TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Nachrichtentechnik

System and Cross–Layer Design for Mobile Video Transmission

Thomas Stockhammer

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor–Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.–Prof. Dr.–Ing. habil. Gerhard Rigoll
Prüfer der Dissertation:	
	1. Univ.–Prof. Dr.–Ing., Dr.–Ing. E.h. Joachim Hagenauer, i.R.
	2. Univ.–Prof. Dr.–Ing. Eckehard Steinbach

Die Dissertation wurde am 19.06.2008 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 24.09.2008 angenommen.

Preface

I have made this letter longer than usual, only
because I have not had the time to make it shorter.
Blaise Pascal, (1623-1662), Lettres provinciales.

This thesis has been written during my time as a research assistant at the Institute for Communications Engineering (Lehrstuhl für Nachrichtentechnik, LNT). Many people at the Institute and elsewhere have contributed to this work. Firstly, I would like to thank my supervisor Professor Dr.-Ing. Dr.-Ing. E.h. Joachim Hagenauer for providing me with the opportunity to work at his institute, for all his support of my thesis and for the patience to await its completion. I am also very grateful to Professor Dr.-Ing. Eckehard Steinbach for acting as a co-supervisor and Professor Dr.-Ing. habil. Gerhard Rigoll for heading the commission. At the same time, many thanks also to Professor Antonio Ortega from the University of Southern California (USC) for accepting a place on the commission - despite logistical reasons that made it impossible in the end and to Professor Ken Zeger from the University of California, San Diego (UCSD) for making possible my visit to the UCSD during spring 2000. Also many thanks to all colleagues and partners for exciting collaborations during my standardization work for the ITU-T, MPEG, IETF, 3GPP, DVB, and JVT. In particular, I am grateful to Professor Dr. Thomas Wiegand and the other video experts from the Fraunhofer HHI in Berlin for introducing me to the secrets and mysteries of video coding.

The LNT has always been a place with an enjoyable atmosphere and I was surrounded by extraordinary people. The collaboration with many diploma and master students made the time at the Institute a memorable experience, and many of the colleagues and students have become friends. I would like to mention in this respect Professor Dr. Günter Söder, Dr. Alexander Seeger, Dr. Rainer Bauer, Dr. Christian Weiß, Dr. Volker Franz, Thomas Effern, Daniel Pfeifer and Houda Kamoun. Today, I am delighted to collaborate with some of my former colleagues and partners from the LNT, in particular Professor Hagenauer, Dr. Ingo Viering, Günther Liebl and Christian Buchner. I am very grateful to all my colleagues at Nomor Research for their courage to found and join a company that is heavily based on research work started at the LNT. Also, my thanks to Dr. Michael Luby and all other colleagues at Digital Fountain as well as all my friends at home and all over the world. Thank you all for your support and for encouraging me to complete this thesis.

While generating this work, I became friends with Dr. Michael Mecking and Dr. Hrvoje Jenkač. They truly proved their friendship by not leaving anything undone to incite me on the final stages of this work and proofread faster than I could type. Micki, H, thank you so much. Finally, this thesis would not have been possible without the continuous support of my parents and family, I am deeply grateful for providing me with all the opportunities. I apologize to Lisa-Maria and Maximilian for spending so much time on this work and not with them. Last, but not least, I deeply thank the rock of our family, my wife Christina, for her understanding and love during the generation of this thesis.

Bergen, November 2008 THOMAS STOCKHAMMER

Contents

Kurzfassung

Diese Arbeit untersucht Systementwurfs- und -optimierungsverfahren für die Verwendung von Videoapplikationen im Mobilfunk. Besonderer Wert wird dabei auf das Zusammenspiel und die Optimierung von Funktionen auf verschiedenen Schichten des Übertragungsmodells unter Berücksichtigung der Bedingungen unterschiedlicher Applikationen gelegt. Basierend auf verschiedenen Übertragungsmodellen, die an aktuelle und zukünftige Mobilfunksysteme angelehnt sind, werden adäquate Übertragungs- und Videocodierverfahren vorgestellt. Speziell wird der mitentwickelte H.264/AVC Videocodierstandard sowie eine proprietär entwickelte, skalierbare Erweiterung auf die Verwendbarkeit in Mobilfunk- und Internet-Übertragungsumgebungen untersucht. Hierzu werden neuartige Fehlerschutzverfahren entwickelt, die sich speziell auf die im Mobilfunk zu erwartenden zeit- und ortsvariablen Empfangsbedingungen anpassen können und ein Zusammenspiel mit den vorgeschlagenen Videocodier-Verfahren erlauben. Die Auswahl von Optionen innerhalb der Videocodier- und Fehlerschutzverfahren sowie deren Zusammenspiel wird durch qualitätsoptimierende Auswahlverfahren unterstützt. Die Arbeit wird begleitet von informationstheoretischen Betrachtungen, praktisch relevanten Systementwürfen sowie ausführlichen Simulationsergebnissen, die die Vorzüge der vorgeschlagenen Methoden im Vergleich zu existierenden Systemen belegen und quantifizieren.

Abstract

This work investigates system designs and optimizations for the application of video services in mobile communication systems. Special focus is put on the cooperation and optimization of functions in different layers of a transmission system taking into account the service requirements of different video applications. Based on channel models that are derived from state-of-the-art and emerging mobile communication systems, we introduce suitable transmission and video coding methods. In particular, the co-developed H.264/AVC video coding standard as well as a proprietary scalable extension are intensively analyzed for their applicability in mobile and Internet communication environments. Innovative error protection tools are designed which enable adaptation to the varying reception conditions in mobile communication environments and enable the cross-layer optimization with the proposed video coding tools. The selection of video coding and error protection options as well as their cooperation is supported by the development of quality-optimizing selection and rate allocation schemes. The findings in this work are verified by information-theoretic justifications, practically relevant system designs, as well as extensive simulation results. The benefits of the proposed methods with respect to existing systems are shown and the realizable gains are quantified.

1

Introduction

You see what I see

When AT&T asked the management consulting company McKinsey in the early 1980s to predict the number of mobile phones that will exist in the beginning of the new century, the answer was a mere 900,000 – citing several 'insurmountable' barriers to market growth[1]. The company – whose Bell Labs invented cellphones – listened to McKinsey. The consultants estimate of the market in the year 2000 was off by a factor of 120: From essentially zero in the mid 80's, there are more than 3.3 billion active cellphones — there is now one cellphone for every two humans on Earth. Mobile phones nonetheless continue to get hooked up at a rate of more than 1,000 a minute, so the penetration of multimedia-capable phones will continue to increase. The largest camera maker today in the world is — Nokia. Therefore, not only the numbers of mobile users are increasing, but the requested services are also becoming more complex and data consuming. Besides making regular phone calls, mobile users desire to be able to send multimedia messages, browse the Internet, stream or download music and video, watch sports highlights, or share their excitements using visual means with their friends, family and business partners — *you see what I see*!

Optimized Systems require Optimized Components and Their Cooperation

Clearly, mobile operators see mobile video as a long-term revenue creating service. However, for this to happen users expect satisfying quality: The adoption of new services and the percentage of churn rates will be highly dominated by the perception of the mobile users – nowadays being used to astonishing high-definition video quality or full-screen video streaming over the Internet. In 3G deployments, typically a cell cannot host more than 5-10 video users at the same time. Therefore, to increase the number of viewers, operators only have the choice to increase the system bandwidth or to increase the efficiency in emerging video services. Despite new spectrally efficient radio technologies will emerge, the efficiency of video services is crucial for operators to create an economically viable service.

What is required are innovative approaches that expand the reach of the existing spectrum and

[1] Joel Garreau, Joel, "Our Cells, Ourselves," "Washington Post," February 24, 2008.

at the same improve video quality. There are several aspects to obtain more successful and more efficient mobile services: First of all it is essential to understand the nature and characteristics of mobile radio channels, especially the dynamics and variability in reception conditions. To provide reliable transmission, adaptive schemes are inevitable. As important as this is a clear understanding of the requirements of the video service itself, what are the key performance indicators leading to user satisfaction: These include the perceived video quality or the experienced latency and delay. Note that different video services may have different latency requirements: Whereas for example conversational video require lowest delay, for video streaming these constraints are more relaxed. Such knowledge should be exploited in optimizing the efficiency of mobile video services.

Furthermore, optimized mobile video systems should consist of optimized components: Therefore, highly efficient video coding algorithms are required which make the H.264/AVC standard one of the prime candidates for mobile video services. Furthermore, it is of tantamount importance that the video codecs can be integrated into communication environments: Adaptive, network-optimized video codecs are essential. In addition, error protection tools suitable for transmission over mobile radio channels and flexible enough to compensate the propagation dynamics are integral components of modern mobile communication systems: However, forward error correction, power control schemes, or retransmission protocols need to be adapted to channel conditions and video service requirements. Therefore, the cooperation of source coding and error protection tools taking into account service requirements and mobile radio conditions is an essential ingredient of a well-designed mobile video service. Designing efficient mobile video services requires careful system and cross-layer design.

Main Contributions of this Work

This work investigates system designs and optimizations for the application of video services in mobile communication systems. Specifically we focus on the cooperation and optimization of functions in different layers of a transmission system taking into account the service requirements of different video applications. The main contributions of this work are summarized in the following:

- We have developed several H.264/AVC network integration and error resilience features that have partly been adopted as part of the H.264/AVC standard.

- We have designed and developed a scalable extension of an interim version of the H.264 test model referred to as Progressive Texture Video Coding (PTVC) that is very suitable for network integration.

- We have derived a formalized description for the transmission of non-scalable and progressive pre-encoded video over packet-loss channels.

- We have developed and implemented a novel error protection scheme, namely regressive Unequal Error Protection (UEP) in combination with the Far End Error Decoding (FEED) and path shortening at the receiver suitable for transmitting progressively coded sources.

- Based on information-theoretic considerations we have derived a distortion function and optimized channel coding rates for a multi-layer transmission system that shows that UEP is optimal for multi-layer transmission, but multi-layer transmission introduces a penalty when compared to single layer transmission.

- We have developed a complexity and quality-scalable system for progressive image transmission over fading channels that makes use of the FEED algorithm. The corresponding rate allocation problem is formulated and solved by a dynamic programming approach.
- We have designed, developed and implemented an H.264/AVC-based toolbox for error-resilient video transmission. The tools have been optimized and analyzed in realistic packet loss environment for a set of formalized test conditions.
- A multiple description video coding scheme has been realized by the use of the PTVC and an Unequal Erasure Protection (UXP) scheme. The corresponding rate allocation problem was formulated and solved by a dynamic programming approach.
- We have provided detailed system design guidelines on the usage of H.264/AVC video features, application layer transport features, as well as transport features for different applications in UMTS video service environments. The guidelines have been derived based on a formalized test framework.
- We have proposed several system and cross-layer designs for the transmission of low-delay mobile conversational video. For a well-defined framework different system designs making use of adaptive video coding and error resilience tools have been implemented. For each of the systems, a careful cross-layer design including rate and power allocation procedures has been proposed.
- We have proposed an H.264/AVC-based video streaming system that relies on three major principles: receiver playout buffering, temporal scalability, and advanced bitstream switching using the Switching–Predictive (SP) frame concept of H.264/AVC. For optimized performance of the system a scheduling algorithm has been developed and remarkable performance gains could be shown when compared to state-of-the-art systems.

The present work is structured as follows:

In **chapter 2**, we introduce the technical preliminaries for transmitting video over networks. A high-level system overview is provided including the functions and components of a video transmission system. Suitable video quality measures, the information flow within system, aspects related to timing, delays, and buffering as well as requirements and objectives of different typical video applications are introduced. This chapter is also dedicated to present the considered transmission environment with special focus on packet-based multimedia communication over the Internet. The integration of such multimedia services in packet-radio networks is presented. The chapter also introduces necessary system abstractions and channel models that are intensively used in the remainder of this work: Channel models for the mobile radio channel comprise discrete-time models, models on the link-layer, as well as channel characteristics and models on the application layer.

In **chapter 3**, we introduce source and video coding tools that have been used and partly developed in this work. Some preliminaries on source coding principles are discussed, including theoretical background on lossy source coding, rate and quality adaptive source coders, as well as scalable and progressive source coding. Then, practical issues in video source coding are introduced: We provide an overview on existing and deployed video coding standards. As these standards are usually only dedicated to the specification of the decoder, some typical encoder operations, namely operational encoder and rate control, are introduced. Specific focus in this work

is on H.264/AVC and its usage in network environments, as parts of the standard have been developed in the course of this work: We introduce the H.264/AVC network interface, referred to as Network Abstraction Layer (NAL), the use of the H.264/AVC Video Coding Layer (VCL) in error-prone transport environments, as well as stream switching. The chapter also discusses scalable video coding: We introduce a proprietary extension of H.264/AVC referred to as Progressive Texture Video Coding (PTVC), together with rate control and reference picture selection schemes. The performance of video coders is summarized in different environments. To simplify rate allocation procedures and video transmission simulations, we formalize the presentation of pre-encoded data: We provide a framework for encoding, transmission and decoding of pre-encoded data.

In **chapter 4**, we introduce tools that are essential or at least beneficial to provide reliable transmission in mobile communication systems. Some preliminary information on information-theoretic background, rate-compatible punctured channel coding, Unequal Error Protection (UEP), schemes relying on channel-state information and power control, as well as Automatic Repeat reQuest (ARQ) schemes is provided. Specific realizations of channel codes and their decoding methods are introduced. We specifically focus on convolutional codes and sequential decoding of those codes. The integration of convolutional codes in systems with fading is discussed focussing on the system design and radio access for block-fading channels. The performance of different options in terms of outage probabilities is assessed. Furthermore, we introduce a specific realization of an UEP scheme relying on regressive UEP and the Far End Error Decoding (FEED) principle: An optimized decoding solution based on a trellis-representation is proposed, and beyond this a complexity-reduced decoding based on sequential decoding algorithms is developed. The chapter also introduces Forward Error Correction (FEC) schemes for packet-lossy channels.

In **chapter 5**, we discuss different system designs for progressive source coding and UEP. Some information-theoretic considerations are employed: Based on certain non-restricting assumptions, we are able to provide a distortion estimation for fixed system configuration. This allows the derivation of optimized channel coding rates including special cases for equal transmission block size and equal information part length. One practical example supports the discussion and conclusions on the findings. We also introduce an Unequal Erasure Protection (UXP) framework for progressively coded sources including an optimized rate allocation algorithm. Finally, a progressive image transmission system to operate over fading channels is proposed. This example shows the usefulness and potentials of UEP and progressive source coding: By employing an optimized rate allocation, selected experimental results for our proposed system shows significant gains when compared to a reference system.

In **chapter 6**, we discuss video transmission in packet-lossy environments. Initially, we focus on H.264/AVC-compliant end-to-end video transmission: We formalize the transmission of H.264/AVC packetized video, introduce error concealment algorithms and assess the performance of basic error resilience tools. Motivated by these non-satisfying results, we further optimize error-resilient video transmission: We propose an operational encoder control applicable in error-prone environments and introduce Interactive Error Control (IEC). The performance of these enhanced features is assessed under dedicated test conditions. Furthermore, we propose several systems that make use of FEC: For low-delay applications in packet-loss environments, a simple parity-check based approach is proposed. For less stringent delay requirements, a framework that combines FEC with H.264/AVC as well as of UXP with the PTVC are introduced and assessed.

In **chapter 7**, we introduce video services in UMTS-like environments. Based on system designs incorporating the error-resilience tools introduced in chapter 6, we provide experimental results based on common test conditions for conversational video and moderate–delay video. By making use of these results, we provide system design guidelines for video services in UMTS-like environments.

In **chapter 8**, mobile conversational video service being the most challenging in terms of delay is studied in more detail: For this purpose, we introduce a common channel and service environment to perform an in-depth comparison of different system and cross-layer design options. Initially, we concentrate on H.264/AVC-based source-channel coding system: We discuss system designs and system optimizations and provide experimental results for non-adaptive H.264/AVC single layer encoding. Advanced system designs are discussed and analyzed incorporating and combining features such as source-adaptive code rate selection, video feedback exploitation, H.264/AVC-based data partitioning, and UEP. We also introduce PTVC-based source-channel coding systems with video feedback: The discussed system designs includes the combination with FEC, with ARQ and limited retransmissions, as well as with channel-state information at the transmitter. All system designs are supported by appropriate rate and power allocation schemes and is finally compared by extensive simulations.

In **chapter 9**, we introduce a system suitable for video streaming over variable bit-rate mobile channels. For this purpose, we introduce an abstract framework for the description of mobile links. A similar framework is provided for the description of the pre-encoded video data. We propose an optimized packet scheduling and bitstream switching entity: The operation is based on the periodic update of necessary side information at the transmitter. Implementation aspects and complexity reduction methods for the proposed scheduler are discussed and assessed. Experimental results are provided to show the benefits of the proposed system.

Finally, **chapter 10** summarizes the the major findings of this work, highlights our major own contributions and provides an outlook for some potential subsequent studies.

Publications preceding this thesis

Parts of this thesis have already been published in [HS99], [HSWH99], [BS00], [SBPP00a], [SBPP00b], [HSWD00], [HSX00], [LSB00], [SBL00], [SWH00], [MS01], [BSM+01a], [BSM+01b], [RSL+01], [Sto01], [LSNB01], [SB01], [SM01], [SW01], [WS01], [SZ01], [SWS01], [WSH01], [WHS01], [WSDH01], [LS02], [SOW02], [LSS02a], [LSS02b], [LSSL02], [MS02a], [MS02b], [SW02], [SHW02], [SJW02b], [Sto02], [SJW02a], [SK02], [SWH02], [SWK02], [SWW02], [WS02], [Sto03], [SHW03], [SM03], [HHS03], [JSK03], [SWOO03], [SLJ+03b], [SLJ+03a], [BHS04], [BS04], [OBL+04], [JLSX04], [JSL04], [LJSB04], [SZ04], [JSX05b], [JLSX05], [JS05], [JSX05a], [SWL05], [WHS+05], [LJSB05b], [LJSB05a], [SH05], [JMSX05], [XGS+05], [ATGX06], [ASX+06], [GSAX06], [LSWS06], [LWG+06], [SHG+06], [JSX06a], [JSXAS06], [JSX06b], [GSX06], [JSX06c], [SGWS06], [Sto06], [SLW06], [SSXM06], [SGAS+06], [LGSW07], [SZ07], [SGS+07], [SLK+07], [VS07], [SSW07], [LWGS07], [LWSS07], [YASSZ07], [SJH+07], [YSS+07], [ZAX+07], [ZDS07], [SL07], [SWL08], [SLW08], and [SSW+08].

Supervised Diploma and Master Theses

The following diploma and master theses have been supervised during the course of this work: [Jak97], [Vla97], [Den98], [Ded98], [Gar99], [Ohl98], [Gri99], [Pie99], [Don99], [Per99], [Lie99], [Glo00], [Sch00], [Buc00], [Ned00], [Eff00], [Sol00], [Pur01], [Muf00], [Str01], [Hof01], [Lie01], [Aym01], [Yu01],[Kon01], [Jen01], [Oel01], [Str02], [Obe02], [Xu02], [Moz02], [Fin03], [Pfe03], [Red03], [Zha03], [Tse03], [Blo03], [Kar03], [Rei03], [Haa04], [Sch04], [Zha04b], [Kam04], [Bru04], [Zha04a], [Wal04], [Afz05], [Zhu05a], [Zhu05b], [You06].

2

Technical Preliminaries: Networked Video

2.1 System Overview

2.1.1 Video Transmission System

Multimedia transmission systems consist of several different components and protocols. To obtain insight into specific properties it is necessary to abstract, simplify, and model such systems. One has to be careful to include the essential components, but also to provide an abstraction level which allows to compare different concepts. A simplified system is shown in Figure 2.1.

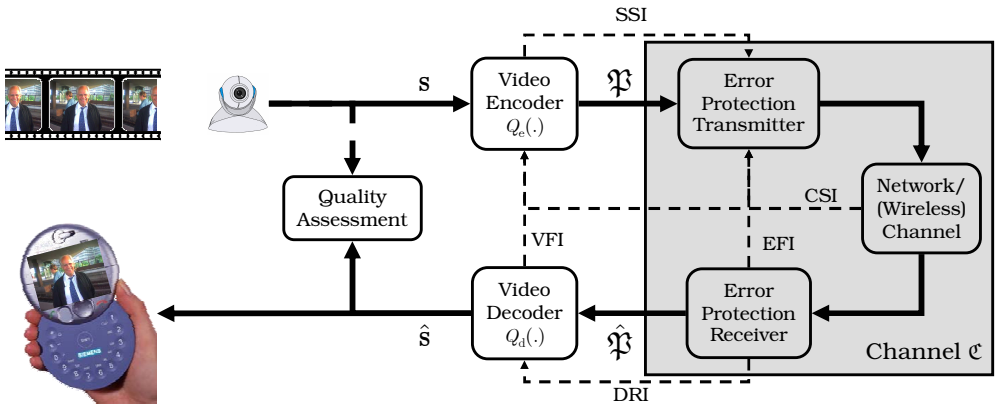


Figure 2.1: Lossy Video Transmission System.

Natural still images or video are usually generated by a video capturing device and are converted into a digital format to be stored, processed, or encoded by subsequent signal processing. For each discrete–time instant $t = 1, 2 \ldots, N_s$, common video capturing devices with appropriate pre–processing generate a two-dimensional discrete-space video signal s_t by projecting a 3-D scene onto the image plane. Cameras in the environments investigated in this work generate video frames

at certain constant[1] frame rate f_s measured in frames per second (fps). Usually the frame rate does not exceed a maximum of 30 fps for the applications and resolutions considered in this work, and commonly produce progressive–scan pictures in Common Intermediate Format (CIF) and Quarter Common Intermediate Format (QCIF) spatial resolution[2]. The sampling time $\tau_{s,t}$ for the sequence of video frames is usually specified as $\tau_{s,t} = t/f_s + \tau_{s,0}$, where $\tau_{s,0}$ is any arbitrary absolute time.

In the Red–Green–Blue (RGB) color system the generated two-dimensional source s_t consists of color component pixels $\{s^{(R)}_{i,j,t}, s^{(G)}_{i,j,t}, s^{(B)}_{i,j,t}\}$. However, the human eye is less responsive to color than to intensity. This is because the density of rods in the retina of the human eye is much lower than the density of cones. These rods are responsible for color perception in the eye. For this reason, the YUV[3] color system is often preferred over the RGB color system when storing and processing raw video data resulting in pixel components $\{s^{(Y)}_{i,j,t}, s^{(U)}_{i,j,t}, s^{(V)}_{i,j,t}\}$. This usage of YUV allows to have a lower resolution for representing the chrominance information of video frames. Some commonly used sub-sampling types are shown in Figure 2.2. The luminance samples are shown as circles, the chrominance samples are represented by a cross. The spatial separation of the chrominance samples is indicated by dashed lines. The 4:4:4 and 4:2:2 formats are almost exclusively used for high quality consumer applications and professional video, 4:1:1 is used in the DV format. Most commonly applied in modern video coding standards not targeting for very high quality applications is the 4:2:0 format and we exclusively use this format in the remainder of this work. In addition, each pixel component $s^{(i)}_{i,j,t}$ is pre–quantized to 8 bits precision such that pixel values are restricted to integer values from 0 to 255 for original pixel values.

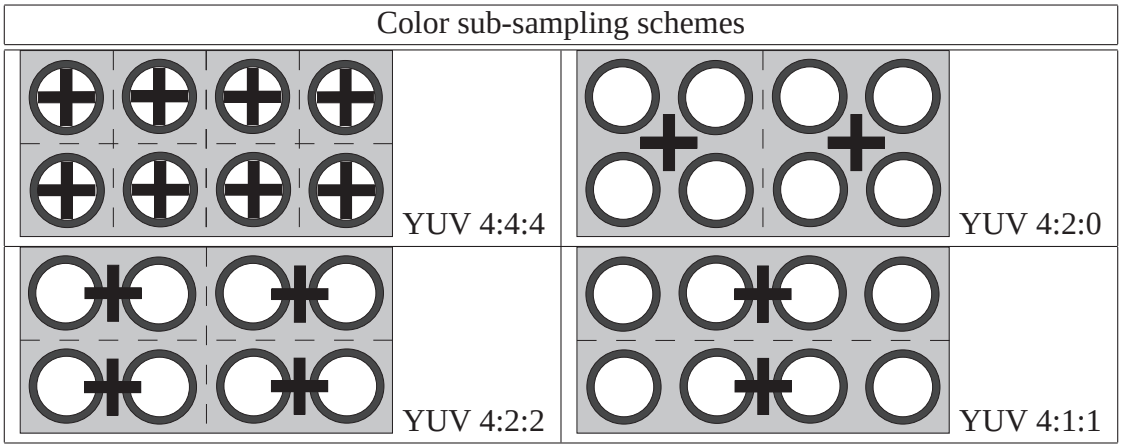


Figure 2.2: Some commonly used color sub-sampling schemes.

Although state–of–the–art video coders allow to process any spatial resolution with height, N_{height}, and width, N_{width}, being multiples of 16, we restrict ourselves to the two most common video formats for Internet and wireless applications, namely CIF and QCIF. The parameters are shown in Table 2.1. In addition to the little sensitivity of the human observer to colors, for most real-life video sequences, the color is a property of the objects that are visible in the sequence, and is tied to these (possibly moving) objects. Although we exclusively operate with colored images,

[1]The frame rate might be changed during the encoding of a video sequence by skipping frames generated by the camera.

[2]Different spatial resolutions are used in the common Internet and wireless applications, but for simplicity we restrict ourselves in the remainder of the thesis to these two most common spatial resolutions.

[3]The colorspace was originally known as YC_bC_r with 'Y' for luminance, or intensity 'U' or 'C_b' for blue chrominance value, and 'V' or 'C_r' for the red chrominance value. Green can be derived from the Y, U, and V values. For conversion between RGB and YUV, see e.g. [Jac96].

Table 2.1: Typical formats for raw video data.

Type	N_{width}	N_{height}	color $N_{\text{width}}/2$	color $N_{\text{height}}/2$	frame rate
QCIF	176	144	88	72	30 Hz
CIF	352	288	176	144	30 Hz

for ease of exposition and for sake of clarity, the video signal is regarded as the luminance signal only and we abbreviate $s_{i,j,t} \triangleq s^{(Y)}_{i,j,t}$ if not stated otherwise.

According to Figure 2.1 we assume that the *video encoder* $\mathcal{Q}_{\text{e}}(.)$ maps the video signal s onto a packet–stream $\mathcal{P}$, i.e.,

$$\mathcal{P} \triangleq \mathcal{Q}_{\text{e}}\left(s\right). \tag{2.1}$$

The packet–stream is defined by a sequence of packets, so called *data units*, $\mathcal{P} = \mathcal{P}_1, \mathcal{P}_2, \ldots$. After encoding, each data unit has a certain size $l_{\mathcal{P}i}$ in bits. We define a presentation time for each data unit as

$$\tau_{\text{PTS},i} \triangleq \tau_{s,t} + \tau_{\text{PTS},1}, \tag{2.2}$$

where $\tau_{\text{PTS},1}$ is any arbitrary offset and therefore set to 0 in the following. The t is the greatest time index of any source s_t contained by the data unit $\mathcal{P}_i$. A one–to–one mapping of a $\tau_{\text{PTS},i}$ to a data unit index i and vice versa cannot be assumed as a single discrete–time source unit s_t can possibly be coded into several data units or a data unit can contain more than one source units. For this purpose we introduce the greatest index i with sampling time $\tau_{\text{PTS},i} \leq \tau_{s,t}$ as i_t, i.e., $i_t = \max\{i : \tau_{\text{PTS},i} \leq \tau_{s,t}\}$. Then, the *sampling curve* or *encoding schedule* B_t is defined as the overall amount of data (e.g., measured in bits) produced by the video encoder up to time index t, i.e.

$$B_t \triangleq \sum_{j=1}^{i_t} l_{\mathcal{P}j}. \tag{2.3}$$

For convenience, we define the number of bits to encode frame s_t as $b_t \triangleq B_t - B_{t-1}$.

Video coding algorithms considered in this thesis are in general lossy, i.e., the reconstruction of the packet–stream $\mathcal{P}$ by the video decoder $\mathcal{Q}_{\text{d}}(.)$ results in a decoded version of the original source $\hat{s}$, i.e.,

$$\hat{s} \triangleq \mathcal{Q}_{\text{d}}\left(\mathcal{P}\right). \tag{2.4}$$

In the following, we assume that the decoded source has the same dimension as the encoded source and for each pixel $s_{i,j,t}$ of the original source we obtain a reconstructed pixel from the decoded source $\hat{s}_{i,j,t}$. The composition of encoder and decoder defines a general video coder $\mathcal{Q}(.)$ such that the decoded version can be generated by consecutively applying encoder and decoder, i.e.,

$$\mathcal{Q} : s \rightarrow \hat{s} = \mathcal{Q}_{\text{d}}\left(\mathcal{Q}_{\text{e}}\left(s\right)\right). \tag{2.5}$$

The packet–stream at the transmitter is forwarded to an entity referred to as *error–protection unit* to keep it very general. Error protection includes different mechanisms such as forward error correction using channel coding, backward error protection such as ARQ schemes, power control, interleaving schemes, any other methods to control and reduce the error probability of packets such as scheduling algorithms or prioritization, as well as combinations of these schemes. The corresponding unit at the receiver includes for example error detection algorithms and channel decoding algorithms.

In the scenarios investigated in this thesis the packet–stream is in general transmitted over a lossy channel. The channel in combination with the error protection unit might loose, corrupt, shorten, and/or delay each packet. The possible alterations for different transmission scenarios will be discussed in more details later in this work. However, let us assume that the channel $\mathcal{C}(.)$ including error protection schemes and physical layer attributes alters the transmitted packet–stream $\mathcal{P}$ to a received packet–stream $\hat{\mathcal{P}}$, i.e.,

$$\mathcal{C} : \mathcal{P} \rightarrow \hat{\mathcal{P}} = \mathcal{C}\left(\mathcal{P}\right). \tag{2.6}$$

At the receiver the possibly corrupted version of the packet–stream $\hat{\mathcal{P}}$ is decoded by the video decoder $\mathcal{Q}_{\mathrm{d}}(.)$ to obtain a reconstructed version $\tilde{s}$ of the original source s by recursively applying the encoder, channel, and decoder, i.e.,

$$\tilde{s} \triangleq \mathcal{Q}_{\mathrm{d}}\left(\mathcal{C}\left(\mathcal{Q}_{\mathrm{e}}\left(s\right)\right)\right). \tag{2.7}$$

Again we assume that the reconstructed source $\tilde{s}$ has the same dimension as the encoded source and for each pixel $s_{i,j,t}$ of the original source we obtain a reconstructed pixel from the decoded source $\tilde{s}_{i,j,t}$. Whenever appropriate, we denote the $\tilde{s}$ as a function of $\mathcal{C}$, i.e., $\tilde{s} \rightarrow \tilde{s}(\mathcal{C})$, to emphasize the dependency of the decoded source on the channel.

2.1.2 Video Quality Measures

Subjective Measures

The investigation, analysis, and assessment of video coding and transmission systems are based on performance measures. As the video is in general played back to a human observer this performance measure should be consistent with the perception of typical users. Basically two different types of quality measures can be used, objective or subjective ones. In case of subjective evaluation video sequences are presented to human observers who have to judge the overall quality, or a number of quality related criteria. For example, the International Telecommunications Union – Telecommunications Sector (ITU-T) [ITU82] provides recommendations for a test environment as well as subjective evaluation methodologies. A set of subjective tests for specific mobile purposes has been performed which is described in detail in [BHS04].

Objective Measures

However, subjective video quality assessments are in general expensive to perform as many subjects and long assessment sessions are necessary. Especially when a large parameter space must be investigated, subjective quality measurements become completely infeasible. Additionally, the operational control of the video encoder and rate allocation algorithms requires meaningful but easily computable performance measures. Hence, we will in the following almost exclusively consider objective quality measures being computed from the original and the reconstructed video signals.

In practice, the most common objective measure to evaluate video coding systems are based on the Mean Square Error (MSE). The MSE $d_t^{(\mathrm{mse})}$ for a single source frame s_t and one color component of a video sequence s reconstructed to $\hat{s}_t$ after encoding and decoding is defined as

$$d_t^{(\mathrm{mse})} \triangleq \frac{1}{N_{\mathrm{height}} N_{\mathrm{width}}} \sum_{j=1}^{N_{\mathrm{height}}} \sum_{i=1}^{N_{\mathrm{width}}} \left| s_{i,j,t} - \hat{s}_{i,j,t} \right|^2. \tag{2.8}$$

For convenience, whenever there is no ambiguity, we drop the label for the MSE and write d instead of $d^{(\text{mse})}$. The MSE d for each frame is commonly converted to the Peak Signal–to–Noise Ratio (PSNR) whereby the mapping for an 8 bit–precise original signal is defined as

$$\text{PSNR}\left(d\right) \triangleq 10\log_{10}\left(\frac{255^2}{d}\right). \tag{2.9}$$

For many cases, only the MSE and PSNR of the luminance component are of interest. Therefore, in the remainder of this work, objective quality measures for images and video sequences always refer to the luminance component. Despite some well–known short–comings of MSE–based measures, they still provide consistent results as long as the video signals compared are affected by the same type of impairment [Gir92]. Still, for the assessment and evaluation of video sequences consisting of N_s individual frames, a single measure is desired. For this purpose we define the average PSNR, $\overline{\text{PSNR}}$, as

$$\overline{\text{PSNR}} \triangleq \frac{1}{N_s}\sum_{t=1}^{N_s}\text{PSNR}\left(d_t\right). \tag{2.10}$$

The $\overline{\text{PSNR}}$ of the luminance component is the most widely accepted measure for visual distortion in the video coding community and has been used, e.g., in the standardization process of H.264/AVC.

For the transmission over error–prone channels also the effects of the channel have to be addressed. For the computation of a meaningful distortion measure as defined in (2.8) after the transmission of the video signal, the reconstruction at the encoder side, $\hat{s}$, should be replaced by the reconstructed signal at the receiver, $\tilde{s}(\mathcal{C})$. However, due to the usually probabilistic nature of the channel $\mathcal{C}$, the reconstruction value $\tilde{s}$ strongly depends on the realization of the channel $\mathcal{C}_n$. Therefore, we define the decoded MSE when applying this channel realization as

$$d_t\left(\mathcal{C}_n\right) \triangleq \frac{1}{N_{\text{height}}N_{\text{width}}}\sum_{j=1}^{N_{\text{height}}}\sum_{i=1}^{N_{\text{width}}}\left|s_{i,j,t}-\tilde{s}_{i,j,t}\left(\mathcal{C}_n\right)\right|^2. \tag{2.11}$$

To obtain a single measure for the performance of the entire system, the randomness of the channel $\mathcal{C}$ has to be taken into account. Therefore, we propose, similarly to the definition in (2.10), to use the expected average PSNR as an appropriate single measure for the decoded quality when operating on a random channel $\mathcal{C}$ as

$$\overline{\text{PSNR}}_{\mathcal{C}} \triangleq \mathbb{E}_{\mathcal{C}}\left\{\overline{\text{PSNR}}\right\} = \frac{1}{N_s}\sum_{t=1}^{N_s}\mathbb{E}_{\mathcal{C}}\left\{\text{PSNR}\left(d_t\left(\mathcal{C}\right)\right)\right\}. \tag{2.12}$$

It is worth to mention at this point that the single value measure average PSNR has some flaws especially when investigating the transmission over error–prone channels. Low image quality for single impaired video frames is often not sufficiently accounted. For this purpose, we introduce an additional measure where the expectation is performed in the MSE–domain rather than in the PSNR–domain, but for convenient representation the final value is transformed in the PSNR–domain. We define the PSNR of the averaged expected MSE as

$$\text{PSNR}_{\overline{\text{mse}}} \triangleq \text{PSNR}\left(\frac{1}{N_s}\sum_{t=1}^{N_s}\mathbb{E}_{\mathcal{C}}\left\{d_t\left(\mathcal{C}\right)\right\}\right) \geq \overline{\text{PSNR}}_{\mathcal{C}}. \tag{2.13}$$

Note that $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ is always greater equal $\overline{\mathrm{PSNR}}_{\mathcal{C}}$ as the function $\mathrm{PSNR}\,(d)$ as defined in (2.9) is a concave function. In this case, high distortions lead to a significant contribution and, therefore, bad frames are weighted much stronger, which is more appropriate for some scenarios.

