OULU 2011
C 380

ACTA UNIVERSITATIS OULUENSIS

C
TECHNICA

Markus Myllylä

DETECTION ALGORITHMS AND ARCHITECTURES FOR WIRELESS SPATIAL MULTIPLEXING IN MIMO-OFDM SYSTEMS

UNIVERSITY OF OULU,
FACULTY OF TECHNOLOGY,
DEPARTMENT OF ELECTRICAL AND INFORMATION ENGINEERING;
UNIVERSITY OF OULU,
CENTRE FOR WIRELESS COMMUNICATIONS;
INFOTECH OULU

ACTA UNIVERSITATIS OULUENSIS
C Technica 380

MARKUS MYLLYLÄ

DETECTION ALGORITHMS AND ARCHITECTURES FOR WIRELESS SPATIAL MULTIPLEXING IN MIMO-OFDM SYSTEMS

Academic dissertation to be presented, with the assent of the Faculty of Technology of the University of Oulu, for public defence in Auditorium IT115, Linnanmaa, on 27 May 2011, at 12 noon

UNIVERSITY OF OULU, OULU 2011

Supervised by
Professor Markku Juntti
Docent Joseph R. Cavallaro

Reviewed by
Professor Peter Nilsson
Professor Helmut Bölcskei

ISBN 978-951-42-9432-7 (Paperback)
ISBN 978-951-42-9433-4 (PDF)
http://herkules.oulu.fi/isbn9789514294334/
ISSN 0355-3213 (Printed)
ISSN 1796-2226 (Online)
http://herkules.oulu.fi/issn03553213/

Cover Design
Raimo Ahonen

JUVENES PRINT
TAMPERE 2011

Myllylä, Markus, Detection algorithms and architectures for wireless spatial multiplexing in MIMO-OFDM systems.
University of Oulu, Faculty of Technology, Department of Electrical and Information Engineering, P.O. Box 4500, FI-90014 University of Oulu, Finland; University of Oulu, Centre for Wireless Communications, Infotech Oulu, P.O. Box 4500, FI-90014 University of Oulu, Finland

Acta Univ. Oul. C 380, 2011
Oulu, Finland

Abstract

The development of wireless telecommunication systems has been rapid during the last two decades and the data rates as well as the quality of service (QoS) requirements are continuously growing. Multiple-input multiple-output (MIMO) techniques in combination with orthogonal frequency-division multiplexing (MIMO–OFDM) have been identified as a promising approach for high spectral efficiency wideband systems.

The optimal detection method for a coded MIMO–OFDM system with spatial multiplexing (SM) is the maximum a posteriori (MAP) detector, which is often too complex for systems with high order modulation. Suboptimal linear detectors, such as the linear minimum mean square error (LMMSE) criterion based detection, offer low complexity solutions, but have poor performance in correlated fading channels. A list sphere detector (LSD) is a tree search based soft output detector that can be used to approximate the MAP detector with a lower computational complexity. The benefits of the more advanced detectors can be realized especially in a low SNR environment by, e.g., increasing the cell coverage. In this thesis, we consider the linear minimum mean square error (LMMSE) criterion based detectors and more advanced LSDs for detection of SM transmission.

The LSD algorithms are not as such feasible for hardware implementation. Therefore, we identify the design choices that relate to the performance and implementation complexity of the LSD algorithms. We give guidelines to the LSD algorithm design and propose the proper trade-off solutions for practical wireless systems. The more stringent requirements call for further research on architectures and implementation. In particular, it is important to address the parallelism and pipelining factors in the architecture design to enable an optimal trade-off between used resources and operating speed. We design pipelined systolic array architecture for LMMSE detector algorithms and efficient architectures with given algorithm properties for the LSD algorithms.

We consider the VLSI implementation of the algorithms to study the true performance and complexity. The designed architectures are implemented on a field programmable gate array (FPGA) chip and CMOS application specific integrated circuit (ASIC) technology. Finally, we present some measurement results with a hardware testbed to verify the performance of the considered algorithms.

Keywords: architecture, ASIC, detector, FPGA, implementation, LSD, MIMO, OFDM, soft-output, tree search

Myllylä, Markus, Ilmaisualgoritmit ja -arkkitehtuurit langattomiin tilakanavoituihin MIMO-OFDM järjestelmiin.
Oulun yliopisto, Teknillinen tiedekunta, Sähkö- ja tietotekniikan osasto, PL 4500, 90014 Oulun yliopisto; Oulun yliopisto, Centre for Wireless Communications, Infotech Oulu, PL 4500, 90014 Oulun yliopisto

Acta Univ. Oul. C 380, 2011
Oulu

Tiivistelmä

Langattoman tietoliikenteen kehitys on ollut nopeaa viimeisien vuosikymmenien aikana ja järjestelmiltä vaaditaan yhä suurempia datanopeuksia ja luotettavuutta. Multiple-input multiple-output (MIMO) tekniikka yhdistettynä monikantoaaltomodulointiin (MIMO-OFDM) on tunnistettu lupaavaksi järjestelmäksi, joka mahdollistaa tehokkaan taajuusalueen hyödyntämisen.

Optimaalinen ilmaisumenetelmä tilakanavoituun (SM) ja koodattuun MIMO-OFDM järjestelmään on maximum *a posteriori* (MAP) ilmaisin, joka on tyypillisesti liian kompleksinen toteuttaa laajakaistajärjestelmissä, joissa käytetään korkean asteen modulointia. Alioptimaaliset lineaariset ilmaisimet, kuten pienimpään keskineliövirheeseen (LMMSE) perustuvat ilmaisimet, ovat suhteellisen yksinkertaisia toteuttaa nykyteknologialla, mutta niiden suorituskyky on varsin heikko korreloivassa radiokanavassa. Listapalloilmaisin (LSD) on puuhakualgoritmiin perustuva pehmeän ulostulon ilmaisin, joka pystyy jäljittelemään MAP ilmaisinta sitä pienemmällä kompleksisuudella. Kehittyneemmät ilmaisimet, kuten LSD, voivat parantaa langattoman verkon suorituskykyä erityisesti ympäristössä, jossa on matala signaalikohinasuhde, esimerkiksi mahdollistamalla suuremman toiminta-alueen. Tässä väitöskirjassa on tutkittu kahta LMMSE ilmaisinta ja kolmea LSD ilmaisinta SM lähetyksen ilmaisuun.

Yleisesti LSD algoritmit eivät ole sellaisenaan toteutuskelpoisia kaupallisiin järjestelmiin. Väitöskirjassa on tämän vuoksi tutkittu LSD:n toteutukseen liittyviä haasteita ja toteutusmenetelmiä ja annetaan suosituksia LSD algorithmien suunnitteluun sekä ehdotetaan sopivia toteutuskompromisseja käytännön langattomiin järjestelmiin. Haastavammat suorituskyky- ja latenssivaatimukset edellyttävät lisätutkimuksia toteutusarkkitehtuureihin ja toteutuksiin. Erityisesti rinnakkaisten resurssien käyttö ja liukuhihnatekniikka toteutusarkkitehtuureissa mahdollistavat optimaalisen kompromissin löytämisen toteutuksessa käytettyjen resurssien ja laskentanopeuden väliltä. Väitöskirjassa suunnitellaan tehokkaat arkkitehtuurit tutkituille LMMSE ja LSD algoritmeille ottaen huomioon niiden ominaisuudet.

Väitöskirjassa tutkitaan algoritmien toteutusta VLSI tekniikalla ja pyritään saamaan realistinen arvio algoritmien kompleksisuudesta ja suorituskyvystä. Algoritmeille suunnitellut arkkitehtuurit on toteutettu sekä FPGA piirille että erillisenä toteutuksena ASIC teknologialla. Väitöskirjassa esitetään myös testilaitteistolla tehtyjä mittaustuloksia ja varmistetaan toteutettujen algoritmien suorituskyky.

Asiasanat: arkkitehtuuri, ASIC, FPGA, ilmaisin, LSD, MIMO, OFDM, pehmeä ulostulo, puuhaku, toteutus

To my parents

Preface

The research for this thesis has been carried out at the Centre for Wireless Communications (CWC), University of Oulu, Finland. I want to thank Professor Matti Latva-aho, Dr. Ian Opperman, and Lic. Tech. Ari Pouttu, the directors of CWC during my stay, for giving me the opportunity to work in such an energetic and inspiring working environment.

I am grateful to my supervisors Professor Markku Juntti and Professor Joseph R. Cavallaro for their invaluable guidance, support, fruitful discussions, and encouragement during my postgraduate research. I would also like to thank Professor Behnaam Aazhang for providing me the possibility to work as a visiting researcher in Rice University, Houston, USA during the years 2007-2008. I would like to express my gratitude to the reviewers of this thesis, Professor Helmut Bölcskei from the ETH, Zürich, Switzerland and Professor Peter Nilsson from Lund University, Lund, Sweden. Their comments significantly improved the quality of the thesis. I am indebted to my line manager from my current employer Renesas Mobile Europe, Kari Rysberg, who has provided me with flexibility and understanding in finalizing this thesis. Dr. Deborah Kaska is acknowledged for proofreading the manuscript. The main part of the work presented in this thesis was carried out in the MIMO techniques for 3G system and standard evolution (MITSE) project and I would like to thank the project managers, Dr. Nenad Veselinovic and Lic. Tech. Visa Tapio, the technical steering group members, as well as my colleagues in the project. In particular, I would like to thank my office mates, regular lunch and ski-trip companions, and traveling companions, Dr. Giuseppe Abreu, Dr. Mehdi Bennis, Dr. Marian Codreanu, Giuseppe Destino, Dr. Kaveh Ghaboosi, Dr. Kari Hooli, Jarkko Huusko, Antti Hyvärinen, Janne Janhunen, Juha Karjalainen, Lic. Tech. Kai Kiiskilä, Dr. Esa Kunnari, Davide Macagnano, Carlos Morais de Lima, Dr. Honglei Miao, Pedro Nardelli, Jarmo Niskanen, Matti Raustia, Stefano Severi, Hanna Saarela, Dr. Harri Saarnisaari, Lic. Tech. Pirkka Silvola, Dr. Attaphongse Taparugssanagorn, Dr. Antti Tölli, Dr. Mikko Vehkaperä, and Dr. Jari Ylioinas for the refreshing moments and fruitful discussions at the office and out there. The administrative support of Antero Kangas, Elina Komminaho, Mari Lehmikangas, Sari Luukkonen, Kirsi Ojutkangas, Hanna Saarela, Jari Sillanpää, Tero Suutari, and Timo Äikas is highly appreciated. Special thanks go to my co-authors Juho Antikainen, Aaron Byman, Juha-

Matti Hintikka, Johanna Ketonen, Matti Limingoja, and as well as Markku Jokinen for the fruitful co-operation.

The research for this thesis has been financially supported by Infotech Oulu Graduate School during the year 2009. Funding through the projects was provided by the Finnish Funding Agency for Technology and Innovation, Elektrobit, Nokia, Nokia Siemens Networks, Texas Instruments, Uninord, and the Academy of Finland, which is gratefully acknowledged. I was privileged to receive personal grants for doctoral studies from the following Finnish foundations: Tauno Tönningin säätiö, Jenny ja Antti Wihurin rahasto, Nokia Oyj:n säätiö. These acknowledgements encouraged me to go on with my research work and they are gratefully recognized.

During the years the research for this thesis was done I was fortunate to have many good friends who are too numerous to be acknowledged individually. The activities, sauna evenings, parties, and numerous other shared moments with them were a lot of fun and helped me forget the research problems when needed. The floorball team, OYUS, and the rinkball team, Karhut, enabled me to stay in form and, thus, helped me stay focused on research. My deepest gratitude goes to my parents Vilho and Raili, for their love, support, and encouragement towards education. The help you have given during my life is invaluable. I wish to thank my brother Mikko for lifelong friendship and help.

Oulu, February 5, 2011 Markus Myllylä

Symbols and abbreviations

$\mathbf{b}$	coded binary bit vector
B_{C}	number of nodes in a complex valued search tree with given configuration
B_{R}	number of nodes in a real valued search tree with given configuration
C_0	squared sphere radius
C_{mem}	memory sphere radius
$\mathbb{C}^{m \times n}$	set of complex $m \times n$ matrices
D	node coefficient used in scheduled search method
$\mathbf{D}_{\mathrm{R}}$	diagonal matrix generated by the SGR algorithm
$\mathscr{D}^2_{\mathrm{LMMSE}}$	squared mean square error metric
Δ_{it}	latency of a LSD algorithm implementation iteration
Δ_{tot}	throughput time of a LSD algorithm implementation
E_{S}	transmitted symbol power
f_c	carrier frequency
f_{D}	Doppler frequency
γ	signal-to-noise ratio
$\mathbf{H}$	channel matrix in frequency domain, $N_{\mathrm{R}} \times N_{\mathrm{T}}$
$\tilde{\mathbf{H}}$	residual channel matrix in frequency domain, $N_{\mathrm{R}} \times N_{\mathrm{T}}$
$\mathbf{H}_{\mathrm{r}}$	real equivalent channel matrix in frequency domain, $M_{\mathrm{R}} \times M_{\mathrm{T}}$
i_{max}	number of applied CORDIC micro rotations
$\mathrm{Im}\{\cdot\}$	imaginary part
K	size of the candidate list with K-best-LSD
κ	precomputed normalization constant in CORDIC micro rotations
L_{A1}	*a posteriori* information in the input of the detector
L_{A2}	*a posteriori* information in the input of the decoder
$L_{\mathrm{D}}(b_k)$	*a posteriori* information of the kth bit
L_{D1}	*a posteriori* information in the output of the detector
L_{D2}	*a posteriori* information in the output of the decoder
L_{E1}	extrinsic information given by the detector
L_{E2}	extrinsic information given by the decoder

L_{avg}	number of node visit resources available in average per subcarrier in scheduled search method
$L_{\text{avg}}^{\text{it}}$	average number of executed LSD algorithm iterations
L_{limit}	threshold value for LLR clipping
L_{node}	maximum search limit in terms of visited nodes with limited search method
$L_{\text{node}}(n)$	maximum search limit in terms of visited nodes for nth subcarrier with scheduled search method
L_{max}	maximum LLR clipping value
L_{min}	minimum number of node visit resources reserved for each subcarrier in scheduled search method
$\mathscr{L}$	LSD candidate list
$\mathscr{L}_i$	K-best-LSD algorithm architecture partial candidate list at layer i
$\mathscr{L}_{k,+1}$	set of candidate vectors having $b_k = +1$
λ	wavelength of the used transmission frequency
M_{R}	dimension for real equivalent system model, $M_{\text{R}} = 2N_{\text{R}}$
M_{T}	dimension for real equivalent system model, $M_{\text{T}} = 2N_{\text{T}}$
n_1	information of extended nodes in LSD algorithm
n_2	information of extended nodes in LSD algorithm
N_{c}	number of OFDM subcarriers
N_{cand}	LSD candidate list size
$N_{\text{cand}}^{\text{max}}$	maximum possible LSD candidate list size for a particular configuration
N_{R}	number of receive antennas
N_{T}	number of transmit antennas
N_{used}	number of OFDM subcarriers used for data transmission
$\mathbf{n}$	noise vector, $N_{\text{R}} \times 1$
$\mathbf{n}_{\text{r}}$	real equivalent noise vector, $M_{\text{R}} \times 1$
$\mathscr{N}$	partial candidate information; subscripts c and f used to denote child and father candidates
Ω	complex QAM constellation
Ω_{r}	real part of QAM constellation
P	power consumption
$\mathbf{p}$	permutation order vector
Q	number of bits in QAM symbol
$\mathbf{Q}$	unitary matrix

$\mathbf{Q}_\mathrm{A}$	matrix consisting of orthogonalized columns of the decomposed matrix by the SGR algorithm
R	coded transmission rate
R_det	implementation detection rate
$R_\mathrm{det}^{(\mathrm{asic})}$	ASIC implementation detection rate
$R_\mathrm{det}^{(\mathrm{fpga})}$	FPGA implementation detection rate
$\mathbf{R}$	upper triangular matrix
$\mathbf{R}_{xx}$	covariance matrix of transmitted symbol vector
$\mathbf{R}_{nn}$	covariance matrix of noise vector
$\tilde{\mathbf{R}}_{nn}$	covariance matrix of residual channel matrix
$\mathbf{R}_\mathrm{RX}$	spatial correlation matrix at the receiver
$\mathbf{R}_\mathrm{TX}$	spatial correlation matrix at the transmitter
$\mathrm{Re}\{\cdot\}$	real part
$\mathbb{R}^{m\times n}$	set of real $m \times n$ matrices
$\mathbf{s}$	partial candidate symbol vector; subscripts c and f used to denote child and father candidates
$\mathscr{S}$	memory set used in the LSD algorithms
σ^2	noise power
$\chi_{k,+1}$	set of possible bit vectors having $b_k = +1$
τ_mac	delay of a processing element in a multiplication array
τ_inv	delay of a processing element in an inversion array
τ_stage	delay of a pipeline stage of K-best-LSD algorithm architecture
T_c	channel coherence time
$\mathbf{U}$	upper triangular matrix generated by the SGR algorithm
V_C	number of visited nodes by the K-best-LSD algorithm in complex valued signal model
V_R	number of visited nodes by the K-best-LSD algorithm in real valued signal model
W_alg	work factor value of algorithm implementation
W_LLR	work factor value of LLR calculation unit implementation
W_R	memory sphere radius scaling variable
$\mathbf{W}$	LMMSE coefficient matrix, $N_\mathrm{R} \times N_\mathrm{T}$
$\mathbf{x}$	transmitted signal vector, $N_\mathrm{T} \times 1$
$\mathbf{x}_\mathrm{r}$	real equivalent transmitted signal vector, $M_\mathrm{T} \times 1$
$\mathbf{x}_i^{M_\mathrm{T}}$	last $M_\mathrm{T} - i + 1$ components of the vector $\mathbf{x}$

$\mathbf{y}$	received signal vector, $N_{\mathrm{R}} \times 1$
$\tilde{\mathbf{y}}$	received signal vector multiplied with orthogonal matrix, i.e., $\tilde{\mathbf{y}} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$, $M_{\mathrm{R}} \times 1$
$\mathbf{y}_{\mathrm{r}}$	real equivalent received signal vector, $M_{\mathrm{R}} \times 1$
$\mathbb{Z}$	set of integer numbers
$\lfloor \cdot \rceil$	rounding to the closest integer
$\lceil \cdot \rceil$	ceil operation, i.e., rounding to the closest higher integer
$\{\cdot\}^k$	kth element in the set
$(\cdot)^*$	complex conjugate of the argument
$(\cdot)^{\mathrm{T}}$	transpose of the argument
$(\cdot)^{\mathrm{H}}$	complex conjugate transpose (Hermitian) of the argument
$\lvert \cdot \rvert$	absolute of the argument
$\lVert \cdot \rVert_1$	1-norm
$\lVert \cdot \rVert_2$	Euclidean norm, i.e., 2-norm
$\lVert \cdot \rVert_{\mathrm{F}}$	Frobenius norm of the matrix
$\hat{(\cdot)}$	estimate of the argument
$\tilde{(\cdot)}$	soft output of the argument
(W,I,S)	fixed-point word length presentation, where W,I and S refer to the total number of bits, the number of bits used for the integer part, and the unsigned or signed type of value representation
x_i	ith element of the vector $\mathbf{x}$
$\mathbf{x_i}$	ith column of the matrix $\mathbf{X}$
$X_{i,j}$	(i,j)th element of the matrix $\mathbf{X}$
$\arctan(\cdot)$	inverse trigonometric function of tangent
$\cos(\cdot)$	cosine of the angle
$d(\mathbf{x})$	squared (partial) Euclidean distance of vector $\mathbf{x}$
$\mathrm{diag}(\mathbf{x})$	diagonal matrix with the elements of vector $\mathbf{x}$ on the main diagonal
$\exp(\cdot)$	exponential function, i.e., $e^{(\cdot)}$
$\mathrm{E}(\cdot)$	expectation of the argument
$f_c(\cdot)$	correlation function in Jacobian algorithm
$\ln(\cdot)$	natural logarithm of the argument
$O(\cdot)$	complexity order
$P(\cdot)$	probability of the argument to occur
$p(\cdot)$	probability density function

$\lvert\mathbf{X}\rvert$	determinant of the matrix **X**
$\mathbf{X}^{-1}$	inverse of the matrix **X**
$\mathbf{I}$	identity matrix; a subscript can be used to indicate the dimension
$\max(\cdot)$	maximum
$\min(\cdot)$	minimum
$\mathrm{sgn}(\cdot)$	sign of the argument
$\sin(\cdot)$	sine of the angle
$S_i(\mathbf{x})$	SEE middle point of the vector **x**
$\tan(\cdot)$	tangent of the angle
$\mathrm{tr}(\mathbf{X})$	trace of matrix **X**
$\approx$	approximative equal
2G	second generation cellular systems
3G	third generation cellular systems
3GPP	Third Generation Partnership Project
4G	fourth generation cellular systems
AD	absolute distance
ADD	addition
APP	*a posteriori* probability
ASIC	application specific integrated circuit
AWGN	additive white Gaussian noise
BER	bit error rate
B3G	beyond 3G
BLAST	Bell Labs space-time architecture
BRAM	block random access memory
BB	baseband
BS	base station
BW	bandwidth
CDF	cumulative distribution function
CDMA	code division multiple-access
CMOS	complementary metal-oxide semiconductor
CNTR	control unit
COMP	comparison
CORDIC	coordinate rotation digital computer
CORR	highly correlated

CP	cyclic prefix
CSI	channel state information
CWC	centre for wireless communications
CU	complexity unit
dB	decibel
D-BLAST	diagonal Bell Labs space-time architecture
DI	decoder iteration
DIV	division
DL	downlink
DMI	direct matrix inversion
EB4G	Elektrobit 4G hardware test platform
ED	Euclidean distance
EDGE	enhanced data rates for GSM evolution
FDD	frequency division duplex
FDMA	frequency division multiple access
FEC	forward error control
FER	frame error rate
FFT	fast Fourier transform
FPGA	field programmable gate array
GE	gate equivalents
GI	global iteration
GPRS	Generalized Packet Radio Service
HDL	hardware description language
HE	horizontal coding
HLS	high level synthesis
HW	hardware
HSPA	high-speed packet access
IFFT	inverse fast Fourier transform
IMT-A	international mobile communications - advanced
IP	internet protocol
IR	increasing radius
ISI	inter-symbol interference
L1	1-norm
L2	2-norm
LLR	log-likelihood ratio

LMMSE	linear minimum mean square error
LORD	layered orthogonal lattice detector
LR	lattice reduction
LS	limited search
LSD	list sphere detector
LST	layered space-time
LTE	long term evolution
LTE-A	long term evolution advanced
LUT	look-up-table
MAC	multiply and accumulate
MAP	maximum *a posteriori*
ML	maximum likelihood
MIMO	multiple-input multiple-output
MISO	multiple-input single-output
MITSE	MIMO techniques for 3G system and standard evolution
MMSE	minimum mean square error
MSE	mean square error
MT	mobile terminal
MUL	multiplication
NLOS	non-line-of-sight
OFDM	orthogonal frequency division multiplexing
OFDMA	orthogonal frequency division multiple access
OSIC	ordered successive interference cancellation
PAD	partial absolute distance
PAPR	peak-to-average power ratio
PDF	probability density function
PE	processing element
PED	partial Euclidean distance
QAM	quadrature-amplitude modulation
QoS	quality of service
QPSK	quadrature phase-shift keying
QRD	decomposition of the matrix into an orthogonal and a triangular matrix
RACE	Radio Access Emulator platform
RAM	random access memory
RF	radio frequency

RTL	register transfer level
RX	receiver
SC-FDMA	single-carrier frequency division multiple access
SD	sphere detector
SDMA	space division multiple access
SDR	semidefinite relaxation
SEE	Schnorr-Euchner enumeration
SGR	Squared Givens rotation
SIC	successive interference cancellation
SINR	signal-to-interference-plus-noise ratio
SIPO	serial-input parallel-output
SISO	single-input single-output
SfISfO	soft-input soft-output
SM	spatial multiplexing
SNR	signal-to-noise ratio
SQRD	sorted QR decomposition
SS	scheduled search
STBC	space-time block codes
STC	space-time codes
STTC	space-time trellis codes
SVD	singular value decomposition
TCM	trellis coded modulation
TDD	time division duplex
TDMA	time division multiple access
TPU	tree pruning unit
TX	transmitter
TU	typical urban
UL	uplink
UNC	uncorrelated
VB	Viterbo-Boutros
V-BLAST	Vertical Bell Labs space-time architecture
VE	vertical encoding
VLSI	very large scale integration
WiMAX	worldwide interoperability for microwave access
WCDMA	wideband code division multiple access

WLAN	wireless local-area network
WMAN	wireless metropolitan-area network
ZF	zero forcing

Contents

1 Introduction

The development of wireless communication systems has been rapid during the last two decades and the data rates as well as the quality of service (QoS) requirements are continuously growing to enable a rich user experience in wireless communication services. This will require high capacity and flexibility from future wireless communication systems and networks given that regulation and other factors render the radio frequency (RF) spectrum a scarce and valuable resource. Therefore, the physical layer of future wireless communication networks must be capable of providing an ever-increasing capacity in terms of high spectral efficiency, higher data rates and larger numbers of simultaneous users. Advanced technologies such as the use of multiple antennas both at the transmitter and receiver enable very efficient utilization of the spectrum. The use of multiple antennas both in the transmitter and receiver results in a so-called multiple-input-multiple-output (MIMO) radio channel [1, 2] as opposed to the conventional single-input-single-output (SISO) radio channel. MIMO in combination with orthogonal frequency-division multiplexing (OFDM) (MIMO–OFDM) have been identified as a promising approach for high spectral efficiency wideband systems [3]. The increasing data rates and higher capacity requirements call for improved receiver implementations and architectural design. The more stringent performance requirements call for further research on architectures and implementation of baseband receiver algorithms as well. The solution for the implementation is a trade-off between hardware complexity and operational performance.

In this thesis, different detector algorithms required in the reception of MIMO–OFDM downlink (DL) signals are considered. In addition to algorithm research and development, the architecture design of the algorithms, and implementation aspects and trade-offs are addressed in order to develop efficient implementations in terms of performance and complexity. Special focus is put on the list sphere detector (LSD) algorithms [4], which have recently emerged as a very promising approach to solving the detection problem.

1.1 Wireless system development

In order to satisfy the need for higher data rates and more efficient wireless systems, the dominating second generation (2G) cellular system Global System for Mobile Communications (GSM) [5], which was still used mainly for voice communication, has been upgraded with advanced features such as Generalized Packet Radio Service (GPRS) [6, 7] and Enhanced Data Rates for GSM Evolution (EDGE) [7]. The 2G systems divided users in separate time, frequency or code domains in the available radio spectrum by time-, frequency-, or code division multiple access (TDMA, FDMA, CDMA) methods, respectively [8, 9]. The third generation partnership project (3GPP) [10] introduced the 3G cellular systems [11] and their improved versions with High Speed Packet Access (HSPA) [12], which provide better support for multimedia and internet based services with higher data rates compared to 2G systems. The 3G systems are currently in active commercial usage and the most common Universal Mobile Telecommunications System (UMTS) technology uses wideband CDMA (WCDMA) [11] as the underlying air interface. At the same time, other wireless networks with less support for mobility, such as OFDM [13–15] based wireless local area network (WLAN) or Wi-Fi [16] and wireless metropolitan area network (WMAN) [17] systems, are undergoing rapid development.

Research and development that extends beyond 3G (B3G) cellular communication systems is also ongoing. The 3GPP is currently working on extensions to 3G standards called 3G Long Term Evolution (LTE) [18], which is based totally on packet data and an internet protocol (IP) network infrastructure. The 3G LTE standard includes advanced wireless technologies such as MIMO communications with multiple antennas at both the base station (BS) and mobile terminal (MT) and OFDM. The LTE specification targets peak data rates up to 100 and 50 Mb/s in DL and uplink (UL), respectively. Other evolving B3G systems also exist, such as the WMAN based Worldwide Interoperability for Microwave Access (WiMAX) system standard [19]. The main physical layer distinction in the 3G LTE systems compared to the 3G systems is the introduction of orthogonal frequency division multiplexing (OFDM) and orthogonal frequency division multiple access (OFDMA) [20] for DL and single-carrier frequency division multiple access (SC-FDMA) [21] for UL instead of the WCDMA based solutions in 3G systems.

Research on the 4th generation (4G) cellular system after B3G systems has also started and is currently called International Mobile Telecommunications-Advanced (IMT-A). The 3GPP will propose the evolution of the 3G LTE system framework as

LTE advanced (LTE-A) and it is included in the 3GPP Releases 10 and beyond. The targeted data rates are 1 Gb/s in the local area and 100Mb/s with outdoor area coverage.

1.2 MIMO–OFDM communications

The high data rate requirements of future wireless communication can be achieved by increasing spectral efficiency since the available RF spectrum bandwidth (BW) is limited in practical systems. The use of multiple antennas at both the transmitter and receiver, which is usually referred to as MIMO communication, offers improved capacity and significant potential for improved reliability compared to single antenna systems without additional transmission time or BW [2, 22–25]. The multiple antenna transmission is also challenging for the analog front-end and RF parts, and calls for methods to compensate for the nonidealities, e.g., see [26, 27]. A MIMO system of N_T transmit and N_R receive antennas for a total of $N_\mathrm{T}N_\mathrm{R}$ links can be applied to performance improvements compared to single antenna systems [2]. The performance improvements achieved by the use of MIMO communication include array gain, diversity gain, interference reduction and spatial multiplexing (SM) gain [2]. Several practical concepts, which are briefly introduced in this section, have been developed to realize the potential of MIMO communication.

Traditionally, multiple antennas have been used in wireless transmission in order to attain array gain or diversity gain. The array gain is achieved by processing at the transmitter or the receiver called beamforming, and results in an increase in average received signal-to-noise ratio (SNR) via coherent combining [2, 28], i.e., the operating range of the communication system can be extended. Beamforming uses multiple correlated antenna elements to focus the energy in the desired directions. Multiple antennas can also be used to reduce co-channel interference, which occurs due to use of the same frequency band in neighboring cells in a wireless system [22, 28]. This is carried out by adjusting the beam pattern so that there are nulls in the directions of the interfering co-channel users and high directivity towards the desired user. Interference reduction allows frequency reuse and thus an increase in multi-cell capacity. For more information about adaptive antennas, see [29, 30].

The diversity gain can be used to mitigate fading in a wireless fading channel by transmitting the same signal multiple times over independently fading paths in time/frequency/space and combining the signal at the receiver. Thus, the diversity gain increases the link reliability and QoS in a wireless communication system. Spatial diversity can be gained if there are multiple spatially separated transmit and/or receive antennas that are used to transmit and/or receive multiple redundant signals. Space-time codes (STC) can be designed to jointly correlate transmitted symbols in spatial and temporal domains in order to improve the reliability of the transmission and increase the data rate, when the channel state information (CSI) is only known at the receiver [25, 31–33]. Space-time trellis codes (STTCs) [33], which may be interpreted as a generalization of trellis coded modulation (TCM) [34] to multiple transmit antennas, achieve both diversity and coding gain with multiple antennas, but require rather high decoding complexity. A simple transmit diversity technique [35] was proposed by Alamouti for two transmit antennas, which achieves full diversity gain, but requires only simple linear processing for decoding. A generalization of the technique, space-time block codes (SBTCs), was introduced to an arbitrary number of antennas in [36]. The STBCs achieve full diversity with regard to the number of transmit and receive antennas, but do not offer any additional coding gain.

Spatial multiplexing, where the transmitted information bit sequence is split into multiple parallel streams that are transmitted simultaneously over the different transmit antennas at the same frequency band, is a simple method to gain a linear increase in the spectral efficiency in a MIMO channel. The capacity increase is linearly proportional to the minimum number of transmit and receive antennas in a rich scattering environment [1, 37]. Forward error control (FEC) coding, also called channel coding, is typically applied with the schemes to guarantee a certain error performance and several different encoding options such as vertical encoding (VE) and horizontal encoding (HE) can be applied to spatial multiplexing schemes [2]. Layered space-time (LST) architectures combined with channel coding offer pragmatic and powerful methods to increase the data rate, i.e., achieve spatial multiplexing gain, in systems with multiple antennas at both transmitter and receiver [38–41]. Very high spectral efficiency can be achieved in Bell Labs' layered space-time (BLAST) systems by employing multiple antennas at both transmit and receive sides. The original proposal by Foschini [38] is known as diagonal BLAST (D-BLAST). A simplified version of BLAST known as vertical BLAST (V-BLAST) was introduced in [39]. Since the individual spatial layers are superimposed during the transmission, they have to be separated at the receiver by a detector. The separation of the layers at the receiver is based on differences in the spatial signatures in the MIMO channel and MIMO systems exploit multipath propagation to separate the multiplexed data streams instead of mitigating them as in conventional antenna array systems. Different detection algorithms are discussed in more detail in the following section.

MIMO communications techniques typically assume a frequency-flat fading radio channel environment, i.e., the coherence bandwidth of the channel is larger than the bandwidth of the signal. However, MIMO communications will be mainly used in wideband systems that experience frequency-selective fading, and, thus, intersymbol interference (ISI) [42, 43]. This may cause significant performance degradation to the MIMO communications schemes if suitable techniques such as equalization are not applied for mitigating ISI [44]. Another possibility is to apply a multicarrier transmission scheme such as OFDM [13, 14, 45]. OFDM divides the frequency-selective channel into a set of parallel frequency-flat fading channels and applies a guard interval in the OFDM symbol, called a cyclic prefix (CP), which should be long enough to accommodate the delay spread of the channel. Thus, OFDM does not require any additional equalization techniques for mitigation of ISI. OFDM can be implemented with an inverse fast Fourier transform (IFFT) at the transmitter and a fast Fourier transform (FFT) at the receiver, which makes it simple and attractive for practical use. A downside of OFDM is that it is sensitive to synchronization errors and suffers from a larger peak-to-average power ratio (PAPR) [46]. However, MIMO–OFDM has been identified as a promising approach for high spectral efficiency wideband systems and has been included in many upcoming wireless standards.

1.3 Review of the earlier and parallel work

A review of the literature and parallel work related to the detection in MIMO systems is presented in this section. The review includes the main algorithms, architecture designs and implementations introduced in the literature. First, the optimal detection methods are reviewed in Section 1.3.1. The linear detection methods and non-linear improvements are presented in Section 1.3.2. A class of algorithms, generally called as sphere detectors (SD), are reviewed in Section 1.3.3. Finally, other approximations of optimal detection methods are reviewed in Section 1.3.4.

1.3.1 Optimal detection methods

The MIMO detection problem of an uncoded system can be considered as a so-called integer least squares problem, which can be solved optimally with a hard-output maximum likelihood (ML) detector [47]. The ML detector solves optimally the so-called closest lattice point problem by calculating the Euclidean distances (EDs) between the received signal vector and points in the lattice formed by the channel matrix and the received signal, and selects the lattice point that minimizes the Euclidean distance to the received vector. The ML detection problem can be solved with an exhaustive search, i.e., checking all the possible symbol vectors and selecting the closest point. The ML detector achieves a diversity order of N_{R}, i.e., a full spatial diversity with regard to the number of receive antennas; however, it is computationally very complex and not feasible as the set of possible points increases.

Practical communication systems apply FEC coding in order to achieve near capacity performance. The optimal way to process the spatially multiplexed and FEC coded data sequence would be to use a joint detector and decoder for the whole coded data sequence and decode the most probable data sequence. However, the joint detection and decoding problem is non-deterministic polynomial-time (NP)-hard, which makes it computationally very complex with realistic parameters and not feasible with the current technology [48, 49]. However, the optimal receiver can be approximated by using an iterative receiver with a separate soft-input soft-output (SfISfO) detector and decoder, which exchange reliability information between the units [4]. The optimal SfISfO detector would be the maximum *a posteriori* (MAP) detector [50], which is more complex than an ML detector and also too complex for systems with a large number of transmitted spatial layers and high order modulation. The soft output *a posteriori* probabilities (APPs) are expressed with log-likelihood ratio (LLR) values. The Jacobian logarithm [51] and the so-called log-MAP algorithm reduces the complexity of the original symbol-by-symbol MAP algorithm [50], which is also called the BCJR algorithm after the authors. A less complex max-log-MAP approximation can also be applied with rather small performance loss compared to the log-MAP [51].

Implementations

Optimal detection methods are not typically feasible for hardware implementation with a high number of transmit antennas and high order constellations. However, some very large scale integration (VLSI) architecture designs and implementations have been introduced in the literature. VLSI architecture design and implementation of an ML detector with approximate a posteriori probability for a maximum 4 × 4 MIMO has been proposed in [52] and further developed in [53]. A more efficient VLSI design of a 4 × 4

MIMO ML detector has been introduced in [54]. More recently, a VLSI architecture design and implementation of a reduced complexity exhaustive search max-log-MAP algorithm, which can be applied for two transmit antennas with up to 64- quadrature amplitude modulation (QAM), has been introduced in [55].

1.3.2 Linear detection methods and non-linear improvements

Suboptimal linear detectors [38, 56, 57] offer straightforward and low complexity solutions to suppress the interference between transmitted spatial layers. Linear detectors based on the principle of zero-forcing (ZF) suppress the interference among the spatial layers completely by filtering the received signal with an inverse of the channel, but also cause noise enhancement. The linear minimum mean square error (LMMSE) criterion based detectors also take the noise into account and minimize the total expected error, which typically results in better performance compared to the ZF based detector. The linear detectors have adequate performance in low correlated channels, but suffer from rather poor performance in correlated fading channels especially those with a high coding rate [58, 59]. The diversity order of linear detectors is equal to $N_{\mathrm{R}} - N_{\mathrm{T}} + 1$.

An alternative detection strategy known as nulling and cancellation, which is based on linear detection methods, has been proposed [39, 60, 61]. The nulling and cancellation techniques can be divided into two main categories: parallel interference cancellation (PIC) [62] and successive interference cancellation (SIC) [38]. PIC technique applies a linear detector to obtain an initial estimate of the transmitted data layer. Each layer is then nulled with the estimate from other layers followed by an additional detection stage to refine the estimate. A slightly different detection approach, where a tree search algorithm is applied together with the PIC technique, has been presented in [63]. The SIC technique detects the transmitted signal one layer at a time, and removes the interference of the detected layers from the received signal before detecting the next layer. Thus, SIC achieves an increase in diversity order after each detected layer [38]. The performance of SIC depends highly on error propagation of the first detected layers as an incorrect first decision leads easily to decision errors with the rest of the layers. The error propagation can be minimized by detecting the most reliable layers first. The ordered serial interference cancellation (OSIC) detector or the V-BLAST architecture [1, 39] sorts the layers in descending order with the signal to interference plus noise ratio (SINR) and detects the strongest layer first. The SIC detectors achieve a diversity order of $N_\mathrm{R} - N_\mathrm{T} + k$ for the kth detected layer. Thus, the SIC detectors offer better performance compared to linear detectors and the OSIC detector achieves an additional diversity gain for the weakest layers, which are detected last.

Another technique called lattice reduction (LR) can be applied to improve the performance of linear detectors and their non-linear improvements [64–68]. Lattice reduction techniques modify the ill-conditioned channel matrix to nearly an orthogonal basis by linear processing and is, thus, more suitable for linear or SIC detection. The Lenstra-Lenstra-Lovasz (LLL) algorithm [69] is a popular and efficient algorithm to determine a reduced basis for a channel matrix. The LR aided linear detectors are able to achieve the same diversity order as the ML detector in V-BLAST systems with a rather small complexity addition [64].

Implementations

The implementation complexity of linear detectors is feasible as the filtering coefficients need to be recalculated only as the channel state changes, and, thus, the complexities of the linear detectors are much lower compared to the optimal detectors and their variants. Several approaches, such as QR decomposition (QRD), Cholesky decomposition or direct matrix inversion (DMI) [70, 71], exist to solve the matrix decomposition required by the linear detectors and more advanced methods based on them [70, 72]. The QRD based methods, which have good numerical properties with fixed-point implementation, are typically applied in the literature [70]. The implementation of QRD can be done with several different methods, e.g., Gram-Schmidt orthogonalization, Givens rotations, or Householder transformations [70]. The coordinate rotation digital computation (CORDIC) [73] algorithm and the squared Givens rotation (SGR) [74] algorithm are square root free algorithms that can be applied for the calculation of Givens rotations.

The architectural design of matrix operations in the literature is often based on systolic array structures with communicating processing elements (PEs) due to the low latency requirements of MIMO receivers [75, 76]. A highly parallel and complex architecture can be considered for high speed applications [77] and a less complex architecture with easy scalability and time sharing PEs can be considered for less latency critical applications [78].

Various architecture designs and implementations exist in the literature. Pipelined VLSI architecture and implementation of a 4×4 MIMO LMMSE detector based on DMI has been presented in [71]. An efficient QRD and CORDIC based 4×4 MIMO LMMSE detector implementation on a field programmable gate array (FPGA) has been presented in [79]. A low complexity VLSI architecture of a 4×4 MIMO SIC detector based on a square root algorithm via CORDIC is presented in [80]. A VLSI architecture and FPGA implementation of a 4×4 MIMO SIC detector based on QRD and CORDIC has been introduced in [81] and later implemented in an application specific integrated circuit (ASIC) in [82]. The performances and implementation complexities and latencies of a LMMSE and SIC detectors are presented in [83]. Recently various VLSI architecture designs and implementations of LR techniques based on a modified LLL algorithm have been introduced in the literature, e.g., in [84–87].

1.3.3 Sphere detection

A class of algorithms, generally called sphere detectors (SD) [48, 88–91], can be used to solve or approximate the hard output ML solution with reduced complexity compared to the full-complexity ML detector. SDs are based on preprocessing and tree search algorithms and their application to the MIMO detection problem has gained renewed attention in the literature during the last few decades [92]. The preprocessing algorithm, e.g., QRD, is used to convert the lattice search to a form that is problem friendly and applicable to a tree search algorithm. More advanced preprocessing techniques such as LR or ordering of the layers can be applied to hasten the tree search, and, thus, reduce the complexity [47, 93]. The tree search algorithm then aims to find the shortest path in a search tree formed by the MIMO channel matrix and the transmitted symbols, i.e., solves the exact ML solution or suboptimal solution depending on the algorithm search strategy. The algorithms in the literature are often divided into three categories according to the search strategy: the breadth-first (BF) search, the depth-first search (DF), and the metric-first (MF) search [93–95].

Tree search algorithms

Depth-first algorithms are based on a sequential search and go through a variable number of nodes in the search tree depending on the channel realization and the SNR. The algorithms explore the tree along the depth until the cost metric of the path is below a defined threshold called a sphere radius. They then return and pursue another unexplored path. DF algorithms are able to find the exact ML solution if the search is not bounded. The Pohst enumeration method is often considered to be the original sphere algorithm [89]. The algorithm search complexity is bounded by selecting a constant sphere radius, which limits the search in the tree to the most likely paths. The main problem with implementation of the Pohst algorithm is the selection of the sphere radius. The Viterbo-Boutros (VB) implementation [96] introduced a modification of the original Pohst algorithm with adaptive updating of the sphere radius. Thus, with the VB implementation applied, the algorithm can be started with an infinite radius and the problem of the initial radius can be solved. The Schnorr-Euchner enumeration (SEE) [97] can be seen as a more efficient modification of the Pohst enumeration and VB implementation, where the admissible paths, which are also called nodes, of each layer are spanned in a zig-zag order starting with the closest middle point, whereas the Pohst enumeration searches the admissible nodes without any ordering [48].

Breadth-first algorithms, such as the M algorithm [98] or the K-best algorithm [99, 100] with sphere radius, extend the search in a layer-by-layer basis with multiple paths and always proceed in the depth direction of the tree. The algorithms always keep a constant number of candidate paths in each layer of the tree if no sphere radius constraint is introduced, but also requires sorting of the candidate paths at each tree layer. BF algorithms guarantee a fixed number of visited nodes, which makes the algorithm very suitable for implementation. However, the breadth-first search strategy does not guarantee the exact ML solution and the search as such is inefficient in term of visited nodes especially with higher order modulation compared to the other tree search strategies.

Metric-first algorithms are based on a sequential search method and the search always proceeds along a path with the best cost metric among the stored paths in the tree search [93, 95]. MF algorithms are based on Dijktra's algorithm [101, 102], which was originally used to solve the single-source shortest path problem for a graph. The stack algorithm was introduced for the decoding of convolutional coding by Zigangirov [103] and Jelinek [104]. The application of metric-first algorithms for MIMO detection has been applied in [105–107]. MF algorithms find the exact ML solution and the search strategy is efficient in terms of visited nodes in the search tree, but requires storing and ordering of the paths studied [95].

Soft output SDs

Hard output sphere detectors may cause significant performance degradation when used as a detector part in an iterative receiver compared to the optimal soft output MAP detector in a communication system with FEC. However, there are methods proposed in the literature to modify hard output SDs to give soft reliability information of the transmitted bits as an output. The list sphere decoder (LSD), which was originally proposed by Hochwald and ten Brink [4], applies the tree search algorithm to obtain a list of candidate symbol vectors instead of finding only the ML estimate. The candidate list is then used to calculate the soft output LLRs, i.e., the probabilistic soft information of the transmitted coded bits. The size of the candidate list and the bounding of the tree search define the tradeoff between complexity and the quality of the soft output information. List sphere detector algorithms are modified to continue the tree search until a defined list is obtained and the algorithms can often be derived from the sphere detector algorithms with minor modifications. Another approach is to execute separate directed tree searches to find the counterhypotheses bits for the ML solution to be used in the LLR calculation as proposed in [108, 109]. The approach requires multiple tree searches, but fewer nodes have to be studied per search compared to LSD algorithms as only a single solution per search is required. Solutions with lower complexity have been developed further in [110–113]. The soft output sphere detectors typically use max-log-MAP approximation to calculate the LLRs [51].

SD implementations

The practical feasibility of various SD versions is supported recently by architecture designs and practical implementations reported in the literature. Most of the implementations are based on VLSI technology and designed for systems with flat fading channels, e.g., OFDM systems. In hardware implementations of SDs the key problem is to determine the number of nodes visited and the required hardware resources in the detection of the symbol vector with a given channel realization. In general, the BF based SD implementations guarantee a fixed detection throughput with variable performance, while the DF and MF based implementations offer better performance with variable detection throughput. However, various implementation techniques have also been proposed to make the SDs more implementation friendly and they are briefly reviewed below.

VLSI architecture designs with multiple parallel tree pruning units have been proposed for DF based SD in [114, 115]. Other VLSI architecture designs and implementations of a hard output DF based sphere detector for 4×4 MIMO systems with 16-QAM have been considered in [116, 117] and further developed into a soft output detector in [111, 113, 118]. Implementation trade-offs between complexity and performance, such as the use of different norms [117], LLR clipping [4, 111] and bounding of the runtime constraints [111], have also been considered in the literature to enable efficient implementation. Pipelined and parallel VLSI architectures and implementations of soft output BF based SD have been introduced for a 4×4 MIMO system with 16-QAM in [99, 100], where [99] includes a technique to improve the quality of the soft output. An architecture design and implementation of BF based SD with an enhanced enumeration and sorting method has been proposed for 64-QAM in [119]. A slightly different BF based SD design, which does not require sorting, has been introduced in [120] for a 4×4 MIMO system with 16-QAM and with 64-QAM in [121, 122]. A VLSI architecture and implementation of hard output MF based SD has been introduced for 4×4 MIMO system with 16-QAM in [123].

1.3.4 Other approximations of optimal detection

Another approximation method of hard output ML MIMO detection presented in the literature is the semidefinite relaxation (SDR) approach, which was originally introduced to the area of digital communications in [124]. SDR changes the exponentially complex ML detection problem into polynomial. This is done by first solving a simpler relaxed finite alphabet problem on the transmitted symbol vector into a matrix inequality and then applying semidefinite programming to solve the resulting problem. The conceptually simple SDR approach was proposed for MIMO detection in [125, 126] and it has also been shown to achieve maximum diversity order [127].

A probabilistic approximation method has been proposed for the computation of MAP detector soft-outputs via the sequential Monte Carlo methods in [128]. The proposed soft-input soft-output method can be used without CSI at the receiver and it can be easily calculated with parallel computations, which makes it feasible for implementation. The convergence properties and the performance of the methods have been further studied in [129, 130]. Moreover, the implementation details of such a detector have been studied and an efficient hardware architecture presented in [131].

A novel detection method called a layered orthogonal lattice detector (LORD) has been proposed in the literature [132, 133]. LORD achieves MAP performance in the case of two transmit antennas, at the same time keeping complexity much lower than the exhaustive search-based ML detection technique and an approximation of max-log-MAP in the case of more antennas. LORD reformulates the lattice in the detector, which results in complexity savings. A more implementation oriented version of the algorithm was proposed in [134]. Furthermore, improved versions of the LORD called as turbo-LORD were introduced in [135, 136].

A MIMO detection method based on fully connected trellis and a forward-backward recursion is proposed in [137]. The method approximates the MAP detector and is based on a suboptimal double-direction trellis traversal algorithm. A fixed throughput VLSI architecture and implementation is presented in [137].

