MASTER

An adaptive hybrid-ARQ protocol for UWB ad-hoc networks

Vlaming, M.A.J.

Award date:
2006

Link to publication
An adaptive hybrid-ARQ protocol for UWB ad-hoc networks.

M.A.J. Vlaming

Coach: Prof.dr.ir. I. Niemegeers (TU Delft)
Dr.ir. J. Weber (TU Delft)
Dr.ir. A. Lo (TU Delft)
M. Pietrzyk, M.Sc. (TU Delft)

Supervisor: Dr.ing. P.H.A. v.d. Puttten (TU Eindhoven)

Date: February 2006

The Faculty of Electrical Engineering of the Eindhoven University of Technology does not accept any responsibility regarding the contents of Master's Theses.
Abstract

Wireless networking evolves towards providing adaptive and pervasive connections among users, services, and resources. As foreseen, ad-hoc networks will play an important role in this ubiquitous communication paradigm, where it tends to provide seamless radio connectivity within a small region. The Ultra-Wideband Impulse Radio is a promising radio technology for future short range, low power, and high data rate ad-hoc networks. For reliable and efficient utilization of the wireless link, error control coding is required.

In this thesis, a hybrid-ARQ protocol is proposed and analyzed for Ultra-Wideband systems. This protocol, and similar others, are simulated and evaluated in Matlab. Four experiments have been done to examine the characteristics of the protocols and for searching the optimum performance of the proposed protocol by changing elementary parameters. The results show that reliable and efficient high speed wireless communication can be achieved when the proposed hybrid-ARQ protocol is applied.
Acknowledgements

This thesis work was performed from January 2005 to January 2006 at the Mobile and Wireless Communication (WMC) Group of the faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS) at Delft University of Technology.

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. Foremost, Michal Pietrzyk M.Sc. for his help, support, interest and ideas during the first part of the project and dr. ir. Anthony Lo for his valuable hints during the numerous meetings we had, and his help on programming issues in matlab, in the last part of the project.

I want to thank dr. ir. Jos Weber for keeping me encouraged to finish my thesis, and I am deeply grateful to dr. ing. Piet van de Putten for his time and advice in more difficult times during this project.

Special thanks go to my friends in Delft and my friends and fellow musicians in Eindhoven, who always supported me in this project with their humor, understanding, patience, and, above all, music.

Finally, I would like to give my special thanks to my parents for their endless love and support.
# Table of Contents

Abstract

Acknowledgements

Table of Contents

List of Figures

List of Tables

1 Introduction
   1.1 The 7-layer OSI model ........................................... 3
   1.2 Wireless Ad Hoc Networks ......................................... 5
      1.2.1 Multi-hop Networks ............................................ 5
      1.2.2 Infrastructure .................................................. 6
   1.3 Ultra-WideBand ................................................... 8
      1.3.1 Introduction .................................................... 8
      1.3.2 Technology details ............................................. 11
   1.4 Error Control Coding ............................................. 19
      1.4.1 Introduction .................................................... 19
      1.4.2 Automatic-Repeat-reQuest ..................................... 19
      1.4.3 Forward-Error-Correction .................................... 24
      1.4.4 Hybrid ARQ/FEC ............................................... 28

2 Related Work ..................................................... 32
   2.1 Adaptivity ....................................................... 33
      2.1.1 Adaptive- and diversity techniques .......................... 33
      2.1.2 Adaptive Error Control ....................................... 33
   2.2 FEC in UWB systems .............................................. 37
      2.2.1 Conditions for applying FEC .................................. 37
      2.2.2 Coding-modulation scheme .................................... 37
      2.2.3 Proposed error correction techniques ........................ 37
      2.2.4 Interleaved coding schemes .................................. 39
      2.2.5 An adaptive interleaved coding scheme ....................... 39
   2.3 Enhanced ARQ schemes ........................................... 41
      2.3.1 Go-back-N ARQ scheme with selective repeat in intra-block 41
      2.3.2 An ARQ scheme with QoS provisioning ......................... 42
      2.3.3 A N-channel stop-and-wait ARQ scheme ...................... 43
   2.4 Error coding in mobile ad-hoc networks ........................ 45
2.4.1 A reliable multicast protocol .................................. 45
2.4.2 A MAC protocol for UWB ad-hoc networks ................. 45

3 The selection of the hybrid-ARQ scheme ......................... 48
  3.1 Introduction ................................................. 49
  3.2 The applied FEC scheme ...................................... 50
    3.2.1 Repetition coding ..................................... 50
    3.2.2 SOC coding ........................................... 52
    3.2.3 Rate-compatible coding ................................. 53
    3.2.4 Conclusion ............................................ 55
  3.3 The applied ARQ scheme ...................................... 57
    3.3.1 ARQ strategies ....................................... 57
    3.3.2 The hybrid-ARQ type .................................. 59
  3.4 Further improvements of the hybrid-ARQ scheme ................ 61
    3.4.1 Interleaving ........................................... 61
    3.4.2 QoS provisioning ....................................... 61
    3.4.3 General Conclusion .................................... 61

4 Simulation model .................................................. 63
  4.1 Design overview and general assumptions ...................... 64
  4.2 Design Implementation ....................................... 65
    4.2.1 Transmitter design ................................... 65
    4.2.2 Channel design ........................................ 68
    4.2.3 Receiver design ....................................... 68
  4.3 Functioning of the protocol .................................. 71
    4.3.1 Adaptivity ............................................. 71
    4.3.2 Hybrid ARQ type II using RCC codes .................. 71
    4.3.3 Hybrid ARQ type I using SOC encoding ................. 74

5 Simulations and Results .......................................... 76
  5.1 Experiment 1 ................................................. 77
    5.1.1 Simulation ............................................. 77
    5.1.2 Result analysis ....................................... 77
  5.2 Experiment 2 ................................................. 78
    5.2.1 Simulation ............................................. 78
    5.2.2 Result analysis ....................................... 78
  5.3 Experiment 3 ................................................. 80
    5.3.1 Simulation ............................................. 80
    5.3.2 Result analysis ....................................... 81
  5.4 Experiment 4 ................................................. 84
    5.4.1 Simulation ............................................. 84
    5.4.2 Result analysis ....................................... 86

6 Conclusion and Recommendations .................................. 92
  6.1 Conclusion .................................................. 93
  6.2 Future work ................................................ 95

Bibliography .......................................................... 96

A List of Abbreviations .............................................. 101
List of Figures

1.1 The OSI layer model. ............................................. 3
1.2 An example of ad hoc network (left) and multi-hop communica-
tion (right). ..................................................... 5
1.3 FCC emission limit for indoor systems. .......................... 9
1.4 Spatial Capacity of UWB. .......................................... 10
1.5 Relationship between UWB and systems beyond 3G. .............. 11
1.6 Multiband Spectrum Allocation. ..................................... 11
1.7 Four different modulation techniques for UWB pulses showing the
encoding of the data sequence \{0,1,0,1\}: (a) OOK (b) Positive
PAM (c) BPSK and (d) PPM. ....................................... 13
1.8 Gaussian waveforms in the time domain. .......................... 14
1.9 Gaussian waveforms in the frequency domain. ..................... 15
1.10 Orthogonal pulse waveform generation based on PSWF .......... 16
1.11 Multipath effects. .................................................. 17
1.12 Stop-and-Wait ARQ - Waiting for acknowledgement (ACK) from
the remote node (left) and Retransmission due to time expiry
(right). .............................................................. 22
1.13 Go-back-N ARQ - An example of retransmission. ............... 24
1.14 Selective repeat ARQ - An example of retransmission. ........ 24
1.15 Type I Hybrid ARQ - An example of retransmission. .......... 29
1.16 Type II Hybrid ARQ - An example of retransmission. ........ 30
1.17 Type III Hybrid ARQ - An example of retransmission. ........ 31
2.1 Diagram showing the coding-modulation scheme in a UWB-IR
system. .................................................................. 38
2.2 Diagram showing an interleaved coding-modulation scheme in a
UWB-IR system. ...................................................... 40
2.3 Transmission pattern of packets and blocks in the GBN-SR scheme 41
2.4 Transmission pattern of packets and blocks in the Rev-GBN-SR
scheme ................................................................. 42
2.5 Principle of N-channel stop-and-wait hybrid-ARQ (N=4) ........ 44
2.6 Dynamic channel coding ............................................ 47
4.1 The transmitter part in the complete model. ....................... 65
4.2 The frame architecture. .............................................. 66
4.3 The channel part in the complete model. .......................... 69
4.4 The receiver part in the complete model. .......................... 69
4.5 The hybrid-ARQ type II protocol using RCC coding. .......... 74
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Two error correcting coding schemes and their performance.</td>
<td>78</td>
</tr>
<tr>
<td>5.2</td>
<td>The throughput of three different hybrid-ARQ schemes.</td>
<td>79</td>
</tr>
<tr>
<td>5.3</td>
<td>Erroneous received data-packets, $p = 8, N_i = 5$.</td>
<td>80</td>
</tr>
<tr>
<td>5.4</td>
<td>Total amount of retransmissions, $p = 8, N_i = 5$.</td>
<td>81</td>
</tr>
<tr>
<td>5.5</td>
<td>Retransmissions until successful decoding is possible, $p = 8, N_i = 5$.</td>
<td>82</td>
</tr>
<tr>
<td>5.6</td>
<td>Throughput, $p = 8, N_i = 1,3,5,7$.</td>
<td>83</td>
</tr>
<tr>
<td>5.7</td>
<td>Erroneous received data-packets, $p = 8, N_i = 1,3,5,7$.</td>
<td>84</td>
</tr>
<tr>
<td>5.8</td>
<td>Total amount of retransmissions, $p = 8, N_i = 1,3,5,7$.</td>
<td>85</td>
</tr>
<tr>
<td>5.9</td>
<td>Throughput, $p = 4, N_i = 1,3,5,7$.</td>
<td>86</td>
</tr>
<tr>
<td>5.10</td>
<td>Erroneous received data-packets, $p = 4, N_i = 1,3,5,7$.</td>
<td>87</td>
</tr>
<tr>
<td>5.11</td>
<td>Total amount of retransmissions, $p = 4, N_i = 1,3,5,7$.</td>
<td>88</td>
</tr>
<tr>
<td>5.12</td>
<td>Retransmissions until successful decoding is possible, $p = 4, N_i = 1,3,5,7$</td>
<td>89</td>
</tr>
<tr>
<td>5.13</td>
<td>Throughput, $p = 2, N_i = 1,3,5,7$.</td>
<td>89</td>
</tr>
<tr>
<td>5.14</td>
<td>Erroneous received data-packets, $p = 2, N_i = 1,3,5,7$.</td>
<td>90</td>
</tr>
<tr>
<td>5.15</td>
<td>Total amount of retransmissions, $p = 2, N_i = 1,3,5,7$.</td>
<td>90</td>
</tr>
<tr>
<td>5.16</td>
<td>Retransmissions until successful decoding is possible, $p = 2, N_i = 1,3,5,7$</td>
<td>91</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Comparison of Hybrid ARQ schemes.</td>
<td>31</td>
</tr>
<tr>
<td>2.1</td>
<td>Reaction capability of some adaptive ARQ schemes.</td>
<td>34</td>
</tr>
<tr>
<td>4.1</td>
<td>Perforation matrix, $K = 7$, parent code $(117, 127, 155, 171)$</td>
<td>67</td>
</tr>
<tr>
<td>4.2</td>
<td>Nested Codes, additional polynomials, $K = 7$, parent code $(117, 127, 155, 171)$</td>
<td>67</td>
</tr>
<tr>
<td>5.1</td>
<td>Perforation matrix, $K = 7$, $p = 4$, parent code $(117, 127, 155, 171)$</td>
<td>85</td>
</tr>
<tr>
<td>5.2</td>
<td>Perforation matrix, $K = 7$, $p = 2$, parent code $(117, 127, 155, 171)$</td>
<td>85</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The future wireless communication system is envisaged where personalization and ubiquitous access to communication resources are essential [1] [2]. Users will be able to manage the relevant information by having flexible and suitable access to a vast number and variety of devices, ranging from fixed and mobile appliances, personal computing facilities, to entertainment equipments, regardless of the physical locations. This ambitious vision requires a comprehensive integration of existing and future communication- and network technologies, including infrastructure-based- and ad-hoc networks (e.g., the internet, UMTS networks, wireless Local Area Networks (LANs), and wireless Personal Area Networks (PANs), with the user at the center.

As foreseen, the short-range ad-hoc networking will play a key role in the ubiquitous communication paradigms, where it acts as a component to provide seamless connection to the infrastructure-based network among users, taking the advantage of wireless mobility. With the increasing demands of data services, the need for high data rates for video and audio distribution and high-speed data transfer, such as high-speed digital video transfers, interactive video games and printer and scanner connections, increases as well. Against this background, the promising Ultra-Wideband (UWB) radio technology, offers great potential for such short-range, high-data rate ad-hoc networks, and will play an important role in the realization of future smart and pervasive networking.

The wireless connection introduces an unreliable link between transmitter and receiver. Because of the fading effects and interference, wireless links are characterized by bursty error patterns, and high- and time-varying error rates. Therefore, error control schemes are used to protect data-packets against wireless channel impairments. There are two basic categories of error control schemes: ARQ (Automatic Repeat Request) and FEC (Forward Error Correction) schemes. ARQ schemes provide high reliability at good and moderate channel qualities. However, if the channel error rate is high like in a wireless channel, the throughput performance drops rapidly due to the increased frequency of retransmission. In order to counter this effect, hybrid-ARQ schemes are used by combining FEC with ARQ schemes.
The goal of this project is to propose an adaptive hybrid-ARQ protocol for UWB ad-hoc networks. The protocol is adaptive because the error statistics of the wireless channel are time-varying, i.e., the adaptivity of the protocol increases the efficiency of the error control scheme.

To achieve this goal an extensive literature-research in done, which also took a great amount of time in this project. The results can be found in the first chapters of this thesis. On basis of this broad research a proposal done and this is simulated and examined, which can be found second half of this thesis, and which took the second, smaller part of this project.

The structure of this thesis is as follows:

• This chapter introduces the key-topics of this thesis,
  - Section 1.1 presents the OSI layer model in order to understand the layered structure of networks,
  - Section 1.2 discusses the wireless ad-hoc network and focuses on the infrastructure of this network,
  - Section 1.3 gives an in-depth overview of the UWB radio technology,
  - Section 1.4 concentrates on error control coding. ARQ, FEC and Hybrid-ARQ are introduced and discussed here,
• Chapter 2 reviews related work in the field of adaptive protocols, FEC, ARQ and Hybrid-ARQ,
• Chapter 3 discusses several different hybrid-ARQ schemes, and considers whether they are applicable for UWB ad-hoc networks, and finally, a protocol is presented,
• Chapter 4 describes the simulation model, in which the protocol will be implemented, and with which the protocol will be examined,
• Chapter 5 gives a description of the experiments that are done to examine the protocol and presents the results,
• Finally, chapter 6 presents the conclusions that can be drawn from the results and gives recommendations for future work.
1.1 The 7-layer OSI model

In order to fully understand the technical details of underlying functions in a new technology it is necessary to understand the layered structure of networks.

A protocol is a set of rules that are agreed between communication devices on what to communicate, how to communicate and when to communicate. Due to the complexity of communication among devices, the layering concept was introduced in the design of a network. Within each layer, different protocols are defined with different functionalities. Each layer directly interacts with the layer below it and provides services to the layer above it through interfaces. Figure 1.1 depicts the open system interconnection (OSI) layer model developed by the International Organization for Standardization (ISO) [3].

<table>
<thead>
<tr>
<th>Application layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation layer</td>
</tr>
<tr>
<td>Session layer</td>
</tr>
<tr>
<td>Transport layer</td>
</tr>
<tr>
<td>Network layer</td>
</tr>
<tr>
<td>Datalink layer</td>
</tr>
<tr>
<td>Physical layer</td>
</tr>
</tbody>
</table>

Figure 1.1: The OSI layer model.

Here, the different facets of each layer are introduced with some examples, from bottom to top.

1. The **Physical Layer** concerns with the transmission of a bit stream over a physical medium. This layer deals with the mechanical and electrical properties of signals, such as the gain of the used antenna, transmission frequencies, transmission power, etc. The UWB technology, which will be discussed in section 1.3, is an emerging radio technique for transmitting bits at high speed and resides in this layer.

2. The **Datalink Layer** provides the reliable transmission of data bits across the physical medium. It defines the data frame with necessary synchronization, flow regulation, and error control, which is one of the main topics in this thesis, and will be discussed in section 1.4. This layer consists of two sub-layers, the medium access control (MAC) and the logical link control (LCC). The MAC sub-layer coordinates virtual point-to-point communications in a shared physical medium. The LCC sub-layer supervises the MAC sub-layer to offer a reliable communication. The IEEE 802.x set of standards is the main protocol suite in this layer, like IEEE 802.5.
(Token Ring) for wired network, and 802.11 for wireless local area network (WLAN).

3. The Network Layer is responsible for delivering data through various data links between two devices in a network. It defines how to establish, maintain, and terminate connections, for example, routing, forwarding, and congestion control. The Internet protocol (IP) is located in this layer.

4. The Transport Layer provides reliable and transparent data transfer between source and destination. It defines the end-to-end error recovery, flow control, multiplexing, and communication type (point-to-point, multicast, broadcast and anycast). Two main protocols in the layer are the Transport Control Protocol (TCP) and the User Datagram Protocol (UDP).

5. The Session Layer manages a logical connection (session) between two communication applications, such as, keeping track of whose turn it is to transmit.

6. The Presentation Layer provides the syntax and semantics for the representation of application data, for example, how an audio file is defined and exchanges.

7. The Application Layer contains application programs for users, that use protocols including the File Transfer Protocol (FTP), the HyperText Transfer Protocol (HTTP), Voice over IP (VoIP), etc.

In many wireless protocols, like IEEE 802.11b/a/g, and IEEE 802.15.3a, the scope of the standard that is defined is limited to the physical layer (PHY) and the MAC layer, i.e., the lower half of the datalink layer, where the LLC layer is the upper half. In this case, the other layers of the OSI model are often not mentioned.

In conclusion, the 7-layer OSI model was proved to be a good concept and theoretical model. New protocols are emerging in the OSI model to comply with the development of new applications.
1.2 Wireless Ad Hoc Networks

In areas in which there is little or no communication infrastructure or the existing infrastructure is expensive or inconvenient to use, wireless mobile users may still be able to communicate through the formation of a mobile, wireless, multi-hop ad-hoc network, which are only called ad-hoc networks here for simplicity.

Ad-hoc networks consist of mobile nodes interconnected by multi-hop communication paths. Unlike conventional wireless networks, ad-hoc networks have no fixed infrastructure or administrative support. The topology of the network changes dynamically as mobile nodes join or depart the network or radio links between nodes become unusable. See Figure 1.2 (left) for an example of an ad-hoc network and multi-hop communication.

The applications of this wireless infrastructure range from ad hoc networking (e.g., collaborative, distributed computing) to disaster recovery (e.g., fire, flood, earthquake), law enforcement (e.g., crowd control, search-and-rescue), and military (automated battlefield).

Key characteristics of this system are the large number of users (nodes), their mobility, and the need to operate without the support of a fixed (wired or wireless) infrastructure. The absence of a fixed infrastructure means that the nodes of an ad-hoc network communicate directly with one another in a peer-to-peer fashion. Each node in this network operates therefore not only as a host but also as a router, forwarding packets for other nodes in the network that may not be within direct wireless transmission range of each other.

1.2.1 Multi-hop Networks

Due to the limited transmission range of wireless network interfaces, multiple network “hops” may be needed for one node to exchange data with another across the network. An example of a multi-hop connection is depicted in Figure 1.2 (right). However, multi-hop networks can also be used in cellular networks, because of several advantages this technique possesses. Some examples are [4]:

Figure 1.2: An example of ad hoc network (left) and multi-hop communication (right).
Providing *connectivity* for nodes out of communication range. This is mainly an advantage for devices with short communication range (1–100 meters).

- Reducing *energy consumption*, when nodes are small and the communication range is large (100–10000 meters).

- Improving overall network performance via path-diversity, e.g. *throughput*.

- Enabling low *interceptability* by generating least-possible interference. Higher data rates are possible due to improved ratio of the signal to the interference (SIR)\(^1\) values at reception.

### 1.2.2 Infrastructure

As seen above, a mobile ad-hoc network is a network formed without any central administration, consisting of mobile nodes that use a wireless interface to send packet data. Since the nodes in a multi-hop network of this kind can serve as both routers and hosts, they can forward packets on behalf of other nodes, but also run user applications. Because of the dynamic environment of the network, the network functions must run in a distributed fashion, since nodes might suddenly disappear from, or show up in, the network.

An ad-hoc network could be linked to a fixed infrastructure via access points and have access to the Internet or corporate LANs. Peer-to-Peer (P2P) networks are ad-hoc networks in which an overlay network is built on the Internet. In a P2P network, two or more peers can use appropriate information and communication systems to collaborate spontaneously without requiring central coordination.

In the research activities pertaining ad-hoc networks, there are issues that primarily differentiate it from infrastructure-oriented networks. The multi-hop nature and the possible lack of a fixed infrastructure introduce new research problems such as network configuration, device discovery, and topology maintenance, as well as ad-hoc addressing and self-routing.

Below, there are some typical differences and operational characteristics for ad-hoc networks. In this case they are specified for a PAN-oriented ad-hoc network.

- **Distributed operation**: a node in an ad-hoc network cannot rely on a network in the background to support security and routing functions. Instead these functions must be designed so that they can operate efficiently under distributed conditions.