In certain cases it is helpful to provide additional information. Therefore, with this background information, we view the MSE $d_t\,(\mathcal{C}_n)$ for each frame s_t and each channel realization $\mathcal{C}_n$ as defined in (2.11), as well as its corresponding PSNR as a random variable, denoted as PSNR_f as for example suggested in [SZ98].

Monte Carlo Methods for Quality Estimation

Due to the complex nature of the video coding, transmission, and decoding algorithms, the calculation or computation of the expected decoder performance when operating on a random channel $\mathcal{C}$ is in general infeasible. A widely accepted numerical method to estimate statistical parameters of complex systems, not only used in the engineering community, is the so–called Monte Carlo simulation method[4]. Assume $N_{\mathcal{C}}$ independent realizations $\mathcal{C}_n$ with $n = 1, \ldots, N_{\mathcal{C}}$ of the random variable channel $\mathcal{C}$. We can define the operational expected average PSNR when applying any sequence of independent realizations of length $N_{\mathcal{C}}$ as

$$\overline{\mathrm{PSNR}}^{(N_{\mathcal{C}})} \triangleq \frac{1}{N_{\mathcal{C}}} \sum_{n=1}^{N_{\mathcal{C}}} \left(\frac{1}{N_s} \sum_{t=1}^{N_s} \mathrm{PSNR}\,(d_t\,(\mathcal{C}_n)) \right). \tag{2.14}$$

Interestingly enough, the operational expected average PSNR defined in (2.14) are random variables by itself, as they depend on the selection of the channel realizations. However, by the weak law of large numbers [Pap91, pp. 69–71], the operational result in (2.14) converges to the expectation in (2.12) in probability as $N_{\mathcal{C}} \to \infty$. This method of repeating the transmission of one and the same source with $N_{\mathcal{C}}$ independent channel realizations provides a mean to estimate the actual value of the expectation.

Usually the amount of independent channel realizations $N_{\mathcal{C}}$ is finite due to computational constraints. Then, the algebraic mean of the outcome for $N_{\mathcal{C}}$ repetitions is less meaningful due to the randomness of the process. However, the probability that the deviation of the estimated mean $\overline{\mathrm{PSNR}}^{(N_{\mathcal{C}})}$ to the real mean $\overline{\mathrm{PSNR}}_{\mathcal{C}}$ is greater than some confidence interval ϵ with $\epsilon > 0$ is bounded by some confidence level β, i.e.,

$$\Pr\left\{ \left| \overline{\mathrm{PSNR}}^{(N_{\mathcal{C}})} - \overline{\mathrm{PSNR}}_{\mathcal{C}} \right| > \epsilon \right\} = 1 - \beta. \tag{2.15}$$

By the Chebyshev inequality [Pap91, pp. 149–151], the confidence level β can be lower–bounded as $\beta \geq 1 - \frac{\sigma^2_{\overline{\mathrm{PSNR}}_{\mathcal{C}}}}{\epsilon N_{\mathcal{C}}}$, where $\sigma^2_{\overline{\mathrm{PSNR}}_{\mathcal{C}}}$ denotes the variance of the real random variable. In [Rie97, Annex A], the replacement of the $\sigma^2_{\overline{\mathrm{PSNR}}_{\mathcal{C}}}$ by the variance estimated from $N_{\mathcal{C}}$ independent channel repetitions, $\sigma^2_{\overline{\mathrm{PSNR}}^{(N_{\mathcal{C}})}}$, is justified. In addition, it is shown in [Rie97, Annex A] that by an application of the central limit theorem [Pap91], the confidence level for a given confidence interval ϵ can be determined as

$$\beta \simeq \mathrm{erf}\left(\frac{\sqrt{N_{\mathcal{C}}}\epsilon}{\sigma_{\overline{\mathrm{PSNR}}_{\mathcal{C}}}\sqrt{2}} \right), \tag{2.16}$$

[4] It was named by Stanislaw Ulam [MU49], who in 1946 became the first mathematician to dignify this approach with a name, in honor of a relative having a propensity to gamble.

with $\mathrm{erf}(x)$ the error function[5]. Therefore, for a certain specified confidence level β upper and lower bounds on the real average can be provided as

$$\begin{aligned} \overline{\mathrm{PSNR}}_{l,\beta}^{(N_\mathcal{C})} &\triangleq \overline{\mathrm{PSNR}}^{(N_\mathcal{C})} - \frac{\sigma_{\overline{\mathrm{PSNR}}_\mathcal{C}}}{\sqrt{N_\mathcal{C}}}\alpha_\beta, \\ \overline{\mathrm{PSNR}}_{u,\beta}^{(N_\mathcal{C})} &\triangleq \overline{\mathrm{PSNR}}^{(N_\mathcal{C})} + \frac{\sigma_{\overline{\mathrm{PSNR}}_\mathcal{C}}}{\sqrt{N_\mathcal{C}}}\alpha_\beta, \end{aligned} \tag{2.17}$$

with $\alpha_\beta \triangleq \sqrt{2} \cdot \mathrm{erf}^{(-1)}(\beta)$ and $\mathrm{erf}^{(-1)}(x)$ being the inverse of the error function. Some typical values for α_β are: $\alpha_{0.9} = 1.645, \alpha_{0.95} = 1.960, \alpha_{0.99} = 2.576$.

In a similar manner, the operational average expected MSE when applying any sequence of independent realizations of length $N_\mathcal{C}$ can be determined as

$$\mathrm{PSNR}_{\overline{\mathrm{mse}}}^{(N_\mathcal{C})} \triangleq \mathrm{PSNR}\left(\frac{1}{N_\mathcal{C}}\sum_{n=1}^{N_\mathcal{C}} d\left(\mathcal{C}_n\right)\right) \quad \text{with} \quad d\left(\mathcal{C}_n\right) \triangleq \frac{1}{N_s}\sum_{t=1}^{N_s} d_t\left(\mathcal{C}_n\right). \tag{2.18}$$

The upper and lower bounds for the confidence intervals for a certain specified confidence level β for the PSNR of the expected average MSE can be specified as

$$\begin{aligned} \mathrm{PSNR}_{\overline{\mathrm{mse}},l,\beta}^{(N_\mathcal{C})} &\triangleq \mathrm{PSNR}\left(\frac{1}{N_\mathcal{C}}\sum_{n=1}^{N_\mathcal{C}} d\left(\mathcal{C}_n\right) + \frac{\sigma_{d(\mathcal{C})}}{\sqrt{N_\mathcal{C}}}\alpha_\beta\right), \\ \mathrm{PSNR}_{\overline{\mathrm{mse}},u,\beta}^{(N_\mathcal{C})} &\triangleq \mathrm{PSNR}\left(\frac{1}{N_\mathcal{C}}\sum_{n=1}^{N_\mathcal{C}} d\left(\mathcal{C}_n\right) - \frac{\sigma_{d(\mathcal{C})}}{\sqrt{N_\mathcal{C}}}\alpha_\beta\right), \end{aligned} \tag{2.19}$$

with $\sigma_{d(\mathcal{C})}$ being the standard deviation for the random variable $d(\mathcal{C})$. The cumulative distribution of PSNR_f can be estimated by a histogram.

Source Statistics

In some sense also the MSE for each frame s_t as defined in (2.8) and (2.11), or its corresponding PSNR might be viewed as the outcome of a random experiment. This viewpoint is for example appropriate, if the actual channel realization, $\mathcal{C}_n$, directly influences the encoding process, or if the source itself is viewed as a random process and the source frames or groups of frames represent realizations of a generic source model. However, in general, the source statistics change very slowly and cannot be assumed to be ergodic in the investigated time window. For this purpose we use a set of common test sequences which show some specific properties, but usually are generic enough such that the operational values of the PSNR as defined in (2.14) and (2.18) are appropriate measures and give a good indication on the performance of the system for a broad range of source signals. However, the confidence intervals as defined in (2.17) and (2.19) are no more meaningful in this case as the variance of the random variable not only addresses the variability of the experiment, but also the variability of the source statistics.

2.1.3 Information Flow

In addition to the down–stream data flow as indicated with bold arrows in Figure 2.1, additional information may be accessible to enable or enhance the transmission of video within networks. This

[5] $\mathrm{erf}(x) \triangleq \int_0^x e^{-z^2}\, dz$

was for example recognized and formalized by Hagenauer [Hag95] and extended to multimedia transmission in [HS99]. We will characterize this information in the following.

The video encoder passes so–called Source Significance Information (SSI) to the error protection scheme which allows to differentiate data with different Quality-of-Service (QoS) requests. This information may include for each packet $\mathcal{P}_i$ attributes such as importance specifying the amount by which the quality at the receiver increases[6] if some information is correctly decoded. A formal definition of the importance of a video packet is provided in section 3.6. The channel or network usually has some specific properties such as the bit-rate, loss and error rates, the expected channel Signal–to–Noise Ratios (SNRs), so–called Channel State Information (CSI). The system might be designed such that certain properties are maintained or statistics might be gathered by measurements. Commonly, this information can be used at the decoder to judge the reliability of the received data. In addition, if these measurements are conveyed also to the transmitter, this information can be used to design or adapt the video encoding and error protection scheme appropriately.

The receiver of the error protection scheme itself can provide information on the reliability of the decoded data, so–called Decoder Reliability Information (DRI). This includes information about the reliability of the packet content, e.g., Error Indication Flag (EIF), path reliability information (see section 4.4), or soft bits [Hag88]. Finally, the receiver can send feedback messages, e.g. the error protection receiver entity can inform the transmitter via Error–Protection Feedback Information (EFI) messages about the successful or non–successful reception of packets. In similar way, the video decoder can inform the video encoder using Video Feedback Information (VFI). The availability and the successful application of the different information depends on the specific application and the actual underlying transport system. The quality and accuracy depends on quality and the frequency of the measurements as well as on the amount of training data integrated in the media stream. Finally, another crucial aspect is that the information is available at the respective location in–time. These aspects will be discussed in more detail when presenting typical video applications in the Internet as well as in wireless environments.

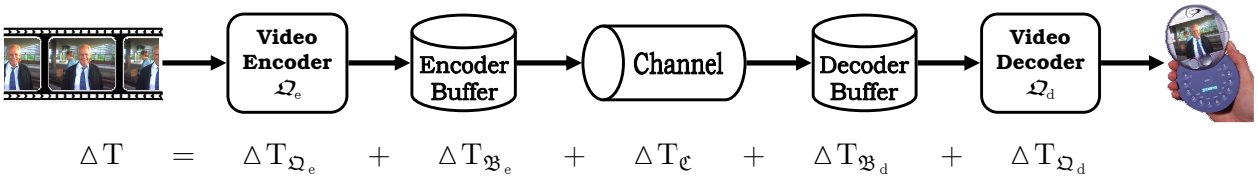


Figure 2.3: Buffering in Video Transmission.

2.1.4 Timing, Delays, and Buffering in Video Transmission

In contrast to still image transmission, video transmission imposes additional timing and delay constraints. This results in different system architectures and buffering concepts for different applications. In the following the buffers and delay components in a video transmission system are introduced according to the simplified system diagram in Figure 2.3. We assume that a certain video frame s_t is generated at sampling time $\tau_{s,t}$. The time interval between the sampling instant $\tau_{s,t}$ and the moment, that data unit $\mathcal{P}_{i_t}$ is generated, denoted as $\tau_{e,t}$, is defined as encoding delay

[6] This could also be expressed as the amount by which the distortion decreases.

$\Delta T^{(\mathcal{Q}_\mathrm{e})}{}_t \triangleq \tau_{e,t} - \tau_{s,t}$. The encoding delay is not necessarily constant and depends on the frame type. It consists of two parts: The processing delay depends on the encoder processor speed and the complexity of the encoding algorithms, but is ignored in the remainder. In contrast, the algorithmic delay component strictly depends on the encoding algorithm applied. For example, if backward predicted frames such as classical B–frames are used in the encoding process, the generation and encoding of the succeeding frame has to be awaited before the encoding of this B–frame is possible resulting in algorithmic delay. In addition, the encoder might decide to delay the encoding to obtain information about the statistics of future frames to be used in the rate control algorithm. The data units generated by the video encoder at encoding time $\tau_{e,t}$ are forwarded to an encoder buffer $\mathcal{B}_\mathrm{e}$ which stores the data units until they are forwarded to the channel at sending time $\tau_{x,t}$. The size of the encoder buffer can be as small as zero and as big as being able to store the entirely encoded video stream. The period of time the entire source unit s_t is contained in the encoder buffer is referred to as encoder buffer delay $\Delta T^{(\mathcal{B}_\mathrm{e})}{}_t \triangleq \tau_{x,t} - \tau_{e,t}$. Source unit s_t is completely received at the far–end at receiving time $\tau_{r,t}$ and the interval between receiving time and sending time defines the channel delay as $\Delta T^{(\mathcal{C})}{}_t \triangleq \tau_{r,t} - \tau_{x,t}$. The channel delay might consist of a constant part, $\Delta T_c^{(\mathcal{C})}$, and a random variable part, $\Delta T_v^{(\mathcal{C})}$. Before the received data units are forwarded to the video decoder at decoding time $\tau_{d,t}$ the decoder buffer $\mathcal{B}_\mathrm{d}$ delays them for certain time $\Delta T^{(\mathcal{B}_\mathrm{d})}{}_t \triangleq \tau_{d,t} - \tau_{r,t}$.

Finally, the video decoder processes the received data unit and releases it for display at presentation time $\tau_{p,t}$ after some decoding delay $\Delta T^{(\mathcal{Q}_\mathrm{d})}{}_t \triangleq \tau_{p,t} - \tau_{d,t}$. The decoding delay again consists of algorithmic and processing delay whereby the latter will again be neglected in the remainder. Buffers before the video encoder to store grabbed frames as well display buffers are not presented as they are of minor importance for the design of a video *communication* system. To summarize, the so called end–to–end delay ΔT_t representing the time between the sampling of a source unit and its presentation, i.e., $\Delta T_t \triangleq \tau_{p,t} - \tau_{s,t}$ is the sum of the individual delays, i.e.,

$$\forall_t \quad \Delta T_t \triangleq \Delta T^{(\mathcal{Q}_\mathrm{e})}{}_t + \Delta T^{(\mathcal{B}_\mathrm{e})}{}_t + \Delta T^{(\mathcal{C})}{}_t + \Delta T^{(\mathcal{B}_\mathrm{d})}{}_t + \Delta T^{(\mathcal{Q}_\mathrm{d})}{}_t. \tag{2.20}$$

To inform the receiver about the latest instant the decoding of a certain data unit $\mathcal{P}_i$ must have taken place and when to display the source unit contained by the data unit, timing information is included in each data unit in form of a Decoding Time Stamp (DTS), $\tau_{\mathrm{DTS},i}$, and a Presentation Time Stamp (PTS), $\tau_{\mathrm{PTS},i}$, respectively. The DTS also usually includes information about the decoding order of data units. Note that DTS and PTS are directly connected via the algorithmic decoding delay $\Delta T^{(\mathcal{Q}_\mathrm{d})}$, and, therefore the DTS can be defined as

$$\forall_i \quad \tau_{\mathrm{DTS},i} \triangleq \tau_{\mathrm{PTS},i} - \quad \Delta T^{(\mathcal{Q}_\mathrm{d})}{}_i. \tag{2.21}$$

A successful presentation in terms of continuous display requires that all discrete–time video frames are exactly presented by its PTS, i.e.,

$$\forall_t \quad \tau_{p,t} = \tau_{\mathrm{PTS},i_t} - \tau_{\mathrm{PTS},1}. \tag{2.22}$$

From (2.2) it is obvious that the video transmission system should take care to maintain a constant end–to–end delay $\Delta T_{\max}$ for each source unit s_t as $\Delta T_t = \Delta T_{\max}$.

To elaborated these connections in further detail, we discuss an exemplary video transmission system with significant practical impact as we will see later. Assume a simple error–free channel with a Constant Bit–Rate (CBR) R, measured for example in bit per seconds. In addition, assume that the channel constantly delays each single bit by some time $\Delta T_c^{(\mathcal{C})}$ and we do not use backward–predictive coding. Then, a typical example for the timing process of encoding, buffering, transmission, and decoding is shown in Figure 2.4. Without loss of generality we assume that

the sampling time instant $\tau_{s,t}$ of the first data unit is zero, i.e., $\tau_{s,1} = 0$. After some encoding delay $\Delta T^{(\mathcal{Q}_\mathrm{e})}$ the encoder produces $b_1 = B_1$ bits and forwards them to the encoder buffer $\mathcal{B}_\mathrm{e}$. Again, after some time $\Delta T^{(\mathcal{B}_\mathrm{e})}$, namely at transmission time $\tau_{x,1}$, the encoder buffer releases the first bit into the channel. The channel adds some constant channel delay $\Delta T_c^{(\mathcal{C})}$ and some variable channel delay $\Delta T_v^{(\mathcal{C})}$ until it is completely contained by the decoder buffer where it is stored for some time, $\Delta T^{(\mathcal{B}_\mathrm{d})}$. Note that the variable channel delay directly depends on the frame size as $\Delta T_{v,i}^{(\mathcal{C})} = b_i/R$. It is finally decoded at decoding time $\tau_{d,1}$ and immediately displayed as presentation time and decoding time are assumed to be identical, i.e., $\tau_{p,1} = \tau_{d,1}$. The process continues as shown in Figure 2.4: Whereas the sampling time, encoding time, and decoding time instants are equidistant, transmission time and receiving time instants depend on the source unit s_t, or more precisely, on the sampling curve B_t.

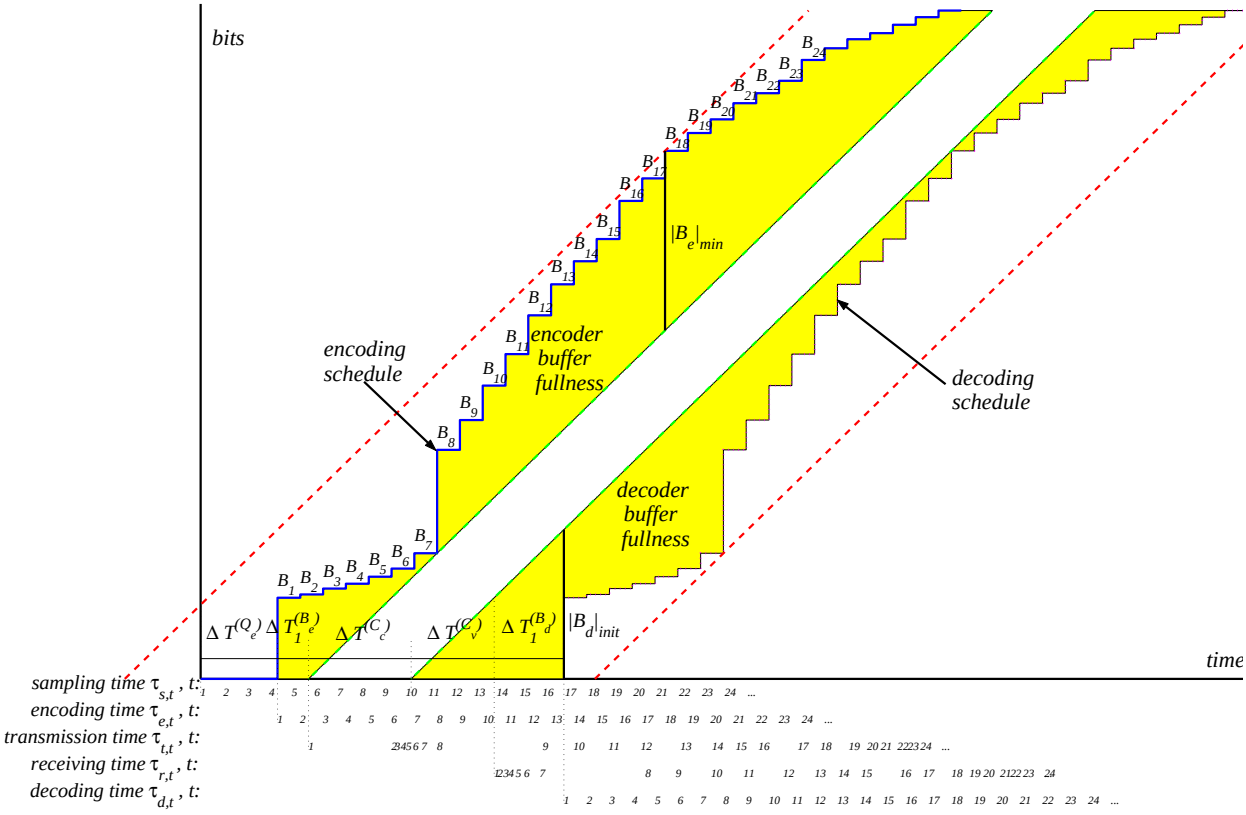


Figure 2.4: Timing process of encoding, buffering, transmission, and decoding when transmitting video over a CBR channel.

Once the transmission and decoding has started, it should be guaranteed that sufficient bits are available at the transmitter at any time instant t to avoid buffer underrun at the transmitter, i.e.,

$$\forall t \quad B_t \geq R \cdot \tau_{x,i_t}, \tag{2.23}$$

and that sufficient bits are available at the receiver such that continuous displaying can be guaranteed, i.e.,

$$\forall t \quad R \cdot \tau_{d,i_t} \geq B_t. \tag{2.24}$$

In Figure 2.4, the connection between the encoding schedule B_t, the initial encoder–buffer delay $\Delta T^{(\mathcal{B}_\mathrm{e})}{}_1$ and the initial decoder–buffer delay $\Delta T^{(\mathcal{B}_\mathrm{d})}{}_1$ are such that both conditions, (2.23) and

(2.24), are fulfilled, and the sum of the two delay components is minimal resulting in minimum end–to–end delay. In addition, in Figure 2.4 the connection between the minimum encoder buffer size $|\mathcal{B}_{\text{e}}|$, the transmission schedule $R \cdot \tau_{x,i_t}$, and the encoding schedule B_t is shown. The parameter tuple $(R, |\mathcal{B}_{\text{e}}|_{\min}, |\mathcal{B}_{\text{d}}|_{\text{init}})$ specifying the bit-rate of the channel, the minimum encoder buffer size, and the minimum initial decoder buffer fullness defined as $|\mathcal{B}_{\text{d}}|_{\text{init}} \triangleq |\mathcal{B}_{\text{e}}|_{\min} - R \cdot \Delta T^{(\mathcal{Q}_{\text{e}})}{}_1$, is referred to as leaky bucket and is metaphor of the encoder buffer $\mathcal{B}_{\text{e}}$. It is said that a leaky bucket model *contains* a bit–stream with encoding schedule B_t if there is no underflow or overflow of the encoder buffer according to the conditions, (2.23) and (2.24).

Although the leaky bucket describes the encoder buffer, it is also a way to inform the decoder on when to start decoding and what buffer size is necessary to decode the expected video stream. If the video stream enters the decoder buffer $\mathcal{B}_{\text{d}}$ with constant bit-rate R, the decoder buffer size $|\mathcal{B}_{\text{d}}|$ is at least as big as $|\mathcal{B}_{\text{e}}|_{\min}$, and decoding is started when the decoder buffer has initial fullness $|\mathcal{B}_{\text{d}}|_{\text{init}}$, then the successful decoding of the video stream can be guaranteed. Note that starting at fullness $|\mathcal{B}_{\text{d}}|_{\text{init}}$ is equivalent to start decoding with initial delay $\Delta T_{\text{init}} \triangleq |\mathcal{B}_{\text{d}}|_{\text{init}} / R$ after the first bit of the video stream enters the decoder buffer. With the specification of the decoding time $\tau_{d,1}$ and the presentation time $\tau_{d,1}$ of the first source unit s_1 and with the DTS and PTS for each source unit specified in the video stream the decoding time and the presentation time of each subsequent source unit s_t is specified as

$$\tau_{d,t} = \tau_{\text{DTS},i_t} - (\tau_{\text{DTS},1} - \tau_{d,1}), \tag{2.25}$$

$$\tau_{p,t} = \tau_{\text{PTS},i_t} - (\tau_{\text{PTS},1} - \tau_{p,1}), \tag{2.26}$$

respectively. As $\tau_{\text{DTS},1}$ and $\tau_{\text{PTS},1}$ are arbitrary number set by the encoder, we can assume without loss of generality, that $\tau_{\text{DTS},1} = \tau_{d,1}$ and $\tau_{\text{PTS},1} = \tau_{p,1}$.

2.1.5 Video Applications and Services: Requirements and Objectives

Whenever video transmission is discussed it is essential to specify the environment the video will be used in. It is important to understand that different applications impose different requirements and focus on different objectives thus requiring different strategies in the encoding, transmission, and decoding process. Basically all applications in common is the necessity for *compression efficiency* imposed by resource and rate constraints on the channel. For example, the costs when using a mobile systems are expected to be proportional to the reserved bit-rate or the number of transmitted bits over the radio link. Also, congestion avoidance algorithms in shared wired networks impose rate constraints, and the memory capacity on storage devices is still limited. However, in addition to compression efficiency, other objectives and goals have to be addressed. We will focus on mobile video applications in the following although most issues discussed also hold for the transmission of video over wireline networks. For example, major service categories for state–of–the–art video were identified in the H.264/AVC standardization process [WHS01]. These services will be briefly presented in the following together with their requirements and objectives and objectives.

Table 2.2 summarizes some characteristics of different applications. Rough numbers on the maximum end–to–end delay are provided, the availability and usefulness of different feedback messages is indicated, the availability of CSI at the transmitter is addressed, and finally it is distinguished if the video is offline encoded and pre–stored or if online encoding is performed. More details are discussed in the following.

Table 2.2: Video Application Characteristics.

Application	maximum delay	VFI		EFI		CSI	online encoding
		avail.	useful	avail.	useful		
Download–and–Play	not def.	no	no	yes	yes	yes	no
On–demand Streaming	$1-5$ s	limited	partly	yes	yes	yes	no
Live Streaming	$0.2-5$ s	yes	partly	yes	yes	yes	yes
Multicast/Broadcast	$1-5$ s	no	no	no	no	partly	yes/no
Video Telephony	250 ms	yes	yes	partly	limited	yes	yes
Video Conferencing	300 ms	no	no	no	no	partly	yes

Download–and–Play Applications Common download–and–play applications include services such as video download in the Internet or Multimedia Messaging Service (MMS) for mobile clients [MMS02]. For these applications encoding, transmission, and decoding of the video are strictly separated. A video sequence is encoded and stored entirely at a *multimedia server*[7] before it is transmitted , i.e., $\tau_{e,N_s} < \tau_{x,1}$. A receiver can request a certain content and downloads it to the decoder buffer realized as a local storage device. File downloads are performed via reliable links which might result in significant channel delays. In addition, both, the encoder buffer delay and the decoder buffer delay are at least the duration of the video clip. However, as sequence play–back does not start until the end of the reception of the entire video sequence, i.e., $\tau_{x,N_s} < \tau_{d,1}$, error–free transmission and continuous playback according to (2.22) can be guaranteed as the DTS and PTS for each data unit are signalled to the decoder.

Streaming Applications In on–demand streaming applications, a streaming server $\mathcal{Q}_\text{s}$ stores pre–encoded content in form of data units $\mathcal{P}$ just as in the case of download–and–play applications. However, instead of separate download and playback, in streaming applications the decoding of already received data units as well as the display of the contained source units is started while still downloading other data units. In uni–directional streaming applications, usually a feedback link from the receiver to server is present which allows to provide the transmitter with VFI, EFI, and CSI. However, note that this information cannot be used in the video encoding process as the video has been encoded offline.

Assume that the encoder buffer is sufficiently large to store the entire video, and the transmission of the video is started after encoding is completed, i.e., $\tau_{x,1} \geq \tau_{e,N_s}$. The receiver side of Figure 2.4 explains the procedure when streaming off-line encoded video. After the server has received a request from a client it starts transmitting the first source unit s_1 at time instant $\tau_{x,1}$. The following source units s_t with $t = 2, 3, \ldots$ are equivalently transmitted at time instants $\tau_{x,t}$. By combining (2.22), (2.20), (2.25), and (2.26), we can conclude that completely successful streaming without delayed decoding of data units is possible if

$$\forall_i \quad \tau_{x,i} + \Delta T^{(\mathcal{C})}{}_i \leq \tau_{\text{DTS},i}, \tag{2.27}$$

with the assumption $\tau_{\text{DTS},1} = \tau_{d,1}$. The equality in (2.22) is achieved by using the decoder buffer $\mathcal{B}_\text{d}$ for de–jittering and storing the data unit in the receiver buffer for some time $\Delta T^{(\mathcal{Q}_\text{d})}{}_i = \tau_{\text{DTS},i} - \tau_{r,i}$. Whereas in the *Timestamp-Based Streaming (TBS)* case the server forwards data unit $\mathcal{P}_n$ exactly at

[7] The server can be viewed as the encoder buffer in our model.

time $\tau_{x,n}$ to the network, advanced streaming servers allow for *Ahead-of-Time Streaming (ATS)*[8]. In this case the sending time $\tau_{x,i}$ is a nominal sending reflecting the latest time instant the data unit should be forwarded to the network. However, the server can possibly transmit data unit $\mathcal{P}_i$ ahead of time, i.e., before nominal sending time $\tau_{x,i}$. According to (2.27), this strategy allows to compensate longer channel delays $\Delta T^{(\mathcal{C})}{}_i$ but also requires that the decoder buffer is sufficiently large, and that the network and intermediate buffers can support higher instantaneous data rates.

When transmitting pre–encoded content, the sampling curve $\{B_t\}$ is a priori specified. If the video stream enters the decoder buffer $\mathcal{B}_\mathrm{d}$ with constant bit-rate R, then the leaky bucket parameters minimum encoder buffer size $|\mathcal{B}_\mathrm{e}|_\mathrm{min}$ and initial decoder buffer fullness $|\mathcal{B}_\mathrm{d}|_\mathrm{init}$ can be determined according to Figure 2.4 such that that the encoded video is contained in this leaky bucket, i.e., successful play–out without decoder buffer underrun or overflow is possible. For more details we also refer to [LOR98], [PS01] or [RCCR03].

In case of live streaming, the system according to Figure 2.3 is directly applicable. Again, in one–to–one unidirectional streaming applications, a feedback link from the receiver to server can be expected enabling the transport of VFI, EFI, and CSI. The timing and delay constraints are directly explained by Figure 2.4. With $\tau_{s,1} = 0$ and $\tau_{\mathrm{DTS},1} = \tau_{d,1}$ as well as sufficiently large buffers, successful streaming without delayed data units is possible if

$$\forall i \quad \tau_{s,i} + \Delta T^{(\mathcal{Q}_\mathrm{e})}{}_i + \Delta T^{(\mathcal{B}_\mathrm{e})}{}_i + \Delta T^{(\mathcal{C})}{}_i \leq \tau_{\mathrm{DTS},i}. \tag{2.28}$$

The decoder buffer $\mathcal{B}_\mathrm{d}$ is again used for de–jittering. However, in this case the sampling curve B_t is not known a priori. Therefore, the leaky bucket model is completely specified prior to encoding, transmission, and decoding. The video has to be encoded such that the sampling curve $\{B_t\}$ is fully contained by the specified leaky bucket.

For many application it is desired to keep the initial delay ΔT_init or the maximum end–to–end delay ΔT_max as low as possible, but still maintaining sufficient video quality. Reasons why this delay should be low are for example: (i) Significant startup delay is annoying to the end user as and if the video does not playback after a certain time the end user might even stop the playback as he assumes that the network is starving. (ii) In interactive applications where the user desires to react and modify the stream long initial delays are disturbing: For example, if switching to a different channel takes too long, the user might miss important information. (iii) In time–critical surveillance applications timely reaction to a certain event might be essential. In general, in streaming applications sufficient compression efficiency is desired under the constraint that the the initial delay ΔT_init is bounded. Typically, the delays should not exceed 1 to 5 seconds.

In multicast and broadcast applications, the same video stream is distributed to several and many users, respectively. This imposes similar constraints as discussed for live or on–demand streaming. However, the conditions in (2.27) and (2.28), respectively, have to be fulfilled not just for each data unit but also for each participating user. Multicast and especially broadcast applications do not allow feedback messages, and therefore, VFI and EFI messages cannot be expected at the transmitter.

Conversational Applications Conversational applications are usually characterized by the constraint of low end–to–end and roundtrip delay. The roundtrip delay in speech communication should not exceed about 500 ms requiring an end–to–to end delay of at most 250 ms [G1196]. As for sufficient quality video telephony or video conferencing applications lip synchronicity should be guaranteed the same constraints hold for the video. Therefore, a main objective in the system

[8] for example nowadays applied by YouTube, see http://www.youtube.com.

design for conversational applications is the minimization of the delay components encoding delay $\Delta T^{(\mathcal{Q}_e)}i$, encoder buffer delay $\Delta T^{(\mathcal{B}_e)}i$, and channel delay $\Delta T^{(\mathcal{C})}i$. These constraints obviously limit the application of several algorithms in video encoding process as well as for error protection. The restrictions usually result in reduced compression efficiency as well as unavoidable transmission errors imposing interesting challenges. The positive effect about bi–directional one–to–one video telephony is the availability of a fast feedback channel which allows to transport accurate CSI and timely VFI. These messages can be used in the encoding process as online encoding is performed in conversational video. In video conferencing systems where the encoded signal is in general distributed to multiple participants, the availability of feedback information is practically impossible. However, one might have access to at least some information regarding the channel characteristics the users experience over a longer range of time.

2.2 Transmission Environment

2.2.1 Concepts

The preliminaries presented in the previous section provided an abstract description of the transmission environment as system which alters the packet stream $\mathcal{P}$. In the following we will introduce state–of–the–art transmission systems which allow to transmit multimedia data with different QoS requirements. We will focus on packet–switched transmission in mobile systems, but most concepts are also applicable to wireline multimedia transmission.

The transmission of multimedia content over different networks to mobile users is an ongoing challenging task due to the complexity of the application data, the heterogeneity and imperfectness of physical transport media, and the switching and routing necessary for world–wide delivery. This has led to a uniquely accepted structuring of this complex task by the introduction of *protocol layers* which on the one hand allow clarifying the concepts and, on the other hand, reduces the implementation complexity. Thereby, the data on the sender side is processed such that layer n takes blocks of data called *packets* from layer $n+1$, adds a *header*, and passes them to layer $n-1$. The receiver reverts the operations with layer n receiving a packet from layer $n-1$, stripping and processing the layer–n header, and passing the *payload* to layer $n+1$.

Although circuit–switched connection–oriented transmission is still widely spread especially for voice data, the main distribution network of multimedia content will be the Internet. The Internet is a global collection of autonomously administered packet networks to form one virtual network which allows all part to communicate with each other via the Internet Protocol [Pos81] (IP). The Internet layering concept [Sch01] according to Figure 2.5 is aligned with the classical seven–layer concept according to Open Systems Interconnection [Zim80] (OSI) except that presentation and session layer are not represented in the Internet. Within the Internet architecture, the task of each layer will be roughly defined in the following. The *physical layer* provides a point–to–point or point–to–multipoint bit transport service over wires, optical fires, or the air. The *link layer* provides a point–to–point or point–to–multipoint packet service possibly including error detection and retransmission of lost or erroneous packets. The *network layer* carries packets end to end across multiple subnets which are connected by *routers*. Routers primarily forward network–layer packets and connect subnets of the same, but also of different technologies resulting in highly heterogeneous end–to–end networks. The path of the packet is determined by the routing protocol which is in case of the Internet architecture exclusively the IP. The *transport layer* as well as all layers above basically only operate within the communication end points. The Internet architecture is based primarily on the User Datagram Protocol [Pos80b] (UDP) for unreliable service

and the Transmission Control Protocol [Pos80a, SK91] (TCP) supplying reliable byte–stream service. Internet applications usually have an additional protocol layer to support a limited set of applications such as the Hypertext Transfer Protocol [FGM+97] (HTTP) for information retrieval in the World-Wide Web (WWW), or the File Transfer Protocol [PR85] (FTP). More details on multimedia transmission will be discussed in subsection 2.2.3.