1.4 Scope and objectives of the thesis

The scope of this thesis in a broad sense is to develop receiver algorithms to enable advanced transmission techniques that enable a more efficient use of the radio spectrum. The motivation for this is the fact that the radio spectrum is a limited resource and, thus, should be used more efficiently to serve the increasing needs. MIMO antenna techniques and OFDM air interface have been identified and widely accepted as key parts of spectrum efficient future wireless communication systems. The reception of a spectrum efficient SM transmission in a MIMO–OFDM system requires a detector in the receiver to separate the transmitted data streams. Different detector algorithms with different performance characteristics can be applied for MIMO signal detection. More advanced algorithms, such as the list sphere detector, offer better performance compared to the more traditional linear detectors and, thus, enable higher data rates and more power efficient transmissions, which reduce the network structure costs and/or guarantee more satisfied customers for the network operators. However, more advanced algorithms typically also require more complex signal processing, which leads to increased silicon area and power consumption at the receiver. A fair comparison of the algorithms requires knowledge of both implementation complexity and performance. The objective is to study and compare the implementation complexity and the performance of different detector algorithms.

The algorithm development process from theory to a feasible implementation to be used in a telecommunication system requires multiple different phases, which are usually carried out by different people in industry. Another objective of this thesis is to apply a joint process between different development phases to ensure efficient algorithm implementation. The theoretical algorithm work and performance simulations with floating point representation are done first to create the algorithm and ensure the performance. The algorithms are modified to be feasible for implementation, which requires some implementation tradeoffs between performance and complexity. The architecture design is a key point in efficient implementation, where the main functional units and the possibilities for parallelism and pipelining of the algorithm are identified and the architecture is designed. The hardware implementation is then done with a handwritten hardware description language (HDL) description or with a high level synthesis (HLS) tool produced HDL description for selected technology. FPGA and ASIC technologies are considered for algorithm implementation.

1.5 Author's contribution and outline of the thesis

The thesis is written as a monograph for the sake of clarity, but parts of the contributions in Chapters 2–5 have been published in thirteen original publications, including two journal papers [138, 139] and eleven conference papers [140–150]. The author has had the main responsibility in developing the original ideas as well as writing the papers [138–148]. The other authors provided ideas, comments and help to the first author with the exceptions explained below. The author also guided and participated closely in the work in [149, 150].

The author has also implemented the detector algorithms in the simulation software and performed all computer simulations. The other parts of the simulation software have been developed by Dr. Nenad Veselinovic and Dr. Mikko Vehkaperä in CWC. The typical urban (TU) channel models, which were produced with the model by Dr. Esa Kunnari [151], and the WINNER [152, 153] channel models were used in the computer simulations. The LMMSE detector algorithm architecture design, hardware implementation, and measurements have been done in co-operation with the authors in [140, 142]. The K-best-LSD architecture design and implementation have been done in co-operation with the first author in [149, 150]. The author has been responsible for the architecture design and the hardware implementation of the other LSD algorithms presented.

The thesis also includes some additional results by the author, which have not yet been published. The results on the L1 norm in Section 3.5 have not been published.

The LSD architecture design in Section 4.2 includes additional details on parallelism and pipelining, and detailed scalability and timing results that have not been published. More complete results, i.e., synthesis results, and comparison of LSD implementation results, have been done on the hardware implementation in Section 5.2 that have not been published. The measurement results on the RACE platform in Section 5.3.2 have not been published. The author had the main responsibility in the unpublished work in Section 3.5 and the more detailed results in Sections 4.2 and 5.2. The measurements with the RACE platform in Section 5.3 have been done by Markku Jokinen and guided by the author. The thesis follows the organization given below.

Chapter 2 considers detection in MIMO–OFDM systems. A generic link level system model is presented for the MIMO–OFDM downlink cellular system. The detector algorithms considered, which include the optimal ML and MAP detectors, the linear MMSE detector, and the list sphere detector, are introduced and described, and the performances of the detectors are compared. Different algorithm choices are introduced for the LMMSE detector and the LSD. The results have been included in part in [138–141, 143, 146]. The contributions of this chapter are as follows. The LMMSE detector is derived from existing algorithms in the literature [38, 73, 74]. Different existing preprocessing and tree search algorithms from the literature are derived as LSD based on [4]. Numerical performance examples of different detectors are presented and compared. The performance and complexity of LSD preprocessing algorithms are also presented and compared.

Chapter 3 identifies and presents some key challenges encountered in the implementation of the LSD algorithms in practical wireless systems. The results have been included in part in [138, 139, 144, 145]. The contributions of this chapter are as follows. The effects of limiting the dynamic range of the soft output LLR values of the LSD are studied. Similar studies with slightly different approaches also exist in the literature, e.g., in [4, 111]. The use of the real valued and the complex valued signal models in the LSD is studied and compared in terms of complexity. Two methods are proposed to limit and fix the search complexity of the sequential search LSD algorithms. Limiting the LSD algorithm search complexity has also been proposed earlier in the literature [4, 111], and the proposed scheduled search (SS) method is further developed from [111]. A novel memory sphere radius is proposed for the IR-LSD algorithm to decrease the number of stored candidates and the complexity of the algorithm tree search. The use of the L1 norm in the LSD algorithm, which has been proposed in [117], is studied and compared to the use of the L2 norm in terms of performance and complexity. A simple scaling method is proposed for the LSD algorithm with the L1 norm to approximate the L2 norm and to compensate for the performance loss with the L1 norm. The performance of the iterative receiver with a variable number of global iterations (GIs) and their effect on the computational complexity have been studied.

Chapter 4 focuses on the architecture design for the detector algorithms considered. The contributions of this chapter are as follows. The key functional units of the algorithms are identified and efficient architectures are designed for MIMO–OFDM systems. The possibilities for parallelism, pipelining and scalability in the microarchitecture units are studied and analyzed. The results have been included in part in [139, 140, 147–150]. The LMMSE detector architectures for both CORDIC and SGR algorithms are designed with systolic array structures. Systolic array based design with slightly different approaches have also been introduced earlier in the literature, e.g., in [75–78]. Architectures are introduced for the SQRD preprocessing algorithm, for the considered LSD algorithms and for the max-log-MAP LLR calculation unit. Slightly different architecture designs have also been introduced earlier in the literature, e.g., for DF based SD in [111, 113, 116–118], for BF based SD in [99, 100], and for MF based SD in [123].

Chapter 5 contains the hardware implementation results of the LMMSE detector and LSD and their designed architectures. The contributions of this chapter are as follows. The synthesis results, which include complexity, power consumption, and latency, are presented and analyzed. The feasibility of the implementations on telecommunication standards is also studied. Moreover, measurement results with the algorithms on FPGA based hardware testbeds are presented. The results have been included in part in [139, 142, 147, 148]. Several earlier and parallel hardware implementation results also exist in the literature, e.g., LMMSE detectors in [71, 79] and SDs in [99, 100, 111, 113, 116–118, 123]. The implementation results presented in the thesis are competitive with the other implementations when performance, complexity, power consumption and detection rate are jointly considered. Furthermore, there have been no general comparison and analysis of different LSD algorithm implementations reported in the literature.

Chapter 6 concludes the thesis. The main results and conclusions are summarized. Moreover, some remaining open questions and directions for future research are discussed.

2 Detection in MIMO-OFDM systems

This chapter describes the MIMO–OFDM system model assumed in the thesis and introduces the considered detection algorithms.

2.1 System model

A MIMO–OFDM system is considered with N_{T} transmit (TX) antennas and N_{R} receive (RX) antennas. A spatial multiplexing (SM) transmission with N_{T} spatial streams is used with a quadrature amplitude modulation (QAM) constellation and with the assumption $N_{\mathrm{R}} \geq N_{\mathrm{T}}$. The received signal at baseband can be expressed for each of the N_{c} subcarriers in terms of code symbol interval as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^{N_{\mathrm{R}} \times 1}$ is the received signal vector, $\mathbf{x} \in \mathbb{C}^{N_{\mathrm{T}} \times 1}$ is the transmit symbol vector and $\mathbf{n} \in \mathbb{C}^{N_{\mathrm{R}} \times 1}$ is the noise vector with independent and complex zero-mean Gaussian elements with equal power σ^2 for both real and imaginary parts. The channel matrix $\mathbf{H} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{T}}}$ contains complex Gaussian fading coefficients with unit variance. The entries of $\mathbf{x}$ are chosen independently from a complex QAM constellation Ω with sets of Q transmitted coded binary information bits $\mathbf{b} = [b_1, \ldots, b_Q]^{\mathrm{T}}$ per symbol, i.e., $|\Omega| = 2^Q$. The corresponding coded transmission rate is $R = N_{\mathrm{T}}Q$ bits per channel use (bpcu).

In the case of sphere detectors, a real equivalent system model is often assumed. Any complex linear MIMO system model can be reduced to an equivalent real model by separating the real and imaginary parts. The real system model can be written as

$$\mathbf{y}_{\mathrm{r}} = \mathbf{H}_{\mathrm{r}}\mathbf{x}_{\mathrm{r}} + \mathbf{n}_{\mathrm{r}}, \tag{2}$$

where the real-valued channel matrix is

$$\mathbf{H}_{\mathrm{r}} = \begin{bmatrix} \mathrm{Re}\{\mathbf{H}\} & -\mathrm{Im}\{\mathbf{H}\} \\ \mathrm{Im}\{\mathbf{H}\} & \mathrm{Re}\{\mathbf{H}\} \end{bmatrix} \in \mathbb{R}^{2N_{\mathrm{R}} \times 2N_{\mathrm{T}}}, \tag{3}$$

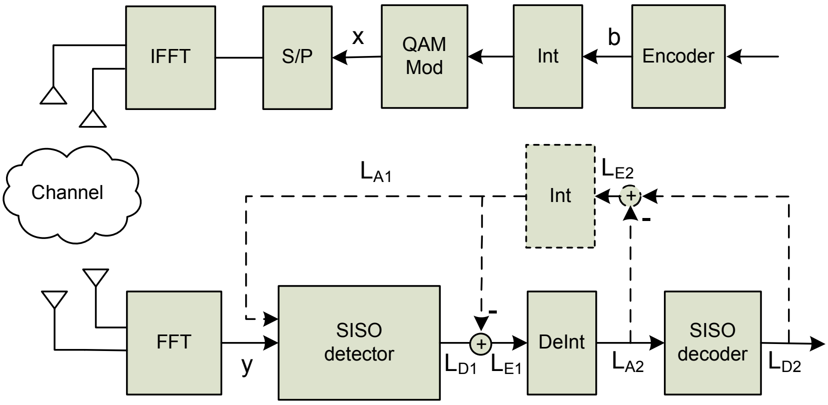


Fig 1. A MIMO-OFDM system with N_T transmit and N_R receive antennas.

and the real-valued vectors are defined as

$$\begin{aligned}\mathbf{y}_r &= \left[\mathrm{Re}\{\mathbf{y}^T\}\ \mathrm{Im}\{\mathbf{y}^T\}\right]^T \in \mathbb{R}^{2N_R\times 1},\\ \mathbf{x}_r &= \left[\mathrm{Re}\{\mathbf{x}^T\}\ \mathrm{Im}\{\mathbf{x}^T\}\right]^T \in \mathbb{Z}^{2N_T\times 1},\\ \mathbf{n}_r &= \left[\mathrm{Re}\{\mathbf{n}^T\}\ \mathrm{Im}\{\mathbf{n}^T\}\right]^T \in \mathbb{R}^{2N_R\times 1},\end{aligned} \tag{4}$$

where $\mathrm{Re}\{\cdot\}$ and $\mathrm{Im}\{\cdot\}$ denote real and imaginary parts, respectively. Let us define the new dimensions $M_R = 2N_R$ and $M_T = 2N_T$. The real symbol alphabet is now $\Omega_r = \mathbb{Z}$, e.g., $\Omega_r \in \{-3,-1,1,3\}$ in the case of 16-QAM. For simplicity, the subindices are omitted in the sequel.

In the practical case of a system with FEC, the optimal way to decode the coded signal would be to use a joint detector and decoder for the whole coded data block. This, however, is computationally very complex and not feasible with the current technology. A suboptimal way is to have a separate soft-input soft-output (SfISfO) detector and decoder at the receiver, where the detector generates soft reliability information as an input to the decoder. The turbo principle can be applied in the receiver so that the detector and decoder exchange the information in an iterative fashion as illustrated in the block diagram of the system in Figure 1. The detector generates soft output APPs L_{D1} from received data $\mathbf{y}$ and *a priori* information L_{A1}, and calculates extrinsic information L_{E1}. This information is fed as *a priori* information L_{A2} to the decoder after interleaving. The decoder output APPs L_{D2} can then be fed back to the detector.

The *a posteriori* probability log-likelihood ratio (LLR) of the kth transmitted bit b_k, conditioned on the received signal vector $\mathbf{y}$, is denoted as $L_D(b_k)$ and is defined to be the ratio of the conditional probabilities of the bit taking its two possible values, i.e.,

$$L_{\mathrm{D}}(b_k) = \ln \frac{P(b_k = +1|\mathbf{y})}{P(b_k = -1|\mathbf{y})}. \tag{5}$$

By using Bayes' theorem, the probability can be written as [4, 154]

$$\begin{aligned} L_{\mathrm{D}}(b_k) &= \ln \left(\frac{p(\mathbf{y}|b_k = +1)}{p(\mathbf{y}|b_k = -1)} \frac{P(b_k = +1)}{P(b_k = -1)} \right) \\ &= \ln \frac{P(b_k = +1)}{P(b_k = -1)} + \ln \frac{p(\mathbf{y}|b_k = +1)}{p(\mathbf{y}|b_k = -1)} \\ &= L_{\mathrm{A}}(b_k) + L_{\mathrm{E}}(b_k|\mathbf{y}), \end{aligned} \tag{6}$$

where $L_{\mathrm{A}}(b_k)$ is the *a priori* information and $L_{\mathrm{E}}(b_k)$ is the extrinsic information of the bits provided by the detector or decoder.

2.2 ML detector

The MIMO detection problem of an uncoded system can be considered as a so-called integer least squares problem. The problem can be solved optimally with a maximum likelihood (ML) detector [47], which minimizes the error probability. The ML detector solves optimally the so-called closest lattice point problem by calculating the Euclidean distances (EDs) between received signal $\mathbf{y}$ and lattice points $\mathbf{Hx}$ and selects the lattice point that minimizes the Euclidean distance to the received vector $\mathbf{y}$, i.e.,

$$\hat{\mathbf{x}}_{\mathrm{ML}} = \arg \min_{\mathbf{x} \in \Omega^{N_{\mathrm{T}}}} ||\mathbf{y} - \mathbf{Hx}||_2^2. \tag{7}$$

The ML detection problem can be solved with an exhaustive search, i.e., checking all the $\Omega^{N_{\mathrm{T}}}$ possible symbol vectors $\mathbf{x}$ and selecting the closest point.

2.3 MAP detector

The optimal SfISfO detector for the calculation of the APP $L_{\mathrm{D}}(b_k)$ is the maximum *a posteriori* (MAP) detector. The probability of a transmitted bit $b_k = +1$ is equal to the sum of all the probability combinations containing a $b_k = +1$ for that given bit. Then, for a system containing additive white Gaussian noise (AWGN), the probability can be determined directly from the cost information known about the candidates as

$$p(\mathbf{y}|b_k = +1) = \frac{2}{|\Omega|^{N_{\mathrm{T}}} \sqrt{2\pi\sigma^2}} \sum_{\mathbf{x} \in \chi_{k,+1}} e^{\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2}}, \tag{8}$$

where $\chi_{k,+1} = \{\mathbf{x}|b_k = +1\}$ is the set of $\Omega^{N_\mathrm{T}-1}$ bit vectors $\mathbf{x}$ having $b_k = +1$. The MAP solution in the logarithm domain, also often called the log-MAP, solution is obtained by calculating the *a posteriori* probability LLRs $L_\mathrm{D}(b_k)$ by using all the possible $\Omega^{N_\mathrm{T}-1}$ bit vectors $\mathbf{x}$ with both conditional probability variables $p(\mathbf{y}|b_k = \pm 1)$ as

$$\begin{aligned} L_\mathrm{D}(b_k) &= L_\mathrm{A}(b_k) + \ln \frac{p(\mathbf{y}|b_k = +1)}{p(\mathbf{y}|b_k = -1)} \\ &= L_\mathrm{A}(b_k) + \ln \frac{\sum_{\mathbf{x}\in\chi_{k,+1}} \exp(\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2})}{\sum_{\mathbf{x}\in\chi_{k,-1}} \exp(\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2})} \\ &= L_\mathrm{A}(b_k) + \ln \sum_{\mathbf{x}\in\chi_{k,+1}} \exp(\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2}) - \ln \sum_{\mathbf{x}\in\chi_{k,-1}} \exp(\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2}), \end{aligned} \tag{9}$$

The exact calculation of (9) is obviously a very complex task as the number of considered bit vectors $\Omega^{N_\mathrm{T}-1}$ increases exponentially with the number of transmit antennas N_T and the constellation Ω used. The well known Jacobian algorithm can be used to compute the logarithm sum in (9) as follows

$$\begin{aligned} \ln(e^{a_1} + e^{a_2}) &= \max(a_1, a_2) + \ln(1 + e^{|a_1 - a_2|}) \\ &= \max(a_1, a_2) + f_c(|a_1 - a_2|), \end{aligned} \tag{10}$$

where $f_c(.)$ is a correction function [51]. The correction function can be approximated with a look-up table with negligible performance degradation [51, 155]. A less complex max-log approximation is calculated by approximating the sum in (10) by only the maximum value as

$$\ln(e^{a_1} + e^{a_2}) \approx \max(a_1, a_2). \tag{11}$$

The correction term $\ln(1 + e^{|a_1 - a_2|})$ is already quite small when $|a_1 - a_2| > 2$ and the performance loss due to a max-log approximation is rather small compared to the log-MAP algorithm [51]. The *a posteriori* probability LLRs $L_\mathrm{D}(b_k)$ can then be calculated as

$$L_\mathrm{D}(b_k) \approx L_\mathrm{A}(b_k) + \max_{\mathbf{x}\in\chi_{k,+1}} (\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2}) - \max_{\mathbf{x}\in\chi_{k,-1}} (\frac{-||\mathbf{y}-\mathbf{Hx}||^2}{2\sigma^2}). \tag{12}$$

2.4 LMMSE detector

The use of suboptimal linear detectors [38] offers a low complexity solution compared to the optimal ML and MAP detectors for the separation of transmitted data streams at the receiver. The linear detector is an attractive alternative if the rather poor performance of the detector in correlated fading channels and in a low signal-to-noise ratio (SNR) environment is acceptable [58]. The linear minimum mean square error (LMMSE) criterion based detector [38] minimizes the mean square error (MSE), i.e., the interference and noise, between the transmitted signal vector $\mathbf{x}$ and the soft output vector $\tilde{\mathbf{x}}$. The LMMSE design criterion for an OFDM subcarrier is

$$\mathscr{D}^2_{\mathrm{LMMSE}} = \min_{\mathbf{W}} \mathrm{E}\big\{ \| \mathbf{x} - \mathbf{W}^{\mathrm{H}}\mathbf{y} \|^2_{\mathrm{F}} \big\}, \tag{13}$$

where $\mathbf{W} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{T}}}$ is the LMMSE coefficient matrix, and $\|\mathbf{A}\|^2_{\mathrm{F}} = \mathrm{tr}\left(\mathbf{A}\mathbf{A}^{\mathrm{H}}\right)$ denotes a squared Frobenius norm of the matrix $\mathbf{A}$. By using the well known Wiener solution [156], the LMMSE detector for MIMO-OFDM can be then reduced to

$$\mathbf{W} = \left(\mathbf{H}\mathbf{R}_{xx}\mathbf{H}^{\mathrm{H}} + \mathbf{R}_{nn}\right)^{-1}\mathbf{H}\mathbf{R}_{xx}, \tag{14}$$

where $\mathbf{R}_{xx}$ and $\mathbf{R}_{nn}$ are the symbol and noise covariance matrices, respectively. Because the LMMSE detector has no prior knowledge of the channel code structure, we assume $\mathbf{R}_{xx} = E_{\mathrm{s}}\mathbf{I}_{N_{\mathrm{T}}}$, where E_{s} is the transmitted symbol power. The thermal noise between receive antennae and subcarriers is also considered to be uncorrelated, i.e., $\mathbf{R}_{nn} = 2\sigma^2\mathbf{I}_{N_{\mathrm{R}}}$. Then the solution of (14) is equivalent to

$$\mathbf{W} = \left(\mathbf{H}\mathbf{H}^{\mathrm{H}} + \frac{2\sigma^2}{E_{\mathrm{s}}}\mathbf{I}_{N_{\mathrm{R}}}\right)^{-1}\mathbf{H}. \tag{15}$$

The soft output of the LMMSE detector $\tilde{\mathbf{x}}$ is then calculated by multiplying the received signal vector $\mathbf{y}$ with the conjugate transpose of the coefficient matrix $\mathbf{W}$ as follows

$$\tilde{\mathbf{x}} = \mathbf{W}^{\mathrm{H}}\mathbf{y}. \tag{16}$$

The calculation of the LMMSE solution in (15) requires a matrix inversion operation which is computationally the most complex task of the detector. The solution for the LMMSE front-end coefficients $\mathbf{W}$ can be seen as a common problem of solving a linear system

$$\mathbf{A}\mathbf{X} = \mathbf{B}, \tag{17}$$

where the matrix to be inverted, the desired LMMSE coefficients and the right hand side of the equation are defined, respectively, as $\mathbf{A} = \mathbf{H}\mathbf{H}^{\mathrm{H}} + \frac{2\sigma^2}{E_{\mathrm{s}}}\mathbf{I}_{N_{\mathrm{R}}} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{R}}}$, $\mathbf{X} = \mathbf{W} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{T}}}$, $\mathbf{B} = \mathbf{H} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{T}}}$. The solution for (17) can be calculated by using QR decomposition (QRD) via Givens rotations [70]. The coordinate rotation digital computer

(CORDIC) [73] and the squared Givens rotations (SGR) [74] algorithms are considered for the calculation of QRD, and the back substitution algorithm [70] and the triangular matrix inversion algorithm [157] to calculate the solution after QRD.

2.4.1 CORDIC algorithm

In QRD a symmetric positive definite matrix **A** from (17) can be factored as $\mathbf{A} = \mathbf{QR}$, where $\mathbf{Q} \in \mathbb{C}^{N_\mathrm{R} \times N_\mathrm{R}}$ is a unitary matrix, i.e., $\mathbf{Q}^\mathrm{H}\mathbf{Q} = \mathbf{Q}\mathbf{Q}^\mathrm{H} = \mathbf{I}$ and $\mathbf{R} \in \mathbb{C}^{N_\mathrm{R} \times N_\mathrm{R}}$ is an upper triangular matrix. The CORDIC method provides pipelined implementations of the Givens rotations for QRD using shifts and additions or subtractions without the need to compute trigonometric functions or square roots [73, 158]. Then (17) can be written as

$$\mathbf{QRX} = \mathbf{B} \tag{18}$$

$$\mathbf{RX} = \mathbf{Q}^\mathrm{H}\mathbf{B}. \tag{19}$$

Matrix **X** can be solved from the upper triangular system using back substitutions [70].

The two-dimensional rotation step in Givens rotations annihilates one element at a time from the given appropriate pairs of rows. The rotation step is repeated several times for the matrix **A** in order to construct **R** and **Q**. In one rotation step the *k*th element of the row $\mathbf{a} = [0, \ldots, 0, a_k, \ldots, a_{N_\mathrm{R}}]$ is to be annihilated by the rotation. Another row $\mathbf{r} = [0, \ldots, 0, r_k, \ldots, r_{N_\mathrm{R}}]$ is applied in order to obtain QRD. For real valued **a** and **r** the rotation is

$$\begin{aligned} \begin{bmatrix} \bar{\mathbf{r}} \\ \bar{\mathbf{a}} \end{bmatrix} &= \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{a} \end{bmatrix} \\ &= \cos(\theta) \begin{bmatrix} 1 & \tan(\theta) \\ -\tan(\theta) & 1 \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{a} \end{bmatrix}, \end{aligned} \tag{20}$$

where θ is chosen so that $\bar{a}_k = 0$. If the angle of θ is such that $\tan(\theta)$ is a power of 2, the multiplication can be done using only bit-shift operations. A general angle can be constructed as a series of such angles with the tangent value equal to the power of 2, and in practice the sum can be approximated with i_{max} values [158]

$$\theta = \sum_{i=0}^{\infty} \rho_i \theta_i \approx \sum_{i=0}^{i_{max}} \rho_i \theta_i, \tag{21}$$

where $\rho_i = \{-1, +1\}$ and θ_i is constrained so that $\tan(\theta_i) = 2^{-i}$.

The rotation in (20) is accomplished in a multistage manner by a series of micro rotations. The micro rotations result in a series of intermediate results. The CORDIC implementation with i_{max} stages results from (20) as [158]

$$\begin{bmatrix}\mathbf{r}^{[0]}\\ \mathbf{a}^{[0]}\end{bmatrix} = \kappa \begin{bmatrix}\mathbf{r}\\ \mathbf{a}\end{bmatrix}, \tag{22}$$

$$\begin{bmatrix}\mathbf{r}^{[1]}\\ \mathbf{a}^{[1]}\end{bmatrix} = \begin{bmatrix}\mathbf{r}^{[0]}\\ \mathbf{a}^{[0]}\end{bmatrix} + \rho_0 2^0 \begin{bmatrix}-\mathbf{a}^{[0]}\\ \mathbf{r}^{[0]}\end{bmatrix}, \tag{23}$$

$$\vdots$$

$$\begin{bmatrix}\bar{\mathbf{r}}\\ \bar{\mathbf{a}}\end{bmatrix} = \begin{bmatrix}\mathbf{r}^{[i_{max}]}\\ \mathbf{a}^{[i_{max}]}\end{bmatrix} + \rho_{i_{max}} 2^{-i_{max}} \begin{bmatrix}-\mathbf{a}^{[i_{max}]}\\ \mathbf{r}^{[i_{max}]}\end{bmatrix}, \tag{24}$$

where $\kappa = \prod_{i=0}^{i_{max}} \cos(\theta)_i$ is a precomputed normalization constant and the sign of the micro rotation is determined by $\rho_i = \mathrm{sgn}(r_k^{[i-1]})\mathrm{sgn}(a_k^{[i-1]})$.

The case of complex input data requires that the leading elements of two processed rows are made real. Thus, the typical step of the Givens approach can be replaced by a more complicated step involving three sub-steps as [158]

$$\begin{bmatrix}\mathbf{r}'\\ \mathbf{a}'\end{bmatrix} = \begin{bmatrix}e^{-j\phi_r} & 0\\ 0 & e^{-j\phi_a}\end{bmatrix} \begin{bmatrix}\mathbf{r}\\ \mathbf{a}\end{bmatrix}, \tag{25}$$

$$\begin{bmatrix}\bar{\mathbf{r}}\\ \bar{\mathbf{a}}\end{bmatrix} = \begin{bmatrix}\cos(\theta) & \sin(\theta)\\ -\sin(\theta) & \cos(\theta)\end{bmatrix} \begin{bmatrix}\mathbf{r}'\\ \mathbf{a}'\end{bmatrix}, \tag{26}$$

where $\phi_r = \arctan\frac{\mathrm{Im}(r_k)}{\mathrm{Re}(r_k)}$, $\phi_a = \arctan\frac{\mathrm{Im}(a_k)}{\mathrm{Re}(a_k)}$ and $\theta = \arctan\frac{a_k'}{r_k'}$. The combination of four CORDIC elements can be applied to a supercell for complex data [158].

2.4.2 SGR algorithm

The QRD with the SGR algorithm is different from that presented in Section 2.4.1. The decomposition of a symmetric positive definite matrix $\mathbf{A}$ with the SGR algorithm is expressed as [74]

$$\mathbf{A} = \mathbf{Q}_\mathrm{A}\mathbf{D}_\mathrm{R}^{-2}\mathbf{U}, \tag{27}$$

where $\mathbf{U} = \mathbf{D}_\mathrm{R}\mathbf{R} \in \mathbb{C}^{N_\mathrm{R}\times N_\mathrm{R}}$ is a upper triangular matrix, $\mathbf{D}_\mathrm{R} = \mathrm{diag}(\mathbf{R}) \in \mathbb{R}^{N_\mathrm{R}\times N_\mathrm{R}}$, $\mathbf{Q}_\mathrm{A} = \mathbf{Q}\mathbf{D}_\mathrm{R} \in \mathbb{C}^{N_\mathrm{R}\times N_\mathrm{R}}$. Matrix $\mathbf{Q}_\mathrm{A}$ consists of the orthogonalized columns of the matrix $\mathbf{A}$.

Now (17) can be written as [74]

$$\begin{aligned}\mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{A}\mathbf{X} &= \mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{B}\\ \mathbf{D}_{\mathrm{R}}\mathbf{Q}^{\mathrm{H}}\mathbf{Q}\mathbf{D}_{\mathrm{R}}\mathbf{D}_{\mathrm{R}}^{-2}\mathbf{U}\mathbf{X} &= \mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{B}\\ \mathbf{U}\mathbf{X} &= \mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{B}\\ \mathbf{X} &= \mathbf{U}^{-1}\mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{B}.\end{aligned} \tag{28}$$

The SGR algorithm is used to determine $\mathbf{Q}_{\mathrm{A}}$ and $\mathbf{U}$ from $\mathbf{A}$ as in (27). The annihilation is done for one element at a time from appropriate pairs of rows as in (20). In the SGR algorithm, the selected pairs of rows $\mathbf{a}$ and $\mathbf{r}$ are first scaled as [74]

$$\begin{aligned}\mathbf{u} &= r_k\mathbf{r}\\ \mathbf{a} &= w^{\frac{1}{2}}\mathbf{v},\end{aligned} \tag{29}$$

where r_k is the kth element of $\mathbf{r}$ and given scalar $w > 0$. With the scaling in (29) only half of the multiplications and no square roots are required in the annihilation of a_k compared to normal Givens rotations [74]. The rotation performed by the SGR algorithm is now

$$\begin{bmatrix}\bar{\mathbf{u}}\\ \bar{\mathbf{v}}\end{bmatrix} = \begin{bmatrix}1 & wv_k\\ -\frac{v_k}{u_k} & 1\end{bmatrix}\begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix}, \tag{30}$$

and $\bar{w} = wu_k/\bar{u}_k$. The relationship to (20) holds with representations

$$\begin{aligned}\bar{\mathbf{u}} &= \bar{r}_k\bar{\mathbf{r}}\\ \bar{\mathbf{a}} &= \bar{w}^{\frac{1}{2}}\bar{\mathbf{v}}.\end{aligned} \tag{31}$$

An upper triangular matrix $\mathbf{U}$, i.e., $\mathbf{U} = \mathbf{D}_{\mathrm{R}}\mathbf{R} = \mathrm{diag}(\mathbf{R})\mathbf{R} \in \mathbb{C}^{N_{\mathrm{R}}\times N_{\mathrm{R}}}$, is formed in the end of the annihilation process of the matrix $\mathbf{A} \in \mathbb{C}^{N_{\mathrm{R}}\times N_{\mathrm{R}}}$ [74].

The desired coefficient matrix $\mathbf{X}$ in (28) is determined by calculating the inverse of matrix $\mathbf{U}$ and by multiplying with $\mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}\mathbf{B}$. The inversion of the upper triangular matrix $\mathbf{U}$ is performed using an algorithm listed in Algorithm 1 [157]. It should be noted that inversion of the upper triangular matrix $\mathbf{U}$ could also be calculated by a back substitution algorithm. However, the algorithm listed in Algorithm 1 is less complex in number of required operations [157, 159].

2.4.3 LLR calculation

The LMMSE gives a soft output estimate $\tilde{\mathbf{x}}$ of the transmitted symbol vector $\mathbf{x}$ as an output. The LMMSE detector aims to compensate for the effect of the channel, but in

Algorithm 1 Inversion of triangular matrix [157]

```
if $i = j$ then
    $\mathbf{U}_{ij}^{-1} = \frac{1}{\mathbf{U}_{jj}}$
else if $i < j$ then
    $\mathbf{U}_{ij}^{-1} = -\frac{1}{\mathbf{U}_{jj}} \sum_{m=1}^{j-1} \mathbf{U}_{im}^{-1} \mathbf{U}_{mj}$
else if $i > j$ then
    $\mathbf{U}_{ij}^{-1} = 0$
end if
```

practice, the compensation is never perfect. Thus, the effect of the residual channel $\tilde{\mathbf{H}}$, which is defined below, has to be taken into account in the calculation of $L_{\mathrm{E}}(b_k|\tilde{x}_l)$. The residual channel matrix is calculated as [160]

$$\tilde{\mathbf{H}} = \mathbf{W}^{\mathrm{H}}\mathbf{H}, \tag{32}$$

and the corresponding interference plus noise covariance matrix as

$$\tilde{\mathbf{R}}_{nn} = \tilde{\mathbf{H}} - \tilde{\mathbf{H}}\tilde{\mathbf{H}}^{\mathrm{H}}. \tag{33}$$

The $L_{\mathrm{E}}(b_k|\tilde{x}_l)$ of the LMMSE detector output can then be calculated as

$$\begin{aligned} L_{\mathrm{E}}(b_k|\tilde{x}_l) &= \ln \frac{\sum_{x_l \in \Omega_{k,+1}} \exp(\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}})}{\sum_{x_l \in \Omega_{k,-1}} \exp(\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}})} && (34) \\ &= \ln \sum_{x_l \in \Omega_{k,+1}} \exp(\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}}) - \ln \sum_{x_l \in \Omega_{k,-1}} \exp(\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}}) && (35) \\ &\approx \max_{x_l \in \Omega_{k,+1}} (\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}}) - \max_{x_l \in \Omega_{k,-1}} (\frac{||\tilde{x}_l - \tilde{H}_{l,l}x_l||}{[\tilde{\mathbf{R}}_{nn}]_{l,l}}), && (36) \end{aligned}$$

where (35) and (36) are the log-MAP and max-log-MAP solutions from the LMMSE soft-output, respectively.

2.4.4 Implementation properties

The required matrix inversion and QRD of the matrix **A** in (17) is the most problematic part of the LMMSE solution in terms of numerical properties. The applied QRD via Givens rotations is shown to offer very favorable roundoff properties with fixed-point implementation [70].

The CORDIC algorithm [73] is a numerically stable method for the calculation of the QRD. The Givens rotation is accomplished with the CORDIC algorithm via simple shift and add operations, and results in an approximation of the rotation, where the required accuracy is obtained by choosing the proper number of iterations and fixed-point word lengths for the coefficients. Furthermore, the CORDIC algorithm does not severely affect the dynamic range of the rotated samples. The effects of the angle approximation error and the rounding error on the accuracy of the CORDIC computation have been analyzed in [158].

The SGR [74] algorithm calculates an exact QRD result as presented in section 2.4.2. However, the algorithm is proposed to be used with floating point arithmetic [74] due to scaling of the coefficients in (29) and (30). Thus, a large dynamic range is required for the coefficient representation with fixed-point implementation. Especially the calculation of $-\frac{v_k}{u_k}$ in (30) is crucial to the numerical accuracy [74]. The problem can be compensated for by introducing adaptive scaling of the variables in the calculation.

The back substitution algorithm and triangular matrix inversion algorithm have been shown to be numerically stable with fixed-point implementation [70, 157].

2.5 List sphere detector

The sphere detector algorithms achieve the ML solution in (7) with a reduced number of considered candidate symbol vectors in the search. When we assume a real equivalent signal model, this is done by limiting the search to points that lie inside a M_{R}-dimensional hyper-sphere $S(\mathbf{y}, \sqrt{C_0})$ centered at $\mathbf{y}$. After QR decomposition (QRD) of the channel matrix $\mathbf{H}$, the condition can be written as [91]

$$||\mathbf{y} - \mathbf{QRx}||_2^2 \leq C_0 \tag{37}$$

$$||\mathbf{Q}^{\mathrm{T}}\mathbf{y} - \mathbf{Rx}||_2^2 \leq C_0 \tag{38}$$

$$||\tilde{\mathbf{y}} - \mathbf{Rx}||_2^2 \leq C_0, \tag{39}$$

where $\mathbf{R} \in \mathbb{R}^{M_{\mathrm{R}} \times M_{\mathrm{T}}}$ is an upper triangular matrix with positive diagonal elements, $\mathbf{Q} \in \mathbb{R}^{M_{\mathrm{R}} \times M_{\mathrm{R}}}$ is an orthogonal matrix, $\tilde{\mathbf{y}} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$, and C_0 is the squared radius of the sphere. Due to the upper triangular form of $\mathbf{R}$ the values of $\mathbf{x}$ can be solved from (39) level by level using the back-substitution algorithm. Let $\mathbf{x}_i^{M_{\mathrm{T}}} = (x_i, x_{i+1}, \ldots, x_{M_{\mathrm{T}}})^{\mathrm{T}}$ denote the last $M_{\mathrm{T}} - i + 1$ components of the vector $\mathbf{x}$. The sphere search can be illustrated with a tree structure as shown in Figure 2, where the algorithm aims at finding the shortest

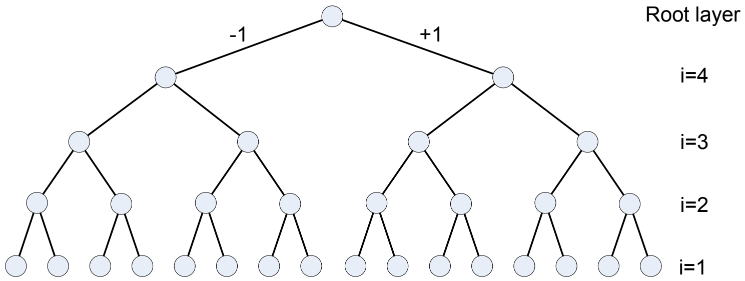


Fig 2. A tree structure of a sphere detector with real signal model, 2×2 antenna system and 4-QAM.

path between the root layer and the bottom layer. First, the last elements of the possible symbol vectors are calculated, i.e, x_{M_T} and then $x_{M_\mathrm{T}-1}$ and so on. The squared partial ED (PED) of $\mathbf{x}_i^{M_\mathrm{T}}$ can be calculated as [117]

$$\begin{aligned} d(\mathbf{x}_i^{M_\mathrm{T}}) &= d(\mathbf{x}_{i+1}^{M_\mathrm{T}}) + |\tilde{y}_i - \sum_{j=i}^{M_\mathrm{T}} R_{i,j} x_j|^2 \\ &= d(\mathbf{x}_{i+1}^{M_\mathrm{T}}) + |b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}}) - R_{i,i} x_i|^2, \end{aligned} \tag{40}$$

where $d(\mathbf{x}_{M_\mathrm{T}}^{M_\mathrm{T}}) = 0$, $b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}}) = \tilde{y}_i - \sum_{j=i+1}^{M_\mathrm{T}} R_{i,j} x_j$, $R_{i,j}$ is the (i, j)th term of $\mathbf{R}$ and $i = M_\mathrm{T}, \ldots, 1$. Depending on the search strategy and the channel realization, the SD searches a variable number of nodes in the tree structure, and aims to find the point $\mathbf{x} = \mathbf{x}_1^{M_\mathrm{T}}$, also called a leaf node, for which the ED $d(\mathbf{x}_1^{M_\mathrm{T}})$ is minimum. The pruning order of the possible nodes, which are often called leaves, at each layer is defined based on the enumeration method. We assume the use of the Schnorr-Euchner enumeration (SEE) [97, 161], which determines the pruning order according to the Euclidean distance. The SEE middle point with real presentation can be solved as follows [91]

$$\begin{aligned} S_i(\mathbf{x}_{i+1}^{M_\mathrm{T}}) &= \left\lfloor \frac{1}{R_{i,i}} \left(\tilde{y}_i - \sum_{j=i+1}^{M_\mathrm{T}} R_{i,j} x_j \right) \right\rceil \\ &= \left\lfloor \frac{b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}})}{R_{i,i}} \right\rceil, \end{aligned} \tag{41}$$

where $\lfloor \cdot \rceil$ denotes rounding to the closest integer. The pruning order can then be determined based on knowledge of the real symbol alphabet [91] and the previously studied nodes. The pruning order can also be determined by calculating (40) with all possible symbols x_i and sorting the results in order, which might be less complex to implement with low order constellations.

The hard ML solution that is given as an output by the SD may cause significant performance degradation compared to the optimal soft output MAP detection in a communication system with FEC. The more appropriate detector is the list sphere detector (LSD) [4] that can be used for obtaining a list of candidate symbol vectors and the corresponding EDs $\mathscr{L} \in \mathbb{Z}^{N_{\text{cand}} \times N_{\text{T}}}$ as an output, where N_{cand} is the size of the candidate list so that $1 \leq N_{\text{cand}} \leq 2^{QN_{\text{T}}}$. The output candidate list can then be used to approximate the MAP solution $L_{\text{D}}(b_k)$. A high level architecture of the list sphere detector structure is shown in Figure 3. The preprocessing unit decomposes the channel matrix $\mathbf{H}$ into an upper triangular matrix $\mathbf{R}$ and an orthogonal matrix $\mathbf{Q}$, which are given as an input with received signal $\mathbf{y}$ to the LSD algorithm. The LSD algorithm unit executes the tree search and gives the candidate list $\mathscr{L}$ as an output. The approximation of $L_{\text{D}}(b_k)$ is calculated in the log-likelihood ratio (LLR) calculation unit using the given candidate list.

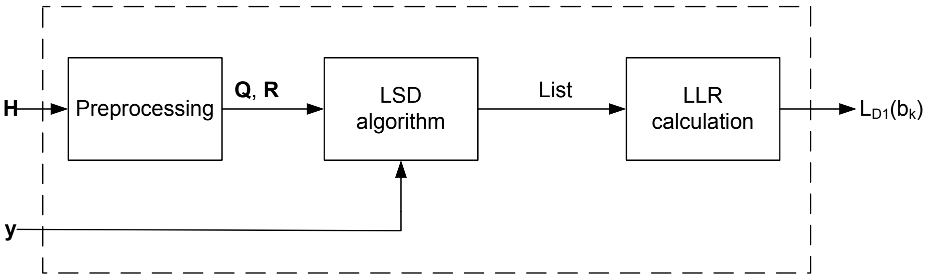


Fig 3. A high level architecture of a list sphere detector.

2.5.1 Preprocessing algorithms

The preprocessing unit is used to decompose the channel matrix $\mathbf{H}$ into an upper triangular form as in (39), which enables the symbol-by-symbol tree search with a back substitution algorithm. Typically QRD is assumed in the literature to perform the channel matrix decomposition into an upper triangular matrix $\mathbf{R}$ and an orthogonal matrix $\mathbf{Q}$, which are given as an input with received signal $\mathbf{y}$ to the LSD algorithm. However, it has been shown that the complexity of the sphere detection algorithm search can be decreased by applying different more sophisticated ordering or preprocessing approaches before the SD algorithm [48, 91, 111, 162]. The preprocessing of the channel matrix has to be recalculated as the channel changes, i.e., it is relative to the channel coherence time. Thus, the complexity reduction of the sphere detector algorithm is obtained with much less effort as the sphere detector algorithm operates at symbol rate, which is typically much higher than the channel coherence time. Obviously one would also think that the complexity of the LSD algorithms can be decreased by similar approaches. We consider two ordering methods for the channel matrix $\mathbf{H}$.

Column ordering based on Euclidean norm

Column ordering according to the Euclidean norm has been proposed for sphere detection, e.g., in [90, 91, 162]. In this method the channel matrix columns $\mathbf{h}_i$ are ordered in a descending order according to the Euclidean norm $||\mathbf{h}_i||$ before the QRD, i.e., the signal from transmit antenna i with the strongest channel gain $||\mathbf{h}_i||$ is ordered to be at the root layer of the search tree. This typically decreases the complexity of the sphere search as the strongest signal decisions are made at the beginning of the tree traversal. The preprocessing algorithm gives the matrices $\mathbf{R}$ and $\mathbf{Q}$, and a vector $\mathbf{p}$ as an output, where $\mathbf{p}$ denotes the permutation order of the columns of $\mathbf{H}$.

Sorted QRD algorithm

The sorted QRD (SQRD) algorithm [163] is an extension to the modified Gram-Schmidt procedure by reordering the columns of the channel matrix prior to each orthogonalization step. The algorithm jointly calculates a very close to optimized detection order, which is achieved by the V-BLAST detection algorithm [39], and the QRD of the channel matrix. This means that the absolute values of the diagonal elements $|R_{i,i}|$ of the resulting upper triangular matrix $\mathbf{R}$ are minimized in the process of calculating the QRD. Thus, the strongest layer is located at the root layer of the search tree. The SQRD algorithm is listed as Algorithm 2 [163].

Algorithm 2 $[\mathbf{Q},\mathbf{R},\mathbf{p}]$ = SQRD($\mathbf{H}$)

```
Initialize matrices $\mathbf{R}=\mathbf{0},\mathbf{Q}=\mathbf{H},\mathbf{p}=(1,\dots,M_{\mathrm{T}})$
for $i=1$ to $M_{\mathrm{T}}$ do
    $\mathbf{norm}_i=\|\mathbf{q}_i\|^2$
end for
for $i=1$ to $M_{\mathrm{T}}$ do
    $k_i=\arg\min_{l=i,\dots,M_{\mathrm{T}}}\mathbf{norm}_l$
    Exchange columns $i$ and $k_i$ in $\mathbf{R},\mathbf{p},\mathbf{norm}$ and in the first $M_{\mathrm{R}}+i-1$ rows of $\mathbf{Q}$
    $R_{i,i}=\sqrt{\mathbf{norm}_i}$
    $\mathbf{q}_i:=\mathbf{q}_i/R_{i,i}$
    for $k=i+1$ to $M_{\mathrm{T}}$ do
        $R_{i,k}=\mathbf{q}_i^{\mathrm{H}}\mathbf{q}_k$
        $\mathbf{q}_k:=\mathbf{q}_k-R_{i,k}\mathbf{q}_i$
        $\mathbf{norm}_k:=\mathbf{norm}_k-R_{i,k}^2$
    end for
end for
```

2.5.2 Tree search algorithms

The tree search algorithms in the literature are often divided into three categories according to the search strategy: the breadth-first, the depth-first, and the metric-first [98]. We introduce a LSD algorithm based on each one of those different search strategies in the next subsections.

K-best-LSD algorithm

The K-best-LSD algorithm listed as Algorithm 3 is a modification from the K-best-SD algorithm [100] to the LSD algorithm. The algorithm is based on the breadth-first strategy, i.e., the search proceeds one layer at a time in the search tree by extending the partial candidates $\mathbf{s}$ with admissible nodes and calculating the PEDs $d(\mathbf{s})$. Then at most the $K=N_{\mathrm{cand}}$ partial candidates with the lowest PED are selected for the next level. The number of nodes studied can be limited by setting a predefined sphere radius C_0 and discarding the candidates with high PED, which have a minor effect on soft output LLR. The partial candidate information $\mathcal{N}(\mathbf{s},d(\mathbf{s}))$, which includes the the partial candidate at layer i, $\mathbf{s} = \mathbf{x}_i^{M_\mathrm{T}}$ and the corresponding PED $d(\mathbf{s})$, is stored into memory sets $\mathscr{L}$ and $\mathscr{S}$, and the kth element in the set $\mathscr{L}$ is described as $\{\mathscr{L}\}^k$. The search is continued in a similar fashion by extending the stored partial candidates with admissible nodes until all the layers have been checked. If more than K partial candidates are stored in set $\mathscr{S}$, a sorting operation is required to determine the K best candidates for the next layer. The K-best-LSD algorithm search goes through a fixed number of nodes in the tree structure if no enumeration method is introduced. However, it should be noted that the output candidate list $\mathscr{L}$ of the algorithm does not necessarily contain the most probable candidates with the lowest EDs, which may result in inaccurate likelihood information approximation and performance loss.

Algorithm 3 [$\mathscr{L}$] = K-best-LSD($\tilde{\mathbf{y}}, \mathbf{R}, C_0, K, \Omega_\mathrm{r}, M_\mathrm{T}$)

```
Initialize set $\{\mathscr{L}\}^0$ with $\mathscr{N}(\mathbf{s}_0 = \mathbf{x}_{N_\mathrm{T}}^{N_\mathrm{T}}, d(\mathbf{s}) = 0)$ and empty set $\mathscr{S}$
for Layer $i = M_\mathrm{T}$ to 1 do
    for $k = 0$ to $|\mathscr{L}| - 1$ do
        Remove $\mathscr{N}(\mathbf{s} = \mathbf{x}_{i+1}^{M_\mathrm{T}}, d(\mathbf{s}))$ from $\{\mathscr{L}\}^k$
        for $j = 1$ to $|\Omega_\mathrm{r}|$ do
            Determine $\mathbf{s}_\mathrm{c} = (x_i, \mathbf{s})^\mathrm{T}$, where $x_i = \{\Omega_\mathrm{r}\}_j$ and calculate $d(\mathbf{s}_\mathrm{c})$
            if $d(\mathbf{s}_\mathrm{c}) < C_0$ then
                Store $\mathscr{N}_\mathrm{c}(\mathbf{s}_\mathrm{c}, d(\mathbf{s}_\mathrm{c}))$ to $\mathscr{S}$
            end if
        end for
    end for
    Sort $\mathscr{S}$ according to the PED if $|\mathscr{S}| > K$
    Move $K$ candidates with smallest PED from $\mathscr{S}$ to $\mathscr{L}$ and empty $\mathscr{S}$
end for
```

SEE-LSD algorithm

The Schnorr-Euchner enumeration (SEE) - LSD is a depth-first search based algorithm and it is listed as Algorithm 4. It is an extension of SEE-SD [97] to a list sphere detector, and the algorithm continues the search until all admissible nodes have been checked and the required candidate list $\mathscr{L}$ has been obtained. The output candidate list $\mathscr{L}$ includes the most probable candidates, i.e., the candidates with the lowest ED. The main difference between the SEE-LSD and the SEE-SD algorithm is that the sphere radius C_0 is not updated until the final list is full, and the C_0 is equal to the candidate with the largest ED in the final candidate list $\mathscr{L}$. The search is continued until all the admissible nodes have been searched.

