathrm{ML}}} \leftarrow d(\mathbf{b})$.

The above described steps concern the update process of the list and the preliminary values. In order to increase the efficiency and traverse the tree only once, the pruning criterion has to be considered. When a node $\mathbf{s}^{(i)}$ is reached, its subtree gets excluded if its traversing does not lead to an update of the ML metric or the metrics of the counter hypothesis. The node $\mathbf{s}^{(i)}$ corresponds to the PSV $\mathbf{s}^{(i)} = [s^{(i)} \dots s^{M_T}]$ with PED $d_i = d(\mathbf{s}^{(i)})$ and the corresponding (partial) bit label $\mathbf{b}^{(i)} = [\mathbf{b}_{(i-1)\cdot l+1}, \mathbf{b}_{(i-1)\cdot l+2}, \dots, \mathbf{b}_{l\cdot M_T-2}, \mathbf{b}_{l\cdot M_T-1}, \mathbf{b}_{l\cdot M_T}]$.

In order to clarify the indexing, again an example will be presented. Let us assume QPSK modulated symbols, i.e. each symbol contains two bits ($l = 2$), and a MIMO scheme with 5 transmit antennas $M_T = 5$. Then, if $i = 4$, we have for the PSV and the corresponding partial bit label:

$$\mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix} \equiv \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \end{bmatrix} \qquad \mathbf{s}^{(i)} = \mathbf{s}^{(4)} = \begin{bmatrix} s_4 \\ s_5 \end{bmatrix} \equiv \begin{bmatrix} b_7 \\ b_8 \\ b_9 \\ b_{10} \end{bmatrix} = \begin{bmatrix} b_{(i-1)\cdot l+1} \\ b_{(i-1)\cdot l+2} \\ b_{(i-1)\cdot l+3} \\ b_{l\cdot M_T} \end{bmatrix} \tag{3.32}$$

The pruning criterion is compiled from two sets:

1. For determining the first set, the bits of the partial bit label $\mathbf{b}^{(i)}$ are compared with the current ML bit label $\mathbf{b_{ML}}$. All bits that are opposite to the ML bit label may have their metrics $\lambda_j^{\overline{\mathrm{ML}}}$ updated when the subtree is further traversed. Thus the first set to consider is

$$\mathcal{N}_1 = \left\{ \lambda_j^{\overline{\mathrm{ML}}} | b_j = b_j^{\overline{\mathrm{ML}}}, j > (i-1) \cdot l \right\}. \tag{3.33}$$

2. The second set is represented by the metrics of all the bits corresponding to the subtree of $\mathbf{s}^{(i)}$ which are not yet determined: b_j with $j = 1, 2 \ldots (i-1) \cdot l$.

$$\mathcal{N}_2 = \left\{ \lambda_j^{\overline{\mathrm{ML}}} | j \leq (i-1) \cdot l \right\}. \tag{3.34}$$

The complete set to consider is the union of the two sets:

$$\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2 = \left\{ \lambda_j^{\overline{\mathrm{ML}}} | b_j = b_j^{\overline{\mathrm{ML}}}, j > (i-1) \cdot l \right\} \cup \left\{ \lambda_j^{\overline{\mathrm{ML}}} | j \leq (i-1) \cdot l \right\}. \tag{3.35}$$

Note that the $\lambda_j^{\overline{\mathrm{ML}}}$ from the set $\mathcal{N}$ would only be updated if the actual PED $d_i = d(\mathbf{s}^{(i)})$ is not already larger than the largest metric in this set. In this case the further calculated metric $d(\mathbf{s})$ will never be smaller than one of the metrics from the set and won't update anything. This

leads to the pruning criterion for the STS algorithm:

$$d(\mathbf{s}^{(i)}) > \max_{\lambda_j^{\overline{\mathrm{ML}}} \in \mathcal{N}} (\lambda_j^{\overline{\mathrm{ML}}}). \tag{3.36}$$

Such a pruning criterion ensures that the list contains always the smallest metric as the ML metric and a valid counter hypothesis. The pruning criterion also ensures that the subtree doesn't lead to an update of the ML solution as its metric is always smaller than the metrics of the counter hypothesis $\lambda_j^{\overline{\mathrm{ML}}} < \lambda^{\mathrm{ML}}$.