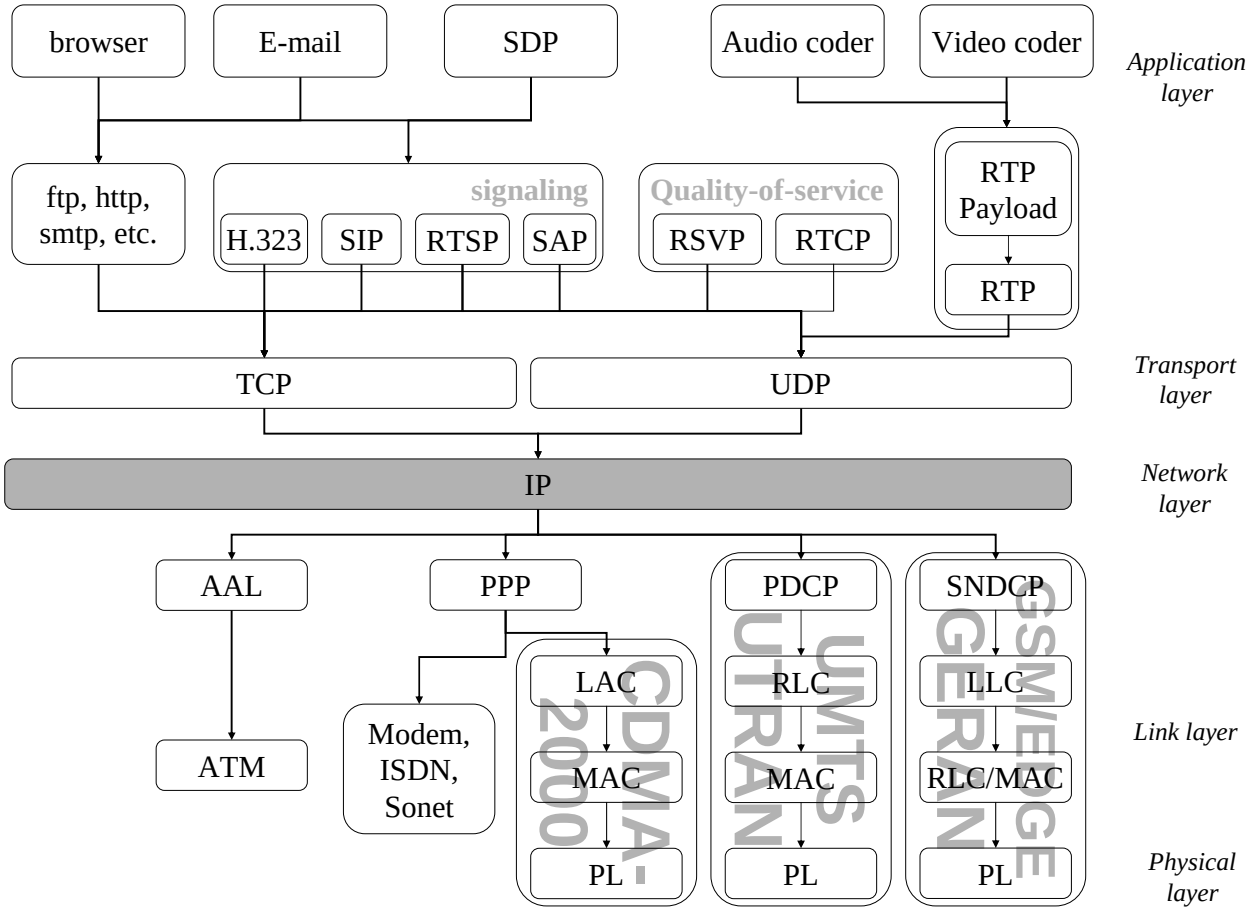


Figure 2.5: Internet Multimedia Protocol Stack.

2.2.2 Internet Principles and Characteristics

One key to the success of the Internet is the simplicity of the network–layer service model: each network packet is logically independent of all others – a principle known as *datagram* or *connection–less* delivery. Packets are transmitted on a best–effort basis, i.e., the network provides no guarantees if or when a packet will arrive at the destination. This minimal service model can be supported by just any link–layer technology and follows the end–to–end architectural principle [SRC84]. However, this principle may make better service available from the lower layers invisible to the Internet applications. This issue will be discussed in more detail in section 2.3.1.

The best–effort principle of the Internet results in two major QoS impairments: packet delays and packet losses. The end–to–end network delay is the time elapsed between sending and receiving a particular packet of size $l_{\mathcal{P}}$ and consists of several components. Whereas *propagation delays* for cable and wireless transmission in the range of 3.3-5 μs/km are negligible, *transmission delays*

can have significant influence on the end–to–end delay. For an uncongested multihop network with N_L links, each link with bit-rate R_i, a packet of size $l_\mathcal{P}$ experiences a transmission delay

$$\delta_{\text{trans}} = l_\mathcal{P} \sum_{i=1}^{N_L} \frac{1}{R_i}. \tag{2.29}$$

Whereas for fibers and cables even over multiple hops this impairment is again negligible, for low bit-rate modems or wireless links this can already add significant delays. In addition, some types of links add *algorithmic delays* caused by interleaving in combination with forward error correction. In addition to this static delay components, variable delays are caused by resource contention and link-layer retransmissions. The former results in a *queueing delay* if the number of packets arriving for a particular link temporarily exceeds the capacity of the outgoing link. Link–layer retransmissions are common for modem links [Goo99] as well as many wireless links, more details will be discussed in section 2.4 and chapter 4. The second major impairment are packet losses caused by different phenomena. In networks, packets are dropped when router queues overflow. In addition, corrupted packets detected by a link–layer or the IP header checksum, are discarded.

The measurement of performance of the Internet is quite difficult as the characteristics of packet losses and delays vary significantly over time, location, different access link technologies and connections. Losses are typically in the range of 1-2%, but might go up to 20% [Sch01]. Reliability in the Internet is achieved by TCP–based transmission. TCP provides retransmission of lost and erroneous packets and flow control to prevent a fast sender from overrunning a slow receiver. Therefore, the sender is allowed transmitting a window of data packets before receiving a confirmation from the receiver. The window size is dynamically adapted to the receiver buffer space and the network congestion. The *TCP congestion control mechanism* [APS99] is the principal means of limiting Internet traffic without explicitly configuring the sender application. However, this mechanism poses problems for real–time multimedia applications. Using TCP, the bit-rate is controlled by the network, not by the applications resulting in significant variations in available bit-rate on short time–scales.

Therefore, many real–time applications rely on the unreliable data service of UDP. These applications have in common a preference for in–time delivery over reliability. The UDP header of 8 bytes in size contains a checksum over the entire UDP packet, which can be used to detect and remove packets containing bit errors. Apart from this, UDP offers the same best effort service as IP does. In contrast to TCP, UDP does not provide a congestion control mechanism. Basically, applications should implement their own congestion control mechanisms but are not forced to. However, since UDP traffic volume was small relative to congestion–controlled TCP flows in former days of the Internet, the network did not break–down. With the growth of real–time applications on the Internet over recent years the situation is about to change. Therefore, an effort has been launched within the Internet Engineering Task Force (IETF) to design a new protocol under the acronym Dynamic Congestion Control Protocol [KHF04] (DCCP) that combines unreliable datagram delivery with built–in congestion control on the transport layer. In the future DCCP may replace UDP.

2.2.3 Multimedia Transmission over IP

The popularity of IP-based multimedia services over the Internet is growing with, e.g. hundreds of new subscribers registered daily for video streaming services and thousands of new participants in Voice-over-IP calls. Many new advances are discussed, but most of the real-time data transmission

relies on common principles and protocols. Therefore, it is important to understand existing solutions, and to exploit the advantages and drawbacks of those before new solutions are introduced. In the following, we will provide a brief overview on state-of-the-art multimedia transmission using IP.

To transport continuous media, three types of protocols are used as shown in Figure 2.5: media transport, QoS related protocols, and signaling protocols. For QoS issues, on the one hand protocols for measuring the end-to-end performance are available, e.g. Real–Time Control Protocol (RTCP) as part of Real–Time Transport Protocol [SCFJ03] (RTP). On the other hand, there exist protocols which allow to prioritize certain flows or certain packets under the acronyms *integrated services* and *differentiated services*, respectively [GS98]. However, the latter approaches are not very wide–spread due to significant increased complexity in intermediate routers. So for many applications and most Internet users, the best–effort networks is still the only accessible network for real-time multimedia transmission.

While media transport and QoS components are the same for continuous–media applications, signaling differentiates according to the application in use. For media-on-demand and streaming applications, Real–Time Streaming Protocol [SRL98] (RTSP) provides a control for multimedia streams. The protocol consists of requests issued by the client and responses returned by the server. Basically, it acts as a remote control on the media stream providing methods such as *play*, *pause*, or *record*. For Internet telephony, Session Initiation Protocol [HSSR99] (SIP) and H.323 [H3298] have been developed which both provide the ability of one party to signal to one or more parties the initiation of a new call being an association between a number of participants. The association between a pair of participants, a so-called connection, only consists of the signaling state at the two end points, not a physical channel as common for circuit-switched connections. Both protocols, SIP and H.323, provide means for user location, feature negotiation, call participant management, and feature changes. While RTSP and SIP are defined for one-on-one sessions, the Session Announcement Protocol [HPW00] (SAP) provides means to announce and join Internet broadcast sessions. All three applications require a mechanism to describe the streams within a multimedia session. The Session Description Protocol [HJ98] (SDP) is the most prominent description protocol which can be carried in any protocol, e.g. SIP, RTSP, or SAP.

Although essential for appropriate setup and control of multimedia sessions, in the remainder of the work signaling issues for different multimedia applications are of less importance. It is assumed that a one-on-one connection is established with appropriate features negotiated. More interest is put into the transport of real–time flows which is commonly based on RTP. Some common requirements of real–time audio and video streams which distinguish them from traditional Internet services are addressed in the RTP specification. For this purpose, this specification defines among others a fixed header format consisting of twelve bytes and a variable length media payload as shown in Figure 2.6. The RTP header indicates the protocol version (V), a possible header extension (X), optional padding of the payload (P), and multicast related issues with providing source identifiers. In addition, RTP addresses the following issues:

Sequencing If packets arrive out-of-order, they are reordered. Additionally, lost packets must be detected. For this purpose, RTP specifies a 16-bit sequence number.

Synchronization Firstly, the playout or display time of each media packet relatively to the previous ones has to be specified. Secondly, if different media are used in one session, synchronization must be established among them, e.g. to achieve lip-synchronicity. Therefore, RTP provides a 32-bit timestamp.

Payload Identification This addresses the possibility to change the encoding for the media on

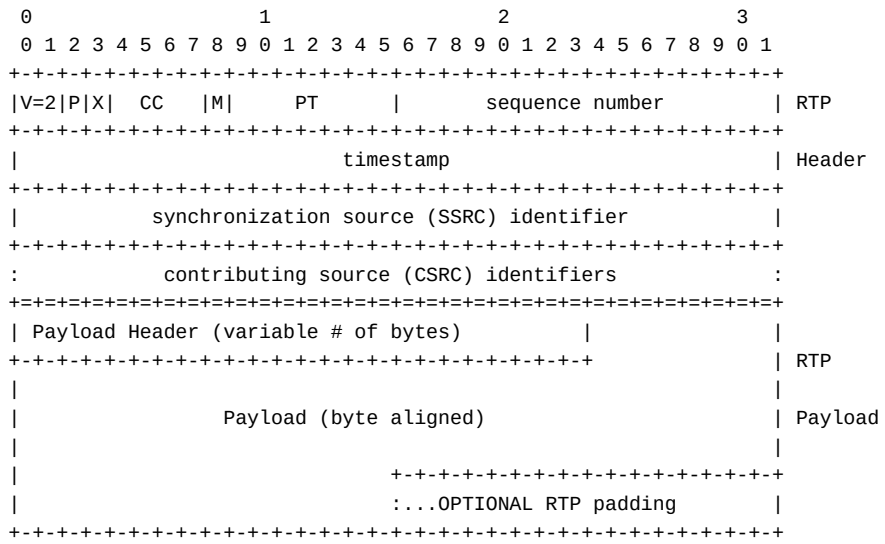


Figure 2.6: RTP header format.

the fly, for example, to adjust bandwidth variations. A 7-bit payload type indicator field is provided where static payload types are specified in the RTP profile document [SS03]. For example, payload type 0 designates the μ-law audio codec. Due to limited space, no further static payload type assignments are made, but dynamic payload types are between 96 and 127 are negotiated using H.323 or SDP allowing for RTP to be used with any kind of media codec developed in the future.

Frame Identification Audio and video are sent in logical units, so called frames. The marker bit (M) allows to signal the end of such a logical unit, if this unit is transported in several RTP packets.

RTP payload formats have been defined by the Audio/Video Transport (AVT)[9] group of the IETF for, among others, different audio codecs and many video codecs such as Joint Photographic Expert Group (JPEG) [BFF+98], H.261 [TH96], H.263 [Zhu97], H.263+ [BCD+98], Moving Pictures Expert Group (MPEG)–1 and MPEG–2 [HFGC98], and H.264/Advanced Video Coding (AVC) [WHS+05]. Furthermore, RTP payload formats are defined to provide generic services. For example, one payload format is defined for simple parity FEC to allow recovering lost packets independently of the codec in use [RS99].

RTP is accompanied by the RTCP providing control and management messages. It is recommended that the fraction of the session overhead added for RTCP be fixed at 5%. Media senders generate *sender reports* describing the amount of data sent so far as well as information to enable synchronization among different media streams. Media receivers send *receiver reports* including instantaneous and cumulative loss rates as well as delay and jitter information. In addition, each application can specify application-specific information to be carried over RTCP. Just recently, RTCP has been extended [FCC03] with the *extended report* packet type allowing to transport information about each RTP packet. For example, a run-length coded version indicating the loss or reception of each RTP packet can be indicated from the receiver to the sender. However, it is not guaranteed that this feedback is timely feedback that would allow a sender to repair the media

[9] for details, see http://www.ietf.org/html.charters/avt-charter.html

stream immediately. Recent efforts within the AVT [OWS+04] will allow providing timely and fast feedback especially for point-to-point scenarios giving senders new possibilities to react to packet losses. A small number of general-purpose feedback messages as well as a format for codec and application-specific feedback information are defined for transmission in the RTCP payloads.

2.3 Multimedia Transmission in Packet-Radio Networks

2.3.1 Background and History

Multimedia transmission in packet–radio networks nowadays relies on data services in existing wireless systems, a brief overview on these services in the European development is provided in the following. The history of wireless systems with main focus to data services is shown in Figure 2.7 together with the North American path based on International Standard (IS)–95. In addition to voice and fax services, as well as Short Messaging Services (SMSs) the initial Global System for Mobile Communication (GSM) system supports basic Circuit Switched Data (CSD) services up to 9.6 kbit/s[10] which allows a user to use their wireless handset as a modem for laptops and other electronic devices via infrared ports or designated data cables. In this case the mobile user establishes an end–to–end connection with Public Switched Telephone Network (PSTN) or Integrated Services Digital Network (ISDN) users applying access methods and protocols, such as X.25 or X.32. The idea is basically the same as for modem links, namely that the data service is on top of the a voice channel. Then, the resources on the air interface as well as in the access network are assigned to the user until the connection is terminated, a typical characteristic of *dedicated channels*. The data link provided to the application and characterized by certain properties, e.g. data rates and delay, as well as QoS attributes, e.g., error rates, is usually referred to as *bearer*.

The shortcomings of the data services in the initial design of second–generation (2G) wireless networks with respect to data rates and packet–friendliness started several new initiatives. On the one hand, the 2G systems were enhanced to support additional services and higher data rates for individual users, on the other hand the idea of the development of third–generation (3G) systems was created to introduce completely new applications for mobile users. With the GSM phase 2 project the CSD services were enhanced with High–Speed Circuit Switched Data (HSCSD) services allowing wireless data to be transmitted at $4 \times 9.6 = 38.4$ kbit/s[11] in both directions by allocating up to four radio slots to a single user.

Whereas the introduction of HSCSD was basically possible with software updates in the mobile system, the enhancements of 2G networks to support packet–radio services required significant changes: General Packet Radio Service (GPRS) consists of a packet wireless access network providing peak data rates up to $8 \times 21.4 = 171.2$ kbit/s and an IP–based backbone providing access to the public Internet. To accomplish this, the legacy GSM architecture has been extended with the Serving GPRS support node (SGSN) substituting the Mobile Switching Center (MSC) in GPRS and the Gateway GPRS support node (GGSN) providing interworking with external packet-switched networks according to Figure 2.8.

The data transmission plane used in GPRS is shown in Figure 2.9. Between the SGSN and the mobile station, the Sub-Network Dependent Convergence Protocol (SNDCP) adapts the upper layer protocols to the functionality of the underlying GPRS layers. The data link layer encompasses three sublayers: The Logical Link Control (LLC) layer establishes a logical link between mobile

[10] this was later extended to 14.4 kbit/s

[11] later 4×14.4 kbit/s have been made possible

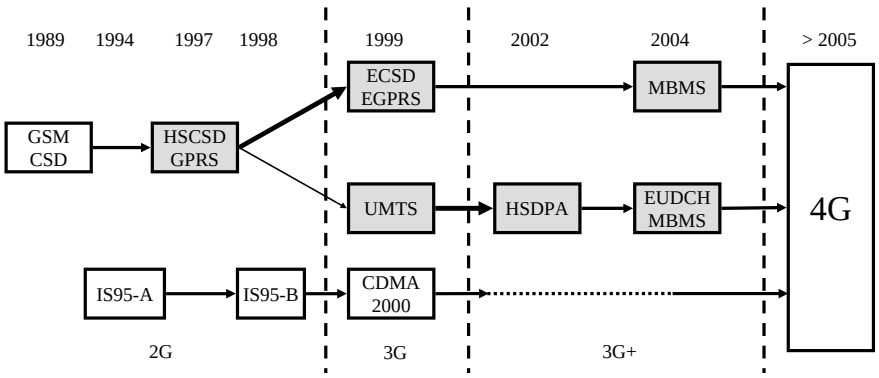


Figure 2.7: Timeline for European and North American mobile systems with focus on data services.

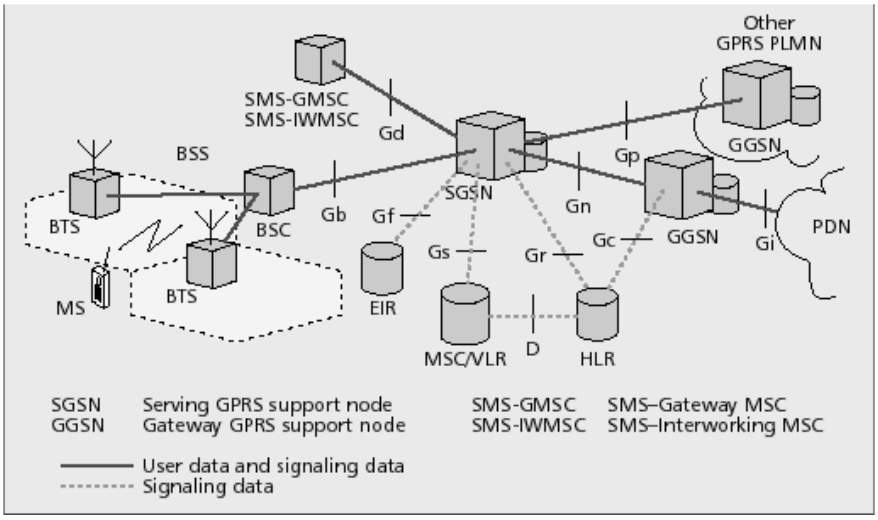


Figure 2.8: GPRS logical architecture[EVB01].

station and SGSN. The LLC is designed independently of the underlying radio network to simplify the introduction of alternative GPRS radio solution. Between the Base Station System (BSS) and the Mobile Station (MS), the Radio Link Control (RLC) protocol provides a reliable link dependent on the radio solution. For example, the RLC layer performs segmentation of the LLC–Packet Data Unit (PDU) into short blocks of fixed length according to one out of four Coding Schemes (CSs). A procedure called link adaptation can be applied to dynamically switch between the CS after every RLC block. This allows adapting the level of error protection and throughput to the channel characteristics. A block check sequence is appended, which in combination with sequence numbering allows the detection of erroneous or lost segments. Furthermore, the RLC layer provides optional retransmission to achieve a reliable data transfer when needed. Finally, the Medium Access Control (MAC) controls the access procedure for the radio channel by performing multiplexing of user data and signaling information. In contrast to dedicated channels where the resources are assigned to the user for the life–time of the connection, GPRS relies on the concept of *shared channels* just as wired Internet transmission. However, in contrast to the multiple access scenario on wired links, the situation in a wireless environment is more difficult as the users share power and bandwidth rather than bit-rate and interfere with each other.

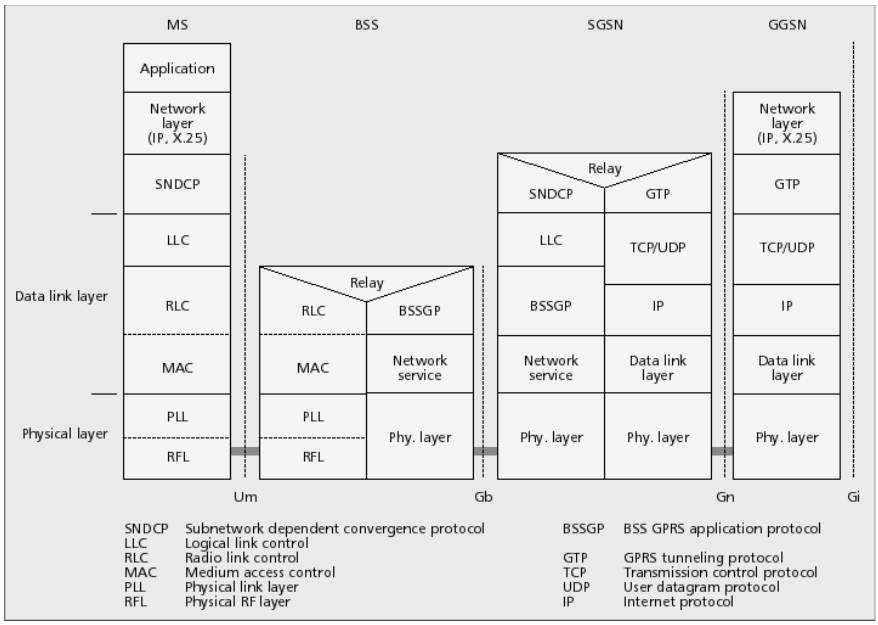


Figure 2.9: GPRS data transmission plane[EVB01].

The introduction of the enhanced GPRS backbone network provided the skeleton for packet–switched data services in mobile systems. However, the need of increased data rates could not be satisfied with GPRS, as on the one hand the peak data rates are still very limited, and, most of the capacity in GSM is already occupied by voice services.

Therefore, the Third–Generation Partnership Project (3GPP) developed evolutionary and revolutionary concepts to enhance data rates in wireless environments as well as to integrate new services. To enhance the data rates within the 2G systems the Enhanced Data Rates for GSM and TDMA/136 Evolution (EDGE) concept has been developed. EDGE provides an evolutionary path from existing 2G standards for services with higher data rate within the already deployed 2G radio resources. EDGE promises fast availability, reuse of existing infrastructure, as well as support for gradual introduction. EDGE was first proposed as an evolution of GSM at the beginning of 1997 and adopted in January 1998 to provide data rates up to about 384 kbit/s. Al-

though reusing the GSM structure, EDGE can be seen as a generic air interface for efficiently providing data services with higher bit-rates. In order to increase the gross bit-rate, 8–Phase Shift Keying (PSK), a linear high-level modulation, is introduced in addition to the binary Gaussian Minimum Shift Keying (GMSK) modulation originally present in GSM. Both GSM data services, HSCSD and GPRS have been extended with EDGE to Enhanced CSD (ECSD) (with peak data rate $4 \times 38.8 = 155.2$ kbit/s) and Enhanced GPRS (EGPRS) (with peak data rate $8 \times 59.2 = 473.6$ kbit/s), respectively. The selection of one out of nine possible combinations of modulation and code rate, referred to as Modulation and Coding Scheme (MCS) is based on regular measurements of link quality. The MCS might be adapted to changing channel conditions even during an active session.

The second path to enhance data rates in mobile environments was much more revolutionary and lead to the specification of 3G wireless systems, in Europe also known as Universal Mobile Telecommunications Systems (UMTS). The core network in UMTS relies on GPRS technology. However, in contrast to the evolutionary path for the access network in EDGE, the air interface for UMTS Terrestrial Radio Access Network (UTRAN) was completely re–designed to accommodate new services with data rates up 2 MBit/s, having lead, among others, to significant delays in the introduction of this new technology. UMTS also distinguishes between dedicated channels and shared channels to accommodate different applications and provides significant flexibility in terms of data rates as well as QoS provisions, for details we refer the interested reader to [HT02].

To further increase packet data throughput to mobile terminals in UMTS release 5 a new shared channel concept under the acronym High–Speed Downlink Packet Access (HSDPA) [HT02] has been introduced. The key new features compared to standard UMTS packet transmission modes are the use of adaptive modulation and coding to perform link adaptation instead of fast power control, fast layer–1 hybrid ARQ with transmission combining, as well as fast scheduling closer to the air interface on a very short time–scale of 2 ms. A complementary concept to HSDPA in the uplink has been introduced in UMTS release 6 under the acronym Enhanced Uplink Data CHannel (EUDCH). In addition, for 3GPP release 6, Multimedia Broadcast/Multicast Service [MBM03, MBM05] (MBMS) has been included. MBMS allows the simultaneous distribution of data and real–time applications such as live video with reasonable quality to several or even many receivers within the existing GSM and UMTS infrastructure with just small modifications in the radio access network. A possible deployment scenario consists of several 10,000 spectators in a football stadium being able to have access to instantaneous replays of the most important scenes just using their everyday cell phone. Similar extensions as HSDPA, EUDCH, and MBMS have also been introduced for CDMA–2000. The migration to 4G wireless networks, mainly considered as all-IP networks, is an exciting and challenging task in the future, but not explicitly considered in the following.

2.3.2 Integration of Multimedia Services

The integration of multimedia services in 3G wireless systems has been addressed in different recommendations of 3GPP, depending on the application as well as the considered protocol stack. Among others circuit–switched and packet–switched conversational services for video telephony and conferencing have been discussed in [CCS02] and [PSS02a], respectively. Furthermore, the integration of live or pre–recorded video Packet–Switched Streaming [PSS02a, PSS02b]s (PSSs) services in [PSS02b] and MMSs in [MMS02] based on a packet–based protocol stack is discussed. Media formats and codecs in 3GPP are specified in [MMS02] which are almost identical to the ones considered in the AVT group of the IETF, but possibly with lower complexity modes only.

We will in the following concentrate on real–time video services. To provide basic video service in the first release of the 3G wireless systems, the well–established and almost identical baseline H.263 [H26b] version 1 and the MPEG-4 visual [MPG] Simple Profile have been integrated. The choice was based on the manageable complexity of the encoding and decoding process as well as on the maturity and simplicity of the design. However, the now established joint ISO/ITU standard H.264/MPEG-4 AVC [H2603] promises significant gains in terms of compression efficiency. Therefore, the baseline profile of this new codec is integrated in release 6 of 3GPP for optional use and will likely be mandatory for future releases. Although in the first roll–out of 3G systems circuit–switched video transport based on the ITU-T Recommendation H.324M [H3297][12] is still considered, it is likely that IP–based packet–switched communication will dominate multimedia transmission in wireless environments. 3GPP has chosen to use SIP and SDP for call control [IPM02] and RTP for media transport [PSS02a]. In other words, the IP–based protocol stack as presented in subsection 2.2.3 will also be the main vehicle to transport real–time multimedia transmission in existing, emerging and future wireless systems. Figure 2.10 shows a

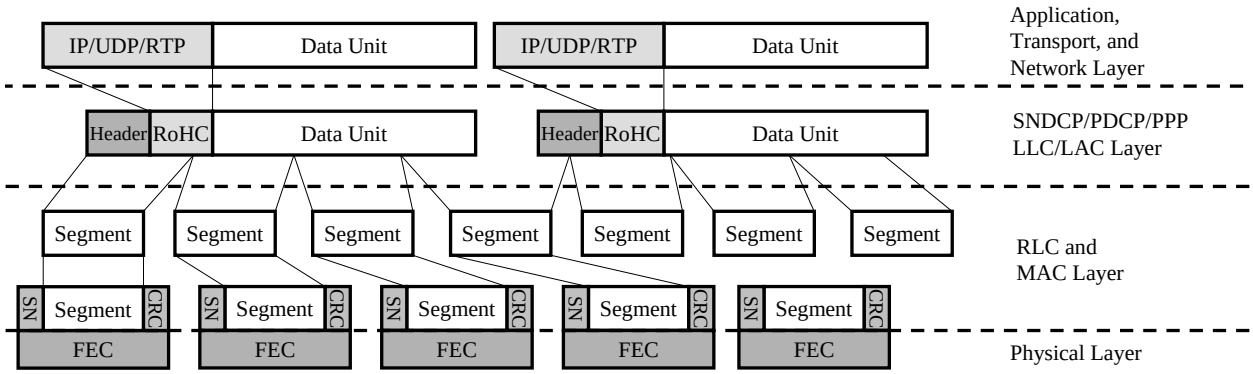


Figure 2.10: Processing of a multimedia data unit through the protocol stack of a wireless system such as GPRS, EGPRS, UMTS, or CDMA–2000.

typical processing of a multimedia data unit, e.g. a NAL unit in case of H.264, encapsulated in RTP/UDP/IP [WHS+05] through the protocol stack of a wireless system such as GPRS, EGPRS, UMTS, or CDMA–2000. We will in the following concentrate on UMTS terminology, the corresponding layers for other systems are shown in Figure 2.10 and Figure 2.5. After Robust Header Compression [HJH+01] (RoHC) this IP/UDP/RTP packet is encapsulated into one Packet Data Convergence Protocol (PDCP) packet that becomes an RLC-Service Data Unit (SDU). As typically this SDU has larger size than a RLC–PDU it is segmented into smaller units or link layer packets of length k_{LLP} which serve as the basic units to be transmitted within the wireless system. The length of these segments depends on the bearer of the wireless system and the CS or MCS in use. The RLC layer in wireless systems can operate in one of basically two different modes, Unacknowledged Mode (UM) and Acknowledged Mode (AM). For all RLC modes, sequence numbering and Cyclic Redundancy Check (CRC) error detection is performed. However, whereas the UM is unidirectional and data delivery is not guaranteed, in the AM an ARQ mechanism is used for backward error correction. In the latter mode, usually a persistent ARQ is used, i.e., retransmissions of erroneous packets are performed until the link layer packet is correctly received.

[12] H.324 [H3295] was primarily established by the ITU-T for low bit-rate circuit–switched modem links but extended for error–prone extensions for mobile circuit switched low bit-rate conversational services under the acronym H.324M. 3GPP adopted H.324M including an error robust multiplexing protocol H.223 Annex B for circuit-switched video communication. Although H.324 and the RTP/UDP/IP stacks a completely different switching philosophy, the loss and delay effects on media data when transmitted over wireless dedicated channels are very similar.

The physical layer generally adds FEC to RLC-PDUs depending on the coding scheme in use such that a constant length channel–coded block is obtained. This channel–coded block is further processed in the physical layer before it is sent to the far end receiver. There, error correction and detection is performed, possibly retransmissions are requested. In general, the detection of a lost segment of an IP packet results in the loss of the entire packet.

Latest extensions to wireless systems such as HSDPA, EUDCH, and MBMS basically reuse the same integration frame work and protocol stack as presented for EGPRS, UMTS, and CDMA–2000, but mainly differ on the processing in the MAC and physical layer. The extensions have initially been introduced for non real–time services. However, we have shown in [LJSB04] that real–time streaming services can be integrated in the HSDPA framework. Similarly for MBMS in which initially also only download–and–play applications are considered, but the performance of real–time services within MBMS is feasible with appropriate bearer definitions as for example shown in [JSL04]. Investigations on the feasibility for real–time services within EUDCH still have to be performed. Although we will not explicitly address HSDPA, EUDCH, and MBMS in the remainder of this work, it is worth to mention that many concepts developed in this thesis are applicable to these extensions of mobile systems.

2.4 System Abstractions and Channel Models

2.4.1 Overview

The previous two sections provided insight into the complexity and variety when transmitting real–time multimedia data over wireless systems. The logical architecture of GPRS as shown in Figure 2.8 and its protocol plane according to Figure 2.9 provides an overview on the integration of IP–based traffic into the system. Unfortunately, the implementation of all system components for the evaluation of different system designs is obviously far too complex and completely infeasible, abstractions are necessary. However, we have also observed that a too simplified model as presented in Figure 2.1 might hide important issues in the system design. In this system model, it is for example assumed that content generation, multimedia coding, and error protection are performed directly at the radio access, and that information can be easily exchanged between signal processing units which are located in different layers. An appropriate level of abstraction for the system is necessary.

Figure 2.11 shows an abstraction of the end–to–end communication system when one of the participants is a wireless client[13]. The arrows indicate the flow direction of different protocols, whereby some protocols are bi–directional with the main stream going into one direction. The media server or a participant in a video conference is assumed to be connected to the Internet over a wired Internet connection. The connection takes place over typical Internet protocols as introduced in subsection 2.2.3. For the connection to the wireless communication partner, the GGSN serves as the packet–based entrance to the system which basically can be viewed as a regular Internet router. The communication between GGSN, SGSN, and BSS is in general performed over over–provisioned and reliable wired links such that delays and losses can be ignored. Therefore, we assume that the Internet router is directly placed in the base station of the system as abstracted in Figure 2.11. The incoming IP packets are then processed as discussed in subsection 2.3.2 and sent to the wireless receiver over the mobile radio channel.

[13]We concentrate in the following on the scenario with just a single wireless client and media transmission in the downlink. The generalization to uplink traffic as well as two or more wireless participants is obviously possible, but does not change the presented concepts significantly.

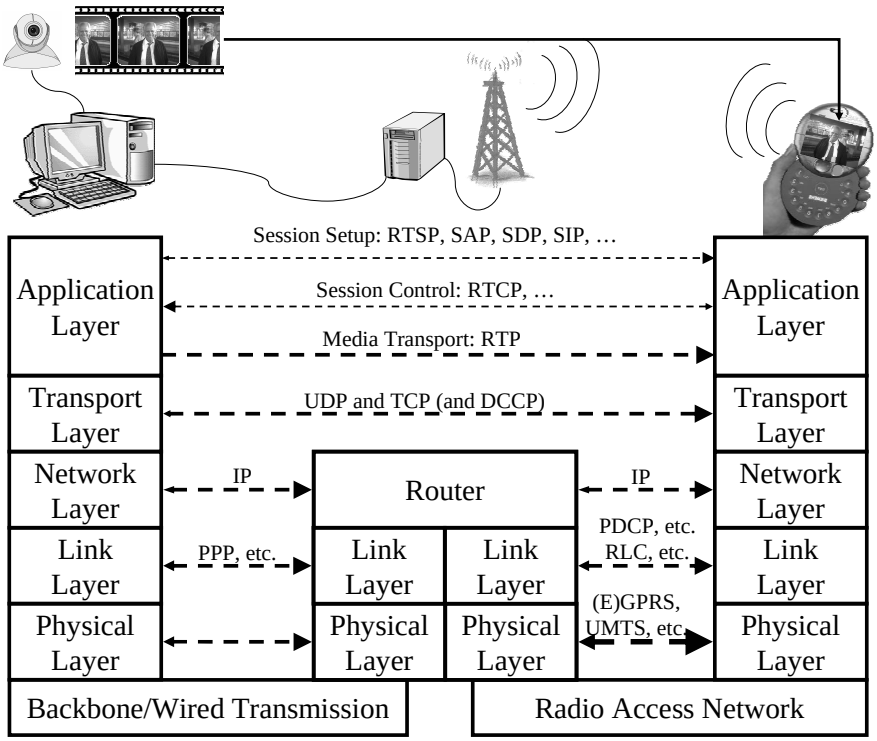


Figure 2.11: Typical wireless multimedia end–to–end connection: the base station is viewed as a router with little intelligence.

