

# **Design of Robust 2D Barcodes for Industrial Environments**

Thesis designated to obtain  
the scientific title of doctor engineer  
of the  
University Politehnica of Timișoara  
in Electronics and Telecommunications  
by

**M.Eng. Dipl.-Ing.(FH) Wolfgang Proß**

Supervisors:                    prof.univ.dr.ing. Marius Oteșteanu  
    prof.univ.dr.ing. Franz Quint  
Scientific committee:        prof.univ.dr.ing. Claude Berrou  
    prof.univ.dr.ing. Monica Borda  
    prof.univ.dr.ing. Ioan Narforniță

Date of the defence of the doctor's thesis: 16.11.2012

Seriile Teze de doctorat ale UPT sunt:

- |   |  |
|---|--|
| 1. Automatică                               | 8. Inginerie Industrială                   |
| 2. Chimie                                   | 9. Inginerie Mecanică                      |
| 3. Energetică                               | 10. Știința Calculatoarelor                |
| 4. Inginerie Chimică                        | 11. Știința și Ingineria Materialelor      |
| 5. Inginerie Civilă                         | 12. Ingineria sistemelor                   |
| 6. Inginerie Electrică                      | 13. Inginerie energetică                   |
| 7. Inginerie Electronică și Telecomunicații | 14. Calculatoare și tehnologia informației |

Universitatea "Politehnica" din Timișoara a inițiat seriile de mai sus în scopul diseminării expertizei, cunoștințelor și rezultatelor cercetărilor întreprinse în cadrul școlii doctorale a universității. Seriile conțin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susținute în universitate începând cu 1 octombrie 2006.

Copyright © Editura Politehnica - Timișoara, 2012

Această publicație este supusă prevederilor legii dreptului de autor. Multiplicarea acestei publicații, în mod integral sau în parte, traducerea, tipărirea, reutilizarea ilustrațiilor, expunerea, radiodifuzarea, reproducerea pe microfilme sau în orice altă formă este permisă numai cu respectarea prevederilor Legii române a dreptului de autor în vigoare și permisiunea pentru utilizare obținută în scris din partea Universității "Politehnica" din Timișoara. Toate încălcările acestor drepturi vor fi penalizate potrivit Legii române a drepturilor de autor.

România, 300159 Timișoara, Bd. Republicii 9,  
tel. 0256 403823, fax. 0256 403221  
e-mail: editura@edipol.upt.ro

## Foreword

The basis for this thesis is a cooperation between the University of Applied Sciences, Karlsruhe and the University "Politehnica" of Timișoara. Enrolled as an external Ph.D student at the University "Politehnica" of Timișoara, the main research activity took place at the University of Applied Sciences in Karlsruhe.

First of all, I wish to thank my doctoral advisor at the University "Politehnica" of Timișoara, Prof. Dr. Ing. Marius Oteșteanu. Without his constant support, advices, and comments this thesis would not have been possible.

I consider myself privileged to have been supervised by Prof. Dr.-Ing. Franz Quint during my research at the University of Applied Sciences, Karlsruhe. I am deeply indebted to him for his guidance, for many fruitful discussions and especially for always being generous with his time. I have greatly benefited from his long experience and extensive knowledge.

I wish to thank Dr.-Ing. Holger Jäkel for his time and assistance. Furthermore, I would like to thank Prof. Dr. Ing. Ioan Nafornița, Prof. Dr. Ing. Miranda Nafornița, Prof. Dr. Ing. Vasile Gui and all members of the Electronics and Telecommunications department of the Politehnica that helped me with constructive criticism and useful suggestions. Many Thanks to all of "my" students for contributing to this work.

It is a pleasure to thank Dr. Ing. Ion-Cosmin Dița for all his help and support. I am very pleased that the collaboration between our supervisors eventually led to our friendship.

Special thanks to Tonio Petrignani for proofreading this thesis. Furthermore, I would like to thank Andreas Bluthardt for his patience and understanding while my energy was down. Thanks to all my friends and to family Miller for relaxing and enjoyable times and their interest and support.

Finally, and above all, I owe my most sincere thanks to my family: my father and mother, my sisters Arnhild and Berlinde, my brother Guntram and especially "meine Ilana" for their continuous love, everlasting support and encouragements. This thesis is dedicated to them.

Timișoara, November 2012

Wolfgang Proß

Proß, Wolfgang

**Design of Robust 2D Barcodes for Industrial Environments**

PhD-Thesis of the UPT, Series 7, Nr. 49, Publisher Politehnică, 2012, 154 pages, 78 figures, 27 tables.

ISSN: 1842-7014

ISBN: 978-606-554-550-2

**Keywords:**

LDPC codes, LDPC code Design, Markov-modulated channel, Estimation-decoding, 2D barcodes, Data Matrix code, 2D hidden Markov model, ED2D decoder, LDPC-based 2D barcode

**Abstract:**

This thesis addresses the decoding of two dimensional (2D) barcodes in industrial environments. A new class of 2D barcodes is introduced that is based on low-density parity-check (LDPC) codes. The special requirements in direct part mark identification (DPMI) applications have been considered during the development in order to design a robust 2D barcode. The major accomplishments of this work include the design and the decoding of both codes, the LDPC code as well as the developed LDPC-based 2D barcode. A design method for short irregular LDPC codes is introduced that yields a better decoding performance for the additive white Gaussian noise (AWGN) channel and the Markov-modulated Gaussian channel (MMGC) compared to optimization methods known from literature. Estimation-decoding and a reestimation procedure of a 2-state channel-model's transition probabilities have been extended for usage with a MMGC. For the application of LDPC codes on 2D barcodes an intelligent interleaver has been developed to place the code word's symbols in the available grid of the 2D barcode in order to increase the error-correction capabilities of the resulting 2D barcode. In addition, a 2-state channel-model has been created that provides a good description of 2D barcodes in industrial environments. Considering the 2-dimensional arrangement of the 2D barcode's modules, the estimation-decoding algorithm has been extended to operate based on a 2D hidden Markov model (HMM). This provided the estimation-decoding in 2 dimensions (ED2D) algorithm that is proposed for the decoding of LDPC-based 2D barcodes. Next to the utilization of regular LDPC codes for the design of the new 2D barcode, irregular LDPC codes have been applied. The latter have been designed by means of the introduced optimization method and the channel-model for 2D barcodes. For the evaluation of the new class of 2D barcodes, a test-environment and an appropriate test-procedure have been created. The test-environment enables one to compare different variants of 2D barcodes in a simulated industrial environment under fair conditions. This test-environment proves that the developed LDPC-based 2D barcode decoded with the ED2D algorithm has a substantially better error-correction performance compared to the standard Reed-Solomon (RS)-based Data Matrix code (DMC).

# Contents

<b>List of Figures</b>	<b>8</b>
<b>List of Tables</b>	<b>11</b>
<b>Acronyms</b>	<b>13</b>
<b>Symbols</b>	<b>15</b>
<b>1 Introduction</b>	<b>19</b>
1.1 Some history . . . . .	19
1.2 Motivation . . . . .	22
1.3 Thesis outline . . . . .	24
1.4 Publications . . . . .	27
<b>2 Barcodes</b>	<b>29</b>
2.1 1D barcodes . . . . .	29
2.2 Stacked barcodes . . . . .	30
2.3 2D barcodes . . . . .	30
2.3.1 Data Matrix code (ECC200) . . . . .	31
2.4 Direct part mark identification . . . . .	32
<b>3 Channel-models</b>	<b>35</b>
3.1 Binary symmetric channel . . . . .	36
3.2 Z channel . . . . .	37
3.3 Additive white Gaussian noise channel . . . . .	37
3.4 Markov-modulated channels . . . . .	38
3.4.1 Gilbert-Elliott channel . . . . .	39
3.4.2 Markov-modulated Gaussian channel . . . . .	39
3.5 Log-likelihood ratio . . . . .	40
3.6 Soft-decision and hard-decision . . . . .	41
<b>4 Low-density parity-check codes</b>	<b>43</b>
4.1 Representation of LDPC codes . . . . .	43
4.2 Ensembles of LDPC codes . . . . .	46
4.3 Encoding of LDPC codes . . . . .	46
4.4 Decoding of LDPC codes . . . . .	47
4.4.1 Log-domain BP decoder . . . . .	47
4.4.2 MS decoder . . . . .	50
4.4.3 MSc decoder . . . . .	51
4.5 Design and construction of LDPC codes . . . . .	51
4.5.1 Cycles and girth . . . . .	51

4.5.2	Computer-based design, finite-geometry codes and codes based on finite fields . . . . .	52
4.5.3	Density evolution . . . . .	53
4.5.4	Progressive-edge-growth (PEG) algorithm . . . . .	53
4.6	Evaluation of LDPC codes . . . . .	56
<b>5</b>	<b>Design of short irregular LDPC codes</b>	<b>59</b>
5.1	Downhill simplex based design . . . . .	60
5.1.1	Constraints . . . . .	61
5.1.2	Function evaluations . . . . .	63
5.1.3	Optimization process . . . . .	63
5.1.4	Initializing simplex . . . . .	64
5.2	Design results . . . . .	66
<b>6</b>	<b>Estimation-decoding</b>	<b>69</b>
6.1	Messages of the LDPC subgraph . . . . .	70
6.2	Messages of the Markov subgraph . . . . .	70
6.3	The interface messages . . . . .	71
6.4	New variant of estimation-decoding . . . . .	71
6.4.1	Reestimation of the transition probabilities . . . . .	72
6.4.2	Variance estimation for the decoding on a 2-state MMGC . . . . .	72
6.5	Analysis and Results . . . . .	73
6.5.1	Effectiveness of the state-estimation . . . . .	74
6.5.2	Effectiveness of the new variant . . . . .	75
6.5.3	Speed considerations . . . . .	76
6.5.4	Estimation-decoding with irregular LDPC codes . . . . .	79
<b>7</b>	<b>LDPC-based 2D barcodes</b>	<b>81</b>
7.1	Symbol-placement . . . . .	81
7.1.1	Distance measurements . . . . .	82
7.1.1.1	Geometrical distance . . . . .	82
7.1.1.2	Tree distance . . . . .	82
7.1.2	Cost of a symbol-placement . . . . .	83
7.1.3	Optimization of the symbol-placement . . . . .	84
7.1.4	Optimization results . . . . .	85
7.2	Channel-model for 2D barcodes . . . . .	87
7.2.1	Acquisition and image processing . . . . .	87
7.2.2	Channel-model in absence of damages . . . . .	89
7.2.3	Channel-model for damaged 2D barcodes . . . . .	92
7.2.4	Computation of soft-decisions for 2D barcodes . . . . .	95
7.3	Decoder design . . . . .	98
7.3.1	Messages of the LDPC subgraph . . . . .	100
7.3.2	Messages of the Markov subgraph . . . . .	100
7.3.3	The interface messages . . . . .	101
7.3.4	Reestimation of the transition probabilities . . . . .	102
7.4	Design of irregular LDPC codes for 2D barcodes . . . . .	103
7.4.1	Application of the 2D barcode's channel-model . . . . .	103
7.4.2	Soft-decisions and decoding . . . . .	105
7.4.3	Design results . . . . .	105
<b>8</b>	<b>Evaluation</b>	<b>107</b>
8.1	Test environment . . . . .	107

---

8.1.1	2D barcode encoders . . . . .	109
8.1.2	Simulations . . . . .	109
8.1.3	Image processing . . . . .	110
8.1.4	Soft-decisions and hard-decisions . . . . .	110
8.1.5	2D barcode decoders . . . . .	112
8.1.6	Comparison . . . . .	112
8.2	Test procedure . . . . .	113
8.3	The effectiveness of the optimized symbol-placement . . . . .	114
8.4	The effectiveness of the ED2D decoding . . . . .	117
8.5	Irregular versus regular . . . . .	121
8.6	LDPC-based 2D barcode versus Data Matrix code . . . . .	124
<b>9</b>	<b>Conclusion and outlook</b>	<b>133</b>
9.1	Summary of contributions . . . . .	133
9.2	Proposals for future work . . . . .	135
<b>A</b>	<b>Symbol-placement sets</b>	<b>137</b>
	<b>Bibliography</b>	<b>147</b>

## List of Figures

1.1	Barcodes proposed by Woodland and Silver . . . . .	20
1.2	KarTrak barcode . . . . .	20
1.3	UPC and EAN-13 barcode . . . . .	21
1.4	Code 39 . . . . .	21
1.5	Data Matrix code and QR code . . . . .	22
1.6	DMC on paper and plastic . . . . .	22
1.7	Main original contributions . . . . .	26
2.1	Code 128 . . . . .	30
2.2	Examples for stacked barcodes . . . . .	30
2.3	Aztec code and MaxiCode . . . . .	31
2.4	Composition of a DMC . . . . .	32
2.5	Examples for DPMI applications . . . . .	33
3.1	Principle of a digital communication system . . . . .	35
3.2	Binary symmetric channel . . . . .	36
3.3	Z-channel . . . . .	37
3.4	Additive white Gaussian noise channel . . . . .	37
3.5	Likelihoods for an AWGN channel . . . . .	38
3.6	2-state Markov-modulated channel . . . . .	38
3.7	Gilbert-Elliott channel . . . . .	39
3.8	2-state Markov-modulated Gaussian channel . . . . .	39
3.9	Likelihoods for a MMGC . . . . .	40
4.1	PCM and Tanner graph example . . . . .	45
4.2	$\phi$ -function . . . . .	48
4.3	Example for a check-node update . . . . .	49
4.4	Example for a symbol-node update . . . . .	49
4.5	Creation of a PEG-tree . . . . .	55
4.6	Evaluation of a LDPC code . . . . .	56
5.2	Example of a downhill simplex based design . . . . .	63
5.1	The downhill simplex algorithm . . . . .	65
5.3	BER comparison: method of Hu et al. versus new design method . . . . .	66
5.4	WER comparison: method of Hu et al. versus new design method . . . . .	67
6.1	Markov-LDPC factor graph . . . . .	69
6.2	Messages in one sector of the Markov-LDPC factor graph . . . . .	70
6.3	One sector of the Markov subgraph . . . . .	71
6.4	Range of values used for the variance estimation . . . . .	73



6.5	Effectiveness of the state-estimation . . . . .	74
6.6	Effectiveness of the new estimation-decoding variant . . . . .	75
6.7	Reestimation of the transition probabilities . . . . .	76
6.8	Decoding speed comparison . . . . .	77
6.9	Extra computational cost for the state-estimation and the reestimation procedure . . . . .	77
6.10	Decoding performance for different points in time considering four decoding variants and $E_b/N_0 = 3.5$ dB . . . . .	78
6.11	Decoding performance for different points in time considering four decoding variants and $E_b/N_0 = 6$ dB . . . . .	79
6.12	Estimation-decoding with irregular LDPC codes . . . . .	80
7.1	Effect of local disturbances for different symbol-placements . . . . .	83
7.2	Cost-function . . . . .	84
7.3	The symbol-placement optimization . . . . .	85
7.4	Old and new costs of 15 symbol-placement optimization rounds . . . . .	86
7.5	Evolution of the costs for 15 independent symbol-placement optimization processes . . . . .	86
7.6	From the marked item to the soft-decisions . . . . .	87
7.7	The camera-based system for the acquisition . . . . .	87
7.8	Bright field and dark field capture of a 2D barcode . . . . .	88
7.9	Marking of modules that are affected by the damage simulation . . . . .	93
7.10	2-state Markov-modulated channel-model for the bright field case . . . . .	96
7.11	2-state Markov-modulated channel-model for the dark field case . . . . .	96
7.13	ED2D graph . . . . .	99
7.12	2D-hidden Markov model . . . . .	99
7.14	Messages in one sector of the ED2D graph . . . . .	100
7.15	WER-computation by means of a histogram-based channel-model . . . . .	103
7.16	2-state histogram-based channel-model for 2D barcodes . . . . .	104
7.17	WER comparison: regular versus irregular LDPC code for 2D barcodes . . . . .	106
8.1	Test environment . . . . .	108
8.2	Effectiveness of the intelligent interleaving in terms of successful decodings depending on the number of pre-decoding bit errors . . . . .	116
8.3	Gain in decoding successes that is obtained by using the optimized symbol-placement . . . . .	116
8.4	Number of decodings of the ED2D decoder and the MSc decoder . . . . .	118
8.5	Effectiveness of the ED2D algorithm in terms of successful decodings depending on the number of pre-decoding bit errors . . . . .	119
8.6	Gain obtained when decoding a LDPC-based 2D barcode with the ED2D decoder instead of the MSc decoder . . . . .	120
8.7	Decoding speed of the ED2D decoder and the MSc decoder . . . . .	120
8.8	Decoding successes plotted over the pre-decoding bit errors considering a 2D barcode based on a regular LDPC code and an irregular LDPC code, respectively . . . . .	122
8.9	Total decoding successes for several regular and irregular LDPC codes depending on the average check-node degree . . . . .	124
8.10	Number of decodings for a LDPC-based 2D barcode and a DMC, respectively . . . . .	126
8.11	Cumulative histograms for the number of decodings considering a LDPC-based 2D barcode and a DMC, respectively . . . . .	127

---

8.12	Decoding successes plotted over pre-decoding bit errors for a LDPC-based 2D barcode and a DMC, respectively . . . . .	128
8.13	Gain obtained when using the LDPC-based 2D barcode instead of the standard DMC . . . . .	128
8.14	Failed decodings for a LDPC-based 2D barcode and a DMC, respectively	129
8.15	Decoding speed for a LDPC-based 2D barcode and a DMC, respectively	129
8.16	Decoding successes plotted over pre-decoding bit errors for a LDPC-based 2D barcode and a DMC, respectively . . . . .	130
8.17	Gain obtained when using the LDPC-based 2D barcode with the MSc decoder instead of the standard DMC . . . . .	130
8.18	Decoding speed for a LDPC-based 2D barcode decoded with the MSc decoder and a DMC, respectively . . . . .	131

## List of Tables

7.1	Histograms of correlation coefficients considering bright field acquisitions	90
7.2	Histograms of correlation coefficients considering dark field acquisitions	91
7.3	Reference oil drops and water drops on different materials . . . . .	92
7.4	Examples for simulation of oil drops and water drops . . . . .	93
7.5	Histograms of correlation coefficients considering bright field acquisitions and damages . . . . .	94
7.6	Histograms of correlation coefficients considering dark field acquisitions and damages . . . . .	95
7.7	Gaussian parameters for the bright field case . . . . .	97
7.8	Gaussian parameters for the dark field case . . . . .	98
7.9	Gaussian parameters for the 2D barcode's channel-model . . . . .	105
7.10	Transition probabilities for the simulation with the 2D barcode's channel-model . . . . .	105
8.1	Examples for simulated pictures . . . . .	110
8.2	Examples for simulated pictures with added damage simulations . . . . .	111
8.3	Effectiveness of the symbol-placement optimization in terms of successful decodings . . . . .	115
8.4	Effectiveness of the ED2D algorithm in terms of successful decodings . . . . .	117
8.5	Decoding successes for a 2D barcode based on a regular LDPC code and an irregular LDPC code, respectively . . . . .	121
8.6	SNDDs of irregular LDPC codes designed for 2D barcodes . . . . .	123
8.7	Check-node degrees of irregular LDPC codes designed for 2D barcodes . . . . .	123
8.8	Decoding successes for a LDPC-based 2D barcode and a DMC, respectively . . . . .	125
8.9	Decoding speed of two decoders in Matlab and in C . . . . .	127
A.1	Optimized symbol-placement for a 2D barcode based on a regular LDPC code with $d_x = 2$ . . . . .	138
A.2	Optimized symbol-placement for a 2D barcode based on a regular LDPC code with $d_x = 3$ . . . . .	139
A.3	Degraded symbol-placement for a 2D barcode based on a regular LDPC code with $d_x = 3$ . . . . .	140
A.4	Optimized symbol-placement for a 2D barcode based on an irregular LDPC code designed with $d_x^{max} = 15$ . . . . .	141
A.5	Optimized symbol-placement for a 2D barcode based on an irregular LDPC code designed with $d_x^{max} = 10$ . . . . .	142
A.6	Optimized symbol-placement for a 2D barcode based on an irregular LDPC code designed with $d_x^{max} = 6$ . . . . .	143

---

A.7	Optimized symbol-placement for a 2D barcode based on an irregular LDPC code designed with $d_x^{max} = 4$ . . . . .	144
A.8	Optimized symbol-placement for a 2D barcode based on an irregular LDPC code designed with $d_x^{max} = 3$ . . . . .	145

# Acronyms

1D	one dimensional
2D	two dimensional
3D	three dimensional
ACE	approximate cycle EMD
AIAG	Automotive Industry Action Group
AIDC	Automatic identification and data capture
APP	a posteriori probability
ASCII	American Standard Code for Information Interchange
ATA	Air Transport Association
Auto-ID	Automatic identification
AWGN	additive white Gaussian noise
AZCW	all-zero code word
BCJR	Bahl-Cocke-Jelinek-Raviv
BER	bit error ratio
BF	bit-flipping
BP	belief propagation
BPSK	binary phase-shift keying
BSC	binary symmetric channel
CND	check-node degree
CNDD	check-node degree distribution
DHS	downhill simplex
DMC	Data Matrix code
DoD	United States Department of Defense
DPMI	direct part mark identification
DVB-S2	digital video broadcasting - satellite - second generation
EAN-13	13 digit European Article Number
ECC	error checking and correcting
ED2D	estimation-decoding in 2 dimensions
EDEP	estimation-decoding and estimation of the transition-probabilities
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport
EIA	Electronic Industry Association
EM	Expectation-Modification
EMD	extrinsic message degree
EXIT	extrinsic-information-transfer
FEC	forward error correction
FG	finite geometry
HD	hard-decision
HMM	hidden Markov model

---

IAQG	International Aerospace Quality Group
IBM	International Business Machines Corporation
IEC	International Electrotechnical Commission
IRA	irregular repeat-accumulate
ISO	International Organization for Standardization
LDPC	low-density parity-check
LLR	log-likelihood ratio
MAP	maximum a posteriori
ML	maximum likelihood
MMGC	Markov-modulated Gaussian channel
MP	message-passing
MPEG	moving picture experts group
MS	min-sum
ms	milliseconds
NAFC	National Association of Food Chains
NASA	National Aeronautics and Space Administration
OCR	optical character recognition
PCM	parity-check matrix
PDF	probability density function
PDF417	Portable Data File, 4 bars and spaces in 17 modules
PEG	progressive-edge-growth
QC	quasi-cyclic
QR code	Quick Response code
RAM	Random-Access-Memory
RC	row-column
RFID	radio frequency identification
RS	Reed-Solomon
SD	soft-decision
SND	symbol-node degree
SNDD	symbol-node degree distribution
SNR	signal-to-noise ratio
SP	sum-product
UPC	Universal Product Code
UPS	United Parcel Service
USA	United States of America
WER	word error ratio

# Symbols

$\bar{a}_0$	Mean of all zero-module-means only considering the zero-modules located in the broken border
$\bar{a}_1$	Mean of all one-module-means only considering the one-modules located in the broken border
$\bar{a}^{rc}$	Mean of all pixels that the module in row $r$ and column $c$ of a 2D barcode consists of
$a_{kl}^{rc}$	Pixel in row $k$ and column $l$ of a module that is located in row $r$ and column $c$ of a 2D barcode
$\alpha_i$	Forward-message of the Forward-Backward algorithm belonging to symbol $x_i$ considering estimation-decoding of LDPC codes
$\alpha_{ij}$	Sign of the message $L(q_{ij})$
$\alpha_{r,c}^h$	Forward-message of the Forward-Backward algorithm belonging to symbol $x_{r,c}$ and the horizontal Markov chain considering the ED2D algorithm
$\alpha_{r,c}^v$	Forward-message of the Forward-Backward algorithm belonging to symbol $x_{r,c}$ and the vertical Markov chain considering the ED2D algorithm
$\bar{b}$	Mean of all pixels that a reference module consists of
$\beta_i$	Backward-message of the Forward-Backward algorithm belonging to symbol $x_i$ considering estimation-decoding of LDPC codes
$\beta_{ij}$	Absolute value of the message $L(q_{ij})$
$\beta_{r,c}^h$	Backward-message of the Forward-Backward algorithm belonging to symbol $x_{r,c}$ and the horizontal Markov chain considering the ED2D algorithm
$\beta_{r,c}^v$	Backward-message of the Forward-Backward algorithm belonging to symbol $x_{r,c}$ and the vertical Markov chain considering the ED2D algorithm
$b_j^{max}$	Upper bound of an interval of which numbers are randomly taken during the initialization of a simplex $\Delta$
$b_{kl}$	Pixel in row $k$ and column $l$ of a reference module
$\mathcal{B}$	Basis of $\mathcal{C}$ consisting of $k$ linearly independent codewords $g$
$\mathcal{B}^\perp$	Basis of the dual code $\mathcal{C}^\perp$
$c_f$	Correction factor used for the MSc decoder
$\mathcal{C}_i$	Set of check-nodes connected to symbol-node $x_i$ in a LDPC code's Tanner graph
$c_j$	Check-node of a LDPC code's Tanner graph
$\mathcal{C}$	$(n, k)$ linear block code
$\mathcal{C}^\perp$	Dual code of $\mathcal{C}$
$c_{x_a}$	Column of a 2D barcode's data region in which the symbol $x_a$ is located
$d$	Degree of a symbol- or check-node $\rightarrow$ number of adjacent edges
$d_{c_j}$	Degree of check-node $c_j$ , i.e. the number of connected symbol-nodes

$d_c^{max}$	Maximum number of adjacent edges to a check-node
$d_E$	Euclidean distance
$d_\sigma$	Factor by which the standard deviation $\sigma_2$ of a Markov-modulated Gaussian channel's bad sub-channel is degraded compared to $\sigma_1$ that belongs to the AWGN of the good sub-channel
$d_s^{max}$	Maximum number of adjacent edges to a symbol-node
$d_T$	Tree distance measured in layers inside of a PEG-tree
$d_{x_i}$	Degree of symbol-node $x_i$ , i.e. the number of connected check-nodes
$E_b$	Energy per information bit
$e_b$	Bit errors
$\mathcal{E}_{x_i}$	Set of edges connected to symbol-node $x_i$
$\epsilon$	Crossover probability for the binary symmetric channel (BSC) and the Z-channel
$e_w$	Word errors
$\mathcal{F}_0$	Set of zero-modules that belong to a 2D barcodes finder pattern
$\mathcal{F}_1$	Set of one-modules that belong to a 2D barcodes finder pattern
$g$	Global girth of a LDPC code, i.e. length of the shortest cycle in the LDPC code's Tanner graph
$\mathbf{G}$	Generator matrix of a linear block code
$\mathbf{g}$	Row vector of $\mathbf{G}$ . One of $k$ linear independent codewords that together form the basis $\mathcal{B}$ of the linear block code $C$ .
$g_{x_i}$	Local girth, i.e. the shortest cycle that symbol-node $x_i$ is involved in
$\mathbf{H}$	Parity check matrix of the linear block code $C$
$\mathbf{h}$	Row vector of the parity-check matrix $\mathbf{H}$ . One of $m$ codewords (only linearly independent if $\mathbf{H}$ has full rank) that together form the basis $\mathcal{B}^\perp$ of the dual code $C^\perp$ .
$h_{\kappa}^{s_{r,c}, x_{r,c}}$	Histogram referring to state $s_{r,c}$ and symbol $x_{r,c}$
$\mathcal{H}$	Total number of bins in a histogram $h_{\kappa}^{s_i, x_i}$
$\mathbf{I}$	Identity matrix $\rightarrow$ a square matrix with ones on the diagonal from northwest to southeast and zeros elsewhere
$K$	Number of pixel-rows in a module
$k$	Length of an information word $\mathbf{u}$
$\kappa$	Index for the bins in a histogram $h_{\kappa}^{s_i, x_i}$
$L$	Number of pixel-columns in a module
$\bar{\lambda}$	Centroid of the simplex $\Lambda$
$\bar{\lambda}'$	Centroid of the simplex $\Lambda$ without considering the worst vertex $\lambda_{N+1}$
$\lambda_e$	Expanded vertex
$\lambda_i$	$i^{th}$ vertex in the simplex $\Lambda$ that represents a symbol-node degree distribution $\lambda(x)$
$\lambda_{ic}$	Inward contracted vertex
$\lambda_{oc}$	Outward contracted vertex
$\lambda_r$	Reflected vertex
$\lambda_d$	Fraction of symbol-nodes having degree $d$ ( $d$ connected check-nodes)
$\lambda_{i,j}$	$j^{th}$ coefficient of the symbol-node degree distribution represented by the $i^{th}$ vertex $\lambda_i$ in the simplex $\Lambda$
$\lambda(x)$	Symbol-node degree distribution
$l_{c_j}$	Layer of a PEG-tree in which check-node $c_j$ is located
$L(Q_i)$	Soft-decision referring to symbol-node $x_i$ considering the decoding of LDPC codes
$L(q_{ij})$	Log-domain message sent from symbol-node $x_i$ to check-node $c_j$ considering the decoding of LDPC codes



$L(r_{ji})$	Log-domain message sent from check-node $c_j$ to symbol-node $x_i$ considering the decoding of LDPC codes
$l_{x_i}$	Layer of a PEG-tree in which symbol-node $x_i$ is located
$L(x_i)$	log-likelihood ratio (LLR) of a received value $y_i$ corresponding to a sent bit $x_i$
$m$	Number of redundant symbols in a code word $x$
$n$	Length of a code word $x$
$n()$	Normalization
$N_0$	Spectral noise density
$N_{s_k}$	Number of times that a Markov-modulated channel-model is in state $s_k$
$n_t$	Number of received values that are greater than 1 or less than $-1$
$\mathbf{P}$	Transition probability matrix
$\mathbf{p}$	Parity check bits in a code word
$p_{kl}$	Transition probability for a transition from state $s_k$ to $s_l$
$p_{\theta}^{s_i, x_i}$	Pool of $M$ numbers representing a histogram $h_{\kappa}^{s_i, x_i}$
$R$	Code rate: ratio of information bits to code bits
$r$	Iteration or round of an iterative procedure
$r_{av}$	Average distance of the vertices to the centroid
$\rho_d$	Fraction of check-nodes having degree $d$ ( $d$ connected symbol-nodes)
$\rho(x)$	Check-node degree distribution
$r_{x_a}$	Row of a 2D barcode's data region in which the symbol $x_a$ is located
$s$	A state part of the set of states $\mathcal{S}$
$\sigma$	Standard deviation
$\sigma^2$	Variance
$\mathcal{S}$	Set of states that a Markov-modulated channel is based on
$s_1$	State 1 representing the good sub-channel of a 2-state Markov-modulated channel-model
$s_2$	State 2 representing the bad sub-channel of a 2-state Markov-modulated channel-model
$\mathcal{S}_c^{d_c^{min}}$	Set of check-nodes that have the same number $d_c^{min}$ of adjacent edges
$s_{r,c}$	State referring to the module in row $r$ and column $c$ of a 2D barcode
$t_1$	Average time of a 2-state Markov-modulated channel-model to be in the good sub-channel (i.e. state $s_1$ )
$t_2$	Average time of a 2-state Markov-modulated channel-model to be in the bad sub-channel (i.e. state $s_2$ )
$\mathbf{u}$	Information word of length $k$
$\hat{\mathbf{u}}$	Estimated information word
$\mathcal{V}$	Vector space spanned by all $n$ -tuples over GF(2)
$\chi$	Log-domain message sent from the LDPC subgraph to the Markov subgraph during the estimation-decoding of LDPC codes
$X$	Channel input modeled as a random variable
$x$	Realization of the channel input random variable $X$
$\mathbf{x}$	Sent codeword of length $n$
$\mathbf{x}_{ldpc}$	Code word of a LDPC code
$\mathbf{x}_{ldpc}^{int}$	Interleaved version of a LDPC code word $\mathbf{x}_{ldpc}$ , i.e. a matrix describing a LDPC-based 2D barcode
$\mathbf{x}_{rs}$	Code word of a RS code
$\mathbf{x}_{rs}^{int}$	Interleaved version of a RS code word $\mathbf{x}_{rs}$ , i.e. a matrix describing a Data Matrix code
$\hat{x}$	Estimate of the channel input $x$
$x_i$	Symbol-node of a LDPC code's Tanner graph

---

$X_j$	Set of symbol-nodes connected to check-node $c_j$ in a LDPC code's Tanner graph
$x_{r,c}$	Symbol-node of a LDPC code located in row $r$ and column $c$ of a 2D barcode
$x'$	binary phase-shift keying (BPSK) modulated input bit $x$
$Y$	Channel output modeled as a random variable
$y$	Realization of the channel output random variable $Y$
$\mathbf{y}$	Received data word = sent code word interfered depending on the channel
$\mathbf{y}_{ldpc}^{int}$	Interleaved version of the correlation coefficients of a LDPC-based 2D barcode
$\mathbf{y}_{rs}^{int}$	Interleaved version of the correlation coefficients of a Data Matrix code
$y_{rc}$	Correlation coefficient of the module in row $r$ and column $c$ of a 2D barcode
$z$	Sample of white Gaussian noise
$\zeta$	Message sent from the Markov subgraph to the LDPC subgraph during the estimation-decoding of LDPC codes

# Chapter 1

## Introduction

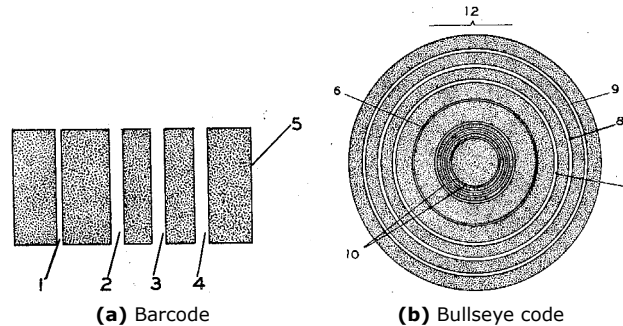
### 1.1 Some history

Since barcodes offer the possibility of reading stored data by means of a computer based system without human involvement, they belong to technologies that are part of Automatic identification and data capture (AIDC) also known as Automatic identification (Auto-ID). In general, AIDC refers to all kinds of technologies that enable one to collect data from objects, pictures, sounds, videos, persons etc. without the need of a manual data entry. Some examples besides barcodes are smart cards, biometrics, radio frequency identification (RFID), optical character recognition (OCR) and voice recognition.

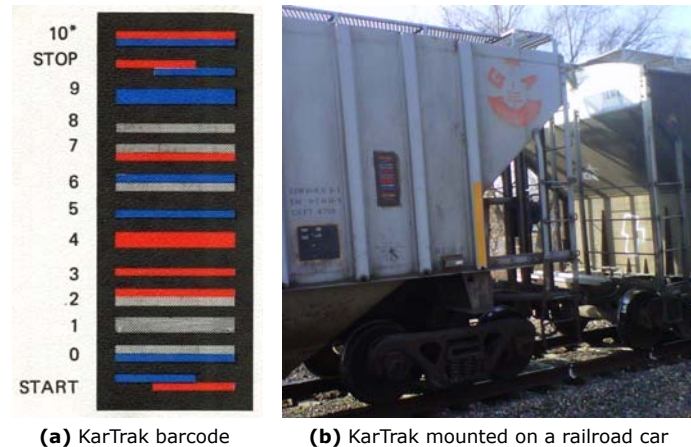
To follow the history of barcodes back to its roots, one has to mention a project started by Wallace Flint in 1932 at the Harvard University Graduate School of Business Administration. The target was the development of a completely automatic checkout procedure for warehouses based on punch cards. The invented system is based on a catalog that lists all available items with an individual punch card assigned to each item. The customer takes a punch card of the desired article, and a clerk then inputs the card into an appropriate reader. The merchandise is then taken fully automatically out of the storeroom, and handed to the customer. A bill is generated, and the inventory data updated. Due to financial reasons, the system was never employed anywhere.

The idea of an automated checkout procedure then inspired Norman Joseph Woodland and Bernard Silver in 1948 to develop a system that enables one to automatically read product information. After a first try based on ultraviolet ink, they introduced the first barcode system in 1949, and applied for a patent. The patent with the name *Classifying apparatus and method* was then issued in 1952 [1]. It describes two barcode systems based on lines (Figure 1.1a) and on a bullseye printing pattern (Figure 1.1b), respectively.

Another milestone in the history of barcodes was the invention of a system by David Collins for automatically identifying railroad cars. The system, developed at the company Sylvania in the United States of America (USA), is called *KarTrak* and is based on reflective orange and blue stripes attached to the railroad cars. Figure 1.2a depicts the KarTrak system, and Figure 1.2b shows a KarTrak barcode mounted on a railroad car. After a first test on gravel cars in 1961, a nationwide standard was established in 1967. However, the system had to face the problem of decreasing reliability in the case of interferences in terms of dirt. For several reasons the system was never applied to the complete fleet of railway cars, and the project failed.



**Figure 1.1:** Barcode and bullseye code developed by N. J. Woodland and B. Silver in 1948 and described in the patent [1].



**Figure 1.2:** KarTrak barcode system developed by D. Collins for automatically identifying railroad cars.

After leaving his former employer Sylvania, David Collins founded the Computer Identics Corporation, and continued to work on barcode systems. The application of helium-neon lasers instead of light bulbs greatly increased reliability when reading the current barcode version based on black and white stripes. Alongside affordable lasers, the development of integrated circuits made barcode scanners more suitably for all kinds of application. One of the first industrial applications of barcodes was the tracking of automobile axle units in a General Motors factory in 1969, based on the system invented by Computer Identics.

However, the main engine for the success of the first barcodes was the grocery industry. Initiated by the National Association of Food Chains (NAFC), guidelines for the development of a standardized barcode were established. After several systems had been proposed and tested, the NAFC chose the Universal Product Code (UPC) in 1973 as their standard for automated checkout procedures. The UPC was developed based on the concept of Silver and Woodland (that was employed at IBM at that time) by George Laurer at IBM.

The first commercial appearance of the UPC was the purchase of a Wrigley's Juicy Fruit gum pack by Clyde Dawson. It was scanned by Sharon Buchanan at 8:01 am on June 26, in 1974, at Marsh's Supermarket in Troy, Ohio, USA. Since then, barcodes have been used more and more in various fields of applications. One of the most famous ones being the superset of UPC: the 13 digit European Article Number (EAN-13) barcode, that is used worldwide for marking all kinds of products. The EAN-13 is standardized in [2]. An example of a UPC and a EAN-13 barcode can be seen in Figure 1.3a and Figure 1.3b, respectively.



**Figure 1.3:** UPC barcode invented by IBM and the EAN-13 barcode that is composed of a UPC and a prefixed zero.

In 1974, David Allais and Ray Stevens developed Code 39 (Figure 1.4) at Intermec. Contrary to the UPC and the EAN-13, the code does not only encode digits but a total of 43 characters. Code 39 was adopted in 1981 by the United States Department of Defense (DoD) in order to mark all kinds of products sold to the United States military. Code 39 is still used by the DoD, and sets a good example for the usage of barcodes in an industrial environment.



**Figure 1.4:** Code 39 used by the United States military.

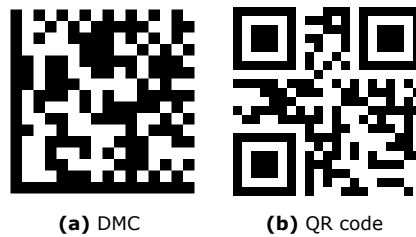
The marking of parts using barcodes offers the possibility of tracking them throughout their entire life cycle, including the manufacturing process and supply chain operations.

Nowadays, one dimensional (1D) barcodes are being replaced by their two dimensional (2D) successors, that offer a higher information density as well as an integrated error-correction in most cases. The international standard [3] refers to bar code verifier conformance specifications for two-dimensional symbols. One 2D barcode<sup>1</sup> that is used more and more in industrial applications is the Data Matrix code (DMC). An example of a DMC can be seen in Figure 1.5a. The DMC was invented by the company International Data Matrix in 1989, and is internationally standardized in [4].

A field of application that drew much attention over the past few years is multimedia applications. In contrast to industrial applications, where the DMC is clearly dominating, a 2D barcode called Quick Response code (QR code) has been spreading intensively. It was invented in 1994 by Denso Wave in Japan, a subsidiary

<sup>1</sup>In this thesis, the word *2D barcode* is used in the context of 2D codes, although their symbols are represented by square modules or dots instead of bars as in the the case of 1D barcodes. This is done to avoid confusion between the barcode and the channel-code used inside the barcode.

of Toyota. Figure 1.5b shows an example of a QR code that is standardized in [5]. However, the focus of this thesis is on the application of 2D barcodes in the context of industrial environments.

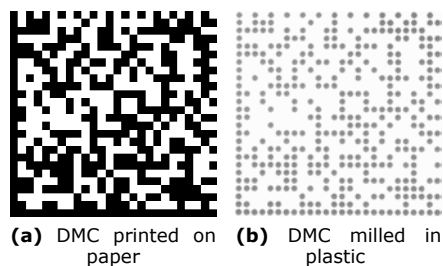


**Figure 1.5:** The two most famous 2D barcodes, the DMC and the QR code.

## 1.2 Motivation

When considering 2D barcode systems in industrial environments, it is important to direct one's attention to the appearance of a 2D barcode's modules. When a DMC, for example, is printed in black on a white surface, each module is represented by a black or a white square referring to a binary one and a binary zero, respectively. This represents the usual case as described in the appropriate standard [4].

However, in the case of direct part mark identification (DPMI) applications, the modules are not printed but instead milled, laser etched or dot peened on various kinds of material. Before the actual decoding can be proceeded, an acquisition of the 2D barcode has to be done. Therefore, a picture is taken in an illumination setting that is adjusted to the surface of the material and the cavities that represent the 2D barcode. In the case of dot peening or milling, the black squares then turn into shapes similar to circles. The sizes and the gray-values of the round modules thereby depend on various parameters like the camera and its setting, the illumination setting, the texture of the cavities and so on. In addition, the white squares, that stand for a binary zero, are now represented by the untouched material, that is not necessarily white anymore as in the printed case. Figure 1.6 shows a standard printed version of a DMC, and an example of a DPMI application where a DMC is milled into plastic.



**Figure 1.6:** The same DMC printed on paper and milled in plastic.

This means that a picture of a 2D barcode looks quite different in the case of DPMI applications compared to the standard case that is characterized by modules depicted

by black or white squares. This mainly affects the following points:

1. The localization of the 2D barcode.
2. The image processing.
3. The error-correction.

The localization of a 2D barcode is much more challenging in the case of DPMI applications, since the edge-detection algorithms given in the standard do not work anymore. This is due to the round shape of the one-modules that do not form detectable edges.

The image processing part, which is responsible for delivering binary information to the following decoder depending on the modules values, suffers from the same issue. It is not possible to decide if a module represents a binary one or a binary zero by means of the provided edge-detection algorithm.

This yields a greater demand on the error-correction capabilities of a 2D barcode, since it is more likely that errors occur due to the increased challenge for the localization and image processing.

Another very important fact that has to be considered in industrial environments is the possibility that damages occur that significantly lower the chance of successful decoding. Typical interferences like blobs, scratches, dirt, rust etc. change the appearance of a 2D barcode's modules, and further impair the conditions for a successful decoding.

The target of this thesis is to design a robust 2D barcode, in order to overcome the unfavorable conditions that a barcode system has to face in DPMI applications.

### 1.3 Thesis outline

#### 1. Introduction

After some barcode history, the motivation for this work is explained. Furthermore, the structure of this thesis is explained, and a list of publications that came out of this work is given.

#### 2. Barcodes

This chapter offers a brief overview of 1D barcodes and 2D barcodes. Typical applications are explained with a focus on 2D barcodes utilized in industrial environments. One 2D barcode called Data Matrix code (DMC) is described in a more detailed manner since it is taken as a reference for the 2D barcode developed in this thesis.

#### 3. Channel-models

Here some basics are provided by considering channel-models that are important throughout this work. Furthermore, it is shown how to compute soft-decisions (SDs) and hard-decisions (HDs) by means of a channels output.

#### 4. Low-density parity-check codes

The 2D barcode designed in this thesis applies a class of channel-codes called low-density parity-check (LDPC) codes. The definition of LDPC codes, the encoding and decoding as well as the available construction and design methods are explained in this Chapter. The focus is thereby on the later application of LDPC codes on 2D barcodes.

#### 5. Design of short irregular LDPC codes

LDPC codes with short block length are addressed since the number of bits that can be stored inside of a 2D barcode is limited by the available space. The class of irregular LDPC codes is promising because these codes have very good error-correction capabilities on a variety of channel-models. There are standard tools available for the determination of the parameters that define irregular LDPC codes that work very good considering long codes. For short block lengths, these tools are not suitable. In this Chapter, an original contribution is given with the development of an optimization method for the design of short irregular LDPC codes that is based on a direct search algorithm. The results based on the new design technique are compared with a method that follows a similar approach. It is proven that the design method developed in this Chapter provides superior decoding performance for the additive white Gaussian noise (AWGN) channel and for the Markov-modulated Gaussian channel (MMGC) compared to well-tried methods.

#### 6. Estimation-decoding

Although estimation-decoding stands for a certain kind of LDPC decoder, it has not been integrated in the Chapter low-density parity-check codes. This is because the basics of estimation-decoding are given first to then explain the concept of a newly developed variant of estimation-decoding. The purpose of using estimation-decoding in general is to increase the error-correction capabilities of a LDPC code by considering the memory of a channel during the iterative decoding. An evaluation proves the effectiveness of using the new variant of estimation-decoding. The developed algorithm is later used when designing the decoder for LDPC-based 2D barcodes.