The sequential algorithm initially starts from the root layer and extends the partial candidate $\mathbf{s} = \mathbf{x}_{M_\mathrm{T}}^{M_\mathrm{T}}$ with the best admissible node determined by the SE enumeration. The search tree pruning loop in the algorithm extends the considered partial candidate $\mathbf{s} = \mathbf{x}_{i+1}^{M_\mathrm{T}}$ with the next best available child node in each iteration until the PED of the extended partial candidate exceeds the sphere radius C_0 or a leaf node $\mathbf{s} = \mathbf{x}_1^{M_\mathrm{T}}$ is found. In the case of a leaf node $\mathbf{s} = \mathbf{x}_1^{M_\mathrm{T}}$, the candidate information $\mathscr{N}(\mathbf{s}, d(\mathbf{s}))$, which includes the candidate $\mathbf{s}$ and the corresponding ED $d(\mathbf{s})$, is added to the final candidate list $\mathscr{L}$ if the ED $d(\mathbf{s})$ is lower than the current sphere radius C_0. The radius is always updated to be equal to the highest ED in the final list when the final candidate list is full and a new leaf node is found. If the extended candidate exceeds the C_0 or all the admissible nodes have been checked, the algorithm moves one layer higher and continues with the next best admissible node. The next best admissible node is determined based on the previously extended nodes.

Algorithm 4 [$\mathscr{L}$] = SEE-LSD($\tilde{\mathbf{y}}, \mathbf{R}, N_{\mathrm{cand}}, \Omega_{\mathrm{r}}, M_{\mathrm{T}}$)

```
Initialize set $\mathscr{L}$, and set $C_0 = \infty$, $m = 0$, $n_1 = 1$, $i = M_{\mathrm{T}}$
Initialize $\mathscr{N}(\mathbf{s} = \mathbf{x}_{M_{\mathrm{T}}}^{M_{\mathrm{T}}}, d(\mathbf{s}) = 0)$
while ($i \neq M_{\mathrm{T}}$ and $n_1 \neq |\Omega_{\mathrm{r}}|$) do
  if $n_1 = |\Omega_{\mathrm{r}}|$ then
    Set $i = i + 1$, determine $n_1$ and continue with $\mathscr{N}(\mathbf{s} = \mathbf{x}_{i+2}^{M_{\mathrm{T}}}, d(\mathbf{s}))$
  else
    Determine the $n_1$th best node $x_i$ for $\mathbf{s}_{\mathrm{c}} = (x_i, \mathbf{x}_{i+1}^{M_{\mathrm{T}}})^{\mathrm{T}}$ and calculate $d(\mathbf{s}_{\mathrm{c}})$
    if $d(\mathbf{s}_c) < C_0$ then
      if $\mathbf{s}_c$ is a leaf node, i.e., $i = 1$ then
        Store $\mathscr{N}_{\mathrm{F}}(\mathbf{s}_{\mathrm{c}}, d(\mathbf{s}_{\mathrm{c}}))$ in $\{\mathscr{L}\}^m$
        Set $m = m + 1$ or, if $\mathscr{L}$ is full, set $m$ according to $\{\mathscr{L}\}^m$ with max ED and $C_0 = \{d(\mathbf{s})\}^m$
        Continue with $\mathscr{N}(\mathbf{s} = \mathbf{x}_{i+1}^{M_{\mathrm{T}}}, d(\mathbf{s}))$, $n_1 + +$ and $i = 1$ if $n_1 + 1 \leq |\Omega_{\mathrm{r}}|$
      else if $i \neq 1$ or $n_1 + 1 = |\Omega_{\mathrm{r}}|$ then
        Set $i = i - 1$ and $n_1 = 1$, and continue with $\mathscr{N}(\mathbf{s}_{\mathrm{c}}, d(\mathbf{s}_{\mathrm{c}}))$
      end if
    else if $d(\mathbf{s}) \geq C_0$ and $i \neq M_{\mathrm{T}} - 1$ then
      Set $i = i + 1$, determine $n_1$ and continue with $\mathscr{N}(\mathbf{s} = \mathbf{x}_{i+2}^{M_{\mathrm{T}}}, d(\mathbf{s}))$
    else
      End the algorithm
    end if
  end if
end while
```

IR-LSD algorithm

The increasing radius (IR) - LSD is listed as Algorithm 5. The IR-LSD algorithm uses the metric first search strategy and it is a modification of Dijkstra's algorithm [101] to a LSD algorithm. The algorithm is optimal in the sense of the number of nodes in the tree structure visited [95, 101]. The output candidate list $\mathscr{L}$ includes the most probable candidates, i.e., the algorithm always gives exactly the same output as the SEE-LSD algorithm. The algorithm uses the so-called metric-first search [95], where the algorithm always extends the partial candidate with the lowest PED in one extend loop.

The algorithm operates in a sequential fashion; it initially starts from the root layer with partial candidate $\mathbf{s} = \mathbf{x}_{M_\mathrm{T}}^{M_\mathrm{T}}$, and determines the next best admissible node x_i at layer i with SEE. The child candidate is then defined as $\mathbf{s}_c = (x_i, \mathbf{x}_{i+1}^{M_\mathrm{T}})^\mathrm{T}$. The algorithm also, if possible, extends the father candidate $\mathbf{s}_f = \mathbf{x}_{i+2}^{M_\mathrm{T}}$ with the next best admissible node x_{i+1}. The SEE, which is used to determine the next best admissible node, requires the information of already extended nodes, and the information is defined as n_1 and n_2 for the considered candidate and its father candidate, respectively. The algorithm uses two memory sets for storing the candidates, the final candidate set $\mathscr{L}$ and the partial candidate set $\mathscr{S}$. In the algorithm search, the partial child candidate information $\mathscr{N}_\mathscr{S}\left(\mathbf{s}_c, d(\mathbf{s}_c), n_1\right)$ and the possible father candidate information $\mathscr{N}_\mathscr{S}\left(\mathbf{s}_f, d(\mathbf{s}_f), n_2\right)$ are stored to set $\mathscr{S}$ after each tree pruning loop. In the case the child candidate $\mathbf{s}_c$ is a leaf node and smaller than the current radius C_0, the candidate information $\mathscr{N}_\mathscr{L}\left(\mathbf{s}_c, d(\mathbf{s}_c)\right)$ is stored to the final list set $\mathscr{L}$. The sphere radius C_0 is updated when $\mathscr{L}$ is full and the candidate with the largest ED is replaced with a new leaf candidate. After storing the candidate(s), the algorithm finds the candidate information $\mathscr{N}_\mathscr{S}$ with the minimum PED $d(\mathbf{s})$ from set $\mathscr{S}$ and continues the algorithm if the PED is smaller than the current radius C_0. It should also be noted that $n_1 = 0$ is used in the tree pruning loop if the extended node is not a leaf node, and the n, which is read from $\mathscr{S}$, is used as n_2.

Algorithm 5 [$\mathscr{L}$] = IR-LSD($\tilde{\mathbf{y}},\mathbf{R},N_{\mathrm{cand}},\Omega,M_{\mathrm{T}}$)

```
Initialize sets $\mathscr{S}$ and $\mathscr{L}$, and set $C_0=\infty$, $m=0$, $n_1=1$
Initialize $\mathscr{N}(\mathbf{s}=\mathbf{x}_{M_{\mathrm{T}}}^{M_{\mathrm{T}}},d(\mathbf{s})=0,n_2=2,i=M_{\mathrm{T}})$
while $C_0<d(\mathbf{s})$ do
    Determine the $n_1$th best node $x_i$ for $\mathbf{s}_{\mathrm{c}}=(x_i,\mathbf{x}_{i+1}^{M_{\mathrm{T}}})^{\mathrm{T}}$ and calculate $d(\mathbf{s}_{\mathrm{c}})$
    Determine the $n_2$th best node $x_{i+1}$ for father candidate $\mathbf{s}_{\mathrm{f}}=(x_{i+1},\mathbf{x}_{i+2}^{M_{\mathrm{T}}})^{\mathrm{T}}$ and calculate $d(\mathbf{s}_{\mathrm{f}})$ if $n_2\leq|\Omega_{\mathrm{r}}|$
    if $d(\mathbf{s}_c)<C_0$ then
        if $\mathbf{s}_c$ is a leaf node, i.e., $i=1$ then
            Store $\mathscr{N}_{\mathrm{F}}(\mathbf{s}_{\mathrm{c}},d(\mathbf{s}_{\mathrm{c}}))$ in $\{\mathscr{L}\}^m$
            Set $m=m+1$ or, if $\mathscr{L}$ is full, set $m$ according to $\{\mathscr{L}\}^m$ with max ED and $C_0=\{d(\mathbf{s})\}^m$
            Continue with $\mathscr{N}(\mathbf{s}=\mathbf{x}_{i+1}^{M_{\mathrm{T}}},d(\mathbf{s}),n_1++,1)$ if $n_1+1\leq|\Omega_{\mathrm{r}}|$
        else
            Store $\mathscr{N}_{\mathrm{c}}(\mathbf{s}_{\mathrm{c}},d(\mathbf{s}_{\mathrm{c}}),n_2=2,i--)$ in $\mathscr{S}$
        end if
    end if
    if $\mathscr{N}_{\mathrm{f}}$ calculated and $d(\mathbf{s}_f)<C_0$ then
        Store $\mathscr{N}_{\mathrm{f}}(\mathbf{s}_{\mathrm{f}},d(\mathbf{s})_{\mathrm{f}},n_2++,i)$ in $\mathscr{S}$
    end if
    Continue with $\mathscr{N}$ with min PED from $\mathscr{S}$ and set $n_1=1$
end while
```

2.5.3 LLR calculation

The LSD algorithm output candidate list $\mathscr{L}$ is used to approximate the extrinsic information $L_{\mathrm{E}}(b_k|\mathbf{y})$ in the log-MAP solution (9). The approximation can be calculated either for the log-MAP or max-log-MAP solution as

$$
\begin{aligned}
L_{\mathrm{E}}(b_k|\mathbf{y}) &= \ln\sum_{\mathbf{x}\in\mathscr{L}_{k,+1}}\exp(\frac{-d(\mathbf{x})}{2\sigma^2})-\ln\sum_{\mathbf{x}\in\mathscr{L}_{k,-1}}\exp(\frac{-d(\mathbf{x})}{2\sigma^2}) \\
&\approx \max_{\mathbf{x}\in\mathscr{L}_{k,+1}}(\frac{-d(\mathbf{x})}{2\sigma^2})-\max_{\mathbf{x}\in\mathscr{L}_{k,-1}}(\frac{-d(\mathbf{x})}{2\sigma^2}),
\end{aligned}
\tag{42}
$$

where $\mathscr{L}_{k,+1}=\{\mathbf{x}|b_k=+1\}$ is the set of candidate vectors $\mathbf{x}$ having $b_k=+1$. The performance of the LSD may suffer due to too small a list size and, thus, inaccurate $L_{\mathrm{D}}(b_k)$

values. If the LSD algorithm output candidate list size N_{cand} is large enough, the impact of the unknown results on the approximation is likely to be relatively small, and the approximation of $L_{\text{D}}(b_k)$ is accurate enough for adequate performance. Depending on the list size, N_{cand}, it provides a tradeoff between the performance and the computational complexity. We will also present a method to compensate for the effect of unreliable LLRs in Section 3.1.

2.6 Numerical examples

The performance of different detector algorithms is studied via computer simulations. A turbo coded MIMO–OFDM system was assumed with QAM constellations, and $N_{\text{c}} = 512$ with $N_{\text{used}} = 300$ used data subcarriers. A bit-interleaved coded modulation (BICM) with $1/2$ rate $[13_o, 15_o]$ turbo code was applied in an uncorrelated (UNC) and highly correlated (CORR) 6 tap typical urban (TU) channel with a user velocity of 120 kmph. The CORR channel spatial correlation matrices $\mathbf{R}_{\text{TX}}$ at the transmitter and $\mathbf{R}_{\text{RX}}$ at the receiver are presented as

$$\mathbf{R}_{\text{RX}} = \begin{bmatrix} 1.00 & -0.69+0.34i & 0.49-0.36i & -0.38+0.35i \\ -0.69-0.34i & 1.00 & -0.69+0.34i & 0.49-0.36i \\ 0.49+0.36i & -0.69-0.34i & 1.00 & -0.69+0.34i \\ -0.38-0.35i & 0.49+0.36i & -0.69-0.34i & 1.00 \end{bmatrix} \tag{43}$$

$$\mathbf{R}_{\text{TX}} = \begin{bmatrix} 1.00 & 0.80+0.34i & 0.42+0.45i & 0.14+0.39i \\ 0.80-0.34i & 1.00 & 0.80+0.34i & 0.42+0.45i \\ 0.42-0.45i & 0.80-0.34i & 1.00 & 0.80+0.34i \\ 0.14-0.39i & 0.42-0.45i & 0.80-0.34i & 1.00 \end{bmatrix}. \tag{44}$$

The system operates with a 5 MHz bandwidth (BW) at a carrier frequency $f_{\text{c}} = 2.4$ GHz. The LSD algorithm soft output is calculated with a max-log-MAP approximation as in 42 and the soft outputs of the detector algorithms are decoded in an iterative max-log-MAP turbo decoder with 8 iterations. The parameters used in the computer simulations summarized in Table 1 correspond to the parameters proposed in [164] for initial performance evaluation in 3G LTE. The performance examples are presented in throughput vs. SNR γ, where the throughput is determined as the number of correctly received bits in a time interval.

Table 1. Working assumptions for computer simulations.

Parameter	Used configuration
Carrier frequency f_c	2.4 GHz
Bandwidth	5 MHz
Number of subcarriers N_c	512
Used data subcarriers N_{used}	300
Subcarrier spacing	15 kHz
OFDM symbol duration	$71.35\mu s$
FFT duration	$66.67\mu s$
Cyclic prefix duration	$4.69\mu s$
Modulation	$4-$, $16-$ and 64-QAM
Code rate	$1/2$
User velocity	120 kmph
Channel model	Typical Urban (6 tap model)

2.6.1 Comparison of detector algorithms

The performances of the K-best-LSD, the SEE-LSD and the IR-LSD with the real valued signal model and with different list sizes N_{cand} are studied with different system parameters, and compared to ML detector, log-MAP detector, max-log-MAP detector, and LMMSE detector based receivers. The throughput of the 4×4 MIMO–OFDM system with the K-best-LSD based receiver with different list sizes and with 16- and 64-QAM is presented in UNC and CORR channels in Figures 4 and 5, respectively. The K-best-LSD is applied with $C_0 = \infty$. The throughput of the 4×4 MIMO–OFDM system with the SEE-LSD and the IR-LSD based receivers with different list sizes N_{cand} and with 16- and 64-QAM is presented in UNC and CORR channels in Figures 6 and 7, respectively. The outputs of the SEE-LSD and the IR-LSD are exactly the same, i.e., output list $\mathscr{L}$ includes the candidates with lowest ED in the search tree, and thus, LSDs are not separated in the performance examples. The results illustrate the effect of the list size N_{cand} on the quality of the max-log-MAP approximation in (42). The maximum list size $N_{\text{cand}}^{\max}$ that correspond to the optimal log-MAP detection output of the exhaustive search algorithm is $2^{N_T Q}$. Thus, the $N_{\text{cand}}^{\max}$ values with a 2×2 MIMO system with 4-, 16- and 64-QAM constellations are 16, 256, and 4096, respectively. The corresponding maximum list sizes $N_{\text{cand}}^{\max}$ for a 4×4 MIMO system are 256, 65536, and

16777216, respectively. It can be seen that the performance loss due to the max-log-MAP approximation compared to the log-MAP detector is approximately 0.2–0.5 dB and that the max-log-MAP performance can be achieved with the LSD based receivers with a much lower list size than the exhaustive search algorithm with the maximum list size $N_{\text{cand}}^{\text{max}}$. The results also illustrate the differences between the different LSD algorithms and the effect of the channel correlation. The K-best-LSD requires list sizes of $N_{\text{cand}} = 256$ and $N_{\text{cand}} = 512$ in the UNC channel in a 4×4 MIMO system with 16-QAM and 64-QAM, respectively. However, the SEE-LSD or the IR-LSD requires only a list size of $N_{\text{cand}} = 128$ and $N_{\text{cand}} = 256$ in the UNC channel in a 4×4 MIMO system with 16-QAM and 64-QAM, respectively. The difference in performance of different LSDs is due to the search strategy of the algorithms and the fact that the K-best-LSD does not guarantee an output candidate list $\mathscr{L}$ with the lowest ED candidates. It can be also noted that a lower list size N_{cand} is required for the CORR channel compared to the UNC channel to achieve max-log-MAP performance. This is due to the higher probability for larger sets $\mathscr{L}_{k,\pm 1}$ for different bits b_k in the calculation of (42) in the CORR channel, which is discussed in more detail in Section 3.1. The determined list sizes N_{cand}, which achieve close to the max-log-MAP detector performance, are shown in Tables 2 and 3 for the K-best-LSD, SEE- and the IR-LSD, respectively. The performance loss of the hard output ML detector compared to the optimal soft output log-MAP detector is approximately 3–4 dB. Although the performance of the LMMSE detector is adequate in an uncorrelated channel, the LMMSE performance suffers significantly in highly correlated channel realization. The LSD implementation aspects and requirements are discussed in more detail in Chapter 3.

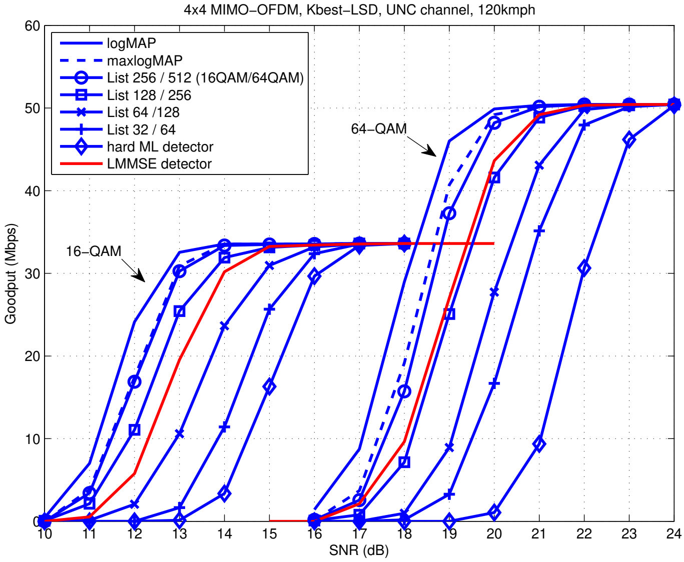


Fig 4. Throughput vs SNR: Performance of the K-best-LSD with different list sizes in a 4×4 antenna system in the UNC channel.

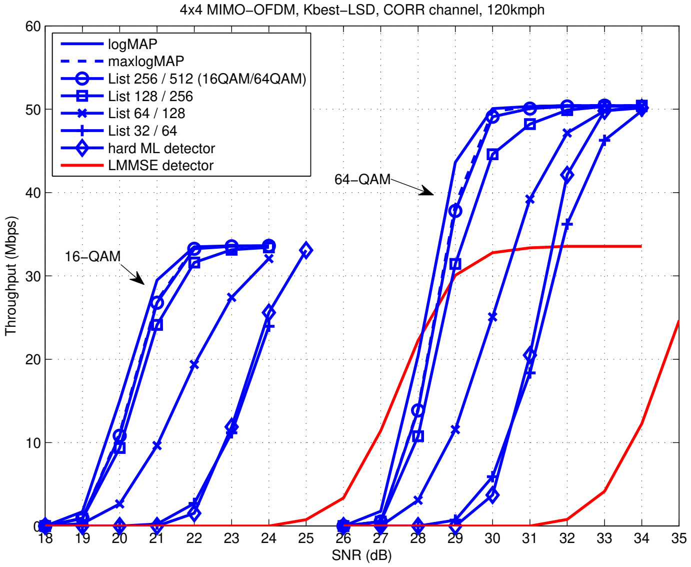


Fig 5. Throughput vs SNR: Performance of the K-best-LSD with different list sizes in a 4×4 antenna system in the CORR channel.

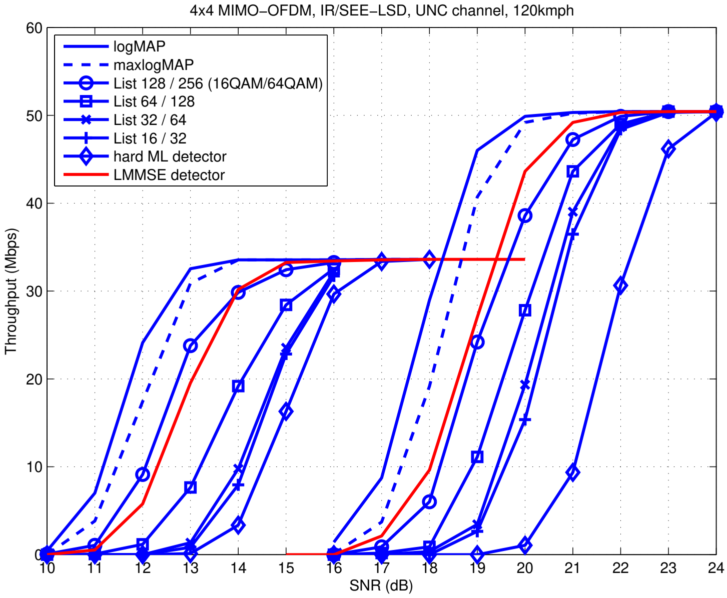


Fig 6. Throughput vs SNR: Performance of the SEE-LSD and IR-LSD with different list sizes in a 4×4 antenna system in the UNC channel.

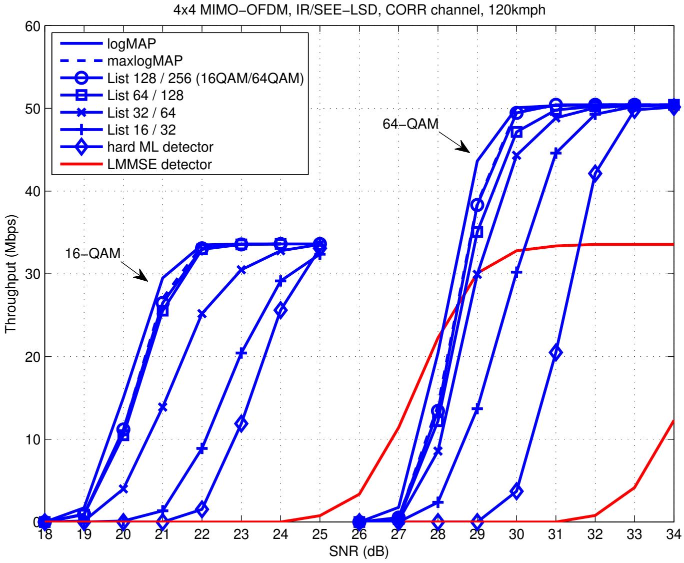


Fig 7. Throughput vs SNR: Performance of the SEE-LSD and IR-LSD with different list sizes in a 4×4 antenna system in the CORR channel.

Table 2. The determined list sizes for K-best-LSD.

	2×2	4×4
4-QAM	$N_{\text{cand}} = 16$	$N_{\text{cand}} = 32$
16-QAM	$N_{\text{cand}} = 128$	$N_{\text{cand}} = 256$
64-QAM	$N_{\text{cand}} = 256$	$N_{\text{cand}} = 512$

Table 3. The determined list sizes for SEE-LSD and IR-LSD.

	2×2	4×4
4-QAM	$N_{\text{cand}} = 16$	$N_{\text{cand}} = 32$
16-QAM	$N_{\text{cand}} = 64$	$N_{\text{cand}} = 128$
64-QAM	$N_{\text{cand}} = 128$	$N_{\text{cand}} = 256$

2.6.2 Comparison of LSD preprocessing methods

We study the effect of LSD preprocessing methods to the LSD based receiver performance. We compare the traditional QRD, column ordering based on Euclidean norm and SQRD methods, which are described in Section 2.5.1, via computer simulations. We also study the effect of advanced preprocessing to the total computational complexity of the LSD.

The performance of K-best-LSD was studied with different preprocessing algorithms and with different list sizes. The number of visited nodes by the K-best-LSD is fixed with given output list size K, and a higher K value results in better performance to certain extend as the LLR approximation gets more accurate. The performance of the K-best-LSD with different list sizes in both UNC and CORR channels is shown in Figure 8. It can be seen that the when the applied list size is high enough in UNC channel, the performance difference between different preprocessing methods is not significant. When the list size is low enough or the channel is highly correlated, the SQRD algorithm with ordering gives approximately 0.2 dB additional gain over the traditional QRD without ordering. The column ordering according to the Euclidean norm, however, actually shows worse performance compared to the other preprocessing methods in the CORR channel. The results indicate that the Euclidean norm of the channel matrix columns is not very good method to determine the detection order of the transmitted layers especially in a correlated channel realization. Also it can be noted that the additional gain by the SQRD algorithm is higher in CORR channel compared to the UNC channel. The number of visited nodes by the sequential search LSD algorithms, the SEE-LSD and the IR-LSD, is a variable that depends on the channel realization.

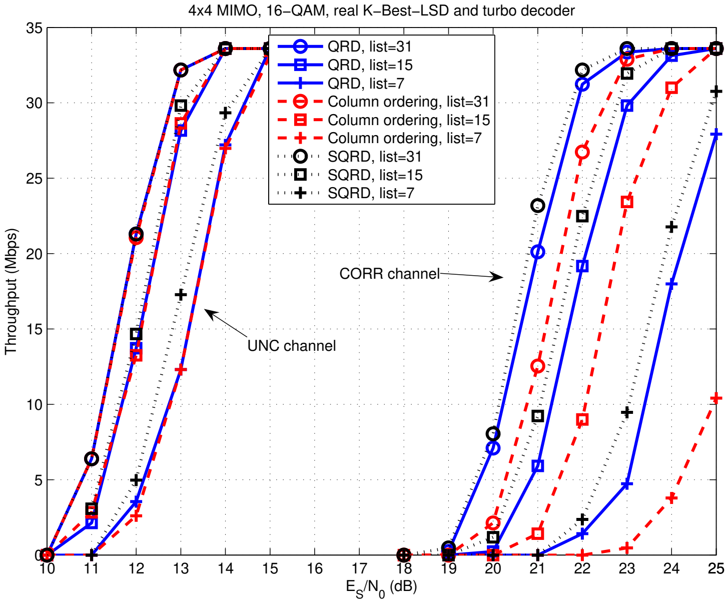


Fig 8. Throughput vs SNR: Performance of the real K-best-LSD with different list sizes and preprocessing methods in 4×4 antenna system with 16-QAM.

The total complexity of the LSD algorithms is relative to the number of visited nodes in the search tree. Thus, we studied the distribution of the number of visited nodes by the LSD algorithms and the performance of the system with limited maximum number of visited nodes. Histograms of the visited nodes by the SEE-LSD and IR-LSD algorithms with different preprocessing methods in UNC and CORR channel are shown in Figures 9 and 10. The average number of visited nodes by the algorithms with different preprocessing methods in both channel scenarios are listed in Table 4. The ratio of visited nodes by the LSD with the column ordering and SQRD preprocessing compared to the traditional QRD preprocessing is shown in brackets. It can be seen that the correlation properties of the channel affect significantly to the number of visited nodes. Figures 9 and 10, and Table 4 show that both the column ordering according to the Euclidean norm and the SQRD decrease the distribution of the number of visited nodes clearly for the UNC channel approximately 20% and 30%, respectively. The results in CORR channel show, similarly as with K-best-LSD, that the column ordering according to the Euclidean norm actually increases the number of visited nodes by the SEE-LSD and the IR-LSD algorithms. The SQRD preprocessing, however, decreases the number of visited nodes approximately 20% on average compared to the QRD.

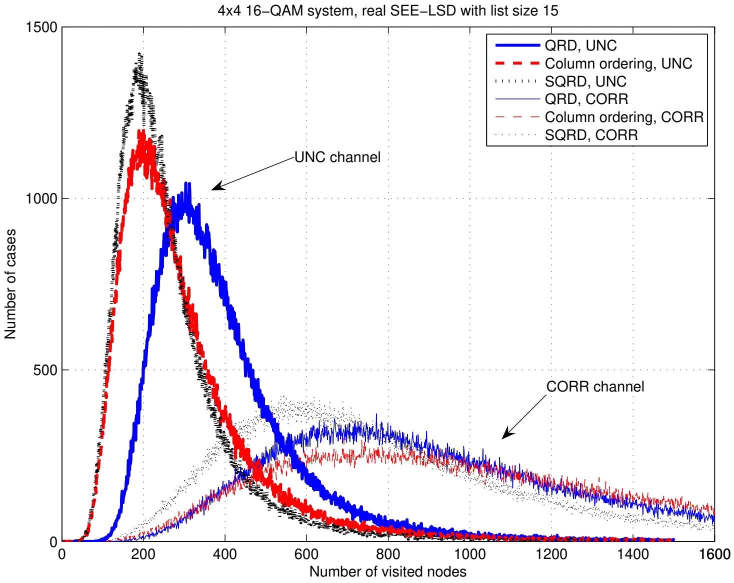


Fig 9. A histogram of visited nodes by the SEE-LSD algorithm with different preprocessing methods.

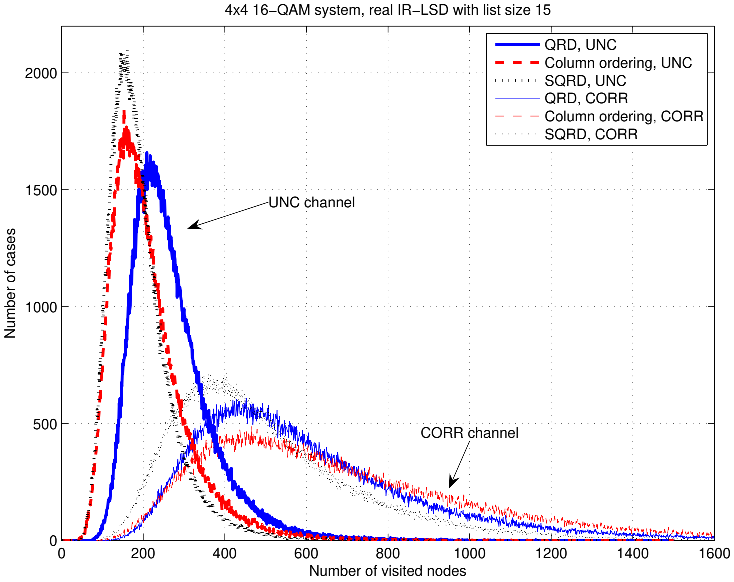


Fig 10. A histogram of visited nodes by the IR-LSD algorithm with different preprocessing methods.

Table 4. The average number of visited nodes by the SEE-LSD† and IR-LSD* algorithms in different channels with different preprocessing methods.

	UNC	CORR
QRD	397† (100%) / 268* (100%)	1001† (100%) / 618* (100%)
Col	302† (76%) / 213* (79%)	1106† (110%) / 705* (114%)
SQRD	269† (68%) / 193* (72%)	821† (82%) / 509* (82%)

The simulations results showed that the SQRD as preprocessing decreases the number of required visited nodes by the LSD algorithms by approximately $20-30\%$. The number of additional multiplication (MUL) and addition (ADD) operations required by the preprocessing algorithms compared to traditional QRD are listed in Table 5. The required number of operations in the PED calculation in (40) are also listed in Table 5 assuming that the average layer in PED calculation is the middle layer of the search tree. The number of additional and saved operations on average for IR-LSD and SEE-LSD algorithms with SQRD preprocessing in a 4×4 MIMO system with 16-QAM are listed in Table 6. It can be seen that a significant number of operations are saved with the SQRD applied as preprocessing.

Table 5. The number of additional real operations due to preprocessing and the PED calculation operations in (40).

	Col	SQRD	PED
MUL	$N_\mathrm{T}N_\mathrm{R}$	$N_\mathrm{T}N_\mathrm{R}+\sum_{j=1}^{N_\mathrm{T}-1} j$	$2N_\mathrm{T}-4+1$
ADD	$N_\mathrm{T}(N_\mathrm{R}-1)$	$N_\mathrm{T}(N_\mathrm{R}-1)+\sum_{j=1}^{N_\mathrm{T}-1} j$	$2N_\mathrm{T}-4+1$

Table 6. The number of added and saved real operations with the IR-LSD and the SEE-LSD with the SQRD in 4×4 MIMO system with 16-QAM and in UNC* and CORR† channels.

	IR-LSD (saved)	SEE-LSD (saved)	SQRD (additional)
MUL	375*/545†	640*/900†	22
ADD	375*/545†	640*/900†	18

2.6.3 Conclusions

The performances of different detector algorithms were studied and compared. The soft output max-log-MAP performance can be achieved with the LSD based receivers with much lower list size than the exhaustive search algorithm with maximum list size $N_{\text{cand}}^{\text{max}}$ and a lower LSD list size N_{cand} is required for the highly correlated channel compared to the uncorrelated channel for similar performance. The K-best-LSD requires higher list size compared to the SEE-LSD or the IR-LSD to achieve the max-log-MAP performance, because the K-best-LSD does not guarantee an output candidate list $\mathscr{L}$ with the lowest ED candidates. The performance loss due to the max-log-MAP approximation compared to the log-MAP detector is approximately 0.2–0.5 dB, and the performance of the hard output ML detector looses approximately 3–4 dB compared to the optimal soft output log-MAP detector. The performance of the LMMSE detector is adequate in uncorrelated channel, but, however, the LMMSE performance suffers significantly in highly correlated channel.

We also studied the effect of the LSD preprocessing methods to the LSD based receiver performance and complexity. The results indicate that the Euclidean norm of the channel matrix columns is not very good method to determine the detection order of the transmitted layers especially in a correlated channel realization. The SQRD algorithm with the K-best-LSD provides some additional gain compared to the traditional QRD algorithm with low LSD list sizes. The SQRD algorithm with the SEE-LSD or the IR-LSD decreases the number of visited nodes approximately 20% on average compared to the QRD with similar performance. Thus, a significant number of operations are saved with the SQRD algorithm applied as preprocessing.

3 LSD implementation trade-offs

In this chapter, we identify and study some key implementation challenges encountered in the implementation of the LSD algorithms in practical wireless systems. We focus on six significant challenges, namely, limiting the dynamic range of the soft output log-likelihood ratio values to the decoder, comparing the real and complex signal models in the LSD, limiting the search complexity of the LSD algorithm, applying the IR-LSD memory sphere radius to lower the required memory access, applying the L1 norm in the LSD algorithm, and analyzing the complexity and performance tradeoffs of an iterative receiver.

3.1 LLR clipping

In the calculation of the soft output LLR values, the exact MAP solution is obtained by calculating the (9) by using all the possible $\Omega^{N_{\mathrm{T}}-1}$ bit vectors $\mathbf{x}$ with both conditional probability variables. The exact calculation of (9) is obviously a very complex task as the number of considered bit vectors $\Omega^{N_{\mathrm{T}}-1}$ increases exponentially with the number of transmit antennas N_{T} and used constellation Ω. The list sphere decoder uses a limited number of elements in the considered sets $\chi_{k,+1} \cap \mathscr{L}$ and $\chi_{k,-1} \cap \mathscr{L}$ by using the LSD output candidate list $\mathscr{L}$ to approximate the likelihood information $L_{\mathrm{D}}(b_k)$ in (12), which decreases the complexity of the calculation of (12) significantly compared to the full set of candidates. The accuracy of the approximation depends on the quality of the candidate list $\mathscr{L}$ and the list size N_{cand}. The SEE-LSD and the IR-LSD algorithms provide a list $\mathscr{L}$ including the most probable candidates, while the K-best-LSD algorithm does not guarantee that. This typically leads to a better approximation with the SEE-LSD and the IR-LSD compared to the K-best-LSD with the same list size. If the size N_{cand} of the candidate list $\mathscr{L}$ is large enough so that both sets $\mathscr{L}_{k,+1}$ and $\mathscr{L}_{k,-1}$ include candidates for the bit b_k, the approximation of the $L_{\mathrm{D}}(b_k)$ is typically accurate enough for adequate performance. However, the performance of the LSD may suffer due to too small a list size, and thus, inaccurate $L_{\mathrm{D}}(b_k)$ values. The error in the approximation of the $L_{\mathrm{D}}(b_k)$ is especially large in the case where all the candidates in $\mathscr{L}$ for the bit b_k belong to either $\mathscr{L}_{k,+1}$ or $\mathscr{L}_{k,-1}$. In that case the approximation of one of the conditional probabilities $p(\mathbf{y}|b_k = \pm 1)$ goes to zero, which leads to an infinite value in

(12). As the information is used as *a priori* information in the decoder, the decoder is most likely not able to correct the falsely detected signals.

3.1.1 Clipping methods

The effect of an unreliable $L_{\mathrm{D}}(b_k)$ may be reduced by limiting the $L_{\mathrm{D}}(b_k)$ range, which is often called LLR clipping. Several LLR clipping methods for different algorithms have been proposed, e.g., in [108, 111, 165, 166]. All the methods aim at reducing the detector algorithm complexity required to achieve a certain performance. The LLR clipping in [108, 111] is used to control the search effort used in the bit counterpart search. The method can not be applied as such for LSD algorithms due to the differences in the algorithms. The LLR clipping in [165, 166] is based on the reliability information and the channel state information (CSI) to determine close to optimal LLR clipping values with additional complexity. We introduce two simple methods to process $L_{\mathrm{D}}(b_k)$ information and study the impact of the methods on the performance of a coded system. Importantly, both presented methods are simple to implement and are suitable for VLSI implementation. The $L_{\mathrm{D1}}(b_k)$ calculated in the detector is given as $L_{\mathrm{A2}}(b_k)$ input to the decoder as illustrated in Figure 1 in Section . By limiting the dynamic range of the variable, the decoder can overcome the wrong information given as $L_{\mathrm{A2}}(b_k)$ in (6).

Method 1: A very simple way to prevent very large $L_{\mathrm{D}}(b_k)$ values is to limit the dynamic range of the $L_{\mathrm{D}}(b_k)$ value as [4]

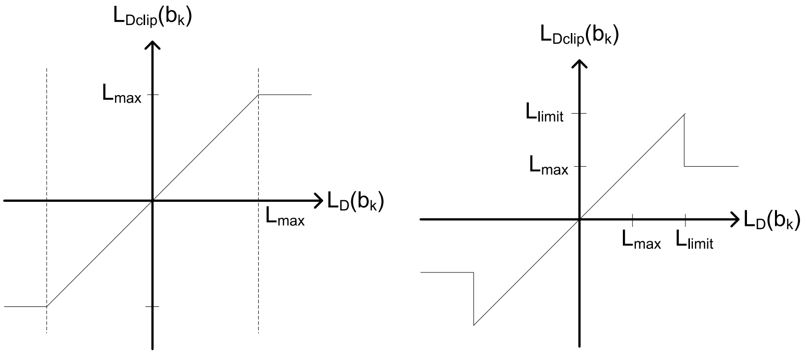
$$L_{\mathrm{D_{clip}}}(b_k) = \begin{cases} L_{\mathrm{D}}(b_k), & \text{if} |L_{\mathrm{D}}(b_k)| \leq L_{\max} \\ \operatorname{sgn}(L_{\mathrm{D}}(b_k))L_{\max} & \text{if} |L_{\mathrm{D}}(b_k)| > L_{\max}, \end{cases} \tag{45}$$

where $L_{\mathrm{D_{clip}}}(b_k)$ is the clipped likelihood information and $L_{\max}$ is the selected maximum value for $|L(b_k)|$.

Method 2: The second method differs slightly from the first. The $L_{\mathrm{D}}(b_k)$ values are clipped to $L_{\max}$ if a threshold value of $L_{\mathrm{limit}} > L_{\max}$ is exceeded as

$$L_{\mathrm{D_{clip}}}(b_k) = \begin{cases} L_{\mathrm{D}}(b_k), & \text{if} |L_{\mathrm{D}}(b_k)| \leq L_{\mathrm{limit}} \\ \operatorname{sgn}(L_{\mathrm{D}}(b_k))L_{\max} & \text{if} |L_{\mathrm{D}}(b_k)| > L_{\mathrm{limit}}. \end{cases} \tag{46}$$

The main idea of Method 2 is to clip only the very large $L_{\mathrm{D}}(b_k)$ values, which are due to a small LSD list size, and bypass the $L_{\mathrm{D}}(b_k)$ values where both bit values are present in (42). This can be achieved by setting the L_{limit} value large enough. Graphical illustrations of both methods are shown in Figures 11(a) and 11(b).



(a) Method 1

(b) Method 2

Fig 11. A graphical illustration of the LLR clipping methods.

3.1.2 Numerical examples

We studied the impact of the two different LLR clipping methods on the performance of the system via computer simulations. The same parameters were assumed as in Section 2.6. We studied the distribution of the calculated LLR values and a histogram of the IR-LSD soft-output LLR distribution with $N_{\text{cand}} = 1024$ and $N_{\text{cand}} = 8$ and with different SNR values is shown in Figure 12. It should be noted that the two dimensional curves are plotted into a three dimensional figure for illustrative purposes. The $E_S/N_0 = 18$dB and $E_S/N_0 = 26$dB cases studied reflect low and high throughput operating points of the system, respectively. The LLR values were limited to $L_{\text{max}} = 100$ to show the very large values. It can be seen that with increasing SNR the deviation of LLR widens and we can see more high LLR values, but the probability of $|L_D| > 40$ and $|L_D| > 20$ is fairly low with the higher and lower SNR values, respectively. The effect of the small list size can be seen as a large amount of L_{max} values in the histogram as the probability of empty set $\mathscr{L}_{k,+1}$ or $\mathscr{L}_{k,-1}$ in the calculation of $L(b_k)$ is significantly higher.

We also studied the impact of different clipping methods and L_{max} values on the performance of the system to determine the optimal clipping method and the threshold value to be used for clipping. It should be noted that while too low a value for L_{max} prevents the detector from giving any significant *a priori* information to the decoder, too high a value for L_{max} allows the possibility for *a priori* information that is too high caused by an $\mathscr{L}$ that is unreliable or too small which the decoder can not correct in the case of a wrong decision. Performance of the IR-LSD with $N_{\text{cand}} = 8$, which is equal to the SEE-LSD with the same list size, and real K-best-LSD with $N_{\text{cand}} = 64$ with both clipping methods applied are shown for a 4×4 MIMO system with 16-QAM in Figure 13. Method 2 is applied with $L_{\text{limit}} = 100$ to clip only the very large $L_{\text{D}}(b_k)$ values. The results show that the performance of a system is clearly improved by applying LLR clipping to limit the effect of the moderate LLR approximation compared to the system without clipping. We also notice that there is no significant performance difference between the two clipping methods with the IR/SEE-LSD. However, we noticed from the results that Method 1 is clearly better with K-best-LSD. The reason for this is the different outputs from the IR/SEE-LSD and the K-best-LSD. The IR/SEE-LSD gives the most probable candidates as an output, and thus, the LLR approximation is rather good and reliable in the cases where both bits are present in (12). The K-best-LSD output, however, may result in a poor LLR approximation when candidates for both bits are present in (12). Thus, we conclude that Method 1 is a good choice as it requires less dynamic range for the $L(b_k)$ before clipping. The simulation results show that $L_{\max} = 8$ gives the best performance, which means that the dynamic range of probability $P(b_k = \pm 1|\mathbf{y})$ is limited between $[0.0003, 0.9997]$. We also studied if and how the code rate possibly inter-plays with the optimal $L_{\max}$ value at the detector. The performance of a IR/SEE-LSD based system with code rates $1/3$ and $4/5$ and with different Method 1 clipping values are shown in Figure 14. We can see that with a lower code rate $1/3$ the value $L_{\max} = 6$ gives the best performance with a difference of 0.1 dB compared to $L_{\max} = 8$. The performance of the system with a higher code rate $4/5$ is maximized with the value $L_{\max} = 10$, but the difference from the value $L_{\max} = 8$ is approximately 0.05 dB. The results indicate that as the decoder has more parity bits to be used in the decoding, the decoder should rely less on the *a priori* information $L_{\text{A2}}(b_k)$ from the detector and the $L_{\max}$ can be set to be a lower value. However, the differences are rather small, and in practice, $L_{\max} = 8$ gives good results.

It was shown that the LLR clipping enhances the performance of LSD based systems with low list size and, thus, impacts on the required list size. The required list sizes for a system without LLR clipping have been determined in [141, 143]. We applied Method 1 with $L_{\max} = 8$ for LLR clipping in the following results. The performance was studied in two channel models, a highly correlated and an uncorrelated typical urban channel, in order to study the effect of the channel correlation. Performance examples of a 4×4 MIMO system with 16-QAM with the IR/SEE-LSD are shown in Figure 15, and the same case with the real K-best-LSD with different list sizes is shown in Figure 16. It can be seen that the required list size with the IR/SEE-LSD decreases significantly with

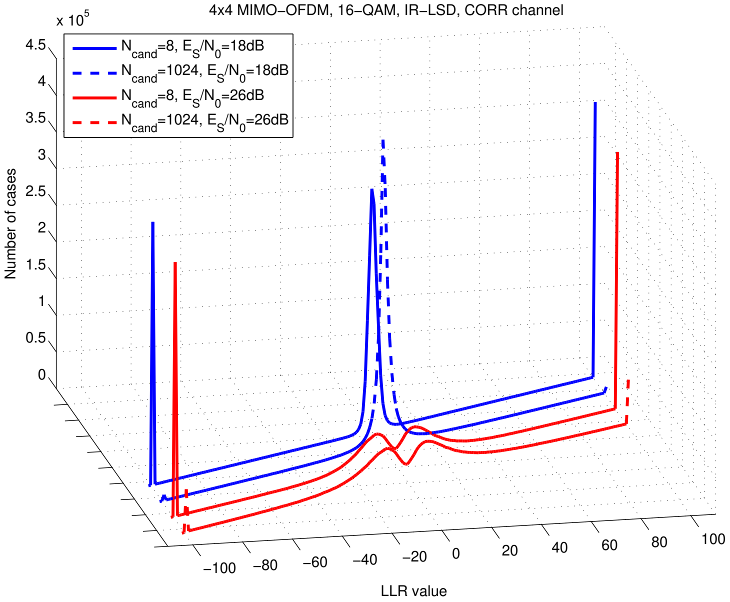


Fig 12. A histogram of the LLR values with IR/SEE-LSD with different list sizes and SNR values in a 4×4 antenna system with 16-QAM. ([138], published by permission of Elsevier).

LLR clipping applied and, e.g., the required list size of IR/SEE-LSD decreases from 64 to 8 in a 4×4 MIMO system with 16-QAM. The results in Figure 16 show that the required list size with the K-best-LSD does not decrease as significantly as with the IR/SEE-LSD as discussed earlier. However, in the same case the LLR clipping still reduces the required list size from 128 to 64. The benefit of the LLR clipping with the real and complex[1] K-best-LSD is smaller than that with the IR/SEE-LSD, because of the breadth-first search strategy that usually leads to having both $b_k = +1$ or $b_k = -1$ candidates in the LLR calculation, but does not provide the most probable candidates. Thus, the quality of the obtained list is not as high as with IR/SEE-LSD and a larger list size is required for similar LLR approximation. However, it can be noted that the quality of the obtained list increases as the channel is more uncorrelated, i.e., the tree search is easier. The required list sizes were determined for 2×2 and 4×4 MIMO systems with 4-QAM, 16-QAM, and 64-QAM, and the results are concluded in Table 7.

[1] The real and complex versions are rigorously defined and considered in more detail in Section 3.2.