These leaves the description and characterization of individual links, whereby the link can be a physical or a virtual connection between any two layers in the system abstraction. The model hides the characteristics of the underlying layers. The description of transmission characteristics of the different wireless and mobile links is manifold and heterogeneous. Moreover, to exploit and compare coding and transmission schemes in the area of source and channel coding as well as wireless multimedia transmission the application of highly sophisticated channel models would be tedious, time–consuming, and most likely, still not comprehensive. Therefore, we restrict ourselves to only a subset of physical and virtual links. In addition, the huge amount of different mobile channel parameters and characteristics in combination with mobility, multi-user aspects, different traffic characteristics, and system parameters, etc. makes a comprehensive simulation of mobile systems completely infeasible. Therefore, only representative case studies – usually reflecting typical or worst–case situations in terms of transmission conditions, user topology, and applications – are selected to evaluate the performance. We attempt to define simple, but meaningful models to evaluate the transmission of video and multimedia data in a mobile environment.

In subsection 2.4.2 characteristics of the mobile radio channel as well as appropriate modelling assumptions for a single user link are discussed. A discrete–time signalling model with a generic interface from the channel to the channel decoders as well as very few additional parameters for a generic mobile radio channel are introduced in subsection 2.4.3. To address system level aspects it is common to define link layer models which model the loss and delay process of link layer packets as shown in Figure 2.10. In subsection 2.4.4 we introduce models for the link layer for EGPRS as well as for UMTS dedicated channels. Finally, in subsection 2.4.5 we will introduce models for the delay and loss of entire RTP/UDP/IP–packets, referred to as application layer packets, showing the effects of backbone Internet channels as well as mobile channels on application layer packets.

2.4.2 The Mobile Radio Channel

The mobile radio channel imposes major challenges for the transmission of data to wireless and mobile users. In contrast to wire–line transmission, the resources in wireless environments are significantly limited and, in addition, the transmission conditions are highly time and location variant. An example scenario for the propagation effects is shown in Figure 2.12. The electro–magnetic wave generated by the modulation of an information–bearing signal with carrier frequency f_c is radiated from the transmitter antenna and received at the mobile. The received signal at any time instant is a superposition of a direct path, and additional paths resulting of reflections from plane objects, refractions at edges, and scattering from rough surfaces. Often, there is no direct path, also called line–of–sight path, between transmitter and receiver. Then, the receiver must rely on signal components resulting from diffractions. The mobile radio channel might be viewed as a linear time–invariant filter for each location, but movement of the receivers, or surrounding objects finally result in a time–variant behavior. This time–variant behavior of the received signal strength is in general known as *fading*. In addition to fading, the transmission on mobile channels is also disturbed by the presence of thermal noise in the receiver and interference from adjacent channels and other users.

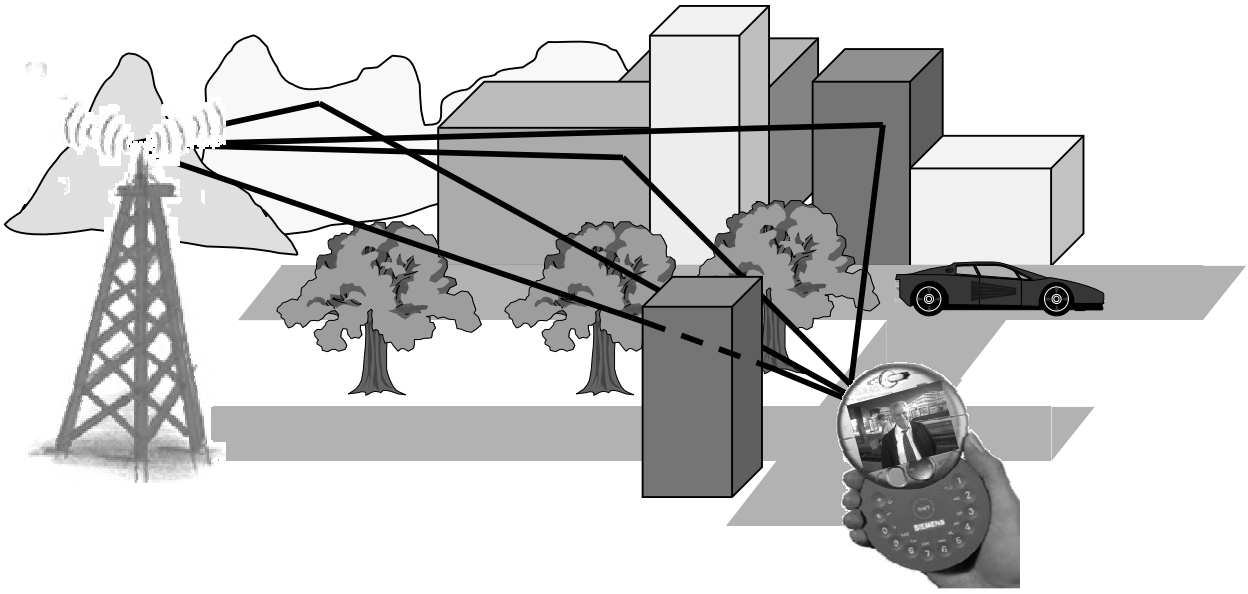


Figure 2.12: Mobile radio channel: attenuation, shadowing, and multipath reception.

To be more specific, let $x(t)$ be the equivalent low–pass signal with band–limitation W_x such that the band–bass signal at carrier frequency f_c,

$$x_{\text{BP}(t)} = \Re\{x(t)\}\cos(2\pi f_c t) - \Im\{x(t)\}\sin(2\pi f_c t)\,,$$

occupies a frequency band between $f_c - W_x$ and $f_c + W_x$. GSM, for example, provides channels with bandwidth $2W_x = 200$ kHz at carrier frequencies around $f_c = 900$ MHz and $f_c = 1800$ MHz. In UMTS, the signal bandwidth is 5 MHz at $f_c \approx 1900$ MHz. The signal $x(t)$ is transmitted over the channel with time–variant channel weighting function $\mathcal{C}(t;\tau)$[14]. The channel output $y(t)$ is

[14]The channel weighting function describes the attenuation/gain of a signal component entering the channel at time $t - \tau$ at time t.

then given as

$$y(t) = \int_{-\infty}^{\infty} \mathcal{C}(t;\tau)x(t-\tau)\,d\tau + \nu(t)\,d\tau, \tag{2.30}$$

where $\nu(t)$ includes thermal noise as well as possible interference from adjacent channels or other users. This component can very well be modelled as circular symmetric complex white Gaussian process with zero mean and power spectral density (psd) N_0[15]. In addition, the channel weighting function, $\mathcal{C}(t;\tau)$, is in general random and depends on the location, the velocity of the user, and other environmental influences. Therefore, the channel output signal y is also random.

For the description and modelling of the channel weighting function $\mathcal{C}(t;\tau)$ one can basically distinguish long–term fading and short–term fading. Long–term fading origins from propagation path loss and shadowing. The former basically depends on the distance between the receiver and the transmitter, the environment and the carrier frequency of the system [OOKF68, Hat80]. Shadowing describes the deviation from the path loss model caused by the presence or absence of large objects between the transmitter and the receiver. The receiving conditions modelled by long–term fading are assumed to be constant within the range of a few meters such that for practical mobile speeds the changes are relatively slow, at least in the range of several seconds.

Short–term fading is caused by multipath propagation as the mobile station moves. Depending on the relative length of each of the multiple signal paths, the sinusoidal wave forms might superpose constructively, or, in less favorable case, the phases of the arriving signals might be such that the signal is attenuated or even vanishes. Path length differences being in the range of $10-20$ cm can already cause deep fades for the typical carrier frequencies used in nowadays mobile systems. The multipath propagation results in time dispersion of a single transmitted impulse whereby the range of values over which delayed path components arrive at the receiver is called *multipath* spread T_{mp}. The inverse of the delay spread, the so–called *channel coherence bandwidth* $W_{\text{coh}} \triangleq 1/T_{\text{mp}}$ provides a measure of the width of the frequency band over which the fading is highly correlated. The value of the delay spread and the channel coherence bandwidth strongly depends on the environment. In urban areas the maximum delay spread is usually in the range of a few microseconds, i.e., $W_{\text{coh}} \approx 0.5$ MHz, but in hilly terrain late echos might cause delay spreads of up to 20 microseconds. If the bandwidth of the signal is significantly less than the coherence bandwidth, i.e., $W_x \ll W_{\text{coh}}$, all signal components at time t experience almost the same attenuation $\mathcal{C}(t)$, i.e., $y(t) = \mathcal{C}(t)x(t)\,d\tau + \nu(t)$. Such a channel is referred to as frequency–nonselective or *flat fading* channel, and $\mathcal{C}(t)$ is called *channel gain*.

Similarly to the delay spread, the Doppler spread W_{Dop} provides a measure of the rapidness of changes in the channel weighting function $\mathcal{C}(t;\tau)$. The maximum Doppler spread can quite well be estimated as $W_{\text{Dop}} = f_c v/c$ with v the velocity of the mobile and the c the speed of light. For GSM at $f_c \approx 900$ MHz, the Doppler spread results in $W_{\text{Dop}} \approx v$ Hz/(km/h) and therefore $W_{\text{Dop}} \ll W_x$. The inverse of the Doppler spread, the so–called *coherence time* $T_{\text{coh}} \triangleq 1/W_{\text{Dop}}$ is a measure for the rapidness of the fading. The *spread factor*, defined as $T_{\text{mp}}W_{\text{Dop}}$, provides an indication, if the channel weighting function $\mathcal{C}(t;\tau)$ can be easily measured at the receiver in the demodulation process. To be able to measure the channel, the delay spread should be $T_{\text{mp}}W_{\text{Dop}} \ll 1$. Note that this is in general the case for GSM and UMTS with $T_{\text{mp}}W_{\text{Dop}} < 5 \cdot 10^{-5} \cdot v/(\text{km/h})$.

For *frequency–selective* channels with $W_x > W_{\text{coh}}$ the channel gains and phase shifts for different frequency components of the transmitted signal $x(t)$ are different. With $W_x \gg W_{\text{coh}}$ and the usage of the sampling theorem [Pro95], the received signal might be re–presented by a tapped–

[15]The complex noise process consists of two independent noise components with psd $N_0/2$ each.

delay–line model as

$$y(t) = \sum_{n=1}^{\lfloor T_{\text{mp}} W_{\text{coh}} \rfloor + 1} \mathcal{C}_n(t) x\left(t - \frac{n}{W_x}\right) d\tau + \nu(t), \tag{2.31}$$

where $\{\mathcal{C}_n(t)\}$ represents complex–valued, time–varying tap coefficients and are usually modelled as stationary mutually uncorrelated random processes resulting in the Wide–Sense Stationary Uncorrelated Scattering (WSSUS) channel model. Note that (2.31) also includes the flat–fading channel with $\mathcal{C}(t) = \mathcal{C}_1(t)$.

This leaves the statistical description and modelling of the channel tap coefficients $\{\mathcal{C}_n(t)\}$ for which there exist several probability distributions. Provided that the number of paths contributing to a multipath component $\{\mathcal{C}_n(t)\}$ is high, and, that many of these path contribute to the total received power for this multipath component, then the central limit theorem leads to a Gaussian process model [Pap91] for the tap coefficients. If the process is zero–mean, then the amplitude of this process at any time–instant, $\alpha_n(t) = |\mathcal{C}_n(t)|$, is Rayleigh–distributed with mean $\bar{\alpha}$ where due to stationarity assumption the mean is independent of the time t. The phase follows a uniform distribution between 0 and 2π. Note that this distribution is described by a single parameter $\bar{\alpha}_n(t)$ for each tap n. For the flat–fading channel, the tap–index n is dropped. Alternative models such as Nakagami-m or Rice distribution suggest to generalize the Rayleigh distribution, whereby the latter takes into account a line–of–sight component in the communication link and is modelled by non–zero mean Gaussian process for the tap–coefficients. However, as the Rayleigh distribution takes into account the worst–case scenario without line–of–sight, and as it can be described by a single parameter, it is most widely used to model the channel tap coefficients.

To describe and model the correlation of the channel tap coefficient $\alpha_n(t)$ it is again assumed that the signal is a superposition of many significant scatterers with random phase, and, in addition, with random arrival angle. Then, the power spectrum of the fading process can be described with the Jakes spectrum [Jak74] showing characteristic peaks at the maximum Doppler frequency W_{Dop}. In [Jak74] also a method to model and simulate the statistical process, the probability distribution of amplitude, phase, and correlation, of channel tap coefficients $\mathcal{C}_0(t)$ based on a Monte–Carlo like methods is proposed.

In the development of a wireless system emulator and demonstrator [LJSB04] for HSDPA a highly sophisticated wireless system model has been integrated taking into account short–term and long–term fading effects as well as different interference and noise components. For the specific scenario of a user walking in an urban environment, the received signal power for each time–instant, compared to the noise and interference, called Signal–to–Noise and Interference Ratio (SNIR), has been recorded for 10 minutes. Figure 2.13 shows the SNIR over time for an example user and different time resolutions. The top level shows distance and slow–fading effects for the moving user; the SNIR results as an average over 2 seconds. Periods of good receiving conditions vary with bad receiving conditions within 10 to 20 seconds. The maximum difference in the SNIR is about 25 dB. Similar effects are also shown for a shorter time–window – 100 seconds in the second plot and average SNIR over 0.5 seconds. Finally, plot 3 shows the average SNIR over 2 ms within 20 seconds: Significant oscillations within just less than 1 second are obvious. The maximum Doppler frequency for this example with $f_c \approx 2$ MHz and $v = 3$ km/h is about $W_{\text{Dop}} \approx 6$ Hz. Observe that within just a few seconds, the SNIR can vary also by about 25 dB. With this figure in mind it is obvious that the performance of real–time multimedia services with time–constraints strongly depends on the characteristics of the channel. However, a comprehensive treatment and modelling of the such channels is not possible. Therefore, we introduce simpler

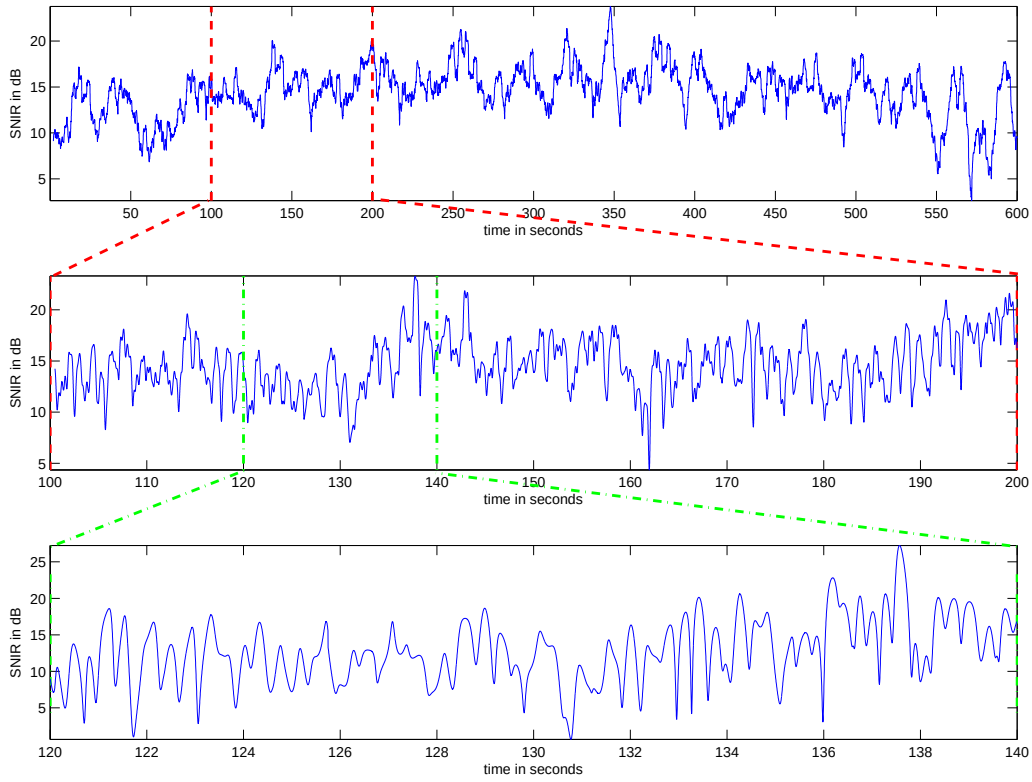


Figure 2.13: Receiver SNIR over time for a walking user in an typical UMTS environment without power control [LJSB04].

channel models which attempt to reflect the variations of the wireless channels on different system levels.

2.4.3 Signaling and Discrete–Time Models

Modulation

Just as for any other physical transmission medium, the mobile radio channel transmits analog signals[16]. For the transmission of digital signals, the modulator at transmitter side and the demodulator a receiver side serve as interfaces between the analog and the digital world. In case of linear modulation schemes, the transmitted equivalent low–pass signal can be expressed as the superposition of shifted and complex–weighted pulses, i.e.,

$$x(t) = \sum_i x_i \phi_x \left(t - iT_x\right), \tag{2.32}$$

[16]"You can not throw bits into the air", J. Hagenauer, 1998, private communication.

with T_x being the modulation interval, $\{x_i\}$ a sequence of complex symbols, and $\phi_x\left(t - iT_x\right)$ any transmission pulse satisfying the orthonormality condition

$$\frac{1}{T_x}\int \phi_x\left(t\right)\phi_x^*\left(t - iT_x\right)\, dt = \mathbf{1}\left\{i = 0\right\},$$

where $\mathbf{1}\left\{A\right\}$ defines the indicator function being 1 if A is true and 0 otherwise. The spectral characteristics of $x(t)$ are defined by the transmission pulse $\phi_x\left(t\right)$ and the symbol interval T_x. The most popular pulse shape used in mobile communication is the root–raised cosine impulse resulting in a bandwidth of the transmitted signal of $W_x = \frac{1+\alpha_{\text{rc}}}{2T}$ with $0 \leq \alpha_{\text{rc}} \leq 1$. Therefore, for minimum $\alpha_{\text{rc}} = 0$, the symbol interval T_x is limited by the bandwidth of the channel to $T_x \geq 1/(W_x)$. As pulse shape filters with $\alpha_{\text{rc}} = 0$ or close to 0 are hard to realize, some bandwidth increase has to be sacrificed. In UMTS, for example, a root–raised cosine pulse shape filter with $\alpha_{\text{rc}} = 0.22$ has been chosen. For GSM orthogonality has been sacrificed at the expense of spectral efficiency with the introduction of a partial–response modulation scheme GMSK such that $W_x T_x = 0.3$.

Provided perfect synchronization at the receiver, the demodulator decomposes the received signal $y(t)$ obtained from the transmission over a channel according to (2.30) with input signal $x(t)$ according to (2.32) using the time–delayed and orthogonal basis functions $\phi_x\left(t - iT_x\right)$ to obtain a sequence $\{y_i\}$ of discrete–time received values as

$$y_i = \frac{1}{T_x}\int y(t)\phi_x^*\left(t - iT_x\right)\, dt = \sum_n \mathcal{C}_{n,i}x_{n-i} + \nu_i, \tag{2.33}$$

where the discrete–time channel coefficients result in

$$\mathcal{C}_{n,i} = \frac{1}{T_x}\int\int \phi_x\left(t - (i-n)T_x - \tau\right)\mathcal{C}(t;\tau)d\tau\phi_x^*\left(t - iT_x\right)\, d\tau dt = \frac{1}{T_x}\int \mathcal{C}_n(t)\phi_x^*\left(t - iT_x\right)\, dt, \tag{2.34}$$

and the discrete–time noise sample yield

$$\nu_i = \frac{1}{T_x}\int \nu(t)\phi_x^*\left(t - iT_x\right)\, dt. \tag{2.35}$$

The noise samples ν_i are independent and identically distributed (iid) due to the orthogonality of the delayed basis functions and the variance of the process is given as $\sigma^2 = N_0 W_x = N_0/(2T_x)$. Therefore, the probability density function (pdf) of the noise is given as

$$f_\nu(\nu, \sigma^2) = \frac{1}{2\pi\sigma^2}\exp\left(\frac{|\nu|^2}{2\sigma^2}\right). \tag{2.36}$$

The bandwidth extension of the received signal due to Doppler effects can be ignored for practical systems. Whereas (2.31) presents an equivalent low–pass model only for the mobile radio channel at any time t, (2.33) now serves as discrete–time equivalent low–pass model which includes modulation, demodulation, the mobile radio channel, interference, and thermal noise. Note that multipath propagation causes inter–symbol interference (ISI), i.e., a received symbol y_i contains contributions from more than one transmitted symbol x_i.

The discrete–time transmission signal x_i can basically be any complex number and the alphabet $\mathcal{S}_x$ for each signal can be infinitely large or even different for each time–instant i. The average power over the random symbols x_i is usually limited to some value P_x, i.e., $P_x = \mathbb{E}\left\{|x_i|^2\right\}$ and therefore, the average energy per transmitted symbol, $\mathcal{E}_s$, is defined as

$$\mathcal{E}_s = \mathbb{E}\left\{\int |x(t)|^2\, dt\right\} = P_x T_x. \tag{2.37}$$

Although high–dimensional input alphabets $\mathcal{S}_x$ provide advantages especially for high SNRs, in mobile communications usually input signal alphabets sizes $|\mathcal{S}_x|$ are restricted as SNRs are in general low, and detection is simplified significantly. In UMTS, Binary Phase Shift Keying (BPSK) is used in the uplink, and Quaternary Phase Shift Keying (QPSK) in the downlink. Only recently 16–Quadrature Amplitude Modulation (QAM) has been introduced for HSDPA. Note, that with Gray mapping and due to the orthogonality of real and imaginary component in the equivalent low–pass signal, QPSK can be viewed as the transmission of two independent BPSK signals.

The modulation in GSM can basically also be viewed as BPSK, if differential pre–coding, GMSK modulation, and de–rotation are modelled as a time–invariant part of the frequency selective channel. For EDGE, a linear 8–PSK modulation scheme with spectral properties equivalent to the GMSK modulation in GSM is used.

Detection and Diversity

Let us assume that messages are transmitted in blocks $\mathbf{x}$ consisting of N_x symbols $x_i, i = 1, \ldots, N_x$ such that all received symbols y_i for this message are independent from any transmitted symbol of any other message, i.e., no ISI takes place over message boundaries. In practice, this is accomplished by the introduction of guard intervals between two messages of sufficient length. For example, in case of GSM the guard interval is roughly $30\ \mu$s. In addition, let us assume a coherent receiver, i.e., the receiver has knowledge of the entire set of channel tap coefficients $\{\mathcal{C}_{n,i}\}$, and the SNR, summarized in $\mathcal{C}$. Based on this channel information and the observed sequence $\mathbf{y}$, maximum a posteriori detectors minimizing the sequence error probability $\Pr\left\{\hat{\mathbf{X}} \neq \mathbf{X}\right\}$ decide for the sequence $\hat{\mathbf{x}}$ which has most likely been transmitted, i.e.,

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}\in\mathcal{S}_x^{N_x}} \Pr\left(\mathbf{X} = \mathbf{x} \,|\mathbf{Y} = \mathbf{y}, \mathcal{C}\right) \tag{2.38}$$

$$= \arg\max_{\mathbf{x}\in\mathcal{S}_x^{N_x}} \left(\Pr\left(\mathbf{Y} = \mathbf{y} \,|\mathbf{X} = \mathbf{x}, \mathcal{C}\right) \Pr\{\mathbf{X} = \mathbf{x}\}\right) \tag{2.39}$$

which follows by the application of Bayes' rule. In case of equally probable input messages $\mathbf{X}$ (2.38), is identical to the simpler maximum likelihood detector. For the case of Gaussian noise, the maximum likelihood detector decides for the sequence $\hat{\mathbf{x}}$ with minimum Euclidean distance

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathcal{S}_x^{N_x}} \sum_{i=1}^{N_x} \left| y_i - \sum_n \mathcal{C}_{n,i} x_{n-i} \right|^2 . \tag{2.40}$$

A nice property of the maximum–likelihood detector is its independence of the actual SNR. If the resulting symbols are further processed in the receiver by symbol–based algorithms, e.g., in case of channel decoders or if the measure of interest is the symbol or bit error probability rather than the sequence error probability, a symbol–based detector is preferred which reads as

$$\forall i \quad \hat{x}_i = \arg\max_{\mathbf{x}\in\mathcal{S}_x} \Pr\left(X_i = x \,|\mathbf{Y} = \mathbf{y}, \mathcal{C}\right). \tag{2.41}$$

As already mentioned, for further processing of the detected symbols in the receiver, hard–decisions are usually sub–optimal. In so–called *soft detection* algorithms the detector leaves the decision open and forwards the symbol a posteriori probability (APP) values $\Pr\left(X_i = x \,|\mathbf{Y} = \mathbf{y}, \mathcal{C}\right)$ to the next processing step that can be computed by summing all sequence APP having symbol x at position i. In practice, especially for binary transmission schemes, it has proven to be useful [HOP96] to use the log–likelihood ratio (LLR) of a transmission symbol x given the observed

channel output y. For BPSK with $\mathcal{S}_x = \{\pm\sqrt{P_x}\}$, the LLR is defined as[17]

$$L\left(X_i \,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right) \triangleq \log\left(\frac{\Pr\left(X_i=+\sqrt{P_x}\,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right)}{\Pr\left(X_i=-\sqrt{P_x}\,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right)}\right) \tag{2.42}$$

$$= \psi\left(X_i \,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right) + L\left(X_i\right), \tag{2.43}$$

where $\psi\left(X_i \,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right)$ summarizes the information from the channel, and $L\left(X_i\right)$ takes into account the a priori knowledge about the channel input X_i. So the basic goal of the detector at the receiver side is the generation of the channel LLR part $\psi\left(X_i \,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right)$.

In case of multi–path propagation, the a posteriori information or the LLR can be obtained using for example a symbol–based maximum a posteriori equalizer [Pro95] based on the Bahl–Cocke–Jelinek–Raviv (BCJR)–algorithm [BCJR74]. The equalizer transfers the received sequence $\mathbf{y}$ in a sequence of symbol APPs $\{\Pr\left(X_i=x\,|\mathbf{Y}=\mathbf{y},\mathcal{C}\right)\}_{i=1}^{N_x}$. In GSM for example, this equalizer is jointly used to detect the GMSK modulated signal and to combat the multipath channel. In UMTS, multipath transmission can be eliminated by the use of direct–sequence spread–spectrum signalling in combination with a Rake receiver. There exist many simple or more advanced techniques [Pro95], [BPS98] which avoid or at least limit the influence of ISI.

In addition, the reception of multiple copies of the same signal with independent channel taps – referred to as *diversity* – is very beneficial and one of the key concept in mobile communications. Receiver can for example exploit diversity by multiple receive antennas: Assume that the signal is received over multiple independent single–path Rayleigh fading channels denoted as $\mathcal{C}_n$ with $\mathbb{E}\left\{|\mathcal{C}_n|^2\right\} = 1$. Then, with the use of maximum–ratio combining, the multipath channel can be converted to a flat fading single–path channel with just a single channel coefficient $\mathcal{C}$ as $y_i = \sum_n |\mathcal{C}_{i,n}|^2 x_i + \hat{\nu}_i$. Diversity can also be introduced by appropriate signaling methods. For example, recently many efforts have been made to exploit transmit diversity in systems by the use of Multiple–Input Multiple–Output (MIMO) antenna systems. Broadband transmission with $W_x \gg W_{\text{coh}}$ provides frequency–diversity due to the reception of multiple independent paths. Another form of simple diversity can be obtained by repeatedly transmitting the same signal over time whereby the transmission interval is greater than the coherence time $T_{\textbf{coh}}$. For this reason, in systems with narrowband signaling such GSM methods are introduced to spread the transmitted signal over several independent channel realizations by the use of an *interleaver*. These realizations might be correlated, but for GSM they can be viewed as uncorrelated as *frequency hopping* is commonly applied. In frequency hopping systems, consecutive transmission blocks modulate carriers with spacing larger than the coherence bandwidth W_{coh} resulting in virtually uncorrelated fading in the blocks. The number of blocks, denoted as F in the following, determines the degree of diversity, but also influences the decoding delay. From the channel coding point–of–view, transmitting the same signal over multiple independent paths can be viewed as a repetition code. Obviously, with the introduction of better codes, efficiency can be increased significantly as discussed in chapter 4.

Channel–State Information

As already discussed in a very general context in subsection 2.1.1, the availability of CSI at the receiver and transmitter is beneficial. In mobile communication systems the measurements of the channel weighting function $\mathcal{C}(t;\tau)$ or its corresponding discrete representation $\mathcal{C}_{i,n}$ is possible and actually performed by the use of pilot symbols. If the distance of the pilot symbols is smaller than the coherence time T_{coh} of the channel, we can assume that the receiver has accurate knowledge

[17] $\log$ is the logarithm to base e in the sequel if not stated otherwise.

of the channel coefficients $\mathcal{C}_{i,n}$. Both, GSM and UMTS, provide sufficient pilot information to assume perfect knowledge of the channel coefficients $\mathcal{C}_{i,n}$ at the receiver; the receiver is referred to as *coherent*.

The scenario is different for the transmitter: Only in case of Time–Division Duplex (TDD) systems the transmitter can obtain the knowledge by its own measurements. Otherwise, it must rely on feedback messages from the receiver. In addition, the feedback must be so fast, i.e., within the coherence time T_{coh}, and so accurate, i.e., to contain a sampled version of the entire transmission frequency band with resolution W_{coh}, to be useful for the transmission of future symbols. Both GSM and UMTS are Frequency–Division Duplex (FDD) systems, but power control feedback messages are in use. In GSM, an open–loop power control with very few feedback messages per second is used to compensate long–term fading effects, but short–term fading cannot be compensated. In contrast, UMTS provides a closed–loop power control with high control message frequency which allows to at least partly compensate short–term fading for low to medium speeds of the mobile. The concept of CSI available at the transmitter is even enhanced with HSDPA and EUDCH. With this background the motivation for a separate description of the statistics of long–term and short–term fading effects is obvious. For this reason we normalize the channel tap–power as $\mathbb{E}\left\{\sum_n |\mathcal{C}_n|^2\right\} = 1$ and describe the long–term effects by the SNR as

$$\text{SNR} = \frac{P_x}{N_0 T_x} = \frac{\mathcal{E}_s}{N_0}, \tag{2.44}$$

which describes the average ratio of the received power to the noise power at the receiver. The channel is completely characterized by its transition probability $\Pr\{Y = y \,|X = x, \mathcal{C}\}$ for a given input signal x and a given channel $\mathcal{C}$.

Therefore, these long–term effects are usually compensated at least to some degree by the use of power control, i.e., the transmitter adapts the transmission power of the signal $x(t)$ such that a certain average receive power can be guaranteed. However, the range of transmit power adaptation is limited by the amplifiers in the transmitter as well as by increased intercell interference in interference–limited cellular systems such as GSM or UMTS.

Binary Input Block–Fading Additive White Gaussian Noise (AWGN) Channel

In the light of the discussion on the mobile radio channel, signalling, detection, diversity, and CSI, we will in the following introduce a mobile channel model which definitely includes not all effects, but is sufficiently general and representative in terms of energetic and timing aspects to be used for the assessment of mobile video transmission in combination with channel coding methods. The channel is referred to as block–fading AWGN channel [SOW94] and is shown in Figure 2.14. We restrict ourselves to binary input signals for simplicity, but also as mobile systems usually operate at low SNRs. As we use binary input signaling we simplify the two–dimensional complex signal model to a one–dimensional real–valued. We assume that binary messages can be transmitted in slots denoted by their indices $f = 1, 2, \ldots$. Within each slot, the propagation channel is assumed to be slowly time–varying and frequency–flat presuming that the coherence time T_{coh} is sufficiently long. In particular, the transmission conditions for each slot are defined by a single parameter, the channel gain $\alpha_f \in \mathbb{R}$ which is assumed to be constant over the entire slot f and is normalized to $\mathbb{E}\left\{|\alpha_f|^2\right\} = 1$. Each slot f has equal duration and allows the transmission of N_v binary symbols. Let $\mathbf{x}_f \triangleq (x_{f,1}, \ldots, x_{f,N_v})$, $\mathbf{y}_f \triangleq (y_{f,1}, \ldots, y_{f,N_v})$ and $\nu_f \triangleq (\nu_{f,1}, \ldots, \nu_{f,N_v})$ be the transmitted signal, the received signal and the noise during slot f, respectively. Furthermore, assume that the transmitted symbol $x_{f,v} \in \{\pm 1\}$ has unit power, but is multiplied with $\sqrt{P_{x,f}}$ to adjust the

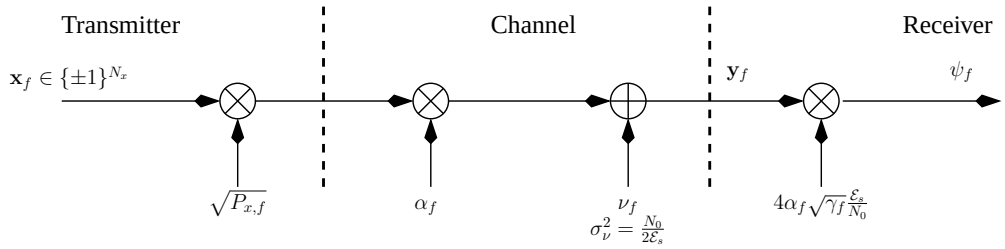


Figure 2.14: Binary input block–fading AWGN channel [SOW94] with power control.

transmission power, possibly for each slot f. Then, the received symbol $y_{f,v}$ is given as

$$\forall_{f,v} \quad y_{f,v} = \alpha_f \sqrt{P_{x,f}} x_{f,v} + \nu_{f,v}, \tag{2.45}$$

where the noise $\nu_{f,v}$ now represents a one–dimensional zero–mean Gaussian random variable variabce psd $N_0/(2T_x)$ in the signaling bandwidth W_x. Assuming constant transmission power for each slot f, i.e., $P_{x,f} = P_x$, the SNR for this particular slot is defined as

$$h_f \triangleq |\alpha_f|^2 \frac{2\mathcal{E}_s}{N_0} = |\alpha_f|^2 \frac{2P_{x,f}T_x}{N_0}, \tag{2.46}$$

where the transmission energy $\mathcal{E}_s$ is defined as $\mathcal{E}_s \triangleq P_x T_x$. If the power can be controlled for each slot independently, and with the definition of a power control factor $\gamma_f \triangleq P_{x,f}/\mathbb{E}\{P_{x,f}\}$ and $\mathbb{E}\{\gamma_f\} = 1$, the SNR with power control for this particular slot f is defined as

$$\hat{h}_f \triangleq \gamma_f h_f, \tag{2.47}$$

whereby in this case the transmission energy $\mathcal{E}_s$ is now specified as the *average* transmission power as $\mathcal{E}_s \triangleq \mathbb{E}\{P_{x,f}\}/T_x$. In the following we include the power control factor γ_f in the presentation; with $\gamma_f = 1$ this simplifies to the case with constant input power $P_{x,f} = P_x$.

For this particular channel model, the computation of the channel part of the LLR $\psi_{f,v}$ for transmitted symbol $X_{f,v}$ according to (2.42) given the product of the channel gain and the power control gain $\alpha_f\sqrt{\gamma_f}$ and the average SNR $\mathcal{E}_s/N_0$ simplifies to

$$\psi_{f,v} \triangleq \psi\left(X_{f,v} \middle| \mathbf{Y} = \mathbf{y}, \alpha_f\sqrt{\gamma_f}, \mathcal{E}_s/N_0\right) = 4\alpha_f\sqrt{\gamma_f}\frac{\mathcal{E}_s}{N_0} y_{f,v}. \tag{2.48}$$

Note that for the flat–fading case the channel LLR $\psi_{f,v}$ only depends on the received symbol $y_{f,v}$ but not on the entire received vector $\mathbf{y}$ as in case of the frequency–selective channel.