Although the STS algorithm increases the efficiency compared to the RTS algorithm, the computational complexity is still large. In the described algorithm the LLR values are not bound and can take any value. Practical systems need to bound the LLR values. This can be implemented into the algorithm instead of clipping them to some maximum value L_{max} after detection. By clipping the LLR values already during detection, the computational complexity of the algorithm gets further reduced as tree pruning is more aggressive. In detail, the implementation of a maximum value is done by limiting the preliminary values of the counter hypothesis metrics according to [16] as:

$$\lambda_j^{\overline{\mathrm{ML}}} \leftarrow \min \left\{ \lambda_j^{\overline{\mathrm{ML}}}, \lambda^{\mathrm{ML}} + L_{max} \right\} \quad \forall j. \tag{3.37}$$

The pruning criterion is updated to

$$d(\mathbf{s}^{(i)}) > \min \left\{ \max_{\lambda_j^{\overline{\mathrm{ML}}} \in \mathcal{N}} (\lambda_j^{\overline{\mathrm{ML}}}), \lambda^{\mathrm{ML}} + L_{max} \right\}. \tag{3.38}$$

The LLR clipping does not provide optimal max-log approximation and therefore introduces some detection performance degradation. A trade-off between computation complexity and detection performance has to be found by choosing a suitable L_{max}. For $L_{max} \to \infty$ we approach the max-log solution, for $L_{max} \to 0$ we approach the performance of a hard-output sphere detector.

4

Simulation Results

In this chapter we investigate the performance of the soft SD and compare it to the previously used ML and ZF decoding techniques. Both, Alamouti and Golden Code STBCs in a 2x2 constellation will be considered in combination with various channel models. The SD implements the STS algorithm described in Section 3.3.2 and is configured to perform LLR clipping with a maximum LLR value of $L_{max} = 10$. The soft output of the SD is passed to a convolutional decoder which implements the BCJR algorithm. The results presented hereafter focus on the regime 1 due to its reliability and importance for safety-related applications. LS block-type estimation is used to estimate the channel coefficients.

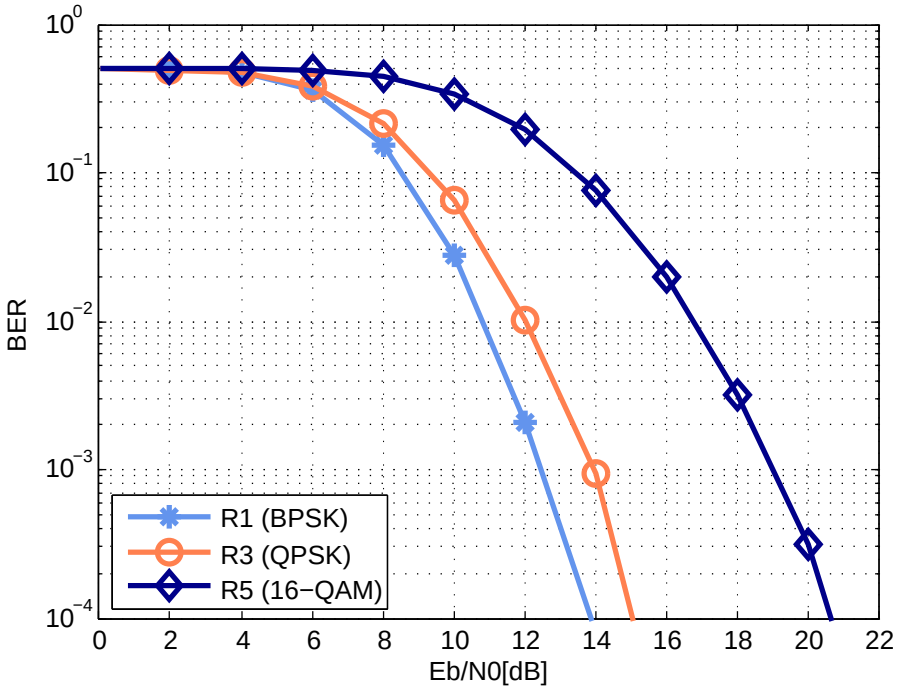


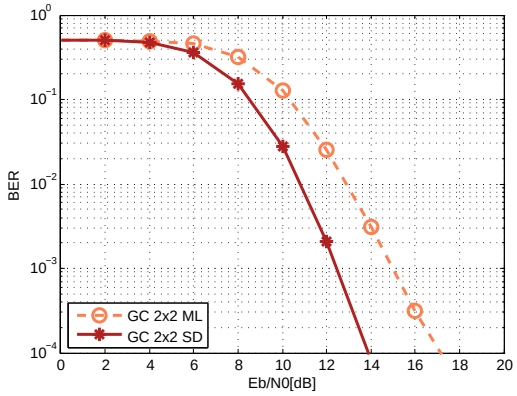
Figure 4.1: *Golden Code SD performance in regime 1, 3 and 5*

4.1 Block-Fading Channels

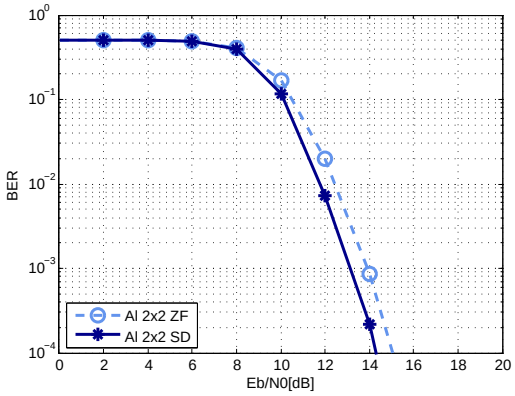
At first, we will investigate the performance of the SD in a block-fading channel, i.e. without time variance. In block-fading channels the channel impulse response coefficients are constant for the duration of one OFDM frame. Multipath propagation is taken into account and results in frequency selective fading, as fading dips occur due to local constructive and destructive interference. Although this channel model is restricted to static transmitter, receiver and IOs, it matches certain real channels very well.