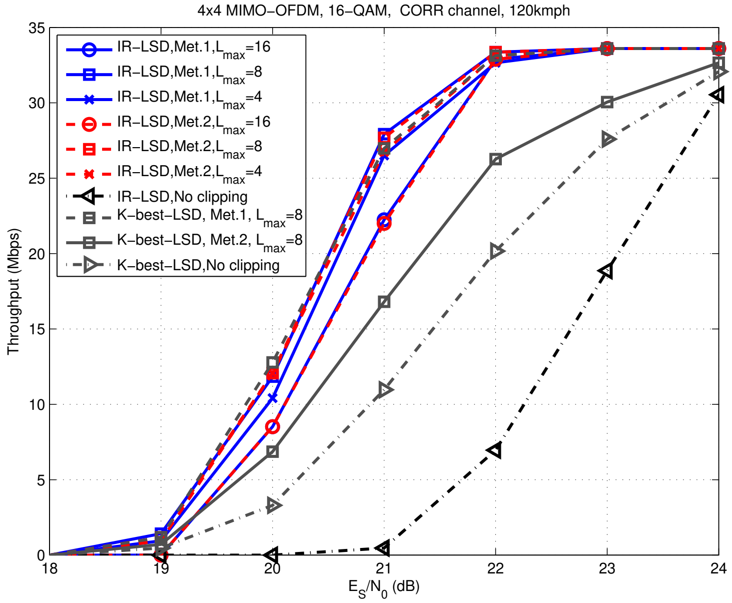


Fig 13. Throughput vs SNR: Performance of the IR/SEE-LSD and the real K-best-LSD with different LLR clipping methods and values in a 4×4 antenna system with 16-QAM. ([138], published by permission of Elsevier).

Table 7. List sizes for IR/SEE-LSD*, real K-best-LSD† and complex K-best-LSD‡ with LLR clipping. ([138], published by permission of Elsevier).

	2×2	4×4
4-QAM	not studied	$N_{\text{cand}} = 8^*/32^\dagger/32^\ddagger$
16-QAM	$N_{\text{cand}} = 8^*/16^\dagger/16^\ddagger$	$N_{\text{cand}} = 8^*/64^\dagger/128^\ddagger$
64-QAM	$N_{\text{cand}} = 16^*/64^\dagger/64^\ddagger$	$N_{\text{cand}} = 16^*/128^\dagger/256^\ddagger$

3.1.3 Conclusions

We introduced two simple LLR clipping methods to process the soft output LLR $L_D(b_k)$ and studied the impact of the methods on the performance of a coded system. We showed that the LLR clipping enhances the performance of LSD based systems with low list size and, thus, lowers the required list size N_{cand}. There is no significant performance difference between the two clipping methods with the IR/SEE-LSD, but Method 1 is clearly better with the K-best-LSD. This is due to the less reliable candidate list $\mathscr{L}$

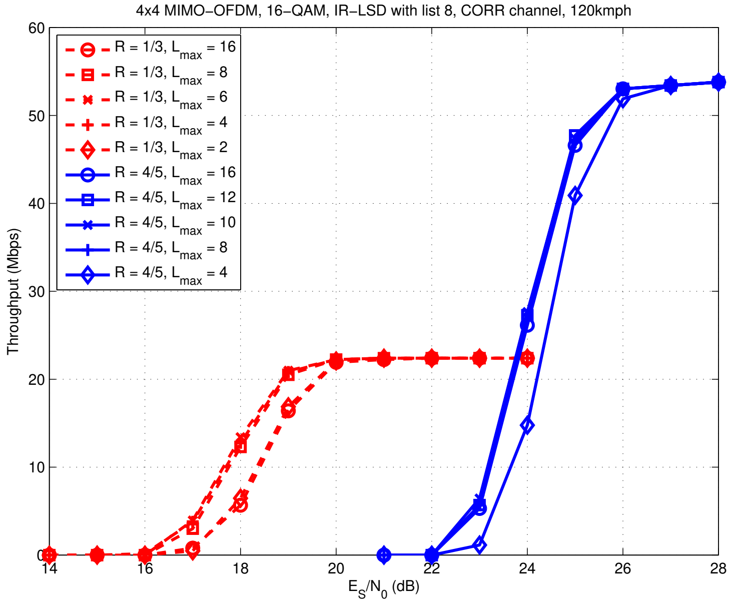


Fig 14. Throughput vs SNR: Performance of the IR/SEE-LSD with different code rates and LLR clipping values in a 4×4 antenna system with 16-QAM. ([138], published by permission of Elsevier).

with K-best-LSD, which might lead falsely to large $L(b_k)$ values even with both $b_k = +1$ or $b_k = -1$ candidates present in the LLR calculation. Thus, we propose to use Method 1 as it ensures good performance and requires less dynamic range for the $L(b_k)$ before clipping. The coding rate has a small effect on the optimal clipping values, but the differences are rather small, and in practice, $L_{\text{max}} = 8$ gives good results, i.e., the dynamic range of $P(b_k = \pm 1|\mathbf{y})$ is limited between $[0.0003, 0.9997]$. It was also noted that the LLR clipping reduces the required list size N_{cand} more with the IR/SEE-LSD than with the K-best-LSD.

3.2 Real and complex signal model

The SD and LSD algorithms are often assumed to apply a real equivalent system model [91, 99, 167–169] especially in the implementation of the algorithms. However, complex valued signal models are also applied in the literature [4, 111, 117]. The definition of the signal model does not affect the mathematical equivalence of the expressions,

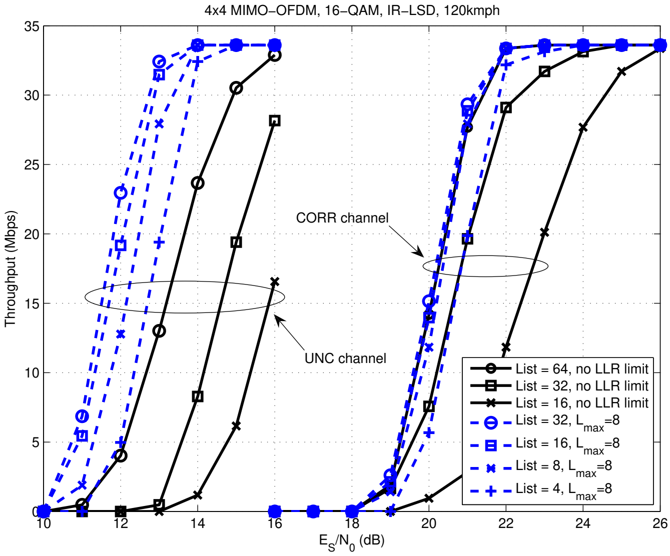


Fig 15. Throughput vs SNR: Performance of the IR/SEE-LSD with different list sizes in a 4×4 antenna system with and without LLR clipping. ([138], published by permission of Elsevier).

but it affects the lattice definition where the LSD algorithm search is executed. Thus, there is a need to justify the selection of the signal model by studying the effect of it on the LSD search and on the total complexity of the search. The complex valued system model can be presented as an equivalent real valued signal model as indicated in Section 3.1.1.

The complexity of the LSD algorithms is relative to the number of visited nodes in the search tree and the size of the search tree, and, as already mentioned, the size of the search tree depends on the applied signal model. The use of a real valued system model doubles the depth of the search tree, i.e., $M_\mathrm{T} = 2N_\mathrm{T}$, but decreases the number of branches at each level compared to the complex signal model, i.e., $|\Omega_\mathrm{r}| = |\Omega|^{\frac{1}{2}}$. Thus, the total number of branches in the search tree with a real valued signal model is given as $B_\mathrm{R} = \sum_{i=1}^{M_\mathrm{T}} |\Omega_\mathrm{r}|^i$, and with complex valued signal model as $B_\mathrm{C} = \sum_{i=1}^{N_\mathrm{T}} |\Omega|^i$. A graphical illustration of the different tree shapes is shown in Figure 17. The sizes of the real and complex valued search trees approach each other as the number of transmit antennas N_T and the constellation size $|\Omega|$ increase, but the search tree size is larger with the real valued signal model with moderate N_T and $|\Omega|$, e.g., in a 4×4 system with 16-QAM

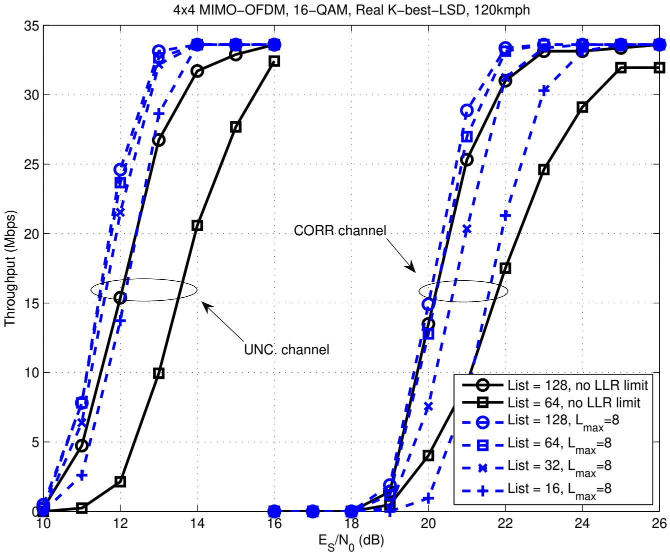
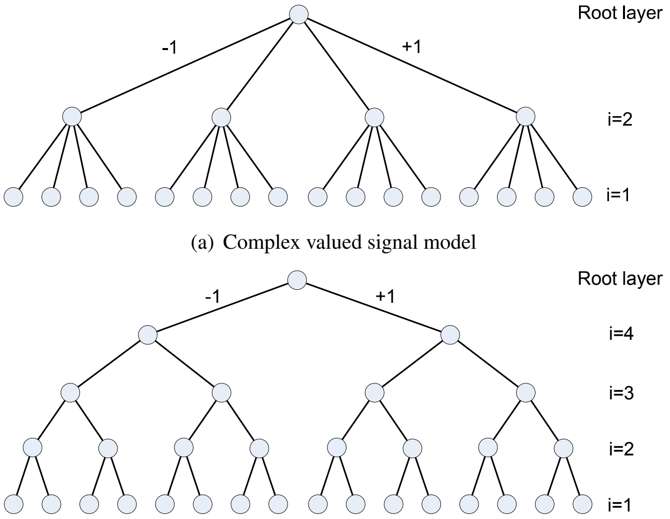


Fig 16. Throughput vs SNR: Performance of the real K-best-LSD with different list sizes in a 4×4 antenna system with and without LLR clipping. ([138], published by permission of Elsevier).

$B_{\mathrm{R}} = 87380$ and $B_{\mathrm{C}} = 69904$. Thus, it is likely that the number of visited nodes by the LSD algorithm increases somewhat with a real valued signal model.



(a) Complex valued signal model

(b) Real valued signal model

Fig 17. A tree structure of a sphere detector with a 2×2 MIMO system and 4-QAM.

The considered LSD algorithms were described in detail in Section 2.5.2. The main complexity of the LSD algorithm comes from the PED calculation in (40), which is executed for each node studied. When considering the choice of the signal model, we should notice that the complexity of the PED calculation in (40) includes different operations with real and complex valued signals. The numbers of operations required to calculate (40) depends on the number of transmit antennas N_T and the current layer i in the search tree. They are listed as real operations in Table 8 given that one complex MUL is equal to three real MULs and five real ADDs and one complex ADD is equal to two real ADDs. As a numerical example, we assume that $N_\mathrm{T} = 4$ and the average studied node in both the real and the complex valued search tree is in the middle of the tree depth, i.e., $\mathrm{E}[i_\mathrm{R}] = M_\mathrm{T}/2 = 4$ and $\mathrm{E}[i_\mathrm{C}] = N_\mathrm{T}/2 = 2$. Then the number of required real operations on average for the PED calculation is 9 MULs and 21 ADDs for the complex signal model, and 5 MULs and 5 ADDs for the real signal model. Thus, we can say that on average the complexity of one LSD algorithm node check in a system with $N_\mathrm{T} = 4$ is approximately double the complex valued signal model compared to the real valued signal model.

Table 8. The number of real operations used for PED calculation in (40). ([138], published by permission of Elsevier).

	Real valued signal model	Complex valued signal model
MUL	$2N_T - i + 1$	$3(N_T - i + 1)$
ADD	$2N_T - i + 1$	$7(N_T - i + 1)$

3.2.1 Numerical examples

We studied the impact of the real and the complex valued signal models on the number of visited nodes by the considered LSD algorithms via computer simulations. The simulations were executed with the same parameters as those in Section 2.6.

The number of visited nodes by the K-best-LSD algorithm depends on the signal model and the applied list size $K = N_{\text{cand}}$. The K-best-LSD algorithm visits a fixed number of nodes given the list size and that no sphere radius is introduced, i.e., $C_0 = \infty$. The number of visited nodes by the real valued K-best-LSD and the complex valued K-best-LSD algorithms are determined as $V_{\text{R}} = \sum_{i=1}^{M_{\text{T}}} |\mathscr{S}||\Omega_{\text{r}}|$ and $V_{\text{C}} = \sum_{i=1}^{N_{\text{T}}} |\mathscr{S}||\Omega|$, where $|\mathscr{S}|$ is the number of stored candidates at each layer as in Algorithm 3. The numbers of visited nodes in different antenna and constellation cases with determined list sizes in Table 7 are listed in Table 9. It can be seen that the K-best-LSD with the real valued signal model visits fewer nodes in all the cases except the 4 × 4 MIMO with a QPSK case compared to the complex valued signal model with the same performance. The reason for this is the difference in possible signal points in one layer between the real and the complex valued signal model, i.e., $|\Omega_{\text{r}}| = |\Omega|^{\frac{1}{2}}$. As the algorithm visits all the possible child nodes of the stored partial candidates at each layer, the search with the real valued signal model is done with less visited nodes in total even though visiting double the number of layers.

Table 9. Number of visited nodes with the real valued K-best-LSD† and the complex valued K-best-LSD‡ with determined list sizes. ([138], published by permission of Elsevier).

	2×2	4×4
4-QAM	not studied	$254^{\dagger}/212^{\ddagger}$
16-QAM	$148^{\dagger}/272^{\ddagger}$	$1364^{\dagger}/4368^{\ddagger}$
64-QAM	$1096^{\dagger}/4160^{\ddagger}$	$5704^{\dagger}/36928^{\ddagger}$

The number of visited nodes by the sequential search LSD algorithms varies with channel realization and SNR. We studied the number of visited nodes by the IR-LSD and the SEE-LSD algorithms with both the real and the complex valued signal model by collecting data from the computer simulations. The data is then plotted as a histogram to illustrate the distribution of the number of visited nodes by the LSD algorithms. The number of visited nodes were studied in highly correlated and uncorrelated TU channel to also examine the effect of the channel realization. The number of visited nodes by the SEE-LSD algorithm in a 4×4 MIMO system with 16-QAM and 64-QAM are illustrated in Figures 18 and 19. It can be seen that the SEE-LSD algorithm with the real valued signal model visits approximately 1.5-2 times the number of nodes visited by the algorithm with the complex valued signal model in the 4×4 MIMO with 16-QAM case, but the difference slightly decreases with 4×4 MIMO with 64-QAM case. The number of nodes visited by the IR-LSD algorithm in 4×4 MIMO system with 16-QAM and 64-QAM are shown in Figures 20 and 21, respectively. The results show that the number of visited nodes by the algorithm is relative to the total number of nodes in the search tree, and the difference between the real and the complex valued signal model decreases from 16-QAM to 64-QAM. The algorithm with the real valued signal model visits approximately 1.3-1.5 times the number of nodes visited by the complex algorithm. It can also be noted that the channel realization and correlation properties of the channel have an effect on the distribution of the visited nodes with both the SEE-LSD algorithm and the IR-LSD algorithm. The results also confirmed that the difference in the number of visited nodes between the real and complex valued signal model decrease as the N_{T} and $|\Omega|$ increase.

3.2.2 Conclusions

We studied the difference between using the real valued or complex valued signal model in the LSD algorithm. The real signal model is clearly the better choice to be applied with LSD algorithms given the number of visited nodes and the complexity difference in required operations in one node check. The complexity of one LSD algorithm node check in a system with $N_{\mathrm{T}} = 4$ is approximately double with the complex valued signal model compared to the real valued signal model. We noted that the distribution of the number of nodes visited by the LSD algorithm varies significantly depending on the LSD algorithm search method. We also noted that the channel realization affects the distribution of the number of nodes visited by the sequential search algorithms.

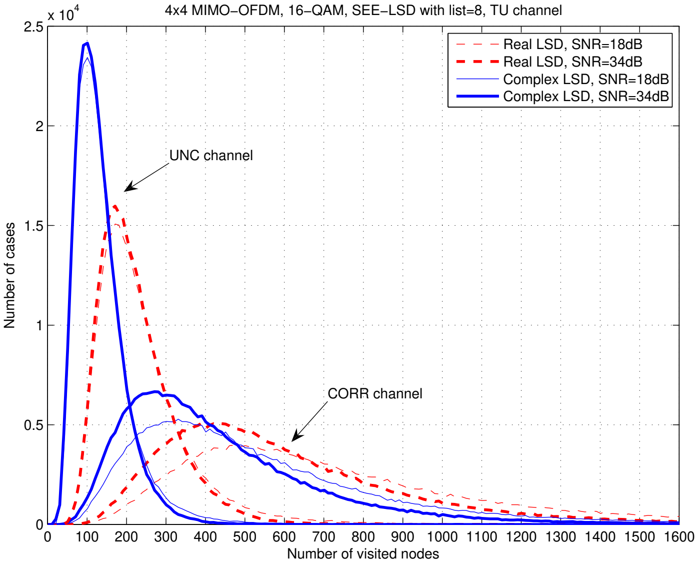


Fig 18. Histogram of the number of visited nodes per symbol vector with real and complex SEE-LSD in TU channel with a 4×4 MIMO system with 16-QAM. ([138], published by permission of Elsevier).

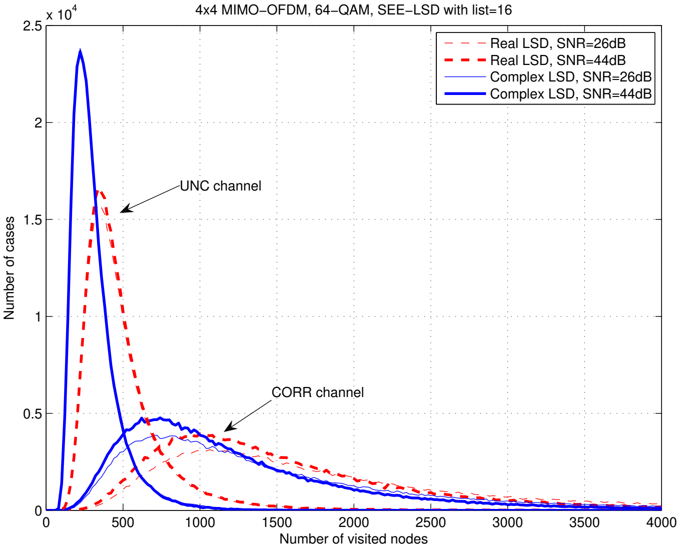


Fig 19. Histogram of the number of visited nodes per symbol vector with real and complex SEE-LSD in TU channel with a 4×4 MIMO system with 64-QAM. ([138], published by permission of Elsevier).

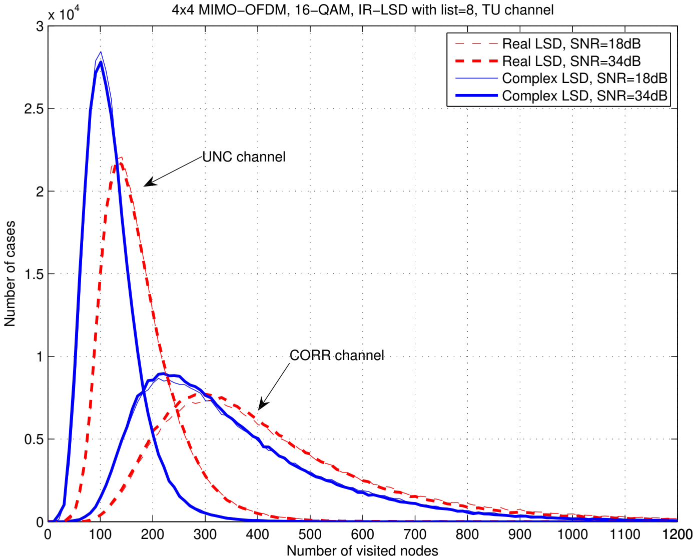


Fig 20. Histogram of the number of visited nodes per symbol vector with real and complex IR-LSD in TU channel with a 4×4 MIMO system with 16-QAM. ([138], published by permission of Elsevier).

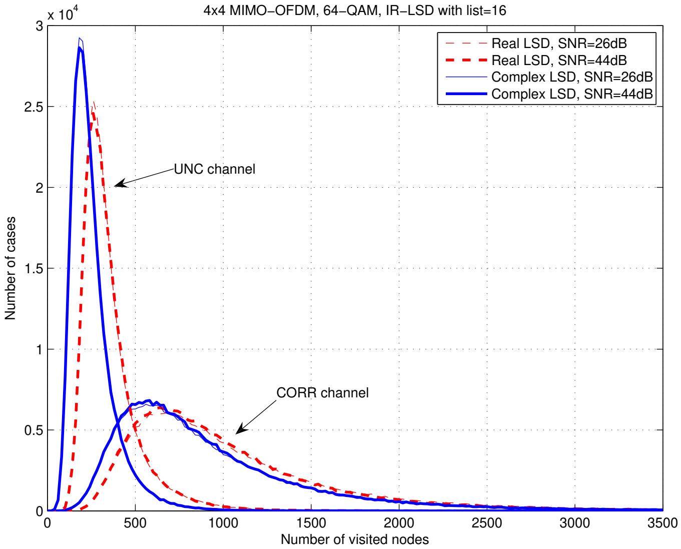


Fig 21. Histogram of the number of visited nodes per symbol vector with real and complex IR-LSD in TU channel with 4×4 MIMO system with 64-QAM. ([138], published by permission of Elsevier).

However, in general, the IR-LSD and the SEE-LSD algorithms with the real valued signal model are less complex compared to the complex valued signal model. The K-best-LSD with the real signal model also visits fewer nodes with all the configurations studied except the 4×4 MIMO with QPSK case compared to the complex signal model with the same performance.

3.3 Limited search

In the hardware implementation of an algorithm for a practical system, there is usually a predetermined time to execute the process that the algorithm carries out. In order to reserve the hardware resources for the algorithm to meet the given timing constraints, we need to determine the so-called worst case scenario and determine the algorithm complexity accordingly. From the considered list sphere detectors, the K-best-LSD checks a fixed number of nodes when $C_0 = \infty$ and, thus, the complexity of the algorithm is fixed. The SEE-LSD and the IR-LSD, however, visit a variable number of nodes depending on the channel realization, and the hardware implementation of these algorithms as such is not feasible for a system with a fixed latency requirement.

In order to fix the complexity of the SEE-LSD and the IR-LSD algorithms, we propose a simple way to modify the algorithms to limit the maximum number of nodes visited by the LSD algorithm, which we call limited search (LS). The *while* loop in the algorithm description of the SEE-LSD in Algorithm 4 and the IR-LSD in Algorithm 5 can be replaced with a for loop, which means that a predefined maximum number of loop runs is set and a maximum of L_{node} nodes studied. If the sphere search is not completed within the defined maximum limit L_{node}, the algorithm is stopped and the current final candidate list $\mathscr{L}$ is given as an output. Another more sophisticated alternative is to use a scheduling algorithm as, e.g., in [111, 170]. We modified the scheduling algorithm in [111] to be more suitable for the SEE-LSD and the IR-LSD algorithms and call it scheduled search (SS). We use the algorithm to determine the search limit $L_{\text{node}}(n)$ for the nth subcarrier in the OFDM symbol as

$$L_{\text{node}}(n) = N_{\text{used}}L_{\text{avg}} - \sum_{i=1}^{n-1} L_{\text{node}}(i) - (N_{\text{used}} - n)L_{\text{min}}, \tag{47}$$

$$L_{\text{node}}(n) = \begin{cases} DL_{\text{avg}}, & \text{if } L_{\text{node}}(n) > DL_{\text{avg}} \\ L_{\text{node}}(n) & \text{if } L_{\text{node}}(n) \le DL_{\text{avg}}, \end{cases} \tag{48}$$

where $N_{\text{used}}L_{\text{avg}}$ is the total node run-time constraint for the whole OFDM symbol and $L_{\min}$ is minimum number of studied nodes reserved for each subcarrier. The $L_{\text{node}}(n)$ is also upper limited to a maximum of DL_{avg} nodes, where D is a node coefficient, to prevent the use of too many resources in one subcarrier. The idea behind the scheduled search is that the algorithm is able to allocate higher maximum limits $L_{\text{node}}(n)$ for subcarriers that have a channel realization resulting in low SNR while subcarriers with easier channel realization can be allocated with lower limits $L_{\text{node}}(n)$.

3.3.1 Numerical examples

We studied the effect of limited search and scheduling algorithm via Monte Carlo simulations to determine the performance of the method. The numerical examples were executed with the same parameters as in Section 2.6, and including LLR clipping with $L_{\max} = 8$ and the real valued signal model. First we studied the number of visited nodes in the search tree by the IR-LSD and the SEE-LSD algorithms to determine the initial limit values L_{node}. The histograms of the number of visited nodes in a 4×4 MIMO–OFDM system with 16-QAM and LSD list size $N_{\text{cand}} = 8$ in the TU channel are shown in Figures 18 and 20 for the SEE-LSD and IR-LSD algorithms, respectively. It can be noted that the channel realization and especially the correlation properties of the channel affect the distribution of the number of nodes visited by the LSD algorithms and it should be taken into account when determining the proper limit parameter values for the search.

Numerical examples of the real SEE-LSD algorithm with both LSD and SS methods and with different parameters in a 4×4 MIMO–OFDM system with 16-QAM and in the TU channel are shown in Figure 22. It can be seen that there is no performance loss with the real SEE-LSD algorithm with the LS method as the LSD algorithm search limit L_{node} is set high enough. However, we note that the proper maximum limit value L_{node} for the depth-first search has to be set relatively high compared to the distribution of the visited nodes and the proper value $L_{\text{node}} = 3000$ in a correlated channel equals to 98.8% of the cumulative distribution function (CDF), i.e., $\Pr[X_{SEE}^{(CORR,18dB)} \leq L_{\text{node}} = 3000] = 0.988$, where $X_{SEE}^{(CORR,18dB)}$ is the CDF determined from the histogram of the TU channel with SNR= 18dB. The corresponding value in an uncorrelated channel, $L_{\text{node}} = 700$, equals to 98.7% of the CDF. The sensitivity to the limiting is due to the depth-first search strategy, where the search does not proceed uniformly in the search tree and the first node decision in the highest layer is emphasized. Thus, if the search is limited,

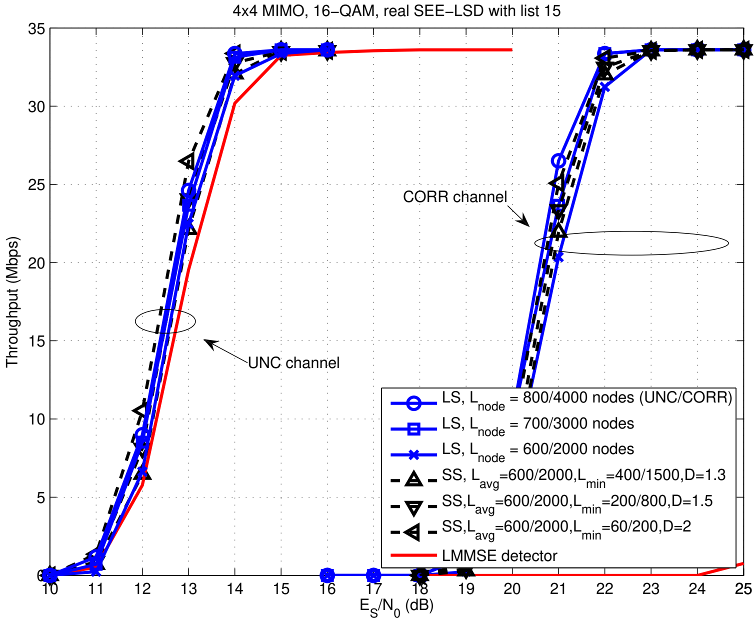


Fig 22. Throughput vs SNR: Performance of the real SEE-LSD with limited maximum number of nodes in a 4×4 MIMO–OFDM system with 16-QAM. ([138], published by permission of Elsevier).

one part of the tree, which could contain a relatively probable path, might be left totally unexamined. The SS method improves the performance of the SEE-LSD based receiver with approximately 0.5 dB compared to the LS method with proper parameters and the same resources, i.e., $L_{\text{node}} = L_{\text{avg}} = 2000$. The SS method with a relatively small L_{min} and relatively large D values shows the best performance results, because the SEE-LSD algorithm is able to detect most of the subcarriers with difficult channel realization properly with large L_{node} and is still able to achieve a non empty candidate list $\mathscr{L}$ for the rest of the subcarriers with L_{min} due to the depth first search strategy. Then the impact of limiting the IR-LSD algorithm search with both LS and SS methods on the performance of the system was studied. Numerical examples of the performance with different limited search parameter values in a 4×4 MIMO–OFDM system with 16-QAM and in TU channels are shown in Figure 23. It can be seen that there is no performance loss with the real-valued IR-LSD algorithm as the LS method limit L_{node} is set high enough, i.e., $L_{\text{node}} = 200/500$ in the UNC/CORR channel. The performance

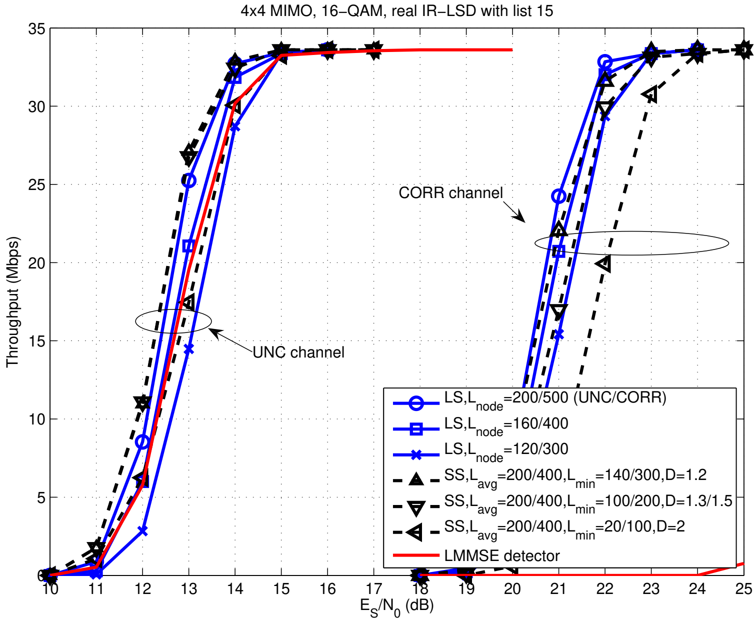


Fig 23. Throughput vs SNR: Performance of the real valued IR-LSD with limited maximum number of nodes in a 4×4 MIMO–OFDM system with 16-QAM. ([138], published by permission of Elsevier).

degradation of the system with the LSD search limit set to $L_{\text{node}} = 500$ nodes, which is a rather low limit compared to the determined distribution of visited nodes $X_{IR}^{(CORR,18dB)}$, is about $0.2-0.3$ dB at maximum compared to the LSD with an unlimited search. The CDF with determined L_{node} equals to 80.6%, i.e., $\Pr[X_{IR}^{(CORR,18dB)} \leq L_{\text{node}} = 500] = 0.806$. The corresponding value in an uncorrelated channel, $L_{\text{node}} = 160$, equals to 79.5% of the CDF. The good tolerance of the IR-LSD algorithm to the LS method is due to the metric-first search strategy, where the algorithm proceeds uniformly in the search tree. A more sophisticated SS method was also applied with $L_{\text{avg}} = 200/400$ and with different L_{min} and D values for the UNC/CORR channel. The SS method slightly outperforms the simple LS method, i.e., by ≈ 0.1 dB, with proper parameter configuration when the same resources are applied $L_{\text{node}} = L_{\text{avg}}$. In practice, L_{min} has to be set high enough and D not too large to guarantee enough nodes for each subcarrier detection and a non empty output candidate list $\mathscr{L}$. We determined the proper search limits L_{node} for the algorithms for highly correlated TU channels and they are listed in Table 10. It should be noted that the highly correlated channel scenario is difficult for the sphere search, and the results listed can be considered as the so-called worst case scenario limits. The results show that the channel correlation properties effect the distribution of nodes in the algorithm, and a feasible search limit L_{node} can be set lower in an uncorrelated channel.

3.3.2 Conclusions

We proposed two methods to limit and fix the search complexity of the SEE-LSD and IR-LSD algorithms: limited search (LS) and scheduled search (SS) methods. The results show clearly that both the considered sequential search algorithms work well with a limited search. The SEE-LSD algorithm performs ≈ 0.5 dB better with the more sophisticated and complex SS method compared to the LS method, but the IR-LSD algorithm achieves only a minor performance gain, which is due to the different search strategies of the algorithms. However, both the SS and the LS methods are feasible for implementation. The results indicate that the CDF of the visited nodes can be used as a guideline for determining the proper L_{node} value.

3.4 IR-LSD memory sphere radius

The IR-LSD algorithm [106, 143], which was described in Section 2.5.2, is optimal in the sense of number of nodes visited in the tree structure [95, 106]. The algorithm always extends the partial candidate $\mathscr{N}$ with the lowest PED in one extend loop, but requires that the visited nodes are maintained in metric order to ensure the optimality, which requires the usage of memory and sorting [98]. The algorithm stores at maximum a single additional partial candidate $\mathscr{N}$ in the memory and, thus, requires a memory unit of L_{node} candidates at maximum when the search is limited to L_{node} iterations. The min-

Table 10. Determined maximum node limits for the real valued IR-LSD† and the real valued SEE-LSD‡ with LLR clipping in a highly correlated TU channel. ([138], published by permission of Elsevier).

	2×2	4×4
4-QAM	not studied	$250^{\dagger}/300^{\ddagger}$
16-QAM	$80^{\dagger}/150^{\ddagger}$	$500^{\dagger}/3000^{\ddagger}$
64-QAM	$200^{\dagger}/400^{\ddagger}$	$1000^{\dagger}/>10000^{\ddagger}$

imum candidate in the memory also has to be searched in each iteration [98]. Sorting or minimum search are costly operations and the complexities of the operations are related to the number of the sorted elements [102, 171]. The memory access in the sorting of an added partial candidate to the memory set $\mathscr{S}$ may be the limiting factor in the implementation if the memory set $\mathscr{S}$ gets close to full, i.e., the sorting time likely increases with a growing number of occupied elements in the memory set $\mathscr{S}$ [102, 171].

We propose to use a novel memory sphere radius C_{mem} to decrease the number of stored candidates and the complexity of the required minimum search [139]. The added extended partial candidates $\mathscr{N}$ are first compared to the C_{mem} and stored to the memory $\mathscr{S}$ only if $d(\mathbf{s}) < C_{\text{mem}}$. Thus, the amount of stored partial candidates $\mathscr{N}$ is reduced depending on the definition of C_{mem}. We define C_{mem} based on the previously solved candidate(s) in the final list(s) with a minimum ED $\min_{\mathbf{x}\in\mathscr{L}}(d(\mathbf{x}))$, which is then scaled with a determined radius scaling variable W_R to store only the potential partial candidates to the partial memory set $\mathscr{S}$. The minimum ED values can be averaged over time and frequency, i.e., OFDM subcarriers, and then the memory sphere radius can be written as

$$C_{\text{mem}} = W_R E[\min_{\mathbf{x}\in\mathscr{L}}(d(\mathbf{x}))]. \tag{49}$$

The impact of C_{mem} on complexity and performance is studied with numerical examples in Section 3.4.1.

3.4.1 Numerical examples

The numerical examples were executed to verify the feasibility of the memory sphere radius. A turbo coded 4×4 MIMO-OFDM system configuration was applied as in the simulations in Section 2.6 in a Winner B1 channel [152] with a user velocity of 60 kmph. The receiver includes an IR-LSD with a list size $N_{\text{cand}} = 15$ and a max-log-MAP turbo decoder with 8 iterations. A SQRD preprocessing is assumed in the LSD and LLR clipping was applied with $L_{\max} = 8$.

We studied the impact of the memory sphere radius C_{mem} on the IR-LSD performance and determined a proper value for W_{R} to be used for the cases studied. We studied the complexity reduction in terms of the average number of executed iterations L_{avg} with $L_{\text{node}}^{\text{16QAM}} = 80$ and $L_{\text{node}}^{\text{64QAM}} = 150$. The memory unit $\mathscr{S}$ and the sorting architecture is designed as a binary heap [102, 171] data structure, which is presented in more detail later in Section 4.2. The sorting in a heap data structure is done via up- and down-heap

Table 11. The average number of visited nodes and executed heap operations, and performance loss in IR-LSD with C_{mem}.

W_R, 16-QAM	C_0	4.0	3.0	**2.0**	1.7
L_{avg}	77	76	70	**47**	35
Avg. up-heaps	1.09	0.68	0.48	**0.25**	0.18
Avg. down-heaps	1.45	1.39	1.19	**0.72**	0.53
Perf. loss (dB)	0.0	0.0	0.0	**0.1**	0.7
W_R, 64-QAM	C_0	4.0	3.0	**2.5**	2.0
L_{avg}	116	113	104	**93**	74
Avg. up-heaps	1.03	0.63	0.44	**0.33**	0.23
Avg. down-heaps	1.59	1.46	1.23	**1.03**	0.76
Perf. loss (dB)	0.0	0.0	0.0	**0.1**	0.4

operations [102, 171], and we also studied the average number of required heap operations in the partial memory $\mathscr{S}$ in one subcarrier detection. The numerical results of the IR-LSD with different W_{R} values are listed in Table 11. The performance of the IR-LSD is not degraded significantly as the W_{R} value is selected to be large enough and only the potential partial candidates are stored to partial memory $\mathscr{S}$ with $W_{\text{R}}^{16\text{QAM}} \geq 2.0$ and $W_{\text{R}}^{64\text{QAM}} \geq 2.5$. As the C_{mem} is relative to the average minimum candidates over time and frequency, any unnecessary resources, i.e., algorithm iterations, are not used to obtain candidates with relatively high ED in the case of difficult channel realizations. Thus, the average numbers of visited nodes decreases without loosing performance as shown in Table 11. The average number of up- and down-heap operations is also decreased, which significantly reduces the required memory access and the latency of the heap sorting.

3.4.2 Conclusions

We proposed to use a novel memory sphere radius C_{mem} with the IR-LSD algorithm to decrease the number of stored candidates and the complexity of the required minimum search during the algorithm tree search. We showed that the average numbers of visited nodes decreases with memory sphere radius without losing performance. Also, if a heap memory structure is used with the IR-LSD algorithm, the average number of up- and down-heap operations decreases, which significantly reduces the required memory access and the latency of the heap sorting.

3.5 L1 norm

Most of the arithmetic operations required by the LSD algorithms are executed in PED calculations. Normally the squared Euclidean distance or the L2 or 2-norm without square root operation is applied in the LSD algorithm decision metric as presented in (40). However, simplified norm calculation methods, such as the absolute norm or infinite norm, have also been introduced to decrease the complexity of the LSD algorithm [172]. The use of the L1 or absolute value norm in the tree search instead of the ED calculation simplifies (40) and results in decreased complexity as one multiplication and one addition less is required per one node extension. We will call the L1 norm based metric calculation a partial absolute distance (PAD) calculation. The total number of operations saved per OFDM subcarrier detection then depends on the total number of nodes visited by the tree search algorithm, i.e., depends on the system configuration and the channel realization. The partial absolute distance calculation with simplified L1 norm algorithm and with a real signal model can then be written as [70]

$$d(\mathbf{x}_i^{M_\mathrm{T}}) \approx d(\mathbf{x}_{i+1}^{M_\mathrm{T}}) + |\tilde{y}_i - \sum_{j=i}^{M_\mathrm{T}} R_{i,j} x_j|. \tag{50}$$

It has been shown that the usage of the L1 norm also results in a degradation in the performance with an uncoded system as the hard output tree search is made based on the L1 norm [172]. The use of the L1 norm in a coded system may result in a different final candidate list $\mathscr{L}$ compared to the use of the L2 norm due to the different decision metrics in the search. The use of absolute distance (AD) values in (42) also results in distorted LLR approximation. We studied the performance of the MIMO–OFDM system with an IR-LSD based receiver in the UNC and CORR channels and compared the L1 and L2 norm based IR-LSD algorithms. We also studied the effect of the L1 norm on the IR-LSD algorithm tree search by using the L1 norm in the tree search and then recalculating the final list candidates' ED values with the L2 norm. The results of a coded 4×4 MIMO–OFDM system with 16-QAM are shown in Figure 24. It can be seen that the performance loss due to the use of the L1 norm is approximately 1–1.4 dB compared to the L2 norm at 4% FER and higher in a highly correlated channel. It can also be seen that the main performance loss in the UNC channel is due to the distorted tree search with the L1 norm based LSD algorithm as the recalculated final list $\mathscr{L}$ with the L2 norm gives the very similar performance. The recalculation of the

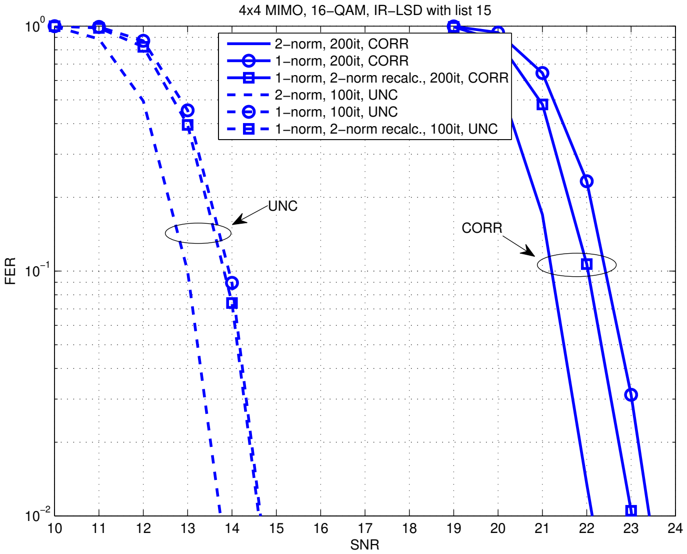


Fig 24. The performance of the IR-LSD based receiver with different norm modifications.

final list $\mathscr{L}$ with the L2 norm gives approximately a 0.4 dB gain in the CORR channel. The performance difference is due to more accurate $L_{\mathrm{D}}(b_k)$ approximation in (42).

3.5.1 L1 norm and Euclidean norm approximation

The distance calculation in the LSD algorithm is done in parts by summing the partial distances of each level together as shown in (40). The use of the L1 norm in the LSD algorithm reduces the number of required arithmetic operations, but results in performance degradation due to a different metric in the tree search and distorted LLR approximation. It is not trivial to compensate for the effect of the L1 norm on the tree search without adding complexity as each PAD calculation by the tree search algorithm should be approximated to the corresponding L2 norm value. The distorted LLR approximation can be compensated for by recalculating the final distance metrics with L2 norm, but complexity reduction of the L1 norm usage is then lost due to the added complexity. Another possibility is to approximate the L2 norm ED by scaling the final list L1 norm AD values properly, e.g., as done by a rather complex method in [173]. In order to illustrate the problem, the correlation between the L1 norm and L2 norm

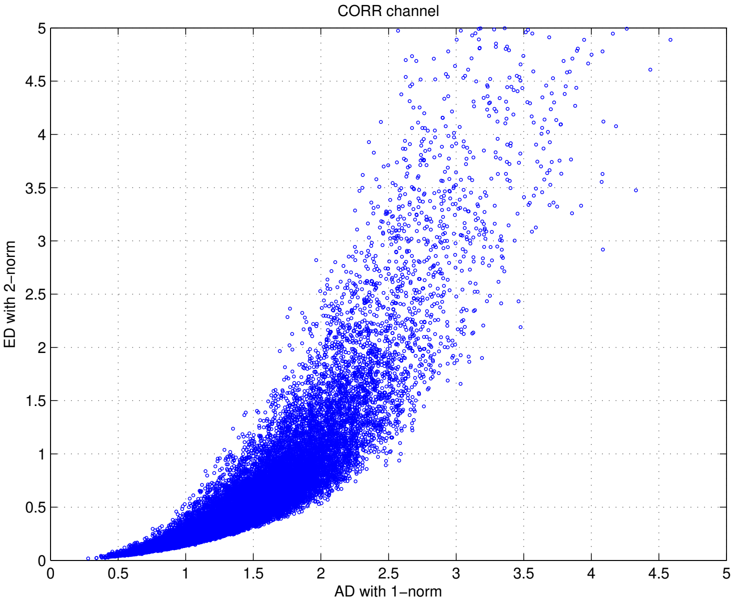


Fig 25. Measurement data showing the candidate list distance metric values with the L1 norm and L2 norm.

distance values are shown by plotting the samples of LSD algorithm final list distances calculated with both L1 and L2 norms in Figure 25. It can be seen that there is no exact method to transform the final list AD values calculated with the L1 norm into corresponding L2 norm ED values. However, the performance of the L1 norm based LSD can be improved by approximating the L2 norm based results. We propose a novel and simple method to compensate for the performance degradation due to the use of the L1 norm in the distance calculation. The proposed solution works especially in spatially correlated MIMO channels, where the use of the L1 norm in the LSD algorithm leads to a distorted LLR approximation as shown in Figure 24. The L2 norm approximation is proposed to be done by multiplying the L1 norm values with a predefined coefficient a and adding a constant b as

$$||\mathbf{x}||_2 = a||\mathbf{x}||_1 + b. \tag{51}$$

The coefficient a and the constant b can be determined for each channel scenario and different values can be used for different distance values. A determined scaling look-up-table (LUT) for a highly correlated TU channel, which is based on Figure 25, is listed in Table 12. The performance of different LSD based receivers with different norm calculation methods is studied via computer simulations. The performance examples of a 4×4 MIMO system with 16-QAM and with LSD based receivers with list size 15 and a turbo decoder in a highly correlated typical urban channel are shown in Figures 26, 27 and 28. The LSD algorithm tree search is implemented with either an L1 norm or L2 norm based decision metric calculation. The LLR approximation is calculated with a final list $\mathscr{L}$ that is based on the L2 norm, the L1 norm, the recalculated L2 norm or the proposed method to scale the AD values as in Table 12. It can be seen that the K-best-LSD with the L1 norm performs approximately 0.7 dB worse at 4% FER compared to the K-best LSD with the L2 norm. The performance of K-best LSD with the L1 norm improves approximately 0.4 dB when the proposed ED scaling is applied. The SEE-LSD with the L1 norm loses 1.2dB compared to the L2 norm based SEE-LSD, and the proposed L2 ED scaling improves the performance of the L1 norm based SEE-LSD by 0.4dB. The performance of IR-LSD with the L1 norm is 1.25dB worse compared to the L2 norm based IR-LSD, and the ED scaling improves performance by 0.25dB. Thus, it can be concluded that the ED scaling can be used to improve the LSD performance when the L1 norm is applied in the LSD algorithm. The required SNR values in the CORR channel for 4% FER performance with LSD based receivers with different norm methods are listed in Table 13.

Table 13. The required SNR of a 4×4 MIMO–OFDM system with 16-QAM and with different LSDs based receivers for 4% FER in the CORR channel.

Method	L2 norm	L1 norm	L1 norm with scaling
K-best-LSD	22.5dB	23.2dB (+0.7dB)	22.8dB (+0.3dB)
SEE-LSD	21.55dB	22.75dB (+1.2dB)	22.3dB (+0.75dB)
IR-LSD	21.55dB	22.8dB (+1.25dB)	22.55dB (+1.0dB)

Table 12. The determined coefficient a and constant b values.

Coefficient a	Constant b	Applied L1 norm value range
$a = 0$	$b = -0.1$	$\|\|\mathbf{x}\|\|_1 < 0.4$
$a = 0.29$	$b = -0.1$	$1.0 \geq \|\|\mathbf{x}\|\|_1 > 0.4$
$a = 0.80$	$b = -0.6$	$2.0 \geq \|\|\mathbf{x}\|\|_1 > 1.0$
$a = 2.40$	$b = -3.6$	$3.0 \geq \|\|\mathbf{x}\|\|_1 > 2.0$
$a = 4.13$	$b = -8.8$	$6.0 \geq \|\|\mathbf{x}\|\|_1 > 3.0$
$a = 5.0$	$b = -15.0$	$\|\|\mathbf{x}\|\|_1 > 6.0$

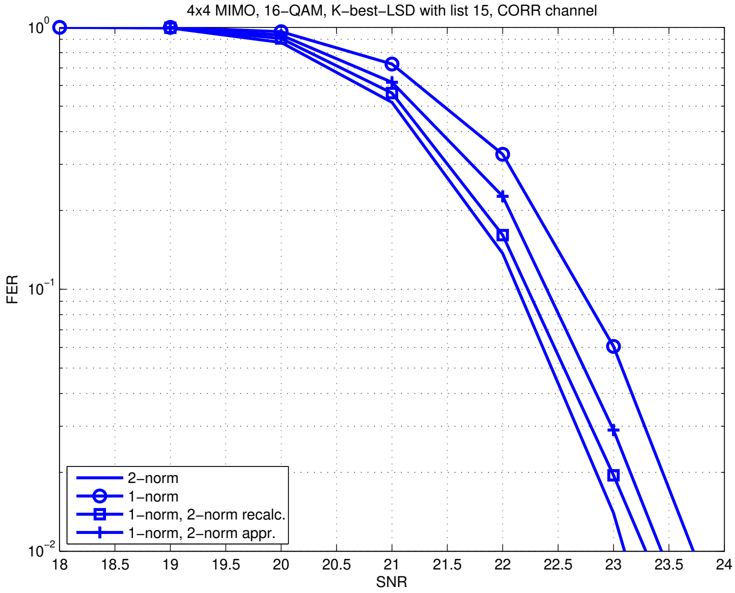


Fig 26. The performance of the K-best-LSD based receiver with different norm modifications.