- **Dynamic network topology**: in general, the nodes will be mobile, which sooner or later will result in a varying network topology. Nonetheless, connectivity in the network should be maintained to allow applications and services to operate undisturbed. In particular, this will influence the design of routing protocols. Moreover, a user in the ad-hoc network will also require access to a fixed network (such as the Internet via a 3G access network) even if nodes are moving around.

\(^1\)SIR represents the reliability of a transmission. Noise is ignored
• **Fluctuating link capacity:** the effects of high bit-error rates might be more profound in a multi-hop ad-hoc network, since the aggregate of all link errors is what affects a multi-hop path. In addition, more than one end-to-end path can use a given link, which if the link were to break, could disrupt several sessions during periods of high bit-error transmission rates.

• **Low-power devices:** in many cases, the network nodes will be battery-driven, which will make the power budget tight for all the power-consuming components in a device. This will affect, for instance, CPU processing, memory size/usage, signal processing, and transceiver output/input power.

These last two characteristics of the infrastructure of the ad-hoc network will be of serious influence in the design of the hybrid-ARQ scheme for this type of network.
1.3 Ultra-WideBand

_Ultra-Wideband_ (UWB) technology brings the convenience and mobility of wireless communications to high-speed interconnects in devices throughout the digital home and office. Designed for short-range, wireless personal area networks (WPANs), UWB is the leading technology for freeing people from wires, enabling wireless connection of multiple devices for transmission of video, audio and other high-bandwidth data.

UWB, short-range radio technology, complements other longer range radio technologies such as wireless local area networks (WLANs) or Wi-Fi, WiMAX, and cellular wide area communication technologies. It is used to relay data from a host device to other devices in the immediate area up to 10 meters.

This section covers that UWB technology. First a general introduction to UWB is given, then the technology of UWB Impulse Radio is extensively discussed.

1.3.1 Introduction

An UWB system is defined as a communication system with the total bandwidth larger than 500 MHz, or a fractional bandwidth of more than 20%. The fractional bandwidth is defined as in Equation (1.1)

\[ BW_f = \frac{2(f_H - f_L)}{(f_H + f_L)} \tag{1.1} \]

where \( f_H \) and \( f_L \) are frequencies\(^2\), defined at a -10 dBm level [5].

UWB technology, also known as baseband, carrier-free or impulse technology, is based on transmission of extremely short pulses, in the order of nanoseconds, which makes it completely different from any other radio technology. A series of pulses uses all frequencies, instead of one particular frequency. In order to turn this amount of frequencies into effective information, the exact timing of the pulses is very important.

The potential strength of the UWB radio technique lies in the extremely wide transmission bandwidths, which results in desirable capabilities including accurate position location and ranging, lack of significant fading, high multiple access capability, covert communications and possible easier material penetration.

UWB for commercial data communication

In the past 20 years UWB has been used for radar, sensing, military communications and niche applications [6]. But a substantial change occurred in February 2002 when the American Federal Communications Commission (FCC) issued a ruling that UWB could be used for commercial data communications as well as radar and safety applications.

The FCC has allocated 7,500 MHz of spectrum for unlicensed use of UWB devices in the 3.1 to 10.6 GHz frequency band. The FCC ruling allows UWB devices to operate without requiring a license, but they must comply with certain regulations to mitigate potential interference with other wireless systems.
communication devices to operate at low power (a peak power-spectral-density\(^3\) (PSD) value of only \(-41.3\) dBm/MHz). The FCC emission limit for indoor systems is depicted in Figure 1.3. These low emission limits for the UWB radio are to ensure that UWB devices do not cause harmful interference to "licensed services and other important radio operations" [5].

![Figure 1.3: FCC emission limit for indoor systems.](image)

The FCC ruling applies only in the USA. In Europe, Japan, China and other parts of the world, local organizations and governing bodies are currently defining UWB regulations for commercial applications.

The unique properties of the UWB wireless technology, including multiple very high data rate streams, low power consumption and low cost, make it a promising technology for fourth generation short-range wireless communications [7].

**Personal Area Networks**

Because of the restrictions on the transmit power, UWB communications are best suited for short-range wireless communications: sensor networks and personal area networks (PANs). The IEEE established a standardization group, IEEE 802.15.3a, late 2001, which has developed a standard for UWB PANs [8], based on multiband UWB. Multiband UWB will be discussed in section 1.3.2. The standard defines data rates of 55 Mb/s, 110 Mb/s, 200 Mb/s and has capabilities for data rates up to 480 Mb/s.

\(^3\)PSD is defined as the amount of power per unit (density) of frequency (spectral) as a function of the frequency. The PSD describes how the power (or variance) of a time series is distributed with frequency.
To compare the potential of UWB to other short-range technologies, 'spatial capacity' is introduced, meaning the throughput as a function of the coverage area. Figure 1.4 shows that none of the other contemporary standards, such as Bluetooth or IEEE 802.11a, is capable of reaching the capacity of the UWB.

An indication for this can be found from the Channel Capacity theorem (Equation 1.2) which states that the channel capacity grows linearly with the bandwidth and only logarithmically with the signal to noise ratio. UWB systems have much more capacity expansion space, because of their large bandwidth, than conventional narrowband systems, constrained by their limited bandwidth.

The Channel Capacity theorem is stated as:

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

(1.2)

where $C$ is the maximum channel capacity (bits/s), $B$ the channel bandwidth (Hz), $S$ the signal power (Watt) and $N$ the noise power (Watt).

The trends that drive recent R&D activities carried out for UWB transmission for commercial communication applications include [9]: (a) increasing demand for low-cost portable devices providing high rate transmission capability at lower power than currently available, (b) lack of available frequencies, and crowding in currently assigned unlicensed frequency bands, (c) increasing availability of wireline high-speed Internet access in enterprises, homes and public places, and (d) decreasing semiconductor cost and power consumption for signal processing.

Figure 1.5 shows the area of application for UWB with respect to existing and future networks. The low power restricts UWB to very short-range high data applications, or very low data rates for moderate-range applications and effectively prohibit UWB from most outdoor applications.
1.3 Ultra-WideBand

1.3.2 Technology details

The research for UWB in this project is done in the frequency domain for commercial data communication, 3.1 to 10.6 GHz, as defined by the FCC ruling of February 2002 [5].

UWB Impulse Radio and Multiband UWB

There are two general ways to use the bandwidth available for UWB. Impulse Radio (IR) was the original approach to UWB. It involves the use of very short-duration baseband pulses that use a bandwidth of several Gigahertz. A more recent approach to UWB is a multiband system where the UWB frequency band from 3.1-10.6 GHz is divided into several smaller bands. Each of these bands has a bandwidth larger than 500 MHz.

Impulse radio has to coexist with existing narrowband systems, however, multiband UWB can avoid transmitting on the frequency bands where other wireless systems are present by not using those frequency bands. Figure 1.6 illustrates the division of the frequency bands.

In comparison with UWB-IR, the multiband approach causes more inter-

Figure 1.5: Relationship between UWB and systems beyond 3G.

Figure 1.6: Multiband Spectrum Allocation.
ference for the same amount of radiated power. The UWB-IR approach offers the greatest performance, lowest power consumption, while causing the least amount of interference of any known method of UWB [10]. Finally, the hardware (in particular the transmitter hardware) can be very simple and therefore small and low-cost; more specifically, UWB impulse-radio signals can be generated without the need for stable RF frequency sources [11].

For this project, Impulse Radio will be used as the UWB radio technique. In the next sections the transmission, channel and reception of an UWB-IR signal will be discussed.

Transmission

In this section is explained how a data-packet from the network layer is converted to an UWB-IR signal that can be transmitted by an antenna.

Data consists of packets of a number of information bits. Before the data can be transmitted, several actions have to be performed. One of them is the application of a selected coding scheme for error detection and correction. Error detection and correction coding will be extensively discussed in section 1.4. After coding is applied, the code symbols are modulated on pulses and these UWB pulses are transmitted through an antenna.

Modulation An UWB pulse itself contains no data. Therefore long sequences of pulses, i.e. a pulse train, defined by the pulse repetition frequency (PRF), with data modulation are used for data transmission or communication. So, by modulating a pulse train, information can be transmitted.

Typically, pulse amplitude modulation (PAM) [12][13], pulse position modulation (PPM) [14][15], biphase modulation (binary phase shift keying, BPSK)[16] or On/Off Keying (OOK) [17] modulation are employed as modulation techniques for UWB. PPM is a modulation technique wherein data is represented by time shifts from a reference time. PAM is a modulation technique that varies the amplitude of the transmitted pulses based on the data to be transmitted. OOK is a special case of PAM UWB modulation wherein the presence or absence of a pulse within a time slot represents a ‘1’ or a ‘0’. In BPSK a specific pulse and its negative are used to represent a ‘0’ and a ‘1’. Another name for biphase modulation is impulse- or pulse polarity modulation. Figure 1.7 shows a sequence of pulses to illustrate each of the above techniques.

Another way of modulating information bits onto waveforms, is the use of different shapes of waveforms for each information bit. This modulation technique is called 'Pulse Shape Modulation' (PSM). Because interference between the waveforms should be prevented, orthogonal waveforms, which are mutually independent, are very suitable for PSM.

Pulses An impulse radio signal is seen as a carrier-less baseband transmission. The absence of a carrier frequency is a very fundamental character that differentiates impulse radio from narrowband applications. The transmitted UWB signal consists of a train of very narrow pulses at baseband, normally on the order of a nanosecond. Each transmitted pulse is referred to as a monocycle.
1.3 Ultra-WideBand

Figure 1.7: Four different modulation techniques for UWB pulses showing the encoding of the data sequence \{0,1,0,1\}: (a) OOK (b) Positive PAM (c) BPSK and (d) PPM.

Certain pulse shapes allow for better control of the frequency content of the radiated UWB emission, which can offer better coexistence with radiocommunication systems. It is fundamental that pulse shapes for UWB communications must have a zero mean so that the frequency spectrum is zero at zero frequency. This is necessary to allow the signal to radiate effectively because an antenna cannot radiate direct current (i.e., signals at zero frequency).

Pulse shapes that have been proposed or used for UWB communication system include [18]: a Gaussian biphase monocycle, a doublet, and a burst carrier wave, which is a number of cycles of a sine wave within an envelope that is a half cycle of a sine wave, both positive and negative. One important feature that this pulse has, is a distinct centre frequency.

**Gaussian waveform** Conventional UWB systems make use of Gaussian functions [19][18]. An UWB pulse is produced by triggering a switch and the effective excitation is a smooth impulse which is approximately a Gaussian pulse (Equation (1.3)). An advantage of using Gaussian functions is that they are easy to describe and work with.

\[
w(t) = Ae^{-\left(\frac{t}{\sigma}\right)^2}
\]  

(1.3)

In Equation (1.3) is \(A\) the amplitude and \(\sigma\) the temporal width parameter.

With the Gaussian pulse as input, the UWB antenna acts as a differentiator to produce at its output the first derivative of the Gaussian pulse, which is known as the Gaussian monocycle (Equation (1.4)).

\[
w(t) = \frac{-2At}{\sigma^2}e^{-\left(\frac{t}{\sigma}\right)^2}
\]

(1.4)
At the receiver side, the receiver antenna performs a differentiation on the signal too. The received signal is the second derivative of Equation (1.3) and is called the Gaussian doublet (Equation (1.5)).

\[ w(t) = \frac{-2A}{\sigma^3} \left(1 - \frac{2t^2}{\sigma^2}\right) e^{-(t^2/\sigma^2)} \]  \hspace{1cm} (1.5)

Figures 1.8 and 1.9 show three examples of the above introduced Gaussian waveforms in the respectively time and frequency domain.

Figure 1.8: Gaussian waveforms in the time domain.

As can be seen from Fig. 1.9 the bandwidth of the Gaussian monocycle does not satisfy the FCC spectral mask. The bandwidth can be controlled and refined by using filters. Filtering UWB signals means taking higher order derivatives. Properties of waveforms produced by taking higher order derivatives of (1.3) have been widely studied [20], and it is shown that the 5th derivative of (1.3) could possibly generate a waveform that meets the spectral requirement of the FCC. This 5th derivative of the Gaussian pulse can be implemented by a 4th order Chebychev highpass filter.

Prolate Spheroidal Wave Functions As seen in the previous section, the Gaussian monocycle violates the frequency restrictions of the FCC, and must be modified and filtered to meet the FCC requirements. Filtering tend to lengthen the pulse duration, thereby reducing the data rate and system capacity. Because of the extra filtering, the transmitter design is more difficult. Furthermore, UWB pulses that are successively higher derivatives of the Gaussian
1.3 Ultra-WideBand

Pulse have successively lower fractional bandwidths (see Eq. 1.1). A small fractional bandwidth results in severe worst-case performance in multipath fading for UWB communication systems. To get around all these disadvantages, different pulse waveform has become popular.

These new techniques for generating UWB pulses make use of the feature of orthogonality of pulses. These orthogonal pulses are generated using modified Hermite polynomials [21] or prolate spheroidal wave functions (PSWF) [22]. Pulses provided by PSWFs use much less hardware complexity than those provided by modified Hermite polynomials, because the latter needs a hardware set for each orthogonal pulse shape, while PSWFs can be generated by one simple source.

More features of PSWFs are that they are both time- and bandwidth-limited, the pulse duration is exactly the same for all pulses, the bandwidth is almost the same for all pulses, the pulses are doubly orthogonal to each other, what makes them very suitable for achieving high throughput even in a severe multipath fading environment (which will be discussed later), the pulses have a non-zero DC component and pulse duration and bandwidth can be controlled simultaneously.

These pulse waveforms \( \psi(t) \) are developed utilizing Prolate Spheroidal Wave Functions (PSWF) [23][24][25] and are the solutions of:

\[
\lambda \psi(t) = \int_{-T_p/2}^{T_p/2} \psi(\tau) \sin W(t - \tau) \frac{d\tau}{t - \tau}.
\]

In Eq. (1.6) \( W \) is the bandwidth and \( \lambda \) is the energy concentration of \( \psi(t) \) that
lies in the time slot \([-T_p/2, T_p/2]\).

Figure 1.10 shows PSWF-based orthogonal pulse waveforms with different orders (order 1, 2 and 3), matching the FCC spectral mask requirement of indoor/outdoor communications.

![Optimized pulse waveform generation based on PSWF](image)

**Figure 1.10**: Orthogonal pulse waveform generation based on PSWF (3.1–10.6 GHz, order of 1, 2 and 3).

\[ M \]-ary\(^4\) PSM can be used for achieving high data rates even in severe multipath environments [24], based on the orthogonality of pulse waveforms of PSWFs. A set of \( M = 2^k \) biorthogonal PSM (BPSM) signals are constructed from \( 2^k-1 \) orthogonal pulse signals by inculding the inverses of the orthogonal pulse signals. It is shown that BPSM [26] has better performance and less complexity than the conventional \( M \)-ary PAM or PPM.

**Propagation channel**

To design systems that make optimal use of the spectrum, a fundamental understanding of the UWB propagation channel, where UWB devices are operating in order to help implement an efficient high data rate communications system, is important. Analysis of communication systems is always based on a channel model, and all results obtained from the analysis can only be as reliable as the model used.

Due to reflection, refraction and scattering, wireless signals usually experience multipath propagation. In Figure 1.11, the reflected path (R) causes a delayed version of the signal that is received via the direct path (D) causing both signals to interfere with each other. In a narrowband system, many reflecting paths exist, which is called multipath fading, while in UWB systems the

\(^4M \) is likely to be \( 2^k \), with \( k \in \mathbb{N} \).
monocycles often do not overlap because the pulse width is often smaller than the channel propagation delays. In case of a non-line-of-sight (NLOS) between transmitter and receiver, multipath fading has a distinct effect.

![Multipath effects](image)

Figure 1.11: Multipath effects.

A widely used channel model is the flat Rayleigh-fading channel. The assumption of flat fading can be used when the considered system bandwidth is so small that the delays of the individual multipath components (MPCs) do not impact the system performance. Thus, at the receiver, all the MPCs can interfere (constructively or destructively, this is called the inter-pulse-interference (IPI) and the inter-symbol-interference (ISI)). If there is a large distribution of MPCs, the complex amplitude has a complex Gaussian distribution, which results in a Rayleigh or Rician distribution of the amplitudes (envelope). This model has been sufficient for narrowband wireless systems.

When bandwidth is larger, the different delays of multipath components influence the system performance, and have to be modeled. Best model that was chosen by the IEEE 802.15.3a is based on the Saleh-Valenzuela (S–V) model.

Reception

The UWB channel introduces large-scale path loss, shadowing\(^5\), small-scale fading, and propagation delay dispersion to the transmitted monocycle train [18]. The distorted waveforms arriving at the receiver are further corrupted by multiple access interference, narrowband jamming, and background noise. The function of the receiver is to extract the information bit sequence modulated on the monocycle train from the distorted and corrupted receiving waveforms with high accuracy. In general, the receiver consists of a detector and a decision device.

Most research on the receiver architectures focuses on correlation or rake receivers, that can achieve the optimal performance. A Rake receiver [27] exploits the diversity by constructively combining the separable monocycles from distinguishing propagation paths for improving transmission performance. A Rake receiver has several antennas, called Rake fingers, to pick up the different propagated signals. However, for large number of multipath components, a rake receiver is not practical because of increased receiver structure complexity.

A Rake receiver combining all the paths of the incoming signal is called an all-Rake (ARake) receiver. Since a UWB signal has a very large bandwidth, the number of resolvable multipath components is usually very large. Hence, an ARake receiver is not implemented in practice due to its complexity. However, it serves as a benchmark for the performance of more practical Rake receivers.

---

\(^5\)A weakening of signal strength caused by obstructions in a radio signal's path
A feasible implementation of diversity combining can be obtained by a selective-Rake (SRake) receiver, which combines the $M$ best, out of $L$, multipath components. Although an SRake receiver is less complex than an ARake receiver, it needs to keep track of all the multipath components and choose the best subset of multipath components before feeding it to the combining stage. A simpler Rake receiver, which combines the first $M$ paths of the incoming signal, is called a partial-Rake (PRake) receiver [28].
1.4 Error Control Coding

1.4.1 Introduction

In modern communication systems, including UWB systems, a certain level of reliability of information that is transmitted from a sender to its destination is needed. Next to that, the communication between sender and receiver should be efficient.

Noise causes the received data to differ slightly from the original data. As Shannon [29] showed in 1948, the noise doesn’t need to cause any degradation in reliability. The noise does impose some limiting capacity on the throughput rate, although that limit is typically well above the throughput rate at which real systems operate.

There are several ways to attain a desired level of efficiency in, and reliability of transporting the information. One way is to transmit a signal with enough power to overcome channel disturbances that cause errors. A different way to achieve this, is to add some redundancy into the transmitted data stream. In that way, the receiver can detect and possibly correct errors that occur during transmission. These techniques are called error control coding, and in the context of UWB systems are the subject of this section.

Error control codes, also referred to as channel codes, are intended to enhance the transmitted information as far as increasing its immunity to noise and other channel impairments is concerned. This is achieved by inserting redundant bits into the transmitted information stream, allowing the receiver to detect, and possibly correct, a number of errors. There exist two main types of error control coding:

• Automatic Repeat reQuest (ARQ) schemes, that recognize whether the of the received information contains an error, and request a retransmission;

• Forward Error Correcting (FEC) schemes, that not only allow to detect, but also to correct errors without the need for retransmission.

The reliability of digital communications systems is often expressed in terms of bit error probability $P_e$, i.e., the probability that a bit delivered to the receiver is different from the corresponding bit generated by the source. This probability depends on the modulation and coding methods, on the type of channel, and on the received signal-to-noise ratio (SNR).

When channel coding is applied, less energy may be required in order to achieve a certain bit error probability, than without coding. The difference $E_b/N_0$, expressed in decibel$^6$ (dB), is called coding gain.

This section is organizes as follows. Subsection 1.4.2 introduces different ARQ schemes. Subsection 1.4.3 presents different FEC schemes. Finally, subsection 1.4.4 discusses hybrid ARQ/FEC schemes.

1.4.2 Automatic-Repeat-reQuest

ARQ sees to it that errors are discovered and that a retransmission is requested. A return channel from the receiver to the transmitter is necessary to provide

$^6$Ratio of energy transmitted per one bit to the noise spectral density
feedback about the status of the received information. Detected lost frames are then retransmitted, until they are successfully received or they are discarded because the allowed overall transmission delay for packet transmission has been reached.

An important parameter of ARQ protocols is persistency, which defines how long the link is allowed to delay the frame before giving up and discarding it. The persistency can be measured in time or in a maximum number of retransmissions.

For the detection of errors, one way is to apply a code that is used on a large scale: the polynomial code, also known as the cyclic redundancy code or CRC. Further, repetition schemes and parity schemes are also used to achieve error detection.

There are two different schemes that are used for the retransmission [3]. The first ARQ system is stop-and-wait. It can be used for half-duplex channels. The second ARQ scheme is continuous. It can be used for full-duplex channels (Stop-and-wait-protocols can be also used on these channels, but not the other way around). There are two options in the case of continuous ARQ schemes: the go-back-N protocol and the selective repeat protocol.

In the next paragraphs, three schemes of error detection and the three different schemes for retransmission are discussed.

Error detection Several schemes exist to achieve error detection, and are generally quite simple: Polynomial codes, repetition schemes and parity schemes.

Polynomial codes When using polynomial codes, bitstrings are used for representations of polynomials with only '0' and '1' as coefficients. A frame of \( k \) bits is considered as the list with coefficients of a polynomial with \( k \) terms, which run from \( x^{k-1} \) to \( x^0 \). The degree of such a polynomial is \( k - 1 \). The most significant bit (most left) is the coefficient of \( x^{k-1} \), the next bit is the coefficient of \( x^{k-2} \), etc. Calculations with polynomials are done modulo 2 and are usually done using hardware.