According to (2.45) the binary input block–fading AWGN channel according to Figure 2.14 can equivalently be described as an AWGN channel with binary input signals $x = \pm 1$ and the variance of the channel noise being constant within one slot as

$$\sigma_f^2 = \frac{1}{\hat{h}_f} = \frac{1}{\gamma_f h_f} = \frac{N_0}{(2\mathcal{E}_s |\alpha_f|^2 \gamma_f)}$$

resulting in the received signal y. The channel LLR ψ is then computed according to (2.48).

The actual channel model is completely characterized by only very few parameters according to Table 2.3. The average SNR $\mathcal{E}_s/N_0$ is assumed to be known at the transmitter and receiver. This is justified by the use of simple open–loop power control. Different distributions of the channel

gain α can describe correlated or uncorrelated channels and take into account the availability of line-of-sight communication. We will mainly focus on the iid Rayleigh case. Note that for the case of Rayleigh fading without power control, the SNR h is is exponentially distributed with mean $\bar{h} = 2\mathcal{E}_s/N_0$. The power control factor γ_f is in general set to 1 if the channel gain α_f is not known at the transmitter. Only if the channel gain realization α_f is known at the transmitter, γ_f is adapted. The product of the number of symbols per slot, N_v, and the Transmission Time Interval (TTI) of the transmission slots, T_f, determines the maximum transmission bit-rate. The TTI is typically in the range of 20 ms for GSM, 10 ms for UMTS and 2 ms for HSDPA.

Table 2.3: Parameters for block–fading AWGN channel model.

Parameter	Description	Range
$\mathcal{E}_s/N_0$	average SNR	$0 \ldots 10$ dB
p_α	distribution of channel gain	AWGN, iid Rayleigh, ...
γ_f	power control factor	1, $f(\alpha_f, \mathcal{E}_s/N_0)$
N_v	transmission symbols per slot	any integer
T_f	TTI of transmission slots	$2 - 25$ ms

2.4.4 Link-Layer Channel Models

Overview

In existing wireless systems, the mobile radio channel as well as the underlying signal processing such as channel coding, multiple access schemes, modulation schemes, and receiver algorithms, are usually completely hidden to the upper layer. Instead, as already discussed, wireless systems provide the transmission of data in link–layer packets, indicated as segments in Figure 2.10. The transmission process of these link–layer packets can be abstracted by a very simple model, namely by its size k_{LLP}, its TTI T_{TTI}, its loss process $\mathcal{L}_{\text{LLP}}$, and its constant transmission delay δ_{LLP}. In other words, the wireless system transmits the application data in junks of packets of size k_{LLP} every T_{TTI} seconds. By the use of sufficiently long block check sequences the probability of non-detected errors is extremely low and link layer packets can be assigned one of two different states at the receiver: they are correctly received or they are lost. The loss process $\mathcal{L}_{\text{LLP}}$ indicates both, the loss probability of link layer packets and also the time–correlation of the loss process. Independent link–layer packet losses can in general not be assumed, but are appropriate for some systems as discussed in the following. We summarize this model described by the four parameters, of the link layer of a wireless system model as $\mathcal{W}$, i.e., $\mathcal{W} \triangleq \{k_{\text{LLP}}, T_{\text{TTI}}, \mathcal{L}_{\text{LLP}}, \delta_{\text{LLP}}\}$.

This simplified link layer model can quite well reflect the behavior of the system and mobile radio channel over a few seconds. Even the loss process can be assumed to be stationary over this time–period, as the use of transmit power control algorithms can guarantee a certain average receive power at least to some extent. However, due to mobility with changing receiving conditions, handovers, and random activity of users, the overall receiving conditions change over a longer time period of time, usually in the range of several seconds. Then, the wireless system can in general not maintain the same service without sacrificing significant performance degradations. State-of-the-art wireless systems such as GPRS, EGPRS, or HSDPA allow to react to changing receiving conditions by the use of adaptive coding and modulation schemes, usually resulting in

a trade–off between the link layer k_{LLP}, and the loss process $\mathcal{L}_{\text{LLP}}$. To address this adaptivity, we define a wireless system not just by a single model $\mathcal{W}$, but introduce a set of $N_{\mathcal{W}}$ different models $\mathcal{W}_i$ reflecting the combination of typical system parameters settings and receiving conditions. Furthermore, the different modes of the transmission bearer, UM and AM, have to be considered by the system model. In AM, also the retransmission delay δ_{RT} of retransmission requests has to be considered which does not influence the throughput behaviour, but the delay characteristics.

An comprehensive discussion of link layer packet modelling can for example be found in [Lie99] and [LSB00] as well as references therein. In the following, we present two link layer channel models in more detail, one for UMTS and one for EGPRS, which will be used in the evaluation and assessment of video transmission strategies in chapter 7 and 9.

3G Dedicated Channel Link Layer Modelling

The modelling of 3G dedicated channels is interesting as they will initially be used for video and multimedia services in wireless environments. In the standardization for the H.264/AVC video coding standard, an environment has been proposed which allows appropriate modelling of 3G systems such as UMTS and CDMA–2000 [RSL+01]. 3G systems are designed to guarantee a certain QoS. Therefore, it can be assumed, that channel characteristics for the duration of the transmission are stationary, and the provided network QoS in terms of bit-rate and packet loss rate is assumed to be nearly constant throughout the session.

To model the radio channel conditions and loss characteristics, bit–error patterns are used, that were captured in different real or emulated mobile radio channels. The error patterns reflect different speeds and different QoS requests on the link layer. The bit–error patterns are captured above the physical layer and below the link layer layer such that in practice they act as the physical layer simulation. As only entire link layer packets of size $k_{\text{LLP}} = 640$ bit are checked, the bit–error pattern can be mapped to a loss process of user data of length $k_{\text{LLP}} = 640$ bit. The TTI depends on the supported data rate, for the most common service with bit-rate $R = 64$ kbit/s the TTI results in $T_{\text{TTI}} = 10$ ms. The characteristics bit error rate (BER) and LLC error rate (LER) for the different bit–error patterns, together with the target transmission mode, are shown in Table 2.4.

Table 2.4: Characterization of 3G link layer patterns [RSL+01].

Pattern	bit-rate R	Speed	T_{TTI}	k_{LLP}	BER	LER	Mode
1	64 kbit/s	3 km/h	10 ms	640 bit	$9.3 \cdot 10^{-3}$	$1.2 \cdot 10^{-1}$	AM
2	64 kbit/s	3 km/h	10 ms	640 bit	$2.9 \cdot 10^{-3}$	$3.6 \cdot 10^{-2}$	AM
3	64 kbit/s	3 km/h	10 ms	640 bit	$5.1 \cdot 10^{-4}$	$1.1 \cdot 10^{-2}$	UM
4	64 kbit/s	50 km/h	10 ms	640 bit	$1.7 \cdot 10^{-4}$	$3.7 \cdot 10^{-3}$	UM
5	128 kbit/s	3 km/h	5 ms	640 bit	$5.0 \cdot 10^{-4}$	$1.2 \cdot 10^{-2}$	UM
6	128 kbit/s	50 km/h	5 ms	640 bit	$2.0 \cdot 10^{-4}$	$3.5 \cdot 10^{-3}$	UM

The bit-errors in the files are statistically dependent, as channel coding and decoding included in 3G systems produces burst errors. Patterns 1 and 2 are mostly suited to be used in conjunction with the AM for video streaming applications, LER up to 12% are typical. Patterns 3 to 6 are meant to simulate a more reliable, lower error–rate bearer that is required in conversational applications with LERs around and below 1%. Note that with higher speed (50 km/h) the channel tends to have lower error–rate than in case of the walking user (3 km/h) as the fading process is faster and

the channel is more ergodic. For the AM, the retransmission delay δ_{RT} depends on the system configuration, but is in general below 100 ms.

EDGE Link Layer and System Modelling

In the 3G model used in the standardization process of H.264/AVC, it is assumed that the transmission conditions are fairly constant for the transmission time of the video. This is a valid assumption to test error resilience features applied to individual frames and short sequences. Still, this model cannot appropriately describe long–term effects such as location–dependent variations of the receive power and overall interference scenarios. However, the performance of real–time video applications in these varying scenarios is of major interest, especially in case of offline encoded video as for example in case on demand streaming.

Therefore, it is suitable to define appropriate models which reflect the properties on higher levels taking into account these long–term effects. We will in the following focus on EGPRS as an appropriate description as well as a modelling is available. However, most of the presented concepts are applicable to other wireless systems with slight modifications and parameter adjustments.

In order to model time-varying EDGE radio channels in real–time, a model has been developed and proposed in [CCCQ00, LLC02] which allows to describe and model short–term as well as long–term effects in the EDGE radio system. The system simulation model basically consists of three levels which reflect typical physical and system properties. We will modify and generalize this model slightly to emphasize the important aspects for our scenario.

The top level of the system simulation model considers the overall interference scenario of the given link, i.e., position of the user in terms of the path loss to the serving base versus the path losses to interfering bases. For example, in [CCCQ00], users in good locations, e.g., close to the base station, are represented by one group with high transmission rates, whereas users with poor receiving condition, e.g., close to the boundary of the cell, are represented by another group. More groups are in general possible. As we mainly deal with real–time applications which require a fixed resource allocation within a certain period of time, we assume that fixed radio resources are assigned to the user under investigation. Therefore, for our model the top level also considers the total amount of these available system resources in terms of time slots available for this specific user, denoted as $n_{\text{slot}} = 1, \ldots, N_{\text{slot}}$ with $N_{\text{slot}} = 8$ for the EGPRS. In [CCCQ00] it is proposed, that upon entering the radio system, a mobile station is randomly assigned to one of the two groups, according to the grouping probabilities. After several seconds, in the following denoted as T_G, it is suggested to randomly reassign a certain group. Similarly, the available number of slots n_{slot} can be re–assigned at group change instances.

The second level characterizes system configurations such as the applied power control, the velocity of the user, the interference conditions, and other system dynamics. This is introduced in the model by states basically corresponding to coding schemes in EGPRS. The characteristics of each state are further described in the third level. Transitions from one state to another model the interference dynamics. Transitions are modelled to occur after N_{TTI} multiples of the TTI T_{TTI}. Finally, the lowest level specifies the transmission condition in a certain state, summarized as $\mathcal{W}$,

Table 2.5: Payload size for applied EGPRS coding schemes.

state	$\mathcal{W}_1$	$\mathcal{W}_2$	$\mathcal{W}_3$	$\mathcal{W}_4$	$\mathcal{W}_5$
payload size k_{LLP} in bit	220	440	660	880	1100

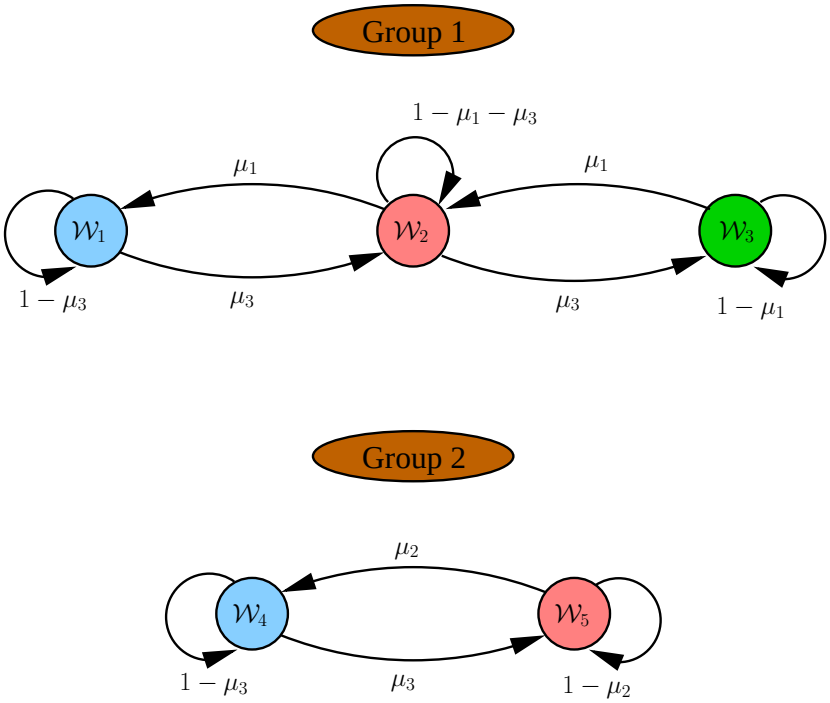


Figure 2.15: EDGE link layer channel model as proposed in [CCCQ00]: Two-group, five-state Markov channel model.

in the same way as already introduced for the 3G model. In EGPRS, each state $\mathcal{W}_i$ basically represents a coding scheme. Thereby, the transmission time interval is fixed to $T_{\text{TTI}} = 20$ ms for all states. The link transmission delay δ_{LLP} is in the range of the TTI, but will be ignored in the following as the influence is negligible. Short–term fading, interference and noise result in losses of link layer packets. However, due to the frequency hopping applied in EGPRS after each TTI, the losses of two subsequent link layer packets are well described by statistical independence, i.e., for the description of the loss process $\mathcal{L}_{\text{LLP}}$ in each state i it is sufficient to specify the loss probability $p_{\text{LLP},i}$. The size of the link layer packet, k_{LLP}, depends exclusively on the coding scheme in EGPRS. Selected payload sizes in EGPRS are shown in Table 2.5[18]. For multislot transmission, it is observed that the link layer packet losses within one TTI are highly correlated as the coherence time T_{coh} of the channel is in general much higher than one TTI. Therefore, we simplify the model such that either all n_{slot} link layer packets within one TTI are received correctly or lost. Hence, in case of multislot transmission with n_{slot} time slots, the size of the link layer packet is multiplied by the number of slots, i.e., $k_{\text{LLP}} \rightarrow n_{\text{slot}} k_{\text{LLP}}$.

Figure 2.15 describes the radio link model specified by a two-group five-state Markov chain according to [CCCQ00]. The radio system is completely characterized by the payload size for each state k_{LLP}, the link layer packet error rate p_{LLP} which is independent of the state, the state transition probabilities μ_1, μ_2 and μ_3, and finally, the group probabilities $p_{G,1}$ and $p_{G,2}$. The parameters depend on radio system configurations such as frequency reuse pattern, power control usage, the number of users per sector, etc. An exemplary set of parameters for an EDGE radio system to be used in this work is presented in Table 2.4.4. For completeness, we also provide the stationary

[18]Note that EGPRS specifies nine coding schemes rather than five and the payload sizes in the standard are slightly different. However, in [CCCQ00] these parameters values have been proposed.

state probabilities $p_{\infty,i}$ as

$$\mathbf{p}_{\infty,i} \triangleq \{p_{\infty,i}\}_{i=1}^{5} = \left\{ \frac{(1-p_{G,2})\frac{\mu_1}{\mu_3}}{1+\frac{\mu_1}{\mu_3}+\frac{\mu_3}{\mu_1}}, \frac{1-p_{G,2}}{1+\frac{\mu_1}{\mu_3}+\frac{\mu_3}{\mu_1}}, \frac{(1-p_{G,2})\frac{\mu_3}{\mu_1}}{1+\frac{\mu_1}{\mu_3}+\frac{\mu_3}{\mu_1}}, \frac{p_{G,2}}{1+\frac{\mu_2}{\mu_3}}, \frac{p_{G,2}}{1+\frac{\mu_3}{\mu_2}} \right\}. \quad (2.49)$$

If we further assume to operate in persistent AM, i.e., lost link layer packets are re–transmitted

Table 2.6: Radio system parameters for EGPRS with frequency hopping, frequency reuse 1/3, and radio aware power control.

users/sector	$p_{G,2}$	μ_3	μ_1	μ_2	p_{LLP}	$\overline{R}/n_{\text{slot}}$
2	0.93	0.3	0.055	0.05	0.11	38.2 kbit/s
8	0.64	0.3	0.094	0.3	0.20	33.7 kbit/s
15	0.28	0.3	0.27	0.59	0.27	24.8 kbit/s

until they correctly received, the channel becomes reliable but with a variable bit-rate whereby the variations occur in short–time ranges due to re–transmissions of lost link layer packets, as well on longer ranges due state and group changes. The average supported bit-rate of this channel in case of single slot transmission is given as

$$\overline{R} = n_{\text{slot}} \sum_{i=1}^{5} k_{\text{LLP},i} p_{\infty,i} \left(1 - p_{\text{LLP}}\right), \quad (2.50)$$

and is also provided in Table 2.4.4 for different system settings. For the AM, also the time interval between the transmission of the primary link layer packet and the retransmission packet, δ_{RT}, is of interest. In general, this value is quite small relatively to application time constraints, especially if the request for retransmission is immediately sent from the receiver to transmitter after the receiver has detected the loss of a link layer packet. Furthermore, the transmitter prioritizes re–transmission packets over new data and we can assume that retransmissions are instantaneous. Therefore, we assume in the remainder that for the EDGE model that the roundtrip time for retransmission requests is negligible, i.e., $\delta_{\text{RT}} = 0$.

2.4.5 Application Layer Channel Characteristics and Models

The most relevant parameter for the application when transmitting video over a lossy channel is the experienced loss or delay of a certain data unit $\mathcal{P}_n$ on the way from the transmitter to the receiver. Assuming that application layer data units are directly mapped to RTP/UDP/IP packets as indicated in Figure 2.10, the application is mainly concerned with the loss and delay characteristics of these application layer packets. However, the collection of appropriate traces for Internet packets as well as the modelling of the statistics itself is a challenging task, see e.g. [JS00, KG04]. For this purpose, in the standardization of video coding algorithms within the ITU-T it has been decided to use relatively simple but still meaningful models based on representative trace files. We will concentrate on these models in the following.

Internet Backbone Performance

Document Q15-I16 [Wen99] contains four error patterns which were carefully selected out of several hundred similar files that were all obtained by experiments on the Internet backbone. These

collected traces reflect the performance of the backbone rather than a bandwidth–limited, non–overprovisioned access links as discussed in the previous section. More information on the experiments conducted and an evaluation of the results is also found in [Wen99]. The authors summarize major observations, namely that neither an obvious relationship between the packet size and the packet loss rate nor between the bit-rate and the packet loss rate can be observed; packet losses occur randomly or in short bursts, and packet loss rates depend on the connection. Therefore, for each of four different connections a trace file is provided which is named by the average error rate in percent, namely 3, 5, 10, and 20. In Figure 2.16 the probability that the packet delay $\delta^{(\mathcal{C})}$ exceeds some value Δ is plotted for the different error files. For sake of unique presentation, we model the loss of an application layer packet by an infinite channel delay, i.e., $\delta^{(\mathcal{C})} = \infty$.

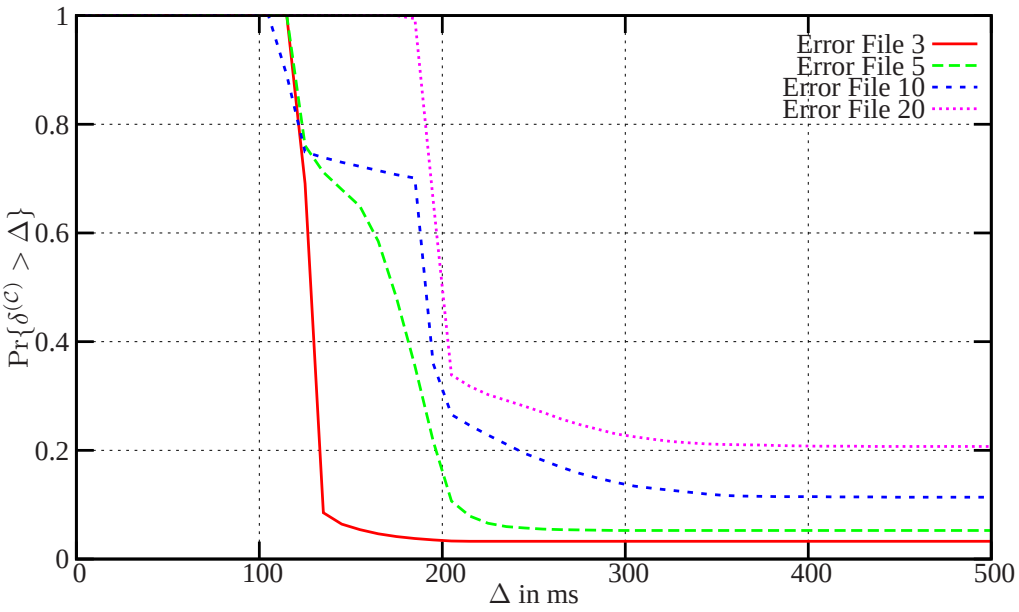


Figure 2.16: Probability distribution of packet delay $\delta^{(\mathcal{C})}$ exceeding some value Δ for different error files according to [Wen99].

From this graph it is obvious that conversational video over a channel with these properties is quite challenging. First of all depending on the connection the packet loss rate is never zero and is between approximately 3% and goes up to over 20% for bad connections. In addition, with the end–to–end delay ΔT in mind for conversational applications of about 250 ms and the distribution of the channel delay, re–transmission are completely infeasible as in any case a constant packet delay of at least $150 - 200$ ms is added. Even more difficult is the problem that the channel delay $\delta^{(\mathcal{C})}$ is close or even exceeds the maximum ΔT for significant amount of packets. This one the one hand requires that all other delay components, namely encoding delay, encoder buffer delay, decoder buffer delay, and decoding delay are kept as low as possible. In addition, an appropriate handling of delayed packets has to be introduced as a strict use of the presentation time stamps with a requested end–to–end delay ΔT below 250 ms would increase the rate of useless packets to about 30% for error file 20 and to 20% for error file 10. In this case, it might be preferable to accept some disturbances in the fluent motion rather than loosing to many packets.

Performance of IP over 3GPP Dedicated Channel with Unacknowledged Mode

As already outlined in section 2.3.1, 3GPP has agreed on a IP-based protocol stack for packet-switched 3G mobile services. Figure 2.10 shows typical packetization of data units encapsulated in RTP/UDP/IP through the 3G user plane protocol stack. UMTS commonly uses RLC–PDUs of size 80 bytes as specified in Table 2.4 such that segmentation of typically larger IP packets is necessary. At the receiver, if any of the RLC–PDUs containing data of a certain encapsulated data unit has not been received correctly, typically the entire data unit is discarded to avoid error propagation through the network or the protocol stack.

Therefore, we investigate the influence of the length of a data unit, $l_{\mathcal{P}}$, on the loss probability when transmitting over a 3GPP dedicated link. For this purpose we use the error patterns introduced in subsection 2.4.4 and specified in more detail in Table 2.4. Figure 2.17 shows the average packet loss rate ε over the packet size $l_{\mathcal{P}}$ for different UMTS scenarios as specified in Table 2.4.

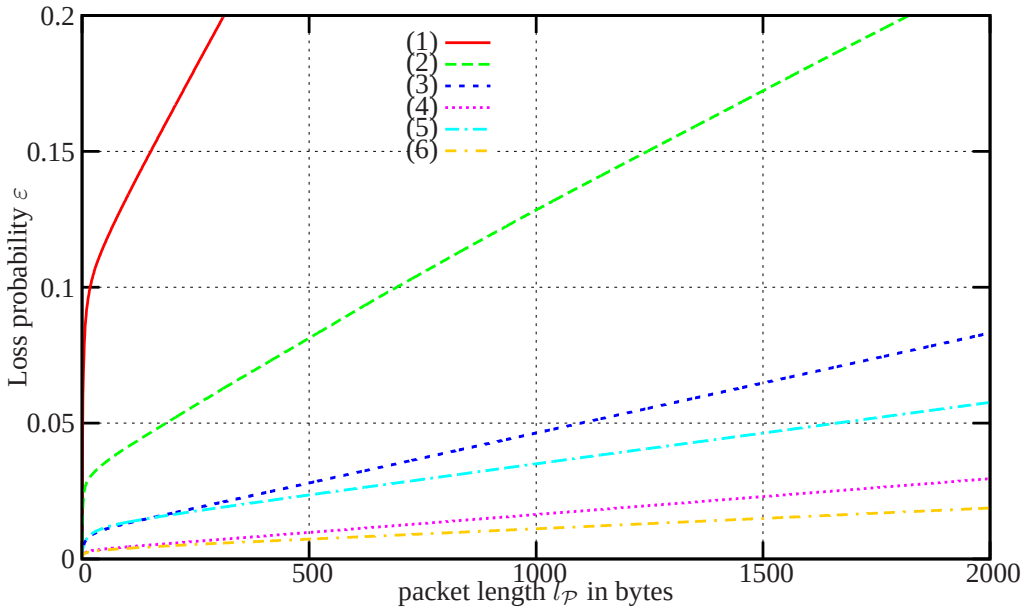


Figure 2.17: Average packet loss rate ε over the packet size $l_{\mathcal{P}}$ for different UMTS scenarios as specified in Table 2.4.

Bearers such as represented by (1) and (2) are unrealistic when used in with the UM as an acceptable quality cannot be achieved with such high error rates without retransmissions even for very short application layer packets. Bearers (3)-(6), however, are more reliable and are therefore more suitable using the UM. The loss probability for reasonable packet sizes up to $l_{\mathcal{P}} = 1000$ bytes is below 5%. This means that typical fixed Internet packet loss probabilities are not exceeded. Note that with higher speed (50 km/h) the channel tends to be ”more ergodic” than in case of the walking user (3 km/h). Therefore, the error rates are usually higher for slowly moving users than for fast moving users.

Performance of IP over 3GPP Dedicated Channel with Acknowledged Mode

Commonly, data services in mobile systems use reliable bearers with retransmissions. However, in case of applications with real–time constraints the retransmissions might result in undesired

latencies. Therefore, we have investigated the channel delay distributions when transmitting real–time data over a bearer using AM. In contrast to the results for Internet backbones as shown in figure 2.16 where the delay distribution is basically independent of individual packet sizes and also the bit-rate, in case of dedicated channels the packet size as well as the bit-rate has significant influence.

To obtain meaningful results for the AM bearers (1) and (2) in Table 2.4 it is assumed that application layer packets $\mathcal{P}_i$ are generated with packet frequency 10 Hz and constant frame size $l_{\mathcal{P}} = R/(10\text{s}^{-1})$ with R chosen appropriately. Figure 2.18(a) shows the probability distribution of packet delay $\delta^{(\mathcal{C})}$ exceeding some value δ for bearer (1) in table 2.4 for an application bit-rate of $R = 52$ kbit/s and different operation modes. In this case time–stamp based transmission is assumed, i.e., if no data is available at sending time the dedicated channel transmits dummy information. The minimum delay in this case is about 80 ms. In case of no retransmissions, corresponding to the UM, application layer packets are lost with an unacceptable ratio of 30%[19]. Retransmission are considered with different retransmission delays, namely $\delta_{\text{RT}} = 40$ ms and $\delta_{\text{RT}} = 100$ ms. The introduction of just one retransmission per RLC-PDU results in a significant decrease of the packet error rate if some additional delay can be accepted. For instance, if channel delays of 250 ms can be accepted the error rate decreases to about 5% whereby the retransmission delay has relatively little influence. Error–free performance can be achieved using the fully persistent AM for delays as low as about 500 ms. The channel delays for this AM, especially for the non–persistent mode might allow to use retransmissions even for conversational applications. However, these results only take into account the delays caused by the wireless links. If additional delays are added by the backbone network, or even by a second wireless link in the communication setup, retransmissions are no more feasible for conversational applications. The situation is different for streaming

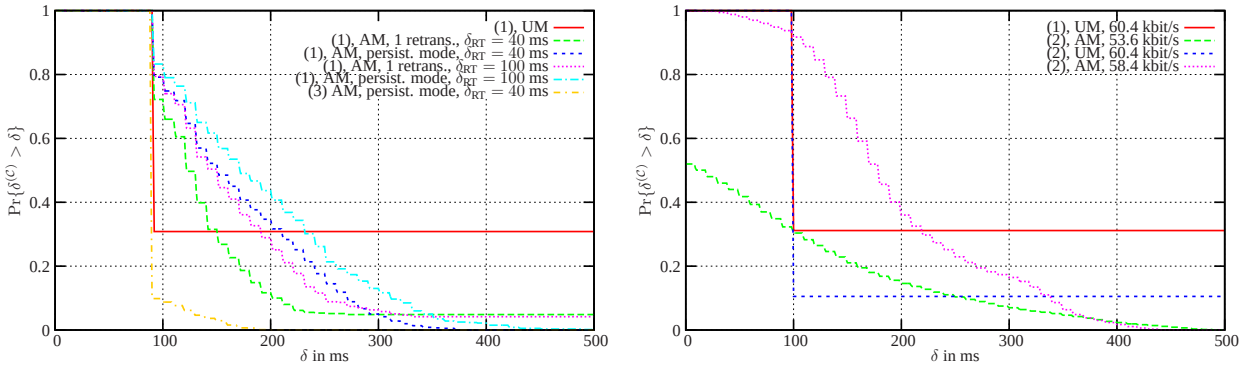


Figure 2.18: Probability distribution of packet delay $\delta^{(\mathcal{C})}$ for (a) bearer (1) and (3) in table 2.4 for an application bit-rate of $R = 52$ kbit/s, timestamp–based transmission and different operation modes; (b) for bearer (1) and (2) in table 2.4, application bit-rate matching the channel throughput and ATS.

applications: The results for the persistent AM in Figure 2.18(a) and the typical streaming bearer (1) are promising as delays in the range of 500 ms can be compensated with sufficient receiver buffers. When using bearer (3), designed for better QoS, one can see that if a channel delay of about 200 ms is acceptable, virtually error–free transmission is possible. This mode might be even attractive for conversational applications. Timestamp–based transmission requires that even for

[19]The loss of an application layer packet is again modelled by an infinite channel delay.

CBR encoded video the bit-rate has to be reduced below the average throughput as there exist periods where no data is available for transmission.

In contrast, for ATS the video can basically be encoded to fully exploit the channel throughput as the encoder buffer never underruns. This is shown in Figure 2.18(b) where the probability distribution of packet delay $\delta^{(\mathcal{C})}$ for bearer (1) and (2) in table 2.4 for an application bit-rate matching the channel throughput is shown. For bearer (1) the application bit-rate can be increased by about 10% compared to TBS at the same initial delays of about 500 ms to become error–free. The possible throughput is even higher for bearer (2) with lower error rates. Obviously, the highest application bit-rate is possible with the UM, but the error rates in this case are not acceptable.

Performance of IP over EGPRS with Acknowledged Mode

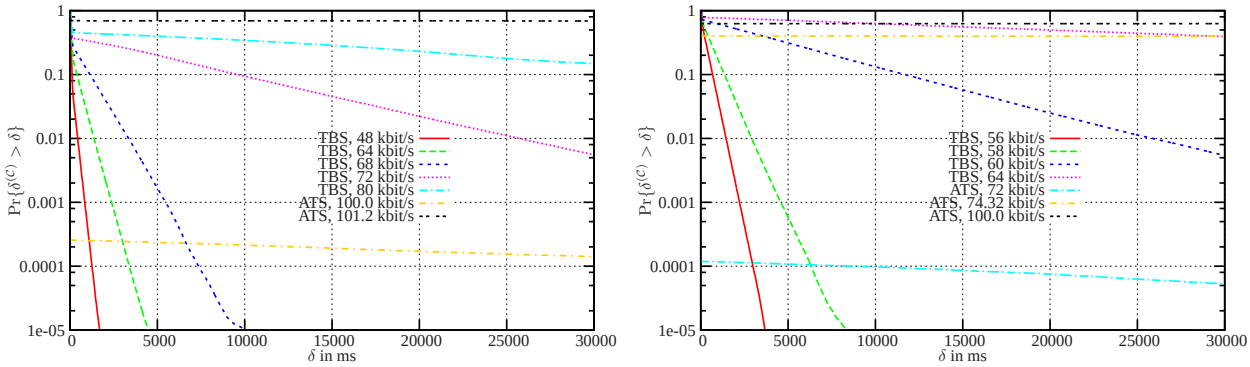
In a similar manner as for the 3GPP scenario, we also present delay distributions for the selected parameters of the EGPRS channel model as introduced in subsection 2.4.5. Again it is assumed that application layer packets $\mathcal{P}$ are generated with packet frequency 10 Hz and constant frame size $l_{\mathcal{P}} = R/(10\text{s}^{-1})$ with the bit rate R chosen appropriately. Figure 2.19 shows the probability for the packet delay $\delta^{(\mathcal{C})}$ exceeding some value δ for three combined time–slots. More specifically, Figure 2.19(a) considers 8 users in the cell according to Table 2.4.4 for different application bit-rates and different streaming operation modes. The average bit-rate for this EGPRS channel results in 101.1 kbit/s. However, for TBS only bit-rates up to about 68 kbit/s can be supported with reasonable delays, for higher bit-rates the delays already grow significantly. In contrast, for ATS bit-rates almost up to the average channel bit-rate can be supported with reasonable delays and very low error rates. However, with application data rates approaching or exceeding the average bit-rate, the delays grow and the support of streaming applications cannot be guaranteed. In addition, for ATS it is obvious that many data units arrive at the receiver long before the their decoding time is due. In addition to the necessary receiver buffer size it also worth to note that although the average bit-rate of the channel has to be maintained by the video, the constant bit-rate for the application by sending packets of equal size is not essential.

Finally, it also worth to note that although application bit-rates about up to 68 kbit/s work well for timestamp–based transmission and up to 100 kbit/s can be supported with ATS, the situation changes for different system setups, e.g., for the 15 users according to Table 2.4.4 as shown in Figure 2.19(b). The average throughput in this case is 74.4 kbit/s. In this case TBS is feasible up 58 kbit/s and ahead–of–time streaming up to about 72 kbit/s. The bit-rates supported by the 8 user case result in non-acceptable delays in case of 15 users.

2.5 Motivation for the Work

The treatment of an IP-based wireless data bearer just as any other hop in an IP networks as indicated in Figure 2.11 hides many properties and the performance of the end-to-end application will for sure be sub-optimal. This is two-fold: First, especially the wireless link is challenging in terms of bit-rates, loss rates and delay such that the end-to-end quality will mainly be determined by the performance of this link. Secondly, wireless links usually can provide a significant amount of QoS which provides optimization potentials to enhance the end-to-end performance.

For this purpose we adopt some recent research trends by specifically focussing on the entrance to the wireless system. We assume that the system architecture is extended by a proxy which has direct access to the wireless transmission system. Figure 2.20 shows the concept of the proposed



(a) (b)

Figure 2.19: Probability distribution of packet delay $\delta^{(\mathcal{C})}$ for different application bit-rates, timestamp–based transmission and ATS; (a) 8 users according to Table 2.4.4, (b) 15 users according to Table 2.4.4.

end–to–end architecture: We assume that the base station contains a proxy with slightly or significantly more intelligence than a simple router. Any kind of the abstract information as presented in Figure 2.1 available at this proxy will allow to improve the end-to-end quality of the multimedia service. The proxy can be as simple as a router, but it can be as powerful as providing transcoding on the application to fully adapt to the underlying system constraints. In addition, the proxy might contain sufficient memory to for example pre-fetch data from a server such that the appropriate source data is available in time. In this work, we will discuss different system designs for different wireless applications possibly taking into account any kind of available information as shown in Figure 2.1. However, a main contribution of this work is the extraction and definition of in general, simple cross-layer interfaces to describe the information such that it would be feasible to transfer this information across layer boundaries as well as between different entities, i.e., between the transmitter and the receiver. In general we will neither discuss the measurement nor the specific syntax of the information flow. However, due to the simplicity of the design, it is straight-forward to integrate such information in future protocol designs.