The use of the L1 norm in the LSD algorithm decreases the complexity compared to the L2 norm based LSD algorithm with a cost of degraded performance. The proposed scaling method with an L1 based LSD algorithm increases the complexity as the final list $\mathscr{L}$ AD values have to be scaled as in (51). The complexity increase is, however, very small compared to the performance improvement. In order to illustrate the complexity difference, the complexity of the K-best-LSD algorithm with list size N_{cand} of 15 and with L2 norm, L1 norm and L1 norm with AD scaling, and LLR calculation for a 4×4 MIMO system with 16-QAM is listed in Table 14. The complexity is listed in required division (DIV), MUL, ADD and comparison (COMP) operations per detection. The use of the L1 norm in distance metric calculations results in 380 fewer MUL and ADD operations compared to the L2 norm. The ED scaling results in only 15 more MUL and ADD operations and 150 COMP operations, because only the final list candidates have to be scaled. The proposed scaling method can be implemented with a look-up-table (LUT) and multiply and accumulate (MAC) unit. The LUT is required to store the coefficients a and constants b, and the MAC unit is used to execute (51) for the LSD output list ED values. Parallel MAC units can also be used for faster implementation.

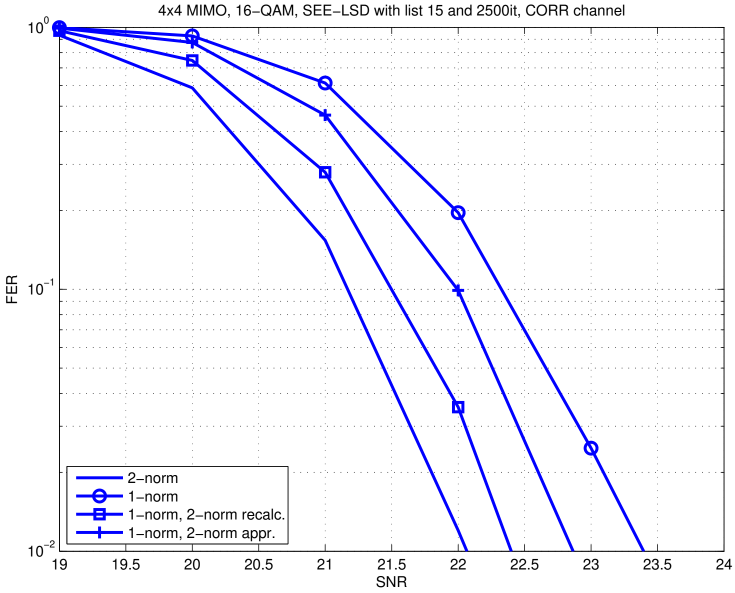


Fig 27. The performance of the SEE-LSD based receiver with different norm modifications.

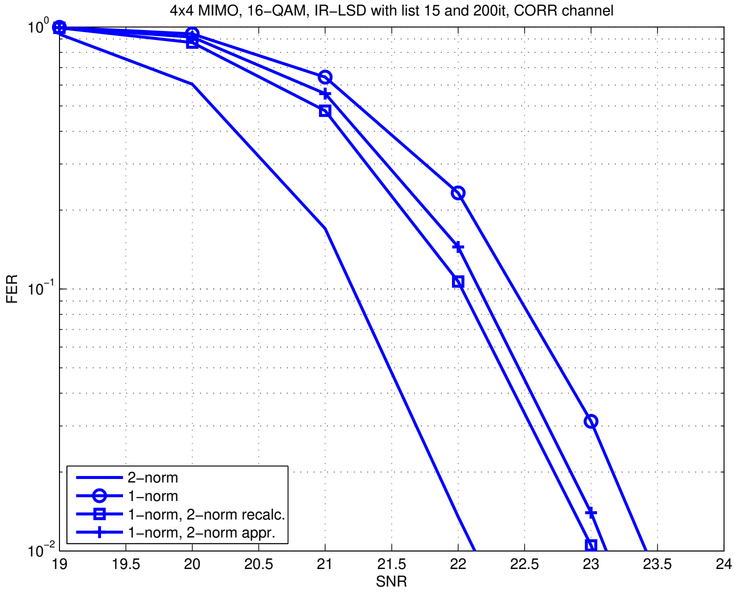


Fig 28. The performance of the IR-LSD based receiver with different norm modifications.

Table 14. The complexity of K-best-LSD with list size $N_{\text{cand}} = 15$ in a 4×4 MIMO system with 16-QAM.

	L2 norm	L1 norm	L1 norm with scaling
DIV	1	1	1
MUL	1915	1535	1550
ADD	1916	1536	1551

3.5.2 Conclusions

We studied the use of the L1 norm in the LSD algorithm tree search, which decreases the complexity of the search, but also degrades the performance of the LSD. We proposed to use a simple scaling of the LSD algorithm output list $\mathscr{L}$ distance metric values to approximate the L2 norm ED. We showed that the scaling method can be used to compensate for the performance loss due to the use of the L1 norm with minor additional complexity. The complexity increase is very small compared to the performance improvement.

3.6 Complexity and performance of an iterative receiver

The optimal joint receiver can be approximated by using an iterative receiver and soft-input soft-output (SfISfO) detector and decoder [4]. The performance of the system is increased to a certain extent with each global iteration (GI), where the soft reliability information is fed back to the detector from the decoder. However, the iterative receiver structure also increases the computational complexity of the receiver with each GI as more signal processing is done. The effect of the receiver convergence properties depend on multiple variables such as decoder iterations, channel code properties and channel realization [174]. Thus, it is not straightforward to determine the optimal receiver configuration and how much computing effort should used in detector, decoder and global iterations in total.

We studied the performance of the iterative receiver with variable GIs and their effect on the computational complexity by numerical examples. The numerical examples were executed for a 4 × 4 MIMO-OFDM system with 16-QAM and with the same system model parameters as in earlier sections with the following details. The LLR

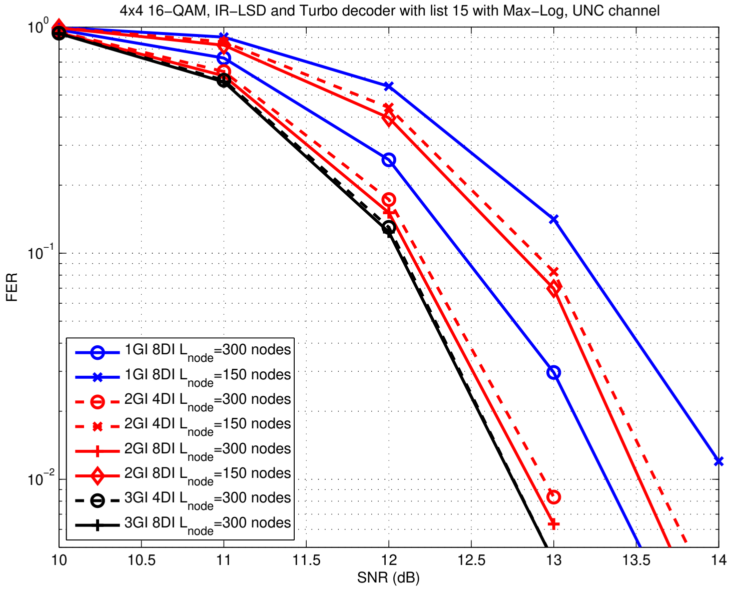


Fig 29. FER vs. SNR: Performance of the real IR-LSD based receiver in a 4×4 MIMO system with 16-QAM. ([138], published by permission of Elsevier).

clipping was applied with $L_{\text{max}} = 8$, the IR-LSD was operating with variable L_{node}, the max-log turbo decoder was operating with 4 or 8 decoder iterations (DI) and the iterative receiver was operating with 1–3 GIs. Performance examples of a real IR-LSD based receiver are shown in Figure 29. We can see that the 2nd GI improves the performance approximately 0.5 dB at FER 10^{-2} and there is not much performance gain with a 3rd GI. It can be also noted that 4 DIs are sufficient with 2 or more GIs and the increase in the IR-LSD L_{node} from 150 nodes to 300 nodes improves performance by approximately 0.7 dB. Similar performance behavior could also be seen from the numerical examples with a K-best-LSD based receiver. Typically the performance and the complexity of the receiver are both important measures, when designing a system. Therefore, we also calculated the required computational complexity for different receiver configurations to compare and determine the most efficient configuration. The complexity calculation includes the required arithmetic operations of LSDs and the max-log turbo decoder, which are MUL, ADD, COMP and DIV, with a given number of iterations for the reception of one OFDM symbol, i.e., 2400 transmitted bits. The memory requirements and word length requirements were omitted from the calculations. In order to get an explicit complexity description, we approximated the computational complexity of the required arithmetic operations according to their relative complexity units (CU) to NAND2 operation, i.e., according to how many 1-bit NAND2 ports are required for the corresponding 1-bit operation. The approximations used are listed in Table 15. The approximation is based on the authors' experience and from [175].

The performance and complexity of K-best-LSD or IR-LSD and max-log turbo decoder based receivers with different configurations are shown in Figures 30 and 31, respectively. The figure curves describe the required SNR for a 4 % target FER and the corresponding complexity of the receiver with the variable LSD L_{node} or list size N_{cand} in calculated complexity units. It can be seen that the complexity from the decoder iterations dominates the total complexity and that there is only minor performance gain with 8 DI compared to 4 DI when 2 or more GIs are executed. Also the performance gain of a 3rd GI with the given system configuration is minor compared to the additional computational complexity required. The cost of increased number of visited nodes by the LSD is minor compared to the additional performance gain with both IR-LSD and K-best-LSD. Thus, we can say that with the given system configuration it makes sense to use the additional computational cost for LSD to achieve a better soft output approximation. The use of the 2nd GI with lower amount of decoder resources (4DI) is also justified.

Table 15. The number of 1-bit NAND2 ports used for the corresponding 1-bit arithmetic operation. ([138], published by permission of Elsevier).

Operation	MUL	ADD	COMP	DIV
Required NAND2's	12	5	4	30

3.6.1 Conclusions

We studied the performance of the iterative receiver with variable GIs and their effect on the computational complexity. The results indicated that with the typical system configuration it is beneficial to use the additional computational cost for the LSD algorithm to achieve a better soft output approximation, i.e., study more tree nodes in the algorithm search. The use the 2nd GI with a lower amount of decoder resources is also justified. However, it should be noted that the numerical examples only give some guidelines to the design and they depend on system configuration.

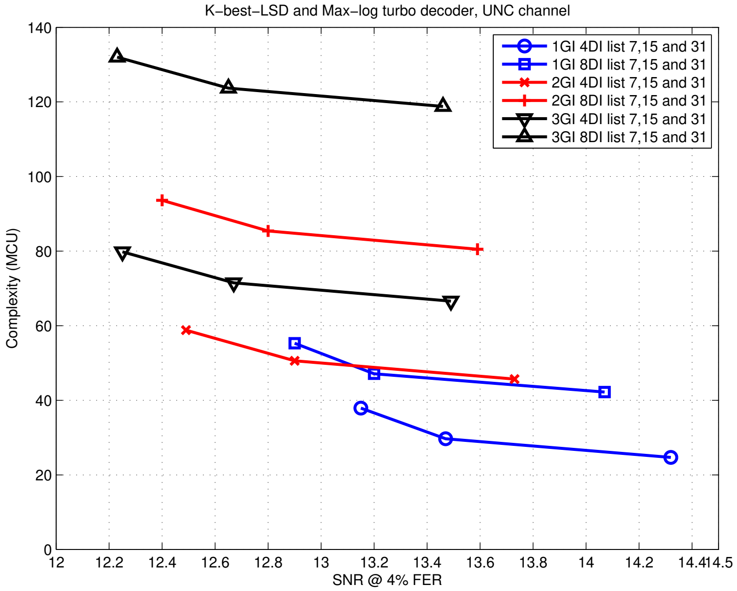


Fig 30. Required SNR for 4% target FER vs. required operation complexity with different K-best-LSD based configurations. ([138], published by permission of Elsevier).

3.7 Summary

In this chapter, we introduced and studied different methods related to the implementation of a list sphere detector. We studied six significant challenges, namely, limiting the dynamic range of the soft output log-likelihood ratio (LLR) values to the decoder, comparing the real and complex signal models in the LSD, limiting the search complexity of the LSD algorithm, applying a IR-LSD memory sphere radius to lower the required memory access, applying the L1 norm in the LSD algorithm, and analyzing the complexity and performance tradeoffs of an iterative receiver.

We introduced two simple LLR clipping methods to process the soft output LLR $L_{\mathrm{D}}(b_k)$ and studied the impact of the methods on the performance of a coded system. We showed that the LLR clipping enhances the performance of LSD based systems with low list size and, thus, lowers the required list size N_{cand}. We proposed to use Method 1 as it ensures good performance and requires less dynamic range for the $L(b_k)$ before clipping. The coding rate had a small effect on the optimal clipping values, but the differences are rather small, and in practice, $L_{\max} = 8$ gives good results.

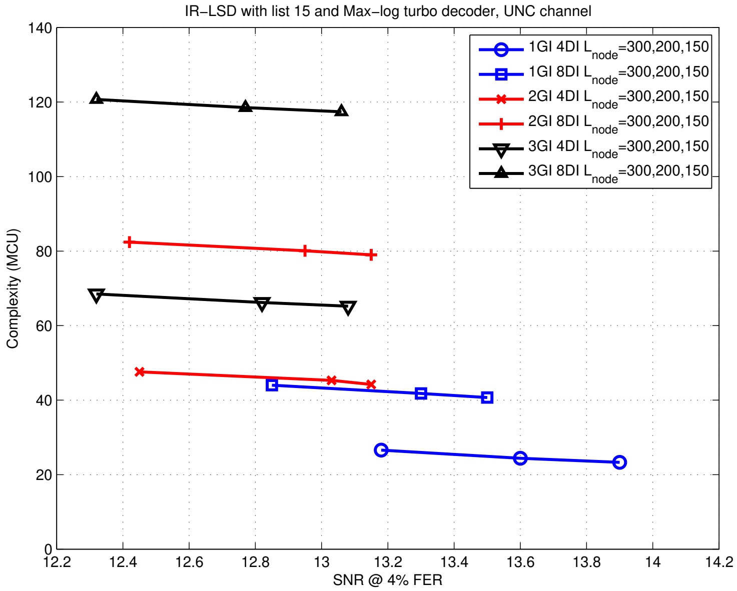


Fig 31. Required SNR for 4% target FER vs. required operation complexity with different IR-LSD based configurations. ([138], published by permission of Elsevier).

We studied the difference between using the real valued or complex valued signal model in the LSD algorithm. The real valued signal model is clearly the better choice to be applied with LSD algorithms given the number of visited nodes and the complexity difference in the required operations in one node check.

We proposed two methods to limit and fix the search complexity of the SEE-LSD and the IR-LSD algorithms: limited search (LS) and scheduled search (SS) methods. The results show clearly that both considered sequential search algorithms work well when the sequential search is limited. Both the SS and the LS methods are feasible for implementation, and a minor performance gain can be achieved with the more sophisticated and complex SS method.

We proposed to use a novel memory sphere radius C_{mem} with the IR-LSD algorithm to decrease the number of stored candidates and the complexity of the required minimum search during the algorithm tree search. We showed that the average numbers of visited nodes decreases with the proper selected memory sphere radius without loosing performance.

We studied the use of the L1 norm in the LSD algorithm tree search, which decreases the complexity of the search, but also degrades the performance of the LSD. We proposed to use a simple scaling of the LSD algorithm output list $\mathscr{L}$ distance metric values to approximate the L2 norm ED. We showed that the scaling method can be used to compensate for the performance loss due to the use of the L1 norm with minor additional complexity.

We studied the performance of the iterative receiver with a variable number of GIs and their effect on the computational complexity. The results indicated that with the typical system configuration it is beneficial to use the additional computational cost for the LSD algorithm to achieve a better soft output approximation, i.e., study more tree nodes in the algorithm search. The use the 2nd GI with a lower amount of decoder resources is also justified.

4 Architecture design

The architecture design is a key point in efficient implementation of an algorithm. In this chapter, we identify and introduce the key functional units of the considered detection algorithms, and design efficient architectures for MIMO–OFDM systems. The possibilities for parallelism and pipelining in the microarchitecture units are introduced and analyzed.

4.1 LMMSE detector

The architecture design is highly dependent on the specific application. In an OFDM system, the detection has to be done separately for each subcarrier. Thus, the detector coefficient matrix $\mathbf{W}$ in (15) also has to be calculated separately for each subcarrier. The calculation of the coefficient matrix $\mathbf{W}$ is computationally the most complex part of the LMMSE detector and, more specifically, the calculation of the matrix inversion via QRD as presented in Section 2.4. The dimensions of the required matrix inversion depend on the number of receive antennas N_{R}. The coefficient matrix $\mathbf{W}$ needs to be updated as the channel changes, i.e., according to the channel coherence time T_{c}. The total complexity of the LMMSE detector depends mainly on these variables. The high level architecture of the LMMSE detector is presented in Figure 32. The architecture consists of an LMMSE coefficients calculation block, adaptive scaling blocks, a data detection block, and memory blocks. The LMMSE detector coefficient matrix calculation in (15) requires several matrix operations such as matrix-matrix multiplications, QRD, and back substitution or inversion of a triangular matrix. The architectural design of matrix operations in the literature is often based on systolic array structures with communicating processing elements (PEs) [75, 76]. We design the architecture and the operations using triangular, two dimensional and linear systolic arrays.

The matrix-matrix multiplication can be implemented using two-dimensional systolic array architecture or a memory shared linear systolic array architecture [70]. The two-dimensional array enables a fast and parallel data flow. The multiplication of two matrices of dimensions $A \times B$ and $B \times C$ requires AC PEs with MAC operation as an $A \times C$ two-dimensional array [76]. The delay of the array is $(A+B+C-1)\tau_{\mathrm{mac}}$ clock cycles, where τ_{mac} is the delay of one PE in clock cycles. The linear array requires less

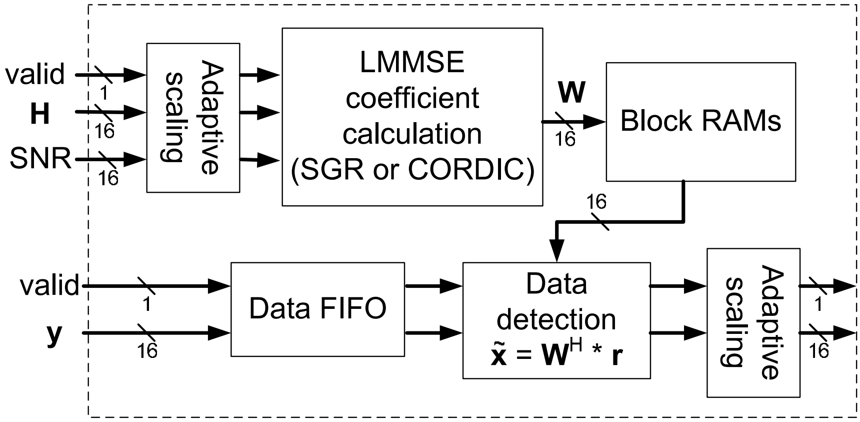


Fig 32. The LMMSE detector high level architecture. ([142], published by permission of IEEE).

resources in hardware implementation, but time-shared scheduling increases the delay of the array. In a linear array one PE is used to compute all elements of a column, and, thus, only C PEs are required [76]. The delay of the linear array is $(AB+C)\tau_{\text{mac}}$ clock cycles for the whole multiplication.

A conventional method for computing the QRD is to use a simple and highly parallel triangular array architecture [75, 76]. A triangular array architecture enables simple data flow, high throughput with pipelining, and is feasible for matrices with low dimensions, e.g., for 2×2 matrices. Both the algorithm for inversion of a triangular matrix [157] and the back substitution algorithm [70] can be implemented using a triangular array architecture. The matrix inversion of an $A \times A$ square matrix can be computed using a cascade of two triangular arrays with total of $A(A+1)$ PEs [157]. The total time required for matrix inversion with such an architecture is $5A\tau_{\text{inv}}$ clock cycles, where τ_{inv} is the delay of one PE in the array [157].

However, the triangular architecture has certain drawbacks, such as a growing number of required PEs with increasing matrix dimensions and, thus, a lack of easy scalability. As an alternative structure, a linear array architecture could be considered for larger systems. A derivation of a linear QR array from a triangular QR array has been presented, e.g., in [78, 176]. A linear array mapping of the triangular matrix inversion algorithm has been presented in [177]. The linear array architecture for an $A \times A$ matrix inversion employs only $2A$ PEs [78]. The total latency of such a linear array architecture

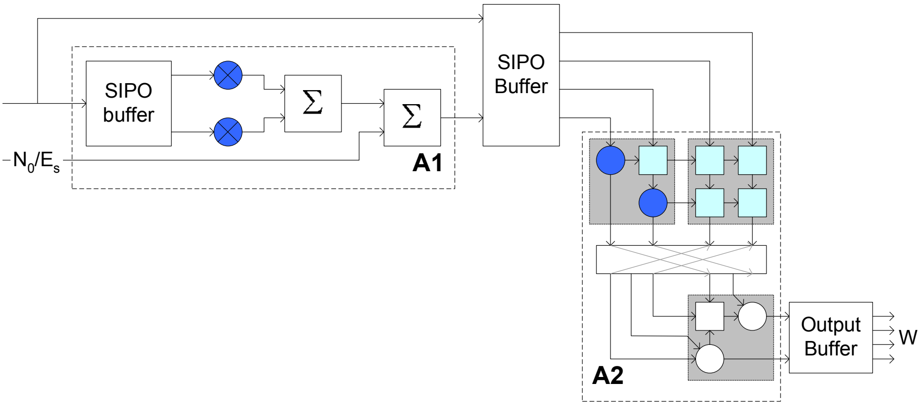


Fig 33. The CORDIC based LMMSE detector architecture for a 2×2 MIMO system. ([140], published by permission of IEEE).

is $(4A^2 + 2A - 2)\tau_{\text{inv}}$ clock cycles. A detailed description of the designed architectures for the calculation of the coefficient matrix $\mathbf{W}$ is presented for both CORDIC and SGR based designs in the following subsections. Architecture solutions are designed for 2×2 and 4×4 MIMO systems.

4.1.1 CORDIC Based Design

The CORDIC based LMMSE detector architecture for a 2×2 MIMO system is illustrated in Figure 33. The matrix $\mathbf{A}$ from (17) is formed in part A1 using a linear array of complex multipliers and summation blocks. The complex multipliers are used in parallel to calculate one element of $\mathbf{A}$ in one time interval. The matrices $\mathbf{A}$ and $\mathbf{B}$ from (17) are then fed via a serial-input parallel-output (SIPO) buffer to part A2 which consists of two systolic arrays. The calculation of the matrices $\mathbf{R}$ and $\mathbf{Q}^{\mathrm{H}}\mathbf{B}$ from (19) is carried out in the upper part of A2 with CORDIC based triangular and two-dimensional systolic arrays. Then the lower triangular systolic array applies the back substitution algorithm to form the desired matrix $\mathbf{X} = \mathbf{W}$. The architecture presented in Figure 33 does not require much control logic and the mapping of data flow is relatively easy. The latency of the architecture is very feasible for high data rate applications and it enables efficient pipelining [157]. The applied architecture is designed for systems with rather low matrix dimensions, i.e., a 2×2 MIMO system, as the complexity of the triangular array architecture grows dramatically with increasing matrix dimensions. Thus, we designed a less complex architecture to be used with larger matrix dimensions. The QRD

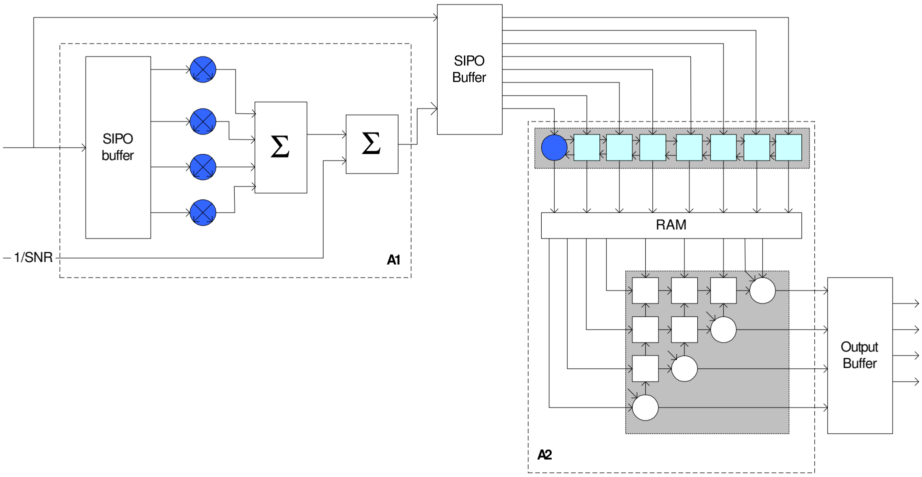


Fig 34. The CORDIC based LMMSE detector architecture for a 4×4 MIMO system. ([140], published by permission of IEEE).

array, which is the most complex part of the design, is replaced with a less complex linear systolic array. The linear array requires more control logic and the overall latency of the calculation of detector coefficients is higher, but the required complexity is less compared to the triangular array architecture. The CORDIC based LMMSE detector architecture for a 4×4 MIMO system is illustrated in Figure 34. The triangular array used for back substitution is not replaced with a linear array, because the complexity of the array is only a fraction compared to the complexity of the QRD array. Also the matrix multiplication part A1 includes four parallel multipliers to enable the calculation of one element of the matrix $\mathbf{A}$ in one time interval.

CORDIC and Back Substitution Array

The CORDIC based QRD array cells and the used word lengths are shown in Figure 35. The array contains two types of cells, the round vectoring cells and the square rotating cells. The round boundary cell performs the vectoring operation, i.e., it computes the angles needed for annihilation of the incoming data samples. Two real CORDIC blocks are needed for complex implementation. The boundary cell sends the angle values to the inner square cells in the same row. The inner square cell calculates the new rotated sample values based on the angle values given from the boundary cell. Three real CORDIC blocks are needed for each block using complex valued arithmetic. The dataflow in the array is illustrated for 2×2 and 4×4 MIMO system designs in Figure 36. In the 2×2 MIMO system design the values of the upper triangular matrix $\mathbf{R}$ can be read from the array after all samples are inside of the array and the values of the array $\mathbf{B}$ are the result of multiplication $\mathbf{Q}^{\mathrm{H}}\mathbf{B}$. In the time-shared design for a 4×4 MIMO system, the implementation requires several shift registers to handle multiple values simultaneously. The complexity of the CORDIC-array is determined by the number of CORDIC iterations and the word length used [73, 158]. The back substitution array cells and the used word lengths are illustrated in Figure 37. The triangular array structure includes two different types of cells. The boundary round cell performs a complex by real division operation. The inner cell contains a complex multiplication and an arithmetic subtraction operations. the back substitution block must be connected correctly to the QR and the input samples must be organized in certain way. A new sample is latched in every clock interval in a 2×2 array with twice the clock rate as with a QRD array. The problem with a 4×4 array is that results from QR are ready at very different times and they must be stored to a memory. Thus, random access memory (RAM) blocks are used to store both $\mathbf{R}$ and $\mathbf{Q}^{\mathrm{H}}\mathbf{B}$ real and imaginary values. The values are then read simultaneously to the 4×4 array. The overall complexity of the back substitution array is relatively low compared to the QRD array and it is dominated by the reciprocal divider blocks.

4.1.2 SGR Based Design

The SGR based LMMSE detector architecture for the 2×2 MIMO system is presented in Figure 38. The architecture is designed to support pipelining and to be feasible for high data rate applications. Thus, two-dimensional arrays are used for matrix multiplications and conventional fast triangular arrays for the matrix inversion. The matrix $\mathbf{A}$ in (17) is calculated in part A1 using a two dimensional multiplication array. The dataflow of such an array is illustrated, e.g., in [76]. The matrix inversion by QRD and triangular matrix inversion are done in part A2 using a cascade of two triangular arrays [157]. The lower triangular array in part A2 also executes the calculation of $\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{Q}_{\mathrm{A}}^{\mathrm{H}}$. The two dimensional array in the A3 part, which is similar to the array in part A1, calculates the matrix multiplication of terms $\mathbf{A}^{-1}$ and $\mathbf{B}$ in (28). The architecture presented in Figure 38 is more suitable for systems with rather low matrix dimensions. We also designed an architecture with linear systolic arrays for increasing matrix dimensions, e.g., 4×4 MIMO system and larger. The SGR based LMMSE detector architecture for

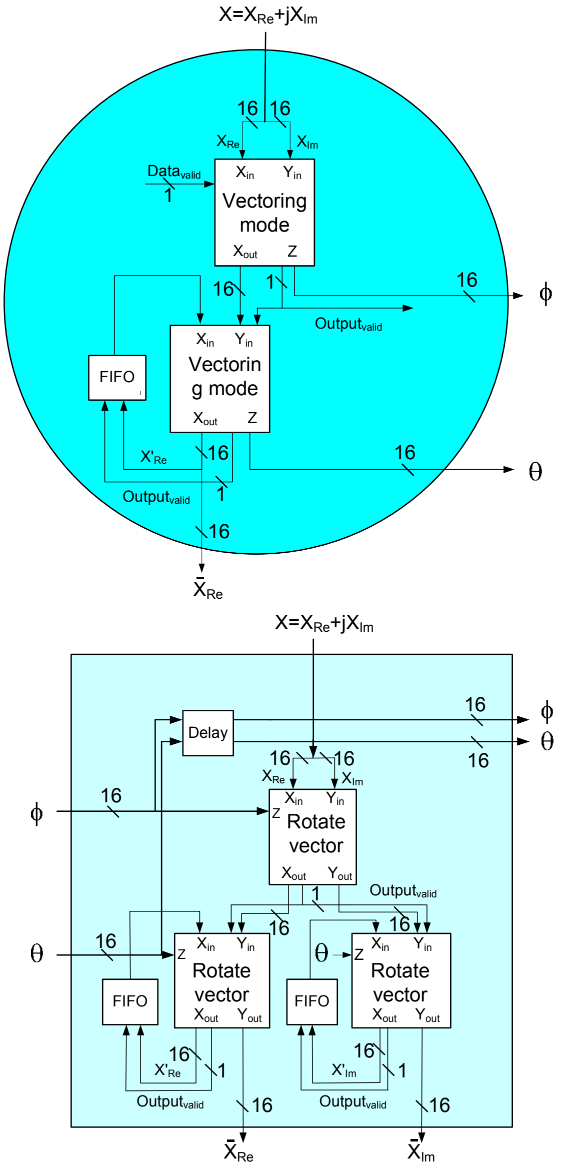


Fig 35. Hardware realization of the CORDIC vectoring and rotating cells. ([140], published by permission of IEEE).

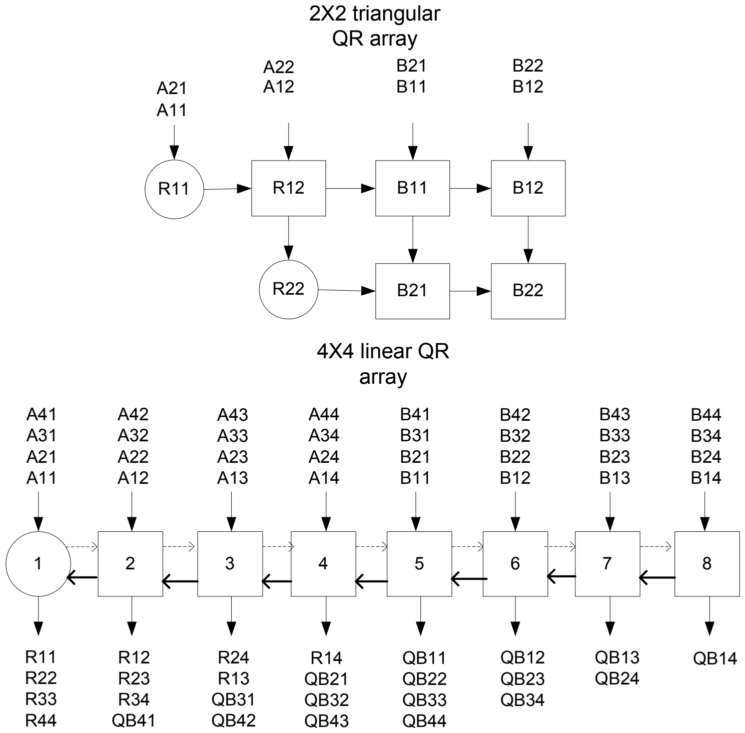


Fig 36. The dataflow of the CORDIC 2×2 and 4×4 array.

a 4×4 MIMO system is presented in Figure 39. A linear array architecture is applied for each part in Figure 39. The linear structure used for both matrix multiplications in parts A1 and A3 decreases the required number of PEs from 16 to 4. Also the QRD and triangular matrix inversion arrays in part A2 are replaced with a linear array [78], which decreases the number of PEs from 20 to 8. The linear array requires more control logic and the overall latency of calculation of the detector coefficients is much higher, but the complexity saving compared to a triangular array grows dramatically with increasing matrix dimensions. The dataflow of a linear array architecture is discussed, e.g., in [78, 177, 178].

SGR and Triangular Matrix Inversion Array

The systolic array architecture for the SGR algorithm includes three different kinds of cells as shown in Figures 38 and 39 for 2×2 and 4×4 MIMO systems, respectively. In the architecture the round boundary cell is only a delay element except for the last

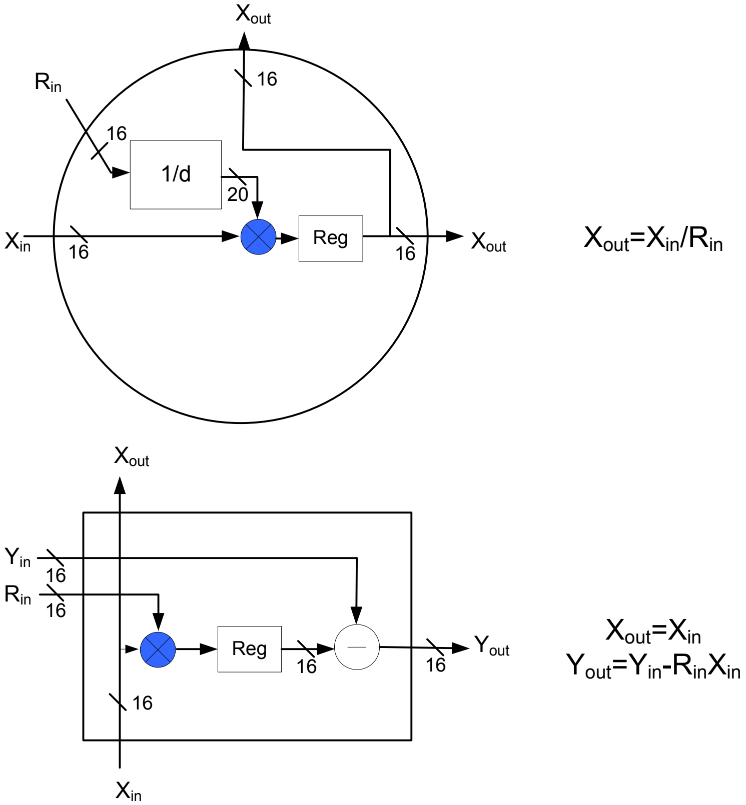


Fig 37. Hardware realization of the Back substitution array cells. ([140], published by permission of IEEE).

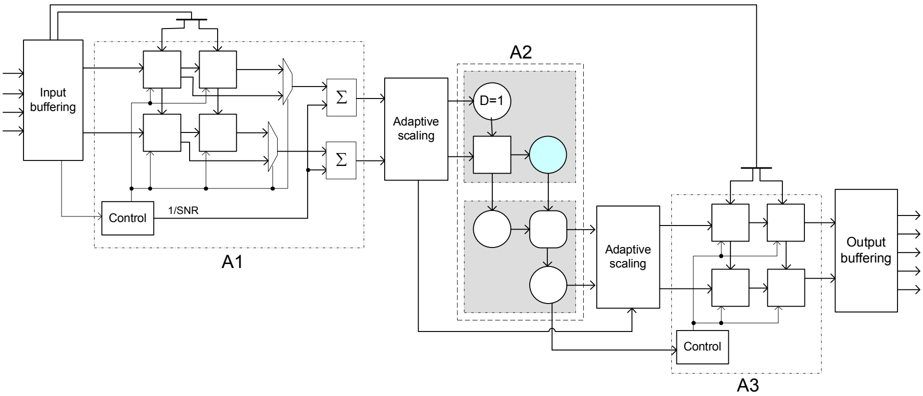


Fig 38. The SGR based LMMSE detector architecture for a 2×2 MIMO system. ([140], published by permission of IEEE).

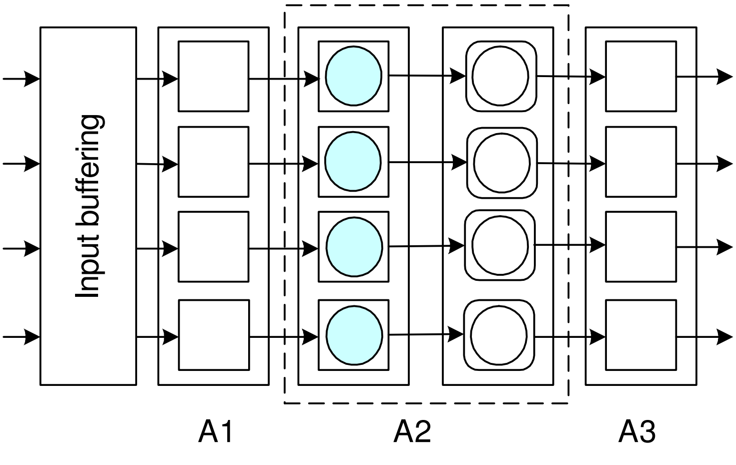
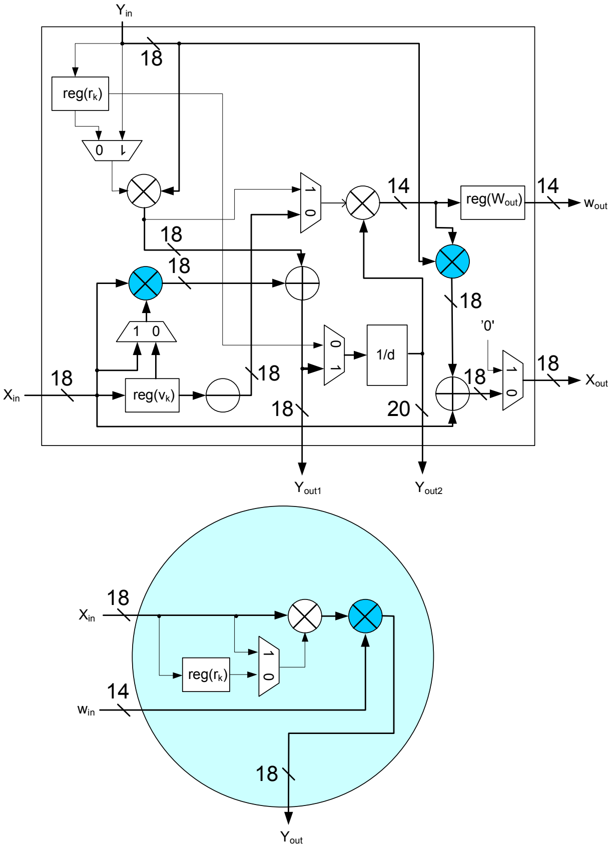


Fig 39. The SGR based LMMSE detector architecture for a 4×4 MIMO system. ([140], published by permission of IEEE).



Mode1 (Yin with '*')

$Y_{out1} \leftarrow Y_{in} \cdot Y_{in} + X_{in} \cdot {}_{'*'} X_{in}$

$X_{out} \leftarrow 0$

$r_k \leftarrow Y_{in}, v_k \leftarrow X_{in}$

$w_{out} \leftarrow \frac{Y_{in} \cdot Y_{in}}{Y_{out1}}$

$Y_{out2} = 1 / Y_{out1}$

Mode2 (Yin without '*')

$X_{out} \leftarrow -\frac{v_k}{r_k} \cdot Y_{in} + X_{in}$

$Y_{out1} \leftarrow r_k \cdot Y_{in} + v_k \cdot X_{in}$

Mode1(Xin with '*')

$Y_{out} \leftarrow w_{in} \cdot X_{in} \cdot X_{in}$

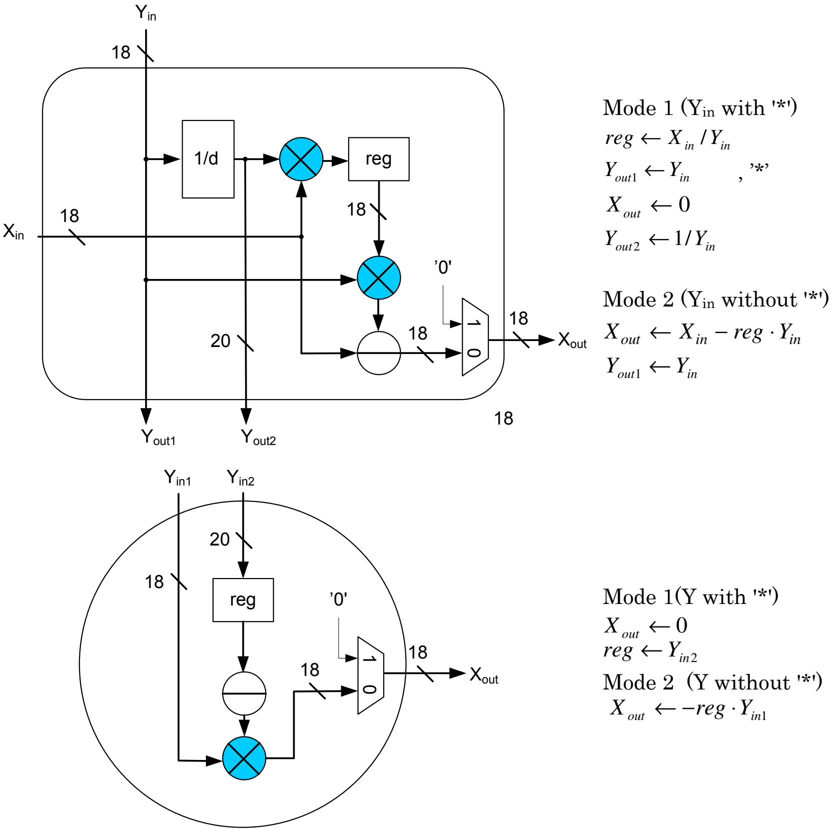
$r_k \leftarrow X_{in}$

Mode2(Xin without '*')

$Y_{out} \leftarrow w_{in} \cdot r_k \cdot X_{in}$

Fig 40. Hardware realization of the SGR array cells. ([140], published by permission of IEEE).

shaded cell. Hardware realizations and the used word lengths of the last round boundary cell and the square internal cell are presented in Figure 40. The main operations of the SGR algorithm are executed in the square internal cell. Each cell consists of arithmetic blocks such as divider, multipliers, adders, multiplexers, and registers. The shaded blocks and the bold lines depict complex signal representation. The complexity of the SGR array is dominated by the complex reciprocal divider block, which is executed in the square internal cell. It should be noted that all the cells in the linear array in Figure 39 include both the boundary cells and the square internal cell. The triangular matrix inversion array cells and the used word lengths are presented in Figure 41. The array includes two different kinds of cells. The linear array architecture cells include the operations of both cells. The design is based on [157] with certain simplifications.



Mode 1 (Y_{in} with '*')
$reg \leftarrow X_{in} / Y_{in}$
$Y_{out1} \leftarrow Y_{in}$, '*'
$X_{out} \leftarrow 0$
$Y_{out2} \leftarrow 1 / Y_{in}$

Mode 2 (Y_{in} without '*')
$X_{out} \leftarrow X_{in} - reg \cdot Y_{in}$
$Y_{out1} \leftarrow Y_{in}$

Mode 1(Y with '*')
$X_{out} \leftarrow 0$
$reg \leftarrow Y_{in2}$
Mode 2 (Y without '*')
$X_{out} \leftarrow -reg \cdot Y_{in1}$

Fig 41. Hardware realization of the triangular matrix inversion array cells. ([140], published by permission of IEEE).

The round cells do not need to calculate the reciprocal division operation, because the required operation is already calculated in another cell and can be directed to the cell as $Y_{\text{in}2}$. This decreases significantly the complexity of the array. The complexity of the array is dominated by the reciprocal divider block, which is executed in the square internal cell. The dataflow of the 2×2 SGR and triangular matrix inversion array is illustrated in Figure 42. Matrix $\mathbf{A}$ is fed to the QRD array in rows in every clock interval. It is followed by an identity matrix $\mathbf{I}_2$ to form the matrix $\mathbf{Q}_\text{A}$. The lower array calculates the inversions of the matrices and the output of the array is the matrix $\mathbf{U}^{-1}\mathbf{Q}_\text{A}^\text{H}$.

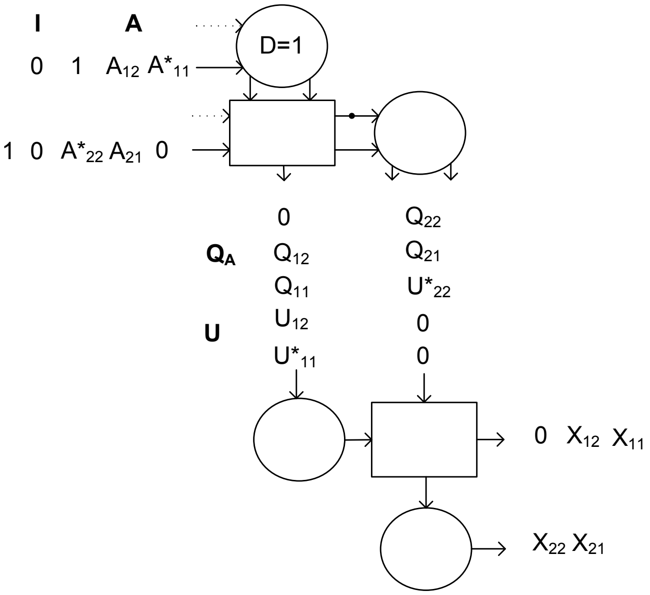


Fig 42. The dataflow of SGR and triangular matrix inversion array for a 2×2 MIMO system.

4.1.3 Scalability

The designed architectures are scalable to be used with different antenna configurations, but the number of PEs in the arrays have to be modified according to the applied configuration. The required dimensions of the systolic array architecture are related to the applied number of transmit antennas N_{T} and receive antennas N_{R}. The dimensions of the channel matrix $\mathbf{H} \in \mathbb{C}^{\mathbf{N}_{\mathrm{R}} \times \mathbf{N}_{\mathrm{T}}}$ determine the number of PEs required in the architecture used for the matrix-matrix multiplications in parts A1 and A3 of the architecture figures. The two-dimensional and linear arrays can then be derived easily for the applied system as discussed earlier in section 4.1. The matrix inversion of the $\mathbf{A} \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{R}}}$ is calculated in part A2 as shown in the architecture figures. The size of the inversion array is dependent on the dimensions of the $\mathbf{A}$, i.e., the number of receive antennas N_{R}. Thus, the required triangular or linear array can be easily scaled for the applied system.

4.2 List sphere detector

A high level architecture of the list sphere detector consists of the preprocessing unit, the LSD algorithm unit and the LLR calculation unit as illustrated in Figure 3. The units in the LSD architecture have different performance requirements that should be taken into account in the architecture design. The preprocessing unit, which calculates the decomposition of the channel matrix **H** for each OFDM subcarrier, recalculates the operations as the channel realization chances, i.e., according to the channel coherence time T_c. The LSD algorithm unit, however, executes the tree search for each OFDM subcarrier within the OFDM symbol interval and the LLR calculation unit then calculates the soft outputs also for each subcarrier in the OFDM symbol interval. Thus, typically the preprocessing unit outputs can be used in the LSD algorithm for several consecutive OFDM symbols and the latency requirements are not as strict as with the other units in the LSD architecture. It should also be noted that multiple units can be used in parallel to support higher data rate requirements in the OFDM system as the processing operations of separate subcarriers are independent. In the following subsections, we design architectures for the SQRD preprocessing algorithm [163], for the considered LSD algorithms and for the LLR calculation unit.