When using the polynomial code for error discovery, sender and receiver have to agree on a generator polynomial \( G(x) \) in advance. To calculate the checksum of a frame of \( b \) bits that corresponds to the polynomial \( M(x) \), the frame has to be longer than the generator polynomial. The concept used here, is that there is a checksum added at the end of the frame in such a way, that the polynomial that is represented by the frame, extended with the checksum, is divisible by \( G(x) \). If the receiver gets a frame with checksum, he tries to divide it by \( G(x) \). If there is a remainder, then there has been a transmission error.

Using a polynomial code, all errors can be detected, except errors which correspond to a polynomial that had \( G(x) \) as a factor. This can be easily seen from Equation (1.7).

\[
\text{Checksum} = \frac{T(x) + E(x)}{G(x)} \quad (1.7)
\]

\(^7\)In a half-duplex channel, there can only be one-way transmission at a time
\(^8\)In a full-duplex channel, there can be simultaneous two-way transmission
1.4 Error Control Coding

$T(x)$ is the transmitted frame with the checksum and $E(x)$ the transmission error. $T(x)/G(x)$ is 0, so the result of the calculation is equal to $E(x)/G(x)$. If $E(x)$ is a factor of $G(x)$, the result is 0, and the received frame is accepted and no retransmission is requested.

**Repetition schemes** Detection of repetition codes is based on the majority of the received bits. Given a stream of data that is to be sent, the data is broken into blocks of bits, and during the transmission, each block is sent some predetermined number of times. As one group is not the same as the other groups, there can be determined that an error has occurred. The system is not very efficient, and can be susceptible to problems if the error occurs in exactly the same place for each group.

**Parity schemes** Parity schemes are based on the amount of '1's in a block of bits. Given a stream of data that is to be sent, the data is broken up into blocks of bits, and the number of '1' bits is counted. Based on this number, one bit of redundant data is used, specified as follows: if the number of '1' bits is even, this bit is '0', otherwise, if the number of '1' bits is odd, this bit is '1'.

Error detection is achieved by recalculating the redundant data bit at the receiver side, using the first eight bits. It can be seen that even if the redundant data bit is corrupted, error detection still occurs.

There is a limitation to the parity scheme. Suppose that two errors occur, on recalculation of the redundant data bit, the receiver finds that this bit is correct that the block is received correctly! So, parity schemes detect only single-bit errors.

**Stop-and-wait ARQ protocol** For better understanding how the three retransmission protocols work, there is first some elementary information presented about frames and data packets that are used in the datalink layer. After that, the protocol is presented.

**Frames** The datalink layer on the sender side, receives data from network layer and sends it on to the physical layer. The datalink layer at the receiver side, gets data-frames from the physical layer and has to send it on to the network layer. During the processing of the data in the datalink layer, there are several operations done, like synchronization of data frames, flow regulation and error control.

When looking at the receiver side, the datalink layer receives frames from the physical layer. A frame consists of several fields, for example: kind, seq, ack and info. The first free contain operating information and the last one may contain the actual data, that has to be sent on to the network layer. The fields that contain the operating information are jointly called the frame-header.

The field kind indicates what kind of frame it is, because some protocols make a difference between frames with only operation information and frames that contain data as well. The fields seq and ack are used for sequence numbers, respectively acknowledgements. The field data of the frame contains only one data-packet.
The protocol The protocols that are used when the sender transmits only one frame and then waits for an acknowledgement, are referred to as *stop-and-wait-protocols*.

When the receiver receives a data packet, it sends a positive acknowledgement (ACK) if the data is received correctly, or a negative acknowledgement (NACK) if the data was not received correctly. The *ack-field* in the frame-header is used for this purpose. The *data-field* is left empty. See Figure 1.12 (left) for an example.

During the transmission from the receiver of the frame to the original sender, the frame with the acknowledgement (positive or negative) could be damaged or could perish completely. In the first case, the sender would discard the damaged package and wait for an indefinite time. In the second case, the sender would also wait indefinitely. To solve this problem, a timer is implemented to recover from these two conditions. See figure 1.12 (right) for an example.

![Figure 1.12: Stop-and-Wait ARQ - Waiting for acknowledgement (ACK) from the remote node (left) and Retransmission due to time expiry (right).](image-url)

In the case that an ACK is sent by the receiver, but perishes along the way, the sender will perform a retransmission after the timer is finished. Then, the receiver would receive a copy of the earlier received frame, and if correct, would send the data on to the network-layer. In that case, this data occurs twice in the network-layer, and this is incorrect. To prevent this, a sequence number is added in the *seq-field* in the frame-header. In this way, the receiver can check the sequence number of each frame to see whether it involves a new frame or that it involves a repetition, which can be discarded.

Under normal transmission the sender will receive an ACK for the received data and will then start transmission of the next data block. For a long delay link, the sender may have to wait an appreciable time for this response. While it is waiting, the sender is said to be in "idle" state and is unable to send further data.

**Go-back-N ARQ protocol** The next two protocols, *Go-back-N* and *Selective Repeat* are sliding window protocols. First, the principle of sliding windows is explained, then, the Go-back-N protocol is presented.

**Sliding windows** In most cases, data-frames are sent in two directions: there is data-transmission between two nodes. In this case of full duplex transmission, the *data-field* is used in both directions, as well as the *ack-field*. By
1.4 Error Control Coding

looking at the type-field in the frame-header, the receiver is able to discern whether an incoming frame is a data-frame or an acknowledgement-frame.

It is possible that the receiver waits before sending an acknowledgement-frame until the network layer provides a data-packet for transmission, and combines these two in one frame. This is known as piggybacking. Thus, instead of sending two frames, there will be sent only one.

Piggybacking introduces new problems too. The timer-period of the sender should be extended. The receiver shouldn’t wait too long for a data-packet to send, otherwise the timer of the sender expires. That’s why the receiver has got a timer as well, and if this one expires and there hasn’t been a packet from the network-layer, a separate acknowledgement-frame is sent.

With all sliding window protocols, each outgoing frame contains a sequence number between 0 and some maximum. Generally, the maximum is $2^n - 1$, so that the sequence number fits into a field of $n$ bits.

The essence of all sliding-window-protocols is that the sender keeps up a list, at any moment, of sequence numbers that correspond with frames, which the sender may transmit. These frames are part of the sending window. In the same way, the receiver keeps up a list, the receiving window, that corresponds to the set of frames the receiver may receive. These two windows don’t have to be equal in size.

The sequence numbers in the window of the sender represent the frames that are not yet acknowledged. Each time a packet enters from the network-layer, the packet receives the next number, and the upper limit of the window is raised with one. If an acknowledgement-frame enters, the lower limit is raised with one.

Because frames that are currently in the window of the sender could be damaged or could perish during the transmission, the sender has to keep all these frames in a buffer for a possible retransmission. The buffer should be as large as the maximum size of the window. When the window reaches its maximum size, the datalink layer has to stop the network layer sending packets, until a place in the buffer is released.

The protocol The stop-and-wait principle is a very easy, but not (yet) very efficient. The sender has to wait for an acknowledgement, before sending a new frame. With the continuous protocols, the sender can keep sending frames during the time that is equal to the transmission time to and fro. That is, the sender keeps sending until the first acknowledgement arrives. The maximum window size is then the number of frames that are sent in that time. This technique is know as pipelining.

Pipelining introduces some new problems too, as far as damaged or lost frames are concerned. With the go-back-N protocol, the receiver discards all frames after an erroneous frame (damaged or wrong order) is received and sends no acknowledgements for the frames that are discarded. The receiver uses a receiving window of size 1. When the timer expires at the sender or when it receives a negative acknowledgement, the sender will retransmit again all unconfirmed frames in the original order. See Figure 1.13 for an example of the go-back-N protocol. When using the go-back-N protocol, a lot of bandwidth could be wasted if the frequency of errors is high.
24 Introduction

Figure 1.13: Go-back-N ARQ - An example of retransmission.

**Selective repeat ARQ protocol** The pipelining technique is also used in the selective repeat protocol. But with the selective repeat protocol, the datalink layer stores, at the receiver side, all correct frames that arrive after the erroneous frame. The receiver uses a receiving window of size > 1. When the sender notices, finally, that something is wrong, it only retransmits the erroneous frame. If the second attempt succeeds, the datalink layer at the receiver side holds a large number successive correct frames. These frames can all quickly be passed on to the network layer, and the highest number can be acknowledged. See Figure 1.14 for an example of the selective repeat protocol.

Figure 1.14: Selective repeat ARQ - An example of retransmission.

1.4.3 **Forward-Error-Correction**

As seen in the last section, ARQ introduces variable delays, because of the retransmissions. With selective repeat, delays are brought back to a minimum, but extra complexity in the protocol is added by using buffers.

Introducing **Forward-Error-Correction** (FEC) to the information means adding sufficient redundancy to the bit stream so that it can be properly reproduced at the receiver side. Thus, there is no retransmission required, there is a fixed transmission time and no return channel is required. However, FEC introduces higher decoding complexity than ARQ.
According to the manner of mapping sequences of message symbols to the sequences of input labels to the moderator, there are two main coding techniques:

- block coding
- convolutional coding

Block codes operate in block-by-block fashion, where each codeword depends on the current input message block.

In convolutional codes, the output of the encoder depends not only on the current input information, but also on the previous inputs or outputs of the encoder, depending on the structure and the memory of the device. The output code sequence can be viewed as a convolution (in discrete time and alphabet) of the input message sequence with the encoder’s impulse response. In contrast to the block codes, the information symbols used in convolutional coding are considered as a stream rather than a block.

In the next two paragraphs, the two main coding techniques will be presented. Block codes will be introduced first, and after that convolutional codes will be discussed.

**Block codes** Before introducing different types of block codes, there will be, next, an introduction to these codes, discussing the fundamentals.

**Introduction** A block code consists of elements called codewords, and \( n \) is called the length of the code. The message generated by the source has length \( k \). The code rate \( R \) is a measure for the natural efficiency of a code. The code rate equals the ratio between \( k \) and \( n \): \( R = \frac{k}{n} \). The block codes can be described using vectors/matrices or polynomials.

The Hamming distance between two codewords of the same length \( n \) is defined as the number of positions in which these two codewords differ. The Hamming distance between a codeword and the all-zero codeword \( \mathbf{0} \) is called the weight of the codeword.

The Hamming distance \( d \) of a code is defined as the smallest Hamming distance between any two different codewords. For linear codes this equals the minimum weight of all codewords except the all-zero codeword.

The Hamming distance \( d \) is an important parameter of a code, because when \( d \) is known, the error probabilities of the code can be derived:

- A code can correct \( t \) errors \( \iff d \geq 2t + 1 \)
- A code can detect \( s \) errors \( \iff d \geq s + 1 \)

Linear block codes of length \( n \), dimension \( k \), and Hamming distance \( d \) may be denoted as \( (n,k,d) \) codes.

Block codes have predetermined lengths, but sometimes this is not desired, because applications demand other lengths. The codes can then still be used by applying techniques like shortening, puncturing, or extension [30].

---

\(^9\) \( k \) is also called the dimension of a code  
\(^{10}\) The all-zero codeword contains only zeros
With shortening, any \((n,k,d)\) code can be shortened to a \((n-a,k-a,d)\) code, where \(a\) is an integer such that \(1 \leq a \leq k-1\). Any \((n,k,d)\) code can be punctured to a \((n-a,k,d-a)\) code, where \(a\) is an integer such that \(1 \leq a \leq d-1\). Any binary code \((n,k,d)\) code for which the Hamming distance \(d\) is odd can be extended to a \((n+1,k,d+1)\) code.

In wireless networks errors tend to occur in bursts. Most error-correcting codes have been designed to handle random errors. A way to deal with this discrepancy is a technique called interleaving. The interleaver is a device that rearranges and spreads out the disposition of a symbol sequence in a deterministic way. The de-interleaver is a device, which through an inverse permutation restores the symbol sequence to its original order.

**Some emerging block codes** One of the oldest error-correcting codes is the class of binary *Hamming Codes*. These codes have a Hamming distance equal to three. Hence, the code is single error-correcting or double error detecting. The length of the codewords is \(n = 2^m - 1\) and the dimension is \(k = 2^m - 1 - m\), with \(m\) an integer at least equal to two.

Cyclic codes are linear codes with the additional property that a cyclic shift of any codeword gives again a codeword. There are two important families of cyclic codes: BCH codes and RS codes.

Binary *Bose, Chaudhuri & Hocquenghem* (BCH) codes are characterized by two parameters \(m\), which determines the length of the code, and \(t\), which determines a lower on the code's Hamming distance. The length of the codewords is \(n = 2^m - 1\), the dimension is \(k \geq 2^m - 1 - mt\) and the Hamming distance is \(d \geq 2t + 1\). BCH codes form a flexible family of codes, where the parameters \(m\) and \(t\) can be chosen to satisfy the applications efficiency and reliability requirements. Increasing the parameter \(t\) leads to higher reliability, as the price of a lower efficiency (code rate).

The *Reed-Solomon* (RS) codes are non-binary codes, which may be defined over any finite field \(GF(q)^{11}\) [30]. However, since most modern applications make use of binary data, RS codes over alphabet \(GF(2^n)^{12}\) are considered here.

Like BCH codes, the RS codes are characterized by the two parameters \(m\), which determines the length of the code, and \(t\), which determines the code's Hamming distance. The length of the codewords is \(n = 2^m - 1\), the dimension is \(k = 2^m - 1 - 2t\) and the Hamming distance is \(d = 2t + 1\). Just like the BCH codes, \(m\) and \(t\) can be chosen flexible. RS codes are particularly suited for the correction of burst errors.

Co-operating codes are codes where two (or more) block codes are combined into new block codes, which have good error correction capabilities for combinations of random and burst errors. The new codes make use of the encoding and decoding of the constituent codes, so the complexity can be held rather low. Two kinds of co-operating codes are considered: the product codes and the concatenated codes.

---

11 Galois Field of order \(q\). The field has a finite number of elements, called the order of the field and is always a power of a prime number.

12 Set of elements of size \(q\) \(\{(\alpha_1, \alpha_2, \ldots, \alpha_q)\}\), mostly \(GF(2)\) (i.e., \(q = 2\), the alphabet is then \(\{0,1\}\))
The *product codes* are co-operating codes where two (or more) codes over the same alphabet are combined. The constituent codes may or may not be identical. The length, dimension, and Hamming distance of the product code are equal to the products of the corresponding parameters of the constituent codes.

The *concatenated codes* are co-operating codes where a hierarchical structure is established by combining an inner code over a low-order alphabet with an outer code over a higher alphabet. Extension to more codes is conceivable.

The concatenated code is defined over the same (mostly binary) alphabet as the inner code. Its length, dimension, and distance are the products of the corresponding parameters of the inner and the outer code.

**Convolutional Codes** A *convolutional code* is a linear code with memory. It encodes a stream of data symbols, taking $k$ simultaneous symbols and producing $n$ simultaneous output symbols on every clock. The *memory* ($M$) of a convolutional encoder is the number of consecutive $k$-symbol blocks that the encoder can store.

At each clock tick, the $n$ output symbols produced are functions of the symbols in memory during the previous time interval and of the inputs.

When the number of memory cells in a convolutional encoder is $M$, the *constraint length* is $M + 1$, and denoted by $K$. Thus, one input bit influences $M + 1$ output bits.

Since every $k$ input bits generated by the source lead to $n$ output bits to be transmitted over the channel, the code rate of the convolutional code can be defined by $R = k/n$.

Several algorithms exist for decoding convolutional codes. For relatively small values of $k$, the *Viterbi algorithm* is universally used as it provides maximum likelihood performance. Longer constraint lengths produce more powerful codes, but the complexity of the Viterbi algorithm increases exponentially with constraint lengths.

Longer constraint lengths are more practically decoded with any of several *sequential decoding* algorithms, of which the *Fano algorithm* is the best known. Unlike Viterbi decoding, sequential decoding is not maximum likelihood, but its complexity increases only slightly with the constraint length, allowing the use of strong, long-constraint-length codes.

*Turbo codes* are a new class of iterated short convolutional codes that closely approach the theoretical limits imposed by Shannon's theorem with much less decoding complexity than the Viterbi algorithm on the long convolutional codes that would be required for the same performance.

A "turbo" encoder is a combination of two simple encoders. The input is a block of $k$ information bits. The two encoders generate parity symbols from...
simple recursive convolutional codes, each with a small number of states. The information bits are also sent uncoded.

The key innovation of turbo codes is an interleaver, which permutes the original $k$ information bits before they're put into the second encoder. The permutation allows that input sequences for which one encoder produces low-weight codewords will usually cause the other encoder to produce high-weight codewords. Thus, even though the constituent codes are individually weak, the combination is powerful.

Decoding is done by two decoders in an iterative process. The received analog signal is sampled and assigned integers indicating how likely it is that a bit is a '0' or '1'.

Then, each decoder takes the noisy data and respective parity information and computes how confident it is about each decoded bit. The two decoders exchange this confidence information repeatedly, and after a number of iterations, typically four to ten, they begin to agree on all decoded bits.

The decoded data is the sum of the noisy data plus the two final strings of confidence values. The output is converted back to binary digits.

1.4.4 Hybrid ARQ/FEC

Up to here, two main error control procedures are presented: FEC and ARQ. In FEC, redundancy is added at the transmitter and is used at the receiver to correctly recover information, even in the presence of some transmission errors. In ARQ, a smaller percentage is added to the data in comparison to FEC, which makes it possible to only detect errors. A return channel from the receiver to the transmitter is necessary to provide feedback about the status of the received information.

A hybrid-ARQ system consists of an FEC subsystem contained in an ARQ system. The function of the FEC subsystem is to reduce the frequency of retransmission by correcting the error patterns that occur most frequently. The combination of FEC and ARQ makes the system more reliable than a FEC system alone and leads to higher throughput than the system with ARQ only.

Hybrid-ARQ schemes are classified based on the nature of the retransmit attempt and the manner in which multiple retransmit attempts are combined at the receiver. They may be grouped in three main subclasses: Type I, II and III. At the end a comparison between the different hybrid-ARQ schemes is given in Table (1.1).

**Type I** A code used in this scheme must be able to correct a certain collection of error patterns and simultaneously detect other error patterns.

In all retransmissions the same frame is sent. At the receiver side, diversity\footnote{Diversity at the receiver side can be implemented for example by using a RAKE-receiver} or code combining\footnote{Combining the retransmitted packets in order to decode the data packet. Thus, increasing the throughput on the channel} may be used to improve reliability. Therefore, packets that are not correctly received are not immediately discarded but may be used to decode the packet. According to whether a Cyclic Redundancy Code (CRC) for error detection is adopted or not, two configurations are possible:
1.4 Error Control Coding

1. Data + FEC overhead
2. (Data + CRC) + FEC

In the first case, when the number of errors in the receiver block exceeds the code correcting capability, the FEC is used as an error detection code. In the second and more reliable configuration, the decoded sequence consists of information data and error detection code, hence increasing the error detection capability. In both cases, all transmissions are self-decodable. An example is given in Figure 1.15

![Type I Hybrid ARQ - An example of retransmission.](image)

In both cases, more redundant bits are needed, as in the corresponding ARQ scheme, which increases the overhead for each transmission and retransmission. As a result, when the channel error rate is low, the type-I hybrid ARQ has a lower throughput than its corresponding ARQ scheme. With higher channel error rate, the hybrid ARQ scheme provides higher throughput.

Type II In this configuration, only parity bits are sent in some of the retransmissions. The concept is that incremental parity-bits for error correction are sent to the receiver only when they are needed. The transmission of additional parity bits and the subsequent combining to form a low rate code, increases the likelihood of reliable decoding. As retransmit attempts in this scheme consists purely of parity bits, the codewords are not self-decodable, and code combining is required before decoding can be attempted. Most type II algorithms adopt the retransmissions scheme with an incremental number of parity bits.

Rate-Compatible-Convolutional Codes (RCPC) [31] are used in type II Hybrid ARQ schemes. RCPC have the property that all coded bits of any code of the family are used by all lower rate codes. Finally, turbo codes have also
been proposed as candidates for packet combining since they are systematic and produce incremental redundancy by puncturing parity bits [32][33]. An example is depicted in Figure 1.16.

![Type II Hybrid ARQ - An example of retransmission.](image)

**Type III** In such a scheme both data and parity bits are included in every retransmission. Packets that are detected in error are not discarded, but are combined with complementary transmissions provided by the transmitter to help recover the transmitted message.

Each transmit attempt differs from previous transmissions and multiple received codewords can be code combined resulting in lower rate code. The scheme differs from type II due to each transmit attempt being self-decodable. The decoder does not have to rely on previously received sequences for the same data packet for decoding, as is generally the case with incremental redundancy ARQ schemes. An example is given in Figure 1.17.
1.4 Error Control Coding

Figure 1.17: Type III Hybrid ARQ - An example of retransmission.

<table>
<thead>
<tr>
<th>HARQ scheme</th>
<th>Type I</th>
<th>Type II</th>
<th>Type III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Same frame sent in all the retransmissions. Packets detected in error are discarded. User data may be recovered from each single transmission.</td>
<td>Only parity bits sent in the retransmissions. Packets detected in error are not discarded. Decoder has to rely on previously received word for the same data packet.</td>
<td>User data and parity bits are included in each retransmission. Packets detected in error are not discarded. User data may be recovered from each single transmission.</td>
</tr>
<tr>
<td>Coding overhead at the first transmission</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Suitability</td>
<td>No time-varying channels</td>
<td>Time-varying channels</td>
<td>Time-varying channels. Better than type II in bursty channels.</td>
</tr>
</tbody>
</table>

Table 1.1: Comparison of Hybrid ARQ schemes. [34]
Chapter 2

Related Work

In this chapter, a number of articles is discussed, that deepen the different topics this thesis, which were presented in the previous chapter. Next to that, some articles are presented, that contain closely related scientific work to the research work done during this project and that is presented in this thesis.