### 7. LDPC-based 2D barcodes

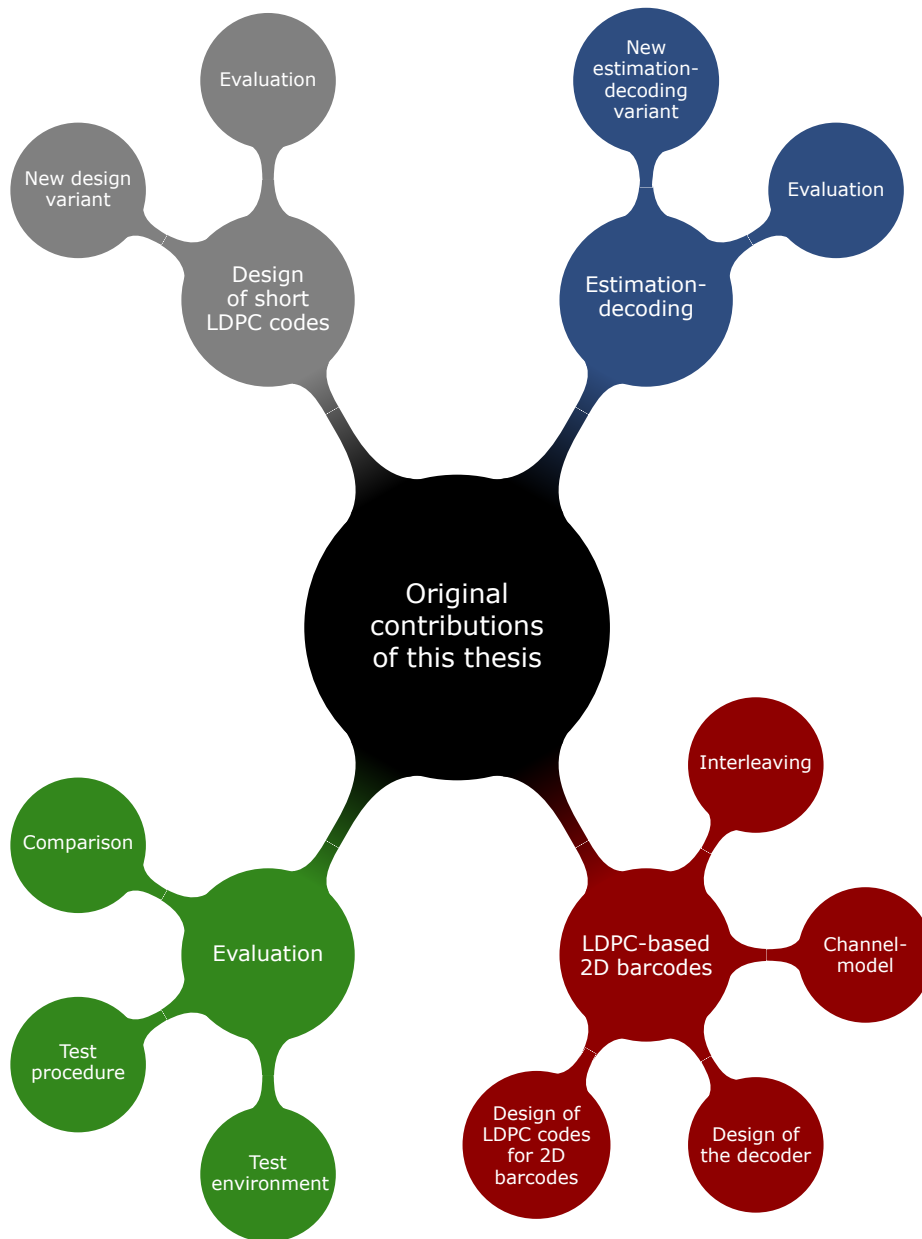
The main contribution of this thesis is explained in this Chapter where the principles of 2D barcodes based on LDPC codes are explained. After a desired information has been encoded by means of a LDPC code, the code word's bits have to be placed in the data region of the 2D barcode. This is done by use of an intelligent interleaver that is developed in order to increase the error-correction capabilities of the 2D barcode. Since the LDPC decoder requires so called soft-decisions (SDs) as an input, a channel-model is constructed by which the computation of SDs is possible. Damages that may occur and that are typical in industrial environments are thereby considered and yield a channel-model with memory. This is why the design of the following decoder is based on the estimation-decoding principle. Furthermore, a 2D hidden Markov model (HMM) is created to represent the channel's memory during the iterative decoding of the LDPC-based 2D barcodes. In a last step short irregular LDPC codes are designed especially for the usage with 2D barcodes in industrial environments. This is done based on the new design method, the developed channel-model and the new variant of estimation-decoding.

### 8. Evaluation

For the evaluation, a test environment and a test procedure are developed that enable a comparison of different versions of 2D barcodes. Based on the test environment and the test procedure, the following evaluations are processed:

- (a) The effectiveness of the optimized symbol-placement by means of the intelligent interleaver is proved.
- (b) The effectiveness of the designed decoder for LDPC-based 2D barcodes is proved.
- (c) The decoding results of 2D barcodes based on regular LDPC codes and irregular LDPC codes are compared.
- (d) The LDPC-based 2D barcode developed in this thesis is compared to the DMC.

The main original contributions of this thesis are graphically summarized in Figure 1.7. The order of appearance in this thesis is thereby given clockwise starting with the design of short LDPC codes. The points in the upper part of Figure 1.7 refer to contributions to the field of LDPC codes in general, whereas the remaining points in the bottom belong to the new class of 2D barcodes developed in this thesis.



**Figure 1.7:** Visualization of the main original contributions of this thesis.

## 1.4 Publications

1. Wolfgang Proß and Franz Quint, "Comparative study of a CDMA2000 Turbo code and a linear time encodable PEG LDPC code over GF(q)," in Proceedings of the 54.IWK - International Scientific Colloquium Conference, Ilmenau, Germany, Sept 2009  
Available: <http://www.db-thueringen.de/servlets/DocumentServlet?id=14478>  
Indexed: scientific commons
2. Wolfgang Proß and Franz Quint, "Decoding performance of Turbo-Codes and LDPC-Codes with short blocklength,"
  - (a) in Proceedings of Doctor ETc 2009, pp.97-102, Timisoara, Romania, Sept 2009
  - (b) in Scientific Bulletin of the Politehnica University of Timisoara - Transactions on Electronics and Communications, Vol. 54(68), No. 1., pp.25-30, 2009  
Available: [http://www.tc.etc.upt.ro/bulletin/2009/vol54\\_68\\_1/vol54\(68\)\\_1\\_5.html](http://www.tc.etc.upt.ro/bulletin/2009/vol54_68_1/vol54(68)_1_5.html)  
Indexed: citeulike
3. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Using PEG-LDPC codes for object identification," in Proceedings of the 9th International Symposium on Electronics and Telecommunications (ISETC2010), pp.361-364, Timisoara, Romania, Nov 2010  
Available: [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5679349&tag=1](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5679349&tag=1)  
Indexed: IEEE, ISI, z150
4. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Estimation-Decoding on LDPC-based 2D-barcode," in Proceedings of the International Conference on Signal Processing and Multimedia Applications (SIGMAP2011), pp.34-39, Seville, Spain, July 2011  
Indexed: ISI, Inspec, DBLP, EI-Compendex, Scopus, Scientific Commons  
Award: Best Student Paper Award
5. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Estimation-Decoding of short blocklength LDPC codes on a Markov-modulated Gaussian channel," in Proceedings of the 3rd IEEE International Conference on Signal Processing Systems (ICSPS2011), pp.383-387, Yantai, China, Aug 2011  
Indexed: ISI, EI-Compendex
6. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Design of short irregular LDPC codes based on a constrained Downhill-Simplex method", in Scientific Bulletin of the Politehnica University of Timisoara - Transactions on Electronics and Communications, Vol. 56(70), No. 2., pp.27-31, 2011  
Available: [http://www.tc.etc.upt.ro/bulletin/pdf/2011vol56\\_70no2.pdf](http://www.tc.etc.upt.ro/bulletin/pdf/2011vol56_70no2.pdf)
7. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Design of short irregular LDPC codes for a Markov-modulated Gaussian channel," in Proceedings of the International Conference on Signal Processing and Multimedia Applications (SIGMAP2012), pp.31-34, Rome, Italy, July 2012  
Indexed: ISI, Inspec, DBLP, EI-Compendex, Scopus, Scientific Commons

8. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Decoding of LDPC-based 2D-barcode using a 2D-Hidden-Markov-Model," in Springer's Lecture Notes in Computer Science (LNCS), Communications in Computer and Information Science (CCIS) series, *in press*  
Indexed: ISI, DBLP, Google Scholar, EI-Compendex, Scopus, SCImago, Mathematical Reviews
9. Wolfgang Proß, Franz Quint and Marius Oteşteanu, "Design of irregular LDPC codes for nonparametric channels," in Proceedings of the 10th International Symposium on Electronics and Telecommunications (ISETC2012), *in press*, Timisoara, Romania, Nov 2012  
Indexed: IEEE

## Chapter 2

# Barcodes

Barcodes in general are defined by machine-readable patterns that store information related to the object that the barcode is attached to. In the context of barcodes, a symbology is defined as the mapping of the data to the barcode. Depending on the chosen barcode, the data that should be encoded can be comprised of digits or a combination of digits and characters. Some barcodes also support special characters like Kanji<sup>1</sup> and Kana<sup>2</sup> in the case of Quick Response codes (QR codes). There is a huge variety of barcodes available that were developed in different times and for different purposes. One can thereby distinguish one dimensional (1D) barcodes and two dimensional (2D) barcodes. In each of the following sections, only a few examples are given since it would go beyond the scope of this thesis to mention all existing barcodes.

### 2.1 1D barcodes

The first barcodes that were invented stored the information linear in one dimension, and thus are denoted as 1D barcodes. The name *barcode* stems from the fact, that most of the former barcodes used bars to encode the desired data. There are 1D barcodes that use only two bar widths, whereas other ones utilize more. In addition, the spaces between the bars may encode information as well.

Figure 1.3a and Figure 1.3b show the Universal Product Code (UPC) and the 13 digit European Article Number (EAN-13) barcode, respectively. Both of them are primarily used to identify consumer products. Contrary to the EAN-13 barcode that is used internationally, the UPC barcode is mainly used in the United States of America (USA) and Canada.

In contrast to the UPC and the EAN-13 barcode, the Code 39 version in Figure 1.4 provides the possibility of encoding characters in addition to numeric digits. This type of barcode is widespread in the automobile and pharmaceutical industries.

The Code 128, that can be seen in Figure 2.1, offers the encoding of alphanumerical characters as well but with a higher information density compared to the Code 39.

Although some of the 1D barcodes offer one check digit, it is not comparable to the error-correction capabilities that most 2D barcodes provide.

---

<sup>1</sup>Logographic Chinese characters adopted for usage in the modern Japanese writing system.

<sup>2</sup>A part of the Japanese syllable script.



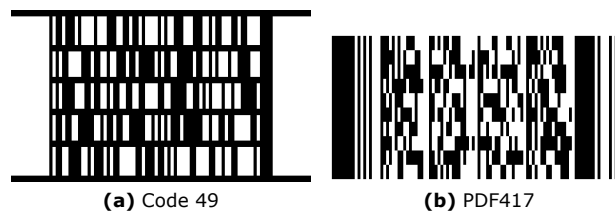
**Figure 2.1:** Code 128.

## 2.2 Stacked barcodes

A stacked barcode comprises several lines with each line being composed of bars and spaces. Usually, the lines share the same start and stop pattern. Contrary to 1D barcodes, most stacked symbologies offer a simple error-correction.

The first stacked barcode was invented in 1987 by David Allais at Intermecc. The integrated error-correction made the system suitable for aerospace applications. The code is called Code 49, and can be seen in Figure 2.2a.

A well-known current representative that is in the public domain is the Portable Data File, 4 bars and spaces in 17 modules (PDF417) that can be seen in Figure 2.2b. It was invented by Ynjiun P. Wang at Symbol Technologies in 1991, and is standardized in [6].



**Figure 2.2:** Stacked barcodes that are composed of several lines.

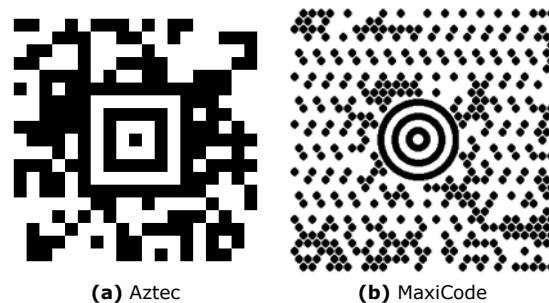
## 2.3 2D barcodes

2D codes store the information in two dimensions. They are also called matrix codes. Even though they are not composed of bars, the common expression 2D barcode is used within this thesis. This is done to avoid confusion between the barcode and the channel-code used inside the barcode. 2D barcodes are state-of-the-art technology in the field of barcodes. The main advantages compared to 1D barcodes and stacked barcodes is a higher information density as well as an integrated error-correction. In the case of the Data Matrix code (DMC) and the QR code for example, the error-correction is obtained by means of a Reed-Solomon channel-code [7]. An example of a DMC and a QR code can be seen in Figure 1.5a and 1.5b, respectively.

One 2D barcode that was developed in 1995 by Andrew Longacre and Robert Hussey at Welch Allyn is the Aztec barcode. The Aztec code is in the public domain and is standardized in [8]. An example can be seen in Figure 2.3a. It is, for example, used by the Deutsche Bahn AG for their online tickets.

The barcode in Figure 2.3b is called MaxiCode and was developed in 1989 by the United Parcel Service (UPS) for the identification of packages. Contrary to most other 2D barcodes that possess square shaped modules, the shape of the MaxiCodes modules is hexagonal.

To complete the list of available barcodes, one should mention the existence of so called *composite symbologies*. These barcodes are in fact a combination of a 1D



**Figure 2.3:** 2D barcodes.

barcode and a 2D barcode. They are mainly used in order to provide a common 1D barcode carrying an article number and in addition adding extra information. This way the article number can be read using a laser-based scanner, and if required the 2D part provides additional information.

### 2.3.1 Data Matrix code (ECC200)

The DMCs are explained in a bit more detail while considering barcodes, since the focus of this thesis is on the DMC. This is because a 2D barcode will be designed for application in industrial environments, and the DMC is widespread in direct part mark identification (DPMI) applications. It is, for example, the only 2D barcode that is specified for the use by GS1<sup>3</sup>.

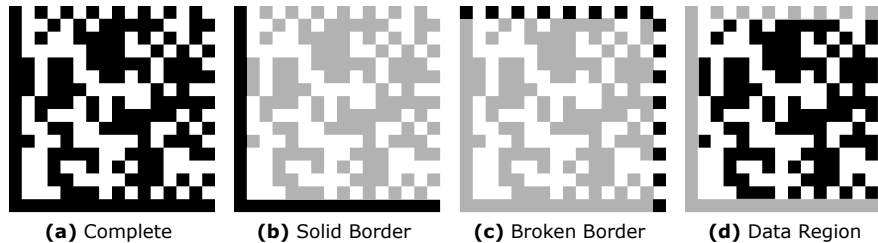
The DMC was invented by International Data Matrix in 1989, and is internationally standardized in [4]. There are several DMC types defined that utilize different variants of error checking and correcting (ECC), and are called ECC000, ECC050, ECC080, ECC100, ECC140 and ECC200. In the following, only the current ECC200 version is considered.

Figure 2.4 shows the three major parts that a DMC consists of. The L-shaped solid border and the broken border together form the finder pattern, which enables one to determine parameters like the physical size, the orientation and the number of modules in the code. A module is thereby one cell of the barcode, and represented by a white or black square. A black square usually stands for a binary one, and a white module for a binary zero. The finder pattern is surrounded by a quiet zone, which is equal to a frame consisting of zero-modules. The actual data can be found in the data region that is inside of the finder pattern. The capacity of a DMC is up to 2,335 alphanumeric characters or 3,116 numbers.

According to the standard [4], there are several encoding schemes available that are American Standard Code for Information Interchange (ASCII), C40, Text, Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT) and Base 256. Throughout this work, only the ASCII encoding is considered. The data is then encoded by means of a Reed-Solomon (RS) channel-code [7] that enables the appropriate decoder to correct errors up to a certain extent.

The high information density together with the possibility of successfully decoding

<sup>3</sup>GS1 is a not-for-profit association that has Member Organizations in over one hundred countries. They mainly work on developing a series of standards called the GS1 system, in order to improve supply-chain management. One of the four key standards thereby refers to barcodes. The GS1 system is the most widely used supply chain standards system in the world.



**Figure 2.4:** Three major parts of a DMC.

a DMC, even if up to 25% of the code is destroyed, have made DMCs a good choice for applications in industrial environments. In addition DMCs are in the public domain.

## 2.4 Direct part mark identification

There are different methods available for attaching a 2D barcode to an item which is referred to as symbol marking. Very often the barcodes are printed in black on a white even surface. However, next to inkjet marking one can find different symbol marking technologies in industrial environments referred to as direct part mark identification (DPMI). DPMI includes all methods where a barcode is permanently marked directly on the appropriate part. Common methods are:

- Dot peening.
- Laser etching.
- Milling.
- Electro-chemical etching.

The barcodes are thereby marked on various kinds of material like metal, plastic, glass and rubber. Figure 2.5a and 2.5b show a dot peening machine and a closeup of a dot peened DMC, respectively. Another example can be seen in Figure 2.5c where a DMC is used on a part of metal and on glass in Figure 2.5d.

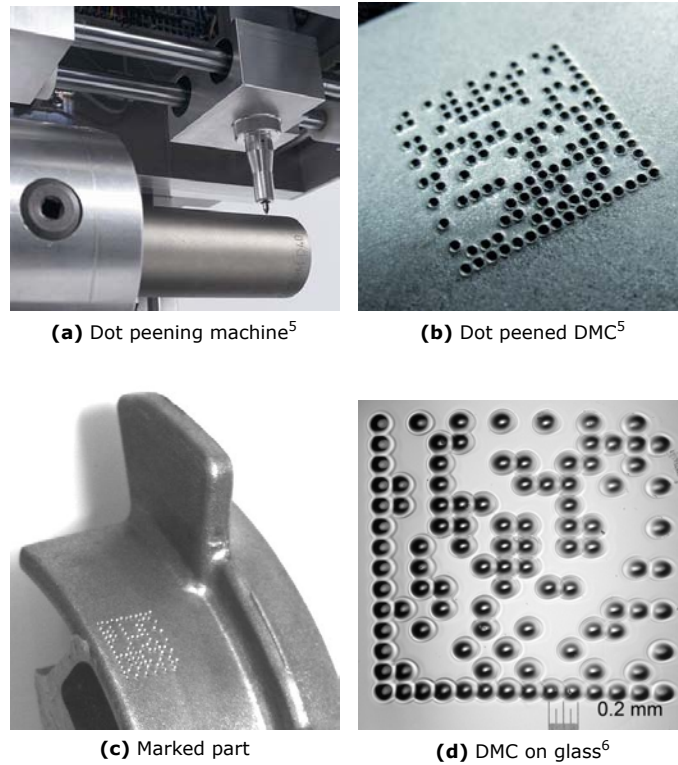
The decision about what type of marking method to use depends not only on the type of material and its composition, but also on factors like the operating environment<sup>4</sup>, the production volume and the space available for marking.

The decoding of barcodes in DPMI applications is much more challenging compared to printed barcodes. This is due to several reasons of which the most important are:

1. The appearance of the 2D barcode's modules changes. In the case of dot peening or milling, a one-module is represented by a round shape instead of a square. This shape varies depending on the quality of the marker and the consistency of the material. Furthermore, an untouched surface in the region of the barcode stands for a zero-module. So the appearance of zero-modules may vary as well, depending on the materials surface.
2. The required illumination to increase the contrast between the one-modules and the zero-modules. The appearance of the modules may also be influenced by the illumination setting. If the light conditions are not exactly the same for all modules for example, the shapes of the modules as well as the contrast varies.

<sup>4</sup>The environment in which the part will be used also considering the part's life time.





**Figure 2.5:** Examples for DPMI applications.

Several international standards have been set up in order to unify the usage of 2D barcodes in DPMI applications, and to ensure a certain quality.

The first industry barcode standard was published in 1984 by the Automotive Industry Action Group (AIAG) in [9]. A current standard of the AIAG referring to 2D barcodes in the context of DPMI applications can be found in [10], which is a supplement of [11]. Other AIAG standards referring to 2D symbologies in industrial environments are [12] and [13]. The International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) defines barcode print quality test specifications for 2D symbols in [14]. The United States Department of Defense (DoD) regulates machine-readable information applications including 2D barcodes in [15]. The Air Transport Association (ATA) provides a standard for Automatic identification and data capture (AIDC) in [16]. The standards [17] and [18] have been published by the Electronic Industry Association (EIA). The National Aeronautics and Space Administration (NASA) and the International Aerospace Quality Group (IAQG) published standards only considering DMCs in [19], [20] and [21], respectively.

Motivated by the inherently difficult conditions in industrial environments, this thesis provides a robust 2D barcode for DPMI applications (see Section 1.2).

<sup>5</sup>Picture published with kind permission of Markator, <http://markator.de/>.

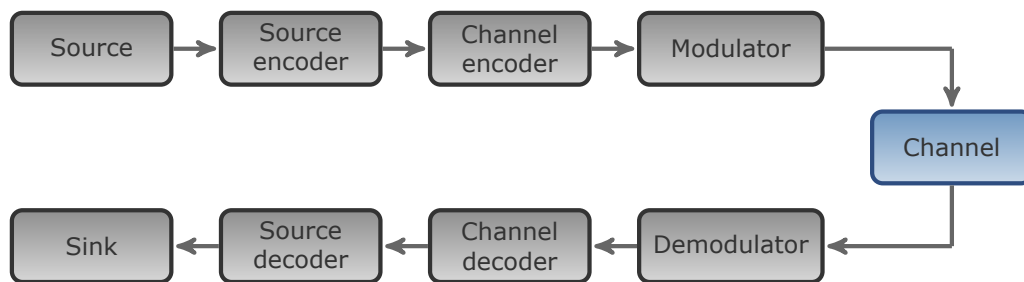
<sup>6</sup>Picture published with kind permission of Vesdo, <http://www.vesdo.com/>.



## Chapter 3

### Channel-models

To understand what a channel-model is and what it is used for, it is useful to have a look at the principle of a digital communication system. Figure 3.1 is derived from the framework that Claude Shannon introduced in [22], and depicts the main important parts involved in a digital communication system.



**Figure 3.1:** Principle of a digital communication system.

1. *Source*  
The Source is represented by binary information.
2. *Source encoder*  
A source encoder enables a compression of the source. This means that the amount of binary information is reduced by removing redundant information. The information word that consists of  $k$  bits is then passed to the channel encoder.
3. *Channel encoder*  
A channel encoder then adds  $m$  bits of redundant information and outputs a codeword of length  $n = k + m$ . This is done according to the established channel-code's underlying scheme that is also known to the appropriate channel decoder.
4. *Modulator*  
A modulator converts the code bits into a signal that is appropriate for the following channel.

### 5. Channel

The channel is the physical medium that is used to transmit the signal from the modulator to the demodulator. The original signal sent from the modulator is thereby subject to noise.

### 6. Demodulator

The demodulator reverses the conversion of the modulator in order to pass the codeword to the channel decoder. The codeword can thereby either consist of bits or soft-decisions (SDs), where a SD is the probability of a bit to be a zero or a one (see Section 3.6 for more details).

### 7. Channel decoder

The channel decoder estimates the original binary information that was passed to the channel encoder. It is thereby possible to correct a certain amount of errors based on the redundant part of the codeword and the knowledge of the channel coding scheme. So the  $m$  redundant bits added by the channel-encoder protect the  $k$  information bits against errors that may occur due to the channel's noise. Each code bit thereby carries the information of  $R = k/n$  information bits where  $R$  is called the code rate.

### 8. Source decoder

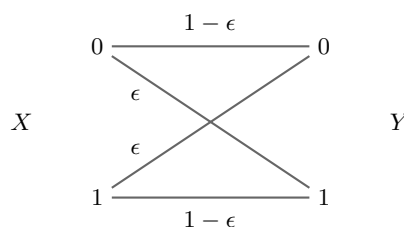
The Source decoder recovers the original digital information, i.e. it adds the redundant part removed by the source encoder. In the case of a lossy source encoding (e.g. MP3<sup>1</sup>), it approximates the source sequence.

As the name suggests, a *channel-model* is a model that represents the channel. A channel-model is, for example, used to choose, design and construct an appropriate channel-code for the usage with a channel that is approximated by the channel-model. During this process, the decoding performance of a specific code can be evaluated by means of a simulation that is based on the chosen channel-model (see Section 4.6).

In the following pages, the basics of some channel-models that are important when considering this thesis are explained.

## 3.1 Binary symmetric channel

The binary symmetric channel (BSC) is depicted in Figure 3.2 with  $X$  and  $Y$  being the channel's *binary* input and *binary* output, respectively.  $\epsilon$  represents the crossover probability  $P(y = 0 | x = 1) = P(y = 1 | x = 0) = \epsilon$ . It is the same for both possible crossovers and hence the channel is called *symmetric*. The probability for receiving the value that has been sent is then  $P(y = 0 | x = 0) = P(y = 1 | x = 1) = 1 - \epsilon$ .



**Figure 3.2:** BSC.

<sup>1</sup>Actually referred to as moving picture experts group (MPEG)-2 Audio Layer 3

### 3.2 Z channel

Contrary to the BSC, the Z channel (Figure 3.3) is asymmetric since the model only covers a possible flip from a binary one to a zero with probability  $P(y = 0 | x = 1) = \epsilon$ . Thus the probabilities for an input bit to not get flipped is  $P(y = 1 | x = 1) = 1 - \epsilon$  and  $P(y = 0 | x = 0) = 1$ . The Z channel is often used for optical communications.

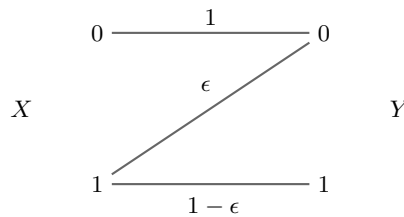


Figure 3.3: Z-channel.

### 3.3 Additive white Gaussian noise channel

In the case of a binary input additive white Gaussian noise (AWGN) channel, white Gaussian noise  $z$  is added to the input value  $x$  (see Figure 3.4).

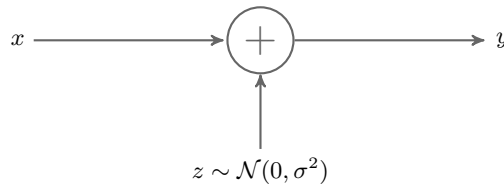


Figure 3.4: AWGN channel.

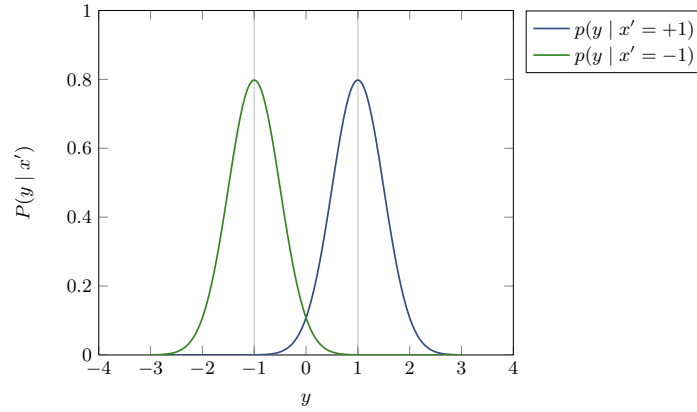
If a binary phase-shift keying (BPSK) modulation is used with

$$x' = -2x + 1, \quad (3.1)$$

the received value  $y$  is distributed according to

$$P(y | x') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - x')^2}{2\sigma^2}\right). \quad (3.2)$$

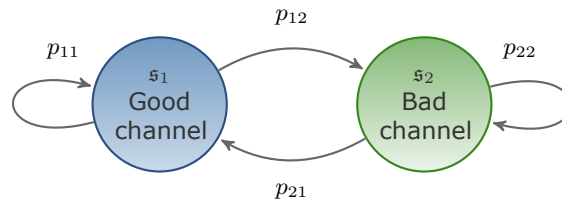
This can be seen in Figure 3.5 that shows an example of the so called *likelihood functions* (the conditional probability density functions (PDFs) of  $y$  conditioned on  $x'$ ). The abscissa shows the range of values for  $y$  where the ordinate represents the probability for the appropriate  $y$ -value.



**Figure 3.5:** Likelihoods in the case of BPSK modulation and an AWGN channel.

### 3.4 Markov-modulated channels

So far, only memoryless channels have been presented where a current value received at the channel's output is independent of the value received before or after the current observation. In contrast to that, Markov-modulated channels are channels with memory. The memory of the channel is modeled by means of channel states. Figure 3.6 for example, shows a channel with memory based on a set  $\mathcal{S} = \{s_1, s_2\}$  of two states. Dependent on the current state  $s = s_k$  that the channel is inside, a sent bit is affected by one of the two sub-channels represented by the two states. The state for the next channel use is determined based on the current state and the transition probabilities between the states. Since a received value is dependent on the current state which in turn is dependent on the states before, the channel is said to have a memory. When looking at a Markov-modulated channel from a time perspective, it yields a hidden Markov model (HMM) [23][24]. The received values  $y$  are known, whereas the channel-states  $s$  and the sent bits  $x$  are hidden sequences.



**Figure 3.6:** 2-state Markov-modulated channel.

In the case of two states, it is common to denote the sub-channel that is worse in terms of noise as *bad channel* and the other one as *good channel*. In the example in Figure 3.6 and throughout this thesis, the good channel and the bad channel are represented by state  $s_1$  and state  $s_2$ , respectively. The transition probabilities are often written in matrix form, which for the 2-state Markov-modulated channel becomes

$$\mathbf{P} = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix} \quad (3.3)$$

with  $p_{11} = 1 - p_{12}$  and  $p_{22} = 1 - p_{21}$ . The average time  $t_1$  of the channel-model to be in the good sub-channel is computed according to

$$t_1 = \frac{p_{21}}{p_{12} + p_{21}}, \quad (3.4)$$

and  $t_2$  by

$$t_2 = \frac{p_{12}}{p_{12} + p_{21}}. \quad (3.5)$$

### 3.4.1 Gilbert-Elliott channel

The Gilbert-Elliott channel depicted in Figure 3.7 is a 2-state Markov-modulated channel with each of the two sub-channels being represented by a BSC (see Figure 3.2). The crossover probability  $\epsilon_1$  of the good channel is thereby lower than  $\epsilon_2$  of the bad channel. The concept of the Gilbert-Elliott channel was introduced in [25].

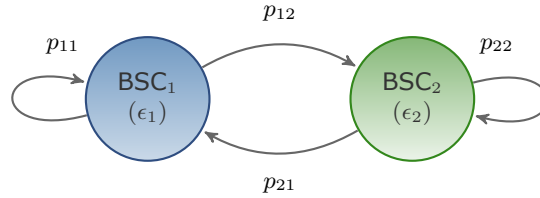


Figure 3.7: Gilbert-Elliott channel.

### 3.4.2 Markov-modulated Gaussian channel

When the two sub-channels are represented by AWGN channels, the resulting channel-model is called Markov-modulated Gaussian channel (MMGC). It is depicted in Figure 3.8 where  $\sigma_2 = d_\sigma \cdot \sigma_1$  with  $d_\sigma > 1$  and thus  $\sigma_1^2 < \sigma_2^2$ .

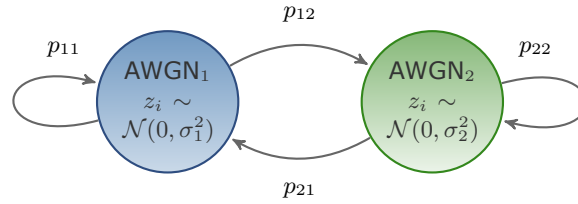
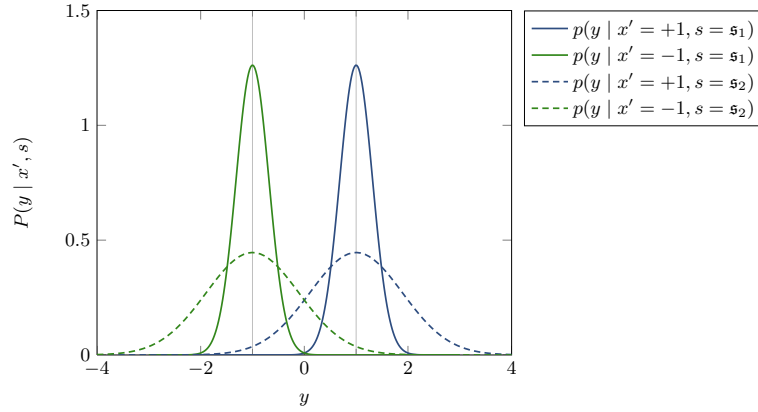


Figure 3.8: 2-state Markov-modulated Gaussian channel.

Figure 3.9 shows an example of the likelihood-functions that are conditional PDFs for the random variable  $Y$  (the channel output) conditioned by the channel input  $X$  and the state  $s = \mathbf{s}_k$  (i.e. the sub-channel). The likelihood  $P(y | x', s = \mathbf{s}_k)$  is computed by

$$P(y | x', s = \mathbf{s}_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(y - x')^2}{2\sigma_k^2}\right) \quad (3.6)$$

with  $s \in \mathcal{S} = \{\mathbf{s}_k\}_{k=\{1,2\}}$ .



**Figure 3.9:** Likelihoods in the case of BPSK modulation and a MMGC.

### 3.5 Log-likelihood ratio

The log-likelihood ratio (LLR) is a probability measure often used in the context of soft-decisions (SDs) referring to single bits. It is computed by

$$L(x) = \log \left[ \frac{P(x=0 | y)}{P(x=1 | y)} \right] \quad (3.7a)$$

which in the case of BPSK modulation becomes

$$L(x) = \log \left[ \frac{P(x' = +1 | y)}{P(x' = -1 | y)} \right] \quad (3.7b)$$

where  $P(x | y)$  is called the *a posteriori probability*<sup>2</sup>. Following the Bayes' theorem, it is computed according to

$$P(x | y) = \frac{P(y | x) \cdot P(x)}{P(y)}. \quad (3.8)$$

$P(y | x)$  is thereby the *likelihood*, and  $P(x)$  is called the *a priori probability*. From Equation (3.7a) and (3.8), it follows

$$L(x) = \log \left[ \frac{P(x=0 | y)}{P(x=1 | y)} \right] = \log \left[ \frac{P(y | x=0)}{P(y | x=1)} \right] + \log \left[ \frac{P(x=0)}{P(x=1)} \right]. \quad (3.9)$$

For the AWGN channel, the LLR computed based on Equation (3.7b) becomes

$$\begin{aligned} L(x) &= \log \left[ \frac{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-1)^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y+1)^2}{2\sigma^2}\right)} \right] \\ &= -\left(\frac{(y-1)^2}{2\sigma^2}\right) + \left(\frac{(y+1)^2}{2\sigma^2}\right) \\ &= \frac{2y}{\sigma^2}. \end{aligned} \quad (3.10)$$

<sup>2</sup>Wherever  $\log[]$  is used in this thesis, the natural logarithm is assumed. The notation  $L(x)$  is used to simplify the notation which would actually be  $L(x | y)$ .



### 3.6 Soft-decision and hard-decision

Before passed to the channel-decoder, the received values  $y$  are either transferred to SDs or hard-decisions (HDs) depending on the format the decoder expects. Based on the channel-model, the probabilities for each bit to be a zero or a one are computed. These probabilities are called SDs since a final decision for the bit to be a zero or a one has not been made yet. SD decoders often process LLRs that are computed based on Equation (3.9).

In the case of a HD-based decoder, a decision (HD) for each bit to be a zero or a one is made before the decoding process starts. In fact, the decision is an estimate  $\hat{x}$  on the originally transmitted bit  $x$ , and is based on the probabilities (i.e. the SDs) so that

$$\hat{x} = \begin{cases} 0 & \text{if } P(x = 0 | y) > P(x = 1 | y); \\ 1 & \text{else} \end{cases} \quad (3.11a)$$

which for LLRs becomes

$$\hat{x} = \begin{cases} 0 & \text{if } L(x) > 0; \\ 1 & \text{else.} \end{cases} \quad (3.11b)$$

The decision rule based on Equation (3.11) is known as maximum a posteriori (MAP), and is based on the a posteriori probabilities (APPs)  $P(x = 0 | y)$  and  $P(x = 1 | y)$ . However, if the a priori probability  $P(x)$  is not known, it is usually assumed that  $P(x = 0) = P(x = 1)$ . Hence the last term in Equation (3.9) is zero.

$$\begin{aligned} L(x) &= \log \left[ \frac{P(x = 0 | y)}{P(x = 1 | y)} \right] = \log \left[ \frac{P(y | x = 0)}{P(y | x = 1)} \right] + \underbrace{\log \left[ \frac{P(x = 0) = 0.5}{P(x = 1) = 0.5} \right]}_{=0} \\ &= \log \left[ \frac{P(y | x = 0)}{P(y | x = 1)} \right]. \end{aligned} \quad (3.12)$$

The decision based on Equation (3.11) and (3.12) is then made by only considering the likelihoods  $P(y | x = 0)$  and  $P(y | x = 1)$ , and is thus known as maximum likelihood (ML).



## Chapter 4

# Low-density parity-check codes

### 4.1 Representation of LDPC codes

Low-density parity-check (LDPC) codes are a class of linear block-codes, and belong to the field of forward error correction (FEC) also referred to as channel-coding. With the introduction of LDPC codes by Rober Gallager in 1962 [26], a practical counterpart was found to the *Shannon limit* defined in 1948 by Claude E. Shannon [22]. He theoretically derived an upper bound (known as *channel capacity*) for an error-free transmission of information through a channel. The concept of LDPC codes then provided a coding scheme that is asymptotically optimal. This means that the channel capacity can be reached under the condition of infinite block length. Even though the capacity approaching LDPC codes have been forgotten for more than 30 years. This was mainly due to the absence of computer processing power that we have today. This made these codes impractical at that time.

The first practical coding scheme that provided near Shannon limit performance was given with the introduction of turbo codes in 1993 by Claude Berrou [27]. Shortly after that David J. C. MacKay and Radford M. Neal [28] rediscovered LDPC codes in parallel to Niclas Wiberg [29] in 1995.

Since LDPC codes belong to the class of linear block codes, the following definitions also hold for any linear block code. The notation thereby follows the approach in [30]. The information sequence of a block-code is segmented into blocks of  $k$  symbols where in the case of a binary code a symbol is represented by one bit. Thus there are  $2^k$  possible messages  $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$ . A message  $\mathbf{u}$  is then encoded to obtain a codeword  $\mathbf{x} = (x_0, x_1, \dots, x_{n-1})$  of length  $n$ . Since  $n > k$ ,  $\mathbf{x}$  includes a redundant part of  $m = n - k$  bits. Based on the redundant bits, it is possible for an appropriate decoder to correct a certain amount of errors that may occur. The amount of redundancy determines the *code rate*  $R$  that is defined by

$$R = \frac{k}{n}. \quad (4.1)$$

An  $(n, k)$  block code is defined by the mapping of  $2^k$  distinct messages to  $2^k$  distinct codewords. The block code is called linear if the  $2^k$  codewords form a  $k$ -dimensional subspace  $\mathcal{C}$  of the vector space  $\mathcal{V}$  spanned by all binary  $n$ -tuples. Then there exist  $k$  linearly independent codewords  $\mathbf{g}_0, \mathbf{g}_1, \dots, \mathbf{g}_{k-1}$  in  $\mathcal{C}$  that together form the basis  $\mathcal{B}$  of  $\mathcal{C}$ . A codeword

$$\mathbf{x} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \dots + u_{k-1}\mathbf{g}_{k-1} \quad (4.2)$$

is then a linear combination of  $\mathcal{B}$ . If written in matrix style this becomes

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G} \quad (4.3)$$

with  $\mathbf{G}$  being the generator matrix.

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix} \quad (4.4)$$

$\mathbf{G}$  is called the generator matrix since a linear combination of the rows of  $\mathbf{G}$  with the information bits in  $\mathbf{u}$  as coefficients generates a codeword  $\mathbf{x}$  (see Equation (4.2)).

The *null space* of  $\mathcal{C}$  is an  $m$ -dimensional subspace  $\mathcal{C}^\perp$  of  $\mathcal{V}$  that is also called *dual space* or *dual code* of  $\mathcal{C}$ . There are  $m$  linearly independent codewords  $\mathbf{h}_0, \mathbf{h}_1, \dots, \mathbf{h}_{m-1}$  in the basis  $\mathcal{B}^\perp$  of  $\mathcal{C}^\perp$ . Thus the dual code is a  $(n, m)$  linear block code. The generator matrix for this dual code is then

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{m-1} \end{bmatrix} = \begin{bmatrix} h_{0,0} & h_{0,1} & \cdots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & \cdots & h_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m-1,0} & h_{m-1,1} & \cdots & h_{m-1,n-1} \end{bmatrix}. \quad (4.5)$$

In the context of the  $(n, k)$  linear block code  $\mathcal{C}$ ,  $\mathbf{H}$  is called the parity-check matrix (PCM). Because a codeword in  $\mathcal{C}$  and a codeword in the dual-code  $\mathcal{C}^\perp$  are two orthogonal vectors whose cross-product is zero,  $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{O}$  with  $\mathbf{O}$  being a  $k \times m$  all-zero matrix. Thus the code  $\mathcal{C}$  is also defined by the parity-check equations

$$\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0} \quad (4.6)$$

with  $\mathbf{0}$  being a all-zero  $m$ -tuple. It then follows that

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{V} : \mathbf{x} \cdot \mathbf{H}^T = \mathbf{0}\} \quad (4.7)$$

which enables one to check any combination of  $n$  bits for being a valid codeword of  $\mathcal{C}$ . In other words, each row vector of  $\mathbf{H}$  performs a parity-check on a received data word  $\mathbf{y}$ . This is where the name PCM comes from.

So a linear block code is specified by two matrices, the generator matrix  $\mathbf{G}$  and the PCM  $\mathbf{H}$ . The encoding is usually done by means of  $\mathbf{G}$  with Equation (4.3) and the decoding based on  $\mathbf{H}$ .

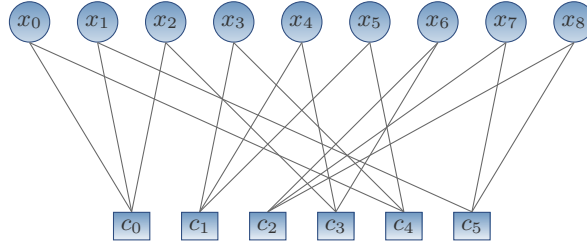
LDPC codes are linear block codes characterized by a low density of non-zero elements in their PCM. This is where the first part of the name low-density parity-check (LDPC) code stems from. The second part refers to the  $m$  parity-check equations that  $\mathbf{H}$  consists of. So one common way to describe a LDPC code is given by means of the underlying PCM.

There has been some notable exceptions during the 33 years where LDPC codes were disregarded, namely the work of Margulis [31], Zyablov and Pinsker [32] and Tanner [33]. The latter introduced an alternative graphical representation of LDPC codes by means of a bipartite graph called *Tanner graph*. It consists of  $n$  symbol-nodes and  $m$  check-nodes representing the  $n$  columns and  $m$  rows of the appropriate PCM, respectively. There are as many edges in the Tanner graph as non-zero entries in the PCM. In the case of a binary LDPC code, the symbols of a codeword as well as the entries of a PCM are bits, and thus non-zero entries are in fact one-entries. The edges

connect the symbol-nodes and check-nodes corresponding to the non-zero entries in the PCM. Figure 4.1 shows a small example of a PCM, the respective parity-check equations and the according Tanner graph.

$$\begin{array}{l}
 c_0 = x_0 + x_1 + x_2 \\
 c_1 = x_3 + x_4 + x_5 \\
 c_2 = x_6 + x_7 + x_8 \\
 c_3 = x_2 + x_4 + x_6 \\
 c_4 = x_0 + x_3 + x_5 \\
 c_5 = x_1 + x_7 + x_8
 \end{array}
 \begin{pmatrix}
 x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\
 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1
 \end{pmatrix}$$

(a) PCM



(b) Tanner graph

**Figure 4.1:** PCM of a LDPC code and the appropriate Tanner graph.

The number of edges connected to the symbol-nodes (i.e. the column weight of the PCM) is described by use of a polynomial  $\lambda(x)$  which is called the symbol-node degree distribution (SNDD).

$$\lambda(x) = \sum_{d \geq 2}^{d_x^{max}} \lambda_d x^d. \quad (4.8)$$

$\lambda_d$  is the fraction of symbol-nodes having  $d$  adjacent edges with  $d_x^{max}$  being the maximum number of connected edges. A symbol-node  $x_i$  with  $d_{x_i}$  connected edges is denoted to have degree  $d_{x_i}^{-1}$ . The check-node degree distribution (CNDD) is given by

$$\rho(x) = \sum_{d \geq 2}^{d_c^{max}} \rho_d x^d. \quad (4.9)$$

If a pair of degree distributions (for the symbol-nodes and check-nodes) is represented by monomials, all symbol-nodes and all check-nodes exhibit the same number of adjacent edges, respectively. Such a LDPC code is called *regular* LDPC code.  $\lambda(x) = x^3$  for example, denotes a LDPC code with three adjacent edges for all symbol-nodes and thus a column-weight of three for all columns. In contrast to regular LDPC codes, *irregular* LDPC codes exhibit several row weights and column weights.

<sup>1</sup>The variable  $x$  of the polynomial that describes the SNDD (or the CNDD) is not to be mistaken for a sent bit or a symbol-node  $x_i$ .

## 4.2 Ensembles of LDPC codes

In literature, it is common to analyze *ensembles* of LDPC codes instead of a single specific LDPC code. Ensembles of LDPC codes are thereby groups of LDPC codes that share certain properties. One example would be several LDPC codes with the underlying PCM constructed based on the same SNDD. Although the symbol-nodes of the different LDPC code share the same number of adjacent edges, they may be connected to different check-nodes. Speaking in terms of the PCM, the column weight of the PCMs would be the same, whereas the locations of the ones in  $\mathbf{H}$  could be different.

The basis for performance analysis of LDPC code ensembles is given by Richardson and Urbanke in [34]. They proved that the average performance of LDPC codes of an ensemble are well approximated by the performance of nearly every individual code in the ensemble. This is called the *concentration theorem* which is restricted to long codes. It is therefore possible to predict the performance of a specific LDPC code based on the knowledge of the average performance of the appropriate ensemble. This is shown in [34] and [35]. For the evaluation of the average performance of an LDPC code ensemble, a method called density evolution is available (see Section 4.5.3).