4.2.1 SQRD algorithm

The SQRD algorithm high level architecture is illustrated in Figure 43. The architecture operates in a sequential fashion, and calculates one row of the **R** and one column $\mathbf{q}_i$ of the **Q** in each iteration. The norm calculation unit calculates the channel matrix column norms, which are used to determine the initial permuting order **p** of the columns. The norm calculation requires a total of M_T^2 MUL operations, and, thus, different levels of parallelism and pipelining can be applied for the microarchitecture of the unit. The control logic unit defines the permutation order **p** of columns at iteration i as $i = 1 \dots M_\mathrm{T}$, and controls the calculation units and the memory access. The memory unit is used for storing the **Q** and **R** matrices during the decomposition. The registers are used to temporarily store the currently used rows of **R** and columns of **Q**, and the norm values. The actual calculation of the diagonal element $R_{i,i}$ and the column $\mathbf{q}_i$ is executed in the calculation unit, which requires a square-root, a reciprocal division operation and M_R MULs. Parallelism and pipelining can be applied in the MUL operations. The iterative update unit updates the elements in $R_{i,k}$, the columns $\mathbf{q}_k$, and the norm values $|\mathbf{h}_k|^2$,

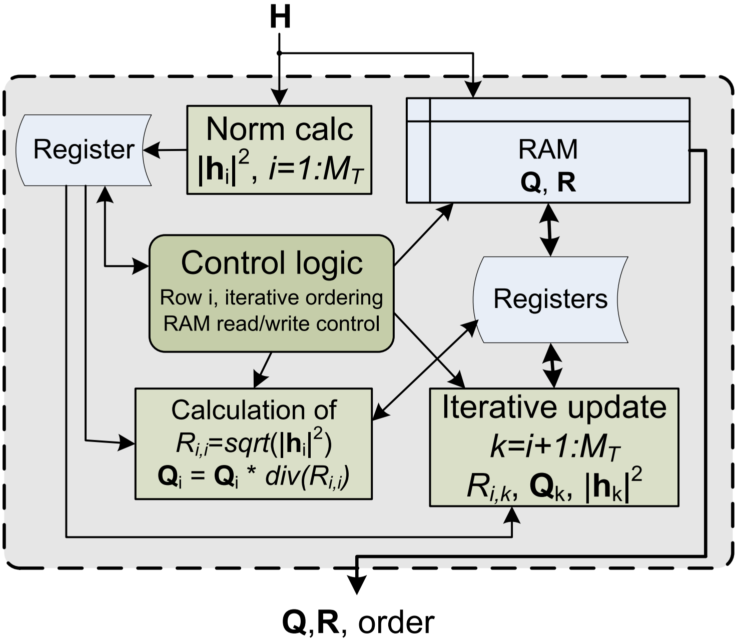

H
Register
Norm calc
$|\mathbf{h}_i|^2$, $i=1:M_T$
RAM
Q, **R**
Control logic
Row i, iterative ordering
RAM read/write control
Registers
Calculation of
$R_{i,i}=sqrt(|\mathbf{h}_i|^2)$
$\mathbf{Q}_i = \mathbf{Q}_i * div(R_{i,i})$
Iterative update
$k=i+1:M_T$
$R_{i,k}$, $\mathbf{Q}_k$, $|\mathbf{h}_k|^2$
Q,**R**, order

Fig 43. A high level architecture of a sorted QR decomposition unit.

where $k = 1, \ldots, M_\mathrm{T}$. The update of the variables can be carried out by MAC units, but the number of computations depend on the current iteration i. We designed an efficient time-sharing microarchitecture for the calculations, which enables different levels of parallelism and pipelining, and it is illustrated in Figure 44. The parallel MAC units are time-shared to calculate first the $R_{i,k}$ variable with a given k, and then the column $\mathbf{q}_k$ and the norm value $|\mathbf{h}_k|^2$ are updated. The architecture calculates iteratively all k values. As the number of different values assigned for the parameter k varies depending on the decomposition phase, the maximum efficient level of parallelism is to use M_T MAC units. The presented architecture calculates one decomposition at a time and a new decomposition can be started as the previous values are read from the RAM. Another alternative way to design the architecture would include M_T parallel processing units, where each of the units executes the processing for a single column of the decomposed matrix. Such an architecture can be pipelined to enhance the processing speed. However, more resources are also needed, and, thus, we decided to proceed with the presented architecture.

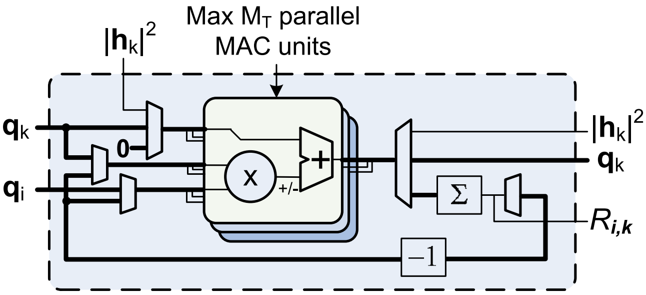


Fig 44. The designed microarchitecture for the SQRD iterative update unit. ([147], published by permission of IEEE).

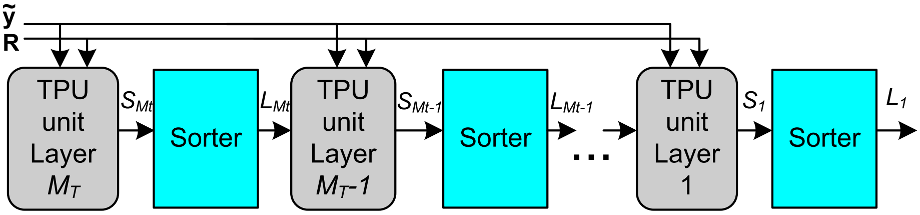


Fig 45. The pipelined architecture for the K-best-LSD algorithm.

4.2.2 K-best-LSD algorithm

The K-best-LSD algorithm is based on a breadth first search strategy and it prunes a fixed number of nodes in the search when $C_0 = \infty$. These characteristics make the algorithm very suitable for hardware implementation with parallel and pipelined architecture. An architecture for the K-best-LSD algorithm is shown in Figure 45. The architecture consists of separate tree pruning units (TPUs) and sorter units, which are combined in a sequential fashion to enable an efficient pipeline structure. There are a total of M_T TPUs in the architecture, where each TPU executes the calculation of (40) with all the possible symbols for the partial candidates in the corresponding tree search layer i in Algorithm 3. The sorting unit is required after the TPU in the ith layer if the number of output partial candidates $|\mathscr{S}_i|$ is larger than the applied variable K, i.e., $K < |\mathscr{S}_i|$. The sorting unit sorts K smallest partial candidates according to their EDs and they are given as output $\mathscr{L}_i$ to the next TPU layer. After the last pipeline stage, the final candidate list $\mathscr{L}_1$ is given as an output.

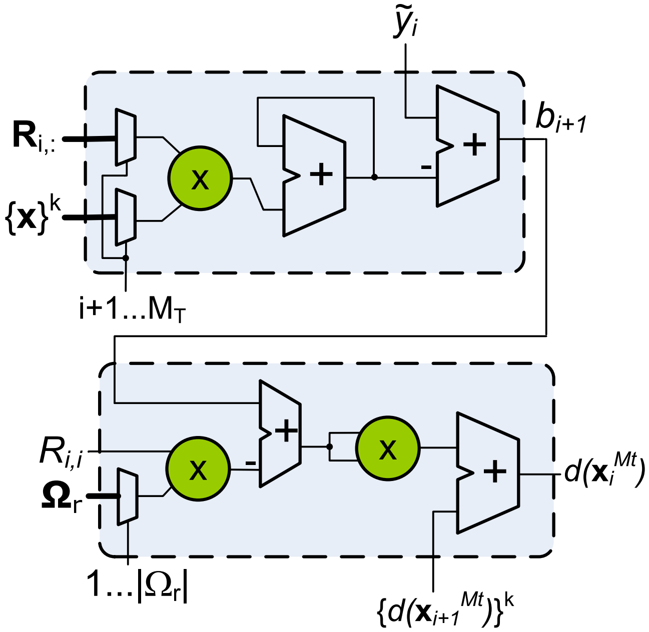


Fig 46. The K-best-LSD TPU microarchitecture.

TPU

The K-best-LSD algorithm TPU microarchitecture is illustrated in Figure 46. The TPU in Figure 46 calculates the node extensions with all possible symbols Ω_r for a partial candidate $\{\mathscr{L}_i\}^k$ given as input to the unit, and a total of $|\mathscr{L}_i|$, i.e., up to K, partial candidates are pruned. The TPU, which consists of two subunits, calculates first the $b_{i+1}(\mathbf{x}_{i+1}^{M_T})$ for the particular partial candidate as in (40) and then the PED with all possible symbol extensions Ω_r. The number of required operations depends on the search layer i: the calculation of $b_{i+1}(\mathbf{x}_{i+1}^{M_T})$ requires $M_T - i$ MUL and $M_T - i - 1$ ADD operations and the PED calculation requires $|\Omega_r|$ MUL and ADD operations. It should also be noted that no operations are required in the first layer in the calculation of $b_{i+1}(\mathbf{x}_{i+1}^{M_T})$. Both subunits can be applied with parallelism and/or pipelining to enhance the processing time of a single partial candidate $\{\mathscr{L}_i\}^k$. Also multiple partial candidates can be pruned in parallel by applying multiple parallel calculation units of the one shown in Figure 46.

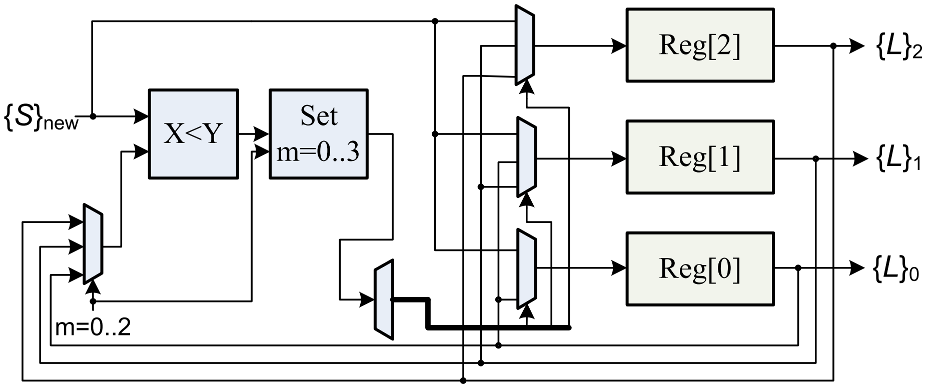


Fig 47. The microarchitecture for the insertion sorter unit with $n = 3$.

Sorting unit

The sorting unit, which is executed after the TPUs, is performed with an insertion sorting algorithm [102, 149, 171]. The algorithm is a modification of the parallel insertion sorting algorithm presented in [179]. The insertion sorter is easy to implement and is efficient for rather small data sets [102, 171]. The insertion sorter has a worst case complexity of $O(kn)$, where k is the number of input elements and n is the number of the smallest elements to be sorted. The microarchitecture of the insertion sorter for $n = 3$ sorted elements is illustrated in Figure 47. The insertion sorter includes n registers, where the sorted elements are stored, comparator(s), and control logic. The partial candidates to be sorted are fed to the sorter in a serial fashion. The PED of the new partial candidate is then compared to the stored partial candidates' PEDs in ascending order until the comparison condition is fulfilled or all the stored partial candidates have been compared (when $K < n$). When the correct sorted position is found, the partial candidates with larger PEDs are shifted to higher registers and the partial candidate in the last register in the sorter is discarded. The insertion sorter microarchitecture can also be implemented with parallelism and pipelining to enhance the processing speed. When parallel comparison units and logic are applied for different sorting elements, a pipelining structure can be implemented to the sorter, where each pipelining stage consists of one register and comparison logic. After the sorter, the sorted n candidates can be passed to the output.

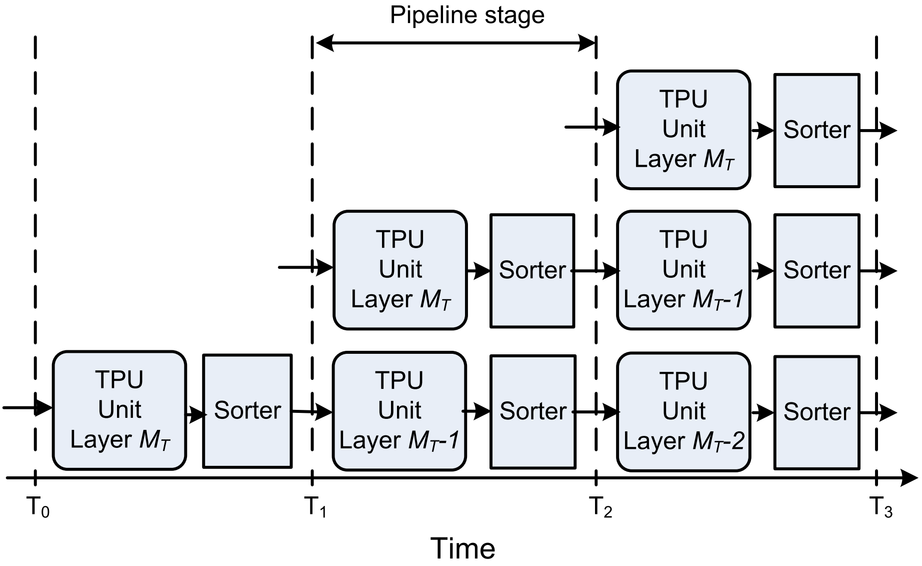


Fig 48. Timing diagram of the pipelined K-best-LSD algorithm architecture.

Timing and Scalability

The K-best-LSD algorithm architecture is designed to include pipelining, where each stage executes the tree pruning for the corresponding tree search layer. The throughput period of different stages, i.e., the TPU and sorter unit, should be made as close to each other as possible to achieve efficient design, because the throughput period of the pipeline structure is equal to the latency of the longest pipeline stage. Thus, the TPUs and sorter units that are at the end of the pipeline and include more operations should be implemented with more parallel processing. The timing diagram of the pipelined K-best-LSD architecture is illustrated in Figure 48. Each pipeline stage processes on different subcarrier at a certain time instant, and a fully loaded pipeline structure operates on M_{T} subcarriers. Total latency of a single subcarrier detection is $M_{\mathrm{T}}\tau_{\mathrm{stage}}$, where τ_{stage} is the pipeline throughput period.

The architecture is scalable as such for different constellations Ω, but the number of transmit antennas N_{T} determine the required number of stages $M_{\mathrm{T}} = 2N_{\mathrm{T}}$ in the architecture. Thus, the number of stages should be selected according to the largest supported N_{T}. The parallelism and pipelining choices should also be designed according to the largest supported Ω to maximize the throughput of the architecture.

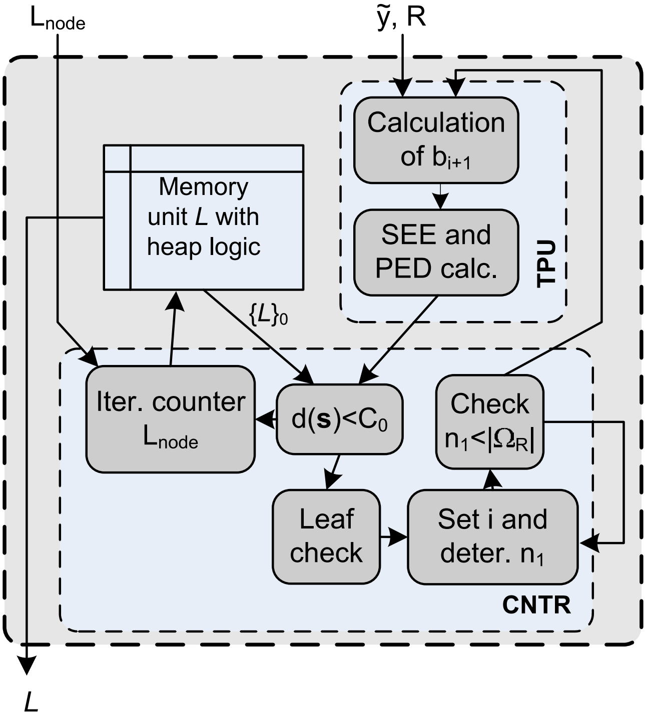


Fig 49. A scalable architecture for the SEE-LSD algorithm.

4.2.3 SEE-LSD algorithm

The SEE-LSD algorithm is a depth-first search based algorithm as listed in Algorithm 4. The SEE-LSD algorithm is not as such very suitable for implementation, because the algorithm prunes a variable number of nodes in the search tree depending on the system configuration and channel realization. A scalable architecture for the SEE-LSD algorithm, which consists of a tree pruning unit (TPU), a control unit (CNTR) and a memory unit, is shown in Figure 49. The architecture operates in sequential fashion and prunes a single node in the search tree in each iteration. The TPU executes the tree pruning, and the CNTR determines the partial candidate for the next iteration and the possible final candidate to be stored in the memory unit. The problem of variable complexity is solved by applying an input variable L_{node}, which sets a maximum limit for the number of nodes to be pruned by the architecture as discussed in Section 3.3. After a total of L_{node} iterations the architecture gives the candidate list $\mathscr{L}$ as an output.

TPU

The SEE-LSD algorithm TPU microarchitecture is illustrated in Figure 50. The TPU microarchitecture is divided into two sub-units that can be implemented with different levels of parallelism and pipelining. It should be noted that the SEE-LSD algorithm TPU microarchitecture has to be able to calculate the tree pruning for partial candidates in different search layers as opposed to the K-best-LSD algorithm TPU, which is designed for a particular search layer. Typically the TPU should be made as fast as possible with the proper parallelism and pipelining configuration as the latency of the unit directly affects the throughput of the SEE-LSD algorithm architecture.

The first unit calculates $b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}})$, which is the part of PED calculation that is independent of the new symbol x_i, as in (40). The unit can be implemented with different levels of parallelism and/or pipelining for faster calculation of the MUL operations. The number of required multiplications is $M_\mathrm{T} - i - 1$ and, thus, depends on the current layer i, where $i = 1, \ldots, M_\mathrm{T}$. We have determined via Monte Carlo simulations that a large portion of the depth-first search node extensions are executed in the upper part of the search tree. Thus, less than $M_\mathrm{T}/2$ parallel MULs should be used in general to have an efficient implementation.

The second unit executes the Schnorr-Euchner enumeration, i.e., determines the nth best node x_i, and calculates the PED of the extended partial candidate accordingly. The enumeration is designed in a modified fashion from the method presented in [91, (14)]. Instead of calculating the costly and high latency division operation, we calculate the absolute value in (40) with $|\Omega_\mathrm{r}|$ different symbols x_i. The calculation can be implemented with different levels of parallelism and/or pipelining, i.e., $1 - |\Omega_\mathrm{R}|$ separate parallel MAC units can be used. The desired nth best node is determined by defining first the node, i.e., the symbol, with the minimum PED. The information with the sign of value is used to determine the desired nth best node [91] and the PED is then updated.

Memory unit

The memory unit is used to store the N_cand final candidates with the smallest EDs, which are found during the SEE-LSD algorithm tree search. The memory unit is designed as a binary heap [102, 171] data structure, which keeps the stored elements in order according to the selected cost metric. The memory unit $\mathscr{L}$ is implemented as max-

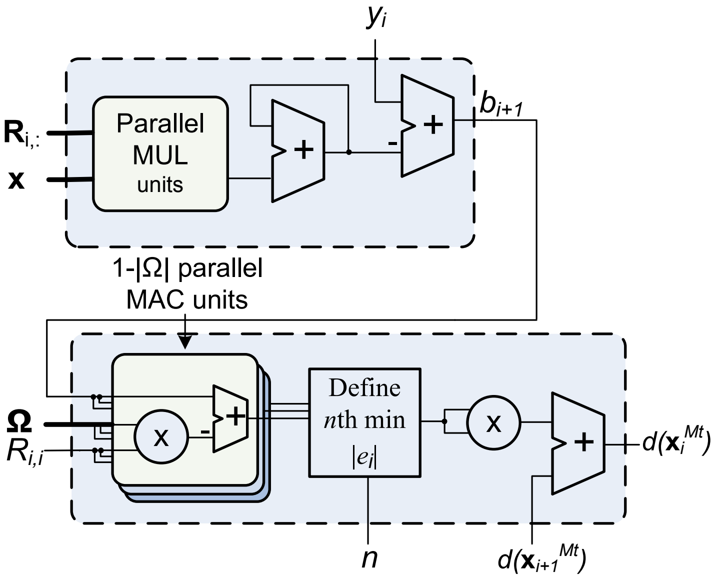


Fig 50. The microarchitecture for the extension of the candidate. The first unit calculates the b_{i+1} variable, and the second units execute the enumeration and updates the PED. ([147], published by permission of IEEE).

heap, where the new element $\mathscr{N}_{\mathrm{F}}(\mathbf{s}_{\mathrm{c}}, d(\mathbf{s}_{\mathrm{c}}))$ is always ordered in the heap as it is stored. The heap elements are kept in order so that the final candidate with the maximum ED is always at the top of the heap [102, 171].

The new final candidate is stored to the next available memory slot at the bottom of the heap or to the top of the heap if the memory is full and the ED of the new candidate is smaller than the ED of the current maximum candidate in the memory. The sorting is done via an up-heap operation from the bottom of the heap or with a down-heap operation from the top of the heap [102, 171]. Then the element is swapped to the correct position with the up-heap operation, which requires a read and a write operation to the memory, or with the down-heap operation, which requires two read operations and a write operation to the memory. The storing of a new element requires a time complexity of $O(\log_2(k))$ in the worst case [102, 171], where k is the size of the memory and $\lceil \log_2(k) \rceil$ is the height of the heap. The memory unit microarchitecture with up- and down-heap logic is illustrated in Figure 51. The memory unit can be implemented with dual port memory to enhance the memory access. However, the

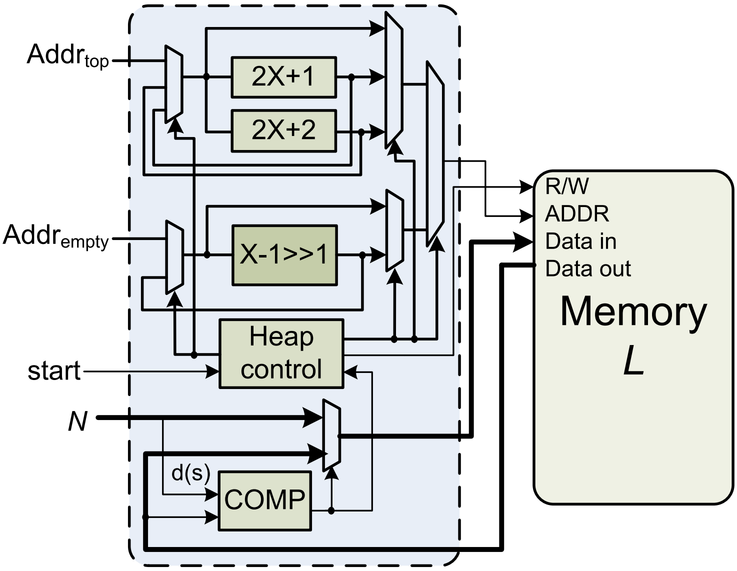


Fig 51. A microarchitecture of the memory unit with heap logic. ([139], published by permission of IEEE).

latency of the final memory unit is not a critical issue, because it is not accessed often in the SEE-LSD algorithm architecture and the size of the memory is quite small.

CNTR

The required control logic for the SEE-LSD algorithm architecture is rather simple. The logic determines the next search level i and next admissible node n_1 for the next algorithm iteration based on the partial candidate, which was extended in the TPU. If the extended candidate is a leaf node and $d(\mathbf{s}) < C_0$, the final candidate is stored to the memory unit and the sphere radius C_0 is possibly updated. The CNTR unit also terminates the search after L_{node} iterations.

Timing and Scalability

The SEE-LSD algorithm architecture operates in sequential fashion a total of L_{node} iterations, where the parameter L_{node} should be selected as suitable to provide the desired performance. The latency of the algorithm iterations consists of the latency of the CNTR and the latency of the TPU or the memory unit. The TPU and memory unit oper-

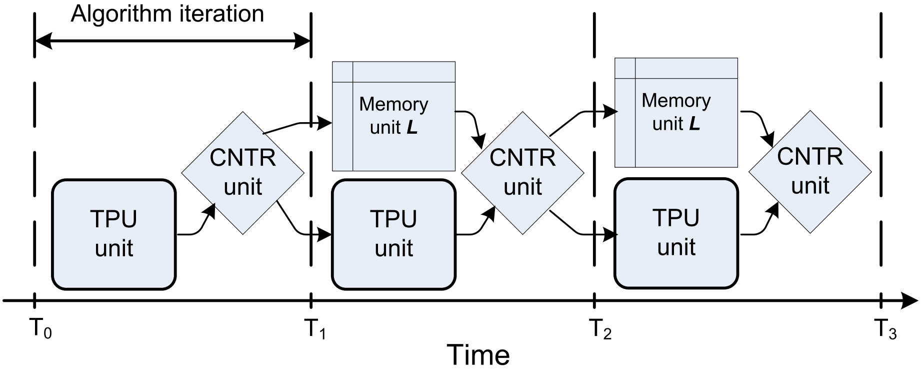


Fig 52. Timing diagram of the SEE-LSD algorithm architecture.

ations are designed to be executed in parallel, where the TPU is typically the slower unit as it includes more operations and the memory unit is executed only seldom. The timing diagram of the SEE-LSD algorithm architecture is illustrated in Figure 52. In order to maximize the throughput of the SEE-LSD algorithm architecture, the TPU should be implemented with proper parallelism and pipelining. The parameter L_{node} can also be lowered to increase the throughput with the cost of decreased performance.

The SEE-LSD algorithm architecture is as such scalable for system configurations with different number of transmit antennas N_{T} and different constellation Ω. The change in the configuration affects the required value in the parameter L_{node} with a given performance requirement. The required list size N_{cand} and the amount of operations in the TPU also increase with N_{T} and Ω, which results in increased latency if no additional resources are applied. Thus, the parallelism and pipelining choices should be designed according to the highest supported configuration.

4.2.4 IR-LSD algorithm

The IR-LSD algorithm is a metric-first search based algorithm as listed in Algorithm 5. The IR-LSD algorithm as such is the least suitable for implementation of the considered LSD algorithms, because the algorithm prunes a variable number of nodes in the search tree depending on the system configuration and channel realization. The algorithm also requires that the visited nodes be maintained in metric order to ensure optimality, which requires the usage of memory and sorting [98]. The IR-LSD algorithm architecture is shown in Figure 53, and includes a tree pruning unit (TPU) with two calculation

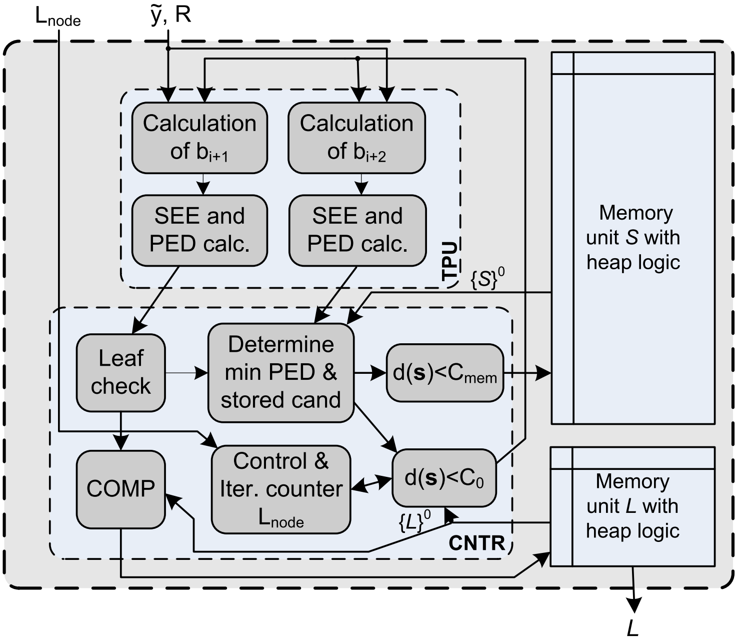


Fig 53. A scalable architecture for the IR-LSD algorithm. ([139], published by permission of IEEE).

modules, a partial candidate memory unit, a final candidate memory unit, and a control logic (CNTR) unit. In each iteration, the TPU executes the tree pruning for two partial candidates, and the CNTR determines the partial candidate for the next iteration and the possible final candidate to be stored in the memory unit. The problem of variable complexity is solved by applying an input variable L_{node}, which sets a maximum limit for the number of nodes to be pruned by the architecture as discussed in Section 3.3. After a total of L_{node} iterations the architecture gives the candidate list $\mathscr{L}$ as an output.

TPU

The IR-LSD algorithm TPU microarchitecture is illustrated in Figure 54. The IR-LSD algorithm architecture TPU is similar to the TPU in the SEE-LSD algorithm architecture with two similar candidate extension modules, which execute the tree pruning for the new selected candidate and the corresponding father candidate in parallel. The latency of the parallel units, i.e., the parallelism and pipelining choices, should be designed to be as similar as possible for efficient design. The extension of the candidate

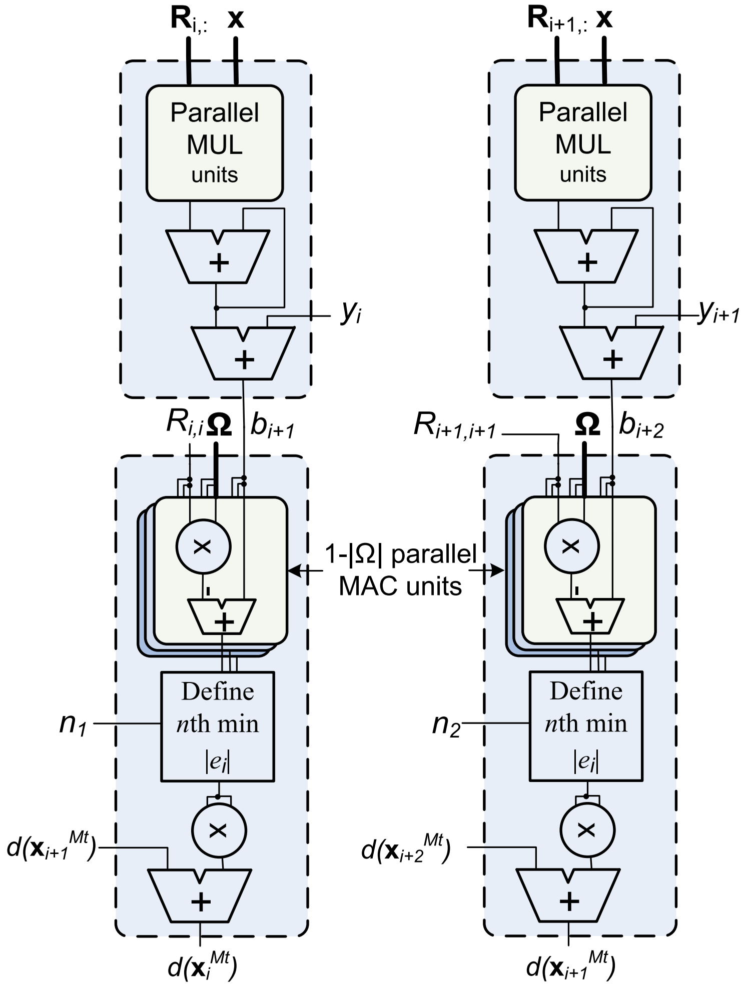


Fig 54. The microarchitecture for the TPU in IR-LSD algorithm architecture.

is also divided into two sub-unit microarchitectures that can be implemented with different levels of parallelism as illustrated in Figure 54 in detail. The first unit calculates $b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}})$ and $b_{i+2}(\mathbf{x}_{i+2}^{M_\mathrm{T}})$ for the new partial candidate and the father candidate, which is the part of PED calculation that is independent of the new symbols x_i and x_{i+1}, as in (40). The unit can be implemented with different levels of parallelism and/or pipelining for faster calculation of the MUL operations. The number of required multiplications in the calculation of $b_{i+1}(\mathbf{x}_{i+1}^{M_\mathrm{T}})$ is $M_\mathrm{T} - i - 1$ and, thus, depends on the current layer i, where $i = 1, \ldots, M_\mathrm{T}$. We have determined via Monte Carlo simulations that a large portion of the IR-LSD search node extensions are executed in the upper part of the search tree. Thus, less than $M_\mathrm{T}/2$ parallel MULs should be used in general to have an efficient implementation.

The second parallel units execute the Schnorr-Euchner enumeration for both candidates, i.e., determine the n_1th and n_2th best nodes for x_i and x_{i+1}, and calculate the PEDs of the extended candidates accordingly. The enumeration is designed in a modified fashion from the method presented in [91, (14)]. Instead of calculating the costly and high latency division operation, we calculate the absolute value in (40) with $|\Omega_\mathrm{r}|$ different symbols x_i. The calculation can be implemented with different levels of parallelism and/or pipelining, i.e., $1-|\Omega_\mathrm{r}|$ separate parallel MAC units can be used. The degree of parallelism should be decided depending on the slowest parallel unit in the whole IR-LSD algorithm architecture to optimize the performance. The desired nth best node is determined by defining first the node, i.e., the symbol, with the minimum PED. The information with the sign of value is used to determine the desired nth best node [91] and the PED is then updated.

Memory Units

There are two memory units in the IR-LSD architecture: the partial candidate memory set $\mathscr{S}$ and the final memory set $\mathscr{L}_\mathrm{F}$. The memory units are designed as binary heap [102, 171] data structures, which keep the stored elements in order according to the selected cost metric. The partial candidate memory set $\mathscr{S}$ is implemented as min-heap, where the elements $\mathscr{N}(\mathbf{s}, d(\mathbf{s}), n_2, i)$ are ordered so that the candidate with the minimum PED is always sorted to be at the top of the heap. The final memory set $\mathscr{L}_\mathrm{F}$, which is similar to the memory unit in the SEE-LSD architecture, is implemented as max-heap, where the stored final candidates $\mathscr{N}(\mathbf{s}, d(\mathbf{s}))$ are sorted according to the ED. The storing of a new element requires a time complexity of $O(\log_2(k))$ in the worst case [102, 171], where k is the size of the memory and $\lceil \log_2(k) \rceil$ is the height of the heap. The size of the partial candidate memory $\mathscr{S}$ is equal to L_node elements since, at maximum, the minimum candidate $\{\mathscr{S}\}^0$ is removed and two additional candidates $\mathscr{N}_c$ and $\mathscr{N}_f$ are added to the memory in each iteration. In practice, the partial memory size is always larger than the final memory size and the latency of the ordering of the elements might become a limiting factor in the IR-LSD algorithm implementation with a large L_node.

We modified the access to the memory unit to limit unnecessary storing and removing of partial candidates. The new partial candidate(s) $\mathscr{N}_c$ and possible $\mathscr{N}_f$ are first compared to the minimum candidate $\{\mathscr{S}\}^0$ in the CNTR unit, and if the candidate on the top of the heap has the minimum PED, the candidate is removed from the memory and the new partial candidate(s) are added to the heap. The addition of the first candidate is done by storing it to the top of the heap over the minimum candidate and applying the down-heap operation [102, 171] to sort the heap. Otherwise, the new candidate (child or father) is added to the next free memory address and the heap is sorted via an up-heap operation [102, 171]. We also apply the memory sphere radius C_{mem} to decrease the amount of memory access as the updated candidates are discarded if $d(\mathbf{s}) < C_{\text{mem}}$. The memory unit microarchitecture with up- and down-heap logic is illustrated in Figure 51.

CNTR

The control logic unit includes an iteration counter for the IR-LSD algorithm architecture and determines the candidates to be stored in the memory units and to be used in the search in the next algorithm iteration. The candidate to be used in the TPU unit in the next iteration is determined as the candidate with minimum PED from the extended candidates $\mathscr{N}_c$ and $\mathscr{N}_f$, and the minimum candidate in partial memory $\{\mathscr{S}\}^0$. If either one of the extended candidates $\mathscr{N}_c$ or $\mathscr{N}_f$ is selected for the next algorithm iteration, $\{\mathscr{S}\}^0$ remains in the memory. Thus, unnecessary memory access is minimized as the candidates $\mathscr{N}_c$ and $\mathscr{N}_f$ are not directly stored to the memory. The extended partial candidate(s) to be stored in $\mathscr{S}$ are also conditioned with C_{mem} to minimize memory access. If the extended candidate $\mathscr{N}_c$ is a leaf node and $d(\mathbf{s}) < C_0$, the final candidate is stored to the memory unit and the sphere radius C_0 is possibly updated. The CNRT also terminates the search after L_{node} iterations.

Timing and Scalability

The IR-LSD algorithm architecture and its timing are designed to minimize the latency in one algorithm iteration by introducing parallel operations. The straightforward data flow mapping of Algorithm 5 would first extend the new candidates, then store them in memory units, and finally determine the new candidate for the next iteration. However, the data flow can be designed more efficiently to reduce the latency of one algorithm iteration. The timing diagram for the architecture is illustrated in Figure 55. After the TPU extends the partial candidates in the current iteration, the control logic unit de-

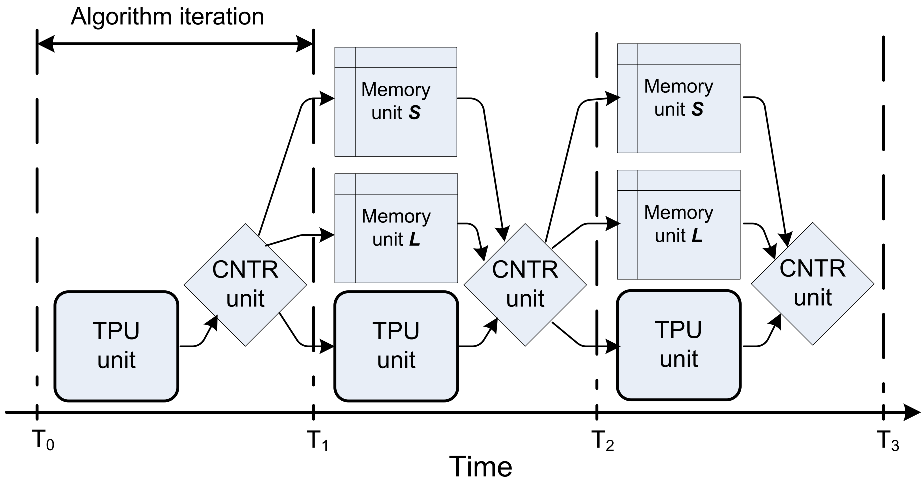


Fig 55. Timing diagram of the IR-LSD algorithm architecture, which illustrates the parallel operations in the architecture.

termines the new candidate for the TPU at the next iteration and the stored candidates for the memory units from the current iteration. The TPU and memory units are then executed in parallel, which decreases the latency significantly compared to the straightforward mapping of the algorithm. In order to maximize the throughput of the IR-LSD algorithm architecture, the TPU and partial memory unit should be implemented with proper parallelism and pipelining. Also the parameter L_{node} can be lowered to increase the throughput with the cost of decreased performance.

The limit for the number of algorithm iterations L_{node} should be defined separately for different system configurations or according to the most complex supported configuration. A proper L_{node} value depends on the channel realization and on the search tree size, i.e., on the number of independent data streams and the constellation size $|\Omega|$. A larger tree size requires a higher L_{node} value. Partial memory resources of L_{node} elements are reserved for the memory unit $\mathscr{S}$ according to the highest supported system configuration. The amount of parallelism and pipelining in the TPU unit can be modified based on latency requirements. However, the TPU unit latency should be optimized to match the memory unit $\mathscr{S}$ and its logic, which are executed in parallel, for efficient implementation.

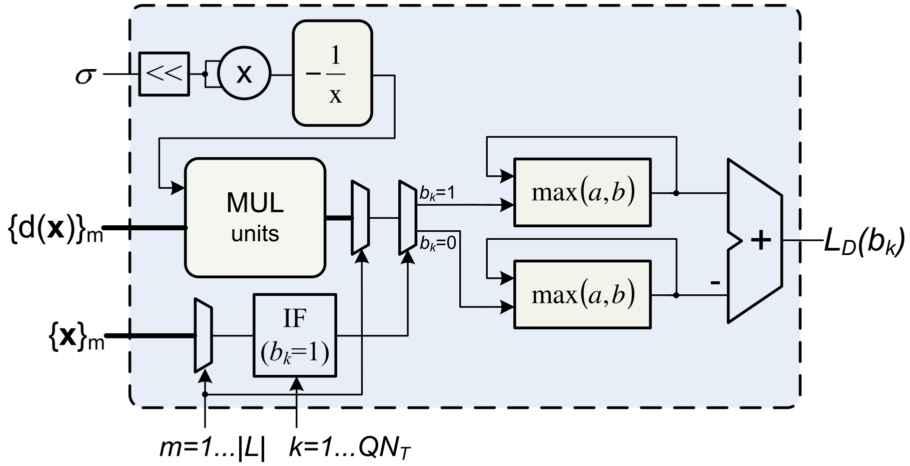


Fig 56. The designed microarchitecture for an LLR calculation unit that uses the max-log-MAP approximation. Different levels of parallelism can be applied to the MUL operations and to the logic for searching the maximum values. ([139], published by permission of IEEE).

4.2.5 LLR calculation

The soft output information $L_D(b_k)$ is calculated with the LSD algorithm output list $\mathscr{L}$ by using the LLR calculation unit. The LLR calculation unit microarchitecture, which calculates the max-log-MAP approximation as in (6), is illustrated in Figure 56. The architecture can be divided into two main parts: the scaling of the ED values and the search for maximum values for each bit. The two main parts can be applied with pipelining structure to enhance the throughput of the unit. The ED values in the candidate list $\mathscr{L}$ are scaled by multiplying them with the inverse of the noise variance $1/(2\sigma^2)$, i.e., a reciprocal division and a total of N_{cand} MUL operations are required. Different levels of parallelism and pipelining can be applied for the MUL operations in order to speed up the calculations as illustrated in Figure 56. The max-log-MAP approximation is calculated for each bit b_k, where $k = 1 \dots QN_{\text{T}}$. The calculation of $L_D(b_k)$ requires that all the N_{cand} ED values in the candidate list $\mathscr{L}$ be checked in order to determine the maximum values for both bit counterparts. Thus, two sequential logic loops are required in the calculation with the final list index m and bit value index k as illustrated in the architecture description in Figure 56. The latency of the loops can be decreased by applying parallel logic and/or pipelining to check multiple ED values or bits in parallel. It should be noted that the possibility for parallel implementation of the logic is a clear benefit of the max-log-MAP approximation compared to the log-MAP algorithm [51]. The log-MAP algorithm requires an addition of a look-up table value after each max operation, which makes the design of parallel logic challenging.

The soft output LLR calculation unit can be used as such for different system configurations. The total throughput of the unit depends on the LSD output list size N_{cand} and the number of bits in the each subcarrier QN_{T}.

4.3 Summary

In this chapter, we designed efficient architectures for the considered detector algorithms for MIMO–OFDM systems. We identified and introduced the key functional microarchitecture units and exploited the possibilities for parallelism and pipelining. It should also be noted that multiple architecture units can be used in parallel to support the higher data rate requirements in an OFDM system.

The introduced LMMSE detector architectures were based on systolic array structures with communicating PEs. We focused on the calculation of the LMMSE coefficient matrix via QRD as it is computationally the most complex part of the detector. We presented a fast pipelined triangular array and a less complex time shared linear array architecture for both the SGR and the CORDIC based 2×2 and 4×4 LMMSE detectors, respectively. The systolic array cell structures and the dataflow were designed and introduced in detail. The presented triangular or linear array structures can be easily scaled for the systems with different antenna configurations with modified array structure.

We designed architectures for different list sphere detector subunits. Architectures were introduced for the SQRD preprocessing algorithm, for the considered LSD algorithms and for the max-log-MAP LLR calculation unit. The SQRD algorithm architecture operates in a sequential fashion, and calculates one matrix decomposition at a time. A parallel and pipelined architecture was designed for the K-best-LSD algorithm, which makes the algorithm very suitable for hardware implementation. The K-best-LSD algorithm architecture is scalable as such for different constellations Ω, but the number of transmit antennas N_{T} determine the required number of stages $M_{\mathrm{T}} = 2N_{\mathrm{T}}$ in the architecture. Scalable architectures were designed for the SEE-LSD algorithm and the IR-LSD algorithm, which operate in a sequential fashion. The architectures include implementation tradeoffs introduced in Section 3, which make them more suitable for hardware implementation. The SEE-LSD algorithm and the IR-LSD algorithm architectures are as such scalable for system configurations with different numbers of transmit antennas N_{T} and different constellation Ω. A parallel and pipelined architecture was also introduced for the max-log-MAP LLR calculation unit.

5 Hardware implementation

In this chapter, we focus on the hardware implementations of the considered detector algorithms and their designed architectures. The LMMSE detector implementation results are first presented in Section 5.1. The LSD implementation results are then presented in Section 5.2. Measurement results with algorithms on FPGA based hardware testbeds are presented in Section 5.3. Finally, the results are summarized in Section 5.4.

5.1 LMMSE detector

The CORDIC based LMMSE detector architectures for 2×2 and 4×4 MIMO systems and the SGR based LMMSE detector architecture for a 2×2 MIMO system were implemented to a Xilinx Virtex-II XC2V6000 FPGA chip. The CORDIC based LMMSE detector was done in handwritten very high speed integrated circuits (VHSIC) hardware description language (VHDL) and functionally verified in ModelSim. The SGR based LMMSE detector architecture was developed and simulated in System Generator for DSP software tool [180] from Xilinx. The Mentor Graphics Leonardo synthesis tool was used for the synthesis for the Xilinx Virtex-II XC2V6000 FPGA chip. All the FPGA implementations were designed for 66MHz frequency and optimized for low area and low latency. The implementation results in this section are presented for the LMMSE coefficient matrix calculation block, which dominates the total complexity of the LMMSE detector, and for the whole detector structure. The only difference in the two considered designs is the LMMSE coefficient calculation block, i.e., other parts of the detector are the same as illustrated in Figure 32. The division operation in the boundary cell is implemented using a reciprocal divider from the Xilinx intellectual property core library [181] and two real multipliers.

5.1.1 Synthesis results

The device utilizations of the CORDIC based LMMSE detector implementations for 2×2 and 4×4 MIMO systems have been listed in Table 16. The CORDIC based implementation uses 16 bit fixed-point internal word lengths in the coefficient matrix $\mathbf{W}$ calculation, which includes matrix-matrix multiplications, a CORDIC based QRD,

Table 16. Synthesis results of the CORDIC based LMMSE detector for the Xilinx Virtex-II XC2V6000 FPGA chip. ([140], published by permission of IEEE).

Resource	Coeff. calc. 2×2	Whole detector 2×2	Whole detector 4×4
CLB Slices	11910	15834	20722
Block RAMs	6	20	58
Block Multipliers	20	32	58

and a back substitution operation. The CORDIC algorithm that is used to calculate the QRD is implemented with $i_{\max} = 10$ iterations. It should be noted that the number of iterations and fixed-point word lengths may be decreased depending on the required accuracy. A design with $i_{\max} = 7$ iterations and 12 bit internal word length would require approximately 30% less slices in the synthesis. The device utilization of the SGR based LMMSE detector implementation for a 2×2 MIMO system has been listed in Table 17. The architecture for a 4×4 MIMO system was not implemented in a FPGA, because the EB4G hardware testbed used was configured only for a 2×2 MIMO. However, the complexity estimate is approximately in the same ratio as with the CORDIC based implementation. The SGR based implementation uses mainly 18 bit fixed-point internal word lengths in the coefficient matrix calculation, which includes matrix-matrix multiplications, a SGR based QRD and a triangular matrix inversion operation. The most crucial part in terms of word length requirements is the SGR based QRD. The variables are scaled adaptively before the matrix inversion due to the high dynamic range requirements of the SGR algorithm. The matrix $\mathbf{A}$ to be inverted in the A2 part is scaled to a desired level according to the highest value of each matrix. The scaling is then compensated for after the matrix inversion as illustrated in Figure 38. In the QRD array, the reciprocal divider is the most accuracy demanding operation and requires the largest dynamic range. The implemented divider uses 20 bit fixed-point word length and also three step adaptive scaling to reduce the required dynamic range of the signal. It can be noted from the area reports in Tables 16 and 17 that the CORDIC based design requires more slices and less block multipliers compared to the SGR based design. This is due to normal arithmetic applied in the SGR algorithm and the rotation based arithmetic applied in the CORDIC algorithm. Also the required word lengths with fixed-point arithmetic are a little higher with the SGR based design.

Table 17. Synthesis results of the SGR based LMMSE detector for a 2×2 MIMO system for the Xilinx Virtex-II XC2V6000 FPGA chip. ([140], published by permission of IEEE).

Resource	Coeff. calc.	Whole detector
CLB Slices	7422	11346
Block RAMs	14	28
Block Multipliers	77	89

5.1.2 Latency

The detector implementations are designed to support pipelining in order to efficiently calculate the LMMSE coefficients for multiple subcarriers in an OFDM system. The pipeline is able to give the next output after four clock cycles for a SGR based detector. The CORDIC based pipeline is able to give the output matrix every clock cycle. The total latency for the design with a given number of subcarriers consists of the latency of the fixed data path for the first calculated subcarrier, the latency from the input buffer, and the pipeline latency for the additional pipelined subcarriers. Most of the latency in the detector implementations is due to the QRD array.