Figure 4.1 shows the performance of the SD in transmission regimes 1, 3 and 5 with Golden Code STBC. The simulation is performed for 1000 frames with frame length of 500 Bytes. As regime 1 has the lowest modulation order (BPSK) which makes the transmission robust, it results in the lowest error rate. For higher order modulation schemes we obtain 1 dB and 6 dB performance loss for regimes 3 (QPSK) and 5 (16-QAM), respectively. This is because these modulation schemes have more densely spaced symbol constellations at the given Signal-to-Noise-Ratio (SNR) and therefore higher error probabilities. Although regime 3 has 1 dB performance loss, it delivers the double data rate compared to regime 1.

Comparison of hard detection and soft detection is presented in Figure 4.2. As expected, the use of soft information brings improvements of 3 dB over ML detection for the Golden Code and about 1 dB over ZF detection for the Alamouti Code at higher SNRs.



(a) Golden Code decoding with SD and ML



(b) Alamout Code decoding with SD and ZF

Figure 4.2: *Performance improvement with soft information for transmission regime 1, block-fading channel model*

4.2 Time-Variant Channels

In order to simulate more realistic channels, movement of transmitter, receiver and scatterers has to be considered. These movements result in time variety of channel states. Such channels are called time-variant channels.

4.2.1 Jakes Channel Model

The Jakes channel model is described in Section 2.1.2 and simulates a time-variant Rayleigh fading channel with NLOS. Simulation results show that the LS block-type estimator doesn't deliver accurate estimation of the channel coefficients since in contrast to block-fading channels they change for one OFDM frame over time. The LS block-type estimator provides one channel coefficient per subcarrier for the whole duration of the frame, but, as the channel changes over time, we have more deviation for longer transmission times. This means that a shorter frame, whose transmission lasts less time, will experience less channel state change and the estimated coefficients will match the real coefficients with small deviations. For longer transmission times the deviation of the estimated channel coefficients is large and results in an error floor that cannot be compensated with increased SNR. Figure 4.3 shows the performance of SD with Golden Code for different frame lengths in transmission regime 1. The SD improves the performance, but as this is the case with hard detection, it can not compensate for the loss of CSI. While the lengths of 100 and 150 Bytes result in acceptable performance, frames with a length of 200 Bytes suffer strong Bit Error Rate (BER) increase. If recommended length of 500 Bytes is chosen, the transmission is impossible and results in an error floor already at $2 \cdot 10^{-1}$. The Golden Code can be used with the LS estimator

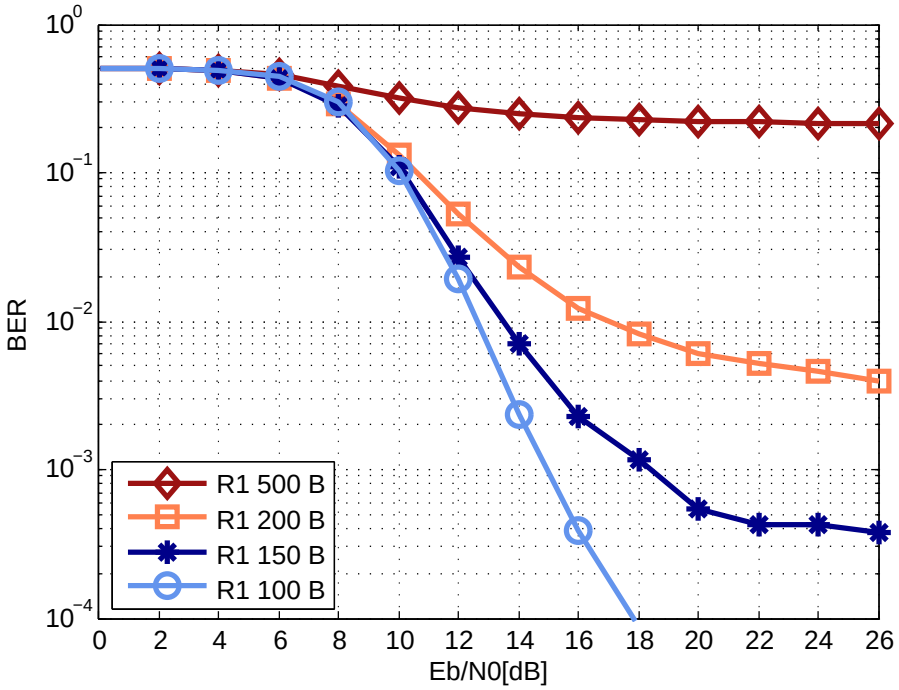
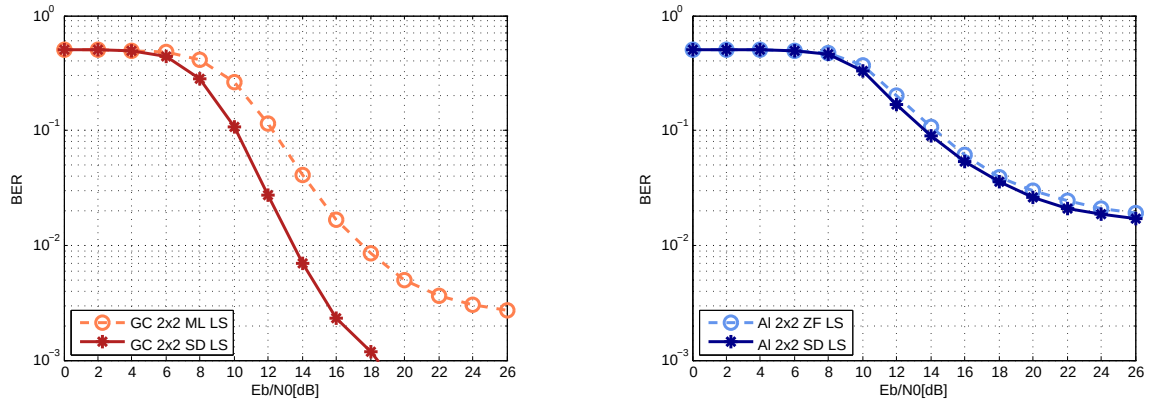