This chapter is structured as follows:

First, in Section 2.1, the aspect of adaptivity in error control protocols is introduced. Both ARQ and FEC schemes can be made adaptive, in order to make better use of the channel capacity, in the case that error statistics on that channel are time-varying. When protocols are adaptive, they can improve the throughput and efficiency of the communication on the dynamic wireless channel.

Next, in Section 2.2, the applicability of FEC in UWB systems will be considered. Conditions for applying FEC to UWB systems, and the suitability of different FEC schemes will be discussed.

After that, in Section 2.3, different articles are discussed, each presenting a different enhancement to a standard ARQ scheme, which makes it more suitable for specific conditions in a wireless network.

Finally, in Section 2.4, two protocols are presented that are used in ad-hoc networks. One protocol makes multicasting more reliable, the other one is specified for the UWB technique, as a part of the proposed MAC-layer.
2.1 Adaptivity

Adaptivity is an important issue in wireless systems because of the dynamic nature of wireless channels. Such dynamism arises from several sources, mostly from mobility of the transmitter or receiver, furthermore from the fact that the user population of the channel changes, due to the entrance and exit of users and interferers from the channels, and finally due to the bursty nature of many information sources.

Adaptive techniques which have attracted the interest of researchers in the last decade are adaptive modulation and coding (AMC), adaptive antennas and adaptive equalization techniques. An overview of adaptive techniques in wireless communications is given in [34]. This paper focuses on adaptive modulation and adaptive error control mechanisms.

2.1.1 Adaptive- and diversity techniques

When multipath fading and interference are present on the wireless channel, there are basically two ways to fully utilize the channel capacity: adaptation- and diversity techniques.

In the adaptation mode, parameters such as transmission power, symbol rate, constellation size\(^1\), coding rate/scheme or any combination of them are changed in response to time-varying channel conditions.

Conversely, diversity techniques try to take advantage of channel variations or interference levels by resolving several fully or partially de-correlated signals. Time, frequency and space diversity are examples. Channel coding also induces a form of diversity.

When the channel correlation\(^2\) is high, as in small UWB ad-hoc networks, channel adaptive techniques prevail over diversity techniques [34].

2.1.2 Adaptive Error Control

As has been stated in section 1.4, there are two main error control procedures: FEC and ARQ. Adaptive control mechanisms allow the error protection to vary as channel conditions vary, instead of fixing a level of overhead that can handle with worst-case conditions. Therefore, the overhead is always adapted to the current conditions, avoiding both over-optimistic channel coding when conditions are good and resource wasting retransmissions due to insufficient error protection.

Next, adaptive ARQ, adaptive FEC, and adaptive Hybrid ARQ are introduced. As it will be clarified in the rest of this section, hybrid ARQ schemes are always able to adapt the data rate to the channel state, on the average within a data block cycle [35].

Adaptive FEC

Adaptive FEC is performed by adapting code rates to channel conditions. The adaptation of these code rates is done by adopting puncturing: different code

\(^1\)The number of bits per symbol

\(^2\)Channel correlation corresponds inversible to variations of fading coefficients over short time periods.
rates are obtained from a low-rate code by periodic elimination of specific code symbols. The pattern of punctured symbols is called perforation pattern of the punctured code and is commonly described by a matrix called the perforation matrix.

Variable-rate coding coding can be achieved simply by changing the perforation matrix. As a result, variable-rate control provides better performance for low-complexity or delay constrained systems.

Adaptive ARQ

A parameter that measures the efficiency of an ARQ protocol for delivering useful data is the throughput. In [34] are three approaches introduced for adaptive ARQ.

The first approach improves the throughput by choosing a lower block size, when the Bit Error Rate (BER) increases. It estimates the exact BER to monitor the channel state, before changing the packet size. This is done by taking the ratio of n-bits packets in error by the total number of n-bits packets since the last time the packet size was changed. The main weakness is that the observation interval (OBI) over which the BER is computed, becomes arbitrarily long.

Another approach consists of varying the number of contiguous retransmitted copies of the erroneous block. To monitor the channel state, contiguous ACKs or NACKs are counted. This approach reacts more effective to channel state changes than the first approach.

The last approach for adaptive ARQ uses a code-rate adaptive error control. It switches between different ARQ/FEC schemes according to the error state of the channel.

In Table 2.1 the three introduced approaches of adaptive ARQ are compared in terms of reaction capability to channel state variations. This capability is mainly due to the channel state monitor strategy, since an adaptive ARQ scheme can react only when the channel state has been estimated.

<table>
<thead>
<tr>
<th>Adaptive ARQ approach</th>
<th>Channel state monitor strategy</th>
<th>Reaction capability to channel state changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Varying block size</td>
<td>Estimation of the BER</td>
<td>Slow because the OBI algorithm is not a sliding window</td>
</tr>
<tr>
<td>2. Varying the number of contiguous retransmitted copies</td>
<td>Number of contiguous ACKs/NACKs</td>
<td>Higher than in approach 1, since the OBI algorithm is a sliding window</td>
</tr>
<tr>
<td>3. Switching between different ARQ/FEC schemes</td>
<td></td>
<td>Slower than Hybrid ARQ</td>
</tr>
</tbody>
</table>
Adaptive Hybrid ARQ

Hybrid ARQ schemes are particularly suited to adapt to different channel conditions because they use both FEC and ARQ schemes. Actually, hybrid ARQ schemes are already adaptive. They use the ARQ mechanism to inform the sender via acknowledgements of the receiver, and then the reply of the sender is dependent on the hybrid-ARQ type that is applied.

When type II or type III hybrid ARQ schemes are applied, higher reliability of the decoding process is achieved after each retransmission, since erroneously received words are not discarded but they are stored at the receiver and combined for decoding.

In both the hybrid ARQ schemes, type II and type III, the overhead due to the FEC and retransmissions is not fixed, but depends on the channel conditions. In good channel conditions, a minimum level of redundancy is added. Specifically, in type II schemes the redundancy is only due to the error detection coding (like in pure ARQ), while in type III schemes it is due to the error detection plus an error correcting coding that matches the channel noise requirements.

In case of bad channel conditions, more powerful codes are used in the decoding process. Therefore, for time-varying channels, these two schemes are better than type I hybrid ARQ in which the overhead is fixed. In the latter case, the total redundancy is higher than in a pure ARQ scheme, thus resulting in a lower throughput when the BER is low.

To improve the adaptivity of hybrid ARQ schemes to varying channel conditions, the combining of transmitted and retransmitted packets can be applied. This is known as code combining. The basic technique here, is to use the energy of the previously transmitted signal with the new retransmitted signal to decode the block. There are two main schemes for hybrid ARQ, Chase combining and Incremental Redundancy.

Chase Combining  Chase combining, also called Hybrid ARQ type III with one redundancy version, involves the retransmission of the same data packet which was received with errors. Once the retransmission is received, the receiver combines the soft values of the original signal and the retransmitted signal weighted by the SNR prior to decoding the data packet.

The advantages of Chase combining are that each transmission and retransmission can be coded individually (self-decodable), and it introduces a time-diversity gain. A disadvantage is that in the retransmission, the entire packet is sent again, which is a wastage of bandwidth.

Incremental Redundancy  Incremental Redundancy (IR)\(^3\), a Hybrid ARQ type II scheme, is used to get the maximum performance out of the available bandwidth. Here, the retransmitted block consists of only the correction data to the original data that carries no actual information (redundancy). The additional redundant information is sent incrementally when retransmissions are received with errors.

\(^3\)IR will be the abbreviation for Incremental Redundancy, as UWB-IR is the abbreviation for UWB Impulse Radio
The advantage of IR is that the effective data throughput/bandwidth of a user is reduced. A disadvantage is that the systematic bits are sent only in the first transmission and not with the retransmission which makes the retransmission non-self-decodable. So, if the first transmission is lost due to large fading effects there is no chance of recovering from this situation.

**Partial Incremental Redundancy** The partial IR is the combination of Chase combining and IR. The disadvantage with IR is removed by adding the systematic bits along with the incremental redundant bits (different puncturing bits) in the retransmissions. This makes both original and retransmitted signals self-decodable (Hybrid ARQ type III).

**Performance of Hybrid ARQ schemes for the fading channel** In [35], a theoretical method is defined for evaluating and comparing the achievable performance of different hybrid ARQ schemes for the fading channel. Hybrid ARQ type II IR algorithms are compared to Hybrid ARQ type III schemes, in which user data and redundant data are included in every transmission.

The results show that the incremental techniques (type II IR algorithms) offer a more significant performance improvement over pure FEC techniques, than the type III code combining algorithms.

In [36], the performance of turbo hybrid ARQ schemes, type I, type II and type III, are examined for wideband communications based on orthogonal frequency division multiplexing (OFDM).

In the results, the throughput is measured for an increasing SNR. It shows that in a non-coherent channel, the throughput performance of a Hybrid ARQ type II scheme exceeds the other schemes.


2.2 FEC in UWB systems

In section 1.4.3 the concept of error correcting codes was introduced. In this section, the applicability of FEC in systems using the UWB technology will be considered. In [37], a literary survey is done on this subject. The most important findings are presented here.

2.2.1 Conditions for applying FEC

First, when applying FEC to UWB systems, the boundary conditions should be considered before choosing a suitable FEC scheme.

- One of the most important and distinctive properties of the UWB technology is the very large bandwidth. This raises the opportunity to apply channel codes characterized by low and very low code rates.

- The desired bit error (BER) performance should be determined before applying a FEC scheme. Typically, it is characterized by a range of BER that is acceptable for a particular type of service.

- FEC introduces complexity to the system, particularly for the implementation of the decoder. Such complexity results in an additional delay in a high data-rate UWB system, an appropriate selection of the FEC scheme might be critical. The decoder must have enough time to perform the decoding operations. Additionally, the decoder startup delay seems to be an important factor in any UWB system. Regarding the above, the application of the channel codes of rather low or moderate level of complexity has been proposed [9].

- Another important condition is synchronization. The information how the receiver gains synchronization at the beginning of a transmission or at the beginning of a particular data block may play an important role.

2.2.2 Coding-modulation scheme

A standard coding-modulation scheme that can be applied in UWB as well is as follows:

The coding is done on information bits, which are encoded into code symbols. These code symbols are then modulated and matched into the frames. Finally, the frames are transmitted as pulses. This is depicted in figure 2.1. Actual characteristics of the pulses within the frame rely on the modulation. On the receiver side, the inverse process takes place.

2.2.3 Proposed error correction techniques

Several FEC schemes have been proposed for UWB systems. In [38], [39], [37], the emphasis is on three different error correcting techniques: repetition block coding, convolutional coding, and turbo coding. At the end, some other coding schemes are presented.

---

4Or group of bits in case of non-binary codes
5In UWB systems, the frame denotes one pulse, or time interval during which the pulse may occur. This in contrary to the common definition
Figure 2.1: Diagram showing the coding-modulation scheme in a UWB-IR system.

Repetition block coding

Repetition codes represent the simplest form of error correcting codes. Generally, in such codes, every bit is repeated a number of times consecutively. In the UWB technique, there are two types of repetition codes, frame repetition and bit repetition.

In the first one, frame repetition, no coding scheme is applied, i.e., \( k = n = 1 \) and \( N_f > 1 \). This results in transmission of a number of pulses per each information bit. In the second one, bit repetition, \( k = 1, n = N_b > 1, d_1 = d_2 = \ldots = d_n, \) and \( N_f = 1 \), where \( d_n \) is a code symbol.

Repetition block codes are very simple error correction codes to implement, but are not very energy efficient. For every data bit \( N_s \) pulses have to be transmitted, that costs \( N_s \) times the power as it was for one information bit. More energy efficient schemes are convolutional codes and turbo codes.

Convolutional coding

One class of convolutional codes that is very appropriate for UWB systems are the super-orthogonal codes. This is because of the unique properties, such as low code rate, and near-optimal performance.

When super-orthogonal convolutional coding (SOC) is applied, a data information bit is encoded by the SOC encoder with a code rate of \( R = 1/n \), where \( n = 2^{K-2} \) and \( K \) is the constraint length.
Results of simulations show that the super-orthogonal convolutional scheme significantly outperforms the uncoded scheme. For instance, at data rate 5 Mb/s, for 30 users, the bit error probability for the synchronous uncoded scheme equals circa $10^{-2}$, whereas for the coded scheme it is about $10^{-4}$. At the same data rate 5 Mb/s and 30 active users, the BER of the asynchronous uncoded scheme is circa $10^{-4}$, whereas of the coded scheme it is less than $10^{-10}$. By applying SOC, the number of users can increase by a factor of $(4 + \log_2 N_x)/2$ at a given BER.

**Turbo coding**

As seen in section 1.4.3 in the paragraph on turbo coding, turbo codes can perform close to the theoretical limit set by the Shannon Law. They perform well for very low SNRs, but their performance, of turbo codes found so far, decreases for moderate to high SNRs.

For UWB systems it has been proposed to apply SOC codes for error correction coding. The resulting code is referred to as the turbo-like code (TLC) or the super-orthogonal turbo code.

**Other coding schemes**

Other FEC schemes that can are proposed for UWB systems [37] are punctured convolutional codes, LDPC codes and concatenated codes (RS-codes + convolutional codes). These FEC schemes can be applied interchangeably in the UWB system on the “bit level”.

### 2.2.4 Interleaved coding schemes

In [39] *interleaved coding-modulation schemes* are introduced. This extends the general coding-modulation scheme that was introduced in [38] and discussed in 2.2.3 with application of an interleaver, as depicted in figure 2.2.

When using interleaving, diversity is introduced into the transmitted signal, improving its spectral characteristics and the BER performance. Two typical types of interleavers that can be applied are *block* and *random interleavers*.

In the block interleaver, the input sequence is formatted in a matrix of $a$ rows and $b$ columns. During the interleaving process, the input sequence is written row-wise into the matrix and read in column-by-column fashion. The de-interleaving process uses the inverse procedure. In the random interleaver, a block of $L$ input symbols is written into the interleaver and is read randomly.

As can be seen in Figure 2.2, the interleaver is applied on different levels in the coding-modulation scheme: bit, frame, and chip level. This application of interleaving on different levels in the coding-modulation scheme is evaluated with three different coding schemes: SOC, frame, and bit repetition. It proves, that chip interleaving introduces the best BER performance among all evaluated scenarios.

### 2.2.5 An adaptive interleaved coding scheme

Interleaving is used to convert bursty fading into independent fading, and puncturing is used to achieve variable rate encoding. However, for variable rate
adaptive coding with symbol-by-symbol adaptation, dynamic puncturing is a problem. This is because puncturing modes\(^6\) have different puncturing periods that usually span several symbol durations. If there are a number of mode changes within a period, rate compatibility or incremental puncturing is difficult to maintain. Furthermore, when interleaving is considered as well, the puncturing and interleaving tasks are not trivial.

In [40], a solution is presented to this problems by introducing multilevel puncturing and interleaving for high speed mobile wireless channels. The coded bits that are puncturable are labeled and the puncturing action is deferred till after the interleaving. Instead of using a single block interleaver, a number of parallel interleavers of different levels is used. The level-0 interleaver contains coded bits that are non-puncturable. The level-1 interleaver contains coded bits according to puncturing pattern-1. Since incremental puncturing for increasing puncturing levels is assumed, the level-2 interleaver contains extra puncturable bits that are not contained in the level-1 interleaver. Both interleavers are independent of each other.

Bits from the level-0 interleaver must be transmitted by modulation symbols while bits from the other levels of interleavers are puncturable and, therefore, could be transmitted or not transmitted (punctured).

With this layered approach, puncturing can be done after interleaving, which makes the symbol-by-symbol adaptivity possible. The current channel state is obtained for every symbol duration \(T_s\) at the receiver and returned to the transmitter via a low capacity feedback link. Based on the channel state, the appropriate channel number of coded bits and puncturable bits are taken from the appropriate levels of interleavers.

\(^{6}\) Each mode has a different level of puncturing
2.3 Enhanced ARQ schemes

This section introduces three enhanced ARQ schemes. The first part presents a go-back-N ARQ scheme with selective repeat in intra-block. The second parts deals with an ARQ protocol that is used in 3GPP specifications, enhanced with a control mechanism that takes QoS (Quality of Service) provisioning into account. Finally, the last part discusses the $N$-channel stop-and-wait protocol, that is used in HSDPA\(^7\) (high speed downlink packet access).

2.3.1 Go-back-N ARQ scheme with selective repeat in intra-block

In [41], two versions of a go-back-N scheme with selective repeat in intra-block (GBN-SR and a revised version, Rev-GBN-SR) are presented. In both versions, the receiver returns the acknowledge signal blockwise and selective repeat is performed within the retransmission block.

In the first version, GBN-SR, the sender continuously transmits an initial block consisted of $B$ new packets, as depicted in figure 2.3.

![Figure 2.3: Transmission pattern of packets and blocks in the GBN-SR scheme](image)

The receiver stores one block in the receiver buffer, and performs error checking of packets of the received block. If no errors are detected in the received block, the receiver passes the block to the network-layer and stores the next block in the receiver buffer.

When errors are detected in the received block, the receiver discards the succeeding blocks and requests retransmission of the packets in error by using a NACK message that indicates the sequence numbers of the erroneous packets. When receiving the NACK message, the sender retransmits only the requested packets in the succeeding blocks. The size of a retransmitted block reduces its value as retransmissions are repeated.

In the second version, multiple copy retransmission is used to improve the performance at high packet error regions. The proposed scheme is called Rev-GBN-SR and is depicted in figure 2.4.

\(^7\)HSDPA is a packet-based service in W-CDMA with a data transmission speed up to 8-10 Mbps over a 5 MHz bandwidth.
At the sender side, the original packets themselves are transmitted at the first transmission of a block, and multiple copies of a requested packet are retransmitted in succeeding transmissions. At the receiver side, the retransmission request of a packet is issued only when all copies of a packet have an error.

The GBN-SR scheme is the same as the Rev-GBN-SR, when only one packet is sent during retransmission, instead of multiple copies.

A significant improvement in throughput performance, compared with standard go-back-N (GBN), is achieved with the GBN-SR scheme over a wider range of error rates from low to moderate, except high error rates. The throughput can be further improved by increasing the block size of the initial transmission at the expense of the receiver buffer size. The Rev-GBN-SR scheme with two copies, significantly improves throughput performance, also compared to standard GBN, especially at high error conditions.

### 2.3.2 An ARQ scheme with QoS provisioning

In [42], the 3GPP specification of the RLC (radio link control) protocol is assumed [43], which consists of a mobile station and an access point, communicating with each other through the radio channel. They can be both transmitter or receiver entities.

The acknowledged transfer mode is considered, which implements an ARQ scheme with selective repeat. The sender can check on the receiver's status by polling the receiver. The receiver must reply to the sender's poll, otherwise the transmitter keeps polling the receiver until it receives a reply.

The 3GPP specification is enhanced with a control mechanism of the transmitter activity to let the system save radio resources as well as energy in the case of bad channel conditions. Two different modes of operation are possible: greedy or saving mode.

In the case of greedy mode, the sender retransmits the missing data units at once and then goes on with the information transfer. In the case of saving mode, the sender stops transmitting data units and starts polling the receiver.

---

The process of repeatedly checking the status of a resource to determine when state changes occur.
2.3 Enhanced ARQ schemes

To probe the channel status, the transmitter starts retransmitting all missing data units only when the receiver replies to the poll.

At the cost of higher energy consumption, greedy mode is more reactive than saving mode to changes in the channel conditions, and therefore is able to provide better QoS. In order to dynamically trade-off between the needs of providing QoS and saving energy, a control mechanism is introduced on the transmitter: if the number of data units in the transmission buffer is less than a given threshold $T_h$, the sender operates in saving mode, otherwise it enters greedy mode.

Results derived for loss-sensitive traffic (data file transfer and IP telephony) showed that, by changing the buffer threshold, a trade-off between loss probability (QoS performance) and energy consumption can be found.

2.3.3 A $N$-channel stop-and-wait ARQ scheme

In [44], ARQ with selective-repeat (ARQ-SR) and stop-and-wait (SAW-ARQ) ARQ, both most often used in mobile communication, are compared as ARQ schemes for hybrid ARQ in HSDPA.

Although ARQ-SR is generally insensitive to delay and has the favorable property of repeating only those blocks that have been received in error, there are some difficulties seen when it is partnered with hybrid ARQ.

- The sender's memory requirements are high, because samples of each transmission of a block must be stored.

- Hybrid ARQ requires the receiver to reliably determine the sequence number of each transmission, so a strong code must be used to encode the sequence information, effectively multiplying the bandwidth required for signaling.

Hybrid ARQ using a SAW mechanism offers significant improvements by reducing the overall bandwidth required for signaling and the sender's memory. There remains one drawback, the acknowledgements are not instantaneous and therefore after every transmission, the transmitter must wait to receive the acknowledgement prior to transmitting the next block.

$N$-channel SAW hybrid ARQ offers a solution by parallelizing the SAW protocol and in effect running a separate instantiation of the hybrid ARQ protocol when the channel is idle. As a result, no system capacity goes wasted since one instance of the algorithm communicates a data block on the forward link at the same time that the other communicates an acknowledgement on the reverse link. However, the receiver has to store $N$ blocks for this scheme.