Having this information available, we will provide several improvements of source coding, specifically video coding algorithms. We will look at standard-compliant solutions mainly based on H.264/AVC as well as on extensions providing further potentials. Moreover, we will introduce adapted error protection schemes which allow easy adaptation to source and channel conditions. In any case, system optimizations are performed such that the available information is taken into account in the selection of available options. Despite extensive simulation results have been performed to verify the concepts, it is also essential to understand that in any case the objective of the work is not the definition of new systems, but to provide insight in the optimization potentials of video transmission over wireless networks. The concluding remarks will highlight the potentials for research and development to improve such services in the future.

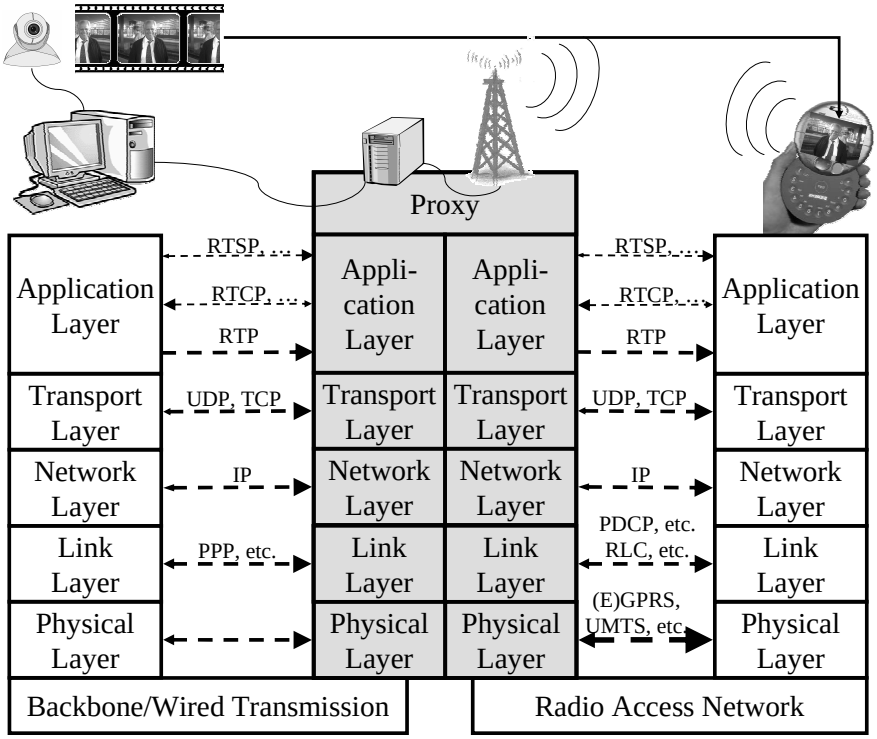


Figure 2.20: Advanced wireless multimedia end–to–end architecture: the base station contains a proxy with slightly or significantly more intelligence than a simple router.

3

Source and Video Coding Toolbox

3.1 Source Coding Preliminaries

3.1.1 Theoretical Background

We have already introduced some basics on video coding and video transmission in section 2.1. Video signals, if sampled and preprocessed appropriately, can be decomposed into a sequence of frames, Macroblocks (MBs), or pixels. For better understanding different video transmission systems we will discuss source coding based on simplified source and source coder models in this section before presenting specific video coding tools in greater detail in the remainder of this chapter. For details on quantization and source coding basics, the interested reader is for example referred to [GN99].

In practice, a source coder is applied to a sequence of real data. However, in theory the source can be abstracted as follows. Let $S = (S_1, \ldots, S_\kappa)$ be a κ-dimensional random variable with probability density function f_S supported in $\mathcal{R}_S$, a closed bounded subset of $\mathbb{R}^\kappa$ with nonempty interior. Without loss of generality we assume the source to have unit variance. Practical source coding is commonly based on two principles, removing of redundancy and irrelevancy. If only the former principle is applied the original and the reconstructed source are identical. However, if also irrelevancy is removed, the original source data and the reconstructed source data in general differ. A fidelity criteria can be applied to quantify the difference. As all source coding algorithms used in this work apply both, redundancy and irrelevancy reduction, we formalize source coding as follows.

A κ-dimensional *source coder*, also referred to as quantizer, consists of a set of decision regions $\mathcal{R} = \{\mathcal{R}_i, i \in \mathcal{I}\}$, where the index set $\mathcal{I}$ is a collection of integers beginning with 1 in our case, together with a set of reproduction points $\{q_i, i \in \mathcal{I}\}$. Then, the source coder can be defined as

$$\mathcal{Q}(S) \triangleq \sum_{i \in \mathcal{I}} q_i \mathbf{1}\left\{\mathcal{R}_i \in S\right\}. \tag{3.1}$$

The goodness of a source coder can be measured by comparing the reproduction $\hat{S} \triangleq \mathcal{Q}(S)$ with

the original S. Although subjective cost functions are usually preferable for many applications, simple computable distortion measures can provide sufficient insight in the performance of source coders. Therefore, a distortion measure $d(s,\hat{s})$ is used that quantifies cost or distortion when reproducing s with $\hat{s} = \mathcal{Q}(s)$. The p-th power distortion $d(s,\hat{s}) = |s-\hat{s}|^p$ and especially the squared error with $p=2$ are the most commonly used distortion measures. Often a single measure is desired to describe the goodness of a source coder $\mathcal{Q}$ for a certain source S. For this purpose the normalized expected distortion

$$D_S(\mathcal{Q}) = \frac{1}{\kappa}\mathbb{E}\left\{d(S,\mathcal{Q}(S))\right\} = \frac{1}{\kappa}\sum_i \int_{\mathcal{R}_i} d(s,\mathcal{Q}(s)) f_S(s) ds, \tag{3.2}$$

is applied quite commonly. If the distortion is measured by the squared error, then $D_{\mathcal{Q}}(S)$ becomes the MSE. However, the source coder $\mathcal{Q}$ is not entirely characterized by the expected distortion as the produced indices i have to be stored or transmitted. To be more specific, a source coder basically consists of three components, a lossy source encoder $\mathcal{Q}_{\mathrm{e}} : \mathcal{R}_S \rightarrow \mathcal{I}$ with $\mathcal{R}_S \subseteq \mathbb{R}^\kappa$, a source decoder $\mathcal{Q}_{\mathrm{d}} : \mathcal{I} \rightarrow \mathcal{R}_{\hat{S}}$ where $\mathcal{R}_{\hat{S}} \subseteq \mathbb{R}^\kappa$ and an invertible mapping $\mathcal{Q}_{\mathrm{m}} : \mathcal{I} \rightarrow \mathbb{B}$ with $\mathbb{B} = \{\mathcal{Q}_{\mathrm{m}}(i), i \in \mathcal{I}\}$ is a collection of, in general, variable length binary vectors fulfilling the prefix condition. Then, the instantaneous rate of the source coder $\mathcal{Q}$ applied to source s is defined as

$$r_s(\mathcal{Q}) \triangleq \frac{1}{\kappa} l\left\{\mathcal{Q}_{\mathrm{m}}(\mathcal{Q}_{\mathrm{e}}(s))\right\}, \tag{3.3}$$

where $l\{.\}$ denotes the length in bits of a certain code word. For an important specific case, referred to as fixed–rate or constant–rate source coder, each index has same length $r = \lceil \log_2(|\mathcal{I}|) \rceil$ for all binary codewords. Similarly to the distortion, the performance of a source coder, can additionally be characterized by the normalized expected rate as

$$R_S(\mathcal{Q}) = \frac{1}{\kappa}\mathcal{E}\{r(S)\} = \frac{1}{\kappa}\sum_i l\left\{\mathcal{Q}_{\mathrm{m}}(i)\right\} \int_{\mathcal{R}_i} d(s,\mathcal{Q}(s)) f_S(s) ds. \tag{3.4}$$

Therefore, the performance of every source coder $\mathcal{Q}$ for a certain source S can be described with a rate–distortion pair $(R_S(\mathcal{Q}), D_S(\mathcal{Q}))$. In the following we drop the source index S and write $(R(\mathcal{Q}), D(\mathcal{Q}))$ whenever any ambiguity is excluded.

A major goal of any compression system is to optimize the rate–distortion tradeoff. Assume for the moment that the we are constrained by the source dimension κ. Then the tradeoff can be formalized by finding a source coder $\mathcal{Q}$ which minimizes the distortion for a constrained rate R as

$$\mathcal{I}(R) = \inf_{\mathcal{Q}:R(\mathcal{Q})\leq R} D(\mathcal{Q}). \tag{3.5}$$

Similarly, a rate-distortion formulation is possible yielding the minimum rate for a constrained distortion. Recently, also the approach based on an unconstrained optimization using a Lagrangian approach has been gotten more and more popular. In this case, the weighted distortion–rate function is defined as

$$J(\lambda) = \inf_{\mathcal{Q}} \left(D(\mathcal{Q}) + \lambda R(\mathcal{Q})\right), \tag{3.6}$$

where λ is a nonnegative number. All these formalizations are essential equivalent, the distortion–rate and rate–distortion functions are duals and every distortion–rate pair on the convex hull of these curves corresponds to the unconstrained formulation for some value λ.

So far we have considered the dimension κ of the source coder to be fixed. However, in the source coding theory as well as in practical applications, the parameter κ might be chosen to

tradeoff complexity or delay versus performance. Let us denote the operational distortion–rate function for dimension κ and rate R as $\mathcal{I}_\kappa(R)$. Then, the operational distortion–rate function for an arbitrary dimension is defined as

$$\overline{\mathcal{I}}(R) \triangleq \inf_{\kappa} \mathcal{I}_\kappa(R), \tag{3.7}$$

being the smallest achievable distortion for any practical source coder at rate R. For an i.i.d. source $\{S_n\}$ Shannon's distortion–rate function $D(R)$ [Sha93a] is defined as the minimum average distortion $\mathbb{E}\left\{d(S,\hat{S})\right\}$ over all conditional distributions of $\hat{S}$ given S for which the mutual information $I(S;\hat{S})$ is at most rate R. Shannon showed that for every rate R, $\overline{\mathcal{I}}(R) = D(R)$ holds. This can be interpreted such as no source coder of any dimension κ with rate R could yield smaller distortion than $D(R)$ and that for some dimension κ there exists a source coder with rate smaller equal R and distortion $D(R)$. In addition, it was shown [Gal68, p. 112] that

$$\overline{\mathcal{I}}(R) = \lim_{\kappa\to\infty} \mathcal{I}_\kappa(R), \tag{3.8}$$

meaning that with high–dimensional source coders the distortion–rate function $D(R)$ can be achieved. Although for some specific scenarios practical source coder with limited dimension might exist which allow achieving $D(R)$, in general an infinite dimension κ is necessary for coders to perform optimally.

A different approach to bound the performance of practical coders uses high–resolution theory. According to Zador [Zad82], the operational rate–distortion function for a κ–dimensional source coder can be estimated as

$$\mathcal{I}_\kappa(R) \simeq \vartheta_\kappa \beta_\kappa 2^{-pR}, \tag{3.9}$$

whereby the "$\simeq$" indicates that the rate R has to be sufficiently large. More specifically, large rate means that the overload distortion is ignored and the density f_S might be approximated as a constant on a small interval. Thereby, ϑ_κ, usually referred to as Gersho constant, is the least normalized moment of inertia of a κ-dimensional tessellating polytopes. Gersho is acknowledged as he made a widely accepted conjecture [Ger79] that when the rate is large, most cells of κ-dimensional source coder with minimum or close to minimum p-th distortion are approximately congruent to some basic κ-dimensional cell shape, e.g. for $\kappa = 1$ to an interval, for $\kappa = 2$ to a regular hexagon and for $\kappa \to \infty$ to a hyper–ball. For the MSE, i.e., $p = 2$, Gersho's constant is bounded as

$$\forall_{\kappa=1,2,\ldots}\, \vartheta_{\kappa=1} = \frac{1}{12} \leq \vartheta_\kappa \leq \frac{1}{2\pi e} = \lim_{\kappa\to\infty} \vartheta_\kappa.$$

The Zador factor β_κ basically takes into account that the reconstruction points can be distributed in a more clever way for higher dimensions κ. For fixed rate quantizers, the Zador factor β_κ depends on the source coder dimension κ and the source distribution f_S. For variable–rate source coders, the Zador factor only depends on the differential entropy of a sequence of random variables with the dimension being the product of the source dimension κ and the order of the entropy coding. For details and more background as well as a more rigorous approach to the high–resolution theory, we refer the interested reader to [GN99] and references therein.

Interesting enough, that the performance of many source coders $\mathcal{Q}$ for memoryless sources behaves very similar to the high–resolution bound presented in (3.9), namely

$$D_{\mathcal{Q}}(R) \simeq \hat{\beta}_\kappa 2^{-pR}, \tag{3.10}$$

where $\hat{\beta}_{\kappa.\mathcal{Q}} \geq \vartheta_\kappa \beta_\kappa$ basically determines the performance of the source coder $\mathcal{Q}$. We refer to any source coder $\mathcal{Q}$ fulfilling the rate–distortion relationship in (3.10) as "good" source coder. Examples for good source coders are scalar quantizers, many lattice quantizers, as well as optimized vector quantizers [GN99]. In Figure 3.1 the MSE[1] of different quantizers over the rate measured in bit per source component for the Gaussian and the uniform source is shown together with rate–distortion curve. The following observations are obvious:

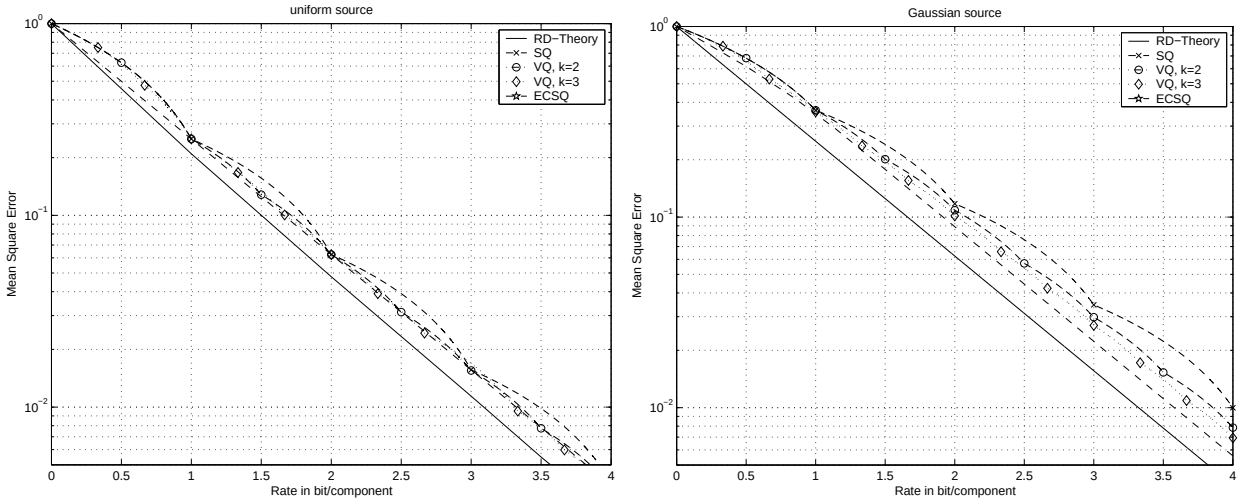


Figure 3.1: MSE of different quantizers over the rate measured in bit per source component for the uniform (left-hand side) and the Gaussian (right-hand side) source together with rate–distortion curve.

- For a uniformly distributed source no gains with increasing vector dimension are visible.
- For $k > 1$ non-integer rates can be realized.
- For a fixed rate only discrete points i/k bit with i any integer are realizable. However, with time–sharing between two discrete points the intermediate points are realizable. Note however that the averaging has to be done on the linear MSE domain, not in the logarithmic. This leads to these rather strange graphs.
- The Entropy Constrained Scalar Quantization (ECSQ) provides excellence performance whereby the distance to the optimum rate–distortion curve is as low as about 0.25 bit.
- For a Gaussian distributed source gains with increasing vector dimension are visible which can be explained by the Zador factor β_κ.

3.1.2 Rate and Quality Adaptive Source Coders

In many cases it is desired that a single source coder can operate at different rates or different qualities. The source coder should allow to adapt varying constraints, for example in terms of the desired quality, or the rate provided by the storage medium or the transmission link. Adaptivity can

[1]The performance of different quantizers has been computed using a modified version of the software package QCCPack [Fow00].

be accomplished by different means. In a straightforward approach one would design and apply a completely new source coder for each desired operation point. For example, some very good source coders for rates at 1 bit per source component have been presented, but they cannot easily be extended to other rates. Therefore, in practice more flexible source coders are usually preferred. Many practical source coding algorithm allow trading quality versus rate by just changing one or several parameters in the encoding process. Ultimately, scalable source coders provide different operation points within one binary coded representation of the source.

Regardless of the approach used we formalize the notation of rate and quality adaptive source coders in the following. We attempt to have different approximations, so-called *versions*, of a source $S = (S_1, \ldots, S_\kappa)$ being a κ-dimensional random variable with probability density function f_S supported in $\mathcal{R}_S$. We define a κ-dimensional source coder providing M different versions as a sequence of κ-dimensional source coders[2] $\mathcal{Q} = \{\mathcal{Q}^{(1)}, \mathcal{Q}^{(2)}, \ldots, \mathcal{Q}^{(M)}\}$. The source coder for each version m consists out of a set of decision regions $\mathcal{R}^{(m)} = \{\mathcal{R}_{i_m}, i_m \in \mathcal{I}_m\}$, where the index set $\mathcal{I}_m$ together with a set of reproduction points $\{q_{i_m}, i_m \in \mathcal{I}_m\}$ forms the source coder for version m. Then, the source coder for version m is defined as

$$\mathcal{Q}^{(m)}(S) \triangleq \sum_{i_m \in \mathcal{I}_m} q_{i_m} \mathbf{1}\left\{\mathcal{R}_{i_m} \in S\right\}. \tag{3.11}$$

The reconstructed source is denoted as $\hat{S}^{(m)} \triangleq \mathcal{Q}^{(m)}(S)$. The reconstruction space for each version is a subset of the entire space, i.e., $\mathcal{R}_{\hat{S}}^{(m)} \subseteq \mathbb{R}^\kappa$, but in general they differ for each version. According to the previous presentation, the adaptive source coder for each version m is separated in three components, an encoder $\mathcal{Q}_{\mathrm{e}}^{(m)} : \mathcal{R}_S^\kappa \to \mathcal{I}_m$, a source decoder $\mathcal{Q}_{\mathrm{d}}^{(m)} : \mathcal{I}_m \to \mathcal{R}_{\hat{S}}^{(m)}$, and a binary mapping $\mathcal{Q}_{\mathrm{m}}^{(m)} : \mathcal{I}_m \to \mathbb{B}_m$. To describe the performance of an adaptive source coder $\mathcal{Q} = \{\mathcal{Q}^{(1)}, \mathcal{Q}^{(2)}, \ldots, \mathcal{Q}^{(M)}\}$ we use the distortion–rate pairs

$$\{d^{(m)}(\mathcal{Q}) \triangleq d(\mathcal{Q}^{(m)}), R^{(m)}(\mathcal{Q}) \triangleq R(\mathcal{Q}^{(m)})\},$$

for each version m. Without loss of generality we can assume that the rates are monotonically increasing with increasing version number, i.e.,

$$R^{(1)}(\mathcal{Q}) < R^{(2)}(\mathcal{Q}) < \cdots < R^{(M)}(\mathcal{Q}),$$

and that for any reasonable source coder the expected distortion also are decreases over the layers, i.e.,

$$D^{(1)}(\mathcal{Q}) < D^{(2)}(\mathcal{Q}) < \cdots < D^{(M)}(\mathcal{Q}).$$

Note that this behavior is not necessarily the case for a certain realization s, but in average. In addition, if the squared error distortion is applied as distortion measure, then operational distortion–rate behavior as well as the performance for any reasonable source coder $\mathcal{Q}$ is convex, i.e.,

$$\forall_{m=1,\ldots,M-1} \quad \frac{D^{(m)}(\mathcal{Q})}{R^{(m)}(\mathcal{Q})} \geq \frac{D^{(m+1)}(\mathcal{Q})}{R^{(m+1)}(\mathcal{Q})}. \tag{3.12}$$

Figure 3.2 shows the MSE over the rate for an entropy–constrained scalar uniform quantizer over the rate measured in bit per source component for the uniform and the Gaussian source together with rate–distortion curve. The uniform quantizer is only controlled by the step size, but basically one and the same encoder can be used for rate adaptivity. From this example it is observed that the

[2] Although basically the dimension of the source coder might be different for each version, we ignore this generalization in the remainder.

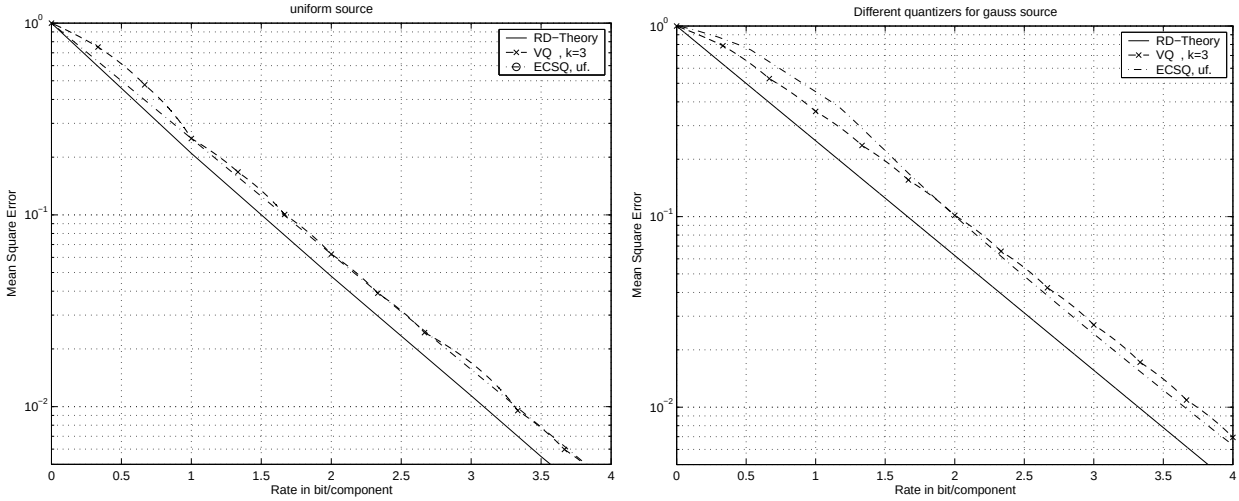


Figure 3.2: MSE of a simple entropy–constrained scalar uniform quantizer over the rate measured in bit per source component for the uniform (left-hand side) and the Gaussian (right-hand side) source together with rate–distortion curve.

simple uniform ECSQ can provide consistently good quality even compared to vector quantizers with just a simple encoder. This result motivates the use of such encoders especially in case of universal sources, i.e., in case that the source statistics are a priori unknown, and to obtain a single encoder which allows trading quality and rate.

3.1.3 Scalable and Progressive Source Coding

Although adaptive source coding allows reacting to different constraints at the transmitter side it still basically implies that the entire code word is available to the receiver. If just a very tiny part is missing, due to whatever reasons, a reconstruction is in general not possible. However, for several applications, it is desired that an improvement in the source reproduction quality can be achieved by sending only an incremental increase in rate in addition to some initial rate. Obviously, a source coder designed to operate at several intermediate rates and distortions with just a single codeword cannot outperform the source coder designed for the highest rate. Immediately one is interested if and for what circumstances there is a penalty when introducing scalability, and if yes, what is the degree of penalty.

Several synonymous or at least similar terms in the area of scalable coding are used. *Scalable Coding* usually refers to a source coder which simultaneously provides encodings of the same data source at different quality levels by extracting a lower quality reconstruction from a single binary description. Scalable coding can be realized using *embedded bit–streams*, i.e., the bit-stream of a lower resolution is embedded in the bit–stream of higher resolution. Depending on the viewpoint of scalability two variants can be considered. *Successive Refinement* addresses the view that information is added such that the initial reproduction is refined. In this case, the emphasis is on a good initial reproduction. In contrast, *graceful degradation* addresses the viewpoint when emphasis is on a good final reproduction quality, but at least an intermediate reconstruction is possible.

Whereas in general scalable coding only refers to a few quality levels, additionally the terminology of *Fine–Granular Scalability (FGS)* has been introduced to indicate that many quality repro-

ductions are inherently involved in a single binary description of a source. This FGS characteristic of is often also referred to as *progressive coding* enabling *progressive transmission*, especially if the bit–stream is completely embedded. This means that the storage format or transmission algorithm treats the source data in such a way that later data continuously adds progressively better quality to an already decoded reconstruction. Progressive coding is for example useful when an image is being sent across a slow communications channel as a low-resolution image may be sufficient to allow the user to decide not to wait for the rest of the file to be received.

In the following we generalize our framework to progressive coding, i.e., the number of layers can be high. In order to formalize progressive coding with M layers, we basically use the same notation as for adaptive source coders. However, the main difference to the adaptive source coder comes from the fact that we attempt to have M partially decodable approximations of the source S based on only prefixes of the binary word $\mathcal{Q}_{\mathrm{m}}^{(M)}\left(\mathcal{Q}_{\mathrm{e}}^{(M)}(S)\right)$ being available at the receiver. A κ-dimensional *progressive source coder with M layers* is a sequence of κ-dimensional source coders $\mathcal{Q}_{[1:M]} = \{\mathcal{Q}^{(1)}, \mathcal{Q}^{(2)}, \ldots, \mathcal{Q}^{(M)}\}$. However, in contrast to just a single index for the single layer coders as presented previously, we emphasize that the coded version of a higher layer embeds the lower versions by defining an ordered sequence of indices, i.e.,

$$i_m \rightarrow \mathbf{i}_{[1:m]} \triangleq i_1, \ldots, i_m.$$

Therefore, after applying an invertible binary mapping $\mathcal{Q}_{\mathrm{m}}^{(m)} : \mathcal{I}_{[1:m]} \rightarrow \mathbb{B}_{[1:m]}$ on each layer and concatenating the binary code words, we obtain an embedded binary representation

$$\mathcal{Q}_{\mathrm{m}}^{(1)}(i_1), \mathcal{Q}_{\mathrm{m}}^{(2)}(i_2), \ldots, \mathcal{Q}_{\mathrm{m}}^{(M)}(i_M).$$

From the viewpoint of rate–distortion theory, i.e., the source coder dimension κ is large, the problem of scalable coding was introduced under the term *divisibility* by Koshelev [Kos91], but Equitz and Cover [EC91] coined the term *successive refinement*. These first papers where concerned with the conditions under which scalable coding is feasible without sacrificing performance when compared to rate–distortion performance of a single layer source coder. A system without loss has been called *successively refinable*. For a source $\{S_n\}$ to be successively refinable it is sufficient and necessary that there exists a conditional probability $p_{\hat{S}^{(1)}\hat{S}^{(2)}\ldots\hat{S}^{(M)}\,|\,S}$ such that for all m the single layer distortion–rate function $D(R^{(m)})$ can be achieved and $S, \hat{S}^{(1)}, \hat{S}^{(2)}, \ldots, \hat{S}^{(m)}$ form a Markov chain, i.e.,

$$p_{S|\hat{S}^{(1)}\hat{S}^{(2)}\ldots\hat{S}^{(M)}} = p_{S|\hat{S}^{(1)}}\, p_{\hat{S}^{(1)}|\hat{S}^{(2)}} \cdots p_{\hat{S}^{(M-1)}|\hat{S}^{(M)}}\,.$$

Equitz and Cover also provide three examples of successively refinable sources, namely discrete–alphabet sources with Hamming distortion, Gaussian sources with squared error distortion, and Laplacian sources with absolute distortion. Also, one counterexample is provided showing that not all discrete–alphabet sources are successively refinable. Rimoldi [Rim94] generalizes results for discrete–alphabet sources by providing achievable rate regions for a given set of distortions. Effros [Eff99] further extends the work by Rimoldi to arbitrary stationary sources for fixed and variable rate source coders. More recently, Lastras and Berger [LB01] showed that for squared error distortion the rate loss is bounded by half a bit at each layer and provide more bounds on the performance for graceful degradation, successive refinement, and layered coding. More recently, Tuncel and Rose [TR03b] introduced a computational approach using an iterative algorithm to exactly compute the layered rate–distortion function. Most of the distortion–rate results indicate that the introduction of layered coding does not significantly degrade the performance. However, to

apply these approaches, both, the source dimension κ has to be high as well as the rate transmitted in each step has to be high.

For some practical cases not rate–distortion theory, but high–resolution theory approaches provide more insight into the performance of scalable and progressive transmission systems. Practical scalable source codes can be realized using Tree–Structured Vector Quantization (TSVQ) which was initially introduced to obtain lower complexity source coders [BGGM80]. In its simplest form, a κ-dimensional TSVQ encoder operating at rate R for the highest resolution makes κR binary decisions resulting in an embedded code with progressive structure. With this greedy approach targeting for good performance at low rates the losses compared to the optimal κ-dimensional source coder at high rates are between 0.11 bit and for $\kappa = 2$ and 0.37 bit for $\kappa \to \infty$. The primary weakness of TSVQ lies in shape of decision regions for higher rates being a mixture of hypercubes, hypercubes cut in half, the latter cut in half and so on. Dürst [Due97] has analyzed the disadvantage of progressive coding using binary TSVQ applying congruent polytopes as decision regions. These self–similar structures are recursively sub–dividable. Although the performance for the lower layers decreases, for high–rates the loss compared to the optimal source coder for $\kappa \to \infty$ is reduced to 0.31 bit. TSVQ can be generalized to larger branching factors than two resulting in a less progressive but more scalable coding, but possible better performance, mainly for the lower layers. A more prominent extension of binary TSVQ relies on unbalanced trees resulting in a variable rate version of TSVQ [CLG89b]. The performance losses in this case when compared to the optimal source coder are almost negligible.

In the scalable and progressive source coding approaches considered up to now a tree–structured coding was inherently assumed. The input signal to the source coder $\mathcal{Q}^{(m)}$ is in general the original source signal S. In contrast, Multi–Stage Vector Quantization (MSVQ) is constrained to just using a single coder $\tilde{\mathcal{Q}}^{(m)}$ within each layer m which operates on difference signal between the reconstructed signal on layer $m-1$ and the original signal S. MSVQ results in a significantly reduced complexity and storage requirements when compared to TSVQ. MSVQ can be defined recursively as

$$\forall_{m=1,\ldots,M} \quad \mathcal{Q}^{(m)}(S) = \sum_{n=1}^{m} \tilde{\mathcal{Q}}^{(m)} \left(S - \mathcal{Q}^{(n-1)}(S) \right),$$

and per definition $\mathcal{Q}^{(0)}(S) = 0$. Distortion–rate bounds for MSVQ have been presented by Tuncel and Rose [TR03c, TR03a] showing that common and additive successive refinement without any rate loss is possible for many continuous sources with common distortion measures. However, they also show counterexamples that there exist sources being successively refinable in a common sense, but not in additive sense.

High–resolution theory also provides some statements on MSVQ–like source coding schemes based on the derivation of the probability density of the residual signal $S - \mathcal{Q}^{(m)}(S)$ in each layer [Lee90]. Variable rate extensions to MSVQ have been presented in, e.g. [KSB95]. Another extension [Lee90] based on the observation that if the first layer in MSVQ has high rate, the decision regions all have similar shape and the source density is almost constant. Hence, for the second layer and all residual layer basically only scaling on the residual signal is necessary to obtain a good low-complexity MSVQ–like source coder performing almost as well as a TSVQ–like source coder.

In Figure 3.3 the MSE over the rate measured in bit per source component for the uniform and the Gaussian source together with rate–distortion curve for different MSVQs is shown. The lower performance of MSVQ compared to regular Vector Quantization (VQ) even for higher dimensions can be observed especially for the case of the Gaussian distribution. If looking for a

low-complexity encoding with good performance the ECSQ is preferred over MSVQ. However, at least for uniform sources the decay of the MSE with rate is almost constant even for MSVQ. In summary from the discussion and the provided example results, it can be concluded that the

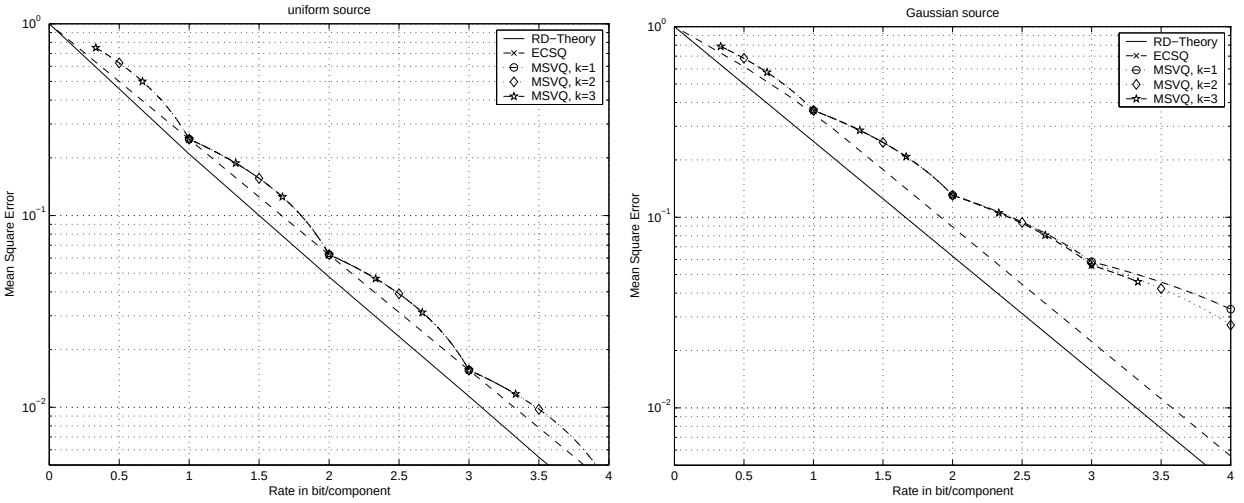


Figure 3.3: MSE of MSVQs over the rate measured in bit per source component for the uniform and the Gaussian source together with rate–distortion curve.

performance of many practical scalable and progressive source coders can be well approximated as

$$D^{(m)}(\mathcal{Q}) = \mathbb{E}\left\{|S - \mathcal{Q}^{(m)}(S)|^p\right\} \simeq \hat{\beta}_{\kappa,\mathcal{Q}} 2^{-pR^{(m)}}. \tag{3.13}$$

In the following we refer to any progressive quantizer that achieves this exponential rate of decay with $R^{(m)}$ a *good progressive quantizer*.

3.1.4 Example: Still Image Progressive Source Coding

An example of a progressive coding scheme is the still image compression algorithm Set Partitioning in Hierarchical Trees (SPIHT) [SP96] introduced by Pearlman and Said in 1996. This algorithm is an extension of Shapiro's Embedded Zerotree Wavelet (EZW) algorithm [Sha93b] and provided a significant breakthrough in lossy image compression. It basically also founded the basis for the standardization of JPEG–2000. All these algorithms allow for progressive transmission. In the following we concentrate on the SPIHT algorithm due to its simplicity and its wide usage in research papers on progressive image transmission. According to Figure 3.4, SPIHT provides substantially higher compression ratios than prior compression techniques like JPEG [PM93]. This is in contrast to some of the observations made previously that for general sources the introduction of progressive coding might result in reduced coding efficiency.

Despite of all these positive news images compressed with any of these advanced algorithms are extremely vulnerable to bit errors or data loss especially in the first part of the frame where compression is very high. Investigations [MM00] have shown that the reconstruction quality of efficient progressive source coding schemes is severely affected by residual errors. Due to the properties of progressive coding any data after the first decoding error cannot be interpreted any more. This property is amplified also due to the adaptive arithmetic coding used in SPIHT and JPEG–2000.