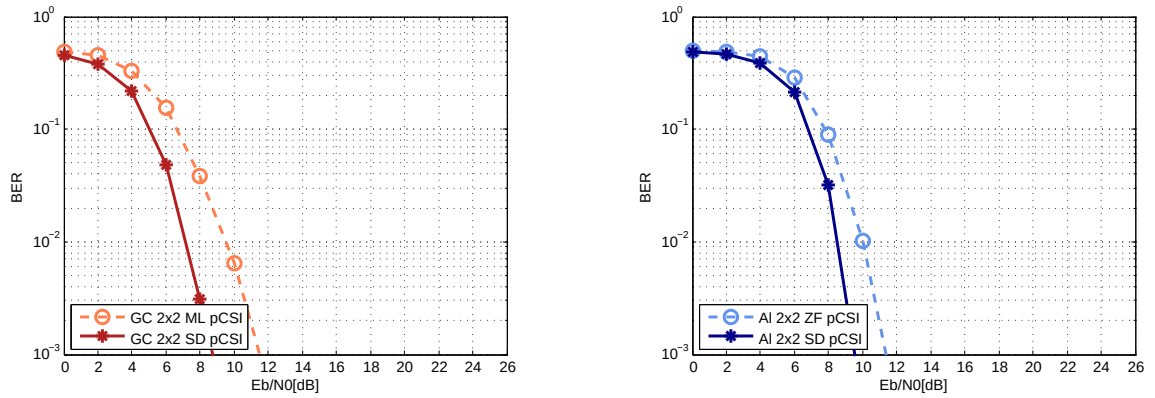
Figure 4.3: *Golden Code SD for different lengths of the OFDM frame, Jakes channel model, regime 1*

when frame length is limited to 150 Bytes, but with reduced throughput of the system. The impact of inaccurate CSI obtained from the LS estimator can be seen in Figures 4.4 and 4.5 for Alamouti and Golden Code STBCs. While for Golden Code OFDM frames with length of 150 Bytes result in reliable communication, the Alamouti code has poor performance at frame lengths of 150 Bytes and the SD can not compensate in this case. An error floor is obtained at $2 \cdot 10^{-2}$ and is unacceptable for practical implementations. This is due to the mapping of alphabet symbols to codeword matrices. The Alamouti Code generates always more codeword matrices than the Golden Code as it maps 2 alphabet symbols to one codeword matrix, while the Golden Code maps 4 symbols to one codeword matrix, see Table 2.1. Thereafter the length of the OFDM frame for Alamouti is always larger than the length of the OFDM frame for Golden Code and we have more deviation of estimated channel coefficients. More sophisticated channel estimation techniques are required to solve this issue. Figure 4.5 depicts the performance of the Golden Code and Alamouti STBCs in case that the channel coefficients are perfectly known at receiver side for regime 1. Although the perfect CSI is an ideal assumption, comparison to Figure 4.4 shows the improvement that could be reached by more accurate channel estimators. With perfect CSI, both STBCs deliver good performance and the SD improves the transmission by 2.5 dB. In transmission regime 3 (Figure 4.6) the