## 4.3 Encoding of LDPC codes

A LDPC code is usually created by constructing the underlying PCM. However, one has to compute the generator matrix  $\mathbf{G}$  in order to encode an information word. This is done by putting the PCM into the following shape.

$$\mathbf{H}_{m \times n} = [\mathbf{I}_{m \times m} \quad \mathbf{P}_{m \times k}^T] \quad (4.10)$$

with  $\mathbf{I}$  being the identity matrix of size  $m \times m$ . This is done via a method called *Gaussian reduction* [36]. One can then rearrange the matrix to obtain

$$\mathbf{G}_{k \times n} = [\mathbf{P}_{k \times m} \quad \mathbf{I}_{k \times k}]. \quad (4.11)$$

A codeword is then computed by means of Equation (4.3). The obtained LDPC code is called a systematic code, which means that a codeword  $\mathbf{x}$  contains the information word  $\mathbf{u}$  in the last  $k$  bits. The codeword is then

$$\mathbf{x} = (p_0, p_1, \dots, p_{m-1}, u_0, u_1, \dots, u_{k-1}). \quad (4.12)$$

The first  $m$  bits are called *parity check bits*. The encoding based on Equation (4.3) is of complexity  $\mathcal{O}(N^2)$ .

An alternative encoding method with linear complexity  $\mathcal{O}(N)$  is described in [37]. It is based on a PCM  $\mathbf{H}$  in an upper-triangular structure.

$$\mathbf{H}_{m \times n} = [\mathbf{H}_{m \times m}^p \quad \mathbf{H}_{m \times k}^u] \quad (4.13a)$$

with

$$\mathbf{H}_{m \times m}^p = \begin{bmatrix} 1 & h_{0,1}^p & \cdots & h_{0,m-1}^p \\ 0 & 1 & & \\ \vdots & \vdots & \ddots & \vdots \\ & & 1 & h_{m-2,m-1}^p \\ 0 & \cdots & 0 & 1 \end{bmatrix}. \quad (4.13b)$$

The parity check bits  $p$  of  $x$  are then computed recursively starting from  $i = m - 1$  to  $i = 0$  according to

$$p_i = \sum_{j=i+1}^{m-1} h_{i,j}^p p_j + \sum_{j=0}^{k-1} h_{i,j}^u u_j, \quad (\text{mod}2). \quad (4.14)$$

This yields a systematic code (see Equation (4.12)).

It is not always necessary to utilize the encoding by means of Equation (4.3) or (4.14) since one can use the so called all-zero code word (AZCW) in many scenarios. The AZCW is a  $n$ -bit vector with all bits set to zero, and since  $\mathbf{0}\mathbf{H}^T = \mathbf{0}$ , it is a valid codeword of every linear block code.

#### 4.4 Decoding of LDPC codes

Gallager also provided a decoding algorithm when he introduced LDPC codes [26] that is nowadays called belief propagation (BP), sum-product (SP) or message-passing (MP) algorithm. Since MP refers to all kind of algorithms where messages are passed around in a graph, and SP only reflects the non-logarithmic decoding algorithm that incorporates sums and products, the term BP is used here. The BP algorithm was originally developed in 1982 to operate on trees [38], and was then used on general graphs in 1988 [39]. Concerning LDPC codes, the BP algorithm operates on the Tanner graph.

The particular characteristic for the decoding of LDPC codes is the iterative behavior that takes advantage of the so called *turbo-principle*. The name is taken from the turbocharger principle of an engine [40]. Similar to the feedback loop by means of an exhaust system, the LDPC decoder draws on the feedback system based on an iterative decoding. The special feature is thereby the computation of *extrinsic* information exchanged in terms of messages between the two types of nodes in the Tanner graph. Extrinsic information means that in each iteration, the nodes receive new information from neighboring nodes. This is possible since a symbol-node of the Tanner graph (i.e. a symbol of a codeword) is involved in several parity-check equations.

In the following, the less complex logarithmic variant of the BP algorithm is described.

##### 4.4.1 Log-domain BP decoder

The BP decoder in the log-domain is comprised of five steps that are explained in the following.

###### 1<sup>st</sup> step: Initialization

Before the decoding process starts, the received data word  $\mathbf{y} = (y_0, y_1, \dots, y_{n-1})$  is processed in order to compute the log-likelihood ratio (LLR)  $L(x_i)$  for each value  $y_i$  according to Equation (3.7). Then the Tanner graph's symbol-nodes are initialized with the LLRs, also referred to as soft-decisions (SDs). The symbol-nodes then send messages to all connected check-nodes containing the initialized SDs. So the message  $L(q_{ij})$ , sent from symbol-node  $x_i$  to check-node  $c_j$ , is given by

$$L(q_{ij}) = L(x_i). \quad (4.15)$$

The message  $L(q_{ij})$ , sent from symbol-node  $x_i$  to a connected check-node  $c_j$ , contains the probability for  $x_i$  to be a zero or a one.

**2<sup>nd</sup> step: Check-node update**

By means of the received messages  $L(q_{ij})$ , the check-nodes compute messages  $L(r_{ji})$  that they subsequently send to the adjacent symbol-nodes. A message  $L(r_{ji})$ , sent from check-node  $c_j$  to symbol-node  $x_i$ , is thereby calculated according to

$$L(r_{ji}) = \prod_{i' \in X_j \setminus i} \alpha_{i'j} \cdot \phi \left( \sum_{i' \in X_j \setminus i} \phi(\beta_{i'j}) \right) \quad (4.16)$$

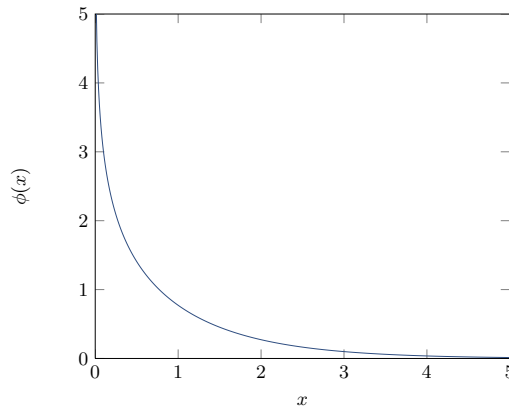
with

$$\begin{aligned} L(q_{ij}) &= \alpha_{ij} \beta_{ij}, \\ \alpha_{ij} &= \text{sign}[L(q_{ij})], \\ \beta_{ij} &= |L(q_{ij})|. \end{aligned}$$

$X_j \setminus i$  thereby denotes the set of symbol-nodes adjacent to check-node  $c_j$ , excluding the symbol-node  $x_i$  to which the current message will be sent. This way check-node  $c_j$  tells symbol-node  $x_i$  the probability for  $x_i$  to be a zero or a one, so that the parity-check equation of  $c_j$  would be fulfilled, considering the messages arrived at  $c_j$  from the other connected symbol-nodes. The  $\phi$ -function is defined by

$$\phi(x) = -\log[\tanh(x/2)] = \log \left[ \frac{e^x + 1}{e^x - 1} \right]. \quad (4.17)$$

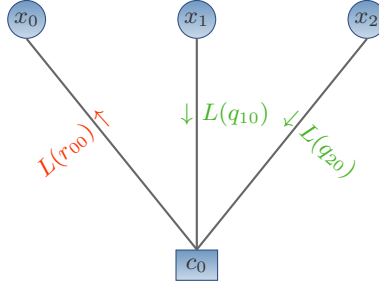
Figure 4.2 shows the  $\phi$ -function that can be implemented by using a look-up table.



**Figure 4.2:**  $\phi$ -function.

Figure 4.3 shows an example of a check-node update where a check-node  $c_0$  is depicted together with three connected symbol-nodes. The new message  $L(r_{00})$  is thereby computed based on the incoming messages  $L(q_{10})$  and  $L(q_{20})$ .





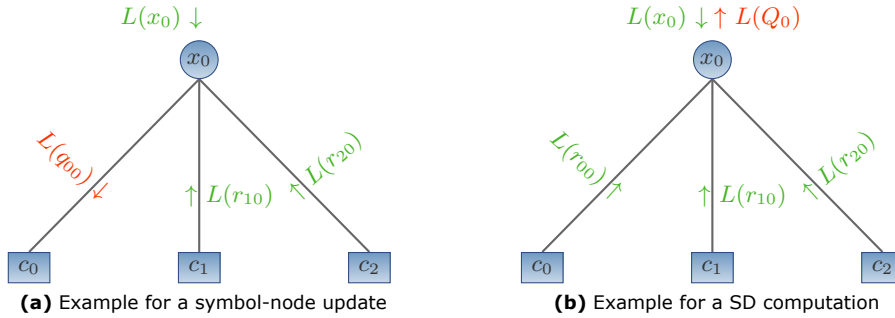
**Figure 4.3:** Example for a check-node update.

### 3<sup>rd</sup> step: Symbol-node update

The symbol-node update is done by computing new messages to send to the adjacent check-nodes. The messages are calculated according to

$$L(q_{ij}) = L(x_i) + \sum_{j' \in C_i \setminus j} L(r_{j'i}). \quad (4.18)$$

$C_i \setminus j$  thereby denotes the set of check-nodes adjacent to symbol-node  $x_i$ , excluding the check-node  $c_j$  to which the current message will be sent. So a message  $L(q_{ij})$ , sent from  $x_i$  to  $c_j$ , contains the probability of  $x_i$  to be a zero or a one considering the received value  $y_i$  as well as the arrived messages of the other connected check-nodes. The symbol-node update is depicted in the example in Figure 4.4a, where symbol-node  $x_0$  has three connected check-nodes. The updated message  $L(q_{00})$  is then computed by means of the incoming messages  $L(r_{10})$  and  $L(r_{20})$  as well as based on the SD  $L(x_0)$  of the received value  $y_0$ .



**Figure 4.4:** Example for the computations at a symbol-node.

In this step, new SDs are computed as well. These SDs contain the probabilities for all symbols to be a zero or a one, and reflect the current state of the decoding process. In contrast to Equation (4.18), all incoming messages are considered for the respective symbol-node.

$$L(Q_i) = L(x_i) + \sum_{j \in C_i} L(r_{ji}). \quad (4.19)$$

This step is visualized in Figure 4.4b.

**4<sup>th</sup> step: Hard-decision**

Based on the SD  $L(Q_i)$ , a decision is made for each symbol to be a zero or a one, which is called hard-decision (HD). The formula for the HD is

$$\hat{x}_i = \begin{cases} 0 & \text{if } L(Q_i) > 0; \\ 1 & \text{else.} \end{cases} \quad (4.20)$$

$\hat{\mathbf{x}} = (\hat{x}_0, \hat{x}_1, \dots, \hat{x}_{n-1})$  is the current estimation of the sent codeword  $\mathbf{x}$ .

**5<sup>th</sup> step: Validation**

After each decoding-iteration  $r$ , the estimated codeword  $\hat{\mathbf{x}}$  is checked for being a valid codeword of the code  $\mathcal{C}$ . This is done as shown in Algorithm 1 based on Equation (4.6).

**Algorithm 1** Validation

---

```

1: if  $\hat{\mathbf{x}}\mathbf{H}^T == \mathbf{0}$  then                                     ▷ Valid codeword found
2:   Output  $\hat{\mathbf{x}}$ ;
3:   Terminate decoding;
4: else
5:   if  $r == r_{max}$  then                                       ▷ Decoding failed
6:     Output  $\hat{\mathbf{x}} \neq \mathbf{x}$ ;
7:     Terminate decoding;
8:   else                                                         ▷ Continue with next iteration
9:     Continue with  $2^{nd}$  step;
10:  end if
11: end if

```

---

The decoding stops if either a valid codeword is found, or if a predefined number of iterations  $r_{max}$  has been processed.

**4.4.2 MS decoder**

The min-sum (MS) algorithm [41] is a famous example for a low-complexity variant of the BP decoder. The complexity when computing the check-node update ( $2^{nd}$  step in Section 4.4.1) is thereby reduced to the cost of some loss in performance.

The five steps for the MS decoder are exactly the same as for the BP decoder in Section 4.4.1 except for the  $2^{nd}$  step. This step changes to

$$L(r_{ji}) = \prod_{i' \in X_j \setminus i} \alpha_{i'j} \cdot \min_{i' \in X_j \setminus i} \beta_{i'j}. \quad (4.21)$$

The reason for that is the assumption that the smallest  $\beta_{i'j}$  in the sum of Equation (4.16) dominates, and thus the following approximation is done.

$$\phi \left( \sum_{i' \in X_j \setminus i} \phi(\beta_{i'j}) \right) \simeq \phi \left( \phi \left( \min_{i' \in X_j \setminus i} \beta_{i'j} \right) \right). \quad (4.22)$$

Next to the advantage of a lower complexity, another convenient fact is that in the case of an additive white Gaussian noise (AWGN) channel there is no need to compute the LLR according to Equation (3.10). Instead one can just take the received value  $y_i$  so that

$$L(x_i) = y_i. \quad (4.23)$$

Hence an estimation of the variance  $\sigma^2$  of the Gaussian noise is not necessary anymore.

#### 4.4.3 MSc decoder

Another approximation of the BP decoder in Section 4.4.1 is given by the MS algorithm including a correction factor  $c_f$ . The check-node update is similar to the one of the MS decoder in Section 4.4.2, except that it adds a correction factor each time a min-operation of two LLRs is processed. The correction factor  $c_f$  is thereby dependent on the absolute values of the two LLRs. Thus a nested computation is required if more than two incoming messages have to be considered. This is described in Algorithm 2 by means of a temporary variable  $tmp$ .  $i'_0, \dots, i'_{d_{c_j}-2}$  thereby indicate the incoming messages excluding the message from  $x_i$  to which the new message will be sent.  $d_{c_j}$  is the degree of check-node  $c_j$  i.e. the number of connected symbol-nodes to check-node  $c_j$ .

---

#### Algorithm 2 Check-node update

---

```

1: tmp = L(q_{i'_0});                                ▷ Initialize tmp
2: for i' = i'_1 → i'_{d_{c_j}-2} do                ▷ Start nested computation
3:   tmp = sign[tmp] sign[L(q_{i'_j})] [min(|tmp|, |L(q_{i'_j})|) + c(|tmp|, |L(q_{i'_j})|)];
4: end for
5: L(r_{ji}) = tmp;                                ▷ Assign current message

```

---

The correction term is determined by

$$c(x, y) = \begin{cases} +c_f & \text{if } |x + y| < 2 \text{ and } |x - y| > 2|x + y|; \\ -c_f & \text{if } |x - y| < 2 \text{ and } |x + y| > 2|x - y|; \\ 0 & \text{else.} \end{cases} \quad (4.24)$$

The correction factor  $c_f$  is usually taken as an argument to maximize the decoding performance and is typically  $c_f \approx 0.5$  [30].

## 4.5 Design and construction of LDPC codes

Contrary to most of the other channel-codes, there are various methods available in order to construct a LDPC code. In general, design techniques exist that yield PCMs possessing either very little structure or more structure. The advantage of the latter is the possibility of a lower complexity considering the encoding and decoding.

### 4.5.1 Cycles and girth

The decoding performance of a LDPC code is highly dependent on the cycles a code exhibits. A cycle is a closed path of consecutive edges in the Tanner graph that connects a node with itself. The number of involved edges defines the length of a cycle. The shortest possible cycle is a 4-cycle that involves four edges. This is equivalent to two columns in the PCM having two rows with ones in common. For each symbol-node  $x_i$  in a Tanner graph, the length of the shortest cycle passing through this symbol-node is denoted as local girth  $g_{x_i}$ . Global girth  $g$  is defined by the length of the shortest cycle that exists in a Tanner graph.

$$g = \min_i g_{x_i}. \quad (4.25)$$

Especially short cycles (and thus a low global girth) have a harmful impact on the decoding performance. The reason for that is explained in the following. If one iteration of the BP algorithm in Section 4.4.1 (or an approximation of it) has been processed, the symbol-nodes received extrinsic information from other symbol-nodes that share the same parity-checks (i.e. that are connected to the same check-nodes). If a valid codeword has not been found yet, another iteration will be examined (see Algorithm 1). During the second iteration, the symbol-nodes receive new information (extrinsic information) that has covered a distance of four edges and hence passed two check-nodes. This way a symbol-node can improve and refine its estimate about being a one or a zero based on the information of symbol-nodes that are four edges away. This means that the "information reach" of a symbol-node is extended by two edges with each iteration.

In the case of a cycle in the Tanner graph, a symbol-node is affected by its own message sent a few iterations before. This decreases the amount of extrinsic information and degrades the decoding performance in case of erroneous estimates. Due to the information reach doubling with processed iterations, such a feedback loop would take effect after  $g_{x_i}/2$  iterations. This means that the higher the global girth  $g$  of a LDPC code, the more decoding iterations can be processed without the degrading feedback loop effect.

Due to the harmful impact of short cycles on the decoding performance, one obvious target when designing a LDPC code is to maximize the global girth. This is, for example, done by using the progressive-edge-growth (PEG) algorithm. At least the row-column (RC) constraint is usually considered when constructing a LDPC code. It claims that neither two rows nor two columns shall have more than one position in common that contains a one.

#### 4.5.2 Computer-based design, finite-geometry codes and codes based on finite fields

The original LDPC codes developed by Gallager [26], also called Gallager-codes, are constructed by means of a computer-based design. The same is true for the codes proposed by MacKay and Neal in 1995 [28]. Both of these constructions yield PCMs with very little structure. More structure is possible based on the PEG algorithm introduced in [37] and the approximate cycle EMD (ACE) algorithm<sup>2</sup> proposed in [42] that both optimize properties of the resulting Tanner graph. Other computer-based designs that yield more structured LDPC codes include protograph-based constructions [43], accumulator-based methods, as well as generalized LDPC codes [33] [44]. A class of accumulator-based LDPC codes called irregular repeat-accumulate (IRA) LDPC codes can be found in the digital video broadcasting - satellite - second generation (DVB-S2) standard in [45].

Another sort of design approach is based on finite mathematics. Finite geometry (FG) LDPC codes, for example, are codes constructed by means of Euclidean or projective geometries. The first type of these codes was introduced in 2000 [46]. They are often used in conjunction with HD-based bit-flipping (BF) decoders [26] due to the effective trade-off between decoding performance, complexity and speed [47]. FG LDPC codes, as well as LDPC codes constructed based on finite fields often lead to

<sup>2</sup>Extrinsic message degree (EMD).

cyclic or quasi-cyclic (QC) codes where the PCM consists of sparse circulants. They enable the use of shift-registers for encoding with linear complexity [48].

Some construction methods, like the PEG algorithm, are initialized by the degree distributions (the SNDD and/or the CNDD, see Section 4.1). In such a case, the design is divided into two subsequent steps: the design of the degree distributions and based on that the construction of the PCM. In the case of the PEG method that is used later in this thesis, only the SNDD is required and subject to optimization.

### 4.5.3 Density evolution

One well-known tool that is available for the design of degree distributions is *density evolution*. By means of density evolution, it is possible to compute the performance of cycle-free codes for a given signal-to-noise ratio (SNR). This is done based on the messages passed around during the iterative decoding. The messages are thereby modeled as random variables, and the density evolution algorithm evaluates the evolution of the variable's probability density functions (PDFs), considering a specific channel. Since long LDPC codes can be viewed as being cycle-free, density evolution can be used to compute the average performance of a LDPC code ensemble. Based on the concentration theorem (see Section 4.2), it is possible to infer from the average performance of the LDPC code ensemble the performance of a LDPC code constructed based on the ensembles degree distributions. The possibility of designing good irregular LDPC codes based on density evolution is shown in [49] and [50].

Although density evolution yields very good results in lots of cases, the assumption of a cycle-free code gives rise to the suspicion that density evolution is not a suitable tool for the design of LDPC codes with short block length. This is due to the fact that the shorter the LDPC code, the more cycles occur. In addition, for short LDPC codes the length of the cycles is short with respect to the decoding iterations required on average, which leads to a harmful impact on the decoding performance. The gap between the real performance of a LDPC code and the predicted performance based on density-evolution is inversely proportional with the block length [51]. Furthermore, the concentration theorem does not hold for short LDPC codes as shown in [52].

Another tool for determining the performance of a LDPC code ensemble is given by extrinsic-information-transfer (EXIT) charts. It is based on the same principal as density evolution. An EXIT chart provides a graphical interpretation of the extrinsic information exchanged by the symbol-nodes and check-nodes during the iterative decoding [53].

More details about density evolution, EXIT charts and other techniques can be found in [30], [54] and [55].

### 4.5.4 Progressive-edge-growth (PEG) algorithm

The PEG algorithm is an optimization procedure by which the global girth  $g$  of a LDPC code is optimized. The algorithm is initialized by the SNDD, the block length  $n$  and the code rate  $R$ . It was developed by Hu, Eleftheriou and Arnold in 2005 [37]. The name is derived from the characteristics of the algorithm that causes a Tanner graph to grow by adding one edge after the other until the Tanner graph is completed. If it is unavoidable that a cycle arises when connecting a check-node  $c_j$  to a symbol-node  $x_i$  by means of an edge  $edge(x_i, c_j)$ , the length of the resulting cycle is maximized. Considering this for all edge-placements, the local girth  $g_{x_i}$  is maximized for all symbol-nodes and resulting from Equation (4.25) the global girth is maximized as well.

Before the algorithm can start, one has to know how many check-nodes to connect to each symbol-node. By multiplying the coefficients  $\lambda_d$  of the SNDD (see Equation

(4.8)) by the block length  $n$ , the number of symbol-nodes that have  $d$  adjacent edges is obtained. The number of edges connected to a symbol-node  $x_i$  is then denoted as the degree  $d_{x_i}$  of  $x_i$ . The degree sequence  $D_x = \{d_{x_0}, d_{x_1}, \dots, d_{x_{n-1}}\}$  then contains the degrees of all symbol-nodes, and is sorted so that  $d_{x_0} \leq d_{x_1} \leq \dots \leq d_{x_{n-1}}$ . The number of check-nodes is determined by means of the code rate  $R$  and Equation (4.1). The PEG algorithm starts with  $x_0$  and connects  $d_{x_0}$  edges to  $x_0$ , then continues with  $x_1$  and so on.

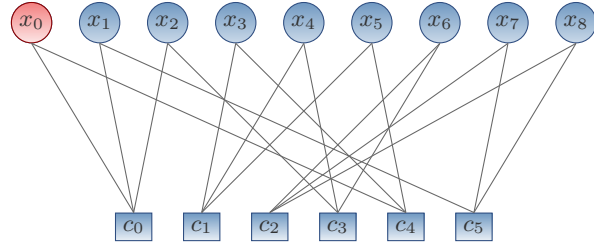
The length of a resulting cycle that may occur is thereby dependent on the decision which check-node  $c_j$  to connect to the current symbol-node  $x_i$ . The longer the shortest path connecting  $x_i$  with  $c_j$  is, the longer the resulting cycle will be. Thus the check-node with the greatest distance  $d(x_i, c_j)$  (in terms of edges) to  $x_i$  is chosen. This is done based on a tree that can be considered as an unfolding of the current Tanner graph.

Such a tree (in the following called PEG-tree) is constructed with  $x_i$  being the root of the tree (layer  $l = 0$ ). Then all neighboring check-nodes that are connected to  $x_i$  in the Tanner graph are added to the next layer  $l = 1$  and connected to the root. The next step is to start from each node in the bottom layer and follow the edges from these nodes in the Tanner graph to their neighboring nodes. If the neighboring nodes have not yet been added to the tree, they are added to a new layer at the bottom. This step is repeated until there are no more nodes that can be connected to the PEG-tree. The distance between the root node  $x_i$  and a check-node  $c_j$  is then  $d(x_i, c_j) = l_{c_j}$  with  $l_{c_j}$  being the layer of the PEG-tree in which  $c_j$  can be found. Figure 4.5b shows a PEG-tree that was constructed based on the Tanner graph in Figure 4.5a with  $x_0$  being the root of the PEG-tree.

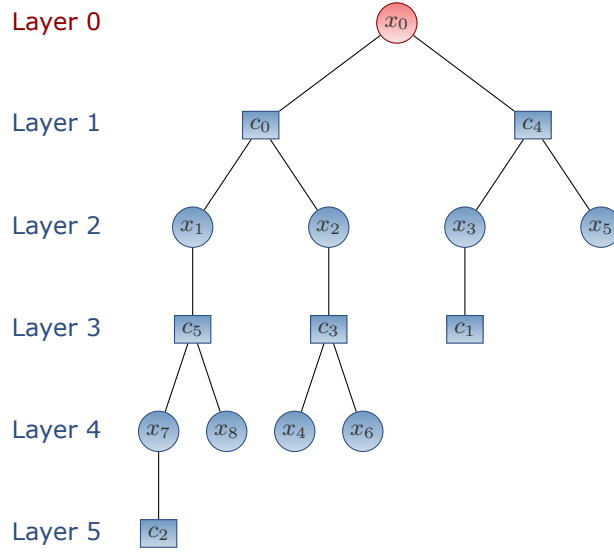
Taking the example in Figure 4.5, one should choose check-node  $c_2$  in case another check-node might be connected to  $x_0$ . This is because  $c_2$  has the maximum distance  $d(x_0, c_2) = 5$  to  $x_0$ .

The PEG algorithm is shown in Algorithm 3.  $d_c^{min}$  is thereby the minimum check-node degree (adjacent edges to a check-node) in the current Tanner graph. If there is a check-node, that has not been connected to any symbol-node so far then  $d_c^{min} = 0$ .  $S_{d_c^{min}}$  is the set of check-nodes that have the same number  $d_c^{min}$  of adjacent edges.  $d_c^{min}(l)$  denotes the minimum check-node degree of the check-nodes that are part of layer  $l$  of the PEG-tree. Thus  $d_c^{min}(l_{max})$  denotes  $d_c^{min}$  only considering the check-nodes in the bottom layer  $l = l_{max}$ . The set of check-nodes in layer  $l_{max}$  that share degree  $d_c^{min}(l_{max})$  form the set  $S_{d_c^{min}(l_{max})}$ . The set of edges connected to a symbol-node  $x_i$  is denoted as  $\mathcal{E}_{x_i}$ , with  $\mathcal{E}_{x_i}^k$  being the  $k^{th}$  edge connected to  $x_i$ .

To enable linear time encoding, one has to construct the PCM according to Equation (4.13). To construct the sub matrix  $H_{m \times m}^p$  (Equation (4.13b)), Algorithm 4 is utilized. The second sub matrix  $H_{m \times k}^u$  in Equation (4.13a) is then constructed by means of Algorithm 3, initialized with  $i = m$ .



(a) Tanner graph



(b) PEG-tree

**Figure 4.5:** Creation of a PEG-tree out of a Tanner graph with  $x_0$  being the root of the tree.**Algorithm 4** PEG algorithm for  $H_{m \times m}^p$  (see Equation (4.13b))

---

```

1: for  $i = 0 \rightarrow m - 1$  do                                ▷ Consider the first  $m$  symbol-nodes
2:   for  $k = 0 \rightarrow d_{x_i} - 1$  do                        ▷ Connect  $d_{x_i}$  edges to  $x_i$ 
3:     if  $k == 0$  then                                    ▷  $1^{st}$  edge is part of the diagonal
4:        $\mathcal{E}_{x_i}^0 \leftarrow \text{edge}(x_i, c_i)$ ;
5:     else
6:       if  $k == 1$  then                                  ▷  $2^{nd}$  edge is part of a zigzag pattern
7:          $\mathcal{E}_{x_i}^1 \leftarrow \text{edge}(x_i, c_{i+1})$ ;
8:       else                                             ▷ Choose check-node from bottom layer of the PEG-tree
9:         Build PEG-tree;
10:         $\mathcal{E}_{x_i}^k \leftarrow \text{edge}(x_i, c_j)$  with  $c_j \in \mathcal{S}_{d_c^{min}(l_{max})}$ ;
11:       end if
12:     end if
13:   end for
14: end for

```

---

**Algorithm 3** PEG algorithm

---

```

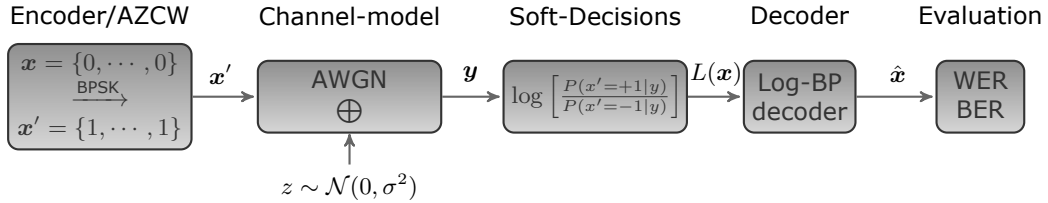
1: for  $i = 0 \rightarrow n - 1$  do ▷ Consider  $n$  symbol-nodes
2:   for  $k = 0 \rightarrow d_{x_i} - 1$  do ▷ Connect  $d_{x_i}$  edges to  $x_i$ 
3:     if  $k == 0$  then ▷ 1st edge to connect to  $x_i$ 
4:        $\mathcal{E}_{x_i}^0 \leftarrow \text{edge}(x_i, c_j)$  with  $c_j \in \mathcal{S}_{d_c^{min}};$ 
5:     else
6:       if  $d_c^{min} == 0$  then ▷ Still unconnected check-nodes
7:          $\mathcal{E}_{x_i}^k \leftarrow \text{edge}(x_i, c_j)$  with  $c_j \in \mathcal{S}_{d_c^{min}};$ 
8:       else ▷ Choose check-node from bottom layer of the PEG-tree
9:         Build PEG-tree;
10:         $\mathcal{E}_{x_i}^k \leftarrow \text{edge}(x_i, c_j)$  with  $c_j \in \mathcal{S}_{d_c^{min}(l_{max})};$ 
11:       end if
12:     end if
13:   end for
14: end for

```

---

**4.6 Evaluation of LDPC codes**

To evaluate the decoding performance of a specific LDPC code, a simulation is processed. This is done by means of a LDPC encoder, a desired channel-model and a LDPC decoder. The LDPC encoder is only necessary with an asymmetric channel-model, otherwise one can skip the encoding by using the AZCW which is always valid for linear block codes. Figure 4.6 depicts a simulation where the AZCW, an AWGN channel and the log-domain BP decoder of Section 4.4.1 are used.



**Figure 4.6:** Simulation in order to evaluate a LDPC code.

For most of the simulations in this thesis, a binary phase-shift keying (BPSK) modulation is utilized according to Equation (3.1). The channel-model is chosen with respect to the real channel that the LDPC code has to face considering the desired application. In the case of SD decoding, the LLRs are then computed based on the channel-output  $y$  according to Equation (3.9). Then the desired LDPC decoder is applied that outputs the estimated codeword  $\hat{x}$ . The decoding performance is then measured by computing the bit error ratio (BER) and the word error ratio (WER). The BER counts the bit errors relative to the total number of bits in the code word and is computed by Equation (4.26). The bit errors  $e_b$  are computed by

$$e_b = \sum_{i=0}^{n-1} (x_i + \hat{x}_i \pmod{2}). \quad (4.26a)$$



Due to the random characteristic of the channel-model's noise, a simulation is repeated  $r_{max}$  times and the total bit errors are calculated by

$$e_b^{total} = \sum_{r=1}^{r_{max}} e_b^r. \quad (4.26b)$$

Then an average BER is computed according to

$$BER = \frac{e_b^{total}}{n \cdot r_{max}} \quad (4.26c)$$

with  $n$  being the number of bits in a code word. The WER counts the number of code word errors (unsuccessful decodings) relative to the total number of decodings (see Algorithm 5).

---

**Algorithm 5** WER computation
 

---

```

1:  $e_w = 0$ ;
2: for  $r = 1 \rightarrow r_{max}$  do
3:   if  $e_b^r$  then                                ▷ if there is at least one bit error
4:      $e_w = e_w + 1$                                 ▷ increment the word errors
5:   end if
6: end for
7:  $e_w^{total} = e_w$ ;                                ▷ total number of word-errors
8:  $WER = e_w^{total} / r_{max}$ ;

```

---

A typical BER and WER curve is obtained by running several simulations for several SNRs, respectively. Considering channel-coding, the SNR is usually expressed by  $E_b/N_0$ .  $E_b$  is the energy per information bit and is computed by

$$E_b = \frac{\mathbb{E}[x_i'^2]}{R} \quad (4.27)$$

with  $\mathbb{E}[x_i'^2]$  being the expected value of  $x_i'^2$ . In the case of BPSK modulation, Equation 4.27 yields  $E_b = 1/R$ .  $N_0$  is the spectral noise density that for the AWGN is calculated according to

$$N_0 = 2\sigma^2. \quad (4.28)$$

The variance  $\sigma^2$  of a AWGN channel is then computed depending on the  $E_b/N_0$  value in dB following from Equation (4.27) and (4.28) by

$$\sigma^2 = \frac{1}{10^{((E_b/N_0)_{dB})/10} \cdot 2R}. \quad (4.29)$$

In the case of a Markov-modulated Gaussian channel (MMGC), the variance computed by Equation (4.29) represents the average variance of the involved sub-channels. The variances of the several AWGN channels are then obtained based on the time that the channel-model is in the according states representing the sub-channels. For a 2-state MMGC, the times are computed based on Equation (3.4) and (3.5). The two standard deviations  $\sigma_1$  and  $\sigma_2$  are then calculated according to

$$\sigma_1 = \sqrt{\frac{\sigma^2}{t_1 + t_2 d_\sigma^2}} \quad (4.30)$$

and

$$\sigma_2 = d_\sigma \cdot \sigma_1. \quad (4.31)$$

When the variance in the case of an AWGN channel or the two sub-variances in the case of a MMGC are calculated, the noise can be added to the code word according to the appropriate channel-model. For representing the results of a simulation, one can then take advantage of the knowledge of the sent code word in order to compute the real variance produced by the applied channel-model. Based on Equation (4.29), one can then compute the  $E_b/N_0|_{dB}$  value to plot the determined BER and WER over the correct SNR values.

For all simulations conducted for this thesis, a minimum of  $e_b^{total} = 200$  bit errors and a minimum of  $e_w^{total} = 100$  word errors was ensured. This is especially important for higher  $E_b/N_0|_{dB}$  values with very low BERs and WERs. Every BER and WER computation is based on a minimum of  $r_{max} = 10^4$  repetitions.

## Chapter 5

# Design of short irregular LDPC codes

In Section 4.5 a brief overview of common design and construction methods for irregular low-density parity-check (LDPC) codes is given. They mainly differ in the amount of structure of the constructed parity-check matrix (PCM) that defines the LDPC code. As already mentioned in Section 4.5.3, the design of LDPC codes with short block length is not comparable to the design of LDPC codes in general. This is mainly due to the fact that for short LDPC codes more cycles occur that are short with respect to the required decoding iterations.

When considering the application of LDPC codes on two dimensional (2D) barcodes, one has to deal with very short block length. The smallest Data Matrix code (DMC) according to standard [4] for example has a data region of size  $8 \times 8$  in which a codeword of maximum length  $n = 64$  would fit. However, due to the required amount of information to be encoded, greater sizes are used in most scenarios. These sizes are still very small in terms of the resulting LDPC code's block length. Another point to consider is the flexibility in terms of the code rate  $R$ , depending on the desired security level of the 2D barcode. Furthermore, the length of the LDPC code is dependent on the size of the 2D barcode's data region. Thus a design method has to be chosen that is suitable for short length LDPC codes, and that provides the possibility of defining the block length as well as the code rate.

In [37] the authors prove the effectiveness of the progressive-edge-growth (PEG) algorithm for the PCM construction considering short LDPC codes. In conjunction with an optimal symbol-node degree distribution (SNDD), their PEG method yields the best short LDPC codes at the time of their publication. The PEG optimization that is described in Section 4.5.4 has been chosen in the context of this thesis due to its effectiveness for short block length LDPC codes, and due to the fact that based on a given SNDD a LDPC code can be constructed for any block length and code rate.

Since the design of the pair of degree distributions (the SNDD and the check-node degree distribution (CNDD)) based on density evolution is not suitable for short LDPC codes (see Section 4.5.3), the authors in [37] propose a design using a simplified version of the downhill simplex (DHS) algorithm. The DHS algorithm is also known as Nelder-Mead algorithm with Nelder and Mead being the authors that first proposed the search method for multidimensional unconstrained nonlinear problems [56]. This direct search method involves direct evaluations of the function itself, and is therefore suitable in cases where derivations of the function are not computable. In [37] a reduced version of the DHS algorithm is used in order to design the SNDD which is then utilized to construct the PCM by means of the PEG algorithm.

## 5.1 Downhill simplex based design

In this section, a design method for irregular LDPC codes is developed that is based on the functionality that only the complete DHS algorithm provides. The development is done in order to design short irregular LDPC codes that have a lower error-rate than the codes designed by means of well tried optimization methods, like the one in [37].

The developed algorithm is based on a simplex

$$\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_N, \lambda_{N+1}\} \quad (5.1)$$

that moves through a  $N$ -dimensional space  $\mathbf{R}^N$ . For the sake of simplicity, Equation (4.8) is rewritten to start the sum from  $j = 1$ .

$$\lambda(x) = \sum_{j=1}^N \lambda_j x^{d_j} \quad (5.2)$$

with  $\{d_j\}_{j=1}^N = \{2, 3, \dots, d_x^{max}\}$  and  $N = d_x^{max} - 1$ . So the symbol-nodes have at least  $d_1 = 2$  connected edges and a maximum of  $d_N = N + 1$  adjacent edges. Each vertex  $\lambda_i$  in the simplex  $\Lambda$  represents a unique SNDD, and thus consists of  $N$  values  $\{\lambda_{i,j}\}_{j=1}^N$  referring to the fractions of symbol-nodes having  $d_j$  adjacent edges. During the code-design process, the vertices are constantly sorted according to their function evaluations so that

$$f(\lambda_1) \leq f(\lambda_2) \leq \dots \leq f(\lambda_N) \leq f(\lambda_{N+1}). \quad (5.3)$$

The function evaluation is thereby represented by the computation of the word error ratio (WER) (see Section 5.1.2).  $\lambda_1$  is then called the best vertex and  $\lambda_{N+1}$  the worst vertex. The iterative algorithm always tries to replace the worst vertex by a better one. The search for a better vertex is done based on an operation called *Reflection* which is computed by

$$\lambda_r = \bar{\lambda}' + \alpha(\bar{\lambda}' - \lambda_{N+1}). \quad (5.4)$$

$\alpha$  is usually set to  $\alpha = 1$  and  $\bar{\lambda}'$  is the centroid of the simplex on which the worst vertex is reflected. It is computed without considering the worst vertex according to

$$\bar{\lambda}' = \frac{1}{N} \sum_{i=1}^N \lambda_i. \quad (5.5)$$

In contrast to [37], where only the reflection operation is used, one of the following four operations is processed, depending on the WER  $f(\lambda_r)$  of the reflected vertex  $\lambda_r$ .

*Outward Contraction:*

$$\lambda_{oc} = \bar{\lambda}' + \beta(\bar{\lambda}' - \lambda_{N+1}); \quad (\beta = 0.5) \quad (5.6)$$

*Reduction:*

$$\lambda_{new} = \lambda_1 + \sigma(\lambda_i - \lambda_1) \quad \forall i \setminus 1; (\sigma = 0.5) \quad (5.7)$$

*Inward Contraction:*

$$\lambda_{ic} = \lambda_{N+1} + \beta(\bar{\lambda}' - \lambda_{N+1}); \quad (\beta = 0.5) \quad (5.8)$$

The computation of the *Expansion* operation is based on Equation (5.6) with  $\beta = 2$ . The chart in Figure 5.1 visualizes the whole DHS algorithm, and shows when to use which operation.

After the worst vertex is replaced, a termination criterion is checked. It has been defined as a threshold for the average distance  $r_{av}$  of the vertices to the centroid of the simplex  $\bar{\lambda}$ .

$$r_{av} = \frac{1}{N+1} \sum_{i=1}^{N+1} \sqrt{\sum_{j=1}^N (\lambda_{i,j} - \bar{\lambda}_j)^2}. \quad (5.9)$$

$\lambda_{i,j}$  denotes the value of the vertex  $\lambda_i$  in the  $j$ -th dimension. In contrast to Equation (5.5), the centroid  $\bar{\lambda}$  of the whole simplex is computed considering the worst vertex as well according to

$$\bar{\lambda} = \frac{1}{N+1} \sum_{i=1}^{N+1} \lambda_i. \quad (5.10)$$

If the termination criterion does not hold, the process continues by entering the next iteration that starts with the *Reflection* again. So the range of the  $N$  coefficients in Equation (5.2) spans a  $N$ -dimensional space  $\mathbf{R}^N$ . For all possible points in  $\mathbf{R}^N$ , there exist WER-values in the  $N+1$ -th dimension that form a surface on which the simplex moves towards a minimum. This is only true if the following constraints are not considered.

### 5.1.1 Constraints

The SNDD optimization is constrained by

$$\sum_{j=1}^N \lambda_j = 1. \quad (5.11)$$

From Equation (5.11) it follows

$$0 < \lambda_j < 1 \quad \forall j \setminus N, \quad (5.12)$$

which is the first inequality constraint. During the optimization, the  $N$ -th parameter is computed by

$$\lambda_N = 1 - \sum_{j=1}^{N-1} \lambda_j. \quad (5.13)$$

The second inequality constraint follows from Equation (5.11) and (5.13) according to

$$0 < \sum_{j=1}^{N-1} \lambda_j < 1. \quad (5.14)$$

To adapt the DHS optimization to the constrained design of SNDDs, we use Algorithm 6 and 7. The first constraint of Equation (5.12) is respected by use of the procedure in Algorithm 6 every time a new vertex is computed.

---

**Algorithm 6** Ensure 1<sup>st</sup> constraint
 

---

```

1: procedure EnsureConstraint1( $\lambda_j$ )
2:   while  $\lambda_j \geq 1$  do
3:      $\lambda_j = \lambda_j - \delta$   $\triangleright \delta = 10^{-5}$ 
4:   end while
5:   return  $\lambda_j$ 
6: end procedure

```

---

The procedure in Algorithm 7 ensures to respect the second constraint of Equation (5.14).

---

**Algorithm 7** Ensure 2<sup>nd</sup> constraint
 

---

```

1: procedure EnsureConstraint2( $\lambda_a, \lambda_b$ )
2:   while  $\sum_{j=1}^{N-1} \lambda_{a,j} \geq 1$  do
3:      $\lambda_{a_{new}} = \frac{\lambda_a + \lambda_b}{2}$ 
4:     for all  $j \setminus N$  do
5:       EnsureConstraint1( $\lambda_j$ )
6:     end for
7:   end while
8:   return  $\lambda_{a_{new}}$ 
9: end procedure

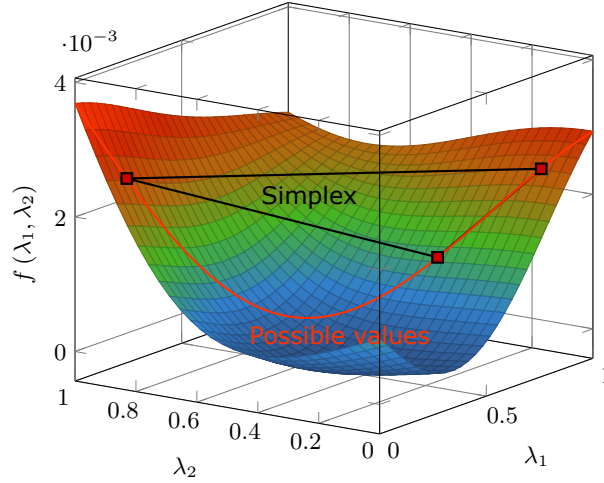
```

---

Depending on the currently processed operation, the following assignments are made to the pair of vertices  $(\lambda_a, \lambda_b)$ :

$$(\lambda_a, \lambda_b) = \begin{cases} (\lambda_r, \bar{\lambda}') & \text{for Reflection;} \\ (\lambda_e, \bar{\lambda}') & \text{for Expansion;} \\ (\lambda_{oc}, \bar{\lambda}') & \text{for OutwardContraction;} \\ (\lambda_{ic}, \lambda_{N+1}) & \text{for InwardContraction;} \\ (\lambda_{i_{new}}, \lambda_1) & \text{for Reduction.} \end{cases} \quad (5.15)$$

Due to the constraints the possible combination of the coefficients is limited. This can be seen in the example in Figure 5.2 that shows the 3-dimensional space spanned by the two coefficients  $\lambda_1$  and  $\lambda_2$ . Without considering the constraints, their would be a function evaluation  $f(\lambda_1, \lambda_2)$  for each possible combination of  $\lambda_1$  and  $\lambda_2$  represented by the surface in Figure 5.2. Due to the constraints, the surface reduces to a line which is drawn in red in the example. An example of a valid simplex is added in black.



**Figure 5.2:** Example of a downhill simplex based design with  $d_x^{max} = 3$ .

### 5.1.2 Function evaluations

Each time the simplex changes, the vertices are sorted according to Equation (5.3). This is done based on the function evaluations for each of the vertices. In the context of SNDD-optimization the function evaluation is represented by the computation of the WER. Based on the SNDD of a vertex, a PCM is created, which is done using the PEG algorithm (see Section 4.5.4). Then a simulation of the resulting LDPC code follows as explained in Section 4.6. The decoder is thereby chosen depending on the appropriate channel that the LDPC code is designed for.

### 5.1.3 Optimization process

There is no loss in computation time compared to [37], despite all the operations of the DHS algorithm. Quite the contrary, computation time per optimization round is gained by only using  $N + 1$  vertices instead of  $2(N - 1)$  vertices as in [37].

Due to the nature of the direct search mechanism, the minimum to which the DHS algorithm converges is not necessarily a global minimum. For that reason, the process explained in Algorithm 8 is utilized in order to increase the probability of a convergence to the global minimum.

---

#### Algorithm 8 Optimization process

---

- 1:  $r = 1$
  - 2: **while**  $r < 10$  **do** ▷ 9 repetitions
  - 3:   create initial simplex;
  - 4:   apply constrained DHS-algorithm;
  - 5:   store  $\lambda_{best}^r$ ;
  - 6:    $r = r + 1$ ;
  - 7: **end while**
  - 8: create initial simplex including  $\{\lambda_{best}^1, \dots, \lambda_{best}^9\}$ ;
  - 9: apply constrained DHS-algorithm;
  - 10: **return**  $\lambda_{best}^{10}$
-

The optimization process showed in Algorithm 8 consists of ten repetitions of the constrained DHS algorithm. This means that an initial simplex is created ten times.