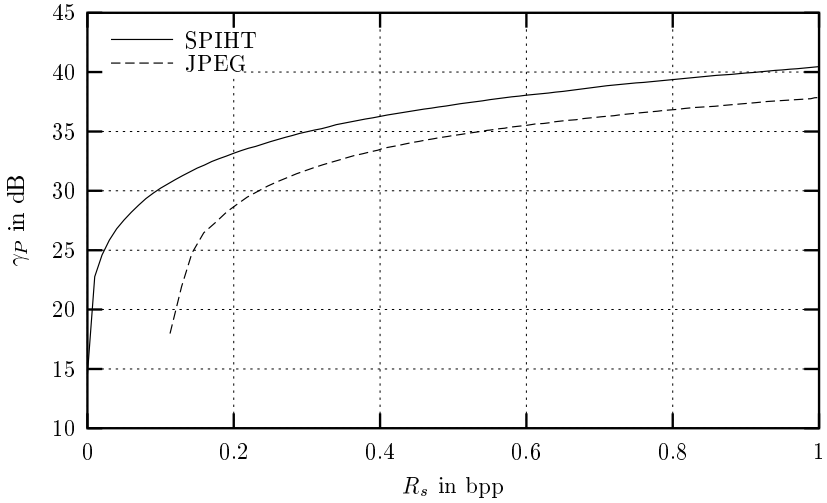


Figure 3.4: Quality in PSNR versus bit-rate in bit per pixel (bpp) for coding the gray–scale image 'lenna' (512×512 pixel) with SPIHT and JPEG.

3.2 Practical Issues in Video Source Coding

Although theory often can provide many insights in the design of a compression algorithms, practical issues often limit the applicability of straightforward source coding algorithm which promise operation close to the distortion–rate bounds, e.g. vector quantization. The major implications for practical source coding algorithms are complexity constraints, delay constraints, the goodness of a single system at many operation points, and the difficulties in the universal characterization of source and video signals. In addition, having in mind an entire video delivery system, a source coding algorithm should also be capable to react to constraints imposed by the transmission system, such as variable transmission bit-rates, delay jitter, or transmission errors.

3.2.1 Video Source Coding Basics

In the design of practical source coders and especially video coding algorithms usually not only bit-rate constraints have to be considered, but also complexity and delay constraints under which the expected quality has to be maximized. As already mentioned video compression is in general based on redundancy and irrelevancy reduction. In video coding the irrelevancy is removed by appropriately pre–processing the original signal in the spatial and temporal domain, and afterwards by applying a source coding algorithm which removes irrelevancy on the pre–processed source. Appropriate pre–processing, especially in the spatial domain, is discussed in more detail in subsection 2.1.2. In the following we will focus on the compression of pre–processed video specified by a certain spatial and temporal resolution.

Instead of applying a straightforward vector quantization approach, practically engineered video coders divide this complex problem into a set of sub–problems, which are optimized separately, but jointly adapted and improved under the common constraints bit-rate, delay, and complexity. In this way a practical video coder can be viewed as a system with optimized components and rich information flow across well-designed components to globally optimize the source coding algorithm. The statistical properties of video sources are hard to analyze especially over a long–

time and considering the infinite amount of possible input data. With this respect a universal source coding approach is much more favorable rather than having different source coders for different video sequences with different statistics. However, some lower order or local statistics are still very similar in many video signals. In addition, researchers and engineers working in the field have gained expertise over the last years to what are typical effects present in video sequences. To accommodate these different effects, more and more components have been added and existing components have been enhanced to improve the performance of video coding algorithms in the past.

In Figure 3.5 such a typical *hybrid video encoder* with the most prominent components is shown. In the following we will provide a short introduction on basic video coding along with some typical effects in video sequences. For more detail we refer the interested reader to [SW05] and references therein. The term *hybrid* emphasizes that the video coder consists of two basic compo-

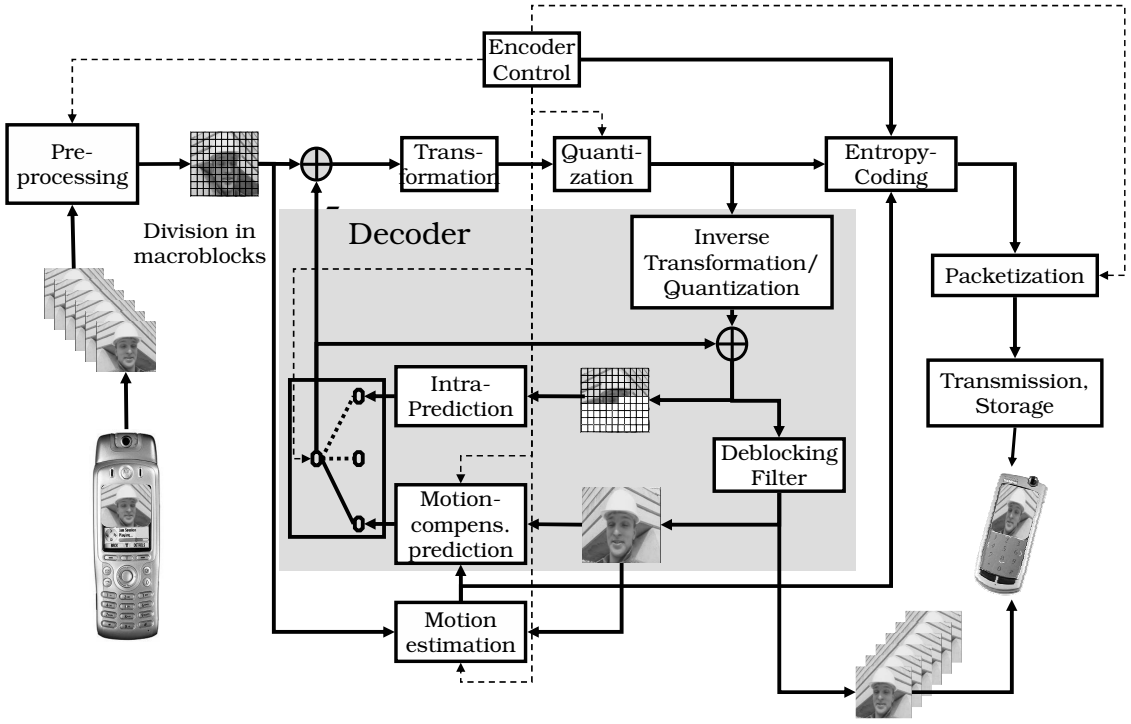


Figure 3.5: Typical video encoder.

nents, namely a prediction part and a refinement part. Although the first image of a sequence as well as the one of a random access point are coded in Intra, we concentrate on the coding of images with temporal reference. According to Figure 3.5 the pre–processed video is separated into smaller blocks according to Figure 3.6, typically into 16×16 MBs which are independently processed. When using YUV colorspace with 4:2:0 subsampling then each MB is subdivided in 4 chrominance blocks and one 1 luminance block for each color component of size where each block has size 8×8. The encoder controls the decision in the encoding process where the decisions should be taken such that the resulting rate–distortion performance is optimized. For the prediction part it can be selected between an intra–prediction mode and a Motion–Compensated Prediction (MCP) mode. The prediction data, either intra–prediction or motion data, is forwarded to the entropy process. The coding of the residual signal, the so–called Displaced Frame Difference (DFD), is accomplished by applying an intra–frame–like coder consisting of a subband–transformation,

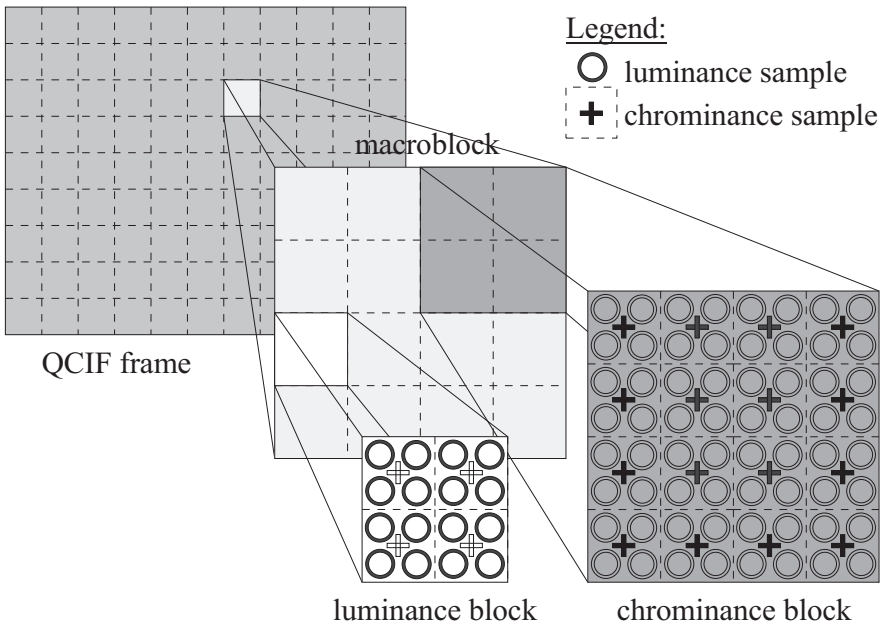


Figure 3.6: Graphical illustration of the coding hierarchy.

commonly Discrete Cosine Transform (DCT)–like, and a lossy quantization part. The quantized coefficients are forwarded to the entropy coding engine which also multiplexes the resulting binary data together with control and prediction data. A packet format suitable for transmission or storage is generated.

The encoder itself contains the most decoder operations as prediction needs to be performed on a signal which is also accessible to the decoder. For this purpose, the residual signal is reconstructed, added to the prediction signal, and the sum signal is forwarded to a de–blocking filter. This signal is the output of the decoder and if used for future reference this signal is also forwarded to the MCP–buffer.

The spatial transform is used to compact the energy to less coefficients. Although the spatial correlation is significantly reduced due to predictive coding, the de-correlation operation can still provide coding gains. The transform coefficients are further processed by a simple uniform scalar quantizer with an adjustable step size Δ which is in general the main source of distortion to the reconstructed video[3]. The remaining redundancy is exploited by using Variable Length Code (VLC) in the form of Huffman codes, Exp-Golomb codes, or arithmetic coding. Usually, the codes have a universal property such that they can be adapted to actual signal statistics.

The preference of scalar coding in combination with entropy coding over any vector quantization techniques acknowledges the extraordinary performance of ECSQ with low complexity and memory demands. However, the typical buffer problems of entropy constrained coding methods need to be addressed by rate–control algorithms and by the appropriate design of buffers. The application of block–based transforms also comes with the drawback that visual blocking artifacts are introduced in the reconstructed signal. This can partly be compensated by using smaller block transforms, e.g. 4×4, but mainly the usage of a de–blocking filter which removes blocking artifacts. If the filter is used before the reconstructed frame is forward to the reference frame buffer, it

[3]Other sources for distortion are for example the dropping isolated coefficients or the use skip mode for coding efficiency reasons and the application of filters for subjective viewing enhancements.

is referred to as in–loop deblocking filter and has also to be applied by the encoder.

A video sequence could basically be viewed as a sequence of many individual pictures and could therefore be encoded with common still image coders. Actually, this kind of video coding in combination with the JPEG standard, also referred to as ”Motion-JPEG”, is in common use nowadays, especially if random access is more important than compression efficiency. Therefore, what fundamentally distinguishes video compression from still image compression is the exploitation of temporal correlations using MCP. Evolving video coding standards have mainly enhanced the temporal prediction by providing more and more accessible coding options to be selected by the encoder for enhanced compression efficiency. These coding options also reflect different common characteristics of video sequence.

The observation that many areas are unchanged, e.g. background in video telephony application with fixed camera position, intra coders have been extended with Conditional Replenishment (CR) such that an area, usually a MB, can be either coded in *Intra* or *Skip* mode. The latter just copies the area in the previous frame to the current frame. This can be extended by adding the possibility of a refinement step to address changes such as brightness, camera noise, etc. The coded version of the difference between the skip mode and the original data is also transmitted. However, this additional mode can still not compensate the most prominent phenomenon in succeeding video frames, namely that either the entire frame is shifted in case of camera movements, or, objects in the image are moving locally relative to the background. Spatial displacement *motion vectors* allow tracking this motion where the encoder searches in the motion estimation process for some suitable motion vectors. In its basic form, one motion vector is assigned to each MB and motion vectors have accuracy of full pixel positions. Over the years, MCP has been extended to

- subsample accurate MCP [JJ81, Gir87] providing motion vectors with fractional precision (typically half–sample and quarter–sample are used) by applying an interpolation of the reference area; this allows a more precise motion representation and also provides signal theoretical advantages [Gir93],
- variable block size MCP [GR75] where the encoder can utilize smaller block sizes and more than one tuple of motion vectors per MB and the motion can be better predicted,
- bi–directional MCP [SB91] allowing averaging of two prediction signals, one from the past and one from the future; the helps in case of continuous motion,
- multi–frame MCP [WZG99] allowing more than the previously decoded frame to exploit long–term statistics,
- multi–hypothesis MCP [Gir00] allowing linearly superimposed prediction signals; this generalizes bi-directional MCP, overlapped block motion compensation [OS94], and sub–sample accurate MCP.

3.2.2 Video Coding Standards

History

Since the finalization of the first digital video standard H.120 in 1984 [H12], mainly two organizations, namely the ITU-T’s Video Coding Experts Group (VCEG) and International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) MPEG, have been working on the development on new standards. The main focus is the former group is on video coding standards for communication applications, namely H.261 [H26a] and H.263 [H26b], whereas

MPEG originally concentrated on different higher quality applications for storage (MPEG-1 [MPG93]) and digital video broadcast of television signals within MPEG-2 [MPG94]. It is worth to mention that the very successful and widely used standard MPEG-2 has been developed in a joint effort of VCEG and MPEG. After this joint effort the VCEG group worked on improvements of the H.263 [H26b], mainly in terms of compression efficiency and error resilience. ISO/IEC launched a new project within the MPEG–4 framework to specify a very general multimedia coding standard. Thereby, more emphasis was put on functionalities such as the coding of video objects with arbitrary shape, sprite coding, and scalability rather than on coding efficiency.

In 1997, the ITU-T's VCEG started working on a new video coding standard with the internal denomination H.26L[4]. Thereby, a completely new design has been targeted rather than enhancing existing standards. The lessons learned from the standardization of H.263, namely that extensive optional modes limit the interoperability, and from MPEG-4, that the most important functionality of a video coding standard is compression efficiency, were the main driving forces in the standardization process. In addition, the integration of video coding standards into networks has been considered from very beginning. Therefore, most proposals were checked against the consistency with the three primary goals, (i) improved coding efficiency, (ii) improved network adaptation, and (iii) simple syntax specification. In August 1999 a remarkable simple and familiar draft model[5] has been adopted and was evolved into a test model long-term (TML) reference design. The draft model basically only contained features known from previous standards and was enhanced by adding additional features, especially for advanced MCP. In addition, network specific features have been addressed by a conceptual separation of compression related tools in the VCL and network integration features in the NAL. In late 2001, MPEG and VCEG decided to cooperate within the Joint Video Team (JVT) in the spirit of the successful MPEG-2 project, and to create a single technical design for a forthcoming ITU-T Recommendation and for a new part of the MPEG-4 standard based on the committee draft version of H.26L at this time. Finally, Recommendation H.264/AVC [H2603] was approved by the ITU-T in May 2003. A similar approval procedure has taken place within ISO/IEC. The normative part of a video coding standard only consists of the appropriate definition of the order and the semantics of syntax elements and the decoding of error-free bit-streams.

Video Codecs in 3GPP Services

Video is an important component in many 3GPP services. Its relevance to the continuous success of mobile broadband networks is significant, as it allows to exploit the provided bit-rates and it may be the surplus to existing services. In contrast to speech and audio codecs, like the Adaptive Multi-Rate (AMR) codec, 3GPP does not specify video codecs for its services, but references existing general–purpose video codec specifications. Referencing here means that for a specific service, 3GPP either mandates or at least recommends the support of a video decoder conforming to a certain profile and level, possibly with some further restrictions or clarifications.

Figure 3.7 provides an overview of selected video services, referenced video specifications, and protocols. On top of 3GPP radio access bearers, end-to-end multimedia services are supported. For real-time packet-switched services, 3GPP currently defines Packet–Switched Conversational [PSC02] (PSC) services, PSS services, MBMS, and Multimedia Telephony over IP Multimedia Subsystem (IMS) [MTS07] (MTSI). Within these service definitions, different applications are supported, of which a majority includes digital video, for example, in video telephony,

[4] The 'L' addressed the idea of a long–term project

[5] The initial model document only contained less than 30 pages of description for the entire codec.

video streaming, or mobile TV applications. However, different applications generally have different service constraints, which leads to the selection of different codec settings, delivery protocols, and radio bearer settings to support the required end-to-end QoS.

Specifically, within Release 7 of 3GPP specifications, only the ITU-Ts H.263 profile 0 and profile 3, ISO MPEG-4 simple profile, and the baseline profile of the H.264/MPEG-4 AVC codec are referenced. Each service may and generally does recommend multiple video codecs. The usage of one or the other codec in a specific session is done through capability exchange and session establishment procedures. As 3GPP relies on existing multimedia protocols, not only the video codecs itself are of essence, but also the support of the encapsulation and packetization into service specific media formats and protocols need to be defined. In this case, 3GPP also almost exclusively relies on outside work, mainly on specification in IETFs AVT, MMUSIC, and RMT group, ITU-Ts SG16, and ISO MPEG for file formats (FF). Similarly, the specific profiles and options of general-purpose specifications are referenced. For audio codecs, 3GPP provides comprehensive

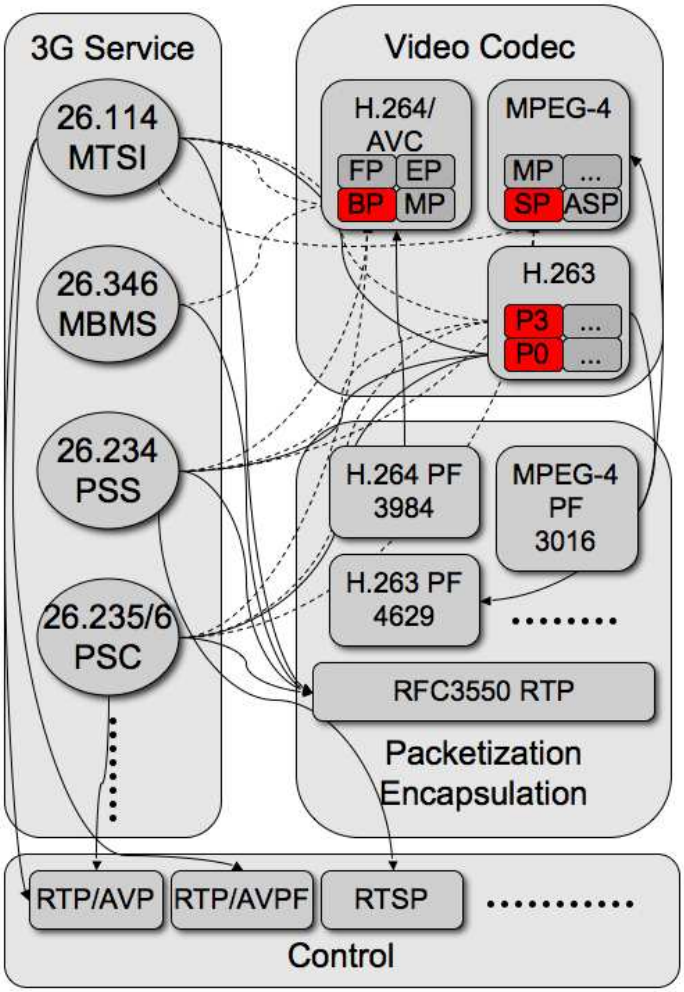


Figure 3.7: Video Codecs in 3G Service Specifications.

specifications of encoder and decoder defined in floating and fixed point implementations, as well as the error handling. Reference software is provided, and the audio codecs are fully characterized by a well-established process, which includes subjective performance metrics, typical and worst-case error conditions, and typical service bit-rates. Especially when compared to speech and audio

codecs, the specifications of video codecs in 3GPP is quite compact. Encoder settings, e.g. bit-rate control, error resilience, or motion estimation processes, are generally completely left open and will be subject of this work.

3.2.3 H.264/AVC Compression Tools

Although the design of the VCL of H.264/AVC basically follows the design of prior video coding standards, it contains many new details that enable a significant improvement in terms of compression efficiency. We will briefly highlight those, for more details we refer to [H2603],[SW05], and [LSW03]. The encoding operation for a picture follows the principle as shown in Figure 3.5 and is summarized in the following. After spatial and temporal pre–processing as well as appropriate color space conversion each picture of a video sequence is separately processed by the video encoder. As shown in Figure 3.6 the picture is even further divided into MBs which serve as the basic building entity of the standard for which the decoding process is defined. All luminance and chrominance of a MB are either spatially or temporally predicted. The availability of certain prediction types depends on the picture type.

Spatial Prediction

MBs can be coded with non–temporal prediction of the picture representation in a so–called intra mode. The use of prediction within the picture has appeared before in still image and video coding, e.g in JPEG, H.263 Annex I, or MPEG–4. In H.264/AVC greater efficiency is achieved by the use of directional prediction in the spatial domain rather than coefficient value prediction in the transform domain. Three basic modes are distinguished: one allows to code parts with many details efficiently, another one enables efficient coding for smooth areas, and a simple Pulse Code Modulation (PCM) mode. Within the first two modes different spatial prediction modes are allowed, namely nine for the first one and four for the second one.

Temporal Prediction

The temporal prediction in form of MCP process uses many advanced tools listed in subsection 3.2.1, namely

- variable block–size MCP with as big as 16×16 and as small as 4×4, as well as rectangular block sizes, resulting in 67 possible partitions for the luminance signal, and up to 16 motion vectors maybe transmitted for a each MB,
- quarter–pel accurate MCP with a six–tap interpolation filter for half–pel positions and bilinear interpolation for quarter–pel positions [WM03],
- multi–frame MCP with reference selection possible for block sizes as small as 8×8 for the luminance signal.
- multi–hypothesis MCP generalizing the B–picture concept of previous standards for block sizes as small as 8×8 for the luminance signal [FG03].

To support the latter a concept for generalized frame-buffering concept has been adopted allowing MCP not just from previous frames but also from future frames. For that, a flexible and efficient signaling method is applied. Still, with appropriate multiple reference frame handling the well-known functionality of disposable B-pictures known from, e.g. MPEG-2, is supported. In addition,

motion vectors are differentially encoded using spatial or directional prediction. Also, the skip mode uses motion vector prediction providing more efficiency especially for constant motion areas.

Transform and Quantization

Similarly to prior coding standards H.264/AVC utilizes transform coding of the prediction error signal. However, the transformation is applied to 4×4 blocks rather than 8×8 and instead of the DCT, a separable integer transformation similar to a 4×4 DCT is used. Since the inverse transform is defined by exact integer operations, inverse–transform mismatches are avoided. Appropriate transformations are used to the four DC-coefficients of each chrominance component (2×2 transform) and for the smooth intra mode by repeating the 4×4 transformation for better decorrelation of these in general smooth areas.

For the quantization of transform coefficients, H.264/AVC uses scalar quantization. The quantization step size Δ, and thus the fidelity of the reconstruction, is determined by the Quantization Parameter (QP) q as

$$\Delta = 2^{\frac{q-12}{6}}, \tag{3.14}$$

where the QP can take on one out of 52 values, i.e., $q = 1, \ldots, 52$. The definition according to (3.14) results in an increase of approximately 12.5% in the quantization step size when increasing q by one and an increase of six doubles the quantization step size Δ. The quantized transform coefficients are generally scanned in a zig-zag fashion and converted into coding symbols by appropriate entropy coding methods. For details and background on transform and quantization the interested reader is referred to [MHKL03].

Entropy Coding

H.264/AVC supports two method of entropy coding. The first one called Universal Variable Length Coding (UVLC) uses one single infinite–extend codeword set. The entropy coding takes care of the binary representation of different syntax elements generated by different processing steps for each MB such as coding mode decisions, spatial prediction information, motion vector data, and quantized coefficients for the luminance and chrominance signal. Each syntax element has in general different statistics. In H.264/AVC, instead of designing a different VLC table for each syntax element, only the mapping to the single UVLC table is customized according to the data statistics. For coding the quantized transform coefficients the UVLC in combination with run–length coding for zig-zag scanned has been replaced by a more efficient method called Context–Adaptive Variable Length Coding (CAVLC)[6]. Thereby, a VLC table out of a family of exp-Golomb like VLC tables is selected depending on previously transmitted symbols [H2603, WSBL03]. The efficiency of entropy coding is further improved if Context–Adaptive Binary Arithmetic Coding (CABAC) is used allowing the assignment of non–integer numbers of bits to each symbol of an alphabet. Additionally, the usage of adaptive codes permits the adjustment to non-stationary symbol statistics and context modeling allows for exploiting statistical dependencies between symbols. For details on the algorithms and performance of CABAC the interested reader is referred to [MSW03].

[6]The replacement of run–length coding was mainly to circumvent intellectual property rights rather than based on technical decisions.

De–blocking Filter

For removing block–edge artifacts, H.264/AVC includes an inloop de–blocking filter, i.e., the block edge filter is applied inside the motion prediction loop. The filtering strength is adaptively controlled by the values of several syntax elements. For details we refer to [LJL+03].

3.2.4 H.264/AVC Functionalities

In the following we will briefly present some features in the H.264/AVC standard which have been introduced not for compression efficiency reasons, but to allow functionalities necessary in different applications. We will skip specific error resilience features at this point and defer this to section 3.3.2.

Slices

A slice is a sequence of MBs and provides spatially distinct resynchronization points within the video data for a single frame. The MBs within a slice are processed in raster–scan order when Flexible Macroblock Ordering (FMO) is not in use (for details on FMO see subsection 3.3.2). A picture is therefore a collection of one or more slices. The conceptual idea of slices is that they are self–contained meaning that the syntax elements can be decoded without knowledge of data from other slices. However, if this idea is strictly applied, than compression efficiency would suffer significantly and basically only intra information can be transmitted. For this reason, temporal prediction over slice boundaries is permitted. Even the deblocking filter might be applied over slice boundaries. Not allowed is the spatial prediction over slice boundaries, motion vector predictions, as well as any syntactical dependencies in the entropy coding.

As long as SP pictures are not in use a slice can be coded with one of three different modes. The mode is signalled in the slice header and restricts the availability of prediction in the MCP process.

- **I slice** only intra prediction, but no temporal prediction is used.
- **P slice** in addition to the modes in the I slice, MCP can be used with at most one MCP signal per prediction block.
- **B slice** in addition to the modes in I and P slices, MCP with two MCP signals per prediction block can be used.

Hypothetical Reference Decoder

For a standard it is important that all decoders compliant with the standard are capable to decode a compressed stream. Compliancy is mainly achieved by applying the appropriate syntax, but also memory and other processing constraints need to be taken into account. The Hypothetical Reference Decoder (HRD) places constraints on compressed streams (and hence encoders) in order to enable cost-efficient decoder implementations [IT97, ISO00]. System specifications of media transmission standards usually discuss and specify the connections between a leaky bucket $(R, |\mathcal{B}_{\mathrm{e}}|_{\min}, |\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}})$ and the sampling curve B_t. For the Video Buffer Verifier (VBV) design in MPEG–2, as well as for the HRD design for H.263, the buffer size $|\mathcal{B}_{\mathrm{e}}|_{\min}$ and the initial buffer fullness $|\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}}$ are fixed. $|\mathcal{B}_{\mathrm{e}}|_{\min}$ is either specified in the bit-stream (MPEG–2) or by some constraint in the specification of the profile and level. $|\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}}$ can be fixed by transmitting the value in the bit–stream (MPEG–2 CBR mode), by filling up the buffer completely (MPEG–2 VBR mode),

i.e., $|\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}} = |\mathcal{B}_{\mathrm{e}}|_{\mathrm{min}}$, or by just using the size of the first frame $|\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}} = B_1$, i.e., the first frame is immediately decoded (H.263). The HRD of H.264/AVC is unique as it provides constraints for the Coded Picture Buffer (CPB)[7] and the Decoded Picture Buffer (DPB) which necessary due to the multiple reference frame concept and the generalized buffer management in H.264/AVC. In addition, by introducing a multiple-leaky-bucket model flexibility is added in trading buffer size, decoding delay, and transmission bit-rate when streaming pre-encoded streams. For more details we refer the interested reader to [RCCR03].

Bit-rate Adaptation

Bit-rate adaptivity is one of the most important features for applications in wireless systems to react to the dynamics due to statistical traffic, variable receiving conditions, as well as handovers and random user activity. Due to the applied error control features these variations mainly result in varying bit-rates in different time scales. For applications where online encoding is performed and the encoder has sufficient feedback on the expected bit-rate on the channel by some channel state or decoder buffer fullness information, rate control for VBR channels can be applied. H.264/AVC obviously supports these features, mainly by the possibility of changing QPs dynamically, but also by the changing temporal resolution. Section 3.2.5 introduces these principles. For streaming offline encoded video, playout buffering at the receiver under HRD constraints can compensate bit-rate fluctuations to some extent.

However, these techniques might not be sufficient to compensate larger-scale bit-rate variations in wireless applications. In this case the bit-rate adaptation has to be performed by modifying the encoded bitstream. One option is to not transmit less important data units, such that the quality degrades gracefully. H.264/AVC provides different approaches to support packets with different importance for bit-rate adaptivity. First, the temporal scalability features [THG05] of H.264/AVC relying on the reference frame concept can be used. Second, if frame dropping is not sufficient, one might apply data partitioning (see section 3.3.2) which can be viewed as a very coarse but efficient method for SNR scalability. Third, flexible macroblock ordering (also refer to section 3.3.2) may also be used for prioritization of regions of interest. For example, a background slice group can be dropped in favour of a more important foreground slice group [HWG04].

For many use cases it is necessary to adapt the bit-rate dynamically within large bit-rate and time scales, e.g., during a streaming session. In such environments bitstream switching provides a simple but powerful mean to support bit-rate adaptivity. In this case the streaming server stores the same content encoded with different versions in terms of rate and quality. In addition, each version provides means to randomly switch into it. IDR pictures provide this feature, but they are generally costly in terms of com-pression efficiency when they are not associated with scene cuts or transitions. The SP-frame concept in H.264/AVC can be used to reduce the loss of compression efficiency in stream switching. More details on stream switching are introduced in subsection 3.3.4.

Profile and Levels

Video coding standards can be applied for many different purposes in many different service and application environments. The environments might be differentiated in terms of encoder and decoder complexity, transport and bit-rate conditions, professional versus private applications, etc. However, it is obvious that for example high quality applications should not be harmed by the constraints of for example handheld video encoders. Therefore, certain compliance points are essential

[7] The CPB is equivalent to the decoder buffer in Figure 2.3

such that interoperability between encoders and decoders is guaranteed in certain application scenarios. Profile and levels guarantee such conformance points and can be negotiated in the setup. H.264/AVC version 1 defines three profiles, baseline, main, and extended profile. Profiles have assigned tools which the decoder must support if it claims to be compliant to a certain profile. The assignment of tools to profiles is shown in Figure 3.8. As seen from the figure, for wireless applications baseline and extended profiles are most interesting as they contain all error resilience and rate adaptivity tools. Furthermore, levels allow specifying some maximum decoder complexity conformance points in terms of spatial and temporal resolution, reference frame buffer size, etc. For mobile applications, obviously content is usually encoded for decoders with lower complexity, e.g. QCIF resolution at 15 fps.

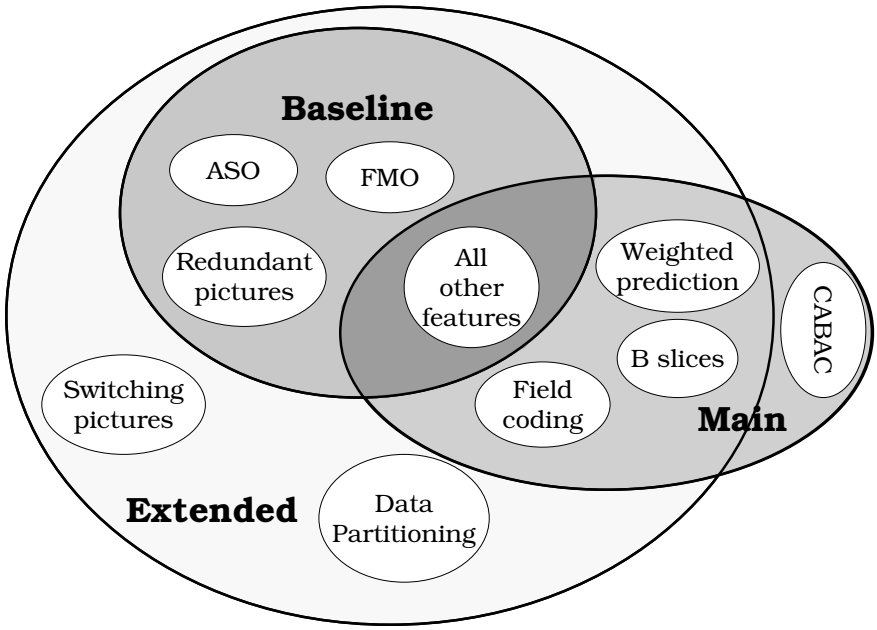


Figure 3.8: Profiles in H.264/AVC.

3.2.5 Operational Encoder and Rate Control

Motivation

The normative part of a video coding standard only consists of the appropriate definition of the order and the semantics of syntax elements and the decoding of compressed streams. The appropriate selection of encoding parameters, referred to as *operational encoder control*, is completely left over to the encoder implementation. Thereby, the encoder must take into account constraints imposed by the application in terms of bit-rates, encoding and transmission delays, as well as buffer sizes [OR98]. To define a rate–distortion performance of video in classical Shannon sense is an impossible task: The reason for this is the lack of sufficient statistical models. Therefore, usually the approach of operational rate–distortion performance is preferred, i.e., the best achievable performance for a given source given a certain coding framework and certain constraints, e.g., in terms of complexity or delay. The coding framework might be described by a set of accessible coding tools such as MCP features, it might described a certain profile and level description of a standard, etc. In complex system design problems both parts, the encoder and the decoder should

be optimized, e.g. by using some iterative methods like the Max–Lloyd algorithm or generalized versions of them [GN99, Eff98]. In contrast, in case of the usage of a standard with a completely specified decoder, parameters in the encoder should be selected such that a good rate–distortion performance is achieved. As the encoder is limited by the syntax of the standard, this problem is referred to as *syntax–constrained rate–distortion optimization* [OR98]. In case of a video coder such as H.264/AVC, the encoder must appropriately[8] select parameters such as motion vectors, MB modes, quantization parameters, reference frames, or spatial and temporal resolution as shown in Figure 3.9.

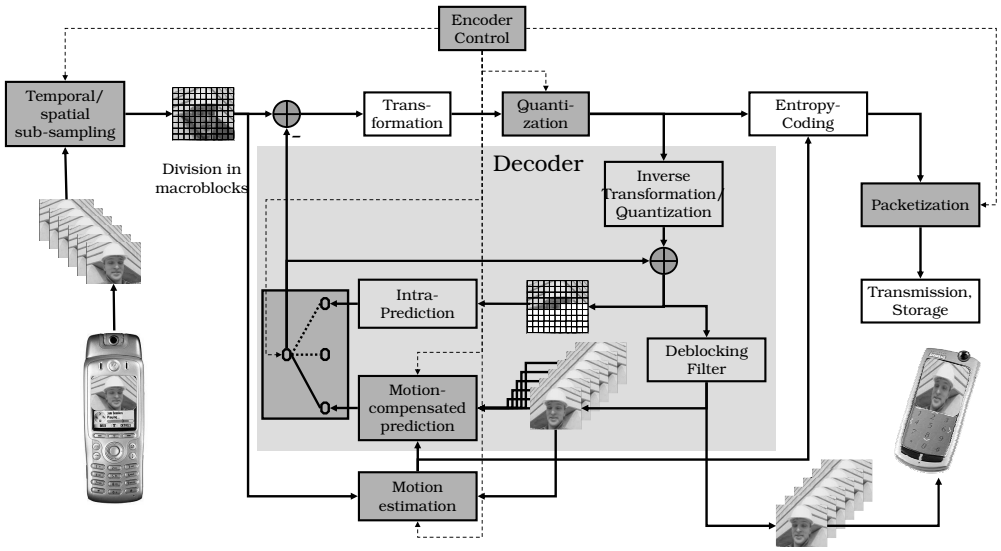


Figure 3.9: H.264/AVC video encoder with selectable encoding parameters highlighted.