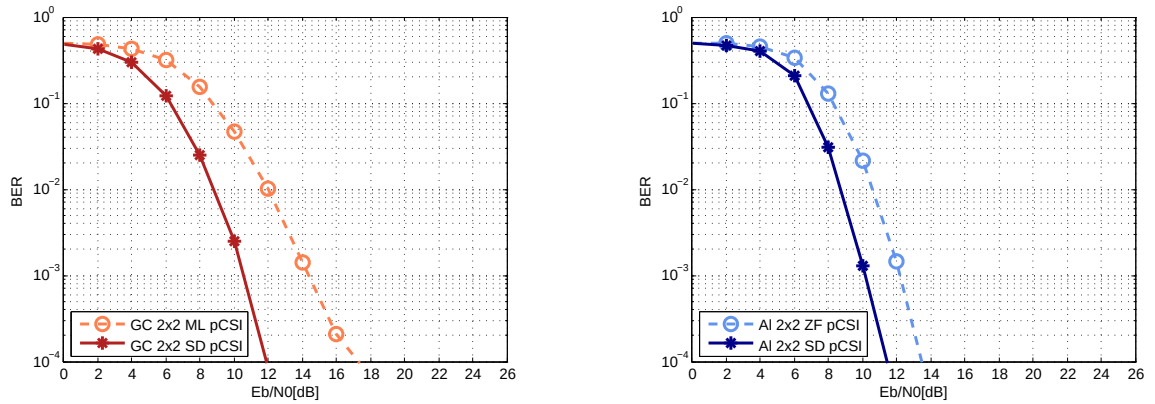
(a) Golden Code SD and ML (b) Alamout Code SD and ZF

Figure 4.4: *LS estimation with frame length 150 Bytes, regime 1, 1000 frames*



(a) Golden Code SD and ML (b) Alamout Code SD and ZF

Figure 4.5: *Perfect CSI with frame length 150 Bytes, regime 1, 1000 frames*



(a) Golden Code SD and ML (b) Alamout Code SD and ZF

Figure 4.6: *Perfect CSI with frame length 150 Bytes, regime 3, 1000 frames*

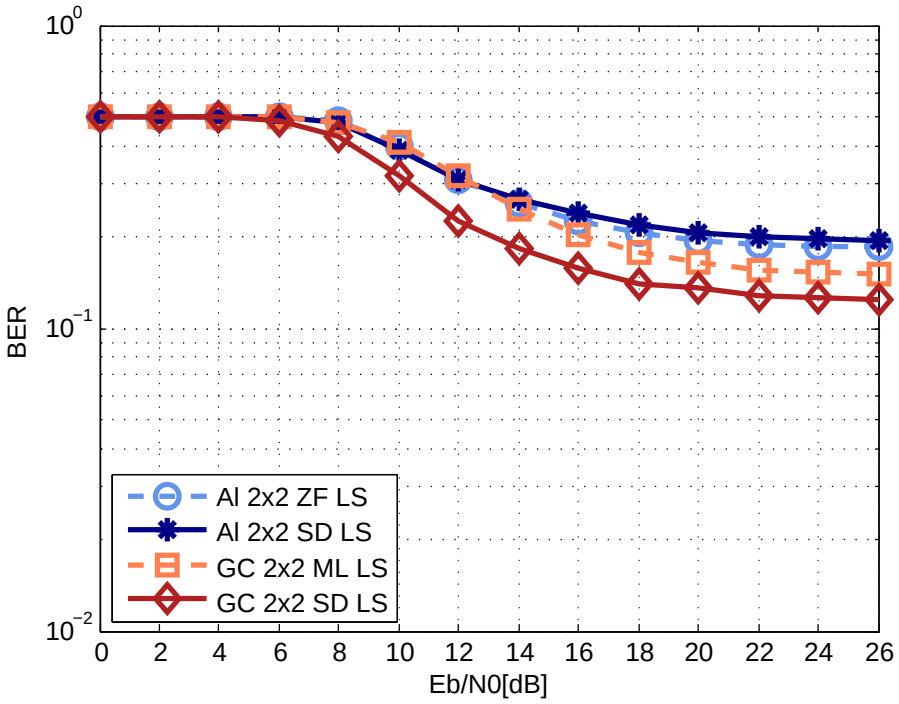


Figure 4.7: *LS estimation in GSCM, regime 1, 200 Bytes*

improvement by the SD for Golden Code is even better, namely 4.5 dB, while for Alamouti Code the improvement remains at 2.5 dB.

4.2.2 Geometry-Based Stochastic Channel Model

As the Jakes channel model does not represent channel conditions in vehicular environments another model has to be considered to evaluate the performance of the SD in such scenarios. The GSCM channel model mentioned in Section 2.1.2 provides a stochastic ray-based simulation of V2I communication in a four lane highway, two in each direction. In contrast to the Jakes channel model a Line-of-Sight (LOS) component is present and there are less time variations. The transmitter is set to move with a speed of 100 km/h, while the receiver is static. Similarly to the Jakes channel model the LS estimator delivers channel coefficients with large deviation and results in an error floor (Figure 4.7). Without improved estimation techniques, investigation of simulation results with the LS estimator does not lead to feasible results for this channel model. Therefore we will simulate perfectly known CSI at the receiver in order to compare the performance improvement with SD in this channel model. Figure 4.8 shows the performance of the SD in transmission regime 1, i.e. its improvement over ML and ZF decoding techniques in a GSCM channel with perfect knowledge of the channel coefficients for both STBCs. We can see a slight performance loss of the Golden Code in comparison to