Multiple wireless telecommunication standards, such as IEEE 802.11n [16], 3GPP LTE [182], and IEEE 802.16e [17], apply the MIMO–OFDM technique. In a MIMO–OFDM receiver, the LMMSE detector coefficients should be recalculated for each subcarrier within the channel coherence time interval, i.e., while the channel can be assumed to be static, in order to achieve good performance. The channel coherence time T_c for 50% correlation can be calculated as $T_\mathrm{c} \approx \frac{9}{16\pi f_\mathrm{D}}$ [183] where f_D is the Doppler frequency. Then, e.g., with a carrier frequency $f_\mathrm{c} = 2.4$ GHz and a velocity of 120 kmph the channel coherence time is $T_\mathrm{c} \approx 670\ \mu$s. The latencies of calculating the LMMSE detector coefficients for different telecommunication standards with different numbers of subcarriers with different detector implementations have been listed in Table 18. The LMMSE detectors for a 2×2 MIMO system calculate the detector coefficients for the $N_\mathrm{used} = 52$ subcarriers in 802.11n in 11μs and 9μs with CORDIC and SGR based designs. The latency of the CORDIC based LMMSE detector for a 4×4 MIMO system is 50μs. The latency for calculating the 2×2 MIMO system LMMSE detector coefficients in the 3GPP LTE system with 20 MHz BW and with $N_\mathrm{used} = 1200$ subcarriers is 32μs and 148μs with CORDIC and SGR based designs, respectively, and 680μs with the CORDIC based LMMSE for a 4×4 MIMO system. The latency for calculating the

Table 18. Latencies for the LMMSE detector coefficients calculation with parameters from different wireless telecommunication standards.

Standard	**802.11**	**3GPP LTE**	**802.16e**
Bandwidth	20 MHz	20 MHz	20MHz
Number of used subcarriers N_{used}	52	1200	1703
Latency, 2×2 CORDIC	$11\ \mu\text{s}$	$32\ \mu\text{s}$	$40\ \mu\text{s}$
Latency, 2×2 SGR	$9\ \mu\text{s}$	$148\ \mu\text{s}$	$209\ \mu\text{s}$
Latency, 4×4 CORDIC	$50\ \mu\text{s}$	$680\ \mu\text{s}$	$970\ \mu\text{s}$

2×2 antenna LMMSE detector coefficients for the maximum $N_{\text{used}} = 1703$ subcarriers in 802.16e is 40μs and 209μs with CORDIC and SGR based design, respectively, and 970 μs with the CORDIC based LMMSE for a 4×4 MIMO system.

5.1.3 Conclusions

The implementation results of the CORDIC and SGR based LMMSE detectors were presented in this section. The area reports in Tables 16 and 17 showed that the CORDIC based design requires more slices and less block multipliers compared to the SGR based design. This is due to arithmetic operations applied in the SGR algorithm and the rotation based arithmetic applied in the CORDIC algorithm. Also the required word lengths with fixed-point arithmetic are a little higher with the SGR based design. Most of the latency in the detector implementations is due to the QRD array. In order to achieve efficient pipelining and dataflow, the latency of different cells should be close to each other. In the SGR implementation, the square cell in Figure 40 is the bottleneck for the dataflow. The latencies of the CORDIC cells are quite close to each other and, thus, the dataflow is rather balanced. However, it should be noted that the latencies of CORDIC cells depend on the number of applied CORDIC iterations and required accuracy.

It can be noted that the LMMSE detector implementations for 2×2 MIMO systems are fast enough to calculate the coefficient matrices for all MIMO–OFDM subcarriers in the example channel coherence time of 670 μs and the FPGA implementation is feasible as such for application in a receiver in different telecommunications standards. The CORDIC based implementation for a 4×4 MIMO system is also feasible in most cases. In the case when the time interval is too tight for one detector, multiple detectors can be used in parallel.

5.2 List sphere detector

The different LSD subunit architectures, which were introduced in Section 4.2, were implemented for both a Xilinx Virtex-5 FPGA chip and 0.18μm complementary metal-oxide semiconductor (CMOS) ASIC technology for a 4×4 MIMO system and the results are presented in this Section. First, the results on the word length study are presented in Section 5.2.1. The SQRD preprocessing algorithm implementation results are then introduced in Section 5.2.2. The implementation results of all three considered LSD algorithms are introduced in Section 5.2.3. The implementation results of the max-log-MAP based LLR calculation unit are introduced in Section 5.2.4.

The implementation of algorithms was done by writing the algorithm architecture description with fixed-point ANSI C++ language and then applying the Mentor Graphics Catapult C Synthesis tool [184] to produce a bit-accurate, parallel register transfer level (RTL) description. The Catapult C Synthesis tool was used to study and compare different microarchitecture solutions with different levels of parallelism and pipelining, and the most efficient design was then selected for implementation. The RTL description was synthesized with Mentor Graphics Precision RTL or Synopsys Design Compiler for the FPGA chip and the ASIC technology, respectively. The SQRD FPGA RTL was designed with 100MHz frequency and the other designs with 150MHz frequency. The ASIC RTL was designed with 250MHz frequency and the ASIC power estimation was done with the Synopsys PrimeTime tool. The Virtex-5 FPGA resource allocation results include control logic block (CLB) slices, block random access memories (BRAMs), and DSP48 units, which includes a 18×18 bit embedded multiplier. The ASIC complexity is given in area and in gate equivalents (GEs), where one GE corresponds to the area of a two-input drive-one NAND gate.

5.2.1 Word length study

A fixed-point word length study was done via computer simulations to determine adequate fixed-point word lengths to be used in the LSD implementation. The simulations were executed for a 4×4 MIMO–OFDM system with 16- and 64-QAM. The fixed-point word lengths presented in this section are listed as (W,I,S), where the W, I, and S refer to the total number of bits, the number of bits used for the integer part, and the unsigned or signed type of value representation, respectively [185, 186]. The numerical range and quantization step of unsigned and signed fixed-point representations are

Table 19. Fixed-point representations.

Type	Numerical range	Quantization step
unsigned (W,I,u)	0 to $(1-2^{-\mathrm{W}})2^{\mathrm{I}}$	$2^{-(\mathrm{W}-\mathrm{I})}$
signed (W,I,s)	$(-0.5)2^{\mathrm{I}}$ to $(0.5-2^{-\mathrm{W}})2^{\mathrm{I}}$	$2^{-(\mathrm{W}-\mathrm{I})}$

Table 20. The determined word lengths for the SQRD algorithm in a 4×4 MIMO system with 16-QAM.

SQRD	**H**	**R**	**Q**	**norm**	*sqrt*()	*div*()
(W,I,S)	$(11,3,s)$	$(26,4,s)$	$(26,3,s)$	$(27,5,u)$	$(23,3,u)$	$(21,3,u)$

listed in Table 19. The SQRD preprocessing algorithm word lengths were defined only for 16-QAM and they are listed for different variables in Table 20. The SQRD algorithm requires up to 27 bits in total for the representation of internal variables in the calculations. We noted that the total number of required bits in the representation of internal variables increases with decomposed matrix columns, i.e., with M_{T}, because the decomposition of the channel matrix $\mathbf{H}$ is executed in a column by column fashion and the previously solved elements are used iteratively in the decomposition calculations of the following columns. The LSD algorithm and the LLR calculation word lengths for both 16- and 64-QAM are then listed for different variables in Table 21. The LSD algorithm variables require up to 12 and 15 bits in the word length representation with 16- and 64-QAM, respectively. The output candidate list ED values should be represented with 10 and 14 bit word lengths in total with 16- and 64-QAM, respectively. The LLR calculation variables, which do not depend on the constellation size, require up to 10 bits for the representation and the soft-output $L_D(b_k)$ requires 6 bits in total word length representation. It should also be noted that the required word lengths can be decreased by applying proper scaling of the variables.

5.2.2 SQRD algorithm

The SQRD algorithm implementation is based on the architecture presented in Figure 43 in Section 4.2.1. The real valued signal model is applied and the implemented algorithm decomposes a real valued channel matrix with $M_{\mathrm{T}}=M_{\mathrm{R}}=8$, which is equivalent to a 4×4 MIMO system. The implementation is done for up to 16-QAM with the fixed-point word lengths presented in Table 20. The SQRD algorithm architecture is implemented with parallelism and pipelining. The calculation of column norms is im-

Table 21. The determined word lengths for the LSD algorithms and the LLR calculation for a 4×4 MIMO system with 16- and 64-QAM.

Signal	16-QAM	64-QAM
$\tilde{\mathbf{y}}$	$(10,4,s)$	$(14,5,s)$
$\mathbf{R}$	$(9,3,s)$	$(13,4,s)$
Ω	$(8,1,s)$	$(12,2,s)$
$\mathbf{b}_i(\mathbf{s})$	$(12,5,s)$	$(15,5,s)$
$d(\mathbf{s})$	$(10,5,u)$	$(14,5,u)$
σ	$(8,0,u)$	$(8,0,u)$
LLR internal	$(10,5,s)$	$(10,5,s)$
$L_D(b_k)$	$(6,4,s)$	$(6,4,s)$

Table 22. Synthesis results of the SQRD preprocessing algorithm with the real valued signal model, $M_\mathrm{T} = M_\mathrm{R} = 8$, and 16-QAM for a Xilinx Virtex-5 FPGA chip.

FPGA	Slices	BRAMs	DSP48s	Latency
SQRD	1159	2	24	1303 cc / 13.03 μs

plemented with two parallel MULs, which are pipelined with a throughput interval of one clock cycle. The update of $\mathbf{Q_i}$ and the iterative update loop are also implemented with two parallel MULs, which are similarly pipelined.

The synthesis results of the SQRD algorithm implementation with the real valued signal model and with $M_\mathrm{T} = M_\mathrm{R} = 8$ are listed in Table 22 for the Virtex-5 FPGA chip. The FPGA implementation requires 1159 slices, 2 BRAMs and up to 24 DSP48s, which include an embedded multiplier unit, due to the rather high word lengths. The latency of a single channel matrix decomposition is 1303 clock cycles (cc) or 13.03 μs, i.e., 76.7k operations per second. The corresponding synthesis results for the $0.18\mu m$ CMOS technology are listed in Table 23. The ASIC implementation requires 96.8 kGEs and consumes 154 mW power. The latency for a singe channel matrix decomposition is 4.34 μs, i.e., 230k operations per second, which is significantly faster than the FPGA implementation. The higher latency of the FPGA implementation is mainly due to the wire routing delay in the chip. We also calculated the required parallel SQRD resources with $0.18\mu m$ CMOS technology resources for a downlink receiver in a 3GPP LTE standard transmission with 20 MHz bandwidth and with $N_\mathrm{used} = 1200$. We assume a 4×4 MIMO–OFDM system with 16-QAM and the channel coherence time is assumed to be $T_\mathrm{c} \approx 670$ μs as in Section 5.1.2. Thus, a 8×8 real-valued SQRD is required, which is

Table 23. Synthesis results of the SQRD preprocessing algorithm with the real valued signal model, $M_{\mathrm{T}} = M_{\mathrm{R}} = 8$, and 16-QAM for 0.18$\mu m$ CMOS technology.

ASIC	**Area** (mm^2)	**kGEs**	**Power** (mW)	**Latency**
SQRD	1.18	96.8	154	1084 cc / 4.34 μs

Table 24. The required SQRD preprocessing ASIC resources in the LSD detection of 3GPP LTE standard with 20MHz BW and with $N_{\text{used}} = 1200$.

4×4 **MIMO–OFDM, 16-QAM**	**Area** (mm^2)	**kGEs**	**Power** (mW)
8×8 real-valued SQRD	9.0	745	1190

able to recalculate the SQRD for all $N_{\text{used}} = 1200$ subcarriers within the channel coherence time. The required $0.18\mu m$ CMOS technology resources are scaled linearly from the SQRD implementation results and are listed in Table 24. The SQRD preprocessing implementation, which is able to recalculated the SQRD for all $N_{\text{used}} = 1200$ subcarriers within the channel coherence time T_{c}, requires a complexity of 745 kGEs and the power usage is $1.2W$.

5.2.3 Tree search algorithms

The synthesis results on the multiplication block and the tree search algorithms, the K-best-LSD algorithm, the SEE-LSD algorithm, and the IR-LSD algorithm, are presented in this Section. Then the detection rates and the efficiency of the LSD algorithm implementations are presented and analyzed.

Multiplication block

The LSD algorithms take $\tilde{\mathbf{y}}$ and $\mathbf{R}$ as an input. The calculation of $\tilde{\mathbf{y}} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$ requires a matrix-to-vector multiplication, which has to be done prior to the tree search. The multiplication is done in multiplication block with a single pipelined MUL, which is designed according to the LSD algorithms implementation latency. The synthesis results of the implementation for a Virtex-5 FPGA chip and for $0.18\mu m$ CMOS technology are listed in Tables 25 and 26. The constellation size affects the required word lengths and, thus, the multiplier block is more complex for a system with higher constellation size. In general, the complexity of the multiplication block is low compared to the total complexity of the LSD.

Table 25. Synthesis results of the multiplication block for an SM system with $N_{\mathrm{T}} = N_{\mathrm{R}} = 4$ for Xilinx Virtex-5 FPGA chip.

Mult. block	**Slices**	**BRAMs**	**DSP48s**	**Latency**
16-QAM	59	1	2	64 cc / 427 ns
64-QAM	73	1	2	64 cc / 427 ns

Table 26. Synthesis results of the multiplication block for an SM system with $N_{\mathrm{T}} = N_{\mathrm{R}} = 4$ for 0.18μm CMOS technology.

Mult. block	**Area** (mm^2)	**kGEs**	**Latency**	**Power** (mW)
16-QAM	0.05	4.5	64 cc / 256 ns	10.2
64-QAM	0.08	6.7	64 cc / 256 ns	14.2

Table 27. Synthesis results of the K-best-LSD algorithm for an SM system with $N_{\mathrm{T}} = 4$ for Xilinx Virtex-5 FPGA chip.

K-best	**Slices**	**BRAMs**	**DSP48s**	**Latency**	**Throughput**
16-QAM	3459	0	36	458 cc / 4580 ns	64 cc / 640 ns
64-QAM	4364	0	36	458 cc / 4580 ns	64 cc / 640 ns

K-best-LSD algorithm

The K-best-LSD algorithm implementation is based on the pipelined architecture presented in Figure 45 in Section 4.2.2. The K-best-LSD algorithm is implemented with $K = 16$ for both 16- and 64-QAM, which means that the difference between the implementations in complexity is due to the higher required word lengths with the larger constellation size. Different pipeline stages in the implementation are optimized to have a similar latency, which means that more parallelism is applied to the latter stages. The synthesis results of the K-best-LSD algorithm implementation for the Virtex-5 FPGA chip and for the 0.18μm CMOS technology are listed in Tables 27 and 28, respectively. The FPGA implementation requires 3459 and 4364 slices, and 36 DSP48s for the 16- and 64-QAM designs, respectively. The throughput period is 640 ns for both designs per detected subcarrier. The ASIC implementation then requires 127 and 194 kGEs with a power consumption of 345 and 458 mW for 16- and 64-QAM designs, respectively. The throughput period is 256 ns for both designs per detected subcarrier.

Table 28. Synthesis results of the K-best-LSD algorithms for an SM system with $N_\mathrm{T} = 4$ for 0.18μm CMOS technology.

K-best	**Area** (mm^2)	**kGEs**	**Latency**	**Throughput**	**Power** (mW)
16-QAM	1.51	127.3	458cc/1.83μs	64cc/256ns	345
64-QAM	2.37	193.9	458cc/1.83μs	64cc/256ns	458

Table 29. Synthesis results of the SEE-LSD algorithm for an SM system with $N_\mathrm{T} = 4$ for Xilinx Virtex-5 FPGA chip.

SEE-LSD	**Slices**	**BRAMs**	**DSP48s**	**Latency**
16-QAM	182	0	5	16 cc / 107 ns per it.
64-QAM	302	0	10	18 cc / 120 ns per it.

Table 30. Synthesis results of the SEE-LSD algorithms for an SM system with $N_\mathrm{T} = 4$ for 0.18μm CMOS technology. ([139], published by permission of IEEE).

SEE-LSD	**Area** (mm^2)	**kGEs**	**Latency**	**Power** (mW)
16-QAM	0.13	10.6	13 cc / 52 ns per it.	25
64-QAM	0.27	22.0	16 cc / 64 ns per it.	38

SEE-LSD algorithm

The SEE-LSD algorithm implementation is based on the architecture presented in Figure 49 in Section 4.2.3. The SEE-LSD architecture TPU for 16-QAM was implemented with four parallel pipelined MULs in the first subunit and four parallel MULs in the latter subunit. The TPU for 64-QAM was implemented with four parallel pipelined MULs in the first subunit and eight parallel MULs in the latter subunit. Both algorithm implementations are done for output list size $N_\mathrm{cand} = 15$. The synthesis results of the SEE-LSD algorithm implementation for the Virtex-5 FPGA chip and for the $0.18\mu m$ CMOS technology are listed in Tables 29 and 30. The FPGA implementation requires 182 and 302 slices, and 5 and 10 DSP48s for the 16- and 64-QAM designs, respectively. The latency is 107 ns and 120 ns for the 16- and 64-QAM designs per algorithm iteration. The ASIC implementation then requires 10.6 and 22 kGEs with a power consumption of 25 and 38 mW for the 16- and 64-QAM designs, respectively. The latency is 52 ns and 64 ns for the 16- and 64-QAM designs per algorithm iteration.

Table 31. Synthesis results of the IR-LSD algorithm for an SM system with $N_T = 4$ for Xilinx Virtex-5 FPGA chip.

IR-LSD	Slices	BRAMs	DSP48s	Latency
16-QAM	798	1	10	18 cc / 120 ns per it.
64-QAM	859	2	20	20 cc / 133 ns per it.

Table 32. Synthesis results of the IR-LSD algorithms for an SM system with $N_T = 4$ for 0.18μm CMOS technology. ([139], published by permission of IEEE).

IR-LSD	**Area** (mm^2)	kGEs	Latency	**Power** (mW)
16-QAM	0.31	25.4	14 cc / 56 ns per it.	57
64-QAM	0.59	48.2	17 cc / 68 ns per it.	90

IR-LSD algorithm

The IR-LSD algorithm implementation is based on the architecture presented in Figure 53 in Section 4.2.4. The IR-LSD architecture TPU for 16-QAM was implemented, as in the SEE-LSD algorithm, with four parallel pipelined MULs in the first subunits and four parallel MULs in the latter subunits. The TPU for 64-QAM was implemented with four parallel pipelined MULs in the first subunit and eight parallel MULs in the latter subunit. Both algorithm implementations are done for output list size $N_{\text{cand}} = 15$. The memory unit $\mathscr{S}$ was implemented with dual port RAM to enhance the memory access. The maximum number of algorithm iterations is limited to 175 and 225 in the 16- and 64-QAM implementation, respectively. Thus, the memory unit size was 175x31 and 225x35 bits for the 16- and 64-QAM, respectively. The synthesis results of the IR-LSD algorithm implementation for the Virtex-5 FPGA chip and for the 0.18μm CMOS technology are listed in Tables 31 and 32. The FPGA implementation requires 798 and 859 slices, 1 and 2 BRAMs, and 10 and 20 DSP48s for the 16- and 64-QAM designs, respectively. The latency is 120 ns and 133 ns for the 16- and 64-QAM designs per algorithm iteration. The ASIC implementation then requires 25.4 and 48.2 kGEs with a power consumption of 57 and 90 mW for the 16- and 64-QAM designs, respectively. The latency is 56 ns and 68 ns for the 16- and 64-QAM designs per algorithm iteration.

Detection Rates

The detection rate R_{det} denotes the amount of transmitted coded bits that the LSD algorithm implementation is able to detect in a certain time with a given complexity. The total detection rate R_{det} of the LSD algorithm implementation can be calculated as

$$R_{\text{det}} = \frac{N_{\text{T}}Q}{\Delta_{\text{tot}}} \text{bits/s}, \tag{52}$$

where Δ_{tot} corresponds to the throughput time of the LSD algorithm implementation. The throughput time for the sequential search algorithm implementations, the SEE-LSD algorithm and the IR-LSD algorithm, is defined as $\Delta_{\text{tot}} = \Delta_{\text{it}} L^{\text{it}}_{\text{avg}}$, where Δ_{it} is the latency per algorithm iteration and $L^{\text{it}}_{\text{avg}}$ is the average number of executed algorithm iterations. Thus, the achievable detection rate R_{det} depends on the defined maximum limit for visited nodes L_{node}, which should be properly selected to meet the desired FER target with a given channel realization and SNR γ. It should be noted that the IR-LSD algorithm implementation checks two nodes in one algorithm iteration and that the K-best-LSD algorithm implementation detection rate is fixed as the algorithm search goes through a fixed number of nodes with variable performance depending on the channel realization and SNR. Also it should be noted that the implementation of multiple parallel LSD algorithms can be used to achieve a higher detection rate.

The detection rates of the LSD algorithm ASIC implementations for 16- and 64-QAM in different channel environments are listed in Table 33. The listed SNR range is selected as the operating range of the LSD based receiver with a given configuration and channel environment. The detection rates of the SEE-LSD algorithm and IR-LSD algorithm implementations are lower at low SNR as more algorithm iterations are required to achieve adequate performance. The detection rates at high SNR correspond to cases where the minimum number of iterations provides adequate performance. The LSD algorithm implementations have different performances and complexities, and, thus, we also compare the efficiency of the implementations. The comparison is done with an algorithm work factor W_{alg}, which is calculated as a multiplication product between the used resources in terms of GEs and the implementation throughput time per subcarrier Δ_{tot}, and a smaller value reflects a more efficient implementation [187, 188]. The algorithm work factor values of the LSD algorithm ASIC implementations for 16- and 64-QAM in different channel environments are listed in Table 34. Also the performances of the implementations relative to the max-log-MAP detector are listed in Table 34. All of the LSD algorithm implementations have advantages in certain channel envi-

Table 33. Detection rates of the LSD algorithm ASIC implementations in different channel environments.

$R_{\text{det}}^{(\text{asic})}$	IR-LSD alg.	SEE-LSD alg.	K-best-LSD alg.
16-QAM, UNC $\gamma = [13 - 19]$ dB	$[4.14, 31.7]$Mbps	$[1.07, 34.2]$Mbps	62.5Mbps
16-QAM, CORR $\gamma = [21 - 26]$ dB	$[1.70, 31.7]$Mbps	$[0.35, 34.2]$Mbps	62.5Mbps
64-QAM, UNC $\gamma = [20 - 25]$ dB	$[3.71, 39.2]$Mbps	$[1.12, 41.6]$Mbps	93.8Mbps
64-QAM, CORR $\gamma = [30 - 35]$ dB	$[1.62, 39.2]$Mbps	$[0.30, 41.6]$Mbps	93.8Mbps

Table 34. Performance and work factor numbers of the LSD algorithm ASIC implementations in different channel environments.

		IR-LSD alg.	SEE-LSD alg.	K-best-LSD alg.
16-QAM, UNC	W_{alg}	$[0.097, 0.013]$	$[0.158, 0.005]$	0.030
$\gamma = [13 - 19]$ dB	Perf.	Max-log - 0.6dB	Max-log - 0.8dB	Max-log - 0.4dB
16-QAM, CORR	W_{alg}	$[0.239, 0.013]$	$[0.472, 0.005]$	0.030
$\gamma = [21 - 26]$ dB	Perf.	Max-log - 0.5dB	Max-log - 0.5dB	Max-log - 1.2dB
64-QAM, UNC	W_{alg}	$[0.311, 0.029]$	$[0.473, 0.013]$	0.050
$\gamma = [20 - 25]$ dB	Perf.	Max-log - 1.2dB	Max-log - 1.2dB	Max-log - 0.9dB
64-QAM, CORR	W_{alg}	$[0.715, 0.029]$	$[1.757, 0.013]$	0.050
$\gamma = [30 - 35]$ dB	Perf.	Max-log - 0.7dB	Max-log - 0.7dB	Max-log - 2.0dB

Table 35. The required LSD algorithm ASIC resources in the LSD detection of 3GPP LTE standard with 20MHz BW and with $N_{\text{used}} = 1200$.

4×4 **MIMO–OFDM, 16-QAM**	**Area** (mm^2)	**kGEs**	**Power** (mW)
IR-LSD alg.	2.6-49.0	216-4010	485-9000
SEE-LSD alg.	1.0-100	83-8140	198-19200
K-best-LSD alg.	6.5	547	1480

ronments and SNR values. The K-best-LSD algorithm implementation achieves rather good performance in the uncorrelated channel with a fixed W_{alg}, but the performance suffers in highly correlated channels. The algorithm work factor W_{alg} is best in low SNR values, but the performance cannot be tuned with the channel as in the sequential search algorithms. The SEE-LSD algorithm implementation is the most efficient in high SNR values, but is the least efficient in low SNR because of the algorithm search strategy. The IR-LSD algorithm implementation is more efficient at low SNR compared to the SEE-LSD algorithm implementation and more efficient at high SNR compared to the K-best-LSD algorithm implementation. Both sequential search algorithm implementations perform much better compared to the K-best-LSD algorithm implementation in highly correlated channels with the cost of additional complexity. The performance of the sequential search algorithms can also be tuned with the penalty of additional complexity according to the requirements.

We also calculated the required parallel LSD algorithm resources with $0.18\mu m$ CMOS technology for a downlink receiver in a 3GPP LTE standard transmission with 20 MHz bandwidth and with $N_{\text{used}} = 1200$. We assume a 4×4 MIMO–OFDM system with 16-QAM, i.e., the LSD algorithm must be capable of the detection rate of 268.8 Mbps. The required $0.18\mu m$ CMOS technology resources are scaled linearly from the LSD algorithm implementation results and are listed in Table 35. The required resources with IR-LSD algorithm and SEE-LSD algorithm implementations depend on the defined performance of the algorithms as discussed earlier in this Section.

5.2.4 LLR calculation

The max-log-MAP LLR calculation unit implementation is based on the architecture presented in Figure 56 in Section 4.2.5. The implementation is designed for LSD algorithm output list size $N_{\text{cand}} = 15$, which was used in the SEE-LSD algorithm and IR-LSD algorithm implementations. The LLR calculation unit consists of two sepa-

Table 36. Complexity and latency of different LLR calculation unit microarchitecture implementations.

Microarchitecture	**1**	**2**	**3**	**4**
Parallelism (scale/max)	1/1	5/15	5/full	2/15
Pipelined loop (scale/max)	no/no	yes/yes	yes/no	yes/yes
Area estimate	98k	358k	1737k	284k
Latency (μs)	1.20	0.112	0.052	0.132
Work factor W_{LLR}	0.117	0.040	0.090	0.037

rate parts, which execute the scaling of the candidate list ED values and the search for maximum ED values for each bit b_k.

The LLR calculation unit can be implemented with various different microarchitectures with different levels of parallelism and pipelining. We used the Catapult C Synthesis tool to compare the different microarchitecture implementations and to select the most efficient microarchitecture. The different microarchitecture implementations are listed in Table 36. The parallelism and pipelining rows denote the amount of parallelism in the scaling of the ED values and the search for the maximum ED values and also if pipelining is applied to the corresponding part. Then a LLR work factor value W_{LLR} is calculated as a product between the area estimate and the latency of the microarchitecture implementation to compare the efficiencies of the microarchitectures. A smaller W_{LLR} denotes more efficient microarchitecture. It can be seen from Table 36 that the design number 4 is the most efficient microarchitecture. Thus, the implementation was done with two parallel and pipelined MULs in the scaling of the EDs and the maximum ED search operation was executed for one bit with parallel and pipelined logic, i.e., 15 EDs are checked at one sequential iteration. The same implementation can be used for both 16- and 64-QAM. The difference with processing a larger constellation size is that the same list size is used for a larger number of bits b_k, i.e., the maximum ED logic loop is executed more times. The synthesis results of the LLR calculation unit implementation for the Virtex-5 FPGA chip are listed in Table 37. The FPGA implementation requires 629 slices and 2 DSP48s. The latency of the soft output calculation for a single subcarrier is 246 ns and 300 ns for the 16- and 64-QAM, respectively. The corresponding detection rates $R_{\mathrm{det}}^{\mathrm{FPGA}}$ are then 65.0 Mbps and 80.0 Mbps for the 16- and 64-QAM. The synthesis results for the $0.18\mu m$ CMOS technology are listed in Table 38. The ASIC implementation complexity is 18.5 kGEs and it consumes 25.6 mW power. The latency of the soft output calculation for a single subcarrier is 132 ns and 164 ns

Table 37. Synthesis results of the LLR calculation unit for a Xilinx Virtex-5 FPGA chip.

FPGA	Slices	BRAMs	DSP48s	Latency	Rate $R_{\text{det}}^{(\text{fpga})}$
16-QAM	629	0	2	246 ns per subc.	65.0 Mbps
64-QAM	629	0	2	300 ns per subc.	80.0 Mbps

Table 38. Synthesis results of the LLR calculation unit for 0.18μm CMOS technology.

ASIC	**Area** (mm^2)	kGEs	Latency	**Power** (mW)	Rate $R_{\text{det}}^{(\text{asic})}$
16-QAM	0.23	18.5	132 ns per subc.	25.6	121.2 Mbps
64-QAM	0.23	18.5	164 ns per subc.	25.6	146.3 Mbps

Table 39. The required LLR calculation unit ASIC resources in the LSD detection of 3GPP LTE standard with 20MHz BW and with $N_{\text{used}} = 1200$.

4×4 **MIMO–OFDM, 16-QAM**	**Area** (mm^2)	**kGEs**	**Power** (mW)
LLR calculation unit	0.51	40.9	57

for the 16- and 64-QAM, respectively. The corresponding detection rates $R_{\text{det}}^{(\text{asic})}$ are then 121.2 Mbps and 146.3 Mbps for the 16- and 64-QAM. We also calculated the required parallel LLR calculation unit resources with 0.18μm CMOS technology for a downlink receiver in 3GPP LTE standard transmission with 20 MHz bandwidth and with $N_{\text{used}} = 1200$. We assume a 4×4 MIMO–OFDM system with 16-QAM, i.e., the LLR calculation unit must be capable of a detection rate of 268.8 Mbps. The required 0.18μm CMOS technology resources are listed in Table 39.

5.2.5 Conclusions

The implementation results of a list sphere detector and its subunits were presented in this section for both FPGA and ASIC technologies. The results included the complexity and the hardware throughput time, and also the power usage for the ASIC results. The preprocessing unit and the LSD algorithm unit are the most complex and power consuming parts of an LSD.

The SQRD algorithm implementation requires high fixed-point word lengths, which partly lead to a high complexity. The complexity could be possibly reduced with a different algorithm choice for decomposition and applying word length scaling in the algorithm. In practice, the recalculation interval for the channel matrix decomposition can be adjusted and adjacent subcarriers might be assumed to be correlated, which would lead to reduced complexity.

The implementation results of the LSD algorithms indicated that the choice of the most suitable algorithm depends on the applied system requirements. Different performance, complexity and power usage requirements may lead to a different LSD algorithm choice. The K-best-LSD algorithm implementation is fairly efficient with a fixed detection rate, but the performance suffers in highly correlated channel realizations. The tradeoff between complexity and performance of both IR-LSD and SEE-LSD algorithm implementations can be adjusted with the L_{node} variable. This may be beneficial in systems with different bandwidth modes, such as the 3GPP LTE [18] and WiMAX [19] standards, as the existing resources can be scheduled more efficiently to the work load, i.e., higher L_{node} values can be applied and better performance achieved with lower bandwidth systems. The IR-LSD algorithm implementation was more efficient in difficult channel realizations and at low SNR while the SEE-LSD algorithm implementation is the most efficient in good channel realization and at high SNR.

The max-log-MAP LLR calculation unit complexity and power usage is rather minor compared to the preprocessing unit and the LSD algorithm unit. Additional performance gain could be achieved in LSD by applying the Jacobian algorithm, i.e., log-MAP, in the LLR calculation unit.

5.3 Measurements

The measurements were done with two different hardware platforms. Real-time measurement results with LMMSE detector implementations and an EB4G hardware platform are introduced in Section 5.3.1. Non-real-time measurement results with all considered detector algorithms and a RACE hardware platform are introduced in Section 5.3.2.

5.3.1 EB4G platform

The performance measurements of LMMSE detector implementations were done with an FPGA based EB4G MIMO–OFDM hardware testbed [189]. A Propsim C8 MIMO channel emulator [190] was used to create the real-time baseband (BB) or radio frequency (RF) channels in the 2×2 MIMO system. A photo of the measurement configuration is shown in Figure 57 and the EB4G main technical parameters are listed in Table 40. A coded SM transmission with quadrature phase shift keying (QPSK) and with bit-interleaving is applied with a soft output LMMSE detector and a Viterbi decoder at the receiver. The 1/2 rate convolutional code is applied with [171_o,133_o] polynomial and the coding is done over one OFDM symbol interval. A least squares based channel estimator is used in the EB4G with two OFDM pilot symbols per frame [191]. A correct SNR value is used in the measurements with no estimation error. Two different non line-of-sight (NLOS) channel profiles from the WINNER [152, 153] channel model are used as a channel in the measurements. The effect of channel correlation properties is measured by using different antenna separations. An antenna separation of 0.5λ, i.e., 1/2 the wavelength of the used transmission frequency, is used at the receiving mobile terminal (MT). Antenna separations of 0.5λ, 4λ, and 10λ are used at the transmitting base station (BS). A lower value of antenna separation corresponds to more correlated antennas.

The FPGA implementations of the LMMSE detectors for 2×2 antennas were mapped to a Xilinx Virtex-II XC2V6000 chip in the EB4G testbed with the Xilinx NGDBuild software tool. The performances of both CORDIC and SGR based LMMSE detector implementations were measured at BB or RF and compared to computer simulation results. The computer simulations have been done in Matlab with a floating point representation. The least squares channel estimator used in the real-time measurements is modeled to Matlab by adding noise to the channel coefficients [191]. The LMMSE coefficient calculation operates at a frame interval and the data detection, i.e., a matrix-vector multiplication, is done for all 52 data subcarriers in a 4 μs OFDM symbol interval. Performance results of the LMMSE detector in the WINNER A1 channel are shown in Figures 58 and 59 for CORDIC and SGR based implementations, respectively. The bit error rate (BER) performances of the LMMSE detector implementations are shown at both BB and RF. It can be noted that the performance loss due to increased correlation between antennas is higher in the measurements compared to the simulated results. The total performance loss of the measured BB system compared to the simulation results is between 2–2.5dB at 1% BER with BS antenna separations of 4λ and 10λ. The antenna separation of 0.5λ, i.e., the high correlation case, results in higher implementation loss. The WINNER A1 channel scenario is rather flat fading and, in the case of a difficult channel realization, it affects multiple subcarriers. Thus, the channel coding and the interleaving are not able to correct the errors. It can be seen from the measurements that the BER performance of the SGR based detector starts to saturate

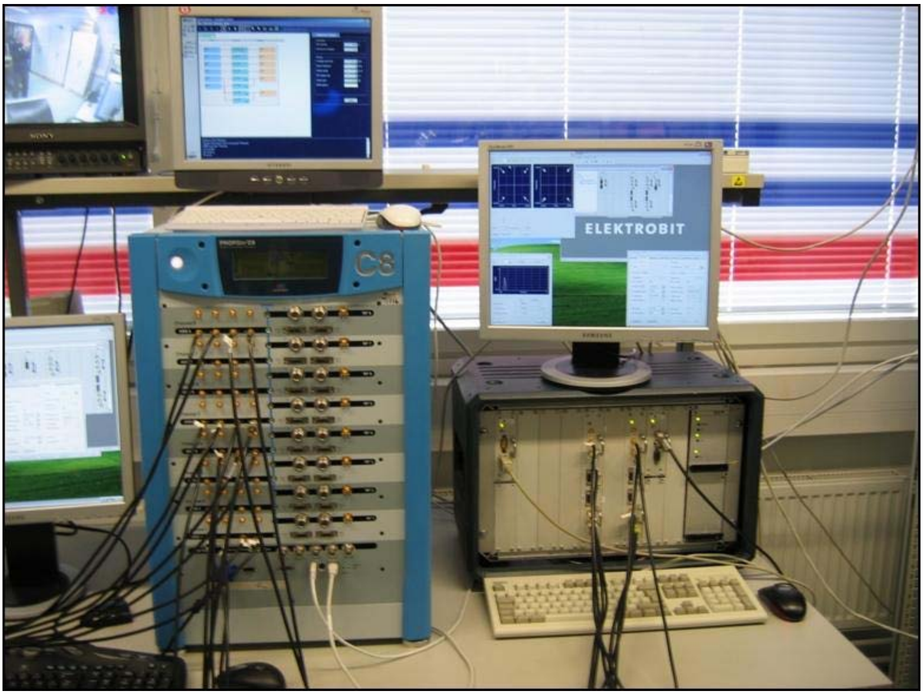


Fig 57. The measurement configuration consisting of a Propsim C8 channel emulator on the left and an EB4G hardware testbed on the right. ([142], published by permission of IEEE).

Table 40. The main parameters of the EB4G platform.

Parameter	Used configuration
RF operating frequency range	2400–2500 MHz
Sample rate	20 MHz
Total signal bandwidth	16.9 MHz
FFT size	64 (52 data subcarriers)
OFDM symbol duration	3.2 μs FFT + 0.8 μs CP
Frame length	80 μs
Data symbols in one frame	16
Convolutional code	1/2 rate, $[171_o,133_o]$ polynomial
Convolutional decoder	Viterbi decoder
Channel estimator	least squares based estimator

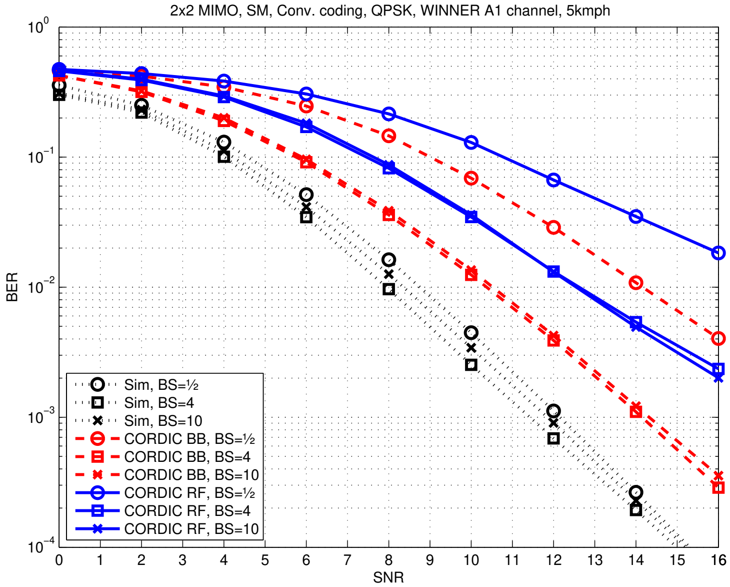


Fig 58. A convolutional coded SM system with the CORDIC based LMMSE detector and the Viterbi decoder in the WINNER A1 channel with 5kmph velocity. ([142], published by permission of IEEE).

earlier than the performance of the CORDIC based detector. This is due to the high dynamic range requirements of the SGR algorithm. The performance loss due to the RF units is approximately 2 dB compared to the BB system. Performance results of the LMMSE detector in the WINNER B1 channel at BB are shown in Figure 60. The performance loss between the BB measurements and the simulation results is between 1.5–2dB at 1% BER for 4λ and 10λ antenna separations. The performance loss with a 0.5λ separation is now lower compared to the WINNER A1 channel. In this case the WINNER B1 channel is more frequency selective, and, thus, the channel coding and the interleaving are able to decrease the effect of bad channel realizations. The performance gap between floating point simulations and fixed-point implementation is also smaller as the occasional errors due to implementation are being corrected more effectively by the channel code. The measurements illustrate the performance of the SM system with a soft output LMMSE detector at the receiver. The performance loss of a measured coded fixed-point implemented system at BB compared to a simulated floating point system varied between 0.5–7dB at a 1% target BER. The measurements showed that the performance of the SM system with a fixed-point representation at the

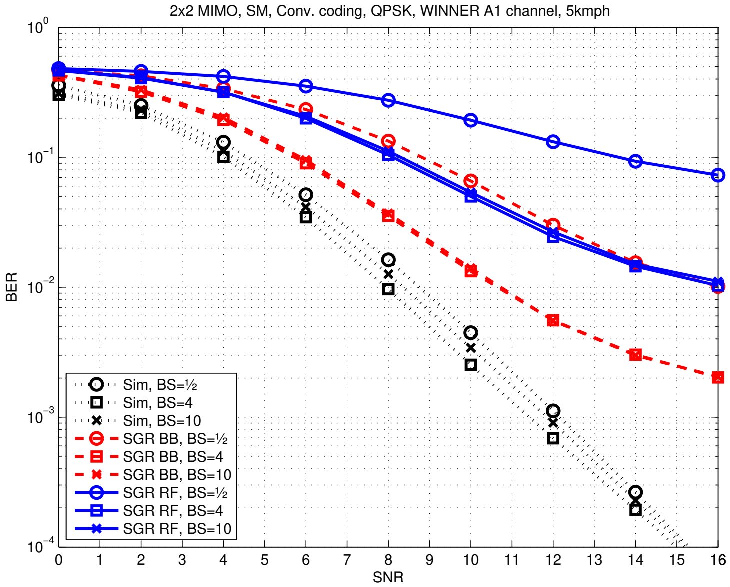


Fig 59. A convolutional coded SM system with the SGR based LMMSE detector and the Viterbi decoder in the WINNER A1 channel with 5kmph velocity. ([142], published by permission of IEEE).

receiver is highly dependent on the channel correlation properties. The channel realization with higher correlation properties results in a higher eigenvalue spread of the channel matrix, which leads to higher dynamic range in the signal representation in the calculation of **W**. It can be concluded that the applied word lengths were insufficient for the A1 channel environment with 0.5λ antenna separation at BS and that the performance loss compared to the simulated results was ≥5dB. Thus, in the case of highly correlated antennas, the larger word length requirements should be recognized. The measurements show that the CORDIC based LMMSE detector is less vulnerable to difficult channel realizations compared to the SGR based LMMSE detector. Thus, the CORDIC based LMMSE detector implementation is more attractive choice due to more feasible fixed-point properties.

5.3.2 RACE platform

The Radio Access Emulator (RACE) platform [192] and the Propsim C8 MIMO channel emulator [190] were used to perform non-real-time measurements with detector

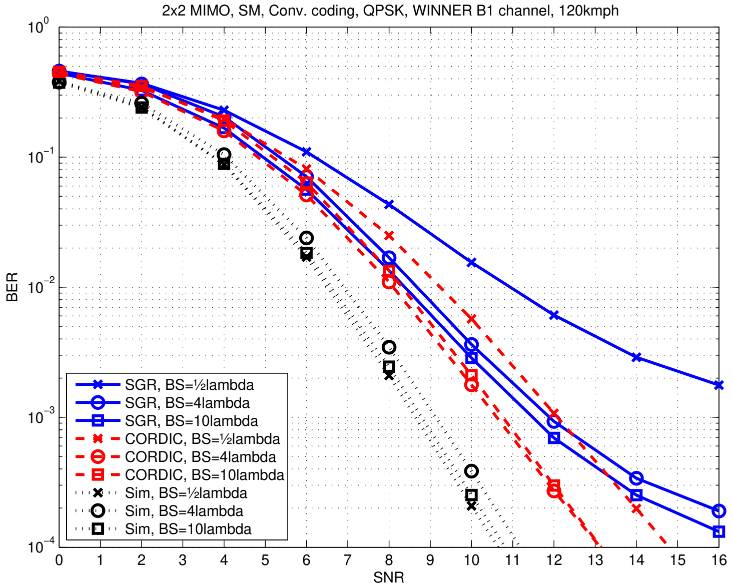


Fig 60. Convolutional coded SM system with a LMMSE detector and Viterbi decoder in the WINNER B1 channel with 120kmph velocity. ([142], published by permission of IEEE).

algorithms. The RACE platform is a MIMO–OFDM capable software defined radio prototyping platform utilizing Matlab and HW acceleration via FPGA signal processing circuits. The OFDM front end, the digital to analog domain conversions, and the RF parts operating in the ranges of 300–2700 MHz and 3500–5800 MHz are executed in real-time in HW. The other baseband algorithms such as channel coding/decoding and detection are executed non-real-time in a Matlab environment. The signal flow between the real-time and non-real-time parts is buffered and the data transmission and reception is executed in bursts. The RACE platform functionality is illustrated in a block diagram in Figure 61 and a picture of the measurement configuration is shown in Figure 62. The RACE platform was configured for link level measurements as a 2×2 MIMO–OFDM system with the same parameters as in Section 2.6. A WINNER B1 channel model was used with Propsim with antenna separations of 0.5λ and 4λ at MS and BS, respectively. The K-best-LSD, the SEE-LSD, the IR-LSD and the LMMSE detectors were executed in Matlab in RACE with floating point representation. The FER measurement results with a RACE platform and the computer simulations results of a turbo coded 2×2 SM system with 16- and 64-QAM and with different detectors

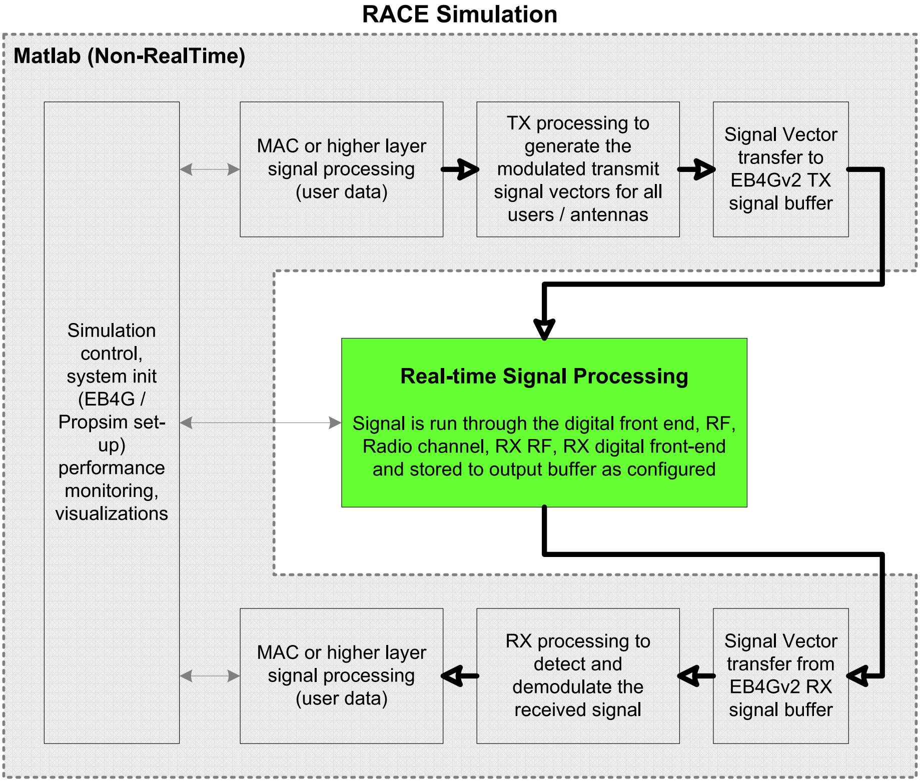

RACE Simulation
Matlab (Non-RealTime)
Simulation control, system init (EB4G / Propsim set-up) performance monitoring, visualizations
MAC or higher layer signal processing (user data)
TX processing to generate the modulated transmit signal vectors for all users / antennas
Signal Vector transfer to EB4Gv2 TX signal buffer
Real-time Signal Processing
Signal is run through the digital front end, RF, Radio channel, RX RF, RX digital front-end and stored to output buffer as configured
MAC or higher layer signal processing (user data)
RX processing to detect and demodulate the received signal
Signal Vector transfer from EB4Gv2 RX signal buffer

Fig 61. The RACE platform functionality.

and a turbo decoder are shown in Figure 63. It can be seen that the performance of the detector algorithms in the RACE platform is within 1 dB until approximately 2% FER. The saturation of performance is due to the noise floor of the RACE platform, which is caused by the combination of finite word lengths in the OFDM front end and digital to analog domain conversions, and RF parts. The measurement results verify the differences in performance between different detector algorithms. The LSD algorithms perform better compared to the LMMSE detector. SEE-LSD and IR-LSD with N_{cand} with a rather large L_{node} also outperform the K-best-LSD with $K = 15$ as expected.

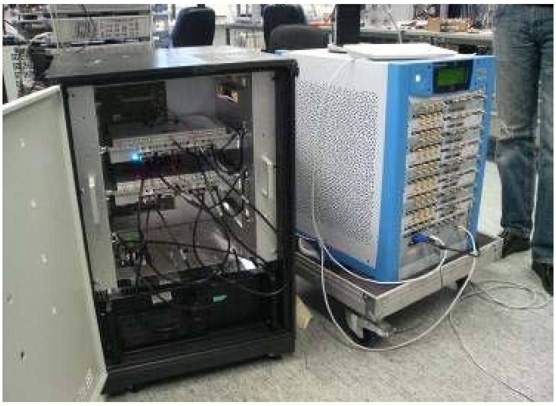


Fig 62. The measurement configuration consisting of a Propsim C8 channel emulator on the right and a RACE platform on the left.