#### 5.1.4 Initializing simplex

For the first round of the optimization process (Algorithm 8), the  $i^{th}$  vertex  $\lambda_i = \{\lambda_{i,1}, \dots, \lambda_{i,N}\}$  of the simplex  $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_N, \lambda_{N+1}\}$  is initialized as follows:

$$\lambda_{i,j} = \begin{cases} \frac{0.5 - \frac{1}{N}}{N-1} & , \forall i \setminus N+1, \forall j \setminus i; \\ 0.5 + \frac{1}{N} & , j = i; \\ random[0, b_j^{max}] & , i = N+1 \end{cases} \quad (5.16)$$

with

$$b_j^{max} = 1 - \sum_{l=1}^{j-1} \lambda_{i,l} \quad (5.17)$$

starting from  $j = 1$ . So for the first  $N$  vertices, all values are exactly the same except for one degree respectively (when  $j = i$ ) to which a bigger value is assigned. For the last of the  $N + 1$  vertices, all values are created randomly under the restriction of the constraints in Equation (5.12) and (5.14). The initializations of the next 8 start-simplexes are made based on the following assignment:

$$\lambda_{i,j} \leftarrow random[0, b_j^{max}]. \quad (5.18)$$

The initializing simplex of the last round is then created by including the best simplexes obtained from all previous optimization rounds. The remaining vertices are constructed according to Equation (5.18). At the end of the optimization process, the very best vertex is returned.



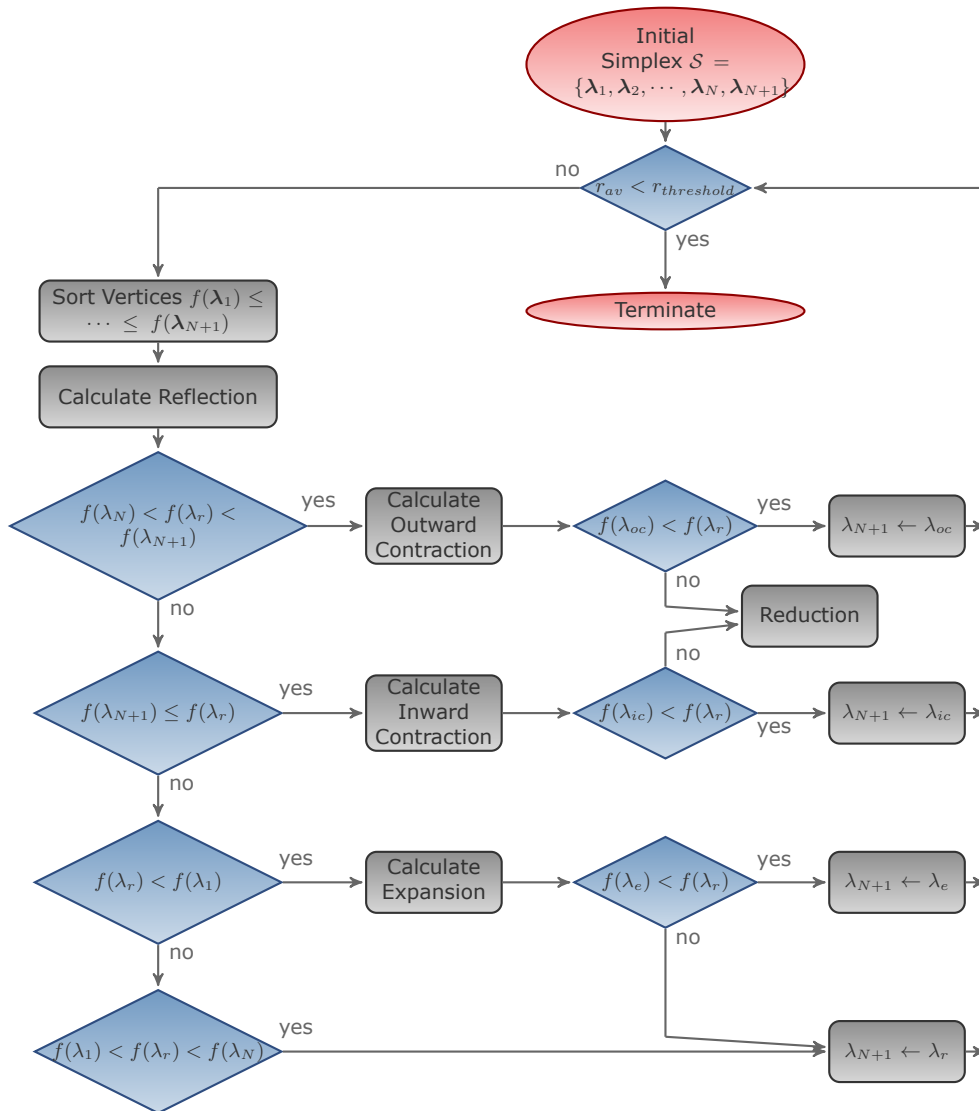


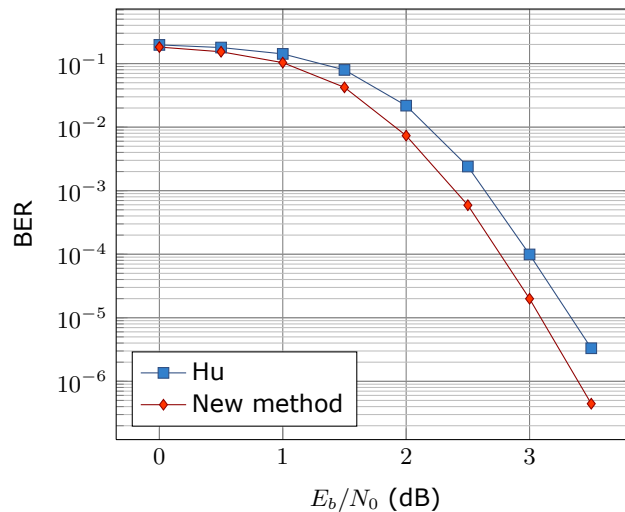
Figure 5.1: Flow-chart of the downhill simplex algorithm.

## 5.2 Design results

The developed optimization method is now to be evaluated. For that, it is compared to the design method in [37], that is based on a simplified version of the DHS algorithm. It yielded the best short irregular LDPC codes as far as the author of this thesis knows.

In order to compare the optimization method developed in this thesis with the results in [37], an irregular LDPC code was designed with the same code rate  $R = 0.5$  and block length  $n = 504$  for the additive white Gaussian noise (AWGN) channel as in [37]. The maximum symbol-node degree was set to  $d_x^{max} = 15$  as well. The resulting SNDD was  $\lambda(x) = 0.429581x^2 + 0.401542x^3 + 0.000167x^4 + 0.077138x^5 + 0.000098x^6 + 0.003617x^7 + 0.000853x^8 + 0.064487x^9 + 0.000284x^{10} + 0.000286x^{11} + 0.003472x^{12} + 0.013787x^{13} + 0.00031x^{14} + 0.004379x^{15}$ . The PEG algorithm of Section 4.5.4 was then used to construct the PCM, and a simulation (MS) was performed as explained in Section 4.6. For the decoding, the min-sum (MS) decoder (see Section 4.4.2) was established.

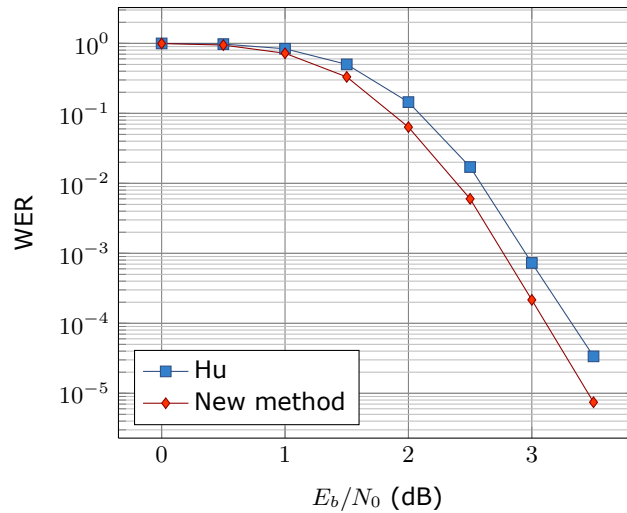
Figure 5.3 shows the resulting bit error ratio (BER) for the LDPC code designed by means of the developed design method and for the LDPC code based on the SNDD of [37].



**Figure 5.3:** BER comparison of two rate  $R = 0.5$  irregular PEG-LDPC codes of length  $n = 504$  with an AWGN channel. The two codes were designed with the method of Hu et al. [37] and the new design technique developed in this thesis.

When looking at the red curve in Figure 5.3 that refers to the new design method developed in this thesis, one can see that it is located below the blue curve of Hu et al. for the entire range of  $E_b/N_0$  values. Considering the BER, the new method beats the reference design technique with up to 0.35 dB.

The advantage of using the introduced design method can also be seen if computing the WER. This appears in Figure 5.4. Once again, the error-ratio of the new optimization technique is lower compared to the results based on the SNDD of [37]. The gain for the WER is thereby up to 0.25 dB.



**Figure 5.4:** WER comparison of two rate  $R = 0.5$  irregular PEG-LDPC codes of length  $n = 504$  with an AWGN channel. The two codes were designed with the method of Hu et al. [37] and the new design technique developed in this thesis.

Thus one can say that the new developed optimization method leads to best results when considering short irregular LDPC codes.

Another revealing result that is based on a Markov-modulated Gaussian channel (MMGC) and that proves the effectiveness of the new design method can be found in Section 6.5.4.

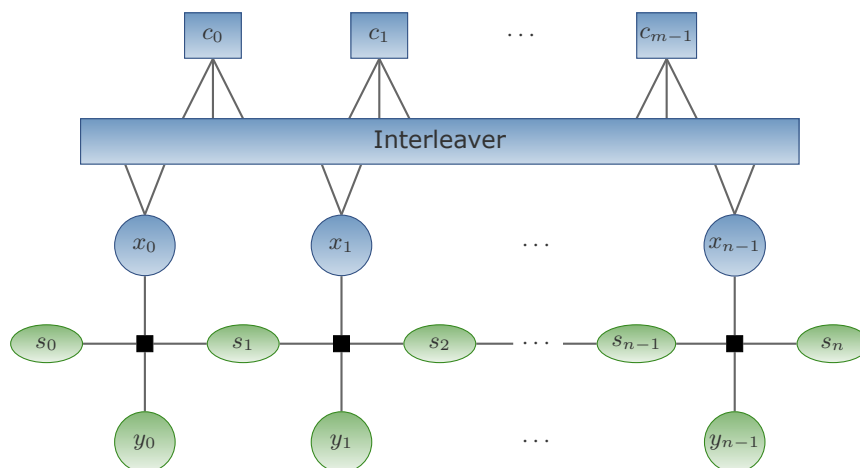


## Chapter 6

# Estimation-decoding

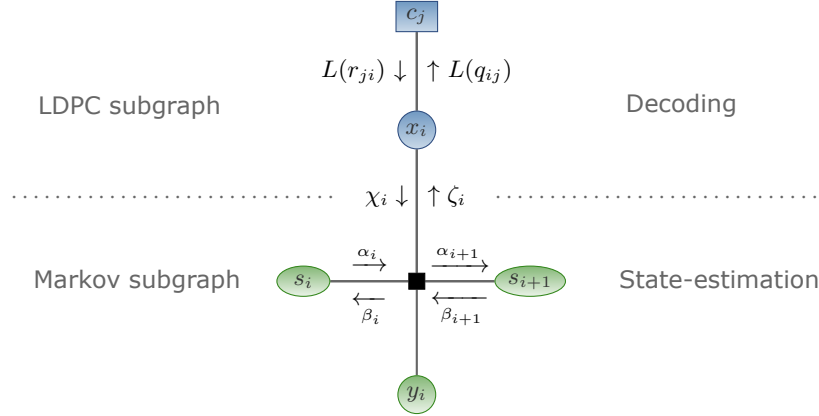
Estimation-decoding is a decoding algorithm developed for the decoding of low-density parity-check (LDPC) codes used with channels with memory like the Markov-modulated channels described in Section 3.4. In many approaches, a channel with memory is assumed to be memoryless when using a channel interleaver. When considering LDPC codes, the random-like connection of symbol-nodes with check-nodes can be interpreted as a build in interleaver. However, it has been shown [57] [58] [59] [60] that significant improvement is obtained by use of an estimation-decoding algorithm that takes the channel's memory into account.

Most decoding algorithms for LDPC codes like the decoders in Section 4.4 are based on the LDPC code's Tanner graph. The estimation-decoding algorithm extends the decoding by adding a hidden Markov model (HMM) to the Tanner graph, which results in the so called Markov-LDPC factor graph that can be seen in Figure 6.1. The Tanner graph, on which the belief propagation (BP) algorithm (or an approximation of it; see Section 4.4) operates, is then called LDPC subgraph. On the Markov subgraph (the HMM), a state-estimation is computed by use of the Forward-Backward algorithm that is similar to the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [61]. This algorithm is bit-wise connected with the decoding algorithm on the LDPC subgraph to form the estimation-decoding algorithm.



**Figure 6.1:** Markov-LDPC factor graph.

So the name *estimation-decoding* stems from the combined *state-estimation* on the Markov subgraph and the *decoding* on the LDPC subgraph. The messages of one sector of the Markov-LDPC factor graph in Figure 6.1 are depicted in Figure 6.2. The computation of the messages is explained in the following considering a 2-state channel-model as described in Section 3.4.



**Figure 6.2:** Messages in one sector of the Markov-LDPC factor graph.

## 6.1 Messages of the LDPC subgraph

The message  $L(q_{ij})$  belongs to the LDPC decoder, and is computed based on Equation (4.18). The message  $L(r_{ji})$ , sent from check-node  $c_j$  to symbol-node  $x_i$ , is calculated according to Equation (4.16) in the case of the log-domain BP decoder, or by Equation (4.21) if the min-sum (MS) approximation is used.

## 6.2 Messages of the Markov subgraph

The forward message  $\alpha_{i+1}$ , belonging to the next point in time  $i+1$ , is computed based on the forward message  $\alpha_i$  according to

$$\alpha_{i+1}(l) = \sum_{k=1}^2 p_{kl} \alpha_i(k) \sum_{x_i \in \{0,1\}} P(x_i | \chi_i) P(y_i | x_i, s_i = \mathfrak{s}_k). \quad (6.1)$$

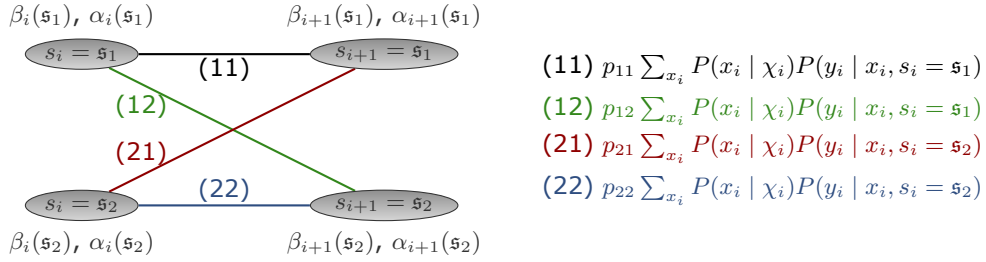
$p_{kl}$  is thereby one of the four transition probabilities of the transition probability matrix  $\mathbf{P}$  in Equation (3.3), and is described by

$$p_{kl} = P(s_{i+1} = \mathfrak{s}_l | s_i = \mathfrak{s}_k) \quad (6.2)$$

where  $\mathfrak{s}_k$  and  $\mathfrak{s}_l$  are the states at time  $i$  and  $i+1$ , respectively.  $P(y_i | x_i, s_i = \mathfrak{s}_k)$  is defined by the channel, and is computed by Equation (3.6) for the Markov-modulated Gaussian channel (MMGC). The backward message  $\beta_i$  is computed by means of the message  $\beta_{i+1}$  of the next point in time  $i+1$  by

$$\beta_i(k) = \sum_{l=1}^2 p_{kl} \beta_{i+1}(l) \sum_{x_i \in \{0,1\}} P(x_i | \chi_i) P(y_i | x_i, s_i = \mathfrak{s}_k). \quad (6.3)$$

At each point in time, there are two  $\alpha$  and two  $\beta$  messages referring to the two states. This can be seen in Figure 6.3 where one sector of the HMM that is represented by the Markov subgraph can be seen. It can be regarded as an unfolding of two neighboring state-nodes (e.g.: the state-nodes  $s_1$  and  $s_2$ ) and the black squared channel-node in between in Figure 6.2.



**Figure 6.3:** One sector of the HMM represented by the Markov subgraph.

### 6.3 The interface messages

The messages  $\chi_i$  and  $\zeta_i$  together form the interface between the LDPC subgraph and the Markov subgraph. They provide extrinsic information to the respective counterpart subgraph. The message  $\chi_i$  is equal to the soft-decision (SD) computed by Equation (4.19). Since it is usually in the log-domain, it has to be transferred to the probabilistic domain by

$$P(x_i | \chi_i) = \begin{cases} \frac{1}{2} + \frac{1}{2} \tanh \frac{\chi_i}{2} & , x_i = 0; \\ \frac{1}{2} - \frac{1}{2} \tanh \frac{\chi_i}{2} & , x_i = 1. \end{cases} \quad (6.4)$$

The channel-message  $\zeta$  that is passed from the Markov subgraph to the LDPC subgraph is computed by

$$\zeta_i = \log \frac{P(x_i = 0 | \alpha_i, \beta_{i+1})}{P(x_i = 1 | \alpha_i, \beta_{i+1})} \quad (6.5)$$

with

$$P(x_i | \alpha_i, \beta_{i+1}) = \sum_{k=1}^2 \sum_{l=1}^2 P(y_i | x_i, s_i = s_k) p_{kl} \alpha_i(k) \beta_{i+1}(l). \quad (6.6)$$

For the computation of the messages  $L(q_{ij})$ , the channel-message  $\zeta_i$  is treated like one more incoming message  $L(r_{ji})$ .

### 6.4 New variant of estimation-decoding

In [57] and [59], the authors assumed the transition probabilities  $p_{kl}$  of the corresponding MMGC to be known when applying the estimation-decoding algorithm. But this is not always the case e.g. in Section 7.3 where a decoder is developed for LDPC-based 2D barcodes.

In this thesis, a new variant of estimation-decoding is developed that prevents decoding based on false transition probabilities. This is done by means of a reestimation method by which the transition probabilities are continually reestimated in each decoding iteration considering a MMGC.

### 6.4.1 Reestimation of the transition probabilities

The newly developed reestimation method is utilized when the transition probability matrix  $\mathbf{P}$  is not known. Then one starts with initial values for  $p_{kl}$ , and continually reestimates  $p_{kl}$  during the iterative estimation-decoding process. A similar method was proposed in [58] but only considering a Gilbert-Elliott channel.

The reestimation is based on the Baum-Welch method [62], and can also be derived from the Expectation-Modification (EM) algorithm [63]. The reestimated transition probability matrix  $\hat{\mathbf{P}}_r$  is thereby calculated in each iteration  $r$ . For the computation of  $\hat{\mathbf{P}}_{r+1}$ , which is the reestimation of the previous transition probability matrix  $\hat{\mathbf{P}}_r$ , one needs to define

$$\gamma_i^r(k) = P(s_i = s_k | \mathbf{y}, \hat{\mathbf{P}}_r) \quad (6.7a)$$

which is the probability of being in state  $s_k$  at time  $i$  and in iteration  $r$ , given the received codeword  $\mathbf{y}$  and  $\hat{\mathbf{P}}_r$ .  $\gamma_i^r(k)$  can be calculated by using the messages  $\alpha$  and  $\beta$  of the Forward-Backward algorithm computed by Equation (6.1) and (6.3).

$$\gamma_i^r(k) = \frac{\alpha_i^r(k)\beta_i^r(k)}{\sum_{k=1}^2 \alpha_i^r(k)\beta_i^r(k)}. \quad (6.7b)$$

Furthermore, it is necessary to compute

$$\xi_i^r(k, l) = P(s_i = s_k, s_{i+1} = s_l | \mathbf{y}, \hat{\mathbf{P}}_r) \quad (6.8a)$$

which is the probability of being in state  $s_k$  at time  $i$  and in state  $s_l$  at the next point in time  $i+1$ , given  $\mathbf{y}$  and  $\hat{\mathbf{P}}_r$ . This is calculated as follows:

$$\xi_i^r(k, l) = \frac{\alpha_i^r(k)\hat{p}_{kl}^r\beta_{i+1}^r(l)}{\sum_{l=1}^2 \alpha_{i+1}^r(l)\beta_{i+1}^r(l)} \sum_{x_i \in \{0,1\}} P(y_i | x_i, s_i = s_k)P(x_i | \chi_i). \quad (6.8b)$$

The elements  $\hat{p}_{kl}^{r+1}$  of  $\hat{\mathbf{P}}_{r+1}$  are then obtained by

$$\hat{p}_{kl}^{r+1} = \frac{\text{expected number of transitions from } s_k \text{ to } s_l \text{ in iteration } r}{\text{expected number of transitions from } s_k \text{ in iteration } r} \quad (6.9a)$$

which is computed based on Equation (6.7b) and (6.8b) according to

$$\hat{p}_{kl}^{r+1} = \frac{\sum_{i=0}^{n-1} \xi_i^r(k, l)}{\sum_{i=0}^{n-1} \gamma_i^r(k)}. \quad (6.9b)$$

The extended estimation-decoding algorithm that includes the above reestimation-procedure is called EDEP algorithm in the following, which stands for **e**stimation-**d**ecoding and **e**stimation of the transition-**p**robabilities.

### 6.4.2 Variance estimation for the decoding on a 2-state MMGC

Although the estimation of  $\sigma^2$  could be avoided due to Equation (4.23) when applying the MS decoder on the LDPC subgraph, it is necessary to estimate the variance for the Forward-Backward algorithm that operates on the Markov subgraph. The computation of  $\alpha$  and  $\beta$  (Equation (6.1) and (6.3)) is based on the channel likelihoods  $P(y_i | x_i, s_i = s_k)$  calculated by Equation (3.6) for which the variances  $\sigma_1^2$  and  $\sigma_2^2$  of the two sub-channels are required.  $\sigma_1$  and  $\sigma_2$  are derived from the average variance  $\sigma^2$  based on Equation (4.30) and (4.31), respectively. However, the average variance  $\sigma^2$  has to be estimated by means of the received data word  $\mathbf{y}$ . This is done based on

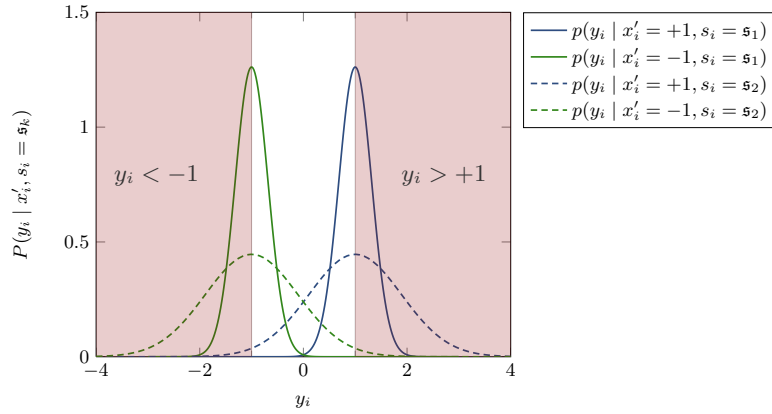


$$\sigma_{est}^2 = \frac{1}{n_t - 1} \left[ \sum_{y_i \in \mathbf{y}: y_i > 1} (y_i - 1)^2 + \sum_{y_i \in \mathbf{y}: y_i < -1} (y_i + 1)^2 \right] \quad (6.10)$$

with

$$n_t = |\{y_i \in \mathbf{y} : y_i < -1 \vee y_i > +1\}|. \quad (6.11)$$

So  $n_t$  represents the number of received values that are greater than 1 or less than  $-1$ . Only these values are used for the unbiased estimation  $\sigma_{est}^2$  of  $\sigma^2$ . Figure 6.4 shows the range for  $y_i$  that is used for the estimation. This selection is done since for values of  $y_i$  for that  $-1 \leq y_i \leq 1$  is true, the probability of an error when assigning  $y_i$  to  $x'_i = 1$  or  $x'_i = -1$  is high. In contrast to that, the probability of a mismatched assignment is very low for the remaining values of  $y_i$ .



**Figure 6.4:** Distributions of the received values  $y_i$ . The red marked area is the one considered for estimating the variance based on the received data word  $\mathbf{y}$ .

## 6.5 Analysis and Results

In this analysis, the following points are investigated:

1. The effectiveness of the state-estimation by adding the Markov subgraph to the Tanner graph.
2. The effectiveness of the new estimation-decoding variant that includes the reestimation-procedure of the transition probabilities.
3. The behavior of irregular LDPC codes in conjunction with a MMGC and estimation-decoding.

The channel-model of interest is thereby a MMGC and is parameterized by  $p_{12} = 0.3$ ,  $p_{21} = 0.6$  and  $\sigma_2 = 5 \cdot \sigma_1$ . All LDPC codes that are used for the following tests share the code rate  $R = 0.608^1$  and block length of  $n = 576^2$ .

<sup>1</sup>The MMGC and the code rate were chosen for comparison purposes with results of [57] in Section 6.5.4.

<sup>2</sup>The block length was chosen with respect to the later application of LDPC codes on 2D barcodes with a data region size of  $24 \times 24$  that provides space for 576 bits.

### 6.5.1 Effectiveness of the state-estimation

In this section, the effectiveness of adding the Markov subgraph to the Tanner graph and thus taking the channel's memory into account, is analyzed. For that, a simulation according to Section 4.6 is performed with a regular LDPC code. The code words are disturbed according to the MMGC, and the received data words are then decoded with two different decoders:

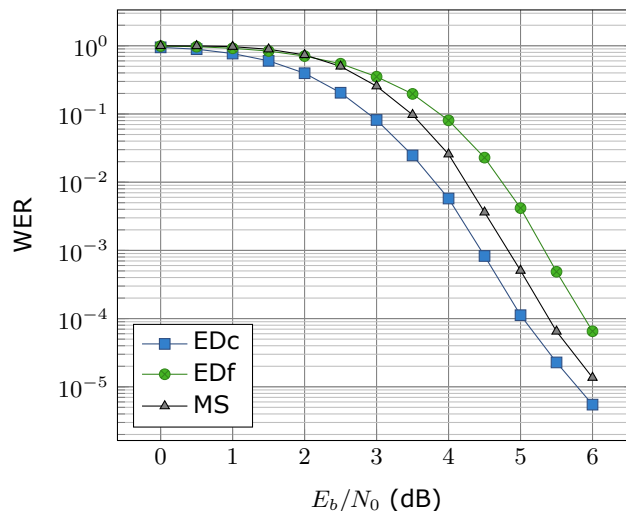
1. The MS decoder of Section 4.4.2 (labeled with MS).
2. The estimation-decoder (that integrates the MS algorithm on the LDPC subgraph).
  - (a) Initialized with correct transition probabilities (labeled with EDc).
  - (b) Initialized with false transition probabilities (labeled with EDf).

The MS decoder wrongly assumes the data word to stem from an additive white Gaussian noise (AWGN) channel, and thus decodes without any state-estimation.

In contrast to that, the estimation-decoding is based on the correct channel-model. Since the correct  $p_{12}^c = 0.3$  and  $p_{21}^c = 0.6$  are provided to the estimation-decoder, the decoder is perfectly matched to the channel-model. This represents an ideal case which is generally not given, but it reveals the maximum possible gain obtained with estimation-decoding.

To analyze estimation-decoding in realistic situations (where the correct transition probabilities are not known) as well, the same simulation is carried out again except that false transition probabilities  $p_{12}^f = 0.05$  and  $p_{21}^f = 0.95$  are provided to the estimation-decoder.

The results based on the two decoders can be seen in Figure 6.5. In order to evaluate the decoding performance, the word error ratio (WER) is computed depending on the average  $E_b/N_0$ -values. For each decoding variant, a total of 100 decoding iterations were performed.



**Figure 6.5:** Comparison of the MS decoder with estimation-decoding. The LDPC code was a regular LDPC code with rate  $R = 0.608$  and block length  $n = 576$ .

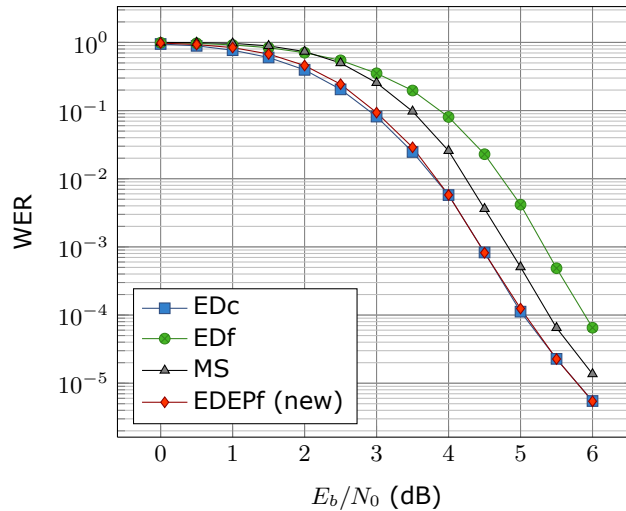
The maximum possible gain that can be obtained based on estimation-decoding can be seen when comparing the curve of the estimation-decoder initialized with the correct  $P^c$  (EDc) with the result of the MS decoder (MS). Figure 6.5 proves a gain of about 0.5 dB when adding the state-estimation to the LDPC decoding.

Particular attention has to be paid to the second estimation-decoding scenario in which the false transition probabilities have been provided to the decoder (EDf). The respective results in Figure 6.5 prove that the decoding performance is not only worse than in the case of the correct transition probabilities, but also worse than in the case of no state-estimation at all.

Therefore, estimation-decoding should only be used in the case of known transition probabilities. But this is only true without considering the new variant of estimation-decoding developed in this thesis.

### 6.5.2 Effectiveness of the new variant

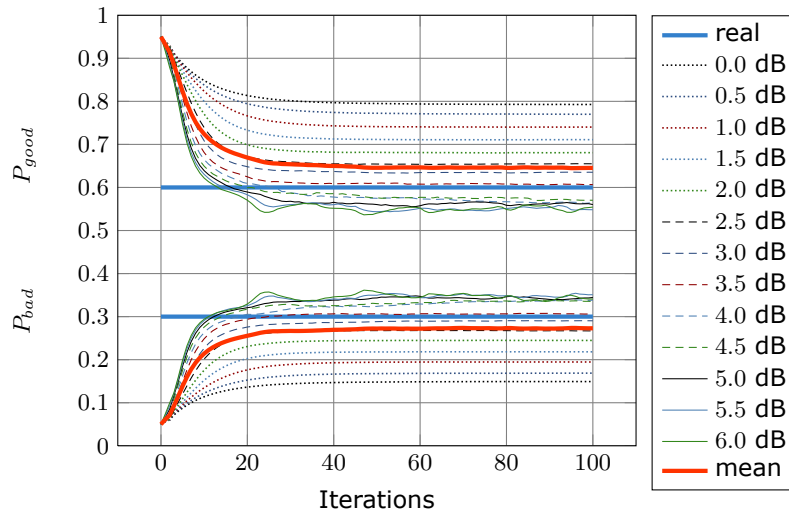
The target of the following analysis is to prove the advantage of using the new estimation-decoding variant developed in Section 6.4. For that, a simulation is conducted with the MMGC and the regular LDPC code as in the Section before (Section 6.5.1). For the decoding, the extended estimation-decoding is used that includes a continual reestimation of the transition probabilities (EDEP decoder). As in the case of the decoder labeled with EDf in the former Section, the new variant is initialized with the false transition probabilities  $p_{12}^f = 0.05$  and  $p_{21}^f = 0.95$ . The results of the simulation are shown in Figure 6.6 (labeled with EDEPf), where the results of Figure 6.5 are added for comparison purposes.



**Figure 6.6:** Comparison of four decoder variants based on different detailed channel information provided to the decoder. The LDPC code was a regular LDPC code with rate  $R = 0.608$  and block length  $n = 576$ .

When looking at the red (EDEPf) and the blue curve (EDc) in Figure 6.6, one can see that the decoding performance in the case of unknown transition probabilities is nearly as good as in the ideal case of known transition probabilities. This is possible by using the new version of estimation-decoding developed in Section 6.4 that includes the reestimation of  $P$  during the iterative decoding.

The basis for the success of the reestimation-procedure is the fast convergence of the reestimated transition probabilities very close to the correct values after only a few decoding-iterations, especially for higher  $E_b/N_0$ -values. This can be seen in Figure 6.7.



**Figure 6.7:** Reestimation of  $p_{21}$  and  $p_{12}$  during the EDEP decoding procedure for several  $E_b/N_0$  (dB) values.

So estimation-decoding is only as good as the transition probabilities that it is based on. Since in most cases one does not know the correct  $P^c$  of the channel, it is advised to use the new estimation-decoding variant developed in this thesis in order to obtain the maximum possible gain.

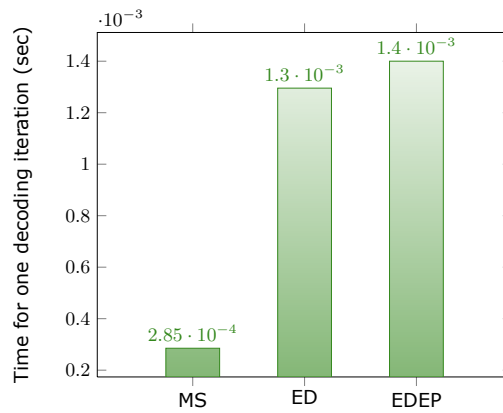
### 6.5.3 Speed considerations

When evaluating the performance of estimation-decoding, it is also important to have a look at the decoding speed. Figure 6.8 gives an overview of the time that is needed for one decoder iteration considering the three reviewed decoder variants of Section 6.5.1 and 6.5.2:

1. MS decoder.
2. Estimation-decoding (EDc and EDf).
3. EDEP decoder (new variant).

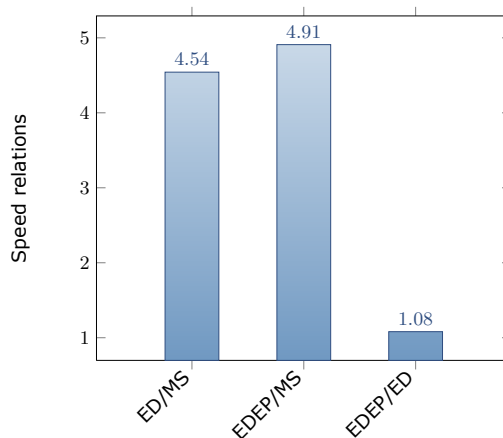
With 0.285 milliseconds (ms) per decoding iteration, the MS-algorithm is the least time-consuming decoder as expected. More computational power is required when adding the state-estimation to the MS-algorithm (ED) that yields 1.3 ms per iteration. Only a bit more is used by additionally enabling the reestimation-procedure (EDEP) where 1.4 ms are used per decoding iteration.

The relation of the time required when using estimation-decoding (ED) to the time used by the MS-decoder gives a measure of the computational cost of the state-estimation. This can be seen in Figure 6.9 that shows that 4.54 times more



**Figure 6.8:** Time for one decoding iteration considering three investigated decoder variants.

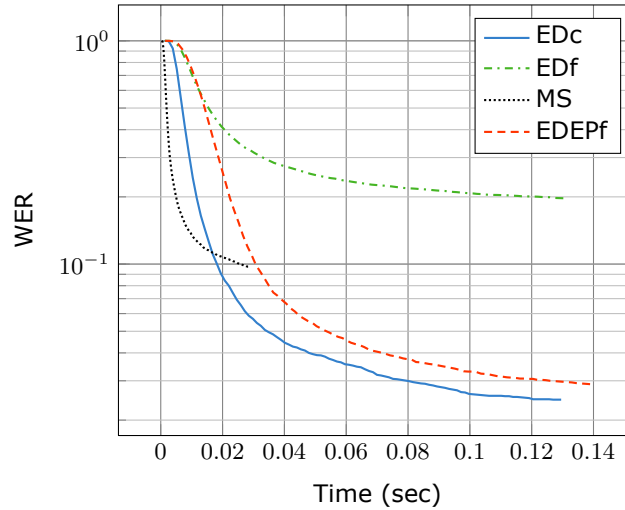
computational power is used when adding the state-estimation. If the reestimation procedure is used additionally, the factor increases to 4.91 (EDEP/MS). Compared to the usual estimation-decoding, the new variant developed in this thesis only takes 1.08 more time per iteration.



**Figure 6.9:** Extra computational cost for the additional state-estimation (ED/MS), for the additional reestimation procedure (EDEP/ED) and for the additional state-estimation together with the reestimation procedure (EDEP/MS).

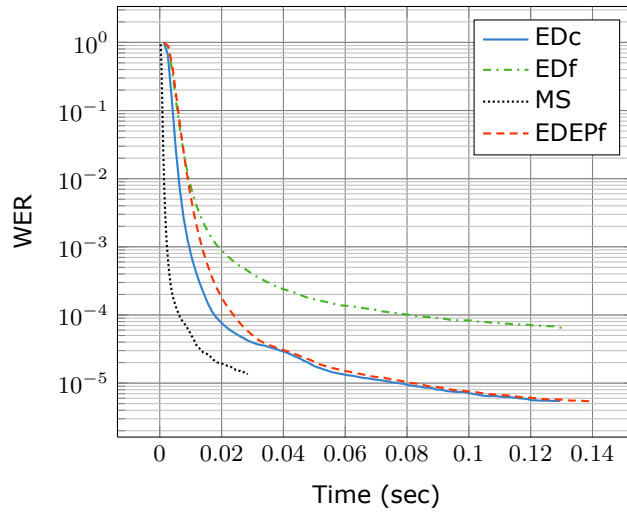
In Figure 6.10, the WER is plotted over the time for  $E_b/N_0 = 3.5$  dB. It can be seen that up to 17 ms, the MS decoder offers the best decoding-performance i.e. the lowest number of word errors. After 17 ms, the MS decoder has processed 60 iterations whereas the ED decoder, initialized with the correct transition probabilities (EDc), has only processed 13 iterations. Despite lagging behind the MS decoder in terms of iterations, the EDc variant yields better results for more than 17 ms. The EDEP decoder initialized with the wrong transition probabilities beats the MS decoder if more than 32 ms are processed. At that time, the MS decoder has already stopped due to the predefined maximum number of 100 iterations whereas the EDEP decoder

has processed 23 iterations. Once again, it can be seen that the success of the EDEP decoder is based on the additional reestimation procedure since without the reestimation, the results would yield the green curve (EDf), which is the worst no matter what time is considered.



**Figure 6.10:** Decoding performance for different points in time considering four decoding variants and  $E_b/N_0 = 3.5$  dB.

Figure 6.11 shows again the WER for different points in time but for  $E_b/N_0 = 6$  dB. When considering this relatively high signal-to-noise ratio (SNR), the decoding performance of the ideal estimation-decoding with known  $\mathbf{P}$  and the results for the developed EDEP decoder are nearly the same for more than 30 MS. After 60 MS of processing, both decoders beat the decoding performance of the MS decoder.



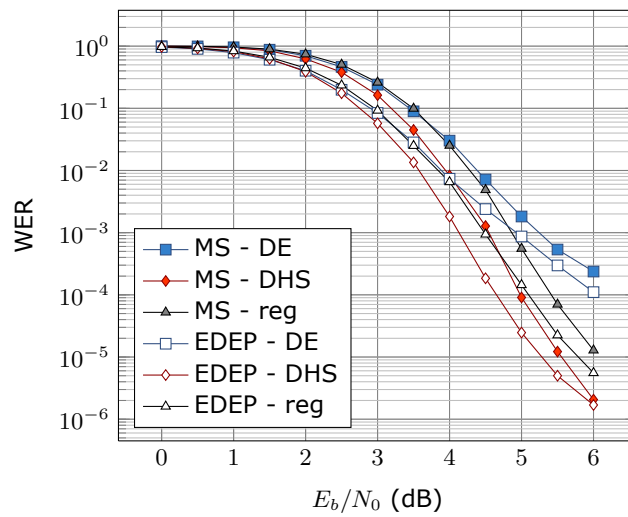
**Figure 6.11:** Decoding performance for different points in time considering four decoding variants and  $E_b/N_0 = 6$  dB.

#### 6.5.4 Estimation-decoding with irregular LDPC codes

So far, only a regular LDPC code has been considered to prove the effectiveness of using estimation-decoding. In order to check if the advantage when using estimation-decoding is the same for irregular LDPC codes, another simulation is performed with two irregular LDPC codes. One irregular LDPC code is designed with the optimization method developed in Section 5.1 and the other one has been designed based on density-evolution in [57]. As in the case of the regular LDPC code, the code rate is  $R = 0.608$  and the block length  $n = 576$  for both codes. The utilized MMGC with  $p_{12} = 0.3$  and  $p_{21} = 0.6$  has been the same again, too. For the decoding, the MS decoder as well as the EDEP decoder developed in Section 6.4 have been used. The EDEP decoder is thereby initialized with the false  $p_{12}^f = 0.05$  and  $p_{21}^f = 0.95$ . The results can be seen in Figure 6.12, where the results of the regular LDPC code are added for comparison purposes.

As in the case of the regular LDPC code, the gain of the EDEP decoder compared to the MS decoder for the irregular codes is up to 0.5 dB. This proves the advantage of using estimation-decoding for irregular LDPC codes as well.

Figure 6.12 also reveals another point. The two irregular codes have both been designed for the same channel-model, and share the same code rate and block length. The irregular LDPC code designed with the downhill simplex (DHS)-based optimization developed in this thesis is up to 1.3 dB better than the irregular code designed in [57] by means of density-evolution. The results based on density-evolution are even worse than the results for the regular LDPC code. This clearly shows that density-evolution is not an appropriate tool for the design of short irregular LDPC codes and one should prefer the design-method developed in this thesis.



**Figure 6.12:** Comparison of three rate  $R = 0.608$  LDPC codes of length  $n = 576$  with a MMGC and decoded with a MS decoder as well as with an estimation-decoder including a reestimation of  $P$  (EDEP). DE stands for the density-evolution design, DHS for the downhill simplex based design described in Section 5.1 and reg for a regular LDPC code.



## Chapter 7

# LDPC-based 2D barcodes

The outer appearance of the low-density parity-check (LDPC)-based 2D barcodes developed in this thesis is similar to that of Data Matrix codes (DMCs) since the finder pattern that surrounds a DMC has been adopted. Contrary to DMCs, which use a Reed-Solomon (RS) code [7] to encode the information to be stored in the data region, a LDPC code is utilized here. Furthermore, only one codeword is used since it is well known that the decoding performance of LDPC codes increases with the block length. In the following sections, the two worlds of LDPC codes and 2D barcodes will be brought together.

In Section 7.1, the design of an interleaver will be explained to place the symbols of a LDPC codeword into the data region, in order to increase the error-correction capabilities of the 2D barcode. Section 7.2 addresses the challenge of finding an appropriate channel-model to represent everything that happens in between the direct part marker and the LDPC decoder. The channel-model is then used for the decoder, that is developed in Section 7.3, to decode LDPC-based 2D barcodes. The design of an irregular LDPC code for 2D barcodes in Section 7.4 is based on the channel-model as well.

### 7.1 Symbol-placement

After the desired information has been encoded with a LDPC code, the symbols of the resulting code word have to be placed in the data region of the 2D barcode. This is done by means of an interleaver that defines the symbol-placement. For the construction of the interleaver, an optimization method has been developed in order to increase the error-correction capabilities of the LDPC-based 2D barcodes.

The downhill simplex (DHS) algorithm, that yielded good results when designing irregular LDPC codes, can not be adapted to the optimization problem considering the symbol-placement. This is due to the interdependencies of the parameters that have to be optimized. These parameters are the possible locations in the 2D barcode's data region for the symbols to be placed at. If one wants to place a symbol at a new location, the symbol that is currently located at that location has to be moved as well. Because of these interdependencies, the DHS algorithm can not be applied. For that reason a specific optimization method has been developed.

The basis for the algorithm is the assumption that typical interferences in direct part mark identification (DPMI) applications (e.g.: dirt, rust, blobs, scratches, unequal illumination etc.) are burst type errors and thus affect just a local part of the 2D barcode. This leads to the advice to place symbol-nodes, that are involved in the

same parity-check equation, as far as possible from each other in the data region of the 2D barcode. This is due to the fact that for the LDPC decoder it is easier to correct two faulty symbol-nodes that affect two check-nodes having a long distance to each other (which means starting from one check-node it takes lots of edges to reach the other one) than two faulty symbols within the same check-equation.

When considering the message-passing principle of the LDPC decoder, the error of one faulty symbol that affects a check-node (and thus the other symbol-nodes adjacent to that check-node) is easily corrected by means of the messages arriving at the neighboring symbol-nodes from other check-nodes. This can be seen in the small example in Figure 7.1a, where only symbol-node  $x_2$  is influenced by the disturbance. For simplicity, everything is kept binary in this example, the symbols as well as the messages. The original codeword is the all-zero code word (AZCW), where all the bits are set to zero, so  $x_2$  is by fault set to one. Everything that is influenced by the faulty symbol-node  $x_2$  in the first decoding iteration is marked with red. According to the belief propagation (BP) algorithm (see Section 4.4.1), a check-node sends a message to a connected symbol-node in which he tells him what he should be, so that the parity-check equation is fulfilled, considering the values of the other connected symbol-nodes.  $c_0$  for example tells the faulty  $x_2$  he should be a zero since  $x_0$  and  $x_1$  are zero and  $c_0 = x_0 + x_1 + x_2 = 0 + 0 + 0 = 0$ . A hard-decision (HD) is then computed based on all incoming messages at a symbol-node together with the received codeword. In the small binary example, the decision if a symbol is a one or a zero is made depending on the amount of guesses for a symbol to be a one or a zero, respectively. In the case of  $x_2$ , there are three messages that guess  $x_2 = 0$  and only one guess (the received bit) for  $x_2 = 1$ . So the HD would yield  $x_2 = 0$  and for the whole code word the AZCW.

