# Contents

**Abstract** **v**

**Resumen** **vii**

**Resum** **xi**

**Acknowledgements** **xiii**

**List of symbols** **xxv**

**Abbreviations and Acronyms** **xxvii**

**1 Introduction and Objectives** **1**
1.1 Background . . . . . 3
1.2 Motivation . . . . . 6
1.3 Objectives . . . . . 7
1.4 Key Contributions . . . . . 9
1.5 Organization of the Thesis . . . . . 10

**2 State-of-the-Art** **13**
2.1 System Overview . . . . . 16
2.1.1 Encoded Transmissions . . . . . 18
2.1.2 Multiuser Scenario . . . . . 20
2.2 System Architecture . . . . . 21
2.3 MIMO Detection . . . . . 21
2.3.1 Hard-Output Detection . . . . . 22
2.3.2 Soft-Output Detection . . . . . 29
2.4 High Performance Simulation Libraries . . . . . 31
2.4.1 Simulation Acceleration using MATLAB . . . . . 31
2.4.2 Simulation Acceleration using IT++ . . . . . 32
2.4.3 MIMO Design using LabVIEW . . . . . 32
2.5 Conclusion . . . . . 33

**3 Tools and Optimization Techniques 35**
3.1 Hardware Tools . . . 38
3.1.1 Multi-core Processors . . . 38
3.1.2 Graphics Processing Units . . . 39
3.1.3 Computer System for simulation testing . . . 41
3.2 Software Tools . . . 42
3.2.1 OpenMP Programming Model . . . 42
3.2.2 CUDA Programming Model . . . 43
3.2.3 MATLAB MEX-Functions . . . 45
3.2.4 HPC Linear Algebra Libraries . . . 45
3.3 Heterogenous computation . . . 47
3.4 Efficient Euclidean Distance Calculation . . . 51

**4 Implementation of Hard-Output MIMO Detectors 55**
4.1 Introduction . . . 58
4.1.1 OpenMP implementation details . . . 62
4.1.2 CUDA implementation details . . . 64
4.1.3 Assessment of parallel algorithms . . . 66
4.2 Zero Forcing SIC Detector Implementation . . . 69
4.2.1 CUDA Implementation . . . 71
4.2.2 Performance Results . . . 72
4.3 ML Exhaustive Detector Implementation . . . 74
4.3.1 CUDA Implementation . . . 78
4.3.2 Performance Results . . . 78
4.4 Schnorr-Euchner SD Implementation . . . 80
4.4.1 CUDA Implementation . . . 83
4.4.2 Performance Results . . . 84
4.5 Automatic Sphere Decoder Implementation . . . 85
4.5.1 CUDA Implementation . . . 90
4.5.2 Performance Results . . . 90
4.6 K-Best Tree-Search Implementation . . . 92
4.6.1 CUDA Implementation . . . 94
4.6.2 Performance Results . . . 95
4.7 Hard-Output Fixed-Complexity Sphere Decoder . . . 98
4.7.1 CUDA Implementation . . . 101
4.7.2 Performance Results . . . 102
4.8 WinTrees: a Divide-and-Conquer framework for Tree-Search-Based MIMO detectors . . . 103

4.9 Conclusions . . . . . 110

**5 Implementation of Soft-Output MIMO Detectors 113**
5.1 Introduction . . . . . 115
5.2 Maximum A Posteriori Probability and Max-Log Detectors Implementation . . . . . 118
5.2.1 CUDA Implementation . . . . . 122
5.2.2 Performance Results . . . . . 123
5.3 Soft Fixed Sphere Decoder Implementation . . . . . 125
5.3.1 CUDA Implementation . . . . . 129
5.3.2 Performance Results . . . . . 131
5.4 Fully Parallel Soft Fixed Sphere Demodulation . . . . . 132
5.4.1 CUDA Implementation . . . . . 137
5.4.2 Performance Results . . . . . 138
5.5 Conclusions . . . . . 141

**6 MIMOPack Software Package 145**
6.1 Introduction and Objectives . . . . . 147
6.2 Design and Specifications . . . . . 149
6.3 Documentation and Website description . . . . . 151
6.4 Support and Development . . . . . 152
6.5 Configurability and Data Structures . . . . . 153
6.5.1 Platform Configuration . . . . . 153
6.5.2 QPSK and QAM Modulation Configuration . . . . . 155
6.5.3 MIMOPack Detector Configuration . . . . . 156
6.5.4 MIMOPack WinTrees Framework Configuration . . 158
6.5.5 MIMO Simulation Configuration . . . . . 159
6.5.6 Simulation Random Data . . . . . 161
6.6 MIMO Detection Functions . . . . . 162
6.7 Installation and Test . . . . . 163
6.8 Example of simulation with MIMOPack . . . . . 164
6.9 Conclusions . . . . . 168

**7 Conclusions 169**
7.1 Main Contributions . . . . . 172
7.2 List of Publications . . . . . 173
7.3 Future Work . . . . . 176
7.4 Institutional Acknowledgements . . . . . 178

**Bibliography** **179**