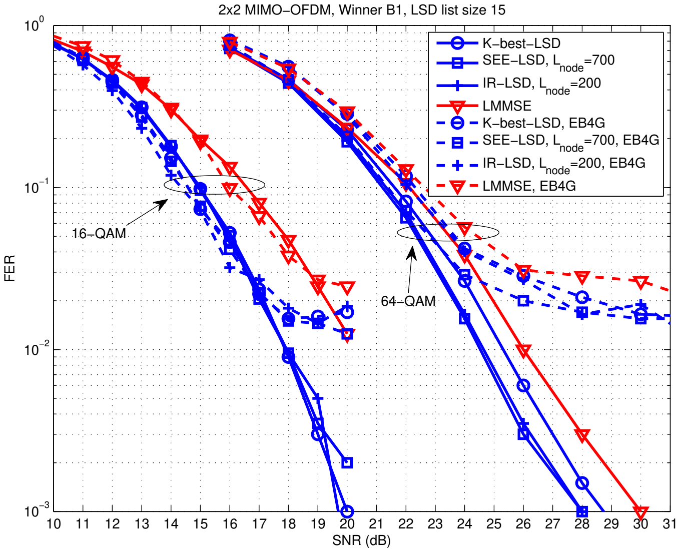


Fig 63. Turbo coded SM system with different detectors and turbo decoder in the WINNER B1 channel with a 60kmph velocity.

5.4 Summary

We presented the FPGA synthesis results of the CORDIC based LMMSE detector for 2×2 and 4×4 MIMO systems and an SGR based LMMSE detector for a 2×2 MIMO system. The CORDIC based design requires more slices and less block multipliers compared to the SGR based design. We also compared the latencies of the implementations to the requirements of a downlink receiver in different telecommunication standards. The 2×2 LMMSE detector implementations are fast enough as such to be used in a receiver in different standards with maximum bandwidths. The 4×4 CORDIC implementation is also feasible in most cases.

The list sphere detector implementations, which included the SQRD algorithm, the LSD algorithms, and the max-log-MAP LLR calculation unit, were presented for a 4×4 MIMO system. The synthesis results and operating rates were presented for both FPGA chip and ASIC technology. The SQRD implementation was done to support up to 16-QAM and the other LSD units were done to support up to 16- and 64-QAM. The SQRD preprocessing unit and the LSD algorithm unit are the most complex and power consuming parts of a list sphere detector while LLR calculation unit complexity and power usage is rather minor compared to the other units. The high fixed-point word length requirements of the SQRD algorithm implementation lead to a high complexity, which could be compensated for by applying word length scaling. The implementation results of the LSD algorithms indicated that the choice of the most suitable algorithm depends on the applied system requirements. Different performance, complexity and power usage requirements may lead to a different LSD algorithm choice. The K-best-LSD algorithm implementation is a good tradeoff between complexity and performance if certain performance degradation can be tolerated. The possibility to adjust the tradeoff between complexity and performance of both IR-LSD and SEE-LSD algorithm implementations is beneficial in systems with different bandwidth modes. We also estimated the total complexity and power usage of the required parallel resources of different LSD units for a downlink receiver in 3GPP LTE standard transmission with 20 MHz bandwidth.

Real-time measurement results were presented with the 2×2 LMMSE detector implementations and an EB4G hardware platform. The measurements showed that the CORDIC based LMMSE detector is less vulnerable to difficult channel realizations compared to the SGR based LMMSE detector due to more feasible fixed-point properties. Non-real-time measurement results were presented with all considered detector algorithms and RACE hardware platform with a 2×2 MIMO configuration. The measurement results verified the differences in performance between different detector algorithms. The LSD algorithms perform better compared to the LMMSE detector. SEE-LSD and IR-LSD with N_{cand} with a rather large L_{node} also outperform the K-best-LSD with $K = 15$ as expected.

6 Conclusion and future work

The scope of this thesis in a broad sense was to develop receiver algorithms for advanced transmission techniques which enable more efficient use of the radio spectrum. This was motivated by the fact that the radio spectrum is a limited resource and, thus, in order to serve the increasing demand, the spectrum should be used more efficiently. Moreover, the objective was to study and compare the implementation complexity and performance of different detector algorithms. Another objective of this thesis was to apply a joint process between different development phases to ensure efficient algorithm implementation.

Chapter 2 introduced the MIMO–OFDM system model and the considered detector algorithms. The main emphasis was given to the linear MMSE detector, which was implemented via CORDIC or SGR algorithms, and the list sphere detector. Three LSD algorithms with different search methods were introduced, namely the K-best-LSD algorithm, the SEE-LSD algorithm, and the IR-LSD algorithm. The performances of the detector algorithms were compared via numerical examples. It was shown that the max-log-MAP performance could be achieved with the LSD based receivers with a much lower computational cost than with the exhaustive search algorithm. It was also shown that a significant number of operations are saved with the SQRD algorithm applied as LSD preprocessing. The performance of the linear MMSE detector is adequate in uncorrelated channel realization although the performance suffers significantly in highly correlated channel realization.

Chapter 3 introduced different methods and implementation tradeoffs to modify the list sphere detector to be less complex and feasible for implementation. It was shown that LLR clipping lowers the required LSD candidate list size, and, thus, lowers the complexity of an LSD. It was also shown that the real valued signal model is clearly the better choice to be applied with LSD algorithms given the number of visited nodes and the complexity difference in required operations in one node check. Two methods were proposed to limit and fix the search complexity of the LSD sequential search algorithms: a limited search method and a scheduled search method. The numerical results showed clearly that both considered sequential search algorithms work well when the search is limited and both proposed methods are feasible for implementation. A minor performance gain can be achieved with the more sophisticated and complex SS method.

A novel memory sphere radius was proposed to be applied with the IR-LSD algorithm to decrease the number of stored candidates and the complexity of the required minimum search during the algorithm tree search. The numerical results showed that the average number of visited nodes could be decreased with the proper selected memory sphere radius without losing performance. The use of the L1 norm in the LSD algorithm tree search, which decreases the complexity of the search, but also degrades the performance of the LSD, was also studied. A simple scaling of the LSD algorithm output list distance metric values was proposed to approximate the L2 norm Euclidean distance. It was shown that the scaling method could be used to compensate for the performance loss due to the use of the L1 norm with minor additional complexity. The performance of the iterative receiver was studied with a variable number of global iterations and their impact on the computational complexity was analyzed. The results indicated that with a typical system configuration it is beneficial to use the additional computational cost for an LSD algorithm to achieve a better soft output approximation, i.e., study more tree nodes in the algorithm search. The use the 2nd global iteration with a lower amount of decoder resources is also justified.

Chapter 4 focused to the architecture design for the considered detection algorithms. The key functional microarchitecture units were identified and introduced and the possibilities for parallelism and pipelining were exploited. The introduced LMMSE detector architectures were based on systolic array structures with pipelined communicating processing elements. The triangular or linear array structures presented can be easily scaled for the systems with different antenna configurations with modified array structure. Architectures were also designed for different list sphere detector subunits. Architectures were introduced for the SQRD preprocessing algorithm, for the considered LSD algorithms and for the max-log-MAP LLR calculation unit. A parallel and pipelined architecture was designed for the K-best-LSD algorithm, which makes the algorithm suitable for hardware implementation. Scalable architectures were designed for the SEE-LSD algorithm and the IR-LSD algorithm, which operate in a sequential fashion. The architectures include implementation tradeoffs introduced in an earlier section. It was noted that multiple architecture units can be used in parallel to support higher data rate requirements in an OFDM system.

The hardware implementation results and the measurement results on hardware testbeds were introduced in Chapter 5. The FPGA synthesis results of the CORDIC based LMMSE detector were presented for 2×2 and 4×4 MIMO systems and an SGR based LMMSE detector was presented for a 2×2 MIMO system. It was noted that the

CORDIC based design required more slices and less block multipliers compared to the SGR based design. The latencies of the implementations were also compared to the requirements of a downlink receiver in different telecommunication standards. The 2×2 LMMSE detector implementations were fast enough as such to be used in a receiver in different standards with maximum bandwidths. The 4×4 CORDIC implementation was also feasible in most of the cases. The list sphere detector hardware implementations included the SQRD algorithm, the LSD algorithms, and the max-log-MAP LLR calculation unit, and the implementations were done for a 4×4 MIMO system. The synthesis results and operating rates were presented for both FPGA chip and ASIC technology. The SQRD implementation was done to support up to 16-QAM and the other LSD units were done to support up to 16- and 64-QAM. The SQRD preprocessing unit and the LSD algorithm unit are the most complex and power consuming parts of a list sphere detector while LLR calculation unit complexity and power usage is rather minor compared to the other units. The high fixed-point word length requirements of the SQRD algorithm implementation leads to a high complexity, which could be compensated for by applying word length scaling. The implementation results of the LSD algorithms indicated that the choice of the most suitable algorithm depends on the applied system requirements. Different performance, complexity and power usage requirements may lead to different LSD algorithm choice. The K-best-LSD algorithm implementation is a good tradeoff between complexity and performance if certain performance degradation can be tolerated. The possibility to adjust the tradeoff between complexity and performance of both IR-LSD and SEE-LSD algorithm implementations is beneficial in systems with different bandwidth modes. The total complexity and power usage of required parallel resources of different LSD units was also estimated for a downlink receiver in 3GPP LTE standard transmission with 20 MHz bandwidth.

Real-time measurement results were presented with the 2×2 LMMSE detector implementations and a EB4G hardware platform. The measurements showed that the CORDIC based LMMSE detector was less vulnerable to difficult channel realizations compared to the SGR based LMMSE detector due to better fixed-point properties. Non-real-time measurement results were presented with all considered detector algorithms and a RACE hardware platform with a 2×2 MIMO configuration. The measurement results verified the differences in the performance between different detector algorithms. It was verified that the LSD algorithms perform better compared to the LMMSE detector.

The results presented in this thesis showed that the list sphere detector is feasible for implementation and it can be applied to receivers in the latest telecommunication standards. The LSD offers close to optimal max-log-MAP performance and outperforms the linear LMMSE detector especially in a correlated channel environment with a price of additional complexity. The implementation aspects and tradeoffs related to LSD implementation offer still some interesting research topics and could be further investigated to provide even less complex practical solutions. In addition, it would be interesting to compare the performance and complexity of the considered LSD algorithms in more detail to the single-tree-search based soft output SD algorithms. Practical problems, such as the affect of channel estimation and synchronization on the performance, could also be studied. Furthermore, the thesis focused on the detector design and implementation for a link level system. The system level perspective and, e.g., the use of cooperative communication systems, provide additional possibilities and open questions on detector implementation, which should be further studied. However, the results in this thesis provide a good background for these additional topics.

References

1. Foschini GJ & Gans MJ (1998) On limits of wireless communications in a fading environment when using multiple antennas. Wireless Personal Communications, Kluwer Academic Publishers 6: 311–335.
2. Paulraj AJ, Gore DA, Nabar RU & Bolcskei H (2004) An overview of MIMO communications — A key to gigabit wireless. Proceedings of the IEEE 92(2): 198–218.
3. Yang H (2005) A road to future broadband wireless access: MIMO-OFDM-based air interface. IEEE Transactions on Magnetics 43: 53–60.
4. Hochwald B & ten Brink S (2003) Achieving near-capacity on a multiple-antenna channel. IEEE Transactions on Communications 51(3): 389–399.
5. Redl S, Wever M & Oliphant M (1998) GSM and Personal Communications Handbook. Artech House mobile communications library.
6. Heine G & Sagkob H (2003) GPRS: Gateway to Third Generation Mobile Networks. Artech House, Incorporated.
7. Halonen T, Romero J & Melero J (2002) GSM, GPRS and EDGE Performance: Evolution Towards 3G/UMTS. John Wiley & Sons, first edition.
8. Goldsmith A (2005) Wireless Communications. Cambridge University Press, New York, USA.
9. Tse D & Viswanath P (2005) Fundamentals of Wireless Communication. Cambridge University Press, Cambridge, UK.
10. 3rd Generation Partnership Project (3GPP). http://www.3gpp.org.
11. Holma H & Toskala A (eds.) (2004) WCDMA for UMTS – Radio Access For Third Generation Mobile Communications. John Wiley and Sons, New York, 3rd edition.
12. Holma H & Toskala A (2004) HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobile Communications. John Wiley & Sons, Ltd, 1st edition.
13. Chang R (1966) Synthesis of band-limited orthogonal signals for multichannel data transmission. Bell Systems Technical Journal 45: 1775 – 1796.
14. Weinstein SB & Ebert PM (1971) Data transmission by frequency division multiplexing using the discrete Fourier transform. IEEE Transactions on Communication Technology 19(5): 628–634.
15. Wang Z & Giannakis GB (2000) Wireless multicarrier communications. IEEE Signal Processing Magazine 17(3): 29–48.
16. ANSI/IEEE Standard 80211,1999 Edition (R2003) (2003). Information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications.
17. IEEE Standard 80216 (2004). IEEE standard for local and metropolitan area networks part 16: Air interface for fixed broadband wireless access systems.
18. 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network (2007) Evolved universal terrestrial radio access (e-utra); user equipment (ue) radio transmission and reception (release 8). Technical report, 3rd Generation Partnership Project (3GPP).

19. IEEE Standard 80216-2005 (2005). IEEE standard for local and metropolitan area networks part 16: Air interface for fixed and mobile broadband wireless access systems.
20. Pietrzyk S (2006) OFDMA for Broadband Wireless Access. Artech House mobile communications.
21. H Myung DG (2008) Single Carrier FDMA: A New Air Interface for Long Term Evolution. John Wiley & Sons, Ltd.
22. Winters J (1987) On the capacity of radio communication systems with diversity in a Rayleigh fading environment. IEEE Journal on Selected Areas in Communications 5(5): 871–878.
23. Boelcskei H, Gesbert D, Papadias CB & van der Veen AJ (2006) Space-Time Wireless Systems: From Array Processing to MIMO Communications. Cambridge University Press, Cambridge, UK.
24. Goldsmith A, Jafar S, Jindal N & Vishwanath S (June, 2003) Capacity limits of MIMO channels. IEEE Journal on Selected Areas in Communications 21(5): 684–702.
25. Gesbert D, Shafi M, Shiu D, Smith PJ & Naguib A (2003) From theory to practice: An overview of MIMO space-time coded wireless systems. IEEE Journal on Selected Areas in Communications 21(3): 281–302.
26. Valkama M, Shahed hagh ghadam A, Anttila L & Renfors M (2006) Advanced digital signal processing techniques for compensation of nonlinear distortion in wideband multicarrier radio receivers. IEEE Transactions on Microwave Theory and Techniques 54(6): 2356 –2366.
27. Zou Y, Valkama M & Renfors M (2008) Digital Compensation of I/Q Imbalance Effects in Space-Time Coded Transmit Diversity Systems. IEEE Transactions on Signal Processing 56(6): 2496 –2508.
28. Godara L (1997) Applications of antenna arrays to mobile communications. I. Performance improvement, feasibility, and system considerations. Proceedings of the IEEE 85(7): 1031 –1060.
29. Paulraj A, Nabar R & Gore D (2003) Introduction to Space-Time Wireless Communications. Cambridge University Press, Cambridge, UK.
30. Bölcskei H, Gesbert D, Papadias C & van der Veen A (2006) Space-Time Wireless Systems: From Array Processing to MIMO Communications. Cambridge University Press, Cambridge, UK.
31. Tujkovic D (2003) Space-Time Turbo Coded Modulation for Wireless Communication Systems, volume C184 of *Acta Universitatis Ouluensis, Doctoral thesis*. University of Oulu Press, Oulu, Finland.
32. Hassibi B & Hochwald BM (2002) High-rate codes that are linear in space and time. IEEE Transactions on Information Theory 48(7): 1804–1824.
33. Tarokh V, Seshadri N & Calderbank AR (1998) Space-time codes for high data rate wireless communication: Performance criterion and code construction. IEEE Transactions on Information Theory 44(2): 744–765.
34. Ungerboeck G (1982) Channel coding with multilevel/phase signals. IEEE Transactions on Information Theory 28(1): 55–67.
35. Alamouti S (1998) A simple transmit diversity technique for wireless communications. IEEE Journal on Selected Areas in Communications 16(8): 1451–1458.
36. Tarokh V, Jafarkhani H & Calderbank AR (1999) Space-time block codes from orthogonal designs. IEEE Transactions on Information Theory 45(5): 1456–1467.

37. Telatar E (1999) Capacity of multi-antenna Gaussian channels. European Transactions on Telecommunications 10(6): 585–595.
38. Foschini G (1996) Layered space–time architecture for wireless communication in a fading environment when using multi-element antennas. Bell Labs Technical Journal 1(2): 41–59.
39. Wolniansky PW, Foschini GJ, Golden GD & Valenzuela RA (1998) V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel. In: International Symposium on Signals, Systems, and Electronics (ISSSE), pp. 295–300. Pisa, Italy.
40. Foschini GJ, Golden GD, Valenzuela RA & Wolniansky PW (1999) Simplified processing for high spectral efficiency wireless communication employing multi-element arrays. IEEE Journal on Selected Areas in Communications 17(11): 1841–1852.
41. Foschini G, Chizhik D, Gans M, Papadias C & Valenzuela R (2003) Analysis and performance of some basic space-time architectures. IEEE Journal on Selected Areas in Communications 21(3): 303 – 320.
42. Bolcskei H (2006) MIMO-OFDM wireless systems: basics, perspectives, and challenges. IEEE Wireless Communications 13(4): 31 – 37.
43. Mietzner J, Schober R, Lampe L, Gerstacker W & Hoeher P (2009) Multiple-antenna techniques for wireless communications - a comprehensive literature survey. IEEE Communications Surveys Tutorials 11(2): 87 – 105.
44. Boubaker N, Letaief K & Ross D (2002) Performance of BLAST over frequency-selective wireless communication channels. IEEE Transactions on Communications 50(2): 196 – 199.
45. Cimini LJ (1985) Analysis and simulation of a digital mobile channel using orthogonal frequency division multiplexing. IEEE Transactions on Communications 33(7): 665–675.
46. Hwang T, Yang C, Wu G, Li S & Ye Li G (2009) OFDM and Its Wireless Applications: A Survey. IEEE Transactions on Vehicular Technology 58(4): 1673 – 1694.
47. Hassibi B & Vikalo H (2005) On the sphere-decoding algorithm I. expected complexity. IEEE Transactions on Signal Processing 53(8): 2806–2818.
48. Agrell E, Eriksson T, Vardy A & Zeger K (2002) Closest point search in lattices. IEEE Transactions on Information Theory 48(8): 2201–2214.
49. Vikalo H & Hassibi B (2006) On joint detection and decoding of linear block codes on Gaussian vector channels. IEEE Transactions on Signal Processing 54(9): 3330–3342.
50. Bahl LR, Cocke J, Jelinek F & Raviv J (1974) Optimal decoding of linear codes for minimizing symbol error rate. IEEE Transactions on Information Theory 20(2): 284–287.
51. Robertson P, Villebrun E & Hoeher P (1995) A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain. Proc. IEEE Int. Conf. Commun. pp. 1009–1013.
52. Garrett D, Davis L & Woodward GK (2003) 19.2 Mbit/s 4x4 BLAST/MIMO detector with soft ML outputs. IEE Electronics Letters 39(2): 233–235.
53. Garrett D, Woodward GK, Davis L & Nicol C (2005) A 28.8 Mbit/s 4x4 MIMO 3G CDMA receiver for frequency selective channels. IEEE Journal of Solid-State Circuits 40(1): 320–3302.
54. Burg A, Felber N & Fichtner W (2003) A 50 Mbps 4x4 maximum likelihood decoder for multiple-input multiple-output systems with QPSK modulation. In: In Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS'03), volume 1, pp. 332–335.

55. Graef N, Hammerschmidt J & Sundberg CE (2009) A low-complexity max-log-MAP detector. IEEE Transactions on Communications 57(8): 2251–2254.
56. Lupas R & Verdú S (1989) Linear multiuser detectors for synchronous code-division multiple-access channels. IEEE Transactions on Information Theory 34(1): 123–136.
57. Xie Z, Short RT & Rushforth CK (1990) A family of suboptimum detectors for coherent multiuser communications. IEEE Journal on Selected Areas in Communications 8(4): 683–690.
58. Artes H, Seethaler D & Hlawarsch F (2003) Efficient detection algorithms for MIMO channels: A geometrical approach to approximate ML detection. IEEE Transactions on Signal Processing 51(11): 2808–2820.
59. Fertl P, Jalden J & Matz G (2008) Capacity-based performance comparison of MIMO-BICM demodulators. In: In Proceedings of the IEEE 9th Workshop on Signal Processing Advances in Wireless Communications (SPAWC'08), pp. 166–170.
60. Wübben D, Böhnke R, Rinas J, Kühn V & Kammeyer KD (2001) Efficient algorithm for decoding layered space-time codes. IEE Electronic Letters 37(22): 1348–1349.
61. Choi J (2004) A bi-directional zero-forcing BLAST receiver. IEEE Transactions on Signal Processing 52(9): 2670 – 2673.
62. Chin W, Constantinides A & Ward D (2002) Parallel multistage detection for multiple antenna wireless systems. Electronics Letters 38(12): 597–599.
63. Ylioinas J & Juntti M (2009) Iterative Joint Detection, Decoding, and Channel Estimation in Turbo-Coded MIMO-OFDM. IEEE Transactions on Vehicular Technology 58(4): 1784–1796.
64. Ma X & Zhang W (2008) Performance analysis for MIMO systems with lattice-reduction aided linear equalization. IEEE Transactions on Communications 56(2): 309–318.
65. Gan YH, Ling C & Mow WH (2009) Complex Lattice Reduction Algorithm for Low-Complexity Full-Diversity MIMO Detection. IEEE Transactions on Signal Processing 57(7): 2701–2710.
66. Kyungchun Lee K, Joohwan Chun J & Hanzo L (2007) Optimal Lattice-Reduction Aided Successive Interference Cancellation for MIMO Systems. IEEE Transactions on Wireless Communications 6(7): 2438–2443.
67. Silvola P, Hooli K & Juntti M (2006) Sub-optimal soft-output MAP detector with lattice reduction. IEEE Signal Processing Letters 13(6): 321–324.
68. Wübben D, Böhnke R, Kühn V & Kammeyer K (2004) Near-maximum-likelihood detection of MIMO systems using MMSE-based lattice-reduction. In: Proc. IEEE Int. Conf. Commun., volume 2, pp. 798–802. Paris, France.
69. Lenstra AK, Lenstra HW & Lovasz L (1982) Factoring polynomials with rational coefficients. Math. Ann. 261: 515–534.
70. Golub GH & Loan CFV (1996) Matrix Computations, 3rd ed. The Johns Hopkins University Press, Baltimore.
71. Burg A, Haene S, Perels D, Luethi P, Felber N & Fichtner W (May, 2006) Algorithm and VLSI architecture for linear MMSE detection in MIMO-OFDM systems. In: In Proceeding of the IEEE International Symposium on Circuits and Systems (ISCAS'06).
72. Haykin S (1991) Adaptive Filter Theory. Prentice Hall, Englewood Cliffs, NJ, USA, 2nd edition.
73. Volder J (1959) The CORDIC Trigonometric Computing Technique. IRE Transactions on Electronic Computers EC-8(3): 330–4.

74. Döhler R (1991) Squared givens rotation. IMA Journal of Numerical Analysis 11: 1–5.
75. Gentleman W & Kung H (Bellingham, Washington, 1981) Matrix triangularization by systolic array. In: In Proc. SPIE, Real-time signal processing IV, volume 298, pp. 19–26.
76. Kung S (1987) VLSI Array Processors. Prentice-Hall.
77. Lightbody G, Woods R & Walke R (2003) Design of a parameterizable silicon intellectual property core for QR-based RLS filtering. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 11(4): 659 – 678.
78. Edman F & Öwall V (2004) An FPGA implementation of a matrix inversion architecture for multiple antenna algorithms. In: In Proceedings of the Nordic Radio Symposium. Oulu, Finland.
79. Boher L, Rabineau R & Helard M (2007) An Efficient MMSE Equalizer Implementation for 4x4 MIMO-OFDM Systems in Frequency Selective Fast Varying Channels. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 1 –5.
80. Guo Z & Nilsson P (2004) An ASIC implementation for V-BLAST Detection in 0.35 um CMOS. In: In Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology (ISSPIT'04), pp. 95 – 98.
81. Sobhanmanesh F & Nooshabadi S (2006) VLSI architecture for 4x4 16-QAM V-BLAST decoder. In: Proceedings on IEEE International Symposium on Circuits and Systems (ISCAS'06), p. 4 pp.
82. Sobhanmanesh F, Nooshabadi S & Kim K (2007) A 212 Mb/s Chip for 4x4 16-QAM V-BLAST decoder. In: In Proceedings of the 50th Midwest Symposium on Circuits and Systems (MWSCAS'07), pp. 1437–1440.
83. Ketonen J, Juntti M & Cavallaro J (2010) Performance-Complexity Comparison of Receivers for a LTE MIMO-OFDM System. IEEE Transactions on Signal Processing 58(6): 3360–3372.
84. Gestner B, Zhang W, Ma X & Anderson D (2008) VLSI Implementation of a Lattice Reduction Algorithm for Low-Complexity Equalization. In: In Proceedings of the 4th IEEE International Conference on Circuits and Systems for Communications (ICCSC'08), pp. 643 –647.
85. Barbero L, Milliner D, Ratnarajah T, Barry J & Cowan C (2009) Rapid Prototyping of Clarkson's Lattice Reduction for MIMO Detection. In: In Proceedings of the IEEE International Conference on Communications, 2009. ICC'09, pp. 1 –5.
86. Youssef A, Shabany M & Gulak P (2010) VLSI implementation of a hardware-optimized lattice reduction algorithm for WiMAX/LTE MIMO detection. In: In Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS'10), pp. 3541 –3544.
87. Bruderer L, Studer C, Wenk M, Seethaler D & Burg A (2010) VLSI implementation of a low-complexity LLL lattice reduction algorithm for MIMO detection. In: In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'10), pp. 3745 –3748.
88. Pohst M (1981) On the computation of lattice vectors of minimal length, successive minima and reduced basis with applications. ACM SIGSAM Bull. 15: 37–44.
89. Fincke U & Pohst M (1985) Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Math. Comput. 44(5): 463–471.
90. Viterbo E & Bouros J (1999) A universal lattice code decoder for fading channels. IEEE Transactions on Information Theory 45(5): 1639–1642.
91. Damen MO, Gamal HE & Caire G (2003) On maximum–likelihood detection and the search for the closest lattice point. IEEE Transactions on Information Theory 49(10): 2389–2402.

92. Damen O, Chkeif A & Belfiore JC (2000) Lattice code decoder for space-time codes. IEEE Communications Letters 4(5): 161–163.
93. Murugan A, El Gamal H, Damen M & Caire G (2006) A unified framework for tree search decoding: rediscovering the sequential decoder. IEEE Transactions on Information Theory 52(3): 933–953.
94. Anderson T (1984) An Introduction to Multivariate Statistical Analysis, Second Edition. John Wiley and Sons, New York.
95. Mohan S & Anderson JB (1984) Computationally optimal metric-first code tree search algorithms. IEEE Transactions on Communications 32(6): 710 – 717.
96. Viterbo E & Boutros J (1999) A universal lattice code decoder for fading channels. IEEE Transactions on Information Theory 45(5): 1639–1642.
97. Schnorr CP & Euchner M (1994) Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Math. Programming 66(2): 181–191.
98. Anderson J & Mohan S (1984) Source and channel coding: An algorithmic approach. IEEE Transactions on Communications 32(2): 169–176.
99. Guo Z & Nilsson P (2006) Algorithm and Implementation of the K-Best Sphere Decoding for MIMO Detection. IEEE Journal on Selected Areas in Communications 24(3).
100. Wong K, Tsui C, Cheng RK & Mow W (2002) A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels. In: In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'02), volume 3, pp. 273–276. Helsinki, Finland.
101. Dijkstra EW (1959) A note on two problems in connexion with graphs. In: Numerische Mathematik, volume 1, pp. 269–271. Mathematisch Centrum, Amsterdam, Netherlands.
102. Knuth D (1997) The Art of Computer Programming, Volume 3: Sorting and Searching, Third Edition. Addison-Wesley.
103. Zigangirov KS (1966) Some sequential decoding procedures. Problemy Peredachi Informatsii 2(4): 13–25.
104. Jelinek F & Anderson JB (1971) Instrumental tree encoding of information sources. IEEE Transactions on Information Theory 17(1): 118 – 119.
105. Baro S, Hagenauer J & Witzke M (2003) Iterative detection of MIMO transmission using a list-sequential (LISS) detector. In: Proc. IEEE Int. Conf. Commun., volume 4, pp. 2653–2657 vol.4.
106. Xu W, Wang Y, Zhou Z & Wang J (2004) A computationally efficient exact ML sphere decoder. In: Proc. IEEE Global Telecommun. Conf., volume 4, pp. 2594–2598.
107. Hagenauer J & Kuhn C (2007) The List-Sequential (LISS) algorithm and its application. IEEE Transactions on Communications 55(5): 918 –928.
108. Yee M (2005) Max-log-MAP sphere decoder. In: In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05), volume 3, pp. iii/1013–iii/1016 Vol. 3.
109. Marsch P, E Z & G F (2005) Smart candidate adding: A new low-complexity approach towards near-capacity MIMO detection. In: In Proceedigs of the European Signal Processing Conference (EUSIPCO'05), volume 3. Antalya, Turkey.
110. Jalden J & Ottersten B (2005) Parallel Implementation of a Soft Output Sphere Decoder. In: In Proceedings of the Asilomar Conference on Signals, Systems and Computers, pp. 581–585.
111. Studer C, Burg A & Bolcskei H (2008) Soft-output sphere decoding: algorithms and VLSI implementation. IEEE Journal on Selected Areas in Communications 26(2): 290 – 300.

112. Milliner D, Zimmermann E, Barry J & Fettweis G (2009) A Fixed-Complexity Smart Candidate Adding Algorithm for Soft-Output MIMO Detection. IEEE Journal of Selected Topics in Signal Processing 3(6): 1016–1025.
113. Studer C & Bolcskei H (2010) Soft-Input Soft-Output Single Tree-Search Sphere Decoding. IEEE Transactions on Information Theory 56(10): 4827–4842.
114. Widdup B, Woodward G & Knagge G (2004) A highly-parallel VLSI architecture for a list sphere detector. In: Proc. IEEE Int. Conf. Commun., pp. 2720–2725. Paris, France.
115. Kim HS, Seo SH & Park SC (2006) Parallelized VLSI Architecture of Single Stack based List Sphere Decoder. In: In Proceedings of the International Conference on Signal Processing, volume 1.
116. Garrett D, Davis L, ten Brink S, Hochwald B & Knagge G (2004) Silicon complexity for maximum likelihood MIMO detection using spherical decoding. IEEE Journal of Solid-State Circuits 39(9): 1544–1552.
117. Burg A, Borgmann M, Wenk M, Zellweger M, Fichtner W & Bölcskei H (2005) VLSI implementation of MIMO detection using the sphere decoding algorithm. IEEE Journal of Solid-State Circuits 40(7): 1566–1577.
118. Witte E, Borlenghi F, Ascheid G, Leupers R & Meyr H (2010) A Scalable VLSI Architecture for Soft-Input Soft-Output Single Tree-Search Sphere Decoding. IEEE Transactions on Circuits and Systems II: Express Briefs 57(9): 706 –710.
119. Chen S, Zhang T & Xin Y (2007) Relaxed K-Best MIMO Signal Detector Design and VLSI Implementation. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 15(3): 328–337.
120. Radosavljevic P, Guo Y & Cavallaro J (2009) Probabilistically bounded soft sphere detection for MIMO-OFDM receivers: algorithm and system architecture. IEEE Journal on Selected Areas in Communications 27(8): 1318 –1330.
121. Mondal S, Eltawil A & Salama K (2009) Architectural Optimizations for Low-Power K-Best MIMO Decoders. IEEE Transactions on Vehicular Technology 58(7): 3145–3153.
122. Mondal S, Eltawil A, Shen CA & Salama K (2010) Design and Implementation of a Sort-Free K-Best Sphere Decoder. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 18(10): 1497–1501.
123. Kim TH & Park IC (2010) High-Throughput and Area-Efficient MIMO Symbol Detection Based on Modified Dijkstra's Search. IEEE Transactions on Circuits and Systems I: Regular Papers 57(7): 1756 –1766.
124. Ma WK, Davidson T, Wong KM, Luo ZQ & Ching PC (2002) Quasi-maximum-likelihood multiuser detection using semi-definite relaxation with application to synchronous CDMA. IEEE Transactions on Signal Processing 50(4): 912–922.
125. Wiesel A, Eldar Y & Shitz S (2005) Semidefinite Relaxation for Detection of 16-QAM Signaling in MIMO channels. IEEE Signal Processing Letters 12(9): 653–656.
126. Sidiropoulos N & Luo ZQ (2006) A Semidefinite Relaxation Approach to MIMO Detection for High-Order QAM Constellations. IEEE Signal Processing Letters 13(9): 525–528.
127. Jalden J & Ottersten B (2008) The Diversity Order of the Semidefinite Relaxation Detector. IEEE Transactions on Information Theory 54(4): 1406–1422.
128. Guo D & Wang X (2003) Blind detection in MIMO systems via sequential Monte Carlo. IEEE Journal on Selected Areas in Communications 21(3): 464 – 473.
129. Henriksen S, Ninness B & Weller S (2008) Convergence of Markov-Chain Monte-Carlo Approaches to Multiuser and MIMO detection. IEEE Journal on Selected Areas in Com-

munications 26(3): 497 –505.

130. Chen RR, Peng R, Ashikhmin A & Farhang-Boroujeny B (2010) Approaching MIMO capacity using bitwise Markov Chain Monte Carlo detection. IEEE Transactions on Communications 58(2): 423 –428.
131. Laraway S & Farhang-Boroujeny B (2009) Implementation of a Markov Chain Monte Carlo Based Multiuser/MIMO Detector. IEEE Transactions on Circuits and Systems I: Regular Papers 56(1): 246 –255.
132. Siti M & Fitz M (2005) Layered Orthogonal Lattice Detector for Two Transmit Antenna Communications. In: In Proceedings of the Allerton Conference On Communication, Control, And Computing.
133. Siti M & Fitz M (2006) A Novel Soft-Output Layered Orthogonal Lattice Detector for Multiple Antenna Communications. In: In Proceedings of the IEEE International Conference on Communications (ICC'06), volume 4, pp. 1686–1691.
134. Tomasoni A, Ferrari M, Bellini S, Siti M & Cupaiuolo T (2010) A Hardware Oriented, Low-Complexity LORD MIMO Detector. In: In Proceedings of the IEEE International Conference on Communications (ICC'10), pp. 1–5.
135. Tomasoni A, Siti M, Ferrari M & Bellini S (2007) Turbo-LORD: A MAP-Approaching Soft-Input Soft-Output Detector for Iterative MIMO Receivers. In: In Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'07), pp. 3504–3508.
136. Tomasoni A, Siti M, Ferrari M & Bellini S (2009) A K-Best Version of the Turbo-LORD MIMO Detector in Realistic Settings. In: In Proceedings of the IEEE International Conference on Communications (ICC'09), pp. 1–5.
137. Sun Y & Cavallaro J (2008) A new MIMO detector architecture based on a Forward-Backward trellis algorithm. In: In Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers, pp. 1892–1896.
138. Myllylä M, Juntti M & Cavallaro J (2010) Implementation aspects of list sphere decoder algorithms for MIMO-OFDM systems. Elsevier Journal on Signal Processing 90(10): 2863–2876.
139. Myllylä M, Cavallaro JR & Juntti M (2011) Architecture Design and Implementation of the Metric First List Sphere Detector Algorithm. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, In press.
140. Myllylä M, Hintikka JM, Limingoja M, Byman A, Cavallaro J & Juntti M (2005) Complexity Analysis of MMSE Detector Architectures for MIMO OFDM Systems. In: In Proceedings of the Asilomar Conference on Signals, Systems, and Computers, pp. 75–81. Pacific Grove, CA, USA.
141. Myllylä M, Silvola P, Cavallaro J & Juntti M (2006) Comparison of Two Novel List Sphere Detector Algorithms for MIMO-OFDM Systems. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 1–5. Helsinki, Finland.
142. Myllylä M, Juntti M, Limingoja M, Byman A & Cavallaro JR (2006) Performance evaluation of two LMMSE detectors in a MIMO-OFDM hardware testbed. In: Proc. Annual Asilomar Conf. Signals, Syst., Comp., pp. 1161–1165. Pacific Grove, USA.
143. Myllylä M, Cavallaro J & Juntti M (2007) A List Sphere Detector based on Dijkstra's Algorithm for MIMO-OFDM Systems. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 1 – 5. Athens, Greece.
144. Myllylä M, Juntti M & Cavallaro J (2007) Implementation Aspects of List Sphere Detector Algorithms. In: Proc. IEEE Global Telecommun. Conf., pp. 3915–3920. Washington, D.C.,

USA.

145. Myllylä M, Antikainen J, Cavallaro J & Juntti M (2007) The effect of LLR clipping to the complexity of list sphere detector algorithms. In: In Proceedings of the Asilomar Conference on Signals, Systems and Computers, pp. 1559 – 1563. Pacific Grove, CA, USA.
146. Myllylä M, Juntti M & Cavallaro J (2008) The Effect of Preprocessing to the Complexity of List Sphere Detector Algorithms. In: Proc. Int. Symp. Wireless Pers. Multimedia Commun., pp. 1–5. Saariselkä, Finland.
147. Myllylä M, Juntti M & Cavallaro J (2008) Implementation and complexity analysis of list sphere detector for MIMO-OFDM systems. In: In Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers, pp. 1852–1856. Pacific Grove, CA, USA.
148. Myllylä M, Juntti M & Cavallaro J (2009) Architecture design and implementation of the Increasing Radius - List Sphere Detector algorithm. In: In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'09), pp. 553–556. Taipei, Taiwan.
149. Kerttula J, Myllylä M & Juntti M (2007) Implementation of a K-best based MIMO-OFDM detector algorithm. In: Proc. European Sign. Proc. Conf., pp. 2149–2153. Poznań, Poland.
150. Kerttula J, Myllylä M & Juntti M (2007) Word length and architecture design of a K-best based MIMO-OFDM detector algorithm. In: Proc. IEEE Works. on Sign. Proc. Adv. in Wirel. Comms., pp. 1–5. Helsinki, Finland.
151. Kunnari E & Iinatti J (2007) Stochastic modelling of Rice fading channels with temporal, spatial and spectral correlation. IET Communications 1(2): 215–224.
152. Salo J, Del Galdo G, Salmi J, Kyösti P, Hentilä L, Milojevic M, Laselva D, Zetterberg P & Schneider C (2005). MATLAB Implementation of the WINNER Phase I Channel Model. [Online]. Https://www.ist-winner.org/phase_model.html.
153. Baum D, El-Sallabi H & et al (Oct, 2006) D5.4, Final Report on Link and System Level Channel Models, ver 1.4. Technical report, IST-WINNER. [Available] https://www.ist-winner.org/DeliverableDocuments/D5.4.pdf.
154. Hagunauer J, Offer E & Papke L (1996) Iterative Decoding of Binary Block and Convolutional Codes. IEEE Transactions on Information Theory 42(2).
155. Hagenauer J, Robertson P & Papke L (1994) Iterative turbo decoding of systematic convolutional codes with the MAP and SOVA algorithms. In: ITG Symp. Source and Channel Coding, pp. pp. 21–29.
156. Kay SM (1993) Fundamentals of Statistical Signal Processing: Estimation Theory. Prentice-Hall, Englewood Cliffs, NJ, USA.
157. El-Amawy A & Dharmarajan KR (1989) Parallel VLSI algorithm for stable inversion of dense matrices. IEE Proceedings on Computers and Digital Techniques 136(6): 575–580.
158. Hu YH (1992) CORDIC-based VLSI architechtucres for digital signal processing. IEEE Signal Processing Magazine 9(3): 16–35.
159. Myllylä M, Vehkaperä M & Juntti M (2005) Complexity Evaluation of MMSE Based Detector for LST Architectures. In: In Proceedings of the IEEE International Workshop on Convergent Technologies (IWCT'05). Oulu, Finland.
160. Vehkaperä M, Tujkovic D, Li Z & Juntti M (2005) Receiver design for spatially layered downlink MC-CDMA system. IEEE Transactions on Vehicular Technology 54(3): 1042–1055.
161. Wong KK & Paulraj A (2004) On the decoding order of MIMO maximum-likelihood sphere decoder: linear and non-linear receivers. In: In Proceedings of the IEEE 59th Vehicular

Technology Conference (VTC'04), volume 2, pp. 698–702 Vol.2.

162. Dai Y, Sun S & Lei Z (11-14 Sept. 2005) A comparative study of QRD-M detection and sphere decoding for MIMO-OFDM systems. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 186–190. Berlin, Germany.
163. Wübben D, Böhnke R, Kühn V & Kammeyer K (2003) MMSE extension of V-BLAST based on sorted QR decomposition. In: Proc. IEEE Veh. Technol. Conf., volume 1, pp. 508–512. Orlando, Florida.
164. 3rd Generation Partnership Project (3GPP) (2005) TSGR1#41 R1-050-520, EUTRA downlink numerology. Technical report, 3rd Generation Partnership Project (3GPP).
165. Zimmermann E, Milliner D, Barry J & Fettweis G (2008) Optimal LLR clipping levels for mixed hard/soft output detection. In: Proc. IEEE Global Telecommun. Conf., pp. 1–5.
166. Milliner D, Zimmermann E, Barry J & Fettweis G (2008) Channel state information based LLR clipping in list MIMO detection. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 1–5.
167. Dick C, Amiri K, Cavallaro J & Rao R (26-29 Oct. 2008) Design and architecture of spatial multiplexing MIMO decoders for FPGAs. In: In Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers, pp. 160–164. Pacific Grove, CA, USA.
168. Seo S & Park S (2006) Efficient VLSI implementation of the list sphere decoder with real-value based tree searching method. In: In Proceedings of the 8th International Conference Advanced Communication Technology (ICACT'06), volume 3, pp. 4 pp.–1697.
169. Antikainen J, Salmela P, Silvén O, Juntti M, Takala J & Myllylä M (2007) Application-specific instruction set processor implementation of list sphere detector. EURASIP Journal on Embedded Systems 2007.
170. Burg A, Borgmanr M, Wenk M, Studer C & Bolcskei H (2006) Advanced receiver algorithms for MIMO wireless communications. In: In Proceedings of the Design, Automation and Test in Europe (DATE'06), volume 1, pp. 6 pp.–.
171. Cormen TH, Leiserson CE, Rivest RL & Stein C (2001) Introduction to Algorithms. MIT Press.
172. Burg A, Borgmann M, Wenk M, Zellweger M, Fichtner W & Bölcskei H (2005) VLSI Implementation of MIMO Detection Using the Sphere Decoding Algorithm. IEEE Journal of Solid-State Circuits 40(7): 1566–1577.
173. Seol C & Cheun K (2008) A low complexity euclidean norm approximation. IEEE Transactions on Signal Processing 56(4): 1721–1726.
174. ten Brink S (2001) Convergence behavior of iteratively decoded parallel concatenated codes. IEEE Transactions on Communications 49(6): 1727–1737.
175. Smith DR (1997) Channel models. In: Gibson JD (ed.) The Communications Handbook, chapter 11, pp. 131–140. CRC Press.
176. Lightbody G, Walke R, Woods R & McCanny J (2000) Linear QR Architecture for a Single Chip Adaptive Beamformer. Journal of VLSI Signal Processing Systems 24(1): 67–81.
177. Edman F & Öwall V (7-10 Sep. 2003) Implementation of a scalable matrix inversion architecture for triangular matrices. In: Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun., pp. 2558–2562.
178. Walke RL, Smith RWM & Lightbody G (1999) Architectures for adaptive weight calculation on ASIC and FPGA. In: In Proceedings of the 33rd Asilomar Conference on Signals, Systems, and Computers, volume 2, pp. 1375–1380.

179. Bengough P & Simmons S (1995) Sorting-based VLSI architectures for the M-algorithm and T-algorithm trellis decoders. IEEE Transactions on Communications 43(234): 514 –522.
180. Datasheet (2010) System Generator for DSP. Technical report, Xilinx. [Available] http://www.xilinx.com/tools/sysgen.htm.
181. Xilinx (2007). Xilinx IP core library. [Online] http://www.xilinx.com/ipcenter/.
182. 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network (2006) Physical layer aspects for evolved UTRA (TR 25.814 version 1.5.0 (release 7)). Technical report, 3rd Generation Partnership Project (3GPP).
183. Rappaport TS (2002) Wireless Communications: Principles and Practice. Prentice Hall, New Jersey, USA, 2nd edition.
184. Datasheet (2008) Catapult Synthesis. Technical report, Mentor Graphics. [Available] http://www.mentor.com/products/esl/high_level_synthesis/ catapult_synthesis/index.cfm.
185. Proakis JG & Manolakis DG (1989) Introduction to Digital Signal Processing: Principles, Algorithms and Applications. Macmillan, New York, USA, 2nd edition.
186. Lyons R (1997) Understanding Digital Signal Processing. Addison-Wesley Longman, Massachusetts, USA, 2nd edition.
187. Ullman J (1994) Computational aspects of VLSI. Computer science press, Inc.
188. Bajwa RS, Owens R & Irwin M (1994) Area time trade-offs in micro-grain VLSI array architectures. IEEE Transactions on Computers 43(10): 1121 – 1128.
189. Hulkkonen A (2004). SDR tools for development of 4G systems. Prosessori Magazine. [Online] http://www.prosessori.fi/es04 /ARKISTO/PDF/OHJELMISTORADIOTEKNIIKKA.PDF.
190. Elektrobit (2007) EB Products and Services: EB Propsim C8 multi-channel emulator. Technical report, Elektrobit Corporation. [Online] http://www.elektrobit.com/index.php?207.
191. Miao H & Juntti M (Nov., 2005) Space-time channel estimation and performance analysis for wireless MIMO-OFDM systems with spatial correlation. IEEE Transactions on Vehicular Technology 54(6): 2003–2016.
192. Byman A, Meinilä J, Rautio M, Roivainen J, Korpi J, Takalo-Mattila J, Skowron M, Mihovska A, Bouttier A, Clermidy F, Rudant L & Mayrargue S (2009) D5.1: WINNER+ Interim Trials Results. Technical report, CELTIC WINNER+ project. [Available] http://projects.celtic-initiative.org/winner+/WINNER+%20Deliverables/D5.1_v1.0.pdf.

ACTA UNIVERSITATIS OULUENSIS

SERIES C TECHNICA

364. Väänänen, Mirja (2010) Communication in high technology product development projects : project personnel's viewpoint for improvement

365. Korhonen, Esa (2010) On-chip testing of A/D and D/A converters : static linearity testing without statistically known stimulus

366. Palukuru, Vamsi Krishna (2010) Electrically tunable microwave devices using BST-LTCC thick films

367. Saarenpää, Ensio (2010) Rakentamisen hyvä laatu : rakentamisen hyvän laadun toteutuminen Suomen rakentamismääräyksissä

368. Vartiainen, Johanna (2010) Concentrated signal extraction using consecutive mean excision algorithms

369. Nousiainen, Olli (2010) Characterization of second-level lead-free BGA interconnections in thermomechanically loaded LTCC/PWB assemblies

370. Taskila, Sanna (2010) Improved enrichment cultivation of selected food-contaminating bacteria

371. Haapala, Antti (2010) Paper machine white water treatment in channel flow : integration of passive deaeration and selective flotation

372. Plekh, Maxim (2010) Ferroelectric performance for nanometer scaled devices

373. Lee, Young-Dong (2010) Wireless vital signs monitoring system for ubiquitous healthcare with practical tests and reliability analysis

374. Sillanpää, Ilkka (2010) Supply chain performance measurement in the manufacturing industry : a single case study research to develop a supply chain performance measurement framework

375. Marttila, Hannu (2010) Managing erosion, sediment transport and water quality in drained peatland catchments

376. Honkanen, Seppo (2011) Tekniikan ylioppilaiden valmistumiseen johtavien opintopolkujen mallintaminen — perusteena lukiossa ja opiskelun alkuvaiheessa saavutettu opintomenestys

377. Malinen, Ilkka (2010) Improving the robustness with modified bounded homotopies and problem-tailored solving procedures

378. Yang, Dayou (2011) Optimisation of product change process and demand-supply chain in high tech environment

379. Kalliokoski, Juha (2011) Models of filtration curve as a part of pulp drainage analyzers

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/

UNIVERSITY OF OULU P.O.B. 7500 FI-90014 UNIVERSITY OF OULU FINLAND

ACTA UNIVERSITATIS OULUENSIS

ISBN 978-951-42-9432-7 (Paperback)
ISBN 978-951-42-9433-4 (PDF)
ISSN 0355-3213 (Print)
ISSN 1796-2226 (Online)

UNIVERSITY of OULU
OULUN YLIOPISTO