There are two different methods for $N$-channel hybrid ARQ:

- Signal the subchannel number explicitly (fully asynchronous)

- Tie the subchannel number to e.g. frame timing (partially asynchronous)
An example of the first method is depicted in Figure 2.5. In this example $N = 4$ and a 3 slot Transmission Time Interval (TTI)$^9$. Each packet is acknowledged during the transmission of other packets so that the channel can be kept occupied all the time when there are packets to transmit.

![Figure 2.5: Principle of N-channel stop-and-wait hybrid-ARQ (N=4)](image)

$N$-channel hybrid ARQ supports asynchronous transmission: different users can be scheduled freely without waiting completion of a given transmission. The transmission for a given user is assumed to continue when the channel is again allocated.

The second method would restrict the positions of retransmissions such that the retransmission can happen at positions $m + k \times N$, where $m$ is the position (TTI) of the first transmission of a given package and $k = 1, 2, ...$. The retransmissions can still be delayed if the channel is allocated to other users in between.

The key property of the $N$-channel SAW protocol is that the received timing of an acknowledgement identifies the acknowledged data block. This is the reason why $N$-channel SAW allows for multiple not-yet-acknowledged data blocks without explicit sequence numbers. As a consequence there are strict requirements on the transmission timing of the acknowledgements (relative to the receive timing of the data block).

Increasing the number ($N$) of parallel hybrid ARQ processes will improve the time diversity per process and could result in longer available processing times both at both transmitter and receiver. Drawbacks are that increased buffering is required and there is a longer delay per process. HSDPA uses a maximum of eight parallel channels ($N = 8$).

---

$^9$TTI is the inter-arrival time of Transport Block Sets, and is equal to the periodicity at which a Transport Block Set is transferred by the physical layer on the radio interface.
2.4 Error coding in mobile ad-hoc networks

In this section, two error coding protocols are introduced that are used in ad-hoc networks. In 2.4.1, an error coding protocol is discussed that is used for multicast and in 2.4.2, an error coding protocol is presented that is used for unicast, which is based on the UWB technology.

2.4.1 A reliable multicast protocol

In [45], a survey of several reliable multicast protocols is presented and advantages and disadvantages of different design features as well as protocol performance are compared.

Multicast protocols play a significant role in many applications of mobile ad-hoc networks. To make them more reliable, many protocols have been proposed, and they can be classified into three categories: ARQ-based, gossip-based and FEC-based.

In ARQ-based reliable multicast protocols, lost packets are retransmitted by the sources until they are recovered at all the receivers. In gossip-based protocols, multicast packets are repeatedly transmitted for a few times by a few of the multicast members in a peer-to-peer fashion. FEC-based protocols use redundant data in each packet before transmitting, so a few packet losses are tolerated, and the original data can be reconstructed using correctly received ones.

Hybrid ARQ, using the FEC technique, in combination with the ARQ mechanism, is used in the Reliable Multicast Data Distribution Protocol (RMDP). In RMDP, the sender splits a file with a large sequence of data packets into slices of $k$ packets. The sender encodes $k$ source packets of each slide into $n$ data packets with $n \gg k$ based on a Vandermonde code [46]. Therefore $(n - k)/n$ percent redundant data is transmitted. For each slice, a receiver counts the number of packets it receives. After $k$ different received packets, it decodes for the original source data. In case of losses, a receiver sends the request to the source in scheduled intervals asking for the number of packets that it needs for reconstruction.

The protocol simplifies the recovery handling by using only the number of packets needed rather than specific packet IDs. However, long packet latency is incurred, because a receiver has to wait for the reception of $k$ packets before it can decode and deliver them to the network-layer. So, the main drawback of RMDP is the computation of the erasure codes in software, which implies in a large overhead in protocol execution. Furthermore, it suffers from the problem of ARQ explosion due to burst packet loss in wireless environments.

RMDP is suitable for networks where high overhead in sending redundant data can be tolerated. With moderate $n$, overhead generated due to redundancy can be less than that in ARQ feedbacks and retransmissions.

2.4.2 A MAC protocol for UWB ad-hoc networks

In [47] and [48], a rate-adaptive MAC protocol is presented that is based on dynamic channel coding for low power UWB ad-hoc networks. A transmitter

\[^{10}\text{Communication between a single sender and multiple receivers on a network} \]
picks a channel code according to the protocol described in the next section 2.4.2, receives feedback from the receiver and, if needed, sends incremental redundancy for the destination to be able to decode.

**Dynamic channel coding**

The variable encoding is achieved by puncturing, and rate compatibility is added, which means that lower rate codes use the same coded bits as the higher rate codes plus one or more additional bit(s). Hence these codes permit the use of incremental redundancy (IR) using a typical hybrid ARQ protocol.

**ARQ procedure** To communicate with a receiver $R$, a transmitter $T$ has to perform the following steps:

- $T$ adds a CRC to the packet and encodes it with the lowest rate $R_N$,
- $T$ punctures the encoded data to obtain the desired code rate $R_i$ and sends the packet. The punctured bits are stored in the case the decoding at $R$ fails.
- Upon packet reception, $R$ decodes the data and checks the CRC. If the decoding is successful, an acknowledgement (ACK) is sent back to $T$. Otherwise, a negative acknowledgement (NACK) is sent.
- As long as $S$ receives NACKs, further packets of punctured bits (IR data) are sent, until transmission succeeds or no more punctured bits are available. In the latter case, $T$ may attempt another transmission at a later time.

When the receiver can't even detect reception of data it can't send a NACK. In this case, the sender will time out and retry communication with a more powerful code.

For good performance and a short transmission delay, sending redundant information should rarely be necessary. Hence, it is more important that the transmission succeeds directly without having to send additional punctured bits than using the highest code possible.

**Rate selection** When nodes communicate for the first time, it is necessary to bootstrap the code adaptation mechanism. The first data packet is encoded with the most powerful (lowest rate) code $R_N$. If the receiver can decode the data packet successfully, it will estimate which higher rate $R_j, j \leq N$ would still allow to decode the data.

Decoding of the data packet with rate $R_N$ is performed by step-wise traversal of the trellis of the Viterbi-decoder [27]. At each step a trellis branch is chosen, where a branch corresponds to a specific decoded bit. The packet is then reproduced from the bits corresponding to the sequence of selected branches. Hence, as soon as the outcome of a decoding step for a higher code rate $R_i > R_N$ differs, code $R_i$ can be eliminated. Because of the rate compatibility functions of RCPC codes, this allows to also eliminate all codes $R_i'$, with $R_i' > R_i$. The highest rate code that remains is still powerful enough to decode the packet.
The transmitter determines the code to use as follows: in case the code indicated by the receiver is higher than the current code, the sender does not directly switch to the new channel code but decreases its code index by one. Otherwise, if a more powerful code is necessary, the sender switches directly to this code.

Transmitters maintain a cache of channel codes for a number of receivers. If a transmitter does not communicate with a receiver for a certain amount of time, the corresponding cache entry times out and the sender bootstraps the code selection procedure with the code $R_N$ as described above. The algorithm is depicted in Figure 2.6

Figure 2.6: Dynamic channel coding
Chapter 3

The selection of the hybrid-ARQ scheme

This chapter deals with the choice for an adaptive hybrid-ARQ protocol, which is appropriate for UWB ad-hoc networks. Based on the related work presented in chapter 2, several protocols are considered. The choice for the hybrid-ARQ protocol is founded by the boundary conditions set by the nature of the ad-hoc network and the UWB technology. Finally, some improvements to the proposed hybrid-ARQ are presented.

This chapter is structured as follows:

First, in Section 3.1, the boundary conditions are stated on which the hybrid-ARQ protocol has to comply. These boundary conditions come from the nature of the UWB technology and the ad-hoc network.

Next, in Section 3.2, the FEC scheme is selected from several different schemes. Repetition coding, SOC coding, and rate-compatible coding are considered for error correction in the hybrid-ARQ protocol.

After that, in Section 3.3, the ARQ scheme for the hybrid-ARQ protocol is selected from the three conventional ARQ schemes: stop-and-wait, go-back-N, and selective repeat. A comparison is made on the basis of efficiency. Next, the hybrid-ARQ type is selected by comparing the three known types, type I, II and III, for application in the hybrid-ARQ scheme for UWB ad-hoc networks.

Finally, in Section 3.4, some improvements are presented that are applicable to the selected hybrid-ARQ scheme: interleaving, and QoS provisioning. These ideas were presented in the previous chapter 2.
3.1 Introduction

In this section, the boundary conditions that exist with regard to the choice for the hybrid-ARQ protocol are briefly discussed.

The basis of a hybrid-ARQ protocol is the FEC protocol. This protocol encodes the information bits and additional bits (the error detection bits, and the header bits) before they are transmitted over the channel. At the receiver side, the transmitted bits are decoded and checked for existence of a detectable error.

When errors are found, the ARQ protocol takes care of the request for a retransmission. Dependant on which type of hybrid-ARQ is used, the nature of the retransmitted bits (information bits, redundant FEC bits, or both) differs, and thus the amount of retransmitted bits. The choice for a hybrid-ARQ type is linked with the choice for a FEC scheme.

The adaptivity of the hybrid-ARQ protocol is implemented by using different code rates dependent on the channel quality. The FEC scheme should therefore be capable of encoding with different code rates.

The complexity of the encoding and decoding enforces restrictions on the FEC scheme. The same applies to the choice for an ARQ scheme with regard to buffers. Processing time is energy consuming and increases the round-trip delay.

The nature of the ad-hoc network induces a large effect on the choice for the hybrid-ARQ scheme. Delay\(^1\), jitter, stability of the path, packet error rate and congestion are all variable parameters in this type of network.

Finally, the UWB radio technique broadens the spectrum with regard to the narrowband systems such as WiFi, GSM, UMTS and HSDPA, and this brings new possibilities for the amounts of redundancy used in FEC schemes. Interference from other devices, multipath propagation, mobility of nodes, and the low power of the UWB signals must be considered in the choice for the hybrid-ARQ scheme.

Conclusion

The hybrid-ARQ scheme for the UWB ad-hoc network has to fulfil several boundary conditions enforced by the UWB properties and the nature of the ad-hoc network.

A strong FEC code is required for the ad-hoc networks, that has a high BER, because of interference, delay, jitter, instability of the path, and congestion. In the wireless network, errors have a bursty nature. Because of the small devices in the network, the protocol should be simple, thus low complexity is required. Further, the protocol should be fast, and thus only few retransmissions are allowed, and finally, it should be adaptive to channel variations, so the amount of redundancy is required to be adjustable to achieve the highest throughput.

\(^1\)Delay is proportional to the number of hops in the ad-hoc network
3.2 The applied FEC scheme

First, the FEC scheme for the hybrid-ARQ scheme will be considered, by presenting several possible schemes. When a suitable FEC scheme is used, few retransmissions are required, which is an important boundary condition for the hybrid-ARQ scheme. In this case, no extra delay is incurred by the ARQ mechanism, which makes it possible to achieve high data rates.

On the other side, the redundancy added by the FEC scheme should not be large, because bandwidth capacity would then be wasted. A tradeoff should be found here.

The FEC schemes that will be considered are repetition coding in section 3.2.1, super-orthogonal convolutional coding in section 3.2.2, and rate-compatible coding in section 3.2.3.

3.2.1 Repetition coding

Repetition is considered to be a simple error coding technique, and can be easily implemented in communication systems. Next, three applications of repetition as a channel coding technique will be considered for the hybrid-ARQ scheme.

Simple repetition coding

Repetition used for error correction is a very simple channel code. When used for UWB, bit or frame repetition can be applied, resulting in transmission of a number of pulses per each information bit.

A repetition code can be obtained, by duplicating (single or multiple duplications), either, when bit repetition is applied, a code bit corresponding to one information bit, or, when frame repetition is applied, a frame corresponding to one modulated code bit.

At the receiver, the received duplicated bits or frames are passed through a threshold detector, yielding the value of the duplicated information bit or frame, respectively.

Repetition coding results in very easy encoding and decoding design. Low rate codes can easily be obtained, but, in general, repetition coding has a low performance gain compared to other coding schemes. Low performance gain represents low power and bandwidth efficiency.

The variable nature of the ad-hoc network requires a high coding gain, a low BER at a low SNR, and thus, the simple repetition scheme is not applicable in the hybrid-ARQ scheme.

Finally, repetition can be applied when already another FEC scheme is used for error coding. Encoded information bits are then repeated to slightly improve the performance gain of the concerned FEC scheme and decrease the code rate of the scheme. In the next two subsections, two enhanced repetition codes are considered for application in the hybrid-ARQ scheme.

Repetition convolutional coding

Convolutional codes have lower BER when compared to block codes with the same SNR. Turbo codes perform even better than convolutional codes, but
3.2 The applied FEC scheme

because of their complex iterative decoding nature, take more time for decoding and consume much more processing power than convolutional- or block codes.

Therefore, convolutional codes are considered to be best applicable in the hybrid-ARQ scheme. Long constraint lengths \( K \) produce stronger codes, but while increasing \( K \), the decoding complexity of the Viterbi algorithm increases exponentially and thus the round trip delay as a result of increased processing time. This should be carefully considered, when choosing \( K \).

To improve the general convolutional codes, repetition can be applied. The duplicating of bits is done after the encoder. There already exist an original rate \( 1/n \) code and constraint length \( K \), called the parent code. This code is completely specified by the generator polynomials \( G^j(D) \)

\[
G^j(D) = g_0^j + g_1^j D + \ldots + g_{K-1}^j D^{K-1},
\]

where

\[
g_j^i \in \{0, 1\}, \quad j = 1, 2, \ldots, n
\]

The code can be seen as a code of rate \( b/nb \) if the information bits to be encoded are considered \( b \) bits at a time, to which correspond \( nb \) coded bits. The repetition code can be obtained by duplicating (single or multiple duplications) a few bits from every \( nb \) code bits corresponding to the encoding of \( b \) information bits by the original rate \( 1/n \) encoder.

The resulting repetition code can be represented by a matrix \( Q \), called a repetition matrix. A repetition matrix has \( n \) rows and \( b \) columns, where each row corresponds to one of the \( n \) encoded bits at the output of the original rate \( 1/n \) encoder, and each column is associated with one encoding cycle. The total number of columns is called the repetition period.

The output of the generator \( G^j(D) \) is compared to the appropriate element in the repetition matrix \( Q \). The elements of \( Q \) represent the number of duplications of each of the \( nb \) coded bits. When the element in \( Q \) is 1, the output from each generator \( G^j(D) \) is transmitted, and duplicated with the value of the element otherwise. The sum of the \( nb \) elements of \( Q \) is the number of coded bits per \( b \) information bits.

As an example, consider the code of rate \( 7/14 \). From every 14 code bits, the first and fifth code bit are duplicated. The code obtained is then of rate \( 7/16 \), and can be represented by a repetition matrix \( Q_1 \) given by

\[
Q_1 = \begin{pmatrix}
2 & 1 & 2 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}
\]

Rate compatible repetition convolutional coding

Rate-compatible repetition convolutional (RCRC) codes are a family of codes in which the lower rate codes are constructed by using all the symbols of the high rate codes plus some extra redundancy symbols. This allows transmission of incremental redundancy in hybrid-ARQ schemes and continuous rate variation, changing from low to high error protection within a data frame.

Two repetition codes obtained from the same parent code are considered to be rate-compatible if the lower rate code uses all the coded bits of the higher
rate code, plus one or more duplicated bits. This means that all the elements in the repetition matrix of the lower code rate must be larger than or equal to the corresponding elements in the repetition matrix of the higher code rate.

For example, for a rate 2/5 repetition convolutional code of $K = 7$ and repetition period $p = 2$, the optimum repetition matrix $Q_2$ is given by

$$Q_2 = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

and the matrix for a RCRC code of rate 2/6 can be any of the matrices

$$\begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 2 & 1 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \text{ or } \begin{pmatrix} 3 & 1 \\ 1 & 1 \end{pmatrix}.$$  

The criterion for the optimum RCRC codes is the optimum distance spectrum (ODS) criterion [49]. ODS convolutional codes give superior distance spectrum compared to all other codes of the same rate and constraint length, where the term superior distance spectrum is defined in [49].

An ODS code has the same free distance as a feed-forward maximum free distance (MFD, i.e., fulfills the Heller bound [50]) code for the rate and memory, but equal or lower information error weights for the most significant error events. A code designed according to the MFD criterion is optimum for one channel with a given $E_b/N_0$ but not necessarily good for other channels and/or other values on $E_b/N_0$. ODS codes on the other hand are channel independent and will result in low error probability for most channels and most $E_b/N_0$.

One drawback with RCRC codes is that they are in general not MFD codes. Very low rate MFD codes are very difficult to find by a full search since the number of search parameters is very large.

### 3.2.2 SOC coding

Super-orthogonal codes outperform [51] the best higher-rate (1/2 and 1/3) convolutional codes at low $E_b/I_0$ (SIR)$^2$ values, but by a decreasingly smaller margin for increasingly higher values. But the super-orthogonal coding gain is always within 0.6 dB of the maximum achievable.

Super-orthogonal codes produce very low codes rates ($R = 1/2^{K-2}$) for even moderate values of the constraint length $K$. By changing the shift register of length $K$ of the encoder, the code rate can be made variable. With this, SOC coding can be applied in an adaptive hybrid-ARQ scheme.

The processing complexity of the decoder grows only linearly with $K$, which is a big advantage of SOC coding when compared to normal convolutional codes, where the decoding complexity grows exponentially. Thus, strong, low rate codes can be easily obtained and can be decoded relatively fast.

Because of the encoder design, the code rates are fixed and grow inversely with the power of 2. These steps are quite large and can be bandwidth inefficient, when large data rates are utilized. Unfortunately, there don't exist repetition or puncturing matrices for SOC codes to produce intermediate code rates and preserve the superorthogonal nature of the codes.

---

$^2$In spread spectrum multiple access systems, the density will be dominated by interference, and is noted by $I_D$. $N_0$ is considered to be negligible.
3.2 The applied FEC scheme

There neither exist puncturing tables for SOC codes, nor code combining is possible. Thus SOC codes can only be applied in a Type I hybrid ARQ scheme.

Finally, the free distance of SOC codes is lower than the free distance of RCRC codes presented before, and the nested codes, presented in the next section 3.2.3. Both codes satisfy the optimum distance spectrum criterion, which results in better free distance codes.

3.2.3 Rate-compatible coding

In section 3.2.1 on repetition coding, the rate-compatible repetition convolutional (RCRC) codes were already introduced. RCRC codes produce low rate codes, as can be done as well with nested convolutional codes. These codes are rate-compatible too, and will be considered in the next paragraph.

Repetition is used to obtain lower rate codes by duplicating code bits, but when redundant bits are punctured, higher code rates can be obtained. When puncturing is done according rate compatibility criterion, these codes are called rate-compatible punctured convolutional (RCPC) codes and are considered in the second paragraph.

Nested Convolutional Coding

A rate $1/n$ convolutional code is completely specified by the encoder generator polynomials $G_i(D)$ as described in Equations (3.1) and (3.2).

Nested convolutional codes are obtained by extending a code of rate $1/n$ to a rate $1/(n+1)$ code by searching for the best additional generator polynomial $G^{n+1}(D)$. When the polynomial is the same as one already existing, than it is a repetition code, and thus nesting is a generalization of repetition. This type of code family is rate-compatible and the advantage is the modular code design that reduces the complexity of the search for low-rate codes.

The Optimum Distance Spectrum (ODS) criterion is used to search for the best encoder. This encoder must satisfy the rate compatibility criterion and must give superior distance spectrum [49]. The nested codes themselves are not necessarily ODS or MFD since a full search for the given rate is not done. However, all the found codes based on the ODS parent code with rate 1/4 turn out to be MFD codes, and are presented in [52]. There exist nested convolutional codes with rates $1/512 - 1/4$ obtained from parents codes of rates 1/2, 1/3, and 1/4 and constraint lengths between 3 and 15.

Because the nested codes are rate-compatible, incremental redundancy can be used when retransmissions are required. After the received data is acknowledged, Viterbi decoding can be applied.

Rate-compatible punctured convolutional coding

The puncturing of the bits is done after information bits have been encoded. This encoder produces a rate $1/n$ code, called the parent code. In general, a high rate, $b/V$ punctured convolutional code with constraint length $K$, with $b/V > k/n$, can be obtained from a rate $1/n$ code with constraint length $K$, by deleting $(nb - V)$ bits from every $nb$ code bits corresponding to the encoding

---

3The optimum distance spectrum criterion [49] will be used for selecting the best code.
of $b$ information bits by the original rate $1/n$ code, according to a well selected perforation pattern. The resulting rate is the $R = b/(nb - (nb - V))$, which is equal to the desired rate $R = b/V$.

The operation of deleting code bits is generally represented by a matrix $P$, called a perforation matrix. In case of rate $b/V$ punctured convolutional codes, obtained from rate $1/n$, the perforation matrix $P$ has $b$ columns and $n$ rows. Each row corresponds to one of the $n$ encoded bits at the output of the rate $k/n$ encoder. Each column is associated with one encoding cycle. The total number of columns is called the puncturing period.

The elements of $P$ are only zeros and ones, corresponding to deleting or keeping the corresponding code bit at the output of the original $1/n$ encoder. The total number of ones of the perforation matrix is the number of remaining bits per $b$ information bits.