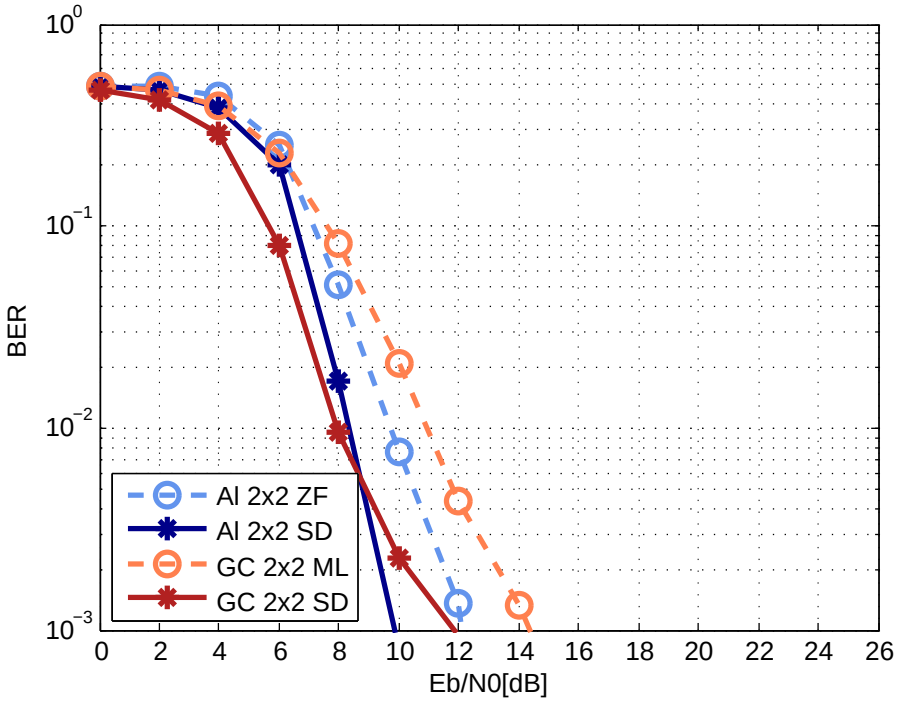


Figure 4.8: *Perfect CSI in GSCM, regime 1, 200 B*

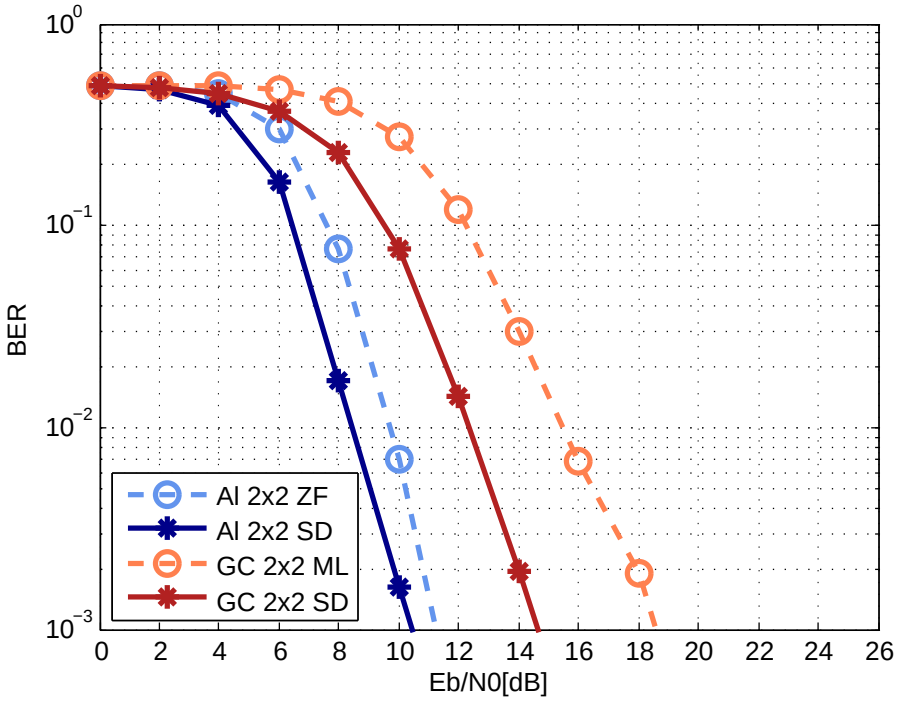


Figure 4.9: *Perfect CSI in GSCM, regime 3, 200 Bytes*

the Jakes channel model while the Alamouti Code does not show this effect. The decrease of the reliability gain of MIMO results from the strong LOS component and less time variance. The reliability and diversity improvement of Alamouti Code is even more emphasized in transmission regime 3 (Figure 4.9). Although the SD brings 4 dB improvement for the Golden Code, the Alamouti Code still performs better for vehicular environments.