Figure 7.1b shows an example for a symbol-placement without the proposed optimization. In the presented case,  $x_1$  would be affected by the disturbance as well and the decoder would not be able to make a decision after the first iteration. This is due to the fact, that for  $x_1$  and  $x_2$  there would be two guesses for the symbols to be a one and two guesses for being a zero, respectively.

The placement of the code word's symbols in the data region of the 2D barcode is constrained by the available grid in the data region and the connections between the symbol-nodes in the Tanner graph. In order to get an optimal placement, an optimization algorithm has been developed that considers the constraints and is based on two parameters.

## 7.1.1 Distance measurements

### 7.1.1.1 Geometrical distance

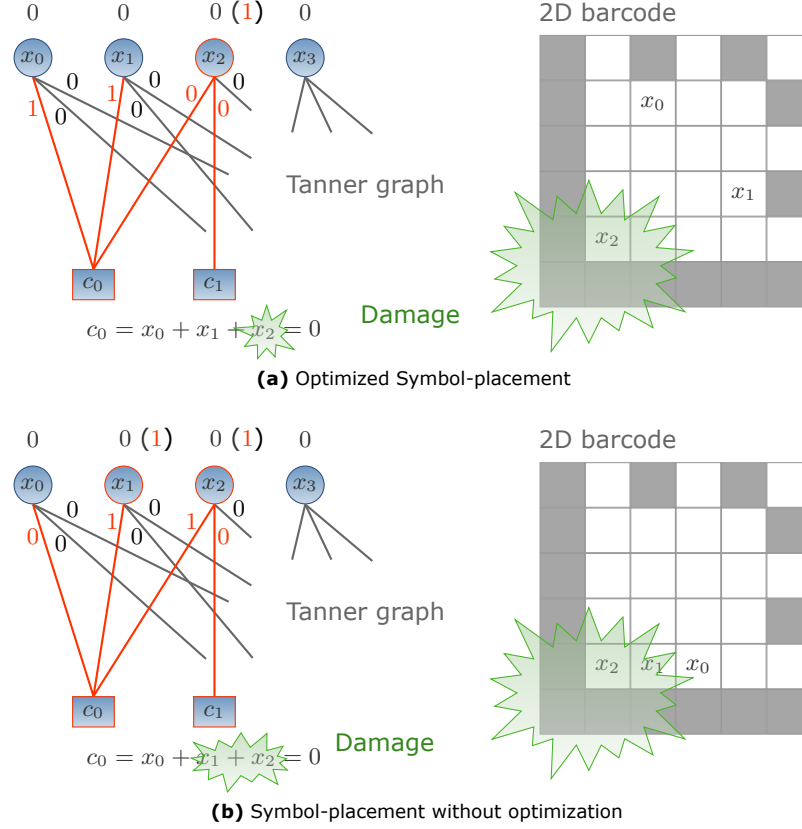
The first is the geometrical distance  $d_E(x_a, x_b)$  between two symbol-nodes  $x_a$  and  $x_b$  in the data region of the 2D barcode. For the computation the Euclidean distance is utilized according to

$$d_E(x_a, x_b) = \sqrt{(c_{x_a} - c_{x_b})^2 + (r_{x_a} - r_{x_b})^2} \quad (7.1)$$

with  $c$  and  $r$  being the row and column of the appropriate symbol in the data region, respectively.

### 7.1.1.2 Tree distance

The second parameter is the distance between two symbol-nodes  $x_a$  and  $x_b$  in the Tanner graph, and is represented by the shortest path that connects them. A path is thereby a connection between two nodes by consecutive edges. To find the shortest



**Figure 7.1:** Effect of local disturbances with and without optimized symbol-placement considering LDPC-based 2D barcodes.

path between two symbol-nodes, the progressive-edge-growth (PEG)-trees introduced in Section 4.5.4 are utilized. The distance between two symbol-nodes in the Tanner graph is denoted as  $d_T(x_a, x_b)$  and is called *tree distance* in the following. If, for example, one wants to know the distances that symbol-nodes have to the symbol-node  $x_a$ , one constructs the PEG-tree with  $x_a$  being the root (layer zero). The tree distance  $d_T(x_a, x_b)$  of any symbol-node  $x_b$  to  $x_a$  is then determined based on the layer where  $x_b$  is located in the PEG-tree. If  $x_b$  would, for example, be located in layer 4 of the PEG-tree with  $x_a$  being the root and thus belonging to layer 0, then the tree-distance would yield  $d_T(x_a, x_b) = 4$ . In Figure 4.5b for example, the tree distance between the root  $x_0$  and  $x_1$  would be  $d_T(x_0, x_1) = 2$ .

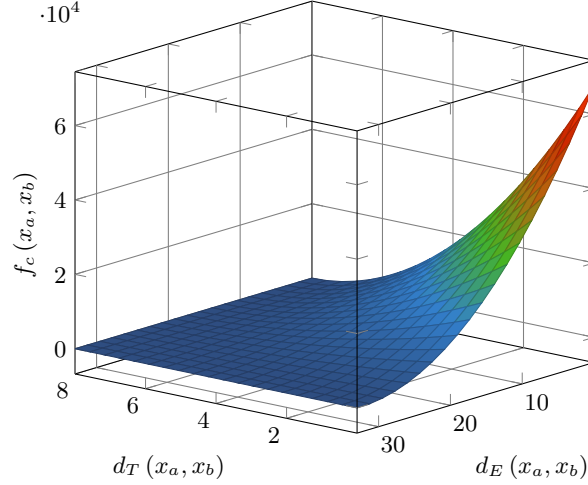
### 7.1.2 Cost of a symbol-placement

Based on the two distance-measurements  $d_E(x_a, x_b)$  and  $d_T(x_a, x_b)$ , a cost-function is defined for each pair of symbol-nodes  $(x_a, x_b)$  by

$$f_c(x_a, x_b) = (d_T(x_a, x_b) - d_T^{max})^2 \cdot (d_E(x_a, x_b) - d_E^{max})^2 \quad (7.2)$$

with  $d_T^{max}$  and  $d_E^{max}$  being the maximum distance  $d_T(x_a, x_b)$  in the Tanner graph and the maximum possible Euclidean distance, respectively. The cost-function is shown in

Figure 7.2 for a data region of size  $24 \times 24$  and  $d_T^{max} = 8$ .



**Figure 7.2:** Cost-function for  $d_E^{max} = 32.53$  (data region of size  $24 \times 24$ ) and  $d_T^{max} = 8$  (a regular LDPC code with SND  $d_x = 3$ ).

To get the cost  $\mathcal{C}$  of a complete symbol-placement of a 2D barCode (2DC), one computes

$$\mathcal{C}(2DC) = \sum_{a=0}^{n-1} \sum_{b=0}^{n-1} f_c(x_a, x_b); \quad \forall a \neq b \quad (7.3)$$

where  $n$  is the number of symbols in the 2D barcode's data region.

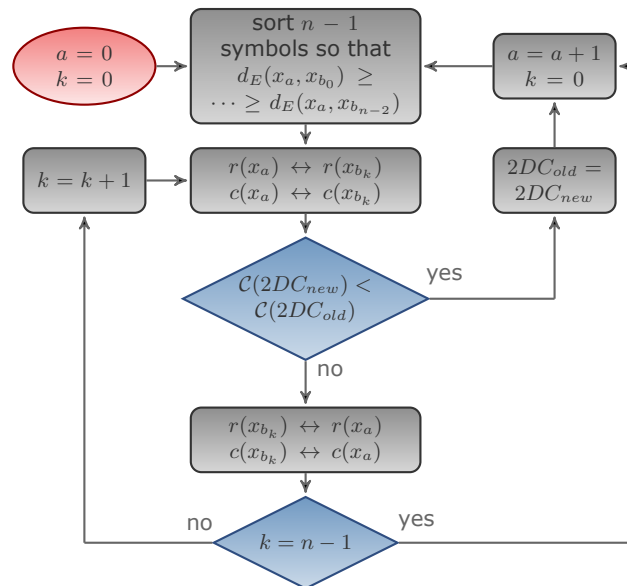
### 7.1.3 Optimization of the symbol-placement

The optimization algorithm is depicted in Figure 7.3.

The optimization starts with a random placement of the LDPC codeword's symbols in the data region and the computation of the resulting cost  $\mathcal{C}(2DC_{old})$  according to Equation (7.3). Then starting with the first symbol  $x_a = x_0$  in the codeword, the Euclidean distances to the remaining symbols are computed with Equation (7.1). The remaining symbols  $x_1, \dots, x_{n-1}$  are then sorted so that

$$d_E(x_a, x_{b_0}) \geq d_E(x_a, x_{b_1}) \geq \dots \geq d_E(x_a, x_{b_{n-2}}). \quad (7.4)$$

So  $x_{b_0}$  is the furthestmost symbol to  $x_a$  in the data region of the 2D barcode and  $d_E(x_a, x_{b_{n-2}})$  is the closest. The algorithm starts with  $x_{b_0}$ , and swaps the position of the current symbol-node  $x_a = x_0$  and the one that has the greatest Euclidean distance to it which is  $x_{b_0}$ . Then the cost of the new symbol-placement  $\mathcal{C}(2DC_{new})$  is computed. If the new cost is lower than the old one ( $\mathcal{C}(2DC_{new}) < \mathcal{C}(2DC_{old})$ ), the algorithm keeps the new symbol-placement and continues with the next symbol  $x_a = x_1$ . If the position-swap of the two symbols  $x_a = x_0$  and  $x_{b_k} = x_{b_0}$  did not yield a better cost, the algorithm continues with the old symbol-placement and swaps  $x_a$  with  $x_{b_k} = x_{b_1}$ . The algorithm terminates if there was no successful swap during a complete round (for  $x_a = \{x_{end}, x_{end+1}, \dots, x_{n-1}, x_0, \dots, x_{end-1}\}$ ).



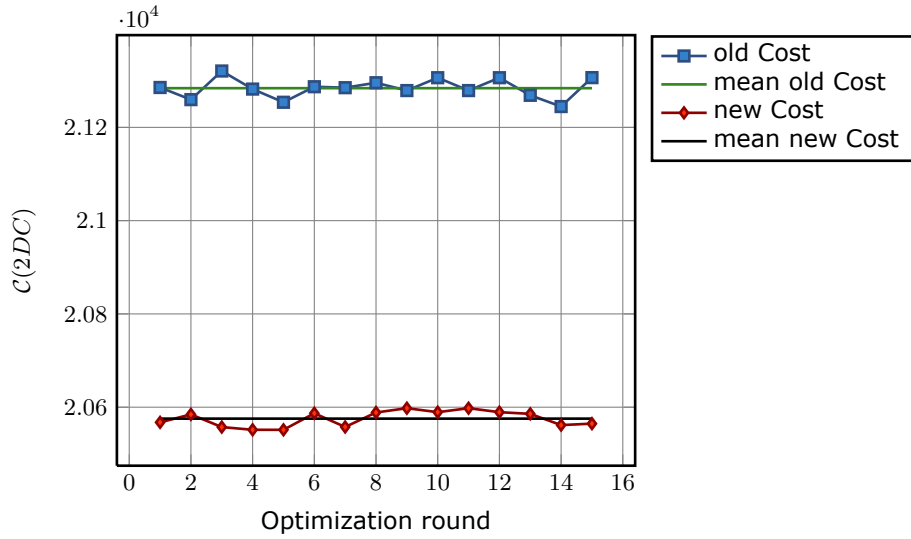
**Figure 7.3:** Flow-chart of the symbol-placement optimization.

Since it is not possible to guarantee the convergence of the algorithm to a global minimum, the optimization process is repeated at least 10 times with different random initialization placements.

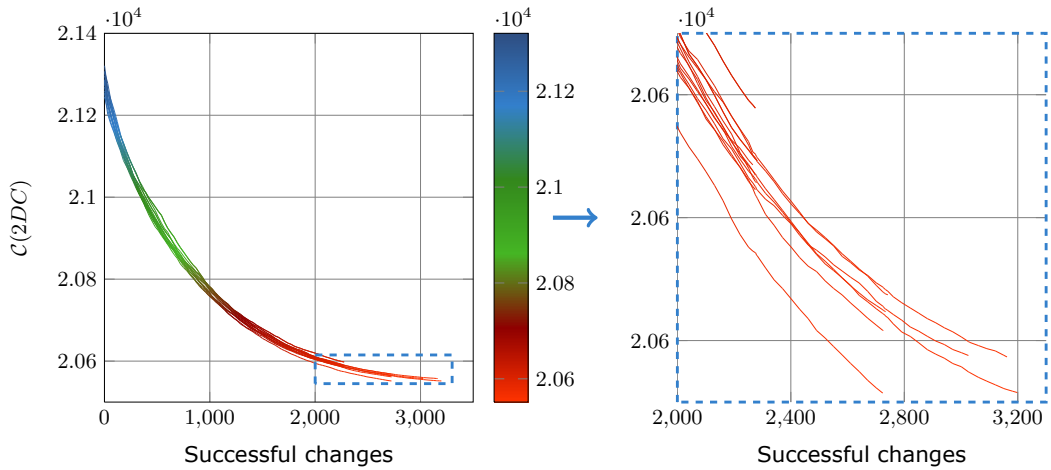
#### 7.1.4 Optimization results

Figure 7.4 shows the costs of 15 initializing symbol-placements (old cost) and the appropriate final costs, when optimizing the symbol-placement for a 2D barcode with a size  $24 \times 24$  data region and a rate  $R = 0.611$  regular LDPC code of length  $n = 576$ .

It can be seen, that in all cases in Figure 7.4, the minima that have been found are quite similar. The standard deviation is  $\sigma = 16.9$ , which is small compared to the absolute value of the cost function that is in the order of magnitude of  $10^4$ . This can also be seen in Figure 7.5 where the costs are drawn over successful swaps for all 15 optimization rounds. The effectiveness of the proposed optimized symbol-placement is analyzed in Section 8.3.



**Figure 7.4:** Initial costs (old Cost) and final cost (new Cost) of 15 independent symbol-placement optimization rounds.

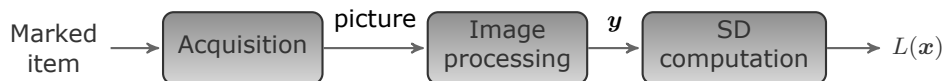


**Figure 7.5:** Evolution of the costs for 15 independent symbol-placement optimization processes.

## 7.2 Channel-model for 2D barcodes

In contrast to the RS decoder, which is used in the case of the standard DMC, the LDPC decoder uses soft-decisions (SDs) as an input. As explained in Section 3.6, one keeps the maximum information related to a symbol when using SDs. The probability of that symbol being a zero or a one is thereby calculated, instead of making a decision for a zero or a one as done in the case of HDs. The SDs are computed by means of a chosen channel-model, and then passed to the LDPC decoder. Thus the decoding success of the presented LDPC-based 2D barcodes is heavily dependent on the choice of an appropriate channel-model.

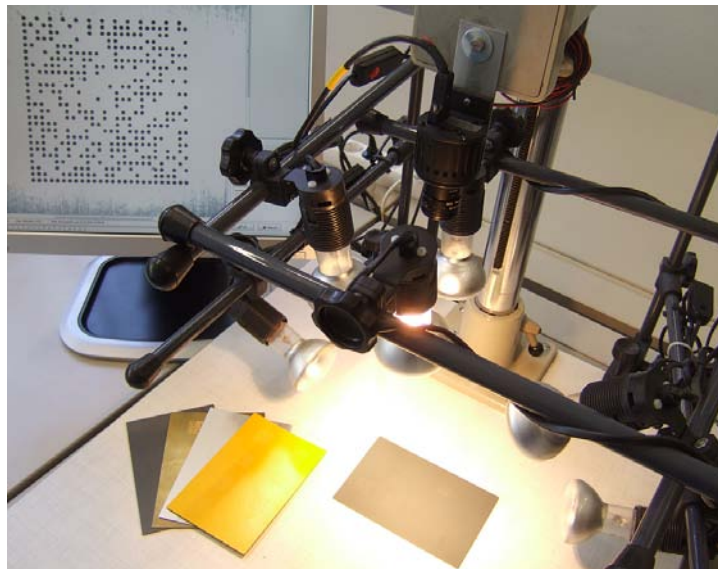
In a common communication scenario, one receives values at the end of a channel. One can then analyze the received values in order to find a channel-model that sufficiently represents the real channel (see Chapter 3). In the case of 2D barcodes, one first has to think of how to get the *received data word*  $y$  out of the 2D barcode that keeps the *sent code word*  $x$  stored in the shape of modules inside of the data region. This is done as depicted in Figure 7.6 and is explained in the following



**Figure 7.6:** How to get the SDs based on an item marked with a 2D barcode.

### 7.2.1 Acquisition and image processing

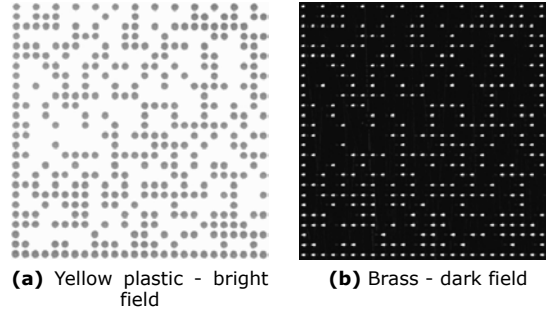
The acquisition is done by means of a camera-based system. All pictures for this thesis have been taken by use of the system shown in Figure 7.7.



**Figure 7.7:** The camera-based system for the acquisition of 2D barcodes.

Depending on the marking method and the material marked with a 2D barcode, a picture is taken in either a *bright field* or a *dark field*. In a bright field, the untouched

surface reflects the employed light back into the camera's lens, whereas the cavities that represent a binary one stay dark on the resulting picture. The situation in the case of a dark field is vice versa. Figure 7.8 shows one example for an acquisition in a bright field and in a dark field, respectively.



**Figure 7.8:** Pictures taken of a 2D barcode in a bright field and a dark field, respectively.

In the case of Figure 7.8, the pictures are already cropped perfectly matching the size of the 2D barcode. This was done since the localization that is part of the image processing system is the main topic of the cooperative doctoral thesis *Optimizarea recunoașterii codurilor Data Matrix în mediul industrial*<sup>1</sup> written by Ion-Cosmin Dita [64]. Thus the localization of 2D barcodes is not considered in this thesis and a perfectly cropped picture is assumed.

To obtain the received data word  $\mathbf{y}$  out of the 2D barcode picture, a correlation-based method is established. It is based on the DMC's finder pattern that has been adopted for the LDPC-based 2D barcodes and that always looks the same (see Figure 2.4b and 2.4c). For each module in row  $r$  and column  $c$  of the 2D barcode's data region, a correlation coefficient is calculated according to

$$y_{rc} = \frac{\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} (a_{kl}^{rc} - \bar{a}^{rc})(b_{kl} - \bar{b})}{\sqrt{\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} (a_{kl}^{rc} - \bar{a}^{rc})^2 \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} (b_{kl} - \bar{b})^2}} \quad (7.5)$$

with  $k$  and  $l$  being the indices for the  $K$  vertical and  $L$  horizontal pixels in each module, respectively. Considering one module in row  $r$  and column  $c$ ,  $a_{kl}^{rc}$  denotes one pixel in row  $k$  and column  $l$  of the module. The mean  $\bar{a}^{rc}$  of the pixels  $a_{kl}^{rc}$  of a module in row  $r$  and column  $c$  is computed according to

$$\bar{a}^{rc} = \frac{1}{K \cdot L} \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} a_{kl}^{rc}. \quad (7.6)$$

$b_{kl}$  is a pixel in the *reference module* which is generated based on an averaging of all modules that belong to the 2D barcode's finder pattern and represent a binary one.

<sup>1</sup>In English: Optimized recognition of Data Matrix codes in industrial environments.



$$\begin{aligned}
b_{kl} = & \frac{1}{2} \left( \overbrace{\frac{1}{R-1} \sum_{i=1}^{R-1} a_{r=i,c=0}^{kl} + \frac{1}{C-2} \sum_{i=1}^{C-2} a_{r=R-1,c=i}^{kl}}^{\text{one-modules of the L-pattern}} \right) \\
& + \frac{1}{2} \left( \overbrace{\frac{2}{R} \sum_{i=0}^{R/2-1} a_{r=2i+1,c=C-1}^{kl} + \frac{2}{C} \sum_{i=0}^{C/2-1} a_{r=0,c=2i}^{kl}}^{\text{one-modules of the broken border}} \right)
\end{aligned} \tag{7.7}$$

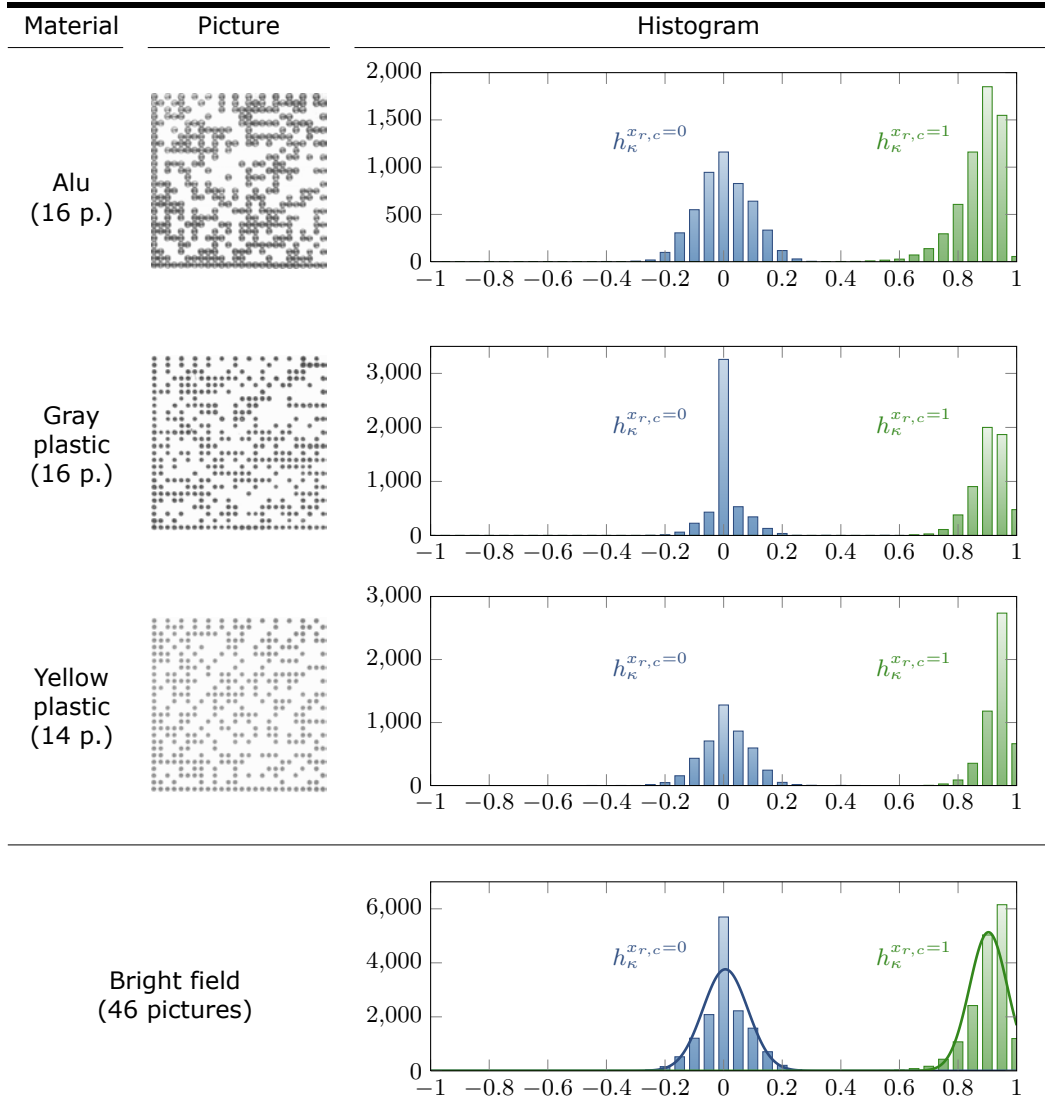
with  $R$  and  $C$  being the total number of rows and columns in the 2D barcode. The mean  $\bar{b}$  of the reference module is then computed by

$$\bar{b} = \frac{1}{K \cdot L} \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} b_{kl}. \tag{7.8}$$

So in the case of 2D barcodes, a correlation coefficient  $y_{rc}$  represents the received value at the end of the channel referring to the sent value  $x_{rc}$  stored in the shape of a module in row  $r$  and column  $c$  of the 2D barcode's data region. Thus, one has to analyze the distribution of the correlation coefficients in representative decoding situations in order to find an appropriate channel-model. Based on the utilized channel-model, the SDs are then computed and subsequently forwarded to the LDPC-decoder.

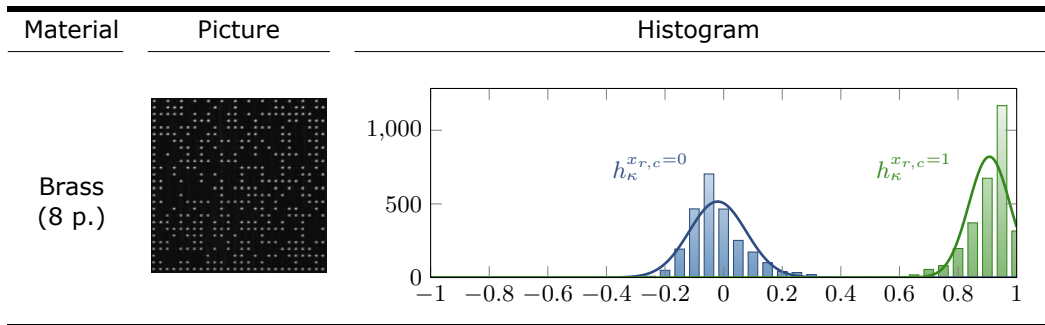
### 7.2.2 Channel-model in absence of damages

To find a suitable channel-model for DPMI applications, 2D barcodes have been milled on different kinds of materials like metal, brass, different colored plastic and copper. Then pictures were taken in a bright field or a dark field, depending on the material. Afterwards, the correlation coefficients were computed by means of the cropped pictures according to Equation (7.5). Based on the knowledge of the 2D barcode, the correlation coefficients have been separated into two data-sets referring to one-modules ( $x_{rc} = 1$ ) and zero-modules ( $x_{rc} = 0$ ), respectively. For each set a histogram  $h_{\kappa}^{x_{r,c}}$  was computed. For the bright field case, 46 pictures taken of 2D barcodes milled on aluminum, gray plastic and yellow plastic were used. These materials showed the biggest variance to each other in the resulting histograms. Based on the histograms for the individual materials, two average histograms were computed (for the one-modules and zero-modules, respectively) to represent the bright field case. The histograms can be seen in Table 7.1 where the number of modules is plotted as a function of the correlation coefficient. The histograms  $h_{\kappa}^{x_{r,c}=1}$  (for the one-modules) and  $h_{\kappa}^{x_{r,c}=0}$  (for the zero-modules) are marked with green and blue, respectively.



**Table 7.1:** Distributions of the correlation coefficients for the one-modules and zero-modules without considering possible damages. The two histograms in the bottom represent the bright field case and are computed based on the three diagrams above.

For the dark field case, 8 pictures of 2D barcodes milled in brass were used. When considering other materials, the pictures taken in a dark field and thus the correlation coefficients looked pretty much the same. The histograms referring to the dark field can be seen in Table 7.2.

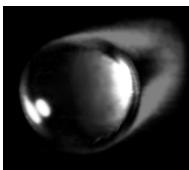



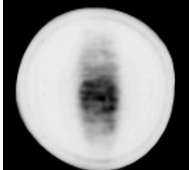
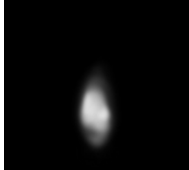


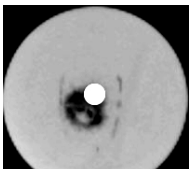
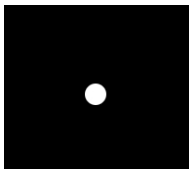
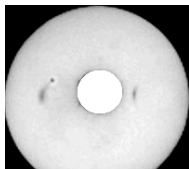
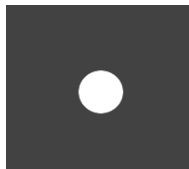
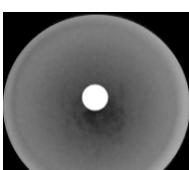
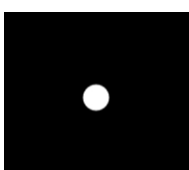
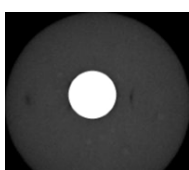
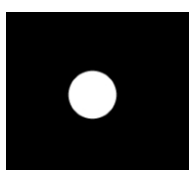


**Table 7.2:** Histograms for the one-modules and zero-modules considering 2D barcode pictures taken in a dark field.

The analysis is separated into the bright field case and the dark field case since for the 2D barcode decoder it is possible to distinguish between these two cases by means of the finder pattern that surrounds every 2D barcode. Furthermore, a Gaussian approximation is added to the histograms in Table 7.1 and 7.2 that represent the bright field case and the dark field case, respectively. With the help of the Gaussian distributions, one could already compute the required SDs, but until now interferences that may occur in industrial applications like dirt, rust, blobs, scratches, unequal illumination and so on have not been considered. All kinds of possible interferences cause contiguous modules to appear darker or brighter and thus affect the distribution of the correlation coefficients. Because of that, an analysis of the histograms influenced by damages is necessary.

### 7.2.3 Channel-model for damaged 2D barcodes

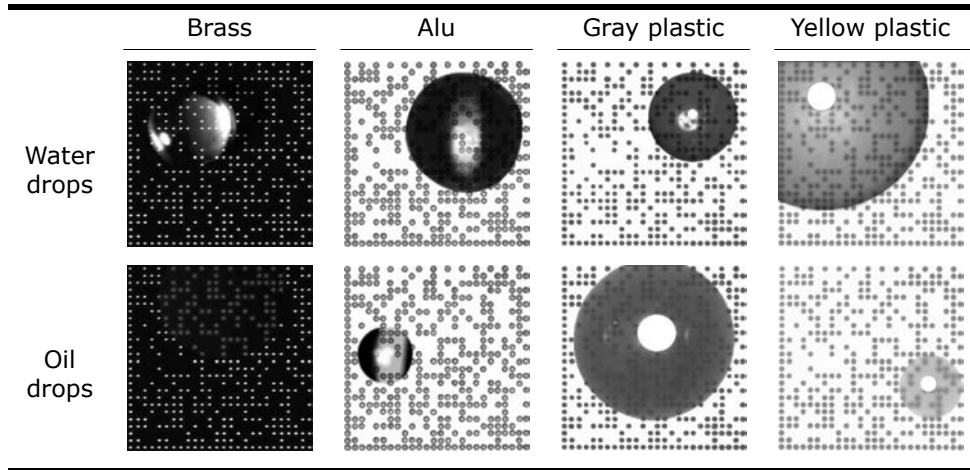
Since it is not possible to get a sufficiently high statistic by manually interfering with 2D barcodes, a simulation of representative damage-patterns has been developed. The established simulation of oil drops and water drops is based on reference drops that are obtained by taking a picture of a real drop on the desired material considering the appropriate illumination setting. The cropped picture containing the damage is then separated into two pictures that together form the reference drop. One picture contains the alpha-channel that represents the transparency of the damage where the other picture contains the color map. Such a reference damage can then be added free scalable to the desired 2D barcode picture. Table 7.3 shows the reference oil drops and water drops for different materials.

Material	Water drops		Oil drops	
	Alpha channel	Color map	Alpha channel	Color map
Brass				
Alu				
Gray plastic				
Yellow plastic				

**Table 7.3:** Alpha channel and color map of reference oil drops and water drops on different materials.

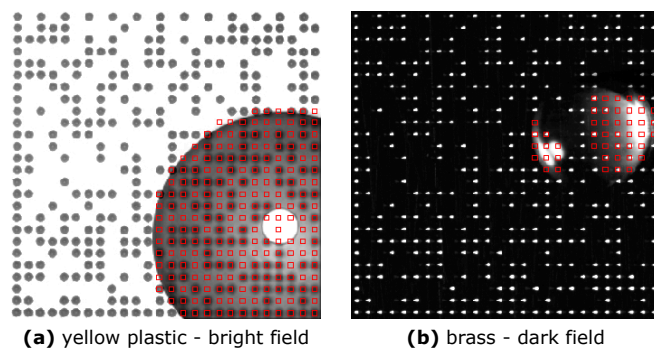
Table 7.4 shows some examples of interference on 2D barcodes with oil drop and water drop simulations based on the reference drops in Table 7.3. The simulations have been added to the barcode pictures considering different sizes and different

locations.



**Table 7.4:** Examples for damage-simulations by means of the patterns in Table 7.3.

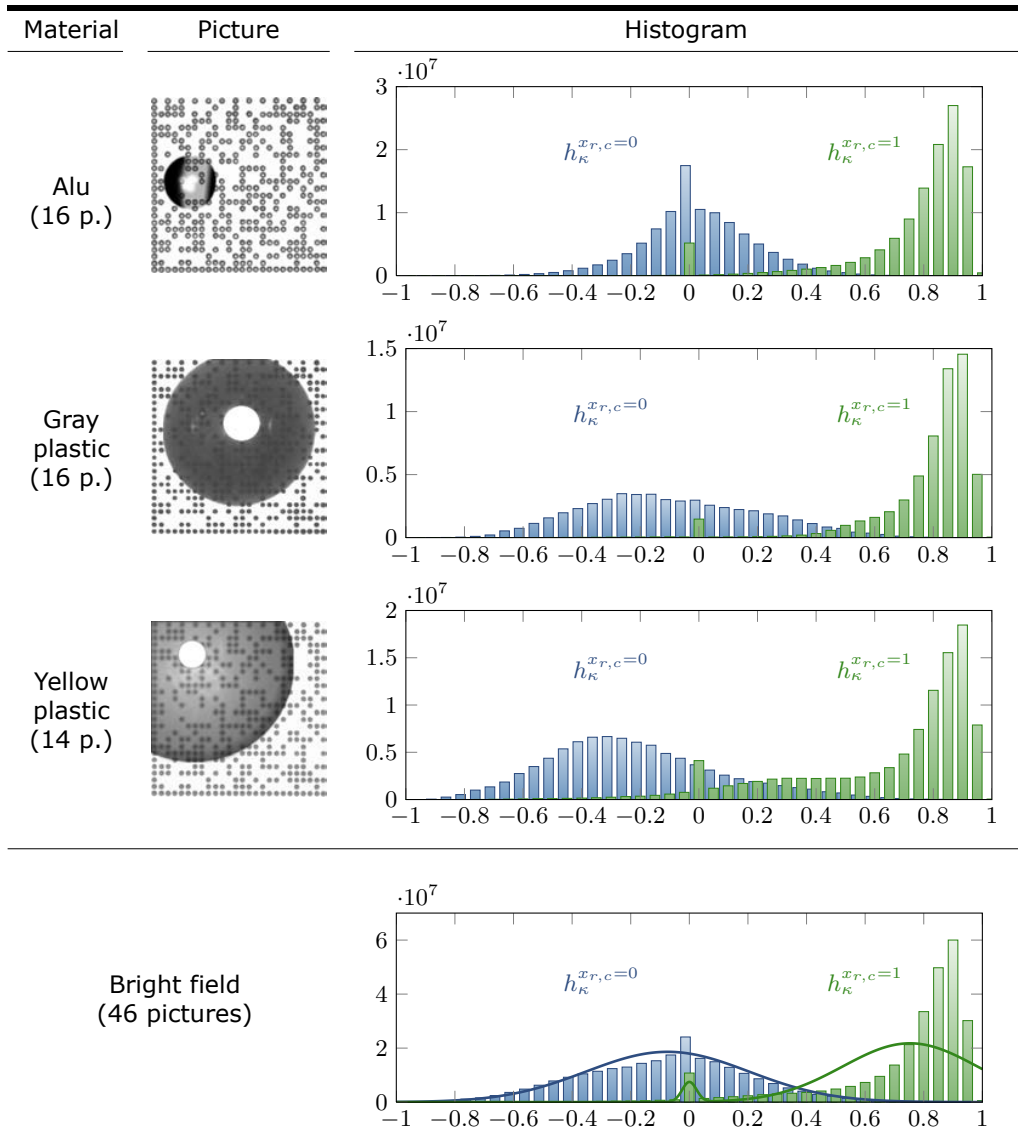
The analysis of the channel including possible damages has then been done based on the introduced oil drop and water drop simulations. Thereby 10000 water drop simulations and 10000 oil drop simulations were added to each of the 54 pictures that Table 7.1 and 7.2 are based on. Then again, the correlation-coefficients have been computed based on Equation (7.5) and subsequently separated into two data-sets referring to the one-modules and the zero-modules, respectively. In contrast to the analysis in Section 7.2.2, only the modules have been considered that have been influenced by the drop simulations. This was done by means of a threshold-based comparison between the module-mean value  $\bar{a}^{rc}$  with the damage simulation and the mean value without the damage simulation. Figure 7.9 shows two different water drop simulations where affected modules have been automatically marked.



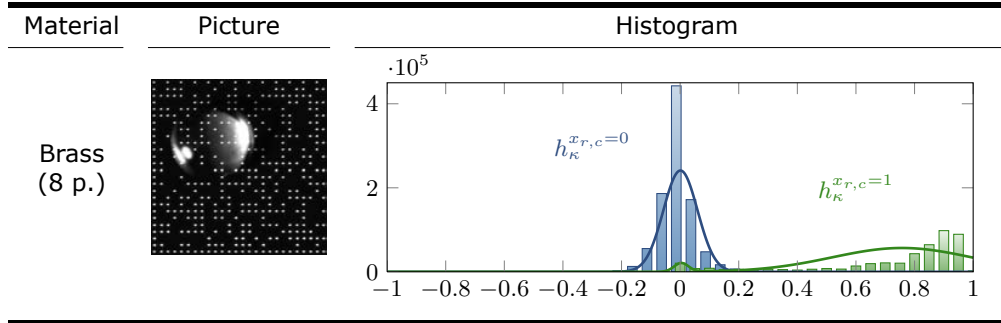
**Figure 7.9:** Same pictures as in Figure 7.8 but with a added water drop simulation. In addition the modules, affected by the damage simulation have been marked.

Since the decoder is not able to distinguish between different kinds of damages, each of the histograms in Table 7.5 and 7.6 is based on both damage simulations, the

oil drop and water drop simulations. The pictures in the second column represent just one example. The diagram in the bottom of Table 7.5 and the histograms in Table 7.6 represent the channel including possible damages for the bright field case and the dark field case, respectively.



**Table 7.5:** Distributions of the correlation coefficients for the one-modules and zero-modules including possible damages. The two histograms in the bottom represent the bright field case and are computed based on the three diagrams above.



**Table 7.6:** Histograms for the one-modules and zero-modules considering 2D barcode pictures taken in a dark field including possible damages.

Since the histograms clearly differ when considering damages (Table 7.5 and 7.6) compared to the case without any damages (Table 7.1 and 7.2), a 2-state Markov-model (see Section 3.4) is utilized. The *good channel* represents the case without any damages, and the *bad channel* takes damages into account. So if a module in row  $r$  and column  $c$  is affected by the good channel, the channel-model is in state  $s_{r,c} = s_1$ , otherwise the module is influenced by the bad channel and  $s_{r,c} = s_2$ .  $x_{r,c}$  is a symbol placed on a module in row  $r$  and column  $c$ , and  $y_{r,c}$  is the correlation coefficient referring to the symbol  $x_{r,c}$ . The histogram  $h_{\kappa}^{s_{r,c}=s_1, x_{r,c}=0}$ , for example, refers to the channel-state  $s_{r,c} = s_1$ , which is the good sub-channel, and a zero-module representing a sent bit  $x_{r,c} = 0$ .

#### 7.2.4 Computation of soft-decisions for 2D barcodes

The SDs are computed based on the 2-state Markov-modulated channel-model established in Section 7.2.3. For the sake of clarity, the histograms in the bottom of Table 7.1 and 7.5 (good and bad channel for the bright field case) and the histograms of Table 7.2 and 7.6 (good and bad channel for the dark field case) are shown again in Figure 7.10 and 7.11 representing the complete 2-state channel-model considering the bright field case and the dark field case, respectively. The good sub-channel is thereby shown in the top and the bad sub-channel in the bottom of Figure 7.10 and 7.11.

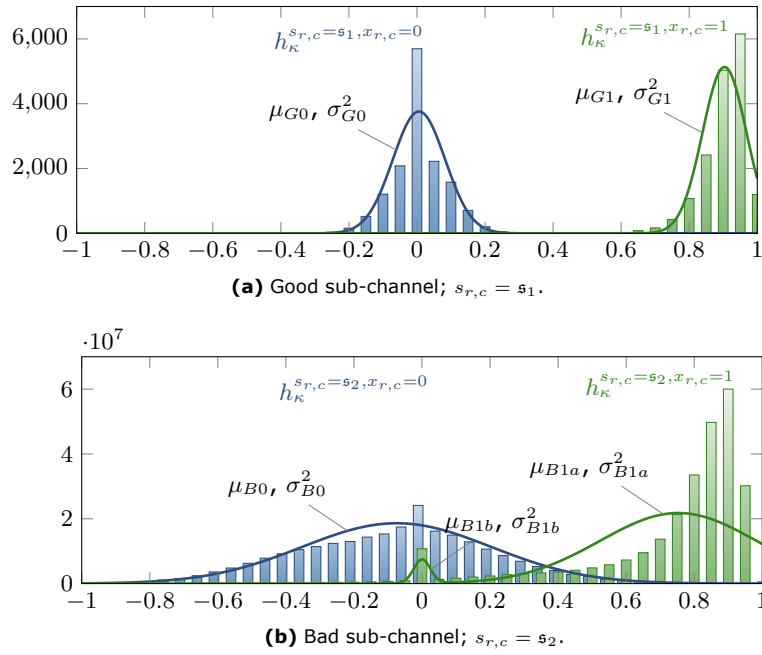


Figure 7.10: 2-state Markov-modulated channel-model for the bright field case.

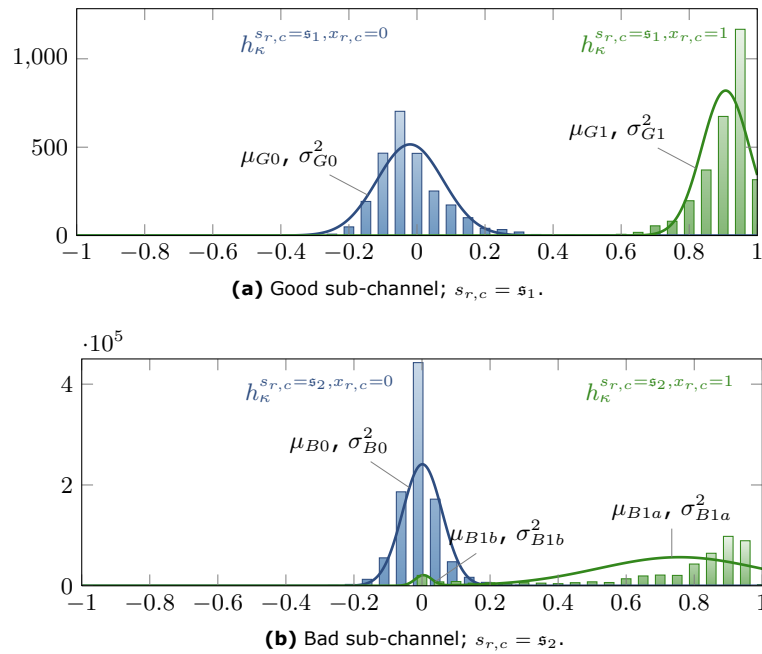


Figure 7.11: 2-state Markov-modulated channel-model for the dark field case.



For the computation of the SDs referring to the good channel, the Gaussian approximations that can be seen in the top in Figure 7.10 and 7.11 are used. The probabilities for the correlation coefficient  $y_{r,c}$  to stem from a one-module or a zero-module are then computed as follows:

$$P(y_{r,c} | x_{r,c} = 1, s_{r,c} = \mathfrak{s}_1) = \frac{1}{\sqrt{2\pi\sigma_{G1}^2}} \exp\left(\frac{-(y_{r,c} - \mu_{G1})^2}{2\sigma_{G1}^2}\right) \quad (7.9a)$$

and

$$P(y_{r,c} | x_{r,c} = 0, s_{r,c} = \mathfrak{s}_1) = \frac{1}{\sqrt{2\pi\sigma_{G0}^2}} \exp\left(\frac{-(y_{r,c} - \mu_{G0})^2}{2\sigma_{G0}^2}\right). \quad (7.9b)$$

The values for the means and variances are listed in Table 7.7 and 7.8 for the bright field and the dark field, respectively. The mismatch that the Gaussian probability density function (PDF) shows to the one-module's histogram on the very right side is harmless since the probability for a module with a correlation coefficient in that region to be a zero-module is close to zero. The same is true for the very left side of the zero-module's histogram in the dark field case.

When approximating the bad channel with a Gaussian PDF, there is a mismatch considering one-modules with  $y_{r,c} > 0.4$ , but that does not have a noticeable drawback since the probability for a module to be a zero is very small in that region. Contrary to that, one has to consider the possibility of a one-module having a correlation coefficient of  $y_{r,c} \approx 0$ . This is respected by using two Gaussian curves for the one-modules and a transition probability  $\epsilon$  for switching from the Gaussian curve  $a$  to the Gaussian curve  $b$  that describes the probability for  $P(y_{r,c} \approx 0 | x_{r,c} = 1)$ . This is much like an integrated Z-channel. The probabilities for  $y_{r,c}$  to stem from a one-module or a zero-module are then computed according to

$$P(y_{r,c} | x_{r,c} = 1, s_{r,c} = \mathfrak{s}_2) = \frac{1 - \epsilon}{\sqrt{2\pi\sigma_{B1a}^2}} \exp\left(\frac{-(y_{r,c} - \mu_{B1a})^2}{2\sigma_{B1a}^2}\right) + \frac{\epsilon}{\sqrt{2\pi\sigma_{B1b}^2}} \exp\left(\frac{-(y_{r,c} - \mu_{B1b})^2}{2\sigma_{B1b}^2}\right) \quad (7.10a)$$

and

$$P(y_{r,c} | x_{r,c} = 0, s_{r,c} = \mathfrak{s}_2) = \frac{1}{\sqrt{2\pi\sigma_{B0}^2}} \exp\left(\frac{-(y_{r,c} - \mu_{B0})^2}{2\sigma_{B0}^2}\right). \quad (7.10b)$$

The values for the means and variances can be seen in Table 7.7 and 7.8 for the bright field and the dark field, respectively.