The perforation matrix is selected so as to yield the best target non-catastrophic\footnote{A convolutional encoder is catastrophic if and only if its state diagram has a zero-output-weight cycle other than the self-loop around the all-zero state} high-rate punctured convolutional code. For example, the perforation matrix $P_1$ of a rate $7/8$ punctured convolutional code\footnote{The optimum distance spectrum criterion [49] will be used for selecting the best code.} with constraint length $K = 7$, obtained from the best rate $1/2$ code with constraint length $K = 7$ and puncturing period $p = 7$ is given by

$$P_1 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (3.3)$$

Two convolutional codes, obtained from the same original code, are said to be rate-compatible if the perforation matrix of the higher code is contained in the perforation matrix of the lower rate code. This means that all code bits in the higher rate code are used in the lower rate code. Given a starting high-rate $b/V$ punctured convolutional code of perforation matrix $P_1$, a lower rate-compatible punctured convolutional code can be generated from the starting code by replacing zeros of $P_1$ with ones down to the rate $1/n$ parent code.

Alternatively, punctured codes can be found by starting from the rate $1/n$ code and performing rate-compatible puncturing. With this method starting at the parent code $1/n$, the code equivalent to the starting high rate code of the first method is slightly worse. However, in most cases, the difference is negligible [53].

In order to get optimal RCPC codes, the ODS criterion is used to search for the best codes. Clearly, if the starting code is a non-catastrophic code, then the low rate code obtained is also a non-catastrophic code.

Optimum RCRC codes are found with parent code $1/n$, $n$ ranging from 2 to 4, constraint lengths $K$ ranging from 3 to 11, repetition period $p$ ranging from 2 to 8, and repeated code bits $n$ up to 40.

For example, from the non-catastrophic code rate $7/8$ punctured convolutional code obtained from rate $1/2$ code with perforation matrix $P_1$ given by Equation (3.3), a rate $7/10$ compatible code is generated by replacing two zeros of $P_1$ with ones. The obtained code is a non-catastrophic code, and its
perforation matrix \( P_2 \) is of the form
\[
P_2 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}
\]
(3.4)

Rate-compatible convolutional coding

When using rate-compatible convolutional (RCC) codes, transmission of incremental redundancy in hybrid-ARQ schemes and continuous rate variation, changing from low to high error protection within a data frame. A very powerful and flexible RCC coding scheme is obtained when by combining puncturing (RCPC coding) and repetition (RCRC coding): puncturing for high-rate codes and repetition for low rate codes.

In [54], it is shown that optimized RCRC codes with high parent rate perform almost as good as optimized RCPC codes with lower parent rate. Thus, when RCC coding is applied, a low parent code rate should be used for the best performance. For RCPC codes, as a rule of thumb, a parent code rate of \( 1/4 \) with a puncturing period \( p = 8 \) seems to be a reasonable choice for larger code family.

A second option for obtaining a powerful and flexible RCC coding scheme is by combining RCPC codes and nested convolutional codes. Again, a set of rate-compatible codes with a wide range of code rates is obtained.

As seen before in the paragraph on RCRC codes, these codes are in general not MFD and are very difficult to find. RCRC coding obtains low rates by repeating bits of the original code while nested codes use new code polynomials but without doing a full search in the design process. Thus nested codes must be at least as good as RCRC.

Finally, nested codes are to be preferred to repetition codes, since nesting allows new polynomials for lower rates while repetition only reuses previous polynomials. However, it can be somewhat less complex to find a repetition code as compared to a nested code. Although there is a difference in the design procedure, when the codes are known, the decoding complexity is approximately the same for a given rate and memory.

3.2.4 Conclusion

In this section, several FEC schemes are considered for application in the hybrid-ARQ scheme for UWB ad-hoc networks. Rate-compatible convolutional (RCC) codes, are found to be good candidates. Next to the RCC coding, superorthogonal convolutional (SOC) coding with variable code rates is also applicable for error correction coding in the hybrid-ARQ scheme.

When using RCC coding, puncturing and nesting can be applied to obtain the best higher and lower rate codes, using the ODS criterion, derived from a parent code.

When nested convolutional codes are compared to SOC codes, the low-rate nested codes, in general, have higher free distances. Also, super-orthogonal codes only exist for a few code rates since the rates and constraint lengths are related by \( R = 1/2^{K-2} \). The nested codes exist for all rates \( 1/n \), where \( 4 \leq n \leq 512 \). In addition, the nested codes are rate-compatible.
Concluding, RCC coding using puncturing and nesting is the best option for error correction in the hybrid-ARQ scheme for UWB ad-hoc networks.
3.3 The applied ARQ scheme

First, the ARQ scheme will be selected by considering the three types of ARQ and then the hybrid-ARQ type will selected among the three different types of hybrid-ARQ.

3.3.1 ARQ strategies

The three types of ARQ scheme, which were presented in section 1.4.2 are: the stop-and-wait ARQ (SAW), the go-back-N ARQ (GBN) and the selective-repeat ARQ (SR).

SAW is the simplest of the three, with low system complexity and very little overhead. GBN provides increased throughput relative to SAW by eliminating off-time which SAW spends waiting for acknowledgements. The system complexity is increased, because the frames in the sending window have to be stored. Although there is no transmitter off-time, there are lengthy periods when the transmitter is repeating frames which have previously been received correctly.

Further improvement in throughput of ARQ systems is be achieved by SR. It includes the logic and frame buffering necessary to reduce retransmissions to only incorrectly received messages. This causes increased system complexity and slightly increased frame overhead compared to the other to schemes: frames have to be numbered in sequence and frames buffering is required at both the transmitter and receiver.

In many articles, the choice for a specific ARQ scheme is not considered, and the ideal SR, which requires infinite buffers in transmitter and receiver, giving the highest throughput, is assumed. In the next subsections a foundation is made on which a choice can be made for an ARQ protocol for hybrid-ARQ in a high speed UWB ad-hoc network.

Stop-and-wait ARQ protocol

First, SAW is considered. Since the transmitter does not use the available channel during time intervals it waits for an acknowledgement, the maximum data transfer rate that can be supported is limited. To compare all three ARQ schemes, a formula for the efficiency is derived.

The round trip time \( t_0 \) is given by

\[
\begin{align*}
t_0 &= 2t_{\text{prop}} + 2t_{\text{proc}} + t_f + t_{\text{ack}} \\
&= 2t_{\text{prop}} + 2t_{\text{proc}} + \frac{n_f}{R} + \frac{n_a}{R}
\end{align*}
\]

where \( 2t_{\text{prop}} \) is the propagation delay, \( 2t_{\text{proc}} \) the processing delay, \( t_f \) and \( t_{\text{ack}} \) the time to transmit a data-frame or acknowledgement-frame. \( n_f \) and \( n_a \) are the total number bits in the data- and acknowledge-frame, respectively, and \( R \) the channel transmission rate.

When independent errors in the channel are assumed to occur, then \( P_f \) is the probability a frame arrives with errors. The efficiency of the SAW scheme is then given by

\[
\eta_{\text{SAW}} = \frac{R_{\text{eff}}}{R} = \frac{n_f - n_a}{t_0/1 - P_f} = \frac{1 - \frac{n_a}{n_f}}{1 + \frac{n_a}{n_f} + \frac{2(t_{\text{prop}} + t_{\text{proc}})R}{n_f} (1 - P_f)}
\]

(3.6)
where $R_{eff}$ is the effective channel transmission rate, and $n_o$ the number of bits that are overhead (header and CRC) in the frame.

**Go-back-N ARQ**

GBN attempts to combine the desirable features of SR and SAW, i.e., packets are transmitted continuously, as is SR, without the need to buffer out of sequence packets and there is no re-sequencing overhead.

GBN is completely efficient, if the window size at the sender, sending window ($W_s$), is large enough to keep the channel busy, and if the channel is error-free. $W_s$ is determined by the product of the delay and the bandwidth, called the delay-bandwidth product. When errors are assumed, the time to deliver a frame is given by

- $t_f$ if the first frame transmission succeeds
- $t_f + W_s t_f/(1 - P_f)$ if the first transmission does not succeed

$$t_{GBN} = t_f (1 - P_f) + P_f \left\{ t_f + \frac{W_s t_f}{1 - P_f} \right\} = t_f + P_f \frac{W_s t_f}{1 - P_f}, \text{ and (3.7)}$$

$$\eta_{GBN} = \frac{n_f - n_o}{R} = \frac{1 - n_o}{1 + (W_s - 1) P_f (1 - P_f)} \text{ (3.8)}$$

**Selective repeat ARQ**

With SR, only the erroneous frames are retransmitted, so frames can be accepted, even when there out of sequence. Hence, packets received out of sequence have to be buffered and sequenced before they can be delivered.

The number of transmissions required to deliver a frame is $t_f/(1 - P_f)$, and

$$\eta_{SR} = \frac{n_f (1 - n_o)}{R} = (1 - \frac{n_o}{n_f}) (1 - P_f) \text{ (3.9)}$$

**Comparison of the ARQ schemes**

Assumed that $n_a$ and $n_o$ are negligible relative to $n_f$, and $L = 2(t_{prop} + t_{proc})R/n_f = (W_s - 1)$ then Equations (3.6),(3.8), and (3.9) become

$$\eta_{SAW} = \frac{1 - P_f}{1 + \frac{n_a}{n_f} + \frac{2(t_{prop} + t_{proc})R}{n_f}} \approx \frac{1 - P_f}{1 + L}, \text{ (3.10)}$$

$$\eta_{GBN} = \frac{1 - P_f}{1 + (W_s - 1) P_f} \approx \frac{1 - P_f}{1 + L P_f}, \text{ and (3.11)}$$

$$\eta_{SR} = (1 - \frac{n_o}{n_f}) (1 - P_f) \approx (1 - P_f). \text{ (3.12)}$$

To determine $L$, first $t_{prop}$ is considered. $t_{prop} = d/v$, $d$ the distance between sender and receiver, and $v$ the propagation speed of EM waves ($3 \times 10^8$ m/s). In a single hop system, $d$ is assumed 10m maximum. $t_{prop}$ is then 33 ns.
3.3 The applied ARQ scheme

The length of $t_{\text{proc}}$ can be assumed to be between 1 and 10 ms, as is common with UMTS devices. With regard to $L$, $t_{\text{proc}}$ will be the main influence. Assuming the worst case, $t_{\text{proc}} = 10$ ms, $n_f = 2000$ bits in every packet, and $R = 400$ Mbps. Then, $L = 4000$, which results in a sending window size of $W_s = 4001$.

When looking at Equations (3.6), (3.8), and (3.9), it can be concluded that SR is the only option for high speed UWB networks, because the efficiency of the other two schemes decreases immediately when there is some error probability, due to the large $L$.

In a multihop environment, the propagation delay $t_{\text{prop}}$ will increase (as $d$ increases), as well as the processing delay (more nodes between sender and receiver) $t_{\text{proc}}$. The extra delay in multihop networks will require larger buffers than in the single hop system. Concluding, the SR scheme is the best ARQ-scheme for a fast hybrid-ARQ protocol.

Conclusion

In this section the ARQ scheme for the hybrid-ARQ scheme for UWB ad-hoc networks is chosen among different considered schemes. There are three different ARQ schemes, namely, stop-and-wait (SAW), go-back-N (GBN), and selective repeat (SR).

SAW sends a packet and wait for an acknowledgement before sending the next packet. The other two send continuously and have a sliding window that contains the number of packets that can be sent when the system is waiting for an acknowledgement. GBN retransmits the whole window when an error occurs, while SR only retransmit the erroneous packets.

SR and GBN perform better than SAW because of the continuous nature. Drawback of GBN is increased complexity, when compared to SAW and high bandwidth usage when many errors occur. SR has an even more complex architecture, but the efficiency is the highest by far in a fast UWB environment, because it is not dependent on the bandwidth-delay product, as the other two schemes are. Concluding, SR is the best option for ARQ in a high speed UWB ad-hoc network.

3.3.2 The hybrid-ARQ type

When the two error control schemes, FEC and ARQ, are compared, ARQ is simple and provides high system reliability. However, ARQ systems have severe drawbacks: their throughput falls rapidly with increasing channel error rate. Systems using FEC maintain constant throughput (equal to the code rate $R = k/n$) regardless of the channel error rate. But it is hard to achieve high system reliability with FEC. These drawbacks in both ARQ and FEC can be obviated if the two control schemes are properly combined.

Hybrid-ARQ has as main drawback that is gives a lower throughput on good channels than simple ARQ, because of the extra transmitted redundancy for error-correction. However, for low $E_b/N_0$, which is likely to perform in UWB ad-hoc networks, hybrid-ARQ gives better throughput.
For hybrid-ARQ type I, the amount of redundant data in a packet is fixed for all channel conditions as well as the amount of retransmitted data. Hybrid-ARQ type II and III are more powerful protocols since they adapt to changing channel conditions (using more redundancy only when required, i.e., incremental redundancy (IR)) for higher throughput on good channels and still offering good throughput on bad channels. Both types combine received (re)transmissions, which give them a performance gain with respect to type I.

In type II, the retransmission are typically non identical with the original transmission. The retransmitted attempt carries additional redundant information for error correction purposes. The retransmitted amount of redundancy is different for different retransmission attempts. Basically, the retransmissions can only be decoded after combining with the previous ones.

In type III, the retransmitted packet also carries additional redundancy information for error correction purposes. And unlike type II, each retransmission is now self-decodable. This implies that the transmitted packet is decodable without the combination of previously transmitted packets. This is useful if some transmitted packets are damaged in such a way that almost no information is reusable. Main drawback of a type III hybrid-ARQ system is the improved complexity, and only there are no codes available with such low code rates as is the case with SOC codes and RCC codes.

In the case of using hybrid-ARQ type II using RCPC codes and nested codes, there will be information bits and redundant bits transmitted (i.e., the nested bits), when all the punctured bits have been transmitted. This happens at the low code rates, when channel conditions are poor. RCC codes combine the advantage of a small amount of redundant bits when using puncturing and information and redundant bits when nesting is used. This makes the type II hybrid-ARQ scheme more applicable than the type III for UWB ad-hoc networks.

**Conclusion**

In this section, the three hybrid-ARQ schemes were considered. Type I hybrid-ARQ is the same as simple ARQ, but adds some extra redundancy for error-correction. Because only one redundancy version exist, the scheme is not very efficient. Type II and type III hybrid-ARQ adapt to varying channel conditions, by varying the amount of redundancy transmitted. In retransmissions is not the whole original packet transmitted, as in type I, but only the extra redundancy information for error correction. In type-III these packets are self-decodable at the cost of an increased complexity at the transmitter and receiver.

When RCPC codes and nested codes are used in the type II hybrid-ARQ scheme, small packages are used for retransmission when puncturing is applied (in good channel conditions), and large packets of information and redundant bits when nested bits are used (in bad channel conditions). This type of hybrid ARQ is best suited for hybrid-ARQ protocol.
3.4 Further improvements of the hybrid-ARQ scheme

Some ideas that were discussed in the previous chapter 2 are presented here, as possible improvements to the hybrid-ARQ scheme that is selected in the previous sections in this chapter.

3.4.1 Interleaving

In subsections 2.2.4 and 2.2.5 interleaving was introduced to spread bursty error events among various code symbols, thus increasing the robustness to bursts. The depth of the interleaver is important, because when it is too short, the spreading will not have enough effect, and when the depth is too large the interleaver causes too much delay. The delay introduced by FEC is minimal compared to the interleaving delay.

Next to the interleaving depth, the position of the interleaver in the coding-modulation scheme is determinative for the spectral characteristics and BER performance of the transmitted signal, as well. There are bit-, frame-, and chip interleaving, of which chip interleaving has the best performance.

3.4.2 QoS provisioning

In subsection 2.3.2, a mechanism is introduced for controlling the transmitter activity in order to let the system save radio resources as well as energy in the case of bad channel conditions.

Depending on the mode, the transmitter retransmits requested packets when errors occur, or stops transmitting and starts polling the receiver to probe the channel status. Which mode to choose depends on the nature of the data: loss-sensitive traffic or other traffic.

This idea of different modes for specific data can be implemented in the scheme to save energy. Another way of implementing QoS into the hybrid-ARQ scheme is applying different amounts of redundancy to particular data. For example, as from the first transmission, the use of lower code rates for loss-sensitive traffic.

3.4.3 General Conclusion

This chapter described the process of selecting an appropriate hybrid-ARQ protocol for UWB ad-hoc networks. This was done on the basis of the boundary conditions that are set by the nature of the UWB ad-hoc network.

When repetition coding, SOC coding, and rate-compatible coding were considered for the FEC scheme in the hybrid-ARQ protocol, the rate-compatible coding, consisting of RCPC coding and nested coding turned out to be the best alternative.

As far as the ARQ mechanism in the hybrid-ARQ protocol was concerned, there were three different strategies. When they were compared on efficiency, the SR ARQ mechanism proved to be the best option.
Finally, the hybrid-ARQ type was chosen. The efficiency of the rate-compatible codes combined with the wide range of code rates resulted in a type II hybrid-ARQ mechanism as the best choice.

Concluding, the proposed hybrid-ARQ protocol is a type II scheme with a SR ARQ scheme and a rate-compatible coding scheme, which consists of punctured and nested coding.

Some enhancements to improve the efficiency of this proposal are the use of an interleaver and the utilization of QoS provisioning.
Chapter 4

Simulation model

In the previous chapter, a proposal was done for an adaptive hybrid-ARQ protocol for UWB ad-hoc networks. To investigate the error control performance of this proposal under realistic UWB channel conditions, a simulation model is constructed.

In order to see the performance of this protocol in perspective, a comparison is made with a FEC scheme for UWB ad-hoc networks, the SOC coding proposed in [38] [39], on which ARQ is added. For this error correction scheme, neither punctured nor nested codes exist, so a type I hybrid ARQ is chosen. Adaptivity for this hybrid-ARQ scheme is achieved by varying the constraint length of the super-orthogonal encoder.

In this chapter, the simulation models for both cases are discussed. The chapter is structured as follows:

First, in section 4.1, a design overview is given of the simulation model of the protocols that are compared and general assumptions for the simulation model are stated.

Next, in section 4.2, the complete implementation of the protocol is presented. Each element of the transmitter and receiver, and the channel model are considered.

Finally, in section 4.3, the functionality of both protocols is discussed with regard to the way packets are treated in the hybrid-ARQ system, both at the sender and receiver side.
4.1 Design overview and general assumptions

The system will be implemented in the *matlab* environment, because this has a faster execution time as the *simulink* environment. For UWB, there don't exist much predefined blocks in *simulink*, and if they do exist, they have to be modified. So, *matlab* is then preferred to use.

The purpose of the simulations will be to examine the effect of the hybrid-ARQ mechanism on the throughput. In order to examine the effect of retransmissions, a stop-and-wait ARQ mechanism is implemented with no time delay. Thus, frames are sent continuous, but when errors occur, the effect of the extra delay for retransmissions is immediate present in the throughput. Throughput will be defined in next chapter on simulations and results.

For error correction, SOC codes and RCC codes are used.

SOC coding is used in a type I hybrid ARQ scheme, where each retransmission consists of the same data-packet encoded with a stronger code.

The RCC coding, which consists of RCPC codes and nested codes, uses Incremental Redundancy when retransmissions are requested and is of a type II hybrid-ARQ. This means that each retransmission must be combined with previous information before decoding is possible. The functionality of both hybrid-ARQ protocols will be discussed in more detail in section 4.3

In the simulation model several assumptions are made to measure the performance under realistic UWB channel conditions, but also to keep the simulation controllable. These assumptions are:

- A bit rate of 100 Mbps when the code rate is $R = 1/4$. In this case, the symbol rate will be 440 Mbps, because there is some overhead added (header and CRC) at each data-packet. Subsequently follows that the maximum bitrate is 440 Mbps, when no coding is applied,

- First, a packet length of 2000 bits was assumed, but to keep simulations controllable and not too time-consuming, this was reconsidered, and a packet length of 320 bits was chosen,

- Ideal synchronization,

- A stationary channel during transmission of one data packet,

- The feedback channel is error free: acknowledgements, positive or negative, are always transmitted and received without errors,

- No influence of antennas on the transmitted pulses or the received signal.

- A continuous equally distributed stream of data from the network-layer
4.2 Design Implementation

In this section, all elements that are implemented in the model in order to simulate the hybrid-ARQ protocol will be discussed. First, all the elements in the transmitter part will be considered, then, the channel design and finally the elements in the receiver part of the model.

4.2.1 Transmitter design

The transmitter is designed in four elements: the data source, the part where the CRC calculation and addition takes place, the FEC encoding part, and the part where the modulation of the coded bits onto pulses takes place. These elements are shown in figure 4.1

![TRANSMITTER Diagram]

Figure 4.1: The transmitter part in the complete model.

Data source

The data source represents the data that is transmitted from the network-layer to the datalink-layer. It is kept in a buffer and each time a data-packet is to be transmitted, there are 320 bits collected from this buffer. In this simulation, the total data in the buffer is assumed to be a multiple of 320.

The data source in the model generates uniformly distributed random binary numbers. A data-packet is generated when a positive acknowledgement is received.

After a data-packet is constructed of 320 bits, a header will be added to it of 4 bits. This header contains information what kind of the packet is constructed, the sequence number, and an acknowledgement field (an acknowledged sequence number). Kinds of packets are: data, acknowledgements or punctured bits.

CRC generation and addition

The packet contains 324 bits and next the CRC bits will be calculated. This is done by dividing the 324 bits data-packet by a 32 bits polynomial. The rest of this division is added at the end of the packet as the CRC. The length of the total packet, i.e., the frame, is now 356 bits. It is depicted in Figure 4.2
The generator polynomial that is used here, as is used in the IEEE 802 standard is:
\[ x^{32} + x^{26} + x^{23} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1, \]

With this method, called CRC-32, bursts of length 32 and all bursts affecting an odd number of bits will be detected.