5

Conclusions

Current research and standardization process for vehicular communication have reached an advanced stage. The developed WAVE protocol stack consists of the IEEE 802.11p physical and MAC layer standard and the IEEE 1609 family of standards, which covers the higher OSI reference model layers. This thesis investigates possibilities to improve performance of the physical layer defined in IEEE 802.11p. Performance evaluation is based upon results obtained from the simulation of the IEEE 802.11p physical layer in Matlab.

The MIMO extension introduces spatial diversity into the system. STBCs are designed to exploit that diversity. Evaluated STBCs are Alamouti and Golden Code in a 2x2 constellation. While previous work has shown that MIMO improves the transmission reliability and increases the throughput, this thesis considers the implementation of a soft SD for MIMO in combination with STBCs. Examined SD algorithms need to provide soft information to the BCJR decoder in order to improve decoding. While the SESD algorithm reduces the complexity compared to an exhaustive search over all lattice points (ML decoding), it does not provide soft information. The RTS algorithm provides soft information, but results in high computational complexity as it traverses the search tree multiple times, performing redundant operations. This can be avoided by the STS algorithm which traverses the tree only once. In order to do that, it administrates a list with preliminary results while traversing the tree. Further complexity reduction is obtained by more aggressive tree pruning with LLR clipping, i.e. limiting the LLRs to a certain maximum value. The LLR clipping makes the STS algorithm tunable, but lower complexity comes hand in hand with performance losses. Although the STS and LLR clipping reduce complexity enormously, still much computational power is required. Simulation results with the soft SD for the Alamouti and Golden Code show a system performance improvement that ranges between 2 dB and 4.5 dB. While

performance gain obtained with the soft SD for Golden Code is larger than for Alamouti, the latter performs better in vehicular environments and still obtains gains about 2 dB compared to hard detection. The channel model used for simulation of vehicular environments contains a strong LOS and less time variance than the classical time-variant Jakes channel model. The Alamouti Code is more robust to these conditions than the Golden Code, which performs better for NLOS propagation. The implemented LS channel estimator turns out to be insufficient for time-varying channels. Recommended frame lengths of 500 Bytes can not be transmitted with acceptable BERs. Such weak performance comes from the outdated channel coefficients obtained from the LS estimator which provides one channel coefficient per subcarrier for the whole OFDM frame. With increased length of the frame, the deviation of the estimated channel coefficients from the real coefficients increases. This is also the reason why the Golden Code performs better with LS estimator than the Alamouti Code for equal payload lengths, as the Golden Code results in shorter OFDM frames than Alamouti.

As expected, the soft SD enhances the overall system performance, but there still are areas which are in need of improvement:

- The LS estimation method is not optimal for vehicular environments. Rapidly changing channel conditions introduce strong dependence on the frame length. More sophisticated estimation techniques, such as pilot-based estimation, are required.

- The different STBCs have variable performance under varying conditions. Intuitive model extension would be adaptive modulation, STBC schemes and coding technique. In order to adapt the parameters, channel variations have to be tracked. Rate-Compatible Puncuted Convolutional Codes (RCPCs), for example, adjust the coding rate of a convolutional code and can be used with the same decoder. By this way it is possible to upgrade the code rate and increase error protection in highly corruptive channels, while in better conditioned channels the transmission rate can be increased.

List of Acronyms

AGC automatic gain control

APP A-Posteriori Probability

AWGN Additive White Gaussian Noise

BCJR Bahl-Cocke-Jelinek-Raviv

BER Bit Error Rate

BPSK Binary Phase Shift Keying

CDD cyclic delay diversity

CP cyclic prefix

CSI Channel State Information

DFT Discrete Fourier Transform

DI distance increment

DSP digital signal processor

ETSI European Telecommunications Standards Institute

FCC Federal Communications Commission

FEC Forward Error Correction

FDMA Frequency Division Multiple Access

FFT Fast Fourier Transformation

FTW Forschungszentrum Telekommunikation Wien

GSCM Geometry-Based Stochastic Channel Model

HT-LTF high throughput long training field

HT-SIG high throughput signal field

HT-STF high throughput short training field

ICI Inter Carrier Interference

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IFFT Inverse Fast Fourier Transformation