Sub-channel	1-modules		0-modules	
	mean	variance	mean	variance
Good	$\mu_{G1} = 0.9038$	$\sigma_{G1}^2 = 0.0042$	$\mu_{G0} = 0.0059$	$\sigma_{G0}^2 = 0.0059$
Bad	$\mu_{B1a} = 0.7531$	$\sigma_{B1a}^2 = 0.0528$	$\mu_{B0} = -0.0744$	$\sigma_{B0}^2 = 0.0775$
	$\mu_{B1b} = 0.001$	$\sigma_{B1b}^2 = 0.000607$		

**Table 7.7:** Bright field case: means and variances of the Gaussian PDFs in Figure 7.10 that represent the channel-model for 2D barcodes.

Sub-channel	1-modules		0-modules	
	mean	variance	mean	variance
Good	$\mu_{G1} = 0.9071$	$\sigma_{G1}^2 = 0.0049$	$\mu_{G0} = -0.0204$	$\sigma_{G0}^2 = 0.0095$
Bad	$\mu_{B1a} = 0.7569$	$\sigma_{B1a}^2 = 0.057$	$\mu_{B0} = 0.0011$	$\sigma_{B0}^2 = 0.0034$
	$\mu_{B1b} = 0.0042$	$\sigma_{B1b}^2 = 0.0031$		

**Table 7.8:** Dark field case: means and variances of the Gaussian PDFs in Figure 7.11 that represent the channel-model for 2D barcodes.

The transition probability  $\epsilon$  was computed as the ratio of the 0-bin surface to the surface of the other bins referring to the one-modules and is  $\epsilon = 0.035$  for the bright field and  $\epsilon = 0.0362$  for the dark field. The variances  $\sigma_{B1b}^2$  have then been computed so that the ratio of the two curves for one-modules and zero-modules at  $y_{r,c} = 0$  showed the same values as the ratio of the two bins at the same position.

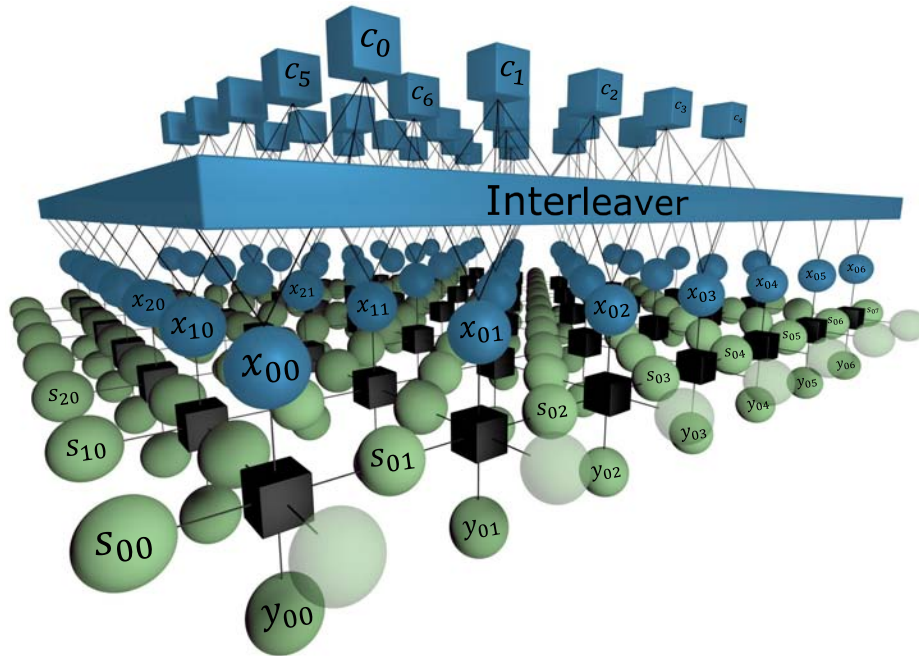
The following decoder operates in the probabilistic-domain as well as in the log-domain (see Section 7.3.3). Thus the likelihoods computed based on Equation (7.9) and (7.10) are passed to the decoder together with the SDs. The SDs are thereby computed based on Equation (3.12), and the times that the channel-model is assumed to be in state 1 and state 2, are computed by Equation (3.4) and (3.5), respectively.

$$L(x_{r,c}) = t_1 \log \frac{P(y_{r,c} | x_{r,c} = 0, s_{r,c} = \mathfrak{s}_1)}{P(y_{r,c} | x_{r,c} = 1, s_{r,c} = \mathfrak{s}_1)} + t_2 \log \frac{P(y_{r,c} | x_{r,c} = 0, s_{r,c} = \mathfrak{s}_2)}{P(y_{r,c} | x_{r,c} = 1, s_{r,c} = \mathfrak{s}_2)}. \quad (7.11)$$

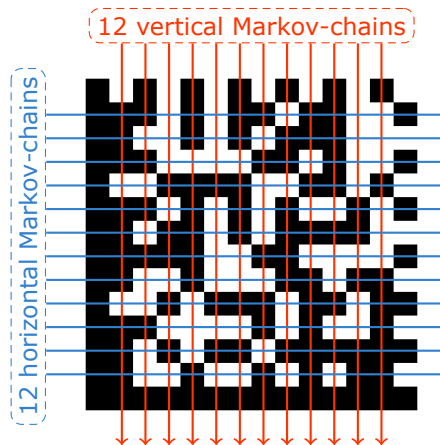
For the computation of  $t_1$  and  $t_2$ , the transition probability matrix  $P_{init}$  by which the decoder is initialized is utilized.

### 7.3 Decoder design

Usually estimation-decoding (Chapter 6) is applied to time dependent and thus one-dimensional communication systems. When considering the application of estimation-decoding together with 2D barcodes, the one-dimensional timescale turns into a geometry of two dimensions, and it is not possible to describe the state-transitions with only one Markov-chain anymore. For this reason, the single Markov-chain is replaced by several Markov-chains that are assigned to each row and each column of the 2D barcode's data region, respectively. This yields a 2D hidden Markov model (HMM) which is depicted in the example in Figure 7.12 where twelve horizontal and twelve vertical Markov-chains are assigned to the rows and columns of the 2D barcode's data region. So the state-estimation referring to a single module is based on two crossing Markov-chains, a vertical Markov-chain and a horizontal Markov-chain. Based on the 2D HMM, the Markov-LDPC factor graph shown in Figure 6.1 turns into a 3D graph and is depicted in Figure 7.13.

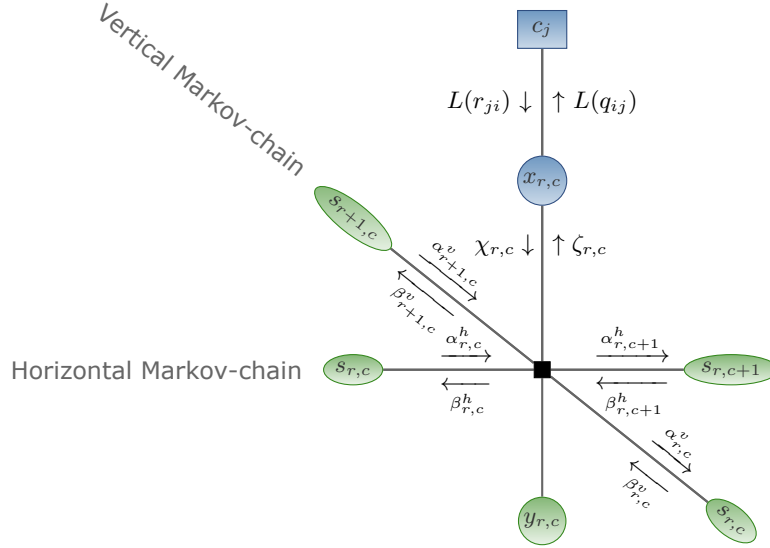


**Figure 7.13:** Markov-LDPC factor graph based on the 2D-Hidden Markov Model in Figure 7.12.



**Figure 7.12:** 2D HMM.

The decoding algorithm that operates on the graph in Figure 7.13 is derived from the new estimation-decoding variant in Section 6.4. The developed decoder for LDPC-based 2D barcodes is denoted as ED2D algorithm, which stands for estimation-decoding in two dimensions. The two dimensions thereby refer to the underlying 2D HMM. The messages sent between the nodes of one sector during the ED2D decoding can be seen in Figure 7.14.



**Figure 7.14:** Messages in one sector of the ED2D graph depicted in Figure 7.13.

The LDPC subgraph is comprised of the symbol-nodes  $x$  and the check-nodes  $c$  where the state-nodes  $s$  and the channel-nodes (black squares) belong to the Markov subgraph.  $r$  and  $c$  are the indices for the rows and columns of the 2D HMM.  $y$  stands for the correlation coefficients computed based on Equation (7.5). The noise added to the binary value of a module in row  $r$  and column  $c$  of a 2D barcode's data region is assumed to stem either from the good sub-channel ( $s_{r,c} = s_1$ ) or the bad sub-channel ( $s_{r,c} = s_2$ ), depending on the state that the state-node  $s_{r,c}$  is estimated to be in.  $\alpha$  and  $\beta$  represent the forward and backward messages of the Forward-Backward algorithm, respectively. The interface between the LDPC subgraph and the 2D HMM are the channel-messages  $\zeta$ , sent from the 2D HMM to the LDPC subgraph, and the messages  $\chi$ , passed from the LDPC subgraph to the 2D HMM. The messages are computed as follows.

### 7.3.1 Messages of the LDPC subgraph

The messages  $L(q_{ij})$  and  $L(r_{ji})$  belong to the LDPC subgraph and are computed according to Equation (4.18) and ((4.16) or (4.21)), respectively. The assignment of the index  $i$  to the pair of indices  $r$  and  $c$  is thereby done according to the applied symbol-placement.

### 7.3.2 Messages of the Markov subgraph

Forward-messages  $\alpha^h$  and  $\alpha^v$  for the horizontal and vertical Markov chains, respectively:

$$\alpha_{r,c+1}^h(s_{r,c+1}) = \sum_{s_{r,c} \in \mathcal{S}} P(s_{r,c+1} | s_{r,c}) \alpha_{r,c}^h(s_{r,c}) \sum_{x_{r,c} \in \{0,1\}} P(x_{r,c} | \chi_{r,c}) P(y_{r,c} | x_{r,c}, s_{r,c}), \quad (7.12a)$$

$$\alpha_{r+1,c}^v(s_{r+1,c}) = \sum_{s_{r,c} \in \mathcal{S}} P(s_{r+1,c} | s_{r,c}) \alpha_{r,c}^v(s_{r,c}) \sum_{x_{r,c} \in \{0,1\}} P(x_{r,c} | \chi_{r,c}) P(y_{r,c} | x_{r,c}, s_{r,c}). \quad (7.12b)$$

Backward-messages  $\beta^h$  and  $\beta^v$  for the horizontal and vertical Markov chains, respectively:

$$\beta_{r,c}^h(s_{r,c}) = \sum_{s_{r,c+1} \in \mathcal{S}} P(s_{r,c+1} | s_{r,c}) \beta_{r,c+1}^h(s_{r,c+1}) \sum_{x_{r,c} \in \{0,1\}} P(x_{r,c} | \chi_{r,c}) P(y_{r,c} | x_{r,c}, s_{r,c}), \quad (7.13a)$$

$$\beta_{r,c}^v(s_{r,c}) = \sum_{s_{r+1,c} \in \mathcal{S}} P(s_{r+1,c} | s_{r,c}) \beta_{r+1,c}^v(s_{r+1,c}) \sum_{x_{r,c} \in \{0,1\}} P(x_{r,c} | \chi_{r,c}) P(y_{r,c} | x_{r,c}, s_{r,c}) \quad (7.13b)$$

where  $P(s_{r,c+1} | s_{r,c})$  and  $P(s_{r+1,c} | s_{r,c})$  are the transition probabilities  $p_{kl}$  of the applied 2-state Markov-modulated channel-model developed in Section 7.2. The channel-model is also the basis for the computation of  $P(y_{r,c} | x_{r,c}, s_{r,c})$  according to Equation (7.9) and (7.10).

### 7.3.3 The interface messages

The channel-messages  $\zeta$  passed to the LDPC subgraph are computed based on the messages  $\alpha^h$  and  $\beta^h$  of the horizontal Markov-chains and the messages  $\alpha^v$  and  $\beta^v$  of the vertical Markov-chains:

$$\zeta_{r,c} = \log \frac{P(x_{r,c} = 0 | \alpha_{r,c}^h(s_{r,c}) \beta_{r,c+1}^h(s_{r,c+1}))}{P(x_{r,c} = 1 | \alpha_{r,c}^h(s_{r,c}) \beta_{r,c+1}^h(s_{r,c+1}))} + \log \frac{P(x_{r,c} = 0 | \alpha_{r,c}^v(s_{r,c}) \beta_{r+1,c}^v(s_{r+1,c}))}{P(x_{r,c} = 1 | \alpha_{r,c}^v(s_{r,c}) \beta_{r+1,c}^v(s_{r+1,c}))} \quad (7.14)$$

with

$$P(x_{r,c} = 0 | \alpha_{r,c}^h(s_{r,c}), \beta_{r,c+1}^h(s_{r,c+1})) = \sum_{s_{r,c} \in \mathcal{S}} \sum_{s_{r,c+1} \in \mathcal{S}} P(y_{r,c} | x_{r,c} = 0, s_{r,c}) P(s_{r,c+1} | s_{r,c}) \alpha_{r,c}^h(s_{r,c}) \beta_{r,c+1}^h(s_{r,c+1}) \quad (7.15a)$$

and

$$P(x_{r,c} = 0 | \alpha_{r,c}^v(s_{r,c}), \beta_{r+1,c}^v(s_{r+1,c})) = \sum_{s_{r,c} \in \mathcal{S}} \sum_{s_{r+1,c} \in \mathcal{S}} P(y_{r,c} | x_{r,c} = 0, s_{r,c}) P(s_{r+1,c} | s_{r,c}) \alpha_{r,c}^v(s_{r,c}) \beta_{r+1,c}^v(s_{r+1,c}). \quad (7.15b)$$

The message  $\chi_r$ , sent from the LDPC subgraph to the Markov subgraph, is represented by the SD computed by Equation (4.19) and is in the log-domain. Since the computations on the Markov subgraph are processed in the probabilistic-domain, the messages have to be transferred according to

$$P(x_{r,c} | \chi_{r,c}) = \begin{cases} \frac{1}{2} + \frac{1}{2} \tanh \frac{\chi_{r,c}}{2} & , x_{r,c} = 0; \\ \frac{1}{2} - \frac{1}{2} \tanh \frac{\chi_{r,c}}{2} & , x_{r,c} = 1. \end{cases} \quad (7.16)$$

Because the values of the 2D barcode's finder pattern are known, the corresponding messages  $\chi$  do not change during the iterative ED2D decoding so that

$$P(x_{r,c} | \chi_{r,c}) = \begin{cases} 0 & , x_{r,c} = 0 \\ 1 & , x_{r,c} = 1 \end{cases} \quad \forall x_{r,c} \in \mathcal{F}_1 \quad (7.17a)$$

and

$$P(x_{r,c} | \chi_{r,c}) = \begin{cases} 1 & , x_{r,c} = 0 \\ 0 & , x_{r,c} = 1 \end{cases} \quad \forall x_{r,c} \in \mathcal{F}_0 \quad (7.17b)$$

with  $\mathcal{F}_1 = \{\text{one-modules of the finder pattern}\}$  and  $\mathcal{F}_0 = \{\text{zero-modules of the finder pattern}\}$ . This means that the initial conditions and the end conditions of the several Markov-chains are known a priori.

### 7.3.4 Reestimation of the transition probabilities

Since the transition probabilities  $p_{kl}$  between the two sub-channels are not known, the reestimation procedure explained in Section 6.4.1 is used for the ED2D decoder. The decoder is initialized with the transition probability matrix  $P_{init}$ . Then the transition probability matrix is reestimated in each decoding iteration  $t + 1$  based on the reestimation  $\hat{P}_t$  of the previous iteration.

The probability

$$\gamma_{r,c}^t(k) = P(s_{r,c} = s_k | \mathbf{y}, \hat{P}_t) \quad (7.18a)$$

of being in state  $s_k$  in row  $r$  and column  $c$  and iteration  $t$ , given the received codeword  $\mathbf{y}$  and  $\hat{P}_t$ , now has to be calculated based on the horizontal Markov-chains as well as on the vertical Markov-chains:

$$\gamma_{r,c}^{h,t}(k) = \frac{\alpha_{r,c}^{h,t}(k) \beta_{r,c}^{h,t}(k)}{\sum_{k=1}^2 \alpha_{r,c}^{h,t}(k) \beta_{r,c}^{h,t}(k)}, \quad (7.18b)$$

$$\gamma_{r,c}^{v,t}(k) = \frac{\alpha_{r,c}^{v,t}(k) \beta_{r,c}^{v,t}(k)}{\sum_{k=1}^2 \alpha_{r,c}^{v,t}(k) \beta_{r,c}^{v,t}(k)}. \quad (7.18c)$$

The probability of being in state  $s_k$  in row  $r$  and column  $c$ , and in state  $s_l$  at the next module in row  $r$  and column  $c + 1$  for the horizontal Markov-chains (or in row  $r + 1$  and column  $c$  for the vertical Markov-chains), given  $\mathbf{y}$  and  $\hat{P}_t$ , is also computed for the horizontal and the vertical Markov-chains, respectively.

$$\begin{aligned} \xi_{r,c}^{h,t}(k, l) &= P(s_{r,c} = s_k, s_{r,c+1} = s_l | \mathbf{y}, \hat{P}_t) \\ &= \frac{\alpha_{r,c}^{h,t}(k) \hat{p}_{kl}^t \beta_{r,c+1}^{h,t}(l)}{\sum_{l=1}^2 \alpha_{r,c+1}^{h,t}(l) \beta_{r,c+1}^{h,t}(l)} \sum_{x_{r,c} \in \{0,1\}} P(y_{r,c} | x_{r,c}, s_{r,c} = s_k) P(x_{r,c} | \chi_{r,c}), \end{aligned} \quad (7.19a)$$

$$\begin{aligned} \xi_{r,c}^{v,t}(k, l) &= P(s_{r,c} = s_k, s_{r+1,c} = s_l | \mathbf{y}, \hat{P}_t) \\ &= \frac{\alpha_{r,c}^{v,t}(k) \hat{p}_{kl}^t \beta_{r+1,c}^{v,t}(l)}{\sum_{l=1}^2 \alpha_{r+1,c}^{v,t}(l) \beta_{r+1,c}^{v,t}(l)} \sum_{x_{r,c} \in \{0,1\}} P(y_{r,c} | x_{r,c}, s_{r,c} = s_k) P(x_{r,c} | \chi_{r,c}). \end{aligned} \quad (7.19b)$$

The transition probability estimates  $\hat{p}_{kl}^{t+1}$  are then obtained by

$$\hat{p}_{kl}^{t+1} = \frac{\text{expected number of transitions from } s_k \text{ to } s_l \text{ in iteration } t}{\text{expected number of transitions from } s_k \text{ in iteration } t} \quad (7.20a)$$

which is computed based on Equation (7.18) and (7.19) according to

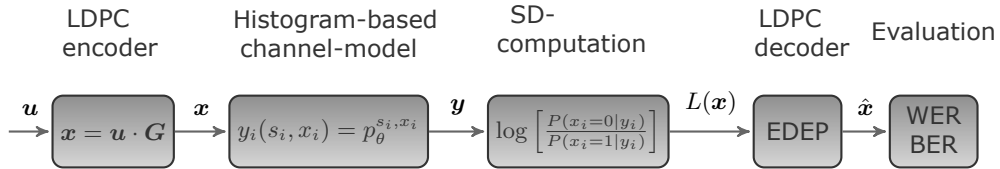
$$\hat{p}_{kl}^{t+1} = \frac{\sum_{r=0}^{R-1} \sum_{c=0}^{C-1} (\xi_{r,c}^{h,t}(k,l) + \xi_{r,c}^{v,t}(k,l))}{\sum_{r=0}^{R-1} \sum_{c=0}^{C-1} (\gamma_{r,c}^{h,t}(k) + \gamma_{r,c}^{v,t}(k))} \quad (7.20b)$$

with  $R$  and  $C$  being the number of rows and columns in the data region of the 2D barcode, respectively.

## 7.4 Design of irregular LDPC codes for 2D barcodes

In this Section, the design of irregular LDPC codes for the application on 2D barcodes is explained. The design is done by means of the DHS-based optimization method developed in Section 5.1. The irregular LDPC code is thereby designed for the usage with a certain channel-model. Considering the application of the designed LDPC code on 2D barcodes, the histogram-based 2-state channel-model constructed in Section 7.2 is chosen. During the design process the LDPC codes are evaluated by computing the word error ratio (WER) according to a time-dependent and thus one-dimensional communication system. However, the application of LDPC-based 2D barcodes that the LDPC code is designed for is a geometry of two dimensions. Even though, it is assumed that a LDPC code designed for the 2D barcode's channel-model shows a good decoding performance in the case of the real application (i.e. LDPC-based 2D barcodes) as well.

As explained in Section 5.1.2, the function evaluations for the vertices are represented by the computation of the WER obtained by means of a simulation. Considering the design for the histogram-based 2-state channel-model of Section 7.2, the simulation slightly changes compared to the description in Section 4.6. This is due to the histograms  $h_{\kappa}^{s_i, x_i}$  that the applied channel-model is based on. The simulation that is conducted in order to compute the WER is depicted in Figure 7.15.

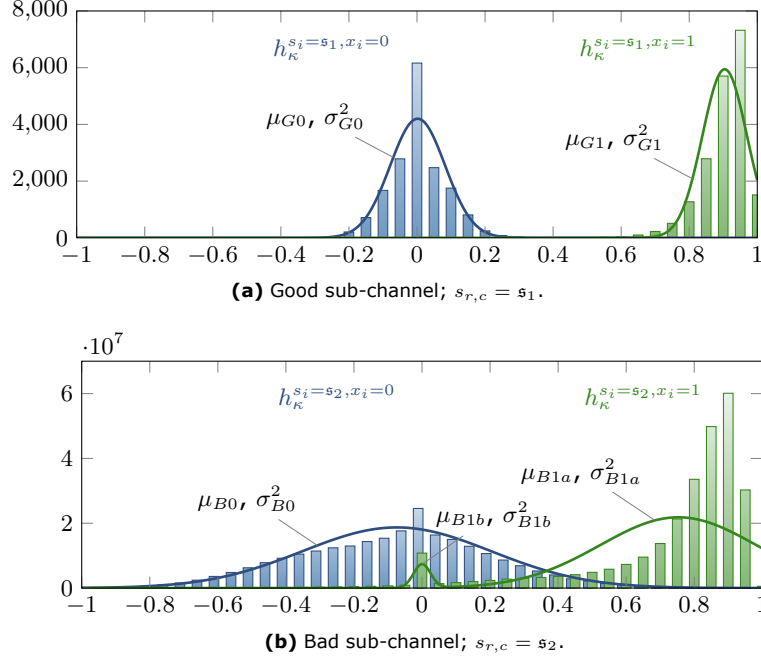


**Figure 7.15:** Computation of the WER by means of a histogram-based channel-model.

The LDPC encoder is applied since the histogram-based 2-state channel-model is asymmetric. The information word  $u$  to be encoded is thereby uniformly distributed random binary information.

### 7.4.1 Application of the 2D barcode's channel-model

The channel-model is applied in order to get the received data word  $y$  corresponding to the sent code word  $x$ . Since only one irregular LDPC code is designed, no matter if the 2D barcode that will apply the LDPC code is captured in a dark field or in a bright field, a channel-model is utilized computed by means of the channel-models for the bright field case (Figure 7.10) and the dark field case (Figure 7.11). The channel-model can be seen in Figure 7.16.



**Figure 7.16:** 2-state histogram-based channel-model used to design irregular LDPC codes for 2D barcodes.

To best represent the channel-model, a pool  $p_{\theta}^{s_i, x_i} = \{p_1^{s_i, x_i}, \dots, p_M^{s_i, x_i}\}$  of  $M$  numbers is created for each of the four histograms  $h_{\kappa}^{s_1, x_i=0}$ ,  $h_{\kappa}^{s_1, x_i=1}$ ,  $h_{\kappa}^{s_2, x_i=0}$  and  $h_{\kappa}^{s_2, x_i=1}$ <sup>2</sup>. The pools are obtained by first normalizing the histograms according to

$$n(h_{\kappa}^{s_i, x_i}) = \frac{h_{\kappa}^{s_i, x_i}}{\sum_{k=1}^{\mathcal{H}} h_{\kappa}^{s_i, x_i}} \quad (7.21)$$

with  $n(\cdot)$  being the normalization and  $\mathcal{H}$  the total number of bins. Then one computes

$$h_{\kappa}^{s_i, x_i} = n(h_{\kappa}^{s_i, x_i}) \cdot M. \quad (7.22)$$

The  $M$  numbers of each pool are then obtained by generating  $h_{\kappa}^{s_i, x_i}$  random numbers for each  $\kappa = \{1, \dots, \mathcal{H}\}$  in the interval of the current bin. This means that a histogram of one of the pools  $p_{\theta}^{s_i, x_i}$  would look like the histogram  $h_{\kappa}^{s_i, x_i}$  that it represents.

When considering a sent code word  $x$ , a received data word  $y$  is obtained by the following assignment.

$$y_i(s_i, x_i) = p_{\theta}^{s_i, x_i} \quad \forall i; \theta \leftarrow \text{random}[1, M]. \quad (7.23)$$

The channel-model is initialized with the transition probabilities  $p_{12} = 0.1$  and  $p_{21} = 0.15$ . If the channel-model is in the state  $s_i = s_1$ , a value is randomly taken either of the pool  $p_{\theta}^{s_1, x_i=0}$  or of the pool  $p_{\theta}^{s_1, x_i=1}$  and assigned to  $y_i$ , depending on  $x_i$ . For the state  $s_i = s_2$  that represents the bad channel, the value is either taken of the pool  $p_{\theta}^{s_2, x_i=0}$  or of the pool  $p_{\theta}^{s_2, x_i=1}$ , depending on  $x_i$ .

<sup>2</sup>Instead of the indices  $r$  and  $c$  used in Section 7.2 the index  $i$  is used here due to the function-evaluations that are based on a one-dimensional communication system considering the design process.



### 7.4.2 Soft-decisions and decoding

Based on the received data word  $\mathbf{y}$ , the SDs are computed according to Equation (7.9) and (7.10) except that the index  $i$  is used instead of the indices  $r$  and  $c$ . The values for the applied Gaussian approximations in Figure 7.16 can be seen in Table 7.9.

Sub-channel	1-modules		0-modules	
	mean	variance	mean	variance
Good	$\mu_{G1} = 0.9042$	$\sigma_{G1}^2 = 0.0043$	$\mu_{G0} = 0.002$	$\sigma_{G0}^2 = 0.0065$
Bad	$\mu_{B1a} = 0.7531$	$\sigma_{B1a}^2 = 0.0528$	$\mu_{B0} = -0.0741$	$\sigma_{B0}^2 = 0.0772$
	$\mu_{B1b} = 0.0001$	$\sigma_{B1b}^2 = 0.0006$		

**Table 7.9:** Means and variances of the Gaussian PDFs in Figure 7.16 that represent the channel-model for 2D barcodes.

The SDs are then further processed by the following LDPC decoder. The latter is the estimation-decoding and estimation of the transition-probabilities (EDEP) decoder of Section 6.4 that includes the reestimation of the transition probabilities (see Section 6.4.1) and the noise estimation of Section 6.4.2. The decoder is thereby initialized with

$$\mathbf{P}_{init} = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.1 \\ 0.15 & 0.85 \end{pmatrix}.$$

### 7.4.3 Design results

An irregular LDPC of length  $n = 576$  and with a code rate of  $R = 0.6111$  has been designed by means of the method explained above. The resulting symbol-node degree distribution (SNDD) of the design was  $\lambda(x) = 0.311626x^2 + 0.362298x^3 + 0.083591x^4 + 0.093844x^5 + 0.033642x^6 + 0.023253x^7 + 0.013902x^8 + 0.025308x^9 + 0.008483x^{10} + 0.010685x^{11} + 0.008096x^{12} + 0.00594x^{13} + 0.016579x^{14} + 0.002753x^{15}$ . Based on this SNDD, a PCM was constructed and a simulation was conducted according to Figure 7.15 with the same channel-model and decoder used for the design. Contrary to the simulation in Section 5.2, the WER was not computed for several values of  $E_b/N_0$ , but instead for several values of  $t_1$ .

The channel-model is parameterized with the transition probabilities  $p_{12}$  and  $p_{21}$  computed based on the given  $t_1$  so that Equation (3.4) is fulfilled. The values used for the simulation can be seen in Table 7.10.

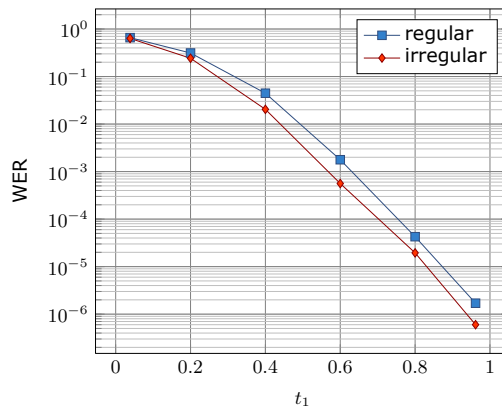
$t_1$	$p_{21}$	$p_{12}$
0.04	0.01	0.25
0.2	0.05	0.2
0.4	0.1	0.15
0.6	0.15	0.1
0.8	0.2	0.05
0.96	0.25	0.01

**Table 7.10:** Transition probabilities  $p_{21}$  and  $p_{12}$ , depending on the values for  $t_1$  that the simulation is based on.

Since the channel-model is based on a random process parameterized with  $p_{21}$  and  $p_{12}$ , the number of times that the channel-model is in state  $s_1$  and  $s_2$  are counted during the simulation. The real  $t_1$  that the simulation is based on is then computed by

$$t_1^{real} = \frac{N_{s_1}}{N_{s_1} + N_{s_2}} \quad (7.24)$$

with  $N_{s_1}$  and  $N_{s_2}$  being the counts for  $s_1$  and  $s_2$ , respectively. The WERs computed based on the simulation are then plotted over the appropriate values of  $t_1^{real}$ . Figure 7.17 shows the results for the designed irregular LDPC code compared to the results of a regular LDPC code. It can be seen that the irregular LDPC code is up to 0.3 dB better than the regular LDPC code.



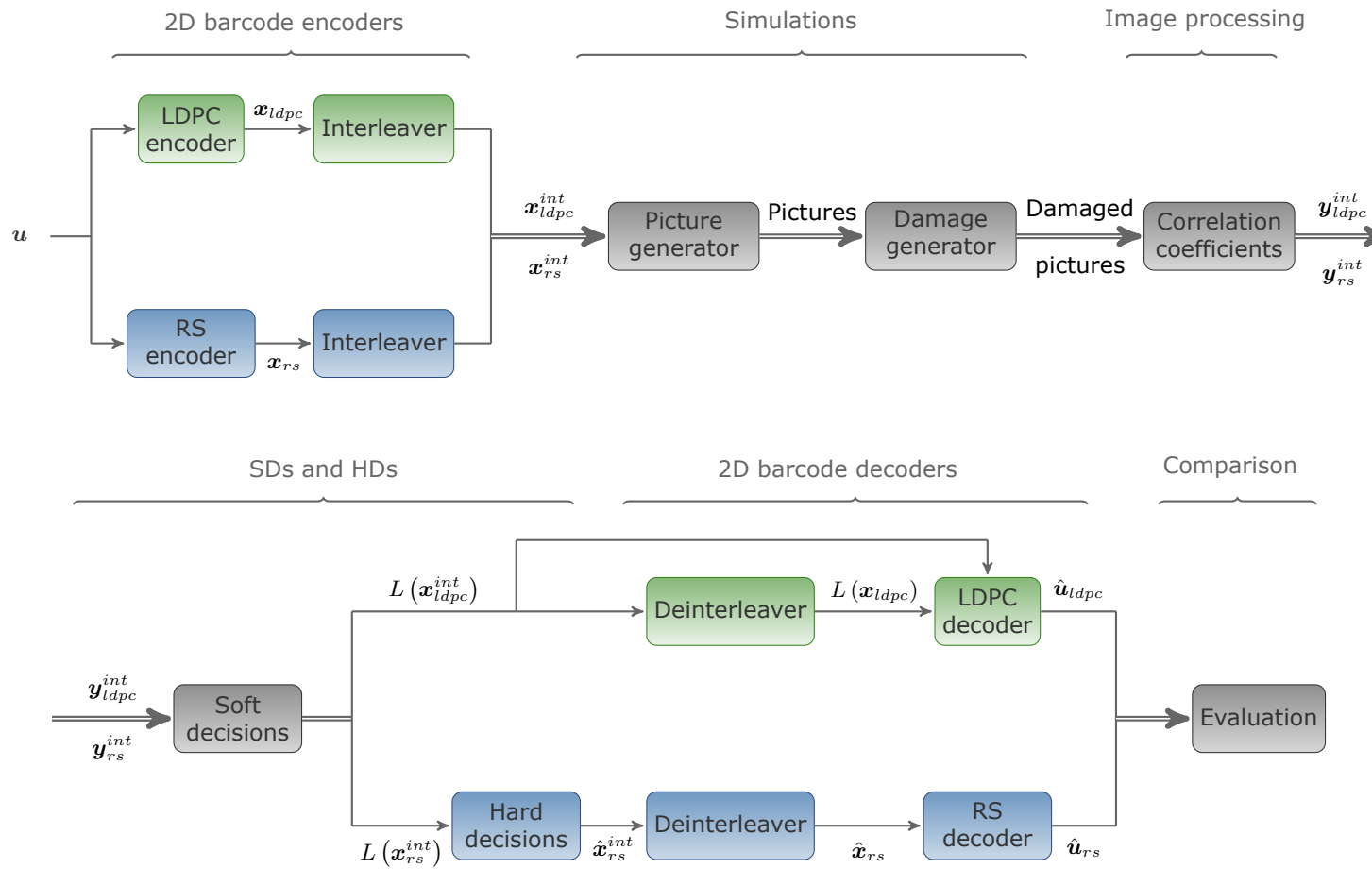
**Figure 7.17:** WER of a regular LDPC code and a irregular LDPC code designed for the 2D barcode's channel-model of Section 7.2.

## Chapter 8

# Evaluation

### 8.1 Test environment

The basis for a significant evaluation considering different types of 2D barcodes is to provide a fair comparison as well as a sufficiently big statistic. For that, a test environment is established that enables a comparison between different 2D barcode variants. Figure 8.1 shows the comparison of low-density parity-check (LDPC)-based 2D barcodes with the Reed-Solomon (RS)-based Data Matrix code (DMC) by means of the test environment. The green blocks and blue blocks are only processed in the case of LDPC-based 2D barcodes and DMCs, respectively. The gray blocks refer to processes that are exactly the same for all tested 2D barcode versions. The test environment is explained in the following based on the block diagram in Figure 8.1.



**Figure 8.1:** Test environment depicted based on a comparison between the Data Matrix code and the LDPC-based 2D barcode developed in this thesis.

### 8.1.1 2D barcode encoders

The LDPC encoding is processed according to Equation (4.14). This is possible since all LDPC codes used in this work are based on a parity-check matrix (PCM) constructed by the progressive-edge-growth (PEG) algorithm described in Algorithm 4 that yields a PCM according to Equation (4.13).

The bits of the LDPC code word  $\mathbf{x}_{ldpc}$  are then placed in the 2D barcode's data region by means of the intelligent interleaver developed in Section 7.1.  $\mathbf{x}_{ldpc}^{int}$  represents the matrix that describes the 2D barcode and thus the 2D barcode's finder pattern including the interleaved version of  $\mathbf{x}_{ldpc}$  located in the 2D barcode's data region.

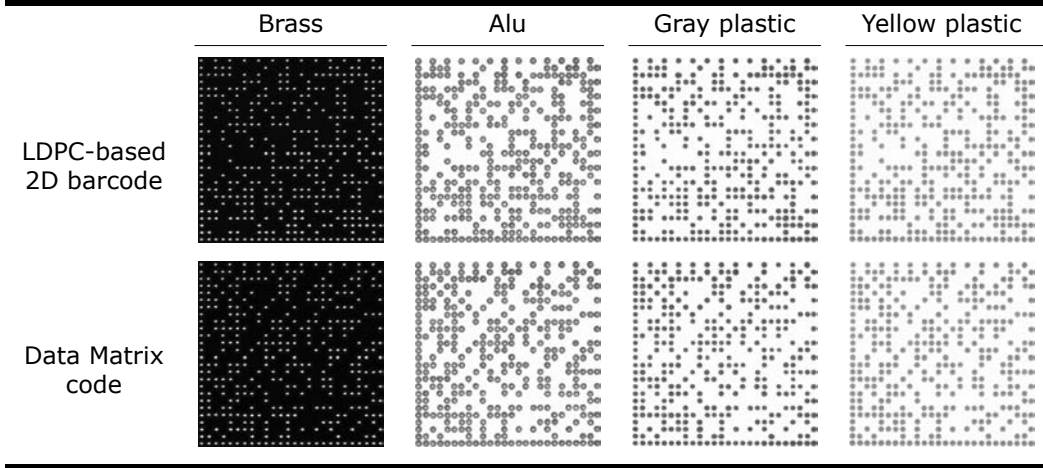
The RS encoding and interleaving in the case of the DMC are done as defined in the standard [4].

### 8.1.2 Simulations

The normal procedure would then be to emboss the 2D barcode defined by the matrix  $\mathbf{x}_{ldpc}^{int}$  (or  $\mathbf{x}_{rs}^{int}$  for the DMC) on a desired part. This has been done in Section 7.2 in order to find a channel-model for 2D barcodes. In contrast to that, a simulation of pictures of 2D barcodes is utilized instead of taking real pictures of the embossed 2D barcodes. This is done in order to equalize the conditions as much as possible considering the different types of 2D barcodes that are to be compared. A simulated picture is obtained by taking a real picture of a 2D barcode apart into disjointed single modules. This is done in a row by row manner where the one-modules and the zero-modules of the data region are separated and stored in two pools, respectively. The finder pattern of the 2D barcode's picture is kept as is. A simulated picture is then created based on the binary matrix  $\mathbf{x}_{ldpc}^{int}$  (or  $\mathbf{x}_{rs}^{int}$  for the DMC) that describes the 2D barcode. The stored finder pattern forms the frame for the simulated picture. Then in a row by row manner again, a one-module or a zero-module is taken out of the appropriate pool, depending on the entry in the matrix. So the simulated pictures of different 2D barcodes show as similar an outer appearance as possible since the finder pattern is exactly the same and the modules stem from the same pools. Hence possible variances that occur due to the embossing and the acquisition and that are caused by the illumination, the material and the milling are eliminated. Table 8.1 shows some examples of simulated pictures for different kinds of materials. The pictures of the LDPC-based 2D barcodes and the DMCs have thereby been generated based on the same pools of modules.

A comparison of a simulated picture and a real picture that are based on the same 2D barcode and the same material is not shown. This is because the pictures would look exactly the same if the real picture has been the source for the modules in the pools.

The next step is to add damages to the generated pictures in order to best reflect the real channel that 2D barcodes have to deal with in industrial environments. The damages are added by means of the water drop and oil drop simulations introduced in Section 7.2.3. This way it is possible to affect different 2D barcode variants with exactly the same damage. This is necessary in order to provide the same conditions for a comparison of the different channel-codes inside the 2D barcodes. Table 8.2 shows water drop and oil drop simulations added to 2D barcodes that are exactly the same for the LDPC-based version and the RS-based DMCs. The pictures are thereby generated based on the picture simulation and are the same as the ones shown in Table 8.1.



**Table 8.1:** Simulated pictures for different materials. The pictures for the LDPC-based 2D barcodes and the RS-based DMCs were generated based on the same pools of modules and thus show as similar an outer appearance as possible.

### 8.1.3 Image processing

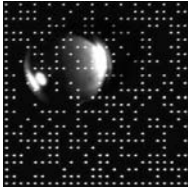
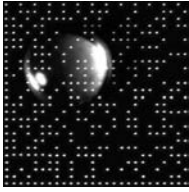
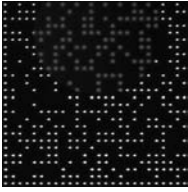
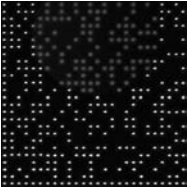
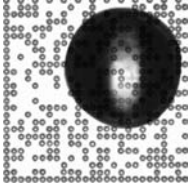
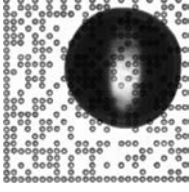
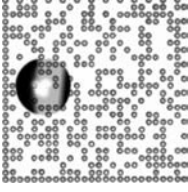
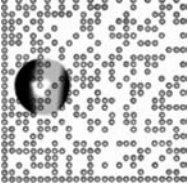
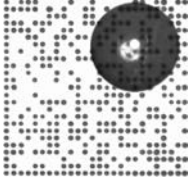
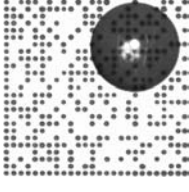
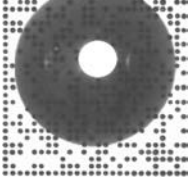
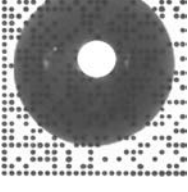
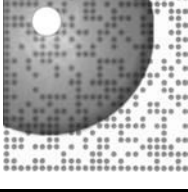
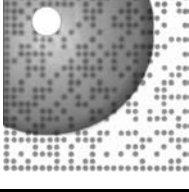
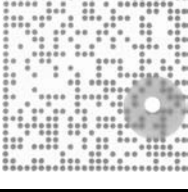
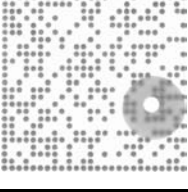
The image processing part of the test environment is represented by the computation of the correlation coefficients according to Equation (7.5). To each module in the 2D barcode a correlation coefficient is assigned that measures the similarity to a reference one-module created based on the one-modules that are part of the finder pattern. The correlation coefficients represent the received data word  $\mathbf{y}$  in a usual communication system. They are denoted as  $\mathbf{y}_{ldpc}^{int}$  and  $\mathbf{y}_{rs}^{int}$  since they are still ordered according to the 2D barcodes structure.

### 8.1.4 Soft-decisions and hard-decisions

Afterwards, the soft-decisions (SDs) are computed by means of the channel-model developed for 2D barcodes in Section 7.2. Since there are two versions of channel-models, one for pictures taken in a bright field (Figure 7.10) and one for dark field acquisitions (Figure 7.11), it is necessary to decide which one to use. The decision is made based on the knowledge of the modules involved in the broken border that is part of the 2D barcode's finder pattern (see Figure 2.4). First the means  $\bar{a}_{rc}$  of all modules that are part of the broken border are computed according to Equation (7.6). Then two values  $\bar{a}_1$  and  $\bar{a}_0$  are calculated to represent the mean of all one-module-means and all zero-module-means, respectively.

$$\bar{a}_1 = \frac{1}{R} \sum_{i=0}^{R/2-1} \bar{a}_{r=2i+1, c=C-1} + \frac{1}{C} \sum_{i=0}^{C/2-1} \bar{a}_{r=0, c=2i} \quad , \quad (8.1)$$

$$\bar{a}_0 = \frac{1}{R} \sum_{i=0}^{R/2-1} \bar{a}_{r=2i, c=C-1} + \frac{1}{C-2} \sum_{i=0}^{C/2-2} \bar{a}_{r=0, c=2i+1} \quad . \quad (8.2)$$

Material	Water drops		Oil drops	
	LDPC	RS	LDPC	RS
Brass				
Alu				
Plastic g				
Plastic y				

**Table 8.2:** The same pictures as shown in Table 8.1 but with added oil drop and water drop simulations that are exactly the same for the LDPC-based 2D barcodes and the RS-based DMCs.

$R$  and  $C$  are thereby the number of rows and columns in the 2D barcode. The decision for the bright field or the dark field is then made according to

$$\begin{cases} \text{Dark field} & \text{if } \bar{a}_1 > \bar{a}_0; \\ \text{Bright field} & \text{if } \bar{a}_1 < \bar{a}_0. \end{cases} \quad (8.3)$$

This can be done since in the case of a dark field capture the means of the one-modules are greater in average than the zero-module-means because a zero-module is represented by an untouched surface that does not reflect any light into the camera and thus appears dark. But the cavity of a one-module reflects light into the camera's lens which results in a higher module-mean. For a bright field capture, the untouched surface of a zero-module reflects more light into the camera's lens than a one-module whose cavity appears dark on the picture.

The modules of the L-pattern are not considered when computing  $\bar{a}_1$  because a damage located on the L-pattern would only affect  $\bar{a}_1$ . This could result in a wrong decision. Contrary to that, the probability is high that a damage on the broken border

affects both,  $\bar{a}_1$  and  $\bar{a}_0$ . Furthermore, it is likely that a damage causes both values,  $\bar{a}_1$  and  $\bar{a}_0$  to be shifted towards 0 or 255 keeping the inequality that the decision in Equation (8.3) is based on.

After the decision for the bright field's channel-model or the dark field's channel-model has been made, the likelihoods are calculated according to Equation (7.9) and (7.10) to represent the good sub-channel and the bad sub-channel, respectively. Then the SDs are computed by means of the likelihoods according to Equation (3.12) for both sub-channels, respectively.