**Error correction encoding**

Now that the header and CRC are added to the data-packet, the complete packet can be encoded. There is one exception: when the channel quality is good, this means that no errors occur with the highest code rate, the packet will not be encoded.

In the FEC encoding element either *SOC coding* or *RCC coding* is applied. Next, these two coding mechanisms are discussed separately.

**SOC coding** The SOC encoder encodes the packet with a code rate, which is dependent on the constraint length. The code rate is \( R = \frac{1}{n}, \) where \( n = 2^{(K - 2)}, \) where \( K \) is the constraint length. The constraint length is used from \( K = 3 \) up to \( K = 6. \) Code rates that are possible are:

\[ \{1, 1/2, 1/4, 1/8, 1/16\} \]

The SOC codes are particularly appropriate for UWB systems, because of their unique properties, such as low code rate, and near-optimal performance [37].

**RCC coding** The rate compatible coding consists of rate compatible *puncturing* convolutional (RCPC) codes and *nested* codes. The codes are *rate compatible*, which means that lower rate codes use the same coded bits as the higher rate codes plus one or more additional bit(s). With these codes, *incremental redundancy* is possible.

For the RCPC codes, a convolutional encoder with constraint length \( K = 7 \) is used, a parent code of \( R = 1/4 \) is chosen, and *puncturing period* \( p = 8. \) The bits to be punctured are described by an \( n \times p = 4 \times 8 \) perforation matrix, consisting of zeros and ones. Generator polynomials for the parent code are in octal form (117, 127, 155, 171).

The perforation matrix is reflected in Table 4.1. All elements of the matrix are '1', but dependent of the code rate some elements are changed to '0'. This is

<table>
<thead>
<tr>
<th>Header</th>
<th>Data field</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bits</td>
<td>320 bits</td>
<td>32 bits</td>
</tr>
</tbody>
</table>

Figure 4.2: The frame architecture.
4.2 Design Implementation

done as follows: for each code rate the '1' bit at the place in the matrix according to that rate and all '1' bits belonging to lower rates up to rate $R = 8/32$ have to be punctured and are thus changed to a '0'.

For example, when the code rate is $R = 8/30$, the perforation matrix contains all '1's, except at the places with the coordinates (0, 7) and (3, 1), which are '0'.

Table 4.1: Perforation matrix, $K = 7$, parent code (117, 127, 155, 171)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8/25</td>
<td>8/17</td>
<td>8/21</td>
<td>8/27</td>
<td>8/9</td>
<td>8/31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>8/26</td>
<td>8/29</td>
<td>8/23</td>
<td>8/24</td>
<td>8/10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8/18</td>
<td>8/15</td>
<td>8/12</td>
<td>8/16</td>
<td>8/14</td>
<td>8/22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8/13</td>
<td>8/30</td>
<td>8/11</td>
<td>8/28</td>
<td>8/19</td>
<td>8/20</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When a packet is encoded with parent rate $R = 1/4$ and then punctured, the punctured packet is then modulated and transmitted, and the punctured bits are saved in a buffer, in case a retransmission is required. If a positive acknowledgement is received, the punctured bits are discarded and the buffer is ready for the punctured bits of the next encoded packet.

The nested codes are obtained by nesting additional coded bits to the coded bits encoded with the parent code. See Table 4.2 for the octal representation of the generator polynomials. Obviously, these rates are still rate compatible with the higher ones. The lowest code rate is chosen to be $R = 1/16$, to compare it to the SOC coding, which also has a lowest code rate of $R = 1/16$.

When RCPC coding and nested coding are combined, 37 possible rates are available:

$$\{1, 8/9, 8/10, 8/11, \ldots, 8/32, 1/5, 1/6, \ldots, 1/16\}$$

These RCC codes are appropriate for UWB ad-hoc networks because of their superior performance compared to orthogonal and super-orthogonal codes [55] and the wide range of code rates available, which with the channel capacity can be better exploited.

Table 4.2: Nested Codes, additional polynomials, $K = 7$, parent code (117, 127, 155, 171)

<table>
<thead>
<tr>
<th>Rate</th>
<th>Add. Pol.</th>
<th>Rate</th>
<th>Add. Pol.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/5</td>
<td>135</td>
<td>1/11</td>
<td>117</td>
</tr>
<tr>
<td>1/6</td>
<td>173</td>
<td>1/12</td>
<td>135</td>
</tr>
<tr>
<td>1/7</td>
<td>135</td>
<td>1/13</td>
<td>157</td>
</tr>
<tr>
<td>1/8</td>
<td>145</td>
<td>1/14</td>
<td>135</td>
</tr>
<tr>
<td>1/9</td>
<td>137</td>
<td>1/15</td>
<td>123</td>
</tr>
<tr>
<td>1/10</td>
<td>155</td>
<td>1/16</td>
<td>175</td>
</tr>
</tbody>
</table>
Modulation and Pulse transmission

Modulation  Modulation is done using Phase Shift Keying (PSK). Because binary symbols (0 and 1) are used, the modulation method is Binary PSK (BPSK). In practice, this means that a binary '0' is converted to a '1', and a binary '1' in converted to a '−1'.

Pulse transmission  Next, the pulses are generated. The pulse is modelled as a Gaussian monocycle. A modulated '1' causes a starting negative flank, a modulated '−1' a starting positive flank. The width of the transmitted pulse $T_w$ corresponds to the channel model time resolution and equals $T_w \approx 0.167$ ns.

In this simulation model, a bit rate of 100 Mbps is assumed, when the code rate is $R = 1/4$. Because of the extra bits of header and CRC, the rate of the total packet is 110 Mbps. Each pulse represents one coded bit, so time between pulses is then 2.27 ns. Each period of 2.27 ns is then sampled with a frequency of 25 GHz. This means that 57 samples represent each period of 2.27 ns.

Because the channel rate is fixed, 440 Mbps, the bit rate will vary when the code rate changes. A maximum bit rate of 400 Mbps is achieved when no coding is applied, and the lowest bit rate of 25 Mbps is attained with code rate $R = 1/16$. Other bit rates $R_b$ can be calculated when the code rate $R$ is known:

$$R_b = 400.10^6 \times R \text{ bits/s}$$

4.2.2 Channel design

In reality, the channel is one element, but in this simulation model the channel is represented by two elements as can be seen in Figure 4.3.

The first element simulates the multipath Rayleigh fading. The rayleighchan function in matlab is used for this, to construct a Rayleigh fading channel object that models each discrete path as an independent Rayleigh fading process. Main parameters for this function are used as suggested in [39] for a NLOS environment. In this case, the mean number of significant paths, $NP_{10dB} = 35$, and RMS delay spread, $\tau_{rms} = 15$ ns.

The second elements adds white Gaussian noise to the faded signal. This is done using the awgn function in matlab. The noise that is added per sample is dependant of the SNR value, that is a parameter of this function.

The main parameters used in the model,

4.2.3 Receiver design

The receiver is designed in four elements: the passband filter, the demodulator, the decoder and the syndrome detector, where is decided if the data-packet is correctly received. These elements are shown in Figure 4.4.

Filter

The filter in the model is a third-order Chebyshev type II passband filter with cutoff frequencies $f_1 = 2$ GHz and $f_2 = 8$ GHz. The stopband attenuation is 20 dB.
4.2 Design Implementation

**CHANNEL**

- Multipath Rayleigh fading
- Additive White Gaussian Noise

Figure 4.3: The channel part in the complete model.

**RECEIVER**

- Filter
- BPSK demodulation
- FEC decoder
- Syndrome decoder

Figure 4.4: The receiver part in the complete model.

**Demodulation**

In the demodulator, the positive and negative part of the filtered signal are integrated and then is decided whether a pulse with a positive starting flank or a pulse with a negative starting flank was received. The first one represents a ‘-1’, the second one a ‘1’, respectively.

**Error correction decoding**

The demodulated signal is then led into the decoder. For decoding, the Viterbi decoding algorithm is used. To compare both hybrid-ARQ schemes equally, the output values of the demodulator are ‘hard’ values. So the input of the Viterbi decoder is also *hard-input*.

When *puncturing* is applied, the punctured places in the packet are filled with zeros, in order to reconstruct a 4 \times 8 matrix. Then, the input, consisting of elements ‘-1’, ‘0’, and ‘1’ is fed to the Viterbi decoder.

**Syndrome detection**

The decoded packet is now the same size as the original frame, which consisted of the data-packet, with a header in front of it, and a CRC-32, at the end. The
decoded packet is then divided by the CRC-32 polynomial, that is also known at the receiver. The result of this division is called the *syndrome*.

When the syndrome is zero, the frame is accepted, and the header and CRC are removed. If it is a data-packet and the sequence number is equal to the one that is expected, the data is moved on to the network-layer. A positive acknowledgement is sent back to the transmitter.

When the syndrome is not zero, an error has occurred. What happens next is dependant on the protocol. The hybrid-ARQ type I scheme with SOC encoding discards the packet, as the hybrid-ARQ type II scheme with RCC encoding saves the packet in a buffer. In both cases a negative acknowledgement is sent back to the transmitter.
4.3 Functioning of the protocol

For both hybrid-ARQ schemes, the ARQ mechanism is the same. A packet is transmitted and if an error occurs, a retransmission is requested. The next packet is sent only when the previous packet is acknowledged by the receiver.

Both schemes are adaptive, i.e., the code rate adapts to the channel quality. This mechanism will be described in the subsections 4.3.2 and 4.3.3.

The difference between the two protocols, apart from the different encoding mechanisms, i.e., convolutional coding or super-orthogonal convolutional coding, is the type of hybrid-ARQ. After the subsection on adaptivity, the two different schemes are discussed as they are implemented in the simulation model.

4.3.1 Adaptivity

When a data-packet is constructed with a header and a CRC, it will be encoded. Before encoding there is decided with which code rate the packet will be encoded. When a positive acknowledgement is received, the code rate will be increased, otherwise it will be decreased.

After the first simulations the conclusion was that this mechanism resulted in a high amount of retransmissions. To decrease this number of retransmissions, a period before changing the code rate was implemented. This was done with a counter. It counts the number of transmitted packets, up to a predefined value, after which the code rate is changed to one higher. In this model, a value of 5 is chosen, before the code rate is increased. Each time a retransmission is required, the counter is reset. As an experiment, the “rate increasing period”, denoted as \( N_i \), will be changed, to see its influence on the throughput of the protocol.

When a negative acknowledgement is received by the transmitter, the code rate for the next packet to be transmitted is decreased by one. This in done each time a retransmission is required. In order to not become lost in an endless loop of retransmissions, the maximum number of retransmissions of the lowest code rate is set to two. The exception to this is when the original message is encoded with the strongest code, then only one retransmission is allowed.

After this last retransmission the packet is accepted, whether it contains errors or not. This is a tradeoff between reliability and throughput.

4.3.2 Hybrid ARQ type II using RCC codes

Both transmitter and receiver architecture are different in the two hybrid ARQ schemes, as far as encoding of the packets and handling of errors, and thus retransmissions, is concerned. What happens in both in the hybrid-ARQ scheme type II that uses RCC coding is described in the following paragraphs.

Transmitter, first transmission

When the code rate is determined, the packet can be encoded. There exist three possibilities: no coding, encoding with a rate higher than or equal to \( R = 1/4 \), and encoding with a rate lower than \( R = 1/4 \).
The first case is straightforward: when the code rate is $R = 1$, no coding is applied and the packet is sent on for modulation.

In the second case, the packet will be encoded with a parent code with rate $R = 1/4$ and then punctured according to Table 4.1 to get the desired higher code rate. The punctured bits are stored in a buffer, in case a retransmission is required. When the rate is equal to $R = 1/4$, no puncturing is applied.

In the third case, to achieve lower code rates than $R = 1/4$, polynomials are added, i.e., nested, to the original polynomials of the $R = 1/4$ code. These polynomials were presented in Table 4.2. In this way, the lower code rates are constructed. No bits are buffered here.

**Transmitter, retransmission**

When a negative acknowledgement is received, there are four options: The original code rate was $R = 1$, it was between $R = 8/9$ and $R = 8/31$, it was equal or lower to $R = 1/4$, or the wrong package has been sent (the sequence number was not correct).

In the first case, when the original rate was $R = 1$, the next lower code rate will be $R = 8/9$. In this case the original uncoded frame, which consists of a data-packet combined with header and CRC will be encoded with rate $R = 1/4$, and then maximal punctured to get the code rate of $R = 8/9$.

In the second case, when the original code rate was between $R = 8/9$ and $R = 8/31$, punctured bits will be retransmitted. The punctured bits of the original message were stored in a buffer, and now the bits that correspond to the last puncturing, i.e., the bits necessary to get a code rate one lower, are selected. This packet is then provided with a header and CRC and transmitted.

In the third case, when the original code rate was equal or lower than $R = 1/4$, a packet is constructed of bits that are generated by a convolutional code with the polynomial, provided in Table 4.2, to get a code rate one lower. These bits are then provided with a header and CRC and ultimately transmitted.

In the last case, the previous packet is retransmitted. In this model this option is not possible, because the feedback channel is assumed error free.

**Receiver, first reception**

When a packet is received, it is filtered and demodulated. The decoding is rather straightforward and already described in 4.2.1 in the paragraph on error correction decoding. Finally, the syndrome is checked. If an error has occurred and/or the sequence number is not the one that is expected, a negative acknowledgement is sent back to the transmitter. Dependent on the error that has occurred, sequence number and/or syndrome, different types of negative acknowledgements are sent.

In case the sequence number is the one that is expected, but the syndrome is not equal to zero, the filtered and demodulated packet is saved in a buffer.

**Receiver, retransmission**

In case a retransmission is requested for a non-zero syndrome, there are three options for the received packet, after it is filtered, demodulated, decoded, and
the syndrome detected: it can contain a data-packet encoded with rate $R = \frac{8}{9}$, when the original code rate was $R = 1$, it can contain punctured bits, or bits for nesting.

In the first case, a data-packet is received with code rate $R = \frac{8}{9}$. It is already decoded and if the syndrome is zero, the data-packet is accepted. Otherwise, another retransmission is requested and the filtered and demodulated packet is stored in the buffer.

In the second case, punctured bits are received. Whether the syndrome is zero or not, these bits are always accepted. They are combined with the packet that was stored in the buffer after the first reception. Then the packet is again decoded with a rate that is one lower than the rate of the previous reception. If the syndrome is zero, the data-packet is accepted, otherwise, another retransmission is requested and the combined packet, i.e., before it was decoded, is stored in the buffer.

In the third case, nested bits are received. These bits are also always accepted. They are combined with the stored packet and together they are decoded at one rate lower than the previous packet was decoded. Also here, if the syndrome is zero, the data-packet is accepted, otherwise, another retransmission is requested and the combined packet, i.e., before it was decoded, is stored in the buffer.

From the afore-mentioned follows that multiple retransmissions are possible. Each new received packet, punctured or nested bits, contains information to decode the stored packet with a rate one lower than was done in the last decoding. As said before, when the lowest code rate is reached, only two retransmissions are then allowed, before the packet is accepted anyhow. Obviously, these packets of nested bits to decode at a rate $R = \frac{1}{16}$ contain each time the same nested bits, encoded with the strongest code.

Example

To better understand the protocol, and the difficulties that are behind the surface, an example is provided. The general functioning of the protocol can be seen in Figure 4.5

In the example is assumed that there is encoding with puncturing at the transmitter and an error occurs which is detected at the receiver side.

Before transmitting the frame, the encoded packet is punctured according to the code rate that is requested. The encoded packet is coded with a rate $R = \frac{1}{4}$, so it contains 1424 bits. Because the encoded packet, 1424 bits long, is not divisible by 32, the length of the perforation matrix, there exist two lengths of packets that contain punctured bits: 44 bits and 45 bits. Thus, the location of the punctured bits in the buffer is important.

When the transmitted packet is received at the receiver, the packet has to be stuffed with 'empty' bits at the places of the original punctured bits, to recover the original size of 1424 bits, after which it can be decoded. The receiver has also a perforation matrix, in order to know where to put the received bits, and where to put stuffed bits.
If, after decoding, an erroneous packet is found by the syndrome, the packet that was received, before it was decoded is stored in a buffer. A negative acknowledgement is sent, in order to receive additional redundant bits.

When the negative acknowledgement is received at the sender, the punctured bits that are needed for the receiver to decode the original packet with a code rate that is one step lower than the previous, are fetched from the buffer with all the punctured bits. These bits are also provided with a header and CRC. This frame is then transmitted to the receiver.

At the receiver side, this message is decoded and, whether is received correctly or not, combined with the previous received packet. The correct location of these bits in the original packet is essential here. Next, the packet is again stuffed to the original 1424 bits, and decoding is attempted.

In this example, there is a correct decoding. If there is again an erroneous packet found, the process is repeated and the next amount of punctured bits is requested, which will be combined with the whole uncoded packet that is constructed up to now. If all punctured bits have been sent, nested bits will sent.

### 4.3.3 Hybrid ARQ type I using SOC encoding

Here, the transmitter and receiver design will be discussed as far as it concerns the type I hybrid ARQ using SOC encoding.
4.3 Functioning of the protocol

Transmitter

The adaptivity mechanism, as described before, determines with which code rate the data-packet, with added header and CRC, will be encoded. If a retransmission is required, the adaptivity mechanism sets the rate one lower, so that the original data-packet, with added header and CRC, is now encoded with a stronger code.

Receiver

When a packet is decoded and the syndrome is not equal to zero, the packet is discarded and a retransmission is requested by sending a negative acknowledgement. The retransmitted packet will be decoded one rate lower than the previous one.

In this adaptive hybrid-ARQ type I model, multiple retransmissions are possible as well as the other model. In this model, it is the same, that when the lowest code rate is reached, only two retransmissions are allowed, before the packet is accepted anyhow. Obviously, these packets, which are encoded at a rate $R = 1/16$, contain each time the same data-packet.
Chapter 5

Simulations and Results

This chapter will present the experiments that were done to examine the hybrid-ARQ protocol performance and give the results that were obtained by these simulations. Next to that, the results of the simulations will be analyzed and discussed.

The simulation model that was presented in the previous chapter will be used for the experiments. This chapter is divided in four sections, where in each section, one of the four experiments is described.

The experiments will compare the two different FEC schemes without ARQ, then both FEC schemes will be used in a hybrid-ARQ type I, and the RCC coding also in the hybrid-ARQ type II. Finally, the hybrid-ARQ type II is examined by changing the puncturing period and the period before changing the code rate.

In each subsection, the results of the experiments will be analyzed and discussed on basis of the results that are presented in that section.
5.1 Experiment 1

5.1.1 Simulation

In the first experiment, a comparison is made between the two different FEC coding schemes: SOC coding and RCC coding. The simulation model is used without the ARQ mechanism, thus only error correction is applied.

For different values of the signal-to-noise ratio (SNR) per coded bit the BER performance is measured. The SNR is used instead of $E_b/N_0$, because in the next experiments $E_b/N_0$ is not a fixed value but it changes when the coding rate changes. This can be seen from Equation 5.1,

$$E_b/N_0 = 10 \log_{10}\left(T_{sym}/T_{samp}\right) - 10 \log_{10}(R) + \text{SNR}$$ \hspace{1cm} (5.1)

where $T_{sym}$ is the signal's symbol period, $T_{samp}$ is the signal's sampling period, and $R$ the code rate. Binary modulation is assumed in this equation, and $E_b/N_0$ and SNR are in dB.

For SOC coding, simulations were performed with constraint length $K = 3$, $K = 4$, and $K = 5$. Code rates are then $R = 1/2$, $R = 1/4$, and $R = 1/8$ respectively.

For RCC coding, the constraint length is $K = 7$, and the punctured code of rate with rate $R = 1/2$, the parent code with rate $R = 1/4$, and the nested code with rate $R = 1/8$, are used.

Data-packets of 320 bits are used, a total packet of 64,000 bits is to be transmitted (200 data-packets). Each simulation is performed five times, and at the end the mean is calculated. The results of this simulation are depicted in Figure 5.1.

5.1.2 Result analysis

The results in Figure 5.1 show that the SOC codes have a better BER performance than RCC codes, when it concerns code rates $R = 1/2$ and $R = 1/4$, and the SNR is low. When the SNR increases, RCC codes have better BER performance than the SOC codes from SNR = 7 dB, when the code rate is $R = 1/2$, and from SNR = 3 dB, when the code rate is $R = 1/4$. When the code rate is $R = 1/8$, the RCC coding outperforms SOC codes for all SNR ratios.

The purpose of this experiment is to estimate the performance of both codes when they are applied in an adaptive hybrid-ARQ scheme. In an adaptive scheme, the code rate will decrease when error statistics diminish. From Figure 5.1 can be seen that for low BER values, RCC codes perform always better than SOC codes. The conclusion from experiment 1 is then, that it is expected that the RCC codes will perform better than the SOC codes in an adaptive hybrid-ARQ scheme, whether it is type I or type II.
5.2 Experiment 2

5.2.1 Simulation

In the second experiment, the hybrid-ARQ protocol is simulated as described in the previous chapter. The puncturing period is \( p = 8 \), and the period before increasing the code rate is \( N_i = 5 \).

For comparison, RCC coding and SOC coding are both simulated in the hybrid-ARQ type I, with the rates used in experiment 1, and with the rate of \( R = 1/16 \) added. In case of RCC this is done with extra nested bits (i.e., extra generator polynomials), and in case of SOC, the constraint length \( K = 6 \) is used. Next to that, the hybrid-ARQ type II using RCC codes is simulated.

Data-packets of 320 bits are used, a total packet of 64,000 bits is to be transmitted (200 data-packets). Each simulation is performed five times, and in the end the mean is calculated.