IO Interacting Object

IPv6 Internet Protocol version 6

ISI Inter Symbol Interference

ISO International Organization for Standardization

ITS Intelligent Transport Systems

ITU International Telecommunication Union

LLC Logical Link Control

LLR Log-Likelihood Ratio

LO local oszillator

LOS Line-of-Sight

LS Least Squares

MAC medium access control

MIB Management Information Base

MIMO Multiple-Input Multiple-Output

ML Maximum Likelihood

MLME MAC Sublayer Management Entity

NLOS Non-Line-of-Sight

OBU onboard unit

OFDM Orthogonal Frequency Division Multiplexing

OSI Open Systems Interconnection

PDP Power-Delay Profile

PED Partial Euclidean Distance

PHY Physical Layer

PLCP Physical Layer Convergence Protocol

PMD Physical Medium Dependent

POI Points of Interest

PSDU PLCP Service Data Unit

PSID Provider Service Identifier

PSV Partial Symbol Vector

PPDU PLCP Protocol Data Unit

QAM Quadrature Amplitude Modulation

QPSK Quadrature Phase Shift Keying

RCP resource command processor

RCPC Rate-Compatible Punctured Convolutional Code

RM Resource Manager

RMA resource manager application

RSU road side unit

RTS repeated tree search

SCH service channel

SD Sphere Decoder

SESD Schnorr-Euchner Sphere Decoder

SISO Single-Input Single-Output

SNR Signal-to-Noise-Ratio

STBC Space-Time Block Code

STS Single Tree Search

TCP Transmission Control Protocol

UDP User Datagram Protocol

U.S. United States

VANET Vehicular Ad Hoc Networks

V2I Vehicle-to-Infrastructure

V2V Vehicle-to-Vehicle

VSA Vendor Specific Action

WAVE Wireless Access in Vehicular Environments

WME WAVE Management Entity

WSMP WAVE Short Messages Protocol

ZF Zero-Forcing

Bibliography

[1] D. Niklos ”Implementation of space-time codes in IEEE 802.11p-based systems.”, Master's thesis, Institute of Telecommunications, Vienna University of Technology, 2012.

[2] V. Shivaldova, ”Implementation of IEEE 802.11p physical layer model in SIMULINK.”, Master's thesis, E389, Vienna University of Technology, 2010.

[3] IEEE Intelligent Transportation Systems Committee, “Trial-use standard for wireless access in vehicular environments (WAVE) - resource manager.” IEEE Std 1609.1-2006, 2006.

[4] IEEE Intelligent Transportation Systems Committee, ”IEEE standard for wireless access in vehicular environments (WAVE) - security services for applications and management messages.” IEEE Std 1609.2-2013, 2013.

[5] IEEE Intelligent Transportation Systems Committee, ”IEEE standard for wireless access in vehicular environments (WAVE) - networking services.” IEEE Std 1609.3-2010, 2010.

[6] IEEE Intelligent Transportation Systems Committee, ”IEEE standard for wireless access in vehicular environments (WAVE) - multi-channel operation.” IEEE Std 1609.4-2010, 2010.

[7] IEEE 802.11p, “Draft standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements: Wireless access in vehicular environments.” IEEE P802.11p/D9.0, September 2009.

[8] A.F. Molisch, 2nd ed., ”Wireless communications.”, 2011.

[9] F. Hlawatsch, ”Modulation and detection techniques.”, Lecture notes, March 2009.

[10] J. Karedal, F. Tufvesson, N. Czink, A. Paier, C. Dumard, T. Zemen, C. Mecklenbräuker, and A. Molisch, "A geometry-based stochastic MIMO model for vehicle-to-vehicle communications.", *IEEE Transactions on wireless communications*, vol. 8, pp. 3646-3657, July 2009.

[11] 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.

[12] G. Matz and C. Mecklenbräuker, "MIMO communications." Lecture Notes, 2011.

[13] H. Yao and G. Wornell, "Achieving the full MIMO diversity-multiplexing frontier with rotation-based space-time codes.", *Proceedings of the Annual Allerton Conference on Communication, Control and Computing*, vol. 41, pp. 400-409, The University of Illinois, 2003.

[14] J.-C. Belfiore, G. Rekaya, and E. Viterbo, "The Golden Code: a 2 x 2 full-rate space-time code with non-vanishing determinants.", *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1432-1436, April 2005.

[15] B. Hassibi, H. Vikalo, "On the sphere-decoding algorithm I. Expected complexity", *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 2806-2818, August 2005.

[16] G. Reise, "MIMO receivers using soft information.", Master's thesis, E389, Vienna University of Technology, 2007.

[17] 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, March 2003.

[18] C. Studer "Iterative MIMO decoding: Algorithms and VLSI implementation aspects", Hartung-Gorre Verlag Konstanz, 2009.

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

[20] E. Viterbo and E. Biglieri, "A universal decoding algorithm for lattice codes", *Colloque GRETSI*, vol. 14, pp. 611-614, September 1993.

[21] C. P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems", *Math. Programming: Series A and B*, vol. 66, no. 2, pp 181-191, September 1994.

[22] R. Wang and G. B. Giannakis, "Approaching MIMO channel capacity with soft detection based on hard sphere decoding", *IEEE Transactions on Communications*, vol. 54, no. 4, pp. 587-590, April 2006.

[23] C. Studer, M. Wnek, A. Burg and H. Bölscskei, "Soft-output sphere decoding: Performance and implementation aspects", *ACSSC '06 Fortieth Asilomar Conference on Signals, Systems and Computers*, 2006.