In the case of LDPC-based 2D barcodes, the SDs  $L(x_{ldpc}^{int})$  (i.e. the log-likelihood ratios (LLRs)) are passed to the decoder developed in Section 7.3. Since the state-estimation on the 2D hidden Markov model (HMM) is not computed in the log-domain, the likelihoods are passed to the decoder as well.

For the DMCs that are based on RS codes, one has to make hard-decisions (HDs) first which is done according to Equation (3.11). The HDs  $\hat{x}_{rs}^{int}$  are then passed to the DMC decoder.

### 8.1.5 2D barcode decoders

The decoder for LDPC-based 2D barcodes developed in Section 7.3 and denoted as estimation-decoding in 2 dimensions (ED2D) algorithm is represented by the deinterleaver and the LDPC decoder in Figure 8.1. This is a simplified illustration where the two paths that enter the LDPC decoder stand for the fact, that the interleaved version  $L(x_{ldpc}^{int})$  of the SDs as well as the SDs ordered according to the sent code word  $x_{ldpc}$  are required during the ED2D decoding. For the messages on the LDPC subgraph the MSc decoder is used with a correction factor of  $c_f = 0.45$ . For the computation on the Markov subgraph the decoder is initialized with the transition probability matrix

$$P_{init} = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{pmatrix}.$$

Since only systematic LDPC codes are utilized, the estimated information word  $\hat{u}_{ldpc}$  of length  $k$  is easily extracted out of the estimated code word  $\hat{x}_{ldpc}$  of length  $n$  by taking the last  $k$  bits (see Equation (4.12)).

The deinterleaving and RS decoding for the DMCs is done as described in the standard [4].

### 8.1.6 Comparison

The different variants of 2D barcodes are then evaluated and compared based on the estimated information words that are the decoding results of the appropriate decoders. Considering 2D barcodes in direct part mark identification (DPMI) applications, only the word error ratio (WER) is suitable to evaluate the decoding performance. In the case of a text stored in a 2D barcode and then decoded and read by a human being, the bit error ratio (BER) would also be interesting. This is because in the case of only a few bit errors a text could still be *decoded* correctly by a human being by means of the redundancy of the applied language. But when considering the application of 2D barcodes in industrial environments, the information stored inside of the barcodes is usually further processed by computer-based systems. A false digit could result from only one single bit error and a resulting error in an article number for example would then refer to a completely different item. Thus only word errors are considered when evaluating the error-correction capabilities of the tested 2D barcode variants.



## 8.2 Test procedure

The evaluation of the different 2D barcode variants is done by means of the test environment of Section 8.1 that is depicted in Figure 8.1.

The size of all tested 2D barcodes is  $26 \times 26$  modules which means that the data region has a size of  $24 \times 24$ . Hence there is space for 576 bits to be stored in the data region. Since only one code word is used in the case of LDPC-based 2D barcodes, all utilized LDPC codes have a block length of  $n = 576$ . The code rate of the RS code used inside a DMC of size  $26 \times 26$  is  $R = 0.6111$  according to the standard (see Table 11 in [4], page 16). Thus all LDPC codes have been constructed with the same code rate  $R = 0.6111$ .

The information word  $u$ , encoded inside the 2D barcodes, is the same for all versions. It is an ASCII-encoded citation of Claude Shannon namely *Information is the resolution of uncertainty*.

The testing is done based on four different materials: brass, aluminum, gray plastic and yellow plastic. This means that for each 2D barcode variant four pictures were generated. Then to each of the pictures 20000 different water drop simulations and 20000 different oil drop simulations were added. It was thereby ensured that the damage simulations have been exactly the same considering different barcode variants.

Before the decoding started, a HD was done, not only for the RS-based barcode but also in the case of the LDPC-based versions. By means of the HD, the number of bit errors have been computed that occurred before the actual decoding started. This offers a measure of how much the damage simulations influenced the 2D barcodes. The same damage pattern added to different types of 2D barcodes that store the same information, often yields a different number of pre-decoding bit errors. This is due to the different arrangement of one-modules and zero-modules.

The damage patterns that caused no pre-decoding bit errors in the case of all examined 2D barcode variants have not been considered. This is why the number of decodings referring to water drops and oil drops is lower than 20000.

The pre-decoding bit errors are also used to evaluate the decoding performance of the several 2D barcode variants considering different amounts of pre-decoding bit errors. This has two main advantages:

1. All circumstances that cause unequal conditions when comparing different types of 2D barcodes are eliminated.
2. The frequency distribution of the pre-decoding bit errors is considered.

The second point is very important since due to the characteristic of the test-environment the probability that 2D barcodes are affected by only a few pre-decoding bit errors is very high compared to cases with more bit errors. Thus a 2D barcode variant that would have a decoding advantage only for more pre-decoding bit errors would be disadvantaged by not considering the frequency distribution of the pre-decoding bit errors.

### 8.3 The effectiveness of the optimized symbol-placement

In this section, it is checked if an increasing error-correction capability is obtained when placing the symbols of a LDPC code word in the 2D barcode's data region by means of the intelligent interleaver developed in Section 7.1.3. For that, two variants of 2D barcodes are compared:

1. 2D barcode based on a regular LDPC code with an optimized interleaving.
2. 2D barcode based on the same regular LDPC code with a bad interleaving.

The symbol-node degree distribution (SNDD) of the utilized regular LDPC code is  $\lambda(x) = x^3$  which means that each symbol-node in the Tanner graph has three connected check-nodes. Since the LDPC code is exactly the same for both LDPC-based 2D barcode variants, they only differ in the way the symbols are placed in the data region.

The optimization of the symbol-placement for the first LDPC-based 2D barcode has been done according to Section 7.1.3. The results for the optimization can be seen in Section 7.1.4. Figure 7.4 shows the initial costs and the minimized costs of 15 processed optimization rounds. Since the cost  $\mathcal{C}(2DC) = 20551.4636$  obtained in the 5<sup>th</sup> optimization round is the lowest one, the corresponding symbol-placement has been chosen for the LDPC-based 2D barcode. The symbol-placement can be seen in the Appendix in Table A.2.

In the case of the second LDPC-based 2D barcode variant, the optimization method in Section 7.1.3 has been slightly modified in order to find a symbol-placement with a high cost. Table A.3 in the appendix shows the resulting degraded symbol-placement that has a cost of  $\mathcal{C}(2DC) = 24761.2034$ . Following the assumptions that the optimization method is based on (see Section 7.1), the resulting 2D barcode is supposed to have a lower error-correction capability than the first variant that benefits from the optimized symbol-placement.

The comparison was done based on the test-environment in Section 8.1 and the test procedure explained in Section 8.2. The results can be seen in Table 8.3 where the successful decodings are shown for the different materials and drop simulations, respectively. Based on the total number of decodings, the percentage of successes is also computed. The results in the case of oil drop simulations on brass are not shown since all 2D barcodes were successfully decoded.

Table 8.3 shows that in total 87% could be successfully decoded in the case of the optimized symbol-placement. For the degraded symbol-placement, only 82% succeeded in decoding. Thus a gain of 5% is obtained with the optimized symbol-placement compared to the bad placement. But in this case, the frequency distribution of the pre-decoding bit errors is not considered.

This is in contrast to the results in Figure 8.2, where the decoding performance is analyzed based on the pre-decoding bit errors. This ensures a fair comparison and reveals the advantage of the intelligent interleaving in detail. Figure 8.2 shows the percentage of successful decodings depending on different amounts of pre-decoding bit errors. The bars referring to the first point on the very left side (10 pre-decoding bit errors) present the number of successful decodings divided by the total number of decodings where 0 to 10 pre-decoding bit errors occurred. The next point refers to all cases in which 10 to 20 pre-decoding bit errors complicated the following decoding and so on.

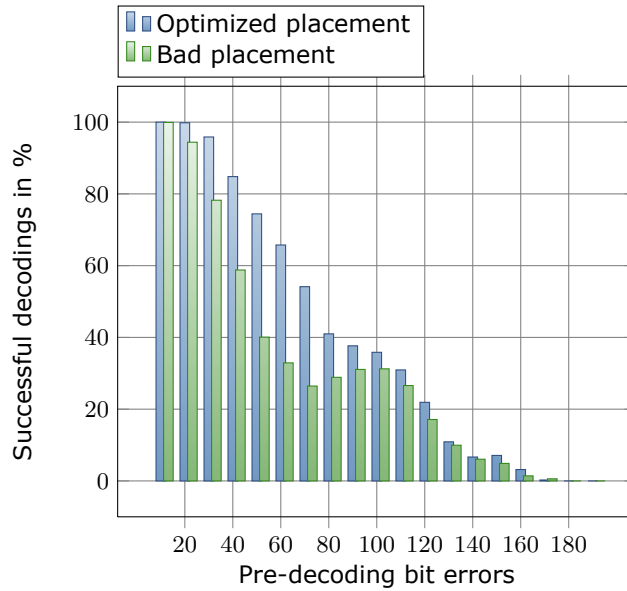
The results in Figure 8.2 show that for up to 10 pre-decoding bit errors the two interleavings yield the same decoding results of 100% successful decodings. But if more pre-decoding bit errors occur, the optimized symbol-placement clearly beats the bad placement. The gain is very high in between 30 and 80 pre-decoding bit errors.

Material	Damage	Decodings	Intelligent interleaving		Bad interleaving	
			OK	%	OK	%
Brass	Water	9460	9298	98	8853	94
Alu	Water	18568	17734	96	17562	95
	Oil	15614	12081	77	9887	63
	Total	34182	29815	87	27449	80
Plastic g	Water	17721	16717	94	16226	92
	Oil	15869	12419	78	11679	74
	Total	33590	29136	87	27905	83
Plastic y	Water	16008	15169	95	14542	91
	Oil	14421	10061	70	9558	66
	Total	30429	25230	83	24100	79
Total	Water	61757	58918	95	57183	93
	Oil	45904	34561	75	31124	68
Total		107661	93479	87	88307	82

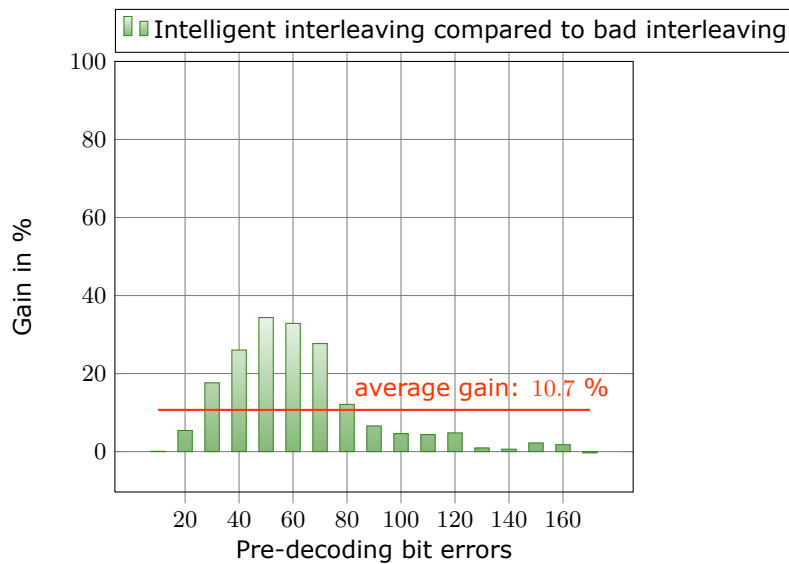
**Table 8.3:** Comparison results of two 2D barcode variants based on the same regular LDPC code. In the case of the 1<sup>st</sup> LDPC-based barcode the symbols have been placed in the data region by means of the optimization procedure of Section 7.1.3. In contrast to that the interleaving for the 2<sup>nd</sup> LDPC-based barcode has been done in order to get a high cost of the resulting symbol-placement.

This can be well seen in Figure 8.3 where the gain in decoding successes that is obtained by using the optimized symbol-placement instead of the bad placement is shown.

The results in Figure 8.3 confirm the effectiveness of the intelligent interleaving. Since an average gain of about 10.7% is obtained with the optimized symbol-placement compared to the bad placement, one can say that the decoding performance of LDPC-based 2D barcodes increases with decreasing cost of the symbol-placement computed based on Equation (7.3).



**Figure 8.2:** Decoding results of two 2D barcode variants based on the same regular LDPC code. The two variants only differ in the way the symbols are placed in the data region.



**Figure 8.3:** Gain in decoding successes that is obtained by using the optimized symbol-placement proposed in Section 7.1.

## 8.4 The effectiveness of the ED2D decoding

In Section 7.3 the ED2D algorithm is introduced for the decoding of LDPC-based 2D barcodes. It is based on a 3D graph that is comprised of a LDPC subgraph and a Markov subgraph. The Markov subgraph is represented by the 2D HMM on which the states of the 2-state channel-model developed in Section 7.2 are estimated during the iterative ED2D decoding. The target of the following analysis is to measure the gain in decoding performance obtained by means of the ED2D algorithm's state-estimation.

For that, a 2D barcode based on a regular LDPC code is decoded with two different decoders.

1. ED2D decoder (including the state-estimation).
2. MSc decoder (without the state-estimation).

The decoding with the ED2D algorithm is done as explained in Section 7.3 where the SDs are computed as shown in Section 7.2.4.

The second decoder is represented by the MSc algorithm that is explained in Section 4.4.3. This algorithm is also integrated in the ED2D algorithm where it operates on the LDPC subgraph. But contrary to the MSc decoder, the ED2D decoder additionally includes a state-estimation on the Markov subgraph as well as a reestimation of the transition probabilities.

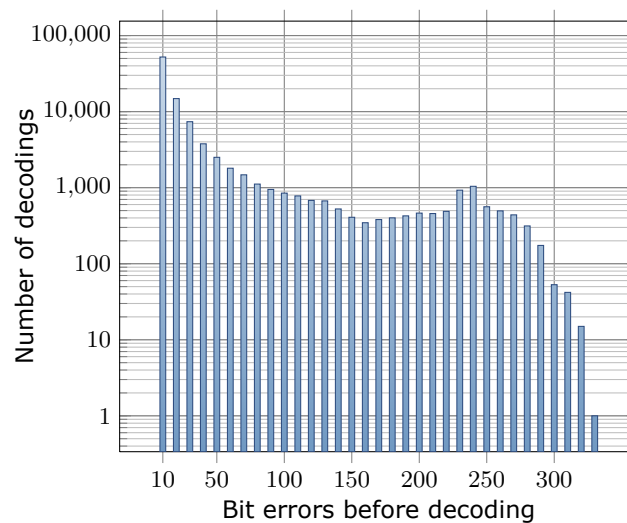
Usually, an additive white Gaussian noise (AWGN) channel is assumed when decoding with a MSc decoder. Even though the 2-state channel-model developed in Section 7.2 for 2D barcodes is used to compute the SDs according to Equation (7.11) in order to only reveal the advantage of the state-estimation.

The results of both decoder variants can be seen in Table 8.4. The frequency distribution of the pre-decoding bit errors is thereby not considered.

Material	Damage	Decodings	ED2D		MSc	
			OK	%	OK	%
Brass	Water	9460	9298	98	9268	98
Alu	Water	18568	17734	96	16920	91
	Oil	15614	12081	77	11013	71
	Total	34182	29815	87	27933	82
Plastic g	Water	17721	16717	94	16772	95
	Oil	15869	12419	78	11845	75
	Total	33590	29136	87	28617	85
Plastic y	Water	16008	15169	95	15263	95
	Oil	14421	10061	70	9749	68
	Total	30429	25230	83	25012	82
Total	Water	61757	58918	95	58223	94
	Oil	45904	34561	75	32607	71
	Total	107661	93479	87	90830	84

**Table 8.4:** Comparison results of the same LDPC-based 2D barcode decoded with two different decoders. The first decoder is the ED2D decoder developed in Section 7.3. The second decoding is done without the state-estimation and thus only by means of the MSc algorithm.

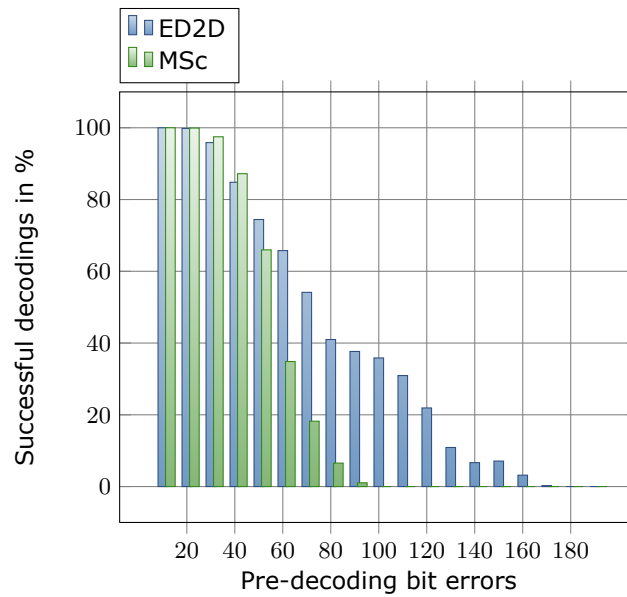
Table 8.4 shows that in total 87% could be successfully decoded by using the ED2D decoder whereas only 84% successes were obtained with the MSc algorithm. Especially in this case, it is important to additionally consider the frequency distribution of the pre-decoding bit errors that can be seen in Figure 8.4. Since both decoders had to decode based on exactly the same disturbed pictures, the number of decodings shown in Figure 8.4 is the same for the ED2D decoder and the MSc decoder. When looking at the total number of decodings for the different numbers of pre-decoding bit errors, it is obvious that the pre-decoding bit errors are not equally distributed.



**Figure 8.4:** Number of decodings of the ED2D decoder and the MSc decoder.

Figure 8.4 shows that most of the data words that the two decoder-variants had to decode were affected by up to 10 bit errors. In fact, 80.64% of all decodings were affected by 40 pre-decoding bit errors or less. But the advantage by means of a state-estimation takes only an effect if more than 40 pre-decoding bit errors occur. This can be seen in Figure 8.5 where the percentage of successful decodings is shown for different pre-decoding bit errors. This means that the results in Table 8.4 are not suitable for a fair comparison. Figure 8.5 shows that for up to 20 pre-decoding bit errors the two decoder variants yield the same decoding results (100% successes). In between 20 and 40 pre-decoding bit errors, there is a slight loss in decoding performance considering the ED2D decoder compared to the MSc decoder. But for more than 40 pre-decoding bit errors, the gain when applying the state-estimation compared to no state-estimation is increasing with the number of pre-decoding bit errors. In the case of more than 90 pre-decoding bit errors, only the ED2D decoder is capable of correctly retrieving the stored information. The results are confirmed when looking at the gain in Figure 8.6. It can be seen that in average a gain of 15.2% is obtained by means of the ED2D algorithm compared to the MSc decoder.

In the context of a ED2D - MSc comparison, it is important to additionally consider the decoding speed of the two decoder variants. The times required for the decoding in the case of decoding-successes are shown in Figure 8.7. For the evaluation of the results, it is important to mention that the test-environment is realized in Matlab (Version 7.7.0.471) and all evaluations have been performed on a 32bit Windows 7



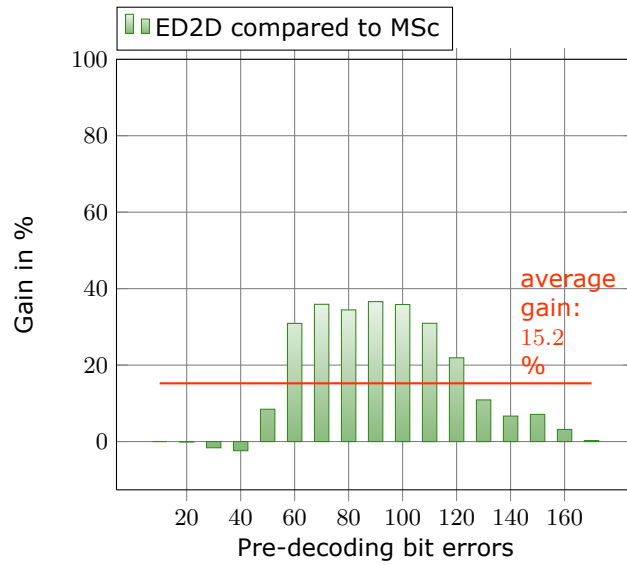
**Figure 8.5:** Decoding results of a LDPC-based 2D barcode decoded with two different decoders, the ED2D decoder developed in Section 7.3 and the MSc decoder, respectively.

machine<sup>1</sup>.

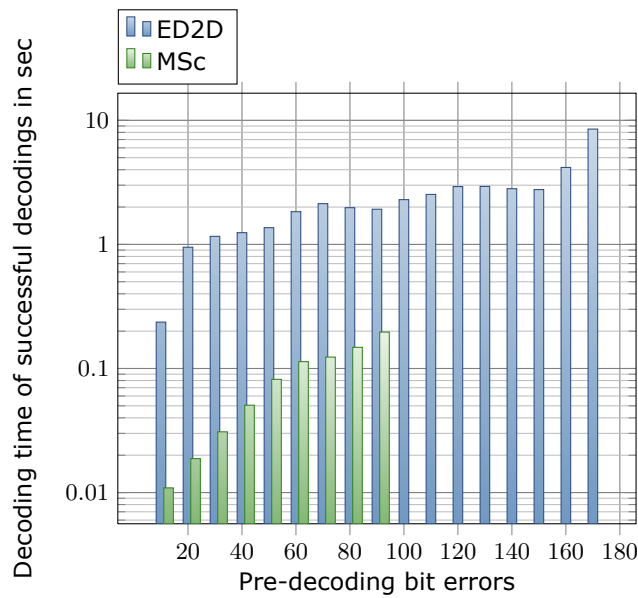
The decoding times in Figure 8.7 are plotted in a logarithmic scale since the decoding speed for the two decoders is very different. The ED2D decoding requires at least 10 times more time compared to the MSc decoding which means that the computations on the 2D HMM are very time consuming. On average, one ED2D iteration takes 103.1 milliseconds (ms) where one iteration with the MSc-decoder only takes 6.6 ms.

One possible conclusion could be to choose the decoder depending on the desired application especially considering possible interferences that cause pre-decoding bit errors, the available hardware and security aspects.

<sup>1</sup>Processor: Intel Pentium Dual-Core E5300 processor with 2.6 GHz and 2 GB RAM.



**Figure 8.6:** Gain obtained when decoding a LDPC-based 2D barcode with the ED2D decoder instead of the MSc decoder.



**Figure 8.7:** Average decoding times for the decoding results shown in Figure 8.5.



## 8.5 Irregular versus regular

To check the error-correction capabilities of 2D barcodes based on irregular LDPC codes, the irregular LDPC code designed especially for the application with 2D barcodes in Section 7.4 is taken. The appropriate SNDD can be found in Section 7.4.3.

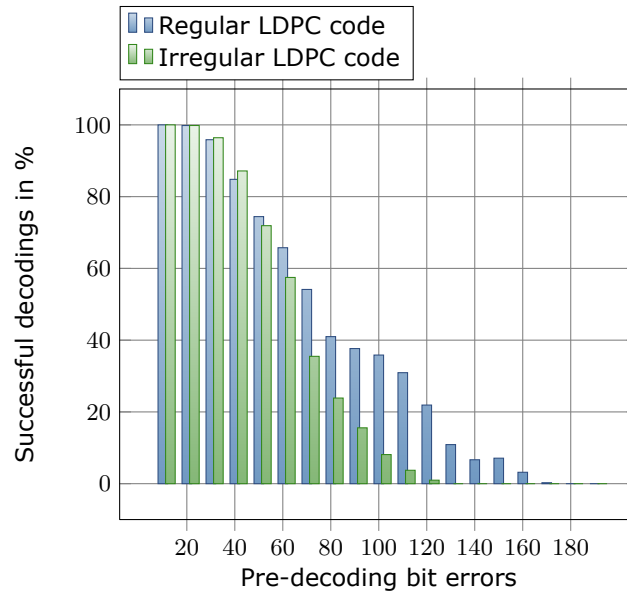
As described in Section 8.2, the code rate was chosen to be  $R = 0.6111$  and the block length  $n = 576$ . The symbol-placement has been optimized by means of the intelligent interleaver of Section 7.1.3, and can be found in the Appendix in Table A.4. The cost of the placement is  $\mathcal{C}(2DC) = 14343.14$  which is much less than  $\mathcal{C}(2DC) = 20551.4636$  in the case of the symbol-placement referring to the regular LDPC code. The reason for that is the value of  $d_T^{max}$  used when computing the cost in Equation (7.2). For the regular LDPC code, the maximum distance  $d_T(x_a, x_b)$  in the Tanner graph is  $d_t^{max} = 8$ . In the case of the irregular LDPC code, this value is only  $d_t^{max} = 6$ , and thus the first term in Equation (7.2) is smaller in average. Due to the LDPC code dependent value of  $d_t^{max}$  involved in the cost-computation, it is not possible to compare different symbol-placements based on the cost  $\mathcal{C}(2DC)$  independent of the utilized LDPC code. A comparison only makes sense for different symbol-placements that are based on the same LDPC code.

The decoding results that are based on the irregular LDPC code can be seen in Table 8.5 and Figure 8.8. The results of the regular LDPC-based 2D barcode considering the optimized symbol-placement have thereby been added for comparison purposes.

Material	Damage	Decodings	Regular LDPC		Irregular LDPC	
			OK	%	OK	%
Brass	Water	9460	9298	98	9276	98
Alu	Water	18568	17734	96	17201	93
	Oil	15614	12081	77	11511	74
	Total	34182	29815	87	28712	84
Plastic g	Water	17721	16717	94	16881	95
	Oil	15869	12419	78	11965	75
	Total	33590	29136	87	28846	86
Plastic y	Water	16008	15169	95	15279	95
	Oil	14421	10061	70	9859	68
	Total	30429	25230	83	25138	83
Total	Water	61757	58918	95	58637	95
	Oil	45904	34561	75	33335	73
	Total	107661	93479	87	91972	85

**Table 8.5:** Comparison results of two LDPC-based 2D barcodes utilizing a regular LDPC code and an irregular LDPC code, respectively. The irregular LDPC code has been designed for 2D barcodes according to Section 7.4.

Following the simulation results in Figure 7.17, one assumes that the performance based on the irregular LDPC code beats the regular LDPC code in terms of WER. But when looking at the results in Table 8.5 and Figure 8.8, one can recognize that the 2D barcode based on the irregular LDPC code is worse than the regular-based version. Table 8.5 shows that the irregular LDPC code yields 85% decoding successes which is 2% less than based on the regular LDPC code that yields 87%. In Figure 8.8, it can be



**Figure 8.8:** Decoding successes of two LDPC-based 2D barcodes utilizing a regular LDPC code and an irregular LDPC code, respectively.

seen that the irregular code is disadvantaged if more than 40 pre-decoding bit errors occur.

The inferiority of the irregular LDPC code is unexpected and might be explained by an interrelation between the number of connected symbol-nodes to a check-node and the resulting effectiveness of the symbol-placement optimization. The more connected symbol-nodes to a check-node exist, the higher the probability that a local damage influences more symbol-nodes connected to the same check-node is.

To check this assumption, four more irregular LDPC codes have been designed as described in Section 7.4. In contrast to the former design where the maximum number of connected check-nodes to a symbol-node was limited by  $d_x^{max} = 15$ , for the new designs  $d_x^{max}$  has been limited by  $d_x^{max} = 10$ ,  $d_x^{max} = 6$ ,  $d_x^{max} = 4$  and  $d_x^{max} = 3$ , respectively. This was done since the number of symbol-nodes involved in a parity-check equation decreases together with a decreasing value of  $d_x^{max}$ . The resulting SNDDs of the irregular LDPC codes can be seen in Table 8.6.

Table 8.7 shows min, max and mean values for the check-node degree (CND) (the number of symbol-nodes connected to a check-node) of the several LDPC codes. In this context a regular LDPC code with a SNDD of  $\lambda(x) = x^2$  was added as well in order to check a case with very low average CND.

For each of the LDPC codes in Table 8.7, a 2D barcode was created and tested according to the test-environment and the test-procedure described in Section 8.1 and 8.2, respectively. In Figure 8.9, the total number of correct decodings is plotted over the average CND.

The results in Figure 8.9 are corroborative of the assumption that the decoding performance decreases with increasing average CND when considering average CND-values greater than 7.7. The only exception is given with  $d_x^{max} = 4$  where the decoding performance is slightly worse than in the case of  $d_x^{max} = 6$ . This effect is probably caused by the maximum CND that in the case of  $d_x^{max} = 4$  is 10 and thus

$d_x^{max}$	$\lambda(x)$
15	$0.311626x^2 + 0.362298x^3 + 0.083591x^4 + 0.093844x^5 + 0.033642x^6 + 0.023253x^7$ $+ 0.013902x^8 + 0.025308x^9 + 0.008483x^{10} + 0.010685x^{11} + 0.008096x^{12} + 0.00594x^{13}$ $+ 0.016579x^{14} + 0.002753x^{15}$
10	$0.363441x^2 + 0.310171x^3 + 0.101764x^4 + 0.073243x^5 + 0.045284x^6 + 0.043384x^7$ $+ 0.039883x^8 + 0.016461x^9 + 0.006368x^{10}$
6	$0.438821x^2 + 0.187372x^3 + 0.021882x^4 + 0.339611x^5 + 0.012315x^6$
4	$0.246838x^2 + 0.229353x^3 + 0.523809x^4$
3	$0.019983x^2 + 0.980017x^3$

**Table 8.6:** SNDDs of irregular LDPC codes designed for 2D barcodes by means of the method described in Section 7.4

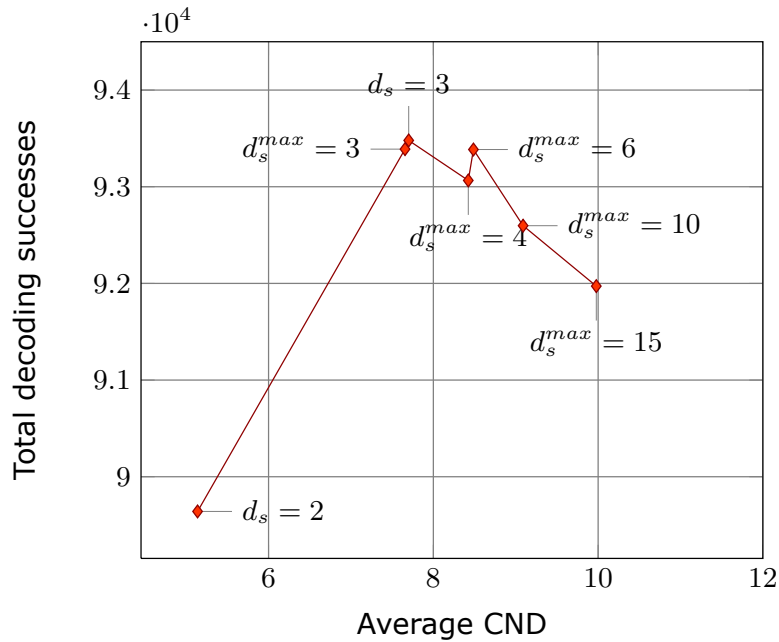
CND	Irregular $d_s^{max} = 15$	Irregular $d_s^{max} = 10$	Irregular $d_s^{max} = 6$	Irregular $d_s^{max} = 4$	Regular $d_s = 3$	Irregular $d_s^{max} = 3$	Regular $d_s = 2$
Min	9	8	8	8	7	7	5
Max	11	10	9	10	8	8	6
Average	9.978	9.089	8.487	8.4241	7.701	7.6563	5.1384

**Table 8.7:** Min, max and means of the number of connected symbol-nodes to the check-nodes (CND) considering different LDPC codes.

greater than 9 for  $d_x^{max} = 6$ .

The worst decoding performance is provided by the regular LDPC code with ( $d_x = 2$ ) although having a very low average CND. This is because the symbol-node degree (SND) is too low, therefore the loss in decoding performance due to the LDPC code properties outweighs the benefit of the symbol-placement.

Following the results in Figure 8.9, no gain is obtained when using irregular LDPC codes instead of regular LDPC codes in the context of LDPC-based 2D barcodes as designed in this thesis.



**Figure 8.9:** Number of total decoding successes plotted over the average CNR for two regular LDPC codes ( $d_x = 3$  and  $d_x = 2$ ) and five irregular LDPC codes designed with several values for  $d_x^{max}$ .

## 8.6 LDPC-based 2D barcode versus Data Matrix code

The final comparison is done to check the error-correction capabilities of the LDPC-based 2D barcode developed in this thesis compared to the current standard in DPMI applications, namely the RS-based Data Matrix code (DMC). For the LDPC-based 2D barcode, the regular LDPC code with  $d_x = 3$  is chosen since this code showed the best decoding results so far. The comparison results can be seen in Table 8.8.

It can be seen that the LDPC-based 2D barcode beats the DMC in all cases except for water-drops on yellow plastic where the DMC is slightly better (1%). In total, the 2D barcode developed in this thesis succeeded in 87% of the decodings and thus shows a gain of 10% compared to the DMC that yields 77%. However, the following analysis reveals that an interpretation of the decoding results in Table 8.8 is insufficient for a fair comparison.

Figure 8.10 shows the total number of decodings for the several numbers of pre-decoding bit errors considering the decodings of the DMC and the LDPC-based 2D barcode, respectively.

The frequency distributions in Figure 8.10 reveal two main points:

1. The pre-decoding bit errors are not equally distributed in both cases.
2. The LDPC-based 2D barcode had to face many more situations with more than 170 pre-decoding bit errors than the DMC.

Due to these two facts, it is essential to additionally consider the distribution of the pre-decoding bit errors when comparing the two variants of 2D barcodes. Especially the first point can be better evaluated by looking at the cumulative histograms in

Material	Damage	Decodings	RS-based DMC		LDPC-based 2D barcode	
			OK	%	OK	%
Brass	Water	9460	8819	93	9298	98
Alu	Water	18568	12579	68	17734	96
	Oil	15614	10876	70	12081	77
	Total	34182	23455	69	29815	87
Plastic g	Water	17721	15192	86	16717	94
	Oil	15869	10351	65	12419	78
	Total	33590	25543	76	29136	87
Plastic y	Water	16008	15318	96	15169	95
	Oil	14421	9779	68	10061	70
	Total	30429	25097	82	25230	83
Total	Water	61757	51908	84	58918	95
	Oil	45904	31006	68	34561	75
	Total	107661	82914	77	93479	87

**Table 8.8:** Comparison results of a 2D barcode based on a regular LDPC code with the standard DMC that utilizes a RS code.

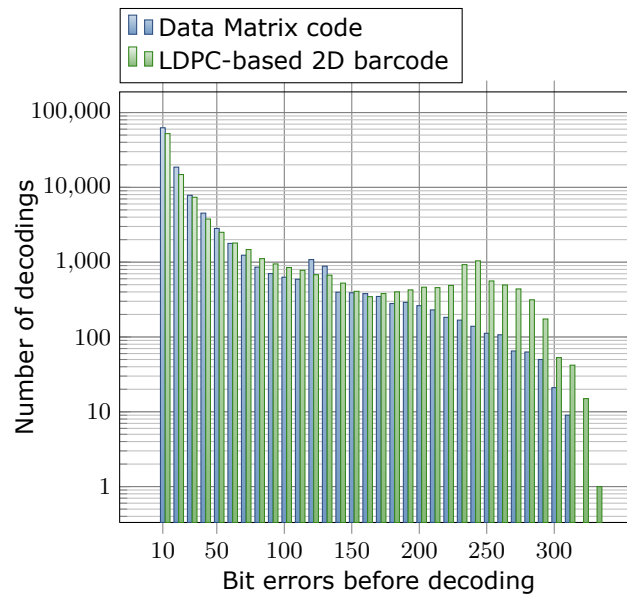
Figure 8.11. It can be seen that most of the decodings are affected with only a few pre-decoding bit errors. 75% and 69% of all decodings, for example, were affected by only up to 20 pre-decoding bit errors in the case of the DMC and the LDPC-based 2D barcode, respectively. Thus there is a strong weighting on situations with just a few pre-decoding bit errors. This weighting is eliminated by evaluating the error-correction capabilities of the 2D barcodes depending on the number of pre-decoding bit errors which is done in the following.

Figure 8.12 shows the successful decodings plotted over the pre-decoding bit errors. It can be seen that up to 10 pre-decoding bit errors, the DMC as well as the LDPC-based 2D barcode both succeeded in 100% of the decodings. In cases with more than 10 pre-decoding bit errors, the error-correction capabilities of the LDPC-based 2D barcode are clearly higher compared to the error-correction capabilities of the DMC.

The advantage of using the new 2D barcode variant developed in this thesis instead of the standard DMC can be well seen in Figure 8.13 where the obtained gain is shown for the several numbers of pre-decoding bit errors. The gain when using the LDPC-based 2D barcode compared to the RS-based DMC increases up to 71.9% for 50 pre-decoding bit errors and then decreases to 0.3% for 170 pre-decoding bit errors. In average a gain of 30.8% is obtained.

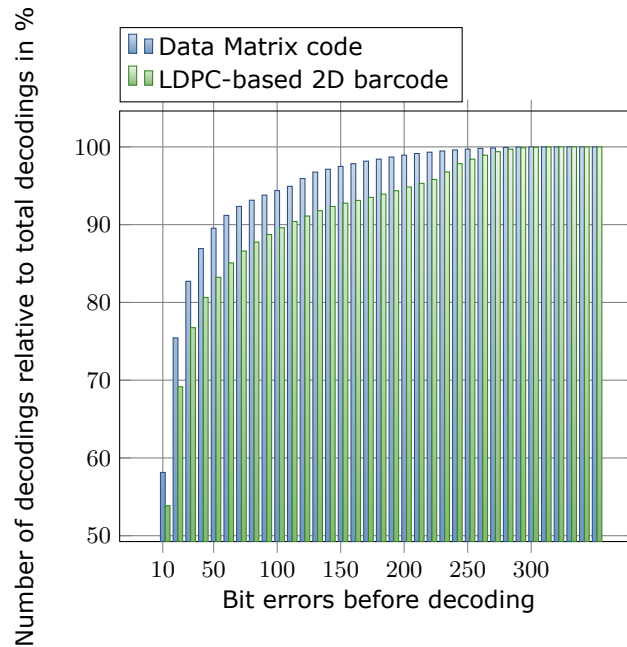
The results in Figure 8.14 provide a different perspective to the decoding performance comparison by showing the failed decodings. It can be seen that the DMC is maxed out on 60 pre-decoding bit errors where the LDPC-based 2D barcode only fails in 34% of the decodings. The LDPC-based 2D barcode is only maxed out for more than 170 pre-decoding bit errors.

When looking at the decoding speed, the RS-based DMC does not provide any advantages. This is shown in Figure 8.15. The average time required for the RS decoding was 3.62 seconds where the ED2D algorithm used in the case of the LDPC-based 2D barcode required 2.46 seconds in average.



**Figure 8.10:** Total numbers of decodings plotted over pre-decoding bit errors for a 2D barcode based on a regular LDPC code and the DMC, respectively.

As already mentioned in Section 8.4, all computations have been made using Matlab. For the RS encoding and decoding the functions *rsenc()* and *rsdec()* provided by Matlab were used, respectively. Due to that, it was not possible to further investigate the reason for the relatively high decoding times. Considering the decoding of LDPC-based 2D barcodes by means of the ED2D decoder developed in Section 7.3, it should be mentioned that the focus was not on optimizing the decoding speed. Furthermore, Matlab is pretty slow. This has been experienced when comparing the results obtained by means of Matlab with the decoding speed during the design of irregular LDPC codes. Contrary to the test-environment, the design described in Chapter 5 and Section 7.4 was written in C. Thus it was possible to compare the decoding speed considering Matlab-code and C-code. Table 8.9 shows the decoding speed of two different decoders for Matlab and C, respectively. One decoder is the MSc decoder of Section 4.4.3 and the other one the EDEP decoder developed in Section 6.4. The EDEP decoder thereby includes the MS decoder on the LDPC subgraph. It can be seen in Table 8.9 that the decoders realized in C-code are much faster. For the MSc algorithm, the C-code is about 2.74 times faster than the Matlab code whereas the C-code is 23.36 times faster than the Matlab code when considering the EDEP decoder. The gain obtained by use of C-code instead of Matlab is heavily dependent on various parameters like the utilized Matlab-functions, the number of computations and the required amount of memory. This comparison is only mentioned in order to emphasize the relative decoding speeds of the several decoders to each other without putting too much value on the absolute values.



**Figure 8.11:** Cumulative histograms for the number of decodings plotted over pre-decoding bit errors considering a 2D barcode based on a regular LDPC code and the DMC, respectively.

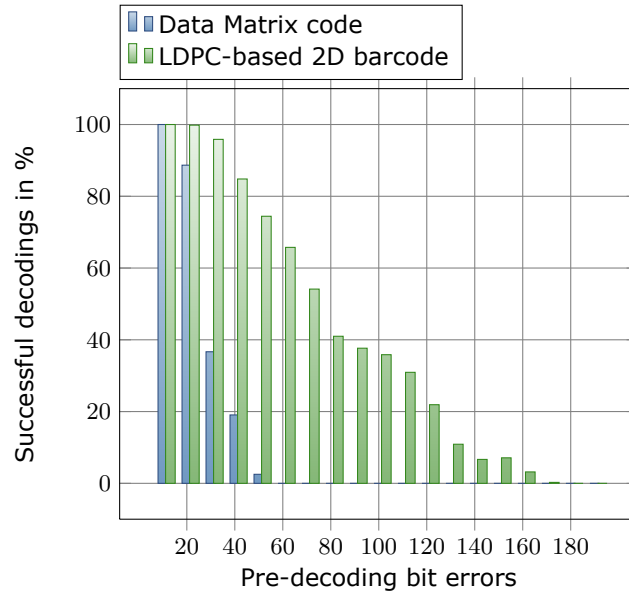
	MSc decoder	EDEP (MS) decoder
Matlab	6.6 ms	32.7 ms
C	2.41 ms	1.4 ms
Matlab/C	2.74	23.36

**Table 8.9:** Decoding speed of two decoders in Matlab and in C.

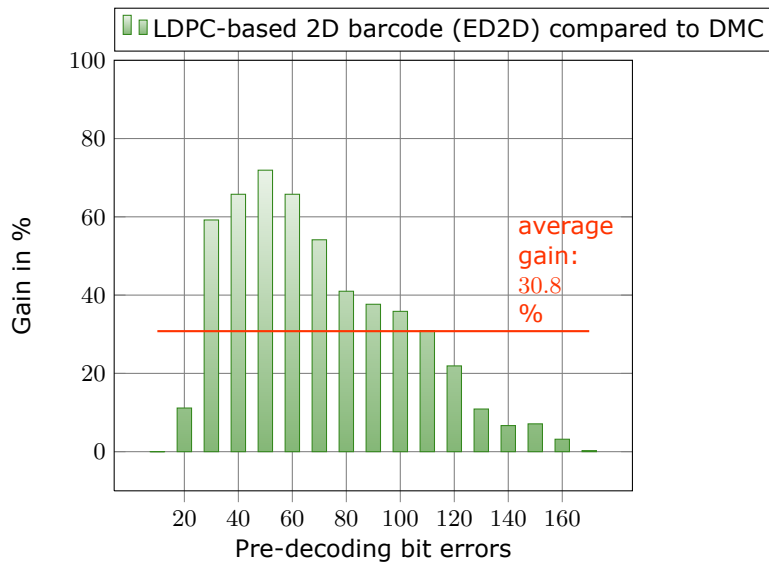
Considering the decoding speed, a comparison of the DMC with the LDPC-based 2D barcode decoded with the MSc decoder is interesting as well and is given in the following. So compared to the analysis based on the ED2D algorithm, the LDPC-based 2D barcodes are now decoded without any state-estimation involved. Even though, the LDPC-based 2D barcode shows a much better error-correction capability than the DMC (see Figure 8.16).

The resulting gain when using the LDPC-based 2D barcode together with the MSc decoder compared to the DMC can be seen in Figure 8.17.

There is still an average gain of 15.5 % when considering decodings up to 170 pre-decoding bit errors as done before. The big advantage when utilizing the MSc decoder instead of the ED2D decoder is the gain in computation time. This can be seen in Figure 8.18. On average, a decoding of a DMC requires 3.621 seconds whereas the MSc decoding of a LDPC-based 2D barcode only takes 86 milliseconds.

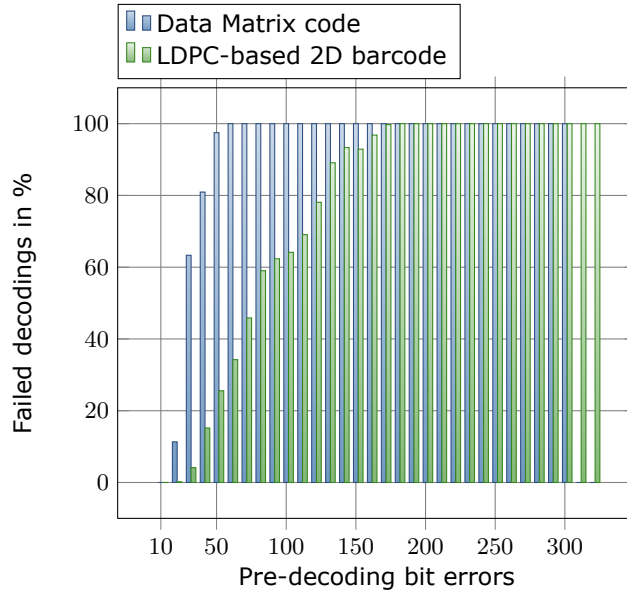


**Figure 8.12:** Successful decodings plotted over pre-decoding bit errors for a 2D barcode based on a regular LDPC code and the DMC, respectively.