In this experiment, the throughput is measured for different values of the SNR. Throughput is defined here as the number of accepted bits at the receiver per unit time. The results are depicted in Figure 5.2.

The BER for all received frames was zero.

5.2.2 Result analysis

The results from this simulation as depicted in Figure 5.2 show that the proposed hybrid-ARQ type II scheme has a higher throughput at all values of the SNR.
than both type I variants. This shows that it is more efficient to retransmit small packets with rate-compatible punctured bits or nested bits than to retransmit the whole packet encoded with a stronger code.

Even when both error coding schemes are applied in the same hybrid-ARQ type, namely type I, the application of RCC codes results in higher throughput than the SOC codes. Considering the results of experiment 1, this was expected.

From SNR = 12 dB and lower, retransmissions become numerous and the strength of the RCC codes becomes visible in the throughput. When the SNR attains low values, the lowest code rates are applied, and from Figure 5.1 can be seen that performance of both coding schemes is almost similar as these values. All three hybrid-ARQ schemes have almost the same throughput at this values.

The incremental redundancy of the type II hybrid-ARQ appears to be a much more efficient way to achieve high throughput than the retransmitting of whole packages as in type I hybrid-ARQ. But at SNR values of 9 and 10 dB, type I and II perform almost equally. To better understand the three protocols, the erroneous received data-packets and the amount of retransmissions that follows will be considered in the next experiment.
5.3 Experiment 3

5.3.1 Simulation

In the third experiment the amount of packets that can not be decoded correctly, and the amount of retransmissions that are needed before a correct decoding is possible, are considered. This is done for all three hybrid-ARQ schemes of experiment 2. The results were obtained in the same simulation as in experiment 2.

First, the amount of erroneous received data-packets, are considered. These are depicted in Figure 5.3. Then the total amount of retransmissions that are transmitted, is depicted in Figure 5.4, and finally the amount of retransmissions that are needed before successful decoding is achieved in depicted in Figure 5.5.

Additionally, these values are also measured with different values for the rate changing period, but only for SNR = 8 dB, SNR = 10 dB and SNR = 12 dB, and only for the hybrid-ARQ type II. These values are chosen because Figure 5.5 shows that the amount of retransmissions is the highest with these values. For the rest, the same simulation parameters are used as in experiment 2.

In experiment 2, the period before the rate was increased, was $N_i = 5$. In this experiment, these values are changed from $N_i = 1, 3,$ and 7. The throughput is measured as well.

In Figure 5.6, the throughput of all different values of $N_i$ are depicted for the three SNR values. In Figure 5.7 the amount of erroneous received packets,
and in Figure 5.8 the total amount of retransmissions are depicted. Because the amount of retransmissions until successful decoding is possible, is not dependant on $N_i$, these values are the same as in Figure 5.5 for the hybrid-ARQ type II.

### 5.3.2 Result analysis

First, Figures 5.3, 5.4, and 5.5 are considered. The first thing that catches the eye in Figure 5.3 is that the amount of erroneous packets in the type II hybrid-ARQ scheme is much lower than the other two type I schemes. But when looking at Figure 5.4, the total amount of retransmissions is the same. The explanation can be found in Figure 5.5, which shows that the amount of retransmissions that must be sent before a successful decoding is achieved is much higher in the type II hybrid-ARQ scheme than in the type I hybrid-ARQ scheme.

Next thing that is striking, is the whimsical behavior in Figure 5.3 of mainly the hybrid-ARQ type II line. All these lines are directly related to amount of retransmissions that are needed before a successful decoding can be achieved, as can be seen in Figure 5.5. With regard to the hybrid-ARQ type II, this Figure shows that between SNR = 5 dB and SNR = 14 dB the amount of retransmissions before a successful decoding is achieved varies between 1 and 3.5 retransmissions.

Below SNR = 7 dB, there are mainly nested codes used, which explains the diminution. Retransmissions with nested bits contain more redundant bits than the small packages with punctured bits. The horizontal parts show that
the Viterbi decoder can correct packets at certain code rates better than others: when starting from SNR = 20 dB, a less whimsical course would be expected, when the SNR decreased.

Because of the large steps between code rates in the type I hybrid-ARQ, retransmissions are almost always correct decodable. But because of the large retransmissions, due to the complete retransmission of the packet and extra redundant data, the type I hybrid ARQ is not as efficient as the type II hybrid-ARQ.

This high number of retransmissions in the hybrid-ARQ type II scheme, especially between SNR values of SNR = 6 dB and SNR = 12 dB, explains why this scheme has an almost equally throughput performance as the type I hybrid-ARQ.

Because the steps between code rates are small in the hybrid-ARQ type II, and thus little redundancy is sent in retransmissions, much retransmissions are required. This causes the code rate to drop significantly, and because the period before the code rate is increased, is equal to $N_i = 5$, many data-packets can be sent, before the code rate is reached, when the error occurred. This explains why the amount of erroneous data-packets is much lower. But at the cost of not using the maximum channel capacity.

In Figures 5.6, 5.7, and 5.8 the effect of the different values of $N_i$ can be seen. Remarkably here is, that the highest throughput is achieved at $N_i = 1$, but that at this value, the amount of erroneous data-packets and thus, the amount.
of retransmissions is much higher than the other values of $N_i$. The total amount of retransmissions is almost 200, so almost each data-packet is received in error. But, nevertheless, the throughput is the highest, because the retransmissions are only small packets of punctured bits, and the data-rate increases quick after it has decreased after several retransmissions, so the channel capacity is optimally exploited.

In ad-hoc networks, the amount of retransmissions should be kept low, because the amount of traffic increases enormous. Thus, a large value of $N_i$ is required, which results in less errors, and retransmissions should contain enough redundant bits that a correct decoding is possible after the first retransmission. Therefore, in the next experiment, the puncturing period is changed, which results in larger steps between code rates, and thus more redundant bits per retransmission, which results in an increased chance of successful decoding after the first retransmission.

Figure 5.6: Throughput, $p = 8$, $N_i = 1,3,5,7$. 
Simulations and Results

5.4 Experiment 4

5.4.1 Simulation

In experiment 4, the effect of the puncturing period in the type II hybrid-ARQ scheme is considered. The puncturing period is related to the amount of retransmissions that are needed before successful decoding can be achieved. When the puncturing period is decreased, the distances between code rates becomes larger, and a retransmitted packet of punctured bits will contain more punctured bits than with a higher puncturing period. More (redundant) punctured bits increases the chance of successful decoding. Conversely, larger retransmitted packets decrease the throughput.

The same simulations are done as in experiment 3, with different values of the rate changing period $N_i$, and for puncturing periods $p = 4$ and $p = 2$.

For $p = 4$, the bits to be punctured are described by a $4 \times 4$ perforation matrix. The same parent code of $R = 1/4$ is used as with $p = 8$. The perforation matrix is depicted in Table 5.1

When $p = 4$, only 25 possible rates are available:

\[ \{1, \frac{4}{5}, \frac{4}{6}, \frac{4}{7}, \ldots, \frac{4}{16}, \frac{1}{5}, \frac{1}{6}, \ldots, \frac{1}{16} \} \]

For $p = 2$, the bits to be punctured are described by a $4 \times 2$ perforation matrix. The same parent code of $R = 1/4$ is used as with $p = 8$. The perforation matrix is depicted in Table 5.2

Figure 5.7: Erroneous received data-packets, $p = 8$, $N_i = 1, 3, 5, 7$. 

5.4 Experiment 4

Figure 5.8: Total amount of retransmissions, $p = 8$, $N_i = 1,3,5,7$.

Table 5.1: Perforation matrix, $K = 7$, $p = 4$, parent code (117, 127, 155, 171).

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4/12</td>
<td>4/14</td>
<td>4/13</td>
<td>4/15</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>4/5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>4/10</td>
<td></td>
<td>4/11</td>
</tr>
<tr>
<td>3</td>
<td>4/8</td>
<td>4/6</td>
<td>4/9</td>
<td>4/7</td>
</tr>
</tbody>
</table>

Table 5.2: Perforation matrix, $K = 7$, $p = 2$, parent code (117, 127, 155, 171).

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2/6</td>
<td>2/7</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2/5</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2/4</td>
<td>2/3</td>
</tr>
</tbody>
</table>

When $p = 2$, only 19 possible rates are available:

$$\{1, \frac{2}{3}, \frac{2}{4}, \frac{2}{5}, \ldots, \frac{2}{8}, \frac{1}{5}, \frac{1}{6}, \ldots, \frac{1}{16}\}$$

In Figure 5.9, the throughput of all different values of $N_i$ are depicted for the three SNR values and $p = 4$. In Figure 5.10 the amount of erroneous received packets, and in Figure 5.11 the total amount of retransmissions are depicted.
The amount of retransmissions until successful decoding is possible, is depicted in Figure 5.12.

![Figure 5.9: Throughput, $p = 4$, $N_i = 1,3,5,7$.]

In Figure 5.13, the throughput of all different values of $N_i$ are depicted for the three SNR values and $p = 2$. In Figure 5.14 the amount of erroneous received packets, and in Figure 5.15 the total amount of retransmissions are depicted. The amount of retransmissions until successful decoding is possible, is depicted in Figure 5.16.

### 5.4.2 Result analysis

From Figures 5.9 and 5.13 can be seen that at values of $N_i = 3$, $N_i = 5$ and $N_i = 7$, that the difference in throughput performance of $p = 4$ and $p = 2$ is always around 10 Mbits/s or less, where $p = 4$ has the best performance. At SNR = 10 dB, the throughput with $p = 4$ performs significantly better than $p = 2$ and $p = 8$, which is depicted in Figure 5.6. At this SNR value, this size of retransmitted packet of punctured bits that are smaller than those constructed with $p = 2$ and larger than constructed with $p = 8$ are the best tradeoff for the best throughput performance. At the other values SNR = 8 dB and SNR = 12 dB, the performance, for $N_i = 3,5$ and 7, and for all puncturing periods ($p = 2,4$ and 8) is comparable.

To consider other arguments for choosing the best $N_i$ and $p$, the amount of erroneous received data-packets is investigated. The amount of erroneous received data-packets that are depicted in Figure 5.10 and 5.14 are also comparable as values $N_i = 3, 5,$ and 7 are considered. In Figure 5.11 and 5.15 can then
be seen that the total amount of retransmissions is slightly lower for $p = 2$, than for $p = 4$. This can explained by the lower amount of retransmissions that is needed at $p = 2$, and that can be seen when Figure 5.12 and 5.16 are compared.

From this figures can also be seen that the amount of retransmissions is independent from the period before the code rate is increased, $N_i$. The deviations originate from rounding in calculations.

After these experiments, two conclusions can be drawn. First, The hybrid-ARQ type II scheme with puncturing period $p = 4$ has a slightly a better throughput over the measured values as the schemes with puncturing period $p = 2$ and $p = 8$.

The second conclusion is that the hybrid-ARQ type II scheme with puncturing period $p = 2$ need less retransmissions before a successful decoding is achievable and that there are in total less retransmissions needed.

When the best applicable hybrid-ARQ scheme for UWB ad-hoc networks is chosen, the nature of the ad-hoc network should be considered. To avoid high amount of retransmissions, and thus a busy network, and achieve a reliable network, where only few retransmissions are required, and if they occur, only a single retransmissions in needed for a correct decoding, then the hybrid-ARQ type II scheme with puncturing of $p = 2$, and a long period before the code rate is changed, $N_i = 7$.

When selective repeat ARQ is applied in the hybrid-ARQ scheme, the sender keeps sending packets, until the acknowledgement, positive or negative, is received. In case of a negative acknowledgement, the code rate is decreased, which
applies to all following transmitted packets, until again an acknowledgement is received. For the rest, the adaptivity protocol works similar as used in this simulation.
Figure 5.12: Retransmissions until successful decoding is possible, $p = 4$, $N_i = 1,3,5,7$.

Figure 5.13: Throughput, $p = 2$, $N_i = 1,3,5,7$. 
Figure 5.14: Erroneous received data-packets, $p = 2, N_i = 1,3,5,7$.

Figure 5.15: Total amount of retransmissions, $p = 2, N_i = 1,3,5,7$. 
Figure 5.16: Retransmissions until successful decoding is possible, $p = 2$, $N_i = 1, 3, 5, 7$. 
Chapter 6

Conclusion and Recommendations

This thesis provides an overview of the process of selecting and examining an adaptive hybrid-ARQ protocol for UWB ad-hoc networks.

In the first two chapters an extensive review of the literature was done on error coding, the UWB technology and ad-hoc networks, and also related work to the subject of this thesis was examined.

On the basis of this research a proposal for the hybrid-ARQ protocol was done in chapter 3, where several candidates for the ARQ and FEC mechanism, as well as the hybrid-ARQ type were investigated.

In chapter 4 the simulation model was presented with which the four experiments, that were described in chapter 5, were performed. Also the functioning of the protocols that were examined in chapter 5, is presented in chapter 4.

This final chapter provides a general conclusion of all the work that is done in this project and presented in this thesis. Next, some recommendations are done for future research that result from the research and experiments done in this project.

The structure of this chapter is as follows:

Section 6.1 presents the general conclusion of this thesis. It looks back at the things that are achieved, the new possibilities that have arisen, and new improvement that are implemented.

Section 6.2 gives some recommendations for future research. New research questions emerged from this project, some research could not be done for several reasons, and ideas for new possibilities in the future have arisen. These are all collected in this section.
6.1 Conclusion

This thesis is constructed in two parts. The first part concerns the extensive literature research in the area of ad-hoc networks, the UWB technology and error coding, and on articles closely related to the topic of this thesis, hybrid-ARQ for UWB ad-hoc networks, and which deepen the three topics that were discussed before.

UWB is a promising radio technology that offers great potential for future short range, low power, low cost and high data rate ad-hoc networks. To achieve reliable and efficient communication between nodes in this single- or multihop network error coding should be used. A mechanism that combines the reliability of ARQ and the efficiency of FEC is hybrid-ARQ.

The UWB radio technique, which can be divided in two different approaches that are currently widely used, Impulse Radio and Multiband, is a rapidly evolving technique. New ways of modulating the signal, like PSM, in combination with new pulse waveforms, like PSWF pulses, improve the properties of UWB for more reliable, faster communication, within the boundaries of the FCC spectral mask requirement for indoor communications.

Channel models for the UWB channel, like the Saleh-Valenzuela model, lead to new designs of receiver architectures. Different types of Rake-receivers are used to exploit diversity by constructively combining the separable monochromes from distinguishing propagation paths for improving transmission performance.

Error control coding is a way to improve the transmission performance on a wireless channel. The bursty nature of the channel, which results in high error rates, requests strong coding schemes (FEC), and smart retransmission algorithms (ARQ). Because the quality of the channel is time-varying, adaptivity is added to protocol, in order to exploit the channel capacity to the full extend.

Different coding schemes exist, like SOC codes, RCC codes, turbo codes, etc, all with different properties when used in a hybrid-ARQ scheme. The choice for a coding schemes is linked with the ARQ mechanism and the type of hybrid-ARQ. Dependant on the situation these properties should be considered.

In this project, a hybrid-ARQ scheme for UWB ad-hoc networks is considered, which implements more boundary conditions for the choice of this protocol. On the basis of the literature research a proposal is done: a rate-compatible coding scheme, which consist of RCPC codes and nested codes, in combination with selective repeat ARQ and a type II hybrid-ARQ. The coding scheme enables a wide range of code rates, from high to very low code rates (for low to high BER conditions in the channel), SR ARQ makes it possible to continuous sent messages, and the type II hybrid-ARQ attends that small amounts of data are sent in retransmissions. Because rate-compatible codes are used, subsequent retransmissions contain complementary information, and not already transmitted redundant bits. Thus the channel is not excessively and useless loaded.

The second part concerns the simulation of the proposed hybrid-ARQ scheme to examine its performance. The simulation model is discussed, as well as the functioning of the protocols. For comparison, two different coding schemes, RCC and SOC, are used, both in the adaptive type I hybrid-ARQ, and RCC as proposed in the type II hybrid-ARQ scheme.
First, the research results show that the applied RCC codes give a lower BER than the SOC codes at the same SNR, when they are examined in a pure FEC scheme. Next, it turns out that the hybrid-ARQ type II scheme performs better than both hybrid-ARQ type I schemes, because it is more efficient in retransmission than the type I. From both type I hybrid-ARQ schemes, the scheme which uses RCC codes performs better than the scheme with SOC codes. This can be explained by the results of the first experiment, where it was shown that the RCC codes perform better than the SOC codes.

When the period before increasing the code rate, \( N_i \), is changed in order to see which affect it has on the throughput, it shows that the best throughput is achieved with a period \( N_i \) of 1 message. This results in a high amount of retransmissions, which is undesired. Moreover, the amount of retransmissions before a successful decoding can be achieved varies between 1 and 3.5. This measurement is done when the puncturing period is \( p = 8 \).

Finally, the puncturing period is changed from \( p = 8 \) to \( p = 4 \) and \( p = 2 \). The steps between subsequent code rates are then larger, and thus the amount of punctured bits is larger. This results in a higher chance of correct decoding after the first retransmission. Furthermore, the period before increasing the code rate, \( N_i \), is changed. It shows that the throughput is slightly higher for longer \( N_i \) with \( p = 4 \) than \( p = 2 \) and \( p = 8 \). However, these simulations also show that the amount of retransmissions is lower with \( p = 2 \) than with \( p = 4 \) and \( p = 8 \) for longer \( N_i \). They show as well that the chance of a successful decoding is higher when \( p = 2 \).

Concluding, from the results show, that when a hybrid-ARQ protocol for UWB ad-hoc networks is considered, a type II hybrid-ARQ scheme, using RCC codes for FEC coding, with a puncturing period \( p = 2 \), with a long period before the code rate is increased of \( N_i = 7 \), and with SR ARQ as the retransmission algorithm, is the best candidate.
6.2 Future work

In this section, some suggestions for future research directions are proposed.

1. The effect of larger data-packets. The simulations in this project were done with data-packets with a length of 320 bits, but a size of 2000 bits is more likely in high speed UWB networks. The effect of ARQ will be slower on larger packets, thus this needs additional research what effect \( N_i \) has here.

2. The protocol should be tested in environments with more nodes, single- and multihop, and with timing delays due to processing and propagation. The effect of a return channel with error statistics should also be considered.

3. The effect of the Selective Repeat ARQ mechanism on the protocol. Because there exists a large sliding window, many (re)transmissions occur, before acknowledgements are received. An improved adaptivity mechanism may be searched here.

4. There should be research done on the size of the buffers that are needed when Selective Repeat ARQ is used. A timing window of 4001 packets is required for only one hop. Research should be carried out on the feasibility of implementation of these buffers in small low power devices.

5. Finally, research should be done on the effect of improvements to the hybrid-ARQ protocol as interleaving, methods for achieving QoS, and different modulation methods and pulse waveforms.


[10] Company webpage containing links to several articles on UWB http://www.freescale.com/uwb
http://www.plextek.com/uwb.htm


Appendix A

List of Abbreviations

AMC  Adaptive Modulation and Coding
ARQ  Automatic Repeat Request
BCH  Bose Chaudhuri & Hocquenghem codes
BER  Bit Error Rate
BPSK Binary Phase Shift Keying
BPSM Biorthogonal PSM
CRC  Cyclic Redundancy Check
FCC  Federal Communications Commission
FEC  Forward Error Correction
FTP  File Transfer Protocol
GBN  Go-back-N ARQ
GSM  Global System for Mobile Communications
HARQ Hybrid-ARQ
HSDPA High-Speed Downlink Packet Access
HTTP HyperText Transfer Protocol
IP   Internet Protocol
IPI  Inter Pulse Interference
ISI  Inter Symbol Interference
ISO  International Organisation for Standardization
LAN  Local Area Network
LCC  Logical Link Control
LDPC Low Density Parity Check codes
MAC  Medium Access Control
MAN  Metropolitan Area Network
MFD  Maximum Free Distance
MPC  Multipath Components
OBI  Observation Interval
ODS  Optimum Distance Spectrum
OOK  On//Off Keying
OSI  Open System Interconnection
PAM  Pulse Amplitude Modulation
PAN  Personal Area Network
PHY  Physical layer
PPM  Pulse Position Modulation
PRF  Pulse Repetition Frequency
PSD  Power Spectral Density
PSK  Phase Shift Keying
PSM  Pulse Shape Modulation
PSWF Prolate Spheroidal Wave Functions
P2P  Peer-to-Peer
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCC</td>
<td>Rate-Compatible Convolutional code</td>
</tr>
<tr>
<td>RCPC</td>
<td>Rate-Compatible Punctured Convolutional code</td>
</tr>
<tr>
<td>RCRC</td>
<td>Rate-Compatible Repetition Convolutional code</td>
</tr>
<tr>
<td>RMDP</td>
<td>Reliable Multicast Data Distribution Protocol</td>
</tr>
<tr>
<td>RS</td>
<td>Reed-Solomon code</td>
</tr>
<tr>
<td>SIR</td>
<td>Signal to Interference Ratio</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
</tr>
<tr>
<td>SAW</td>
<td>Stop-and-Wait ARQ</td>
</tr>
<tr>
<td>SR</td>
<td>Selective Repeat ARQ</td>
</tr>
<tr>
<td>TCP</td>
<td>Transport Control Protocol</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunication System</td>
</tr>
<tr>
<td>UWB</td>
<td>Ultra-Wideband</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>WiFi</td>
<td>Wireless Fidelity, industry name for WLAN</td>
</tr>
<tr>
<td>WiMAX</td>
<td>Wireless MAN</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless LAN</td>
</tr>
<tr>
<td>WPAN</td>
<td>Wireless PAN</td>
</tr>
</tbody>
</table>