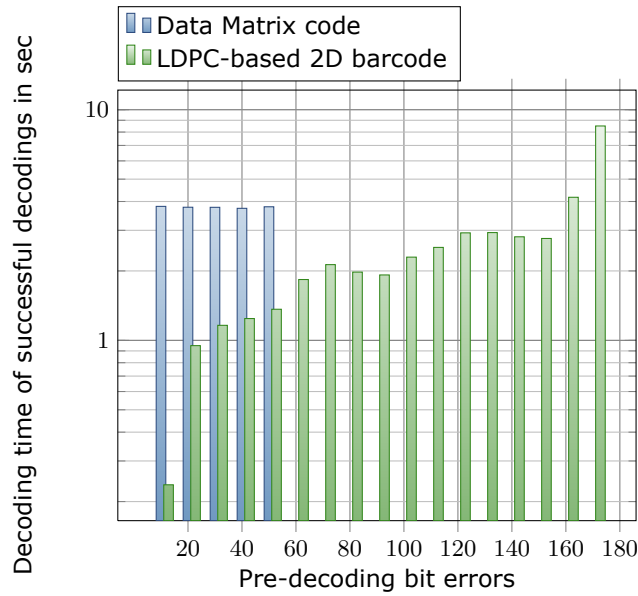


**Figure 8.13:** Gain obtained when using the LDPC-based 2D barcode instead of the standard DMC.

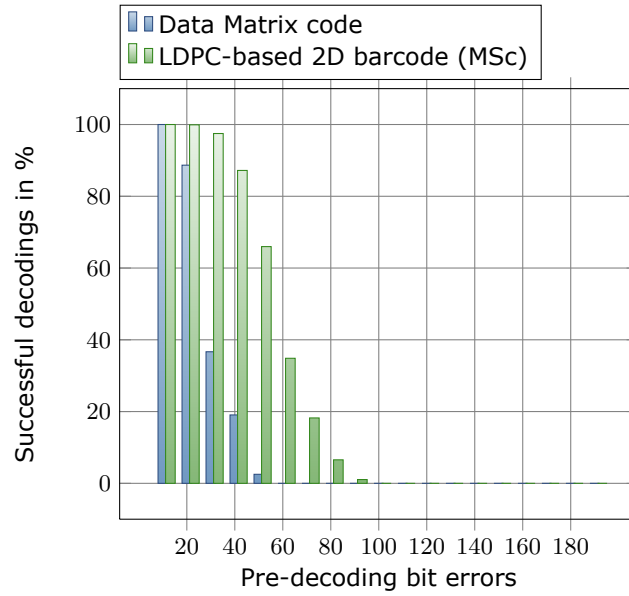




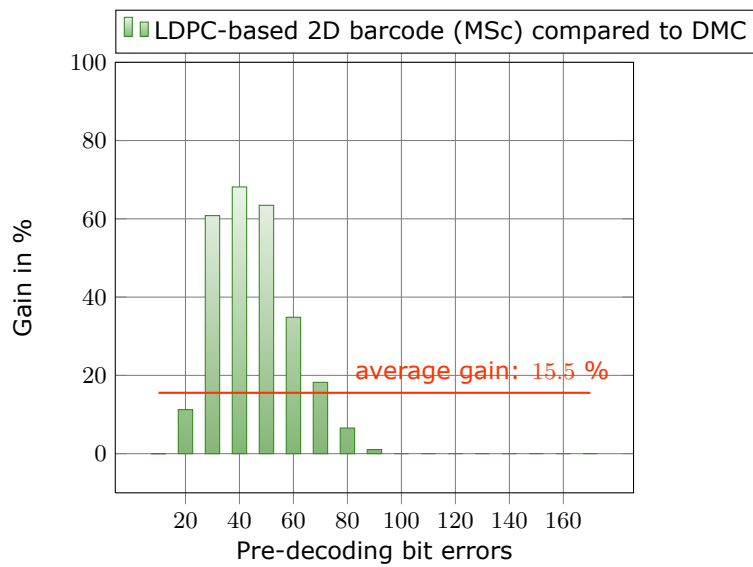
**Figure 8.14:** Failed decodings plotted over pre-decoding bit errors for a 2D barcode based on a regular LDPC code and the DMC, respectively.



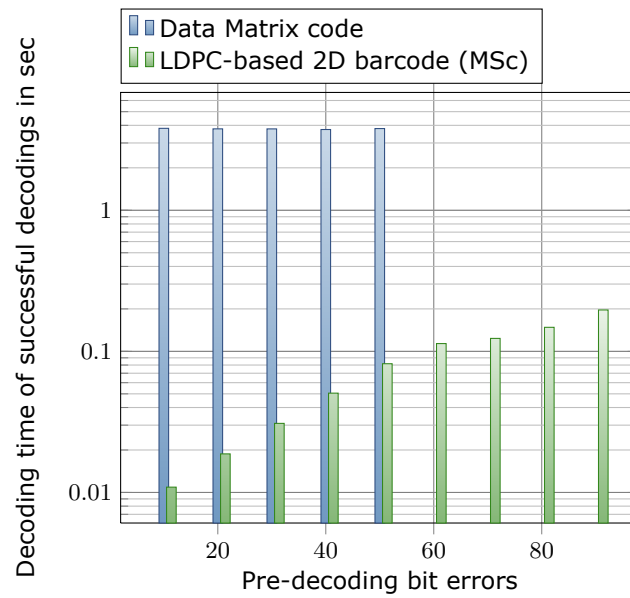
**Figure 8.15:** Decoding speed for a 2D barcode based on a regular LDPC code decoded with the ED2D decoder and the DMC. Only successful decodings are considered.



**Figure 8.16:** Successful decodings plotted over pre-decoding bit errors for a 2D barcode based on a regular LDPC code decoded with the MSc decoder and the DMC, respectively.



**Figure 8.17:** Gain obtained when using the LDPC-based 2D barcode together with the MSc decoder instead of the standard DMC.



**Figure 8.18:** Decoding times for a 2D barcode based on a regular LDPC code decoded with the MSc decoder and the DMC considering successful decodings.



## Chapter 9

# Conclusion and outlook

In this thesis, a new class of 2D barcodes that is based on low-density parity-check (LDPC) codes has been developed for the purpose of reliable object identification in industrial environments. Original contributions have thereby been presented regarding LDPC codes as such as well as their application to 2D barcodes. Particular attention has been paid to the design of the code and its decoding in both cases. In the following, the main results of this thesis are reviewed and proposals for future work are given.

### 9.1 Summary of contributions

#### LDPC codes

- In Chapter 5, a new design method for short irregular LDPC codes has been developed that is based on the downhill simplex (DHS) optimization, and is suited for arbitrary channel-models.
- It has been proven that the proposed design technique yields superior decoding performance for the additive white Gaussian noise (AWGN) channel and for the Markov-modulated Gaussian channel (MMGC) compared to well-tried optimization procedures.
  - ▷ A short irregular LDPC code designed for an AWGN channel with the introduced design method beats the irregular LDPC code designed in [37] based on a simplified version of the DHS algorithm (see Section 5.2).
  - ▷ A short irregular LDPC code designed for a MMGC with the proposed optimization method beats the irregular LDPC code designed in [57] by means of density-evolution (see Section 6.5.4).
  - ▶ One should choose the design-method developed in this thesis when considering short irregular LDPC codes.
- A new variant of estimation-decoding has been developed in Section 6.4 that integrates a reestimation of the transition probability matrix  $P$ .
  - ▷ The effectiveness of estimation-decoding has been proven for regular and irregular LDPC codes in the case of known transition probabilities (see Section 6.5.1).

- ▷ The advantage when using the new estimation-decoding variant in the case of unknown transition probabilities is proved in Section 6.5.2 and 6.5.4.
- ▶ One should use the proposed enhanced estimation-decoding algorithm with integrated reestimation of  $P$  if the transition probabilities are not known a priori.

## 2D barcodes

- In Chapter 7, a class of 2D barcodes based on LDPC codes has been developed.
- In Section 7.1, an intelligent interleaver has been designed to place the LDPC code's symbols in the data region of the 2D barcode.
- For the interleaving, a cost-function was developed based on the geometrical distance of the symbols in the data region and a distance measure evaluating the symbols relation in the LDPC code's Tanner graph (see Section 7.1.1).
  - ▷ The evaluation in Section 8.3 proved an increase in error-correction capability of 10.7 % when using the optimized interleaving compared to an interleaving with high costs.
- In Section 7.2, a channel-model has been constructed for 2D barcodes. It represents everything in between the embossing and the decoding of a 2D barcode considering acquisitions in a bright and a dark field, respectively.
- The Markov-LDPC factor graph of the estimation-decoding algorithm has been extended for the usage based on a 2D hidden Markov model (HMM) in Section 7.3.
- An algorithm called estimation-decoding in 2 dimensions (ED2D) algorithm has been developed to operate on the extended Markov-LDPC factor graph. The ED2D algorithm is based on estimation-decoding and the reestimation of the transition probabilities and is explained in Section 7.3.
  - ▷ In Section 8.4, the effectiveness of the ED2D algorithm that includes a state-estimation based on the 2D HMM is proven. Compared to the decoding without the state-estimation, an average gain of 15.2 % is obtained.
- In Section 7.4, irregular LDPC codes have been designed for the application with 2D barcodes.
  - ▷ The evaluation in Section 8.5 showed that in the context of 2D barcodes, no gain is obtained when using irregular LDPC codes instead of regular LDPC codes.
- A test-environment as well as a test-procedure have been developed in Section 8.1 and 8.2, respectively.
- The developed LDPC-based 2D barcode has been compared with the standard Reed-Solomon (RS)-based Data Matrix code (DMC) by means of the test-environment (see Section 8.6).

- ▷ The comparison in Section 8.6 proved the superiority of the 2D barcode that has been designed in this thesis by means of LDPC codes. In the simulated industrial environments, a gain in decoding performance of 30.8 % is obtained if applying the LDPC-based 2D barcode instead of the DMC.
- ▶ If a robust 2D barcode is required in the context of direct part mark identification (DPMI) applications, one is advised to choose the LDPC-based 2D barcode developed in this thesis.

## 9.2 Proposals for future work

There are still a bunch of open questions that may be interesting to research on in the future.

- Instead of minimizing the word error ratio (WER) when designing irregular LDPC codes for 2D barcodes, one could include the test-environment in order to maximize the decoding gain compared to the 2D barcode that utilizes the regular LDPC code.
- All testings with 2D barcodes within this thesis have been made with a fixed size of  $26 \times 26$  modules. A next step is therefore to extend the work to the application of other sizes as well.
- One interesting question to which an answer has to be found is if the utilization of nonbinary LDPC codes would yield an even better decoding performance.
- Considering the decoding speed, it would be favorable to find a speed-optimized version of the ED2D algorithm.
- An idea would be to combine the great decoding speed of the MSc algorithm with the superior error-correction capabilities of the ED2D algorithm. One could apply the MSc algorithm first and in case of a failed decoding utilize the ED2D algorithm in a second step for example.
- The decoding performance could be increased by means of a teach-in process. During such a first step, the channel-model could be adapted to a specific environment.
- One more subject to research on is the 2D barcode's finder pattern. This is more related to image-processing but it may also be possible to use a low-rate LDPC code to create a finder-pattern that offers both, the possibility of finding the 2D barcode by means of the LDPC code's properties as well as storing some information in the finder pattern.





## **Appendix A**

### **Symbol-placement sets**

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	356	3	7	236	488	253	242	490	502	561	40	231	461	261	345	227	365	569	475	483	457	342	263	10
2	9	63	391	348	289	230	39	193	387	477	274	375	421	426	48	266	219	282	453	501	361	467	576	566
3	6	351	393	370	94	33	45	124	95	450	556	302	510	110	550	308	507	548	56	336	68	75	28	568
4	565	563	38	279	155	323	76	470	298	131	103	269	527	505	414	448	439	353	125	512	188	109	459	19
5	29	293	183	67	165	123	170	314	166	32	189	220	478	416	503	159	153	120	259	93	200	194	223	4
6	233	60	256	275	417	64	333	173	472	380	201	428	529	405	35	113	57	22	553	468	158	97	119	574
7	386	337	404	197	518	389	154	245	546	519	398	262	395	169	156	427	415	92	319	78	206	306	471	474
8	59	182	133	46	285	447	486	297	554	100	316	79	136	410	152	273	339	304	286	96	381	315	438	20
9	354	172	338	369	127	102	394	198	137	55	300	258	280	218	257	443	168	114	536	53	516	520	84	116
10	455	523	91	186	424	267	432	181	400	126	494	167	144	537	396	495	401	36	23	449	74	367	525	454
11	462	277	303	271	321	134	248	517	111	66	376	24	532	54	543	31	284	250	544	157	213	322	358	290
12	452	252	299	150	203	138	539	174	52	21	433	122	212	412	25	524	294	47	379	162	492	442	283	234
13	346	359	171	390	132	552	460	80	61	430	101	320	549	373	557	423	408	204	65	292	130	420	129	464
14	317	207	366	98	476	278	362	164	489	121	112	281	1	288	493	368	264	140	117	418	246	388	513	343
15	352	484	392	434	143	49	210	528	480	444	215	372	411	504	364	270	521	332	224	254	481	139	485	27
16	350	496	99	313	441	526	265	51	58	179	402	384	148	163	190	178	185	145	435	403	205	331	329	239
17	465	202	325	86	409	422	440	555	81	541	50	107	247	326	413	249	89	312	429	199	108	268	161	77
18	232	192	542	90	469	378	287	34	221	558	538	335	73	545	255	499	431	530	419	26	82	69	560	228
19	226	301	237	196	176	479	128	446	43	547	511	42	291	296	506	498	531	70	383	135	540	41	318	340
20	344	357	44	151	328	211	355	509	106	180	487	177	71	118	451	533	225	334	222	184	105	508	522	104
21	14	575	187	514	85	551	208	436	72	209	160	141	377	216	327	330	360	214	87	149	399	497	349	16
22	8	363	463	425	295	535	191	147	175	307	217	473	276	244	482	534	397	260	146	445	374	240	251	243
23	571	347	572	238	341	456	83	324	195	142	305	310	88	62	382	406	437	407	235	562	564	371	385	570
24	13	567	229	2	17	11	241	37	491	515	311	500	559	466	12	272	115	30	309	458	18	15	573	5

**Table A.1:** Optimized symbol-placement for a 2D barcode based on a regular LDPC code with  $d_x = 2$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 21077.39$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	375	121	310	576	389	103	85	398	539	401	261	348	327	334	175	244	110	154	390	122	250	98	248	315
2	113	255	339	573	230	222	264	354	262	241	383	534	109	91	185	62	253	193	363	132	84	247	124	288
3	55	548	345	37	279	108	80	459	488	367	284	27	482	236	176	204	54	225	365	455	494	560	270	303
4	101	399	489	174	366	336	526	519	31	147	475	520	274	194	478	392	267	210	75	565	541	537	56	309
5	135	159	125	569	532	105	444	422	191	202	214	436	423	527	362	465	464	568	219	272	421	285	397	269
6	249	341	251	430	352	6	213	129	333	145	496	34	440	30	152	293	167	381	59	217	227	97	323	307
7	171	343	212	544	67	550	547	556	280	370	533	495	400	209	524	57	76	7	190	198	351	153	374	130
8	324	235	382	517	298	104	160	10	529	483	404	512	11	416	499	361	273	563	454	502	546	15	140	331
9	481	525	68	522	501	437	24	470	360	258	87	447	287	438	90	70	458	516	572	197	223	542	476	332
10	16	112	163	509	146	238	545	106	23	513	228	39	166	246	64	445	278	461	407	549	26	406	187	239
11	321	96	451	123	432	508	371	433	552	20	2	329	9	164	373	93	141	205	523	521	528	538	355	88
12	396	337	377	178	486	211	570	473	201	45	99	484	1	417	376	471	115	511	554	492	102	500	51	276
13	405	134	155	462	86	18	207	4	562	304	505	408	368	442	420	468	5	35	231	12	441	506	391	266
14	388	44	425	378	575	487	289	162	17	77	218	292	429	116	514	503	137	73	558	439	183	94	60	338
15	418	265	180	457	419	25	286	469	364	467	3	192	435	32	358	156	460	291	13	196	412	431	195	157
16	313	308	199	48	409	220	19	297	477	491	240	415	14	340	395	448	449	427	543	81	243	480	47	387
17	63	254	107	226	498	168	150	531	170	530	424	8	173	40	188	283	92	551	300	490	165	296	379	356
18	119	260	177	237	493	52	208	182	450	574	434	385	443	114	466	497	518	456	128	78	139	413	305	393
19	143	74	357	446	50	100	33	61	571	411	479	221	290	428	559	275	144	151	299	43	72	186	312	317
20	318	245	234	349	474	158	302	200	453	111	83	181	535	215	510	29	36	553	504	372	359	557	206	271
21	53	347	277	344	216	179	515	268	540	281	184	294	282	350	95	38	172	224	233	148	127	41	136	259
22	311	242	314	46	142	58	386	342	507	567	414	22	229	79	161	203	138	561	306	403	410	380	189	133
23	320	71	353	536	89	126	452	69	28	485	118	402	325	42	335	66	566	330	463	263	322	49	346	256
24	328	384	326	426	564	21	120	394	232	472	82	65	169	131	149	301	295	257	319	369	555	252	316	117

**Table A.2:** Optimized symbol-placement for a 2D barcode based on a regular LDPC code with  $d_x = 3$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 20551.46$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	523	443	449	297	517	372	225	446	224	300	519	448	3	6	5	4	526	450	10	386	451	306	11	1
2	233	215	514	445	201	370	376	518	373	238	520	236	521	7	525	22	308	27	246	12	453	19	529	2
3	515	516	216	197	293	442	202	211	191	210	522	221	235	8	9	14	454	37	455	16	13	28	25	387
4	231	371	196	188	439	218	207	299	237	377	378	203	240	527	381	456	462	18	20	385	41	24	30	459
5	441	513	193	368	189	294	217	214	447	208	302	296	524	324	384	23	31	532	383	17	29	382	460	530
6	286	190	192	200	198	194	292	199	295	220	379	243	528	244	457	32	21	389	464	461	458	303	39	531
7	182	440	512	438	511	205	204	241	219	206	230	304	380	388	301	15	36	535	26	33	537	38	58	318
8	290	289	508	223	226	369	298	209	234	195	444	239	245	391	42	533	55	40	538	34	321	49	305	46
9	291	374	362	177	176	366	288	213	232	336	315	375	312	319	463	534	467	59	394	35	43	540	542	392
10	435	183	165	506	510	222	174	363	229	367	242	393	452	307	543	390	63	323	56	64	536	48	466	395
11	184	166	507	172	179	509	133	169	163	187	124	313	247	322	402	60	47	68	53	539	465	325	326	50
12	181	185	574	434	283	274	212	171	175	140	342	328	403	344	401	311	337	250	54	468	545	51	52	44
13	437	285	576	168	164	284	178	154	279	345	272	334	117	327	316	398	314	80	400	67	66	399	45	62
14	173	360	502	167	149	186	431	573	259	418	421	320	248	396	310	551	254	338	70	257	110	65	471	544
15	501	436	432	282	504	150	159	130	494	425	266	350	409	262	332	476	78	71	405	82	474	546	470	330
16	228	365	575	498	144	426	343	571	570	422	121	331	125	99	98	309	552	335	69	553	256	252	541	61
17	170	359	180	566	160	569	572	354	277	148	414	485	120	412	484	550	87	548	249	473	83	407	258	76
18	227	280	500	361	158	134	349	564	138	267	139	118	111	100	265	481	95	88	79	81	72	260	333	74
19	161	430	157	427	145	143	567	137	129	356	123	119	486	346	263	103	109	556	96	77	339	329	397	57
20	505	364	146	281	152	495	352	270	493	116	264	351	560	563	255	105	104	94	340	406	73	404	75	472
21	503	497	358	141	142	568	153	278	136	112	490	128	127	559	102	555	557	558	477	253	475	408	549	469
22	429	151	357	162	147	491	126	424	135	268	115	488	108	113	417	101	413	93	90	92	86	480	479	341
23	433	271	156	428	131	423	276	565	419	420	353	122	489	483	416	91	478	97	89	415	482	261	84	85
24	499	287	496	155	355	492	132	273	348	562	275	561	114	347	269	251	107	411	554	487	106	547	410	317

**Table A.3:** Degraded symbol-placement for a 2D barcode based on a regular LDPC code with  $d_x = 3$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 24761.20$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	569	563	534	533	526	528	522	487	459	425	392	440	462	480	502	506	503	527	518	514	513	523	575	576
2	574	547	532	525	515	461	380	416	419	351	358	370	315	426	424	336	394	471	504	500	517	516	536	573
3	537	544	529	507	472	245	199	280	241	331	356	261	357	268	385	219	198	396	410	489	482	501	520	551
4	542	538	493	477	401	253	243	256	306	252	381	284	372	263	196	192	188	251	387	242	390	509	510	531
5	530	481	414	190	301	285	345	207	388	53	189	141	220	221	307	275	382	386	341	254	324	490	498	519
6	535	444	389	423	222	203	335	290	126	272	286	19	54	86	76	88	359	239	278	348	244	237	497	505
7	524	235	259	367	333	255	177	89	173	46	105	124	80	176	152	135	9	71	214	257	201	308	433	492
8	451	434	205	332	208	185	94	77	123	15	79	153	45	90	83	167	352	106	279	314	346	228	250	495
9	474	312	384	303	383	159	20	168	139	57	18	33	179	99	4	85	101	58	11	194	204	375	230	439
10	407	411	334	232	193	127	6	125	36	132	13	8	170	7	163	149	160	2	164	283	267	316	234	485
11	406	355	258	295	281	131	129	38	103	158	109	148	49	72	113	130	51	93	73	155	229	291	299	226
12	448	293	183	289	215	32	172	143	165	178	119	1	12	55	87	111	10	174	96	81	213	209	195	417
13	467	402	309	231	368	48	91	27	122	30	50	34	104	29	14	59	65	202	154	67	264	296	371	445
14	405	409	302	340	217	41	128	114	162	37	16	60	166	108	26	151	134	40	3	84	342	320	395	437
15	412	236	343	377	327	133	95	171	17	56	121	22	175	112	97	117	142	82	115	187	326	224	374	483
16	470	246	339	304	200	186	47	102	146	107	157	140	43	169	31	120	25	78	92	265	364	353	422	469
17	521	435	233	191	294	274	63	138	39	118	110	35	24	61	23	137	145	70	277	379	328	362	393	473
18	539	421	273	378	238	262	180	44	144	5	136	116	68	147	161	74	69	218	216	225	181	247	457	450
19	550	476	400	305	311	210	300	211	66	75	100	42	21	52	150	98	156	287	297	376	288	313	466	508
20	548	540	430	432	347	260	373	330	62	64	28	366	338	282	350	361	266	318	337	182	292	404	484	541
21	553	549	397	447	398	206	325	184	271	269	197	360	240	249	310	323	319	369	212	321	458	464	512	570
22	554	555	545	441	454	413	431	363	428	317	270	329	365	344	354	276	322	248	399	418	455	460	543	571
23	560	556	557	491	449	486	429	427	391	408	227	436	349	403	298	223	453	415	479	478	456	559	572	568
24	564	561	558	552	546	496	438	488	475	468	452	420	463	442	446	465	511	443	494	499	562	567	565	566

**Table A.4:** Optimized symbol-placement for a 2D barcode based on an irregular LDPC code. The irregular LDPC code was designed based on the 2-state channel-model in Figure 7.16. The maximum degree was thereby set to  $d_x^{max} = 15$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 14343.14$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	567	570	520	557	538	493	519	454	449	474	457	407	438	436	451	467	452	466	548	552	555	553	565	576
2	523	524	502	505	510	470	244	243	248	319	268	276	333	421	425	423	429	453	478	465	559	561	547	564
3	522	529	535	460	469	441	265	321	250	223	213	314	346	293	267	344	315	328	394	471	542	540	562	558
4	517	528	512	414	285	252	226	302	318	379	367	298	212	264	296	380	352	299	312	396	397	482	544	556
5	530	527	475	432	374	288	317	387	84	154	89	149	93	187	39	365	384	225	291	353	437	389	491	554
6	537	513	476	409	224	337	112	182	14	53	86	8	157	58	117	97	128	356	217	303	228	433	506	541
7	525	455	428	334	338	215	165	82	160	124	180	30	4	163	95	104	3	23	118	287	355	399	481	518
8	498	426	375	233	341	368	191	114	9	44	48	31	72	122	42	186	107	125	21	339	322	419	413	483
9	473	440	237	332	111	37	56	131	136	92	188	60	195	204	2	109	98	173	99	277	294	336	238	496
10	412	240	388	271	184	15	178	45	206	54	74	120	19	106	55	152	5	179	130	66	260	357	430	459
11	443	427	289	261	85	49	143	65	18	100	181	126	197	113	103	172	12	43	26	79	350	290	227	463
12	410	232	329	310	183	153	77	162	209	134	96	139	17	202	133	71	80	67	155	198	378	251	349	435
13	400	404	279	345	161	13	62	193	35	6	68	78	1	94	36	150	91	7	119	20	326	323	434	398
14	390	385	219	210	24	177	148	175	142	57	164	41	76	83	199	208	135	158	116	203	331	364	330	444
15	422	377	221	275	194	11	28	73	59	170	46	207	16	123	25	108	75	168	144	366	327	256	266	446
16	418	447	235	325	348	38	196	129	32	200	140	64	81	167	169	190	141	101	50	360	358	230	262	439
17	472	402	245	241	263	371	166	29	69	87	174	159	185	61	105	88	40	127	102	335	362	376	431	503
18	479	416	246	295	284	234	151	63	90	115	205	189	52	10	27	22	47	110	156	306	257	342	486	468
19	536	403	401	242	370	281	171	138	51	176	34	121	201	145	33	70	137	297	311	247	270	363	456	504
20	516	458	406	316	372	320	258	383	214	272	313	132	146	192	147	359	307	324	249	253	373	461	501	514
21	546	531	488	462	304	292	220	343	216	381	300	351	308	369	347	386	273	282	255	382	484	511	490	560
22	568	550	526	508	392	424	236	211	218	283	229	301	278	231	361	340	239	269	420	489	495	515	521	572
23	551	532	533	539	477	464	448	274	309	354	280	259	305	286	222	254	445	411	480	500	509	492	569	566
24	575	571	549	543	507	534	499	391	450	405	408	417	395	415	442	393	485	487	497	494	563	545	574	573

**Table A.5:** Optimized symbol-placement for a 2D barcode based on an irregular LDPC code. The irregular LDPC code was designed based on the 2-state channel-model in Figure 7.16. The maximum degree was thereby set to  $d_x^{max} = 10$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 11926.69$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	574	572	570	522	531	413	567	548	489	465	564	488	421	447	376	550	378	380	474	390	425	436	403	576
2	573	470	515	535	408	566	521	404	568	369	257	311	338	328	271	501	445	532	464	481	394	476	437	428
3	575	504	397	519	402	545	371	291	318	259	298	335	286	279	341	284	288	362	538	414	411	478	537	422
4	479	562	552	484	459	336	258	293	273	192	178	14	30	245	56	303	285	316	539	463	494	540	412	440
5	388	506	407	370	355	305	302	6	167	11	41	231	229	19	15	189	70	294	283	296	554	524	502	426
6	374	526	420	330	310	340	57	237	172	116	197	121	148	139	105	31	37	235	40	317	365	503	528	472
7	446	401	499	254	327	87	98	3	135	168	186	111	71	48	163	130	16	43	65	306	334	527	418	434
8	466	415	265	301	349	227	239	47	155	83	182	99	159	198	243	89	185	125	133	210	309	289	492	439
9	541	560	313	333	179	209	169	147	177	61	36	5	76	120	181	151	80	129	106	247	260	297	452	419
10	458	410	347	292	63	22	34	101	23	24	154	202	66	145	158	204	222	104	244	238	124	324	546	525
11	529	277	272	39	234	187	7	122	191	84	38	184	86	2	102	176	26	82	10	173	190	281	416	482
12	454	339	326	152	35	183	94	53	219	58	77	112	194	251	72	213	166	60	44	196	55	351	361	423
13	513	353	350	221	25	97	127	134	149	126	4	142	1	150	242	45	236	136	90	223	73	343	348	544
14	497	262	280	62	85	188	42	171	205	141	212	157	137	29	81	107	214	132	64	17	27	360	363	480
15	510	496	253	358	246	51	140	216	52	180	33	175	203	160	218	199	211	143	146	78	74	290	551	523
16	509	438	287	354	195	117	206	193	88	225	110	100	144	103	113	161	75	156	79	20	307	345	557	514
17	565	453	373	263	8	226	220	91	95	164	50	240	241	162	228	109	46	93	207	68	314	331	553	561
18	430	443	455	282	337	248	13	224	201	128	170	208	230	96	174	131	250	252	233	359	261	543	449	563
19	500	435	495	505	268	300	21	114	28	92	200	232	165	217	49	32	115	18	270	267	367	533	508	433
20	569	512	516	405	332	325	295	54	249	118	138	12	69	153	119	108	67	264	278	255	556	536	396	507
21	375	442	468	460	448	274	356	352	304	269	59	215	123	357	9	344	322	299	417	542	462	534	400	379
22	385	382	450	456	467	457	368	275	319	342	320	312	329	266	308	346	323	364	469	547	555	520	471	429
23	387	477	475	386	398	485	487	424	511	399	366	321	276	315	256	372	377	558	461	441	498	384	432	571
24	383	491	431	493	393	451	427	559	517	389	391	549	530	406	409	490	395	483	473	444	392	381	518	486

**Table A.6:** Optimized symbol-placement for a 2D barcode based on an irregular LDPC code. The irregular LDPC code was designed based on the 2-state channel-model in Figure 7.16. The maximum degree was thereby set to  $d_x^{max} = 6$ . The cost for this symbol-placement according to Equation (7.3) is  $C(2DC) = 24898.11$ .

Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	294	573	339	569	371	440	367	276	494	287	498	457	518	564	286	503	301	450	297	361	453	508	537	281
2	556	357	354	435	403	470	279	432	406	483	373	425	388	456	420	496	532	330	342	542	514	404	343	399
3	340	298	442	364	566	555	285	531	146	147	238	427	466	484	541	368	548	471	458	375	395	299	501	516
4	439	359	312	313	355	305	434	241	232	208	198	214	265	223	226	240	436	455	374	314	379	280	344	522
5	394	278	431	350	416	550	182	246	372	263	155	255	188	203	249	200	253	158	421	412	557	376	507	275
6	349	575	338	417	428	306	160	213	245	80	165	33	17	32	102	217	173	268	402	559	356	445	316	454
7	415	389	317	510	386	181	216	201	207	101	55	50	100	9	54	18	164	193	259	360	426	449	479	461
8	351	331	423	408	546	224	256	45	81	77	125	105	62	70	126	266	35	187	247	544	242	472	492	293
9	521	474	366	536	195	218	30	56	94	15	111	29	116	8	89	117	109	46	179	152	382	437	515	363
10	478	295	485	237	171	257	63	40	27	115	69	65	7	136	22	71	61	60	75	221	272	324	500	385
11	397	377	438	233	209	250	66	43	139	99	129	39	76	11	106	10	12	34	103	162	149	365	288	277
12	509	282	477	153	271	112	84	258	133	110	73	97	119	130	36	21	87	16	262	244	215	512	311	352
13	346	369	196	222	174	90	26	3	67	14	122	1	44	104	74	135	142	72	197	178	190	418	430	447
14	505	307	480	154	189	252	57	107	5	25	140	28	128	131	137	24	13	19	219	192	144	370	325	393
15	513	571	419	176	210	243	53	79	38	134	114	49	58	124	78	120	88	92	261	229	235	381	413	535
16	491	487	353	211	183	143	4	113	118	59	68	64	93	96	127	48	47	98	177	228	167	296	506	302
17	574	407	429	422	236	212	157	85	52	6	121	2	20	37	42	95	23	267	166	150	433	320	321	488
18	337	465	328	323	231	248	184	270	31	83	86	138	132	141	108	51	205	199	254	225	443	464	553	332
19	539	441	481	424	460	562	175	269	260	185	41	91	123	82	273	172	194	230	545	463	383	462	467	347
20	476	401	329	410	411	547	234	156	204	251	274	145	206	159	264	239	163	151	168	529	378	540	576	409
21	326	291	473	489	486	523	527	322	384	202	170	180	161	191	148	227	169	526	495	414	520	519	551	341
22	392	327	362	504	333	517	493	310	304	387	309	220	186	558	549	475	300	444	552	528	502	390	405	490
23	446	554	511	290	567	459	538	335	524	497	482	468	315	319	533	318	398	530	380	570	358	348	534	336
24	283	396	499	525	303	452	345	469	284	308	334	292	561	400	391	451	543	565	289	568	563	560	572	448

**Table A.7:** Optimized symbol-placement for a 2D barcode based on an irregular LDPC code. The irregular LDPC code was designed based on the 2-state channel-model in Figure 7.16. The maximum degree was thereby set to  $d_x^{max} = 4$ . The cost for this symbol-placement according to Equation (7.3) is  $\mathcal{C}(2DC) = 25006.36$ .



Row	Column																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	389	316	393	247	116	572	160	345	49	75	361	159	133	52	273	54	562	317	260	26	403	120	253	464
2	83	164	343	263	231	567	411	18	135	542	332	220	402	140	348	204	252	568	280	225	202	89	261	335
3	327	563	46	277	250	186	53	414	290	237	121	508	459	193	310	468	423	494	242	320	248	112	325	258
4	66	382	486	390	308	350	298	303	223	132	545	80	110	71	206	531	304	155	87	283	500	409	413	214
5	323	337	32	426	184	415	27	475	226	360	532	289	404	487	192	450	228	21	386	92	256	368	533	45
6	170	113	162	119	419	70	233	452	524	510	397	469	291	458	429	59	359	525	425	384	330	352	20	410
7	331	14	292	472	428	379	570	38	479	154	511	142	527	358	573	137	370	364	484	535	205	239	349	396
8	466	199	259	526	324	366	208	173	234	60	275	136	509	518	401	530	515	151	203	556	240	446	222	95
9	33	478	268	514	22	211	90	482	507	297	315	519	12	560	521	74	182	497	371	174	88	540	148	249
10	128	97	387	288	56	434	274	19	517	395	10	498	34	69	127	516	287	536	201	346	81	369	339	130
11	276	78	28	440	163	394	157	481	485	431	520	3	6	139	4	207	319	534	453	569	219	571	493	141
12	270	505	499	451	64	264	227	523	554	8	172	191	11	209	7	122	444	165	175	138	176	198	47	169
13	267	31	557	564	491	460	189	461	473	443	5	9	457	1	238	93	255	353	309	123	405	232	421	427
14	114	347	40	565	463	407	305	107	430	417	496	447	438	105	2	218	322	442	344	144	550	293	312	109
15	131	152	213	188	183	374	278	372	149	549	62	439	433	445	566	224	476	221	48	474	385	576	455	314
16	41	15	61	418	153	492	166	301	529	299	104	465	547	281	376	85	284	147	102	156	25	373	326	43
17	55	307	124	378	29	477	229	558	30	528	67	436	448	490	483	512	68	65	456	161	51	181	377	82
18	266	241	101	73	420	295	217	367	575	462	546	504	406	422	177	180	195	185	286	454	489	296	272	76
19	342	57	179	150	13	194	471	365	354	362	190	441	167	522	357	145	235	210	513	495	251	435	282	257
20	341	432	36	42	294	108	543	17	380	537	143	44	488	302	561	539	212	125	77	306	399	321	338	134
21	126	197	334	503	200	375	351	236	196	23	158	381	470	215	544	363	16	502	50	171	416	340	244	98
22	168	328	99	91	449	243	246	37	412	383	24	552	541	501	106	300	480	146	574	467	356	506	254	84
23	333	400	72	129	285	559	408	555	424	178	216	355	58	391	548	269	103	265	118	230	388	318	437	538
24	271	63	115	279	35	96	336	245	117	94	39	111	79	329	187	551	398	553	100	313	86	311	262	392

**Table A.8:** Optimized symbol-placement for a 2D barcode based on an irregular LDPC code. The irregular LDPC code was designed based on the 2-state channel-model in Figure 7.16. The maximum degree was thereby set to  $d_x^{max} = 3$ . The cost for this symbol-placement according to Equation (7.3) is  $C(2DC) = 20434.43$ .



## Bibliography

- [1] J. N. Woodland and B. Silver, "Classifying apparatus and method," Patent US 2,612,994, 20.10.1949.
- [2] ISO/IEC, "15420:2009: Information technology – automatic identification and data capture techniques – ean/upc bar code symbology specification," 2009.
- [3] ISO/IEC, "15426-2: Information technology – automatic identification and data capture techniques – bar code verifier conformance specification – part 2: Two-dimensional symbols," 15.03.2005.
- [4] ISO/IEC, "16022:2000(e): Information technology – international symbology specification – data matrix," 01.05.2000.
- [5] ISO/IEC, "18004:2006: Information technology – automatic identification and data capture techniques – qr code 2005 bar code symbology specification," 31.08.2006.
- [6] ISO/IEC, "15438: Information technology – automatic identification and data capture techniques – pdf417 bar code symbology specification," 24.05.2006.
- [7] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.
- [8] ISO/IEC, "24778: Information technology – automatic identification and data capture techniques – aztec code bar code symbology specification," 11.02.2008.
- [9] Automotive Industries Action Group (AIAG), "B-1: Barcode symbology standard," 1984.
- [10] Automotive Industries Action Group (AIAG), "B-17: 2d direct parts marking guideline," 2009.
- [11] Automotive Industries Action Group (AIAG), "B-4: Parts identification and tracking application standard," 2003.
- [12] Automotive Industries Action Group (AIAG), "B-13: 2d symbology white paper," 2000.
- [13] Automotive Industries Action Group (AIAG), "B-10: Guideline for use of two-dimensional trading partner labels implementation guideline," 2004.
- [14] ISO/IEC, "15415:2011(e): Information technology – automatic identification and data capture techniques – bar code symbol print quality test specification – two-dimensional symbols," 15.12.2011.

- [15] Department of Defense (DoD), "Mil-std-130: Identification marking of u.s. military property," 17.12.2007.
- [16] Air Transport Association (ATA), "Spec 2000, chapter 9: Automated identification and data capture (aidc)," 2009.
- [17] Electronic Industries Association (EIA), "Eia-802: Product marking."
- [18] Electronic Industries Association (EIA), "Eia-706: Component marking," 1997.
- [19] National Aeronautics and Space Administration (NASA), "Nasa-std-6002: Applying data matrix identification symbols on aerospace parts," 23.09.2002.
- [20] National Aeronautics and Space Administration (NASA), "Nasa-hdbk-6003: Application of data matrix identification symbols to aerospace parts using direct part marking methods/techniques," 23.09.2002.
- [21] International Aerospace Quality Group (IAQG), "As9132: Data matrix (2d) coding quality requirements for parts marking," 16.02.2005.
- [22] C. E. Shannon, "The bell system technical journal 27," *A Mathematical Theory of Communication*, pp. 379–423, 1948.
- [23] R. L. Stratonovich, "Conditional markov processes," *Theory of Probability and its Applications*, vol. 5, p. 156, 1960.
- [24] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.
- [25] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J*, vol. 42, no. 9, pp. 1977–1997, 1963.
- [26] R. G. Gallager, "Low density parity check codes," *IRE Trans on Information Theory*, vol. 1, pp. 21–28, 1962.
- [27] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *IEEE Proceedings of the International Conference on Communications*, Geneva, Switzerland, 1993, pp. 1064–1070.
- [28] D. MacKay and R. Neal, "Good codes based on very sparse matrices," *Cryptography and Coding*, pp. 100–111, 1995.
- [29] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, Linköping, 1996.
- [30] W. E. Ryan and S. Lin, *Channel Codes: Classical and Modern*. Cambridge Univ Pr, 2009.
- [31] G. A. Margulis, "Explicit constructions of graphs without short cycles and low density codes," *Combinatorica*, vol. 2, no. 1, pp. 71–78, 1982.
- [32] V. Zyablov and M. Pinsker, "Estimation of the error-correction complexity of gallager low-density codes," *Problemy Peredachi Informatsii*, no. vol. 11, pp. 22–26, 1975.

- [33] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, pp. 533–547, 1981.
- [34] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, 2001.
- [35] M. Luby, M. Mitzenmacher, A. Shokrollah, and D. Spielman, "Analysis of low density codes and improved designs using irregular graphs," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 249–258.
- [36] M. Anthony and M. Harvey, *Linear algebra: Concepts and methods*. Cambridge: Cambridge University Press, 2012.
- [37] X. Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, 2005.
- [38] J. Pearl, Ed., *Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach*, 1982.
- [39] J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of plausible inference*, 2nd ed., ser. TheMorgan Kaufmann series in representation and reasoning. San Mateo and Calif: Kaufmann, 1988.
- [40] Büchi Alfred, "Verbrennungskraftmaschinenanlage mit kompressor, einem kolbenmotor und einer dahinter geschalteten turbine," Patent DE204 630, 1905.
- [41] X. Y. Hu, E. Eleftheriou, D. M. Arnold, and A. Dholakia, Eds., *Efficient implementations of the sum-product algorithm for decoding LDPC codes*, vol. 2, 2002.
- [42] T. Tian, C. Jones, J. Villasenor, and R. Wesel, Eds., *Construction of irregular LDPC codes with low error floors*, vol. 5, 2003.
- [43] J. Thorpe, "Low-density parity-check (ldpc) codes constructed from protographs," *IPN progress report*, vol. 42, no. 154, pp. 42–154, 2003.
- [44] Y. Wang and M. Fossorier, Eds., *Doubly generalized LDPC codes*, 2006.
- [45] European Telecommunications Standards Institute, "Etsi en 302 307: Digital video broadcasting (dvb); second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications (dvb-s2)," 08.2009.
- [46] Y. Kou, S. Lin, and M. Fossorier, Eds., *Low density parity check codes based on finite geometries: a rediscovery*, 2000.
- [47] Z. Liu and D. Pados, "A decoding algorithm for finite-geometry ldpc codes," *IEEE Transactions on Communications*, vol. 53, no. 3, pp. 415–421, 2005.
- [48] Zongwang Li, Lei Chen, Lingqi Zeng, S. Lin, and W. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Transactions on Communications*, vol. 54, no. 1, pp. 71–81, 2006.

- [49] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, 2001.
- [50] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 585–598, 2001.
- [51] T. R. Amin, T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity check codes," in *IEEE Transactions on Information Theory*, 1999.
- [52] D. J. C. MacKay, S. T. Wilson, and M. C. Davey, "Comparison of constructions of irregular gallager codes," *IEEE Transactions on Communications*, vol. 47, no. 10, pp. 1449–1454, 1999.
- [53] A. Ashikhmin, G. Kramer, and S. t. Brink, "Extrinsic information transfer functions: model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, 2004.
- [54] G. Liva, S. Song, L. Lan, Y. Zhang, S. Lin, and W. E. Ryan, "Design of ldpc codes: A survey and new results," 2006.
- [55] S. Lin and D. J. Costello, *Error control coding: Fundamentals and applications*, 2nd ed. Upper Saddle River and NJ: Pearson/Prentice Hall, 2004.
- [56] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, p. 308, 1965.
- [57] A. W. Eckford, "Low-density parity-check codes for gilbert-elliott and markov-modulated channels," Ph.D. dissertation, University of Toronto, 2004.
- [58] J. Garcia-Frias, "Decoding of low-density parity-check codes over finite-state binary markov channels," *IEEE Transactions on Communications*, vol. 52, no. 11, p. 1841, 2004.
- [59] E. A. Ratzler, Ed., *Low-density parity-check codes on Markov channels*, ser. Proceedings of 2nd IMA Conference on Mathematics and Communications, Lancaster and UK, 2002.
- [60] T. Wadayama, Ed., *An iterative decoding algorithm of low density parity check codes for hidden Markov noise channels*, ser. Proceedings of International Symposium on Information Theory and Its Applications, Honolulu and Hawaii and USA, Nov 2000.
- [61] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.
- [62] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.
- [63] A. P. Dempster, N. M. Laird, D. B. Rubin *et al.*, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

- [64] I.-C. Dița, "Optimizarea recunoasterii codurilor data matrix in mediul industrial," Ph.D. dissertation, Politehnica University of Timisoara, 16.03.2012.
- [65] W. Proß and F. Quint, "Comparative study of a cdma2000 turbo code and a linear time encodable peg ldpc code over  $gf(q)$ ," in *54. IWK - International Scientific Colloquium*, Ilmenau, Germany, 2009.
- [66] W. Proß and F. Quint, "Decoding performance of turbo-codes and ldpc-codes with short blocklength," in *Scientific Bulletin of the Politehnica University of Timișoara - Transactions on Electronics and Communications*, 2009, vol. 1, pp. 25–30.
- [67] W. Proß, F. Quint, and M. Oteșteanu, "Using peg-ldpc codes for object identification," in *9th International Symposium on Electronics and Telecommunications (ISETC)*, Timișoara, Romania, 2010, pp. 361–364.
- [68] W. Proß, F. Quint, and M. Oteșteanu, "Estimation-decoding on ldpc-based 2d-barcodes," in *SIGMAP 2011 - Proceedings of the International Conference on Signal Processing and Multimedia Applications*, Seville, Spain, 2011, pp. 34–39.
- [69] W. Proß, F. Quint, and M. Oteșteanu, "Estimation-decoding of short blocklength ldpc codes on a markov-modulated gaussian channel," in *Proceedings 2011 IEEE 3rd International Conference on Signal Processing Systems (ICSPS2011)*, Yantai, China, 2011, vol. 1, pp. 383–387.
- [70] W. Proß, F. Quint, and M. Oteșteanu, "Design of short irregular ldpc codes based on a constrained downhill-simplex method," in *Scientific Bulletin of the Politehnica University of Timișoara - Transactions on Electronics and Communications*, 2011, vol. 2, pp. 27–31.
- [71] W. Proß, Quint Franz, and Oteșteanu Marius, "Design of short irregular ldpc codes for a markov-modulated gaussian channel," in *SIGMAP 2012 - Proceedings of the International Conference on Signal Processing and Multimedia Applications*, Rome, Italy, 2012, pp. 31–34.
- [72] W. Proß, F. Quint, and M. Oteșteanu, "Decoding of ldpc-based 2d-barcodes using a 2d-hidden-markov-model (in press)," in *E-Business and Telecommunications*, ser. Communications in Computer and Information Science. Springer.
- [73] W. Proß, F. Quint, and M. Oteșteanu, "Design of irregular ldpc codes for nonparametric channels: in press," in *10th International Symposium on Electronics and Telecommunications (ISETC)*, Timișoara, Romania, 2012.