Additional constraints might be imposed by the system or application: Typically, in real–time communication the data units have to arrive in time at the decoder. Assuming a CBR channel with bit-rate R, a data unit of size $l_{\mathcal{P}}$ requires a time of at least $l_{\mathcal{P}}/r$ to get to the decoder. Therefore, the packet length of source unit might be constrained by the delay. A more formal description based on the definitions in subsection 2.1.4 will follow.

In the following we concentrate exclusively on coding efficiency, error resilience issues will be discussed in section 3.3.2. Two simplify matters, we distinguish between two encoder operations: Whereas the *encoder control* performs local decisions, i.e., the selection of MB modes, reference frames, or motion vectors on MB level and below, the *rate control* adapts the global parameters such as quantization parameters, spatial resolution, or the frame rate. Due to the huge amount of encoding options for each MB in H.264/AVC especially the encoder control is important for good performance.

Notations

Let us consider a sequence of N_s source units $\{s_n\}$ where each source unit itself can be coded with one out of M source coder $\mathcal{Q}^{(m)}$. This results in M different operation points for each source unit $\{s_n\}$. For convenience, let us define for each source unit s_n the pair of rate $r_{n,m} \triangleq R^{(m)}(\mathcal{Q}, s_n)$ and distortion $d_{n,m} \triangleq D^{(m)}(\mathcal{Q}, s_n)$. No further assumptions are made on the rate–distortion pairs. We

[8] "appropriately" refers to a good quality given rate constraints.

assume that this data is known by appropriate measurements at the encoder, but also appropriate modelling might be applied to obtain the parameters.

It is now the task of the encoder to assign an appropriate coding option $m(n)$ to each source unit $\{s_n\}$ such that some overall distortion is minimized under the given constraints. For example, given a total rate constraint R_t and if we are interested in finding the minimum average distortion, the optimal coding option vector $\mathbf{m}^* \triangleq \{m^*{}_1, \ldots, m^*{}_n\}$ should be chosen as

$$\text{select} \quad \mathbf{m}^* \quad \text{such that} \quad \sum_{n=1}^{N_s} r_{n,m^*{}_n} \leq R_t \quad \text{and} \quad \sum_{n=1}^{N_s} d_{n,m^*{}_n} \rightarrow \min . \tag{3.15}$$

Encoder Control using Lagrangian Techniques

Lagrangian coder control techniques have become the most widely accepted approach on variable bit-rate source coders. The popularity of this approach is due to its efficiency and simplicity. For completeness, we briefly review the Lagrangian optimization techniques and their application to video coding. Consider a rate–distortion pair $\{r_{n,m}, d_{n,m}\}$ can be measured independently for each source s_n. This would for example be the case if we want to select a single quantization parameter for each frame of a video sequence and the video frames are coded in intra mode only. Then, the most common solution to the problem in (3.15) is a discrete version of Lagrangian optimization first introduced by Everett [Eve63]. In source coding, these techniques were first applied in [SG88, CLG89b, CLG89a] and have been accepted for many source coding and system design.

Intuitively, in Lagrangian optimization the two cost terms rate and distortion are linearly combined and the mode is selected such that the total cost is minimized with $\lambda \geq 0$ being the Lagrange parameter for appropriate weighting of rate and distortion. From [Eve63, SG88] the following is known: If a set of coding options $\mathbf{m}^*$ minimizes

$$\sum_{n=1}^{N_s} \left[d_{n,m^*{}_n} + \lambda r_{n,m^*{}_n} \right], \tag{3.16}$$

then it is also the optimal solution to the constrained problem stated in (3.15) if $R_t = R(\lambda) \triangleq \sum_{n=1}^{N_s} r_{n,m^*{}_n}$ such that $D(\lambda) \triangleq \sum_{n=1}^{N_s} d_{n,m^*{}_n}$ is minimal over all possible $\mathbf{m}$. Since the rate constraint is removed and for given λ, a re–formulation of the constrained optimization problem in (3.15)is possible with

$$\mathbf{m}^* = \arg\min_{\mathbf{m}} \left(\sum_{n=1}^{N_s} \left[d_{n,m_n} + \lambda r_{n,m_n} \right] \right) = \left\{ \arg\min_{m_n} \left[d_{n,m_n} + \lambda r_{n,m_n} \right] \right\}_{n=1}^{N_s} . \tag{3.17}$$

The appealing advantage of the problem formulation on the right hand side in (3.17) is obvious. Rather than having $(N_s \times M)!$ possible combinations, the complexity in the search is significantly reduced to $N_s \times M$ combinations for a given λ. Note that the Lagrangian parameter λ corresponds to the negative slope of the operational rate–distortion curve.

The assumption that coding options can be selected independently is obviously not true for most compression schemes. Whenever predictive coding is involved, the selection of the coding option for the predecessor obviously influences the rate–distortion performance for the predicting source. Usually, tree–like dependencies in encoding process are present and simple search methods are virtually impossible. To circumvent this problem, usually *greedy approaches* are applied for

predictive source coding methods. *Greedy* refers to the case that only the rate and the distortion of the source data to be coded is take into account, still using a Lagrangian approach. However, as the source data is coded in the order of dependencies, greedy method usually ensure that the most important components are coded first. So Lagrangian approaches are also widely used even for predictive source coding.

In Lagrangian encoder control, the appropriate λ for a required total constraint R_t needs to be chosen. Low–complexity approaches have for example been addressed in [SG88], but usually the selection of the Lagrangian parameter is combined with the global rate control and will be discussed later. For a detailed discussion and more insight into this topic the interested reader is referred to [OR98] and references therein.

Mode Selection in Hybrid Video Coding

The application of Lagrangian techniques to control a hybrid video coder is attractive, but not straightforward. The temporal and spatial dependencies of the rate–distortion costs as well as the hierarchy in the codec needs an adaptation of the classical Lagrangian principle in (3.17). We will provide a short introduction aligned to the algorithms in the H.264/AVC test model, more details are for example presented in [SW98, WLM+96, WSJ+03].

Consider a block–based hybrid video codec such as H.264/AVC. Each frame t of the image sequence is partitioned into N_{MB} MBs, and for each MB b the associated pixels be given as $\mathbf{s}_{t,b}$. In case of no ambiguity, we drop the frame number t and write $\mathbf{s}_b$. Assuming a P-frame, for each MB, the encoder can choose one of many coding options from the set $\mathcal{O}$. We can distinguish between intra coding options categorized in $\mathcal{O}_I$ and inter coding options in set $\mathcal{O}_P$. Each mode itself might have additional parameters assigned which have to be selected. Especially complex is the good selection of motion vectors for inter coding modes. Therefore, in the H.264/AVC test model a widely accepted technique is applied which splits the coding of inter MBs in a motion estimation procedure and a final mode decision. A greedy approach is applied for the coding and selection of MB modes in a way that MBs are coded sequentially and the influence of the coding decision on depending MBs, temporary and spatially, is not taken into account in the mode decision.

First, for each possible inter mode in $\mathcal{O}_P$, the optimal motion vectors and reference frames are found from a set $\mathcal{O}_R$ using a Lagrangian formulation where the distortion is measured based on the DFD, and the rate is determined as the number of bits to transmit all motion vectors, and the reference frame index. The option set for the motion vectors and the reference frames, $\mathcal{O}_R$, is basically only constrained by the available data in the reference frame buffer. Sub–pel accurate motion estimation is only optimized locally, i.e., in the neighborhood of good ful–pel positions. Finally, for each inter mode, the resulting prediction signal is transformed and quantized. Therefore, for each inter mode $m \in \mathcal{O}_P$ in MB b, we obtain a distortion d_{b,m_b} and a associated rate r_{b,m_b} as well as some parameters assigned by the motion estimation process. The SKIP mode is assigned to the inter mode set $\mathcal{O}_P$. Similarly, for each intra mode $m \in \mathcal{O}_I$ in MB b, we also obtain a distortion d_{b,m_b} and a associated rate r_{b,m_b}, also with some parameters assigned by the intra prediction. The distortion d_{b,m_b} usually (at least in H.264/AVC test model) reflects the Sum of Squared Differences (SSD) between the original MB $\mathbf{s}_b$ and reconstructed version of the MB $\tilde{\mathbf{s}}_{b,m}$ if coded with option m, i.e.,

$$d_{b,m} = \sum_i |s_{b,i} - \tilde{s}_{b,m,i}|^2 , \tag{3.18}$$

and the rate $d_{b,m}$ is defined by the number of bits necessary to code MB b with option m. Finally,

the coding mode for MB b is selected as

$$\forall_b \quad {m^*}_b = \arg\min_{m\in\mathcal{O}} \left(d_{b,m} + \lambda_{\mathcal{O}} r_{b,m}\right). \tag{3.19}$$

This still leaves the question on how to choose the Lagrangian parameter $\lambda_{\mathcal{O}}$. In [SW98] and [WG01] it is suggested that if the SSD is applied as distortion measure then $\lambda_{\mathcal{O}}$ should be directly proportional to the square of the step size Δ of a uniform quantizer applied. With the rate measured in bits, experimental results in [WG01] then suggest the following relationship for H.264/AVC encoder control:

$$\lambda_{\mathcal{O}}(q) = 0.85 \cdot 2^{\frac{q-12}{3}}. \tag{3.20}$$

If the Sum of Absolute Differences (SAD) instead of the SSD is used – as for example the case for the motion estimation in H.264/AVC test model – then the Lagrange parameter should be changed to $\lambda_{\mathcal{O}}(q) \rightarrow \sqrt{\lambda_{\mathcal{O}}(q)}$. Therefore, with the relation in (3.20), the rate–distortion performance of H.264/AVC test model is equivalently controlled by the selecion of the Lagrangian parameter $\lambda_{\mathcal{O}}$ or by the QP q.

Rate Control

In addition to the local decision, an encoder must take care that some overall rate–constraints are maintained within a certain time window. Basically, an encoder rate control must take care that the generated bit–stream is contained by a specified leaky bucket. Thereby, the basic idea is that a good QP for each MB is selected, and with that, the mode selection according to (3.19) for each MB is performed. The selection of the QP is task of the rate control. Many research publications exclusively deal with good rate control algorithms for different applications basically described by different leaky buckets, e.g. [RCS99], [PS01], and [VK01].

In the H.264/AVC test model rate control has been introduced at a late stage [LGP+03] and is not considered in this thesis. Therefore, we will in the following restrict ourself to two very simple rate control schemes.

To obtain a low–delay coding scheme, a CBR rate control is applied. We assume in the following that the rate–distortion pair for a single video frame $\mathbf{s}_t$ is controlled by a single parameter, the QP[9] q and is denoted as $d_{t,q}$ and $r_{t,q}$. Further, we assume a constant bit-rate R and a constant frame rate f_s. Then, the optimal QP ${q^*}_t$ for each frame $\mathbf{s}_t$ is selected in a greedy way starting with $t = 1$ such that the distortion is minimized and an average bit-rate constraint is fulfilled, i.e.,

$$\forall_{t=1,\ldots,N_s} \quad {q^*}_t = \arg\min_{q\in\mathcal{S}_q} \left\{ d_{t,q} \quad \text{with } q \text{ such that } \quad \sum_{i=1}^{t} r_{i,q} \leq t\frac{R}{f_s} \right\}. \tag{3.21}$$

If equality in the rate constraint in (3.21) can be fulfilled, then this rate control is equivalent to encoding a bit–stream as is contained by a leaky bucket $(R, |\mathcal{B}_{\mathrm{e}}|_{\min} = \frac{R}{f_s}, |\mathcal{B}_{\mathrm{d}}|_{\mathrm{init}} = \frac{R}{f_s})$. Although equality in this constraint cannot be fulfilled, the additional encoding delay can be neglected. If no buffering is applied at encoder and decoder and only backward–predictive coding is used, the end–to–end delay when transmitting over a CBR channel R is therefore only determined by the sum of the inverse frame rate and some additional channel delay, i.e., $\Delta T = 1/f_s + \delta_v^{(\mathcal{C})}$. In the following, we refer to this rate–control as CBR rate control.

For the H.264/AVC test results, the rate–distortion performance was determined by selecting a single QP for the entire sequence and computing the overall rate of the encoded sequence. This

[9] For the the H.264/AVC test model this fixes also $\lambda_{\mathcal{O}}$ with the relation in (3.20).

scheme is basically useless if the bit-rate must be controlled such that leaky bucket constraints are maintained. However, for scenarios where only the average rate is of importance as for example when transmitting over the Internet or for download–and–play applications, this rate–distortion performance can also be viewed as a practical rate control scheme. For example, by monitoring RTCP messages, the QP can be changed over larger time–scales. Therefore, we apply a Variable Bit–Rate (VBR) rate control, which selects the QP for each frame as

$$\forall_{t=1,\ldots,N_s} \quad {q^*}_t = \arg\min_{q\in\mathcal{S}_q}\left\{\sum_{i=1}^{N_s} d_{i,q} \quad \text{with } q \text{ such that} \quad \sum_{i=1}^{N_s} r_{i,q} \leq N_s\frac{R}{f_s}\right\}. \tag{3.22}$$

Global Parameter Selection

The framework of operational encoder control and rate control allows to select good coding modes according to (3.19), and reasonable QPs according to (3.21) or (3.22) based on objective measures such as the MSE or the PSNR for a specified global parameter set of the video to be encoded, namely the spatial resolution $N_{\text{width}} \times N_{\text{height}}$, the frame rate f_s, and the bit-rate R. The selection of these three parameters is crucial for the perceived subjective quality. However, as the quantification of the perceived quality based on objective measure is basically impossible, we have decided to set up a subjective test to get an indication on the selection of these parameters for wireless applications when coded with H.264/AVC baseline profile with rate control according to (3.22) and operational encoder control according to (3.19). The basic idea of the test as well as the results important for the remainder of this work are briefly presented in the following, for details on the test and a complete set of results we refer to [BHS04] and [Bru04].

Assume that the spatial resolution is fixed to QCIF due to the display and level restrictions of wireless devices. Four test sequences representing a mix of typical messaging clips, partly professional, partly amateur, have been encoded with only two parameters varied, namely the QP q and the frame rate f_s. The frame rate was selected from the set of frame rates $\mathcal{O}_{f_s} = \{3, 5, 7.5, 10, 15, 30\}$ fps, the quantizer was chosen from $\mathcal{S}_q = 1, \ldots, 51$. Then, for a given bit-rate R and a given frame rate f_s, the quantizer is selected according to (3.22). For a given bit-rate R, the optimum combination of QP and frame rate, denoted as (q^*, f_s^*), is determined based on the subjective quality $\mathcal{G}_{q,f_s}$ as

$$(q^*, f_s^*) = \arg\max_{q\in\mathcal{S}_q, f_s\in\mathcal{O}_{f_s}} \mathcal{G}_{q,f_s} \quad \text{with } (q, f_s) \text{ such that} \quad \sum_{t=1}^{N_s(f_s)} r_{t,q} \leq N_s(f_s)\frac{R}{f_s}, \tag{3.23}$$

where $N_s(f_s)$ denotes the length of the sequence when pre–processed with frame rate f_s. The subjective quality $\mathcal{G}_{q,f_s}$ is determined as the average value of the estimates of all test results, 30 in our case, on the perceived quality on an Absolute Category Rating (ACR) scale [P9196] with six grades defined as "1, *excellent*","2, *good*", "3, *satisfactory*", "4, *sufficient*", "5, *imperfect*" to "6, *poor*".

The most important conclusions from this experiments are as follows. For H.264/AVC encoded video a bit-rate of $R = 64$ kbit/s is sufficient to provide at least *good* quality for all test sequences. To cover the important range of quality levels, namely from better than *good* to *imperfect* only a very limited set of parameter values is necessary: QPs of 34 and frame rates of 10 fps are sufficient to provide better than *good* quality for almost all sequences. Sports sequences with high activity ask for slightly higher frame rate, but still 15 fps is sufficient. To provide *sufficient* quality quantization parameters above 40 and frame rates below 5 fps should definitely not be used, for

sports sequences at least 7.5 fps, better 10 fps should be used. We will use these guidelines for selection of frame rates in the remainder of this work.

3.3 H.264/AVC in Network Environment

3.3.1 Network Abstraction Layer

For efficient transmission in different environments not only coding efficiency is relevant, but also the seamless and easy integration of the coded video into all current and possible future protocol and network architectures. This includes the public Internet with best effort delivery as well as wireless networks. Therefore, the H.264/AVC design distinguishes between two different conceptual layers, the VCL and the NAL as shown in Figure 3.10. Both, the VCL and the NAL are part of the standard. The NAL concept has been introduced [WS02, SWH02] to address the expectation that H.264/AVC will be integrated in many applications and transport protocols and that many of these new transport protocols will be packet–based. The encapsulation as well as the transport in different transport protocols are not specified in the H.264/AVC standard but by the responsible standardization bodies for transport protocols such as MPEG-2 systems, RTP/IP, MP4 file format, or H.32X.

The NAL decoder interface is normatively defined in H.264/AVC coding standard, whereas the interface between the VCL and the NAL is conceptual and helps in describing and separating the tasks of the two layers.

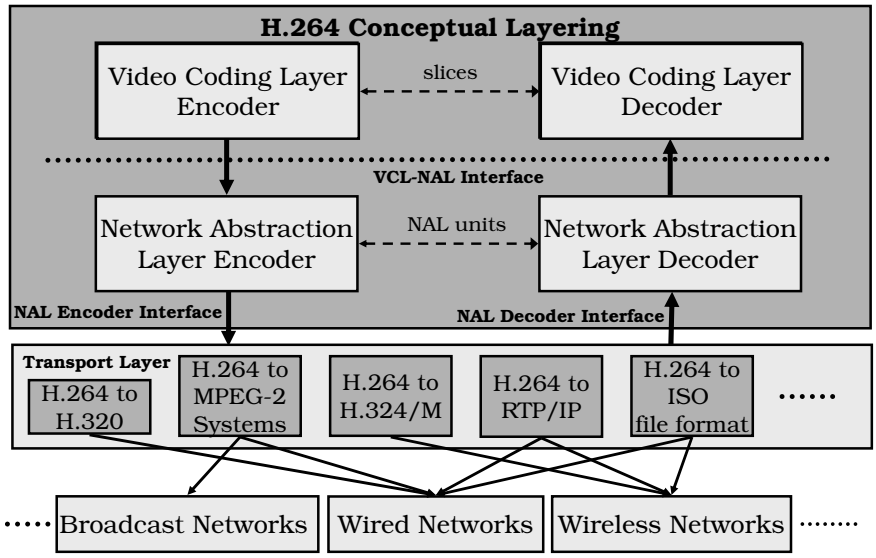


Figure 3.10: Layering concept of H.264/AVC and integration in network environments.

NAL units

The NAL encoder encapsulates the slice output of the VCL encoder into NAL units, which are suitable for the transmission over packet networks or in packet-oriented multiplex environments. In addition, Annex B of the H.264/AVC specification [H2603] defines an encapsulation and framing process to transmit such NAL units over byte-stream oriented networks. Each NAL unit consists of a one–byte header and the payload byte string. The header indicates one out of ten NAL unit

types with five bits, the potential presence of bit errors or syntax violations in the NAL unit payload using one bit, and information regarding the relative importance of the NAL unit for the decoding process using 2 bits. NAL units can be distinguished in

- **VCL-NAL unit types** containing data that represents the values and samples of video pictures. Four types are specified, namely a single slice packet and data partitions A, B, and C.
- **non VCL–NAL unit types** containing associated higher level information. Six types are specified for Supplemental Enhancement Information (SEI), Instantaneous Decoder Refresh (IDR), two parameter sets, picture delimiter, and filler data.
- **unspecified NAL unit types** which can be used as code points by external protocols. These code points are for example partly used by the RTP payload specification [WHS+05].
- **reserved NAL unit types** for future use in new versions in H.264/AVC.

Timing Information

One of the main properties of H.264/AVC is the complete decoupling of the transmission time, the decoding time, and the sampling or presentation time of slices and pictures. The decoding process specified in H.264/AVC is unaware of time, and the H.264/AVC syntax does not carry information such as the number of skipped frames as common in the form of temporal reference in earlier video compression standards. Also, there are NAL units that are affecting many pictures and are, hence, inherently time–less.

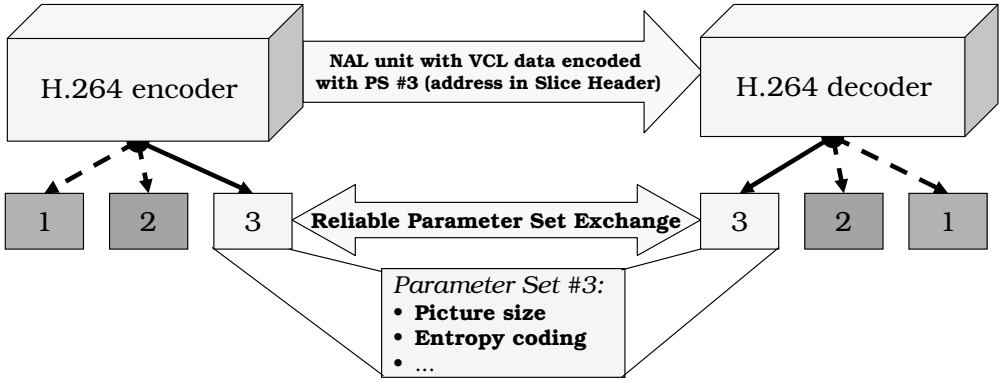


Figure 3.11: Innovative parameter set concept in H.264/AVC.

Parameter Set Concept

One fundamental new design concept of H.264/AVC is the generation of self-contained packets [SWH02] according to Figure 3.11. The way that this is achieved is to decouple information that is relevant to more than one slice from the media stream. This higher layer meta information should be sent reliably, asynchronously and in advance from the media stream that contains the VCL–NAL units. Provisions for sending this information in–band are also available for such applications that do not have an out–of–band transport channel appropriate for the purpose. The combination of the higher-level parameters is called a parameter set. H.264/AVC includes two types of parameter sets: Sequence Parameter Set (SPS) and Picture Parameter Set (PPS). An

active SPS remains unchanged throughout a coded video sequence, and an active PPS remains unchanged within a coded picture. The parameter set structures contain information such as picture size, optional coding modes employed, and MB to slice group map. In order to be able to change picture parameters such as the picture size, without having the need to transmit parameter set updates synchronously to the slice packet stream, the encoder and decoder can maintain a list of more than one sequence and picture parameter set. Each slice header contains a code point that indicates the sequence and picture parameter set to be used for the encapsulated slice. This mechanism allows to decouple the transmission of parameter sets from the packet stream, and to transmit them by external means, e.g. as a side information of the capability exchange, or through a reliable or unreliable control protocol. It may even be possible that they are never transmitted but are fixed by an application design specification.

3.3.2 H.264/AVC VCL in Error-Prone Environments

In addition to the NAL concept, the VCL itself includes several features providing network friendliness and error robustness which will be introduced in the following. For some video applications error resilience features are essential as the network cannot provide sufficient QoS to ensure the successful delivery of all NAL units. Figure 3.12 presents a simplified yet typical system when MCP video is transmitted over error–prone channels. For the time being assume that all MBs of one frame s_t are contained in a single packet $\mathcal{P}_t$, i.e., a NAL unit in case of H.264/AVC, and this packet is transmitted over a channel where correct packets are forwarded to the decoder, $\mathcal{C}_t = 1$, and corrupted packets are perfectly detected and discarded at the receiver, $\mathcal{C}_t = 0$.

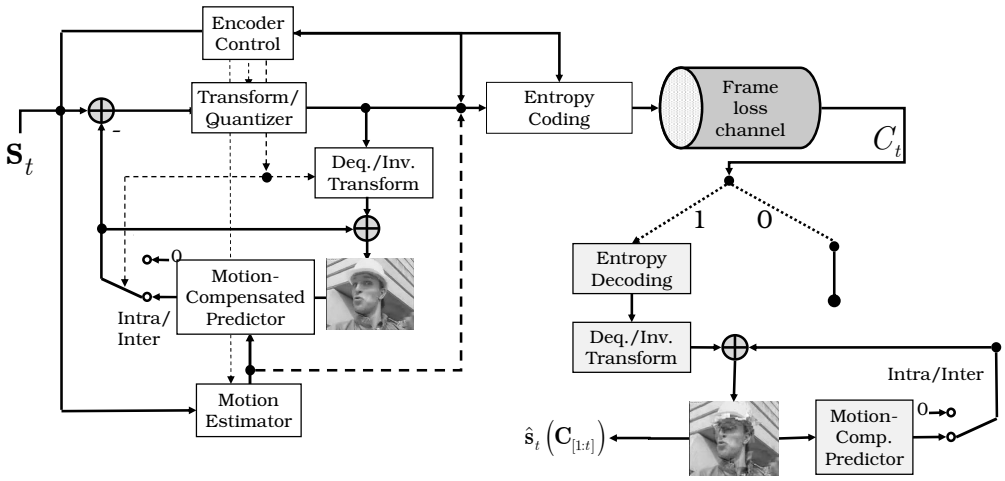


Figure 3.12: Simplified lossy video transmission system.

In case of successful transmission, the packet is forwarded to the regular decoder operation. The coded bitstream is converted into motion information and transform coefficients which allow reconstructing frame $\hat{s}_t$. The frame is forwarded to the display buffer, but also to the reference frame buffer to be used in the MCP process to reconstruct frame $\hat{s}_{t+1}$. In the less favorable case that the code representation of the frame is lost, i.e., $\mathcal{C}_t = 0$ no decoder operation is specified in the standard, but error concealment is necessary. In the simplest form, the decoder just skips the decoding operation and the display buffer, i.e., the decoded frame $\hat{s}_t$ is not updated. The viewer will immediately recognize the loss as the fluent motion is interrupted. However, this is not the only problem: As not only the display buffer is not updated, but also the reference frame buffer,

even in case of successful reception of packet $\mathcal{P}_{t+1}$, frame $\hat{s}_{t+1}$ will in general be not identical to the reconstructed frame $\tilde{s}_{t+1}$ at the encoder. The reason is obvious, as the encoder and the decoder refer to a different reference signal in the MCP process resulting in a so called *mismatch*. Therefore, also frame $\hat{s}_{t+1}$ at encoder and decoder will differ resulting again in a mismatch in reference signal when decoding $\hat{s}_{t+1}$. So it obvious that the loss of a single packets also effects the quality of $\hat{s}_{t+1}$, $\hat{s}_{t+2}$, $\hat{s}_{t+3}$, ... This phenomenon present in any predictive coding scheme is referred to as *error propagation*. If predictive coding is applied in the spatial and temporal domain, it is referred to as *spatio–temporal error propagation*. For MCP video, the reconstructed frame at the receiver, $\hat{s}_t$, not only depends on the actual channel behaviour $\mathcal{C}_t$, but on the previous channel behaviour $\mathcal{C}_{[1:t]} = \{\mathcal{C}_1, \ldots, \mathcal{C}_t\}$ and we write $\hat{s}_t(\mathcal{C}_{[1:t]})$. An example for error propagation is shown in Figure 3.13: the top row presents the sequence with perfect reconstruction, in the bottom row only packet $\mathcal{P}_t$ for frame number $t = 21$ is lost. Although the remaining packets are again correctly received the error propagates and is still visible in decoded frame $\hat{s}_{t=35}$.

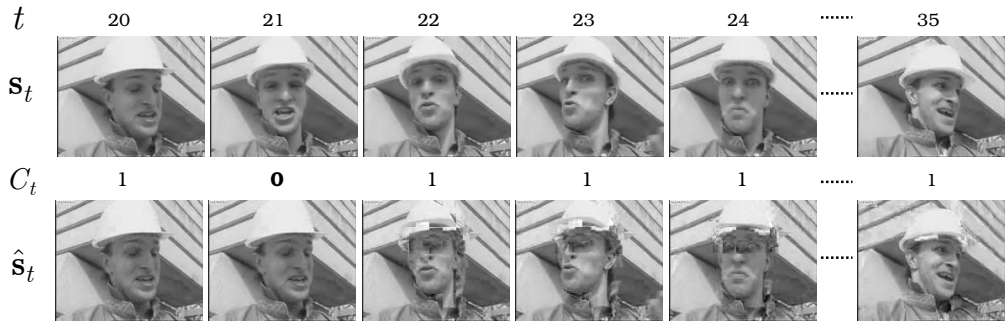


Figure 3.13: Example for error propagation in a typical hybrid video coding system.

Therefore, an error–resilient video coding transmission system should provide the following features:

1. Means which completely avoid transmission errors,
2. features which minimize the visual effect of errors within one frame, and
3. features to limit spatial as well as spatio–temporal error propagation in hybrid video coding.

3.3.3 H.264/AVC Error-Resilience Features

In the following, we briefly present different error-resilience features included in the standard with respect to their functionality. For more details we refer to [Wen03, SHW03] and references therein. Most of them are discussed in more detail in Section 3.3.2 with respect to their applicability and performance in different scenarios.

Compression Efficiency

Although common understanding usually assumes that increased compression efficiency decreases error resilience, this is typically not the case. Increased compression efficiency allows using additional bit-rate for error protection means and consequently the loss probability of highly compressed data can be reduced under the assumption of a constant overall bit-rate. All other error resilience tools listed in the following generally increase the data rate compared to the most efficient compression and therefore, their application should always be considered very carefully

to prevent adversely affecting compression efficiency, especially if lower layer error protection is applicable.

Slice Structured Coding

With the introduction of slices in the encoding spatially distinct resynchronization points within the video data for a single frame are provided. This is accomplished by introducing a slice header which contains syntactical and semantical resynchronization information. In addition, intra-frame prediction as well as motion vector prediction is not allowed over slice boundaries. The reduced prediction with the increased overhead associated with decreasing slice sizes adversely affect coding performance and requires additional overhead per slice. The location of the synchronization points is arbitrary at any MB boundary and can be selected by the encoder.

Within a slice group, the encoder typically can choose between two slice coding options, one with a constant number of MBs within one slice but arbitrary size, and one with the slice size bounded to some maximum number in bytes resulting in an arbitrary number of MBs per slice. Whereas with the former mode, the same slice types as present in H.263 and MPEG-2 can be formed, the latter is especially useful to introduce some quality-of-service as commonly the slice size determines the loss probability in wireless systems due to the processing shown in Figure 2.10. These two different packetizations and the resulting location of the slice boundaries in the bit–stream are shown in Figure 3.14.

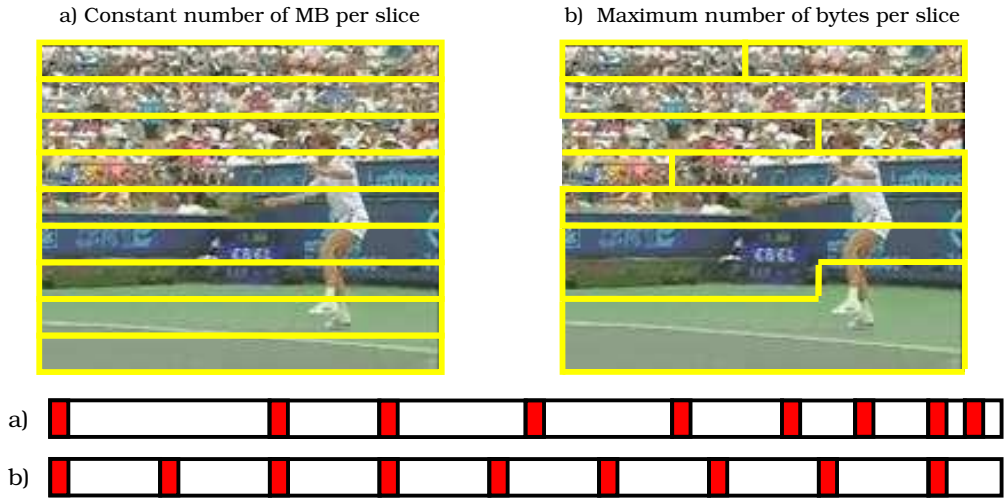


Figure 3.14: Different packetization modes: a) constant number of MB per slice, variable size of slices; b) maximum number of bytes per slice, variable number of MBs per slice.

Flexible Macroblock Ordering

Flexible Macroblock Ordering (FMO) allows the specification of MB allocation maps defining the mapping of MBs to slice groups, where a slice group itself may contain several slices. Therefore, MBs might be transmitted out of raster scan order in a flexible and efficient way. The efficient signalling of the MB map is possible using PPSs. Specific MB allocation maps enable the efficient application of features such as slice interleaving, dispersed MB allocation using checkerboard–like patterns, one or several foreground slice groups and one left-over background slice groups, or sub-pictures within a picture [Wen03] as shown in Figure 3.15. Slice interleaving and dispersed MB

allocation are especially powerful in combination with appropriate error concealment, i.e., when the samples of a missing slice are surrounded by many samples of correctly decoded slices.

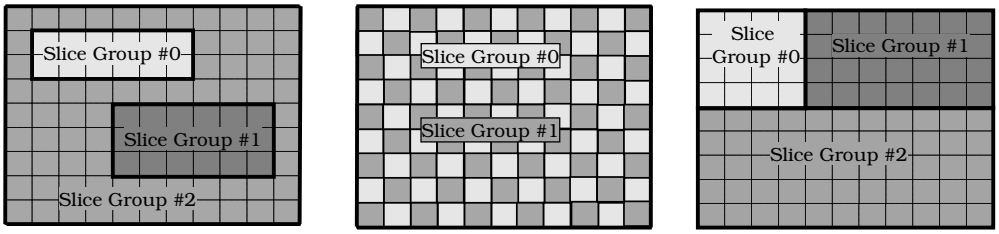


Figure 3.15: Specific MB allocation maps: foreground slice groups with one left-over background slice groups, checkerboard–like pattern with two slice groups, and sub–pictures within a picture.

Arbitrary Slice Ordering

Arbitrary Slice Ordering (ASO) allows that the decoding order of slices within a picture may not follow the constraint that the address of the first MB within a slice is monotonically increasing within the NAL unit stream for a picture. This permits, for example, to reduce decoding delay in case of out–of–order delivery of NAL units.

Data Partitioning

In data partitioning mode, each slice can be separated in a header and motion information intra information, and inter texture information by simply distributing the syntax elements to individual partitions. Rather than just providing two partitions as in H.263 version 2 [H26b] and MPEG–4 version 1 [MPG], one for header and motion information, and one for the coded transform coefficients, H.264/AVC can generate three partitions by separating the second partition in intra and inter information. Due to this reordering on syntax level, coding efficiency is not reduced, but obviously the loss of individual partitions still results in error propagation. In addition, the design is simplified by distributing the syntax elements to partitions, which are encoded in different NAL unit types [SM01].

Intra Updates

H.264/AVC like most previous standards allows encoding image regions in Intra mode, i.e., without reference to a previously coded frame. In a straightforward way completely Intra coded frames might be inserted. However, note that H.264/AVC distinguishes IDR frames and ”open GOP” intra frames whereby the latter ones do not provide the random access property as possibly frames ”before” the intra frame are used as reference for ”later” predictively coded frames. In addition, intra information can be introduced for just parts of a predictively coded image. H.264/AVC allows encoding of single MBs for regions that cannot be predicted efficiently or due to any other case the encoder decides for non–predictive mode. The intra mode can be modified such that intra prediction from predictively coded MBs is disallowed. The corresponding constraint intra flag is signaled in the PPS.

Redundant slices

A redundant coded slice is a coded slice that is a part of a redundant coded picture which itself is a coded representation of a picture that is not used in the decoding process if the corresponding primary coded picture is correctly decoded. The redundant slice should be coded such that there is no noticeable difference between any area of the decoded primary picture and a decoded redundant picture.

Flexible Reference Frame Concept

H.264/AVC allows selecting reference frames in a flexible way on MB basis, provides the possibility to use two weighted reference signals for MB inter prediction, allows keeping frames in short-term and long-term memory buffers for future reference, and finally it provides the possibility to introduce non-referenced frames. Therefore, the classical I, P, B frame concept is replaced by a concept which can be exploited by the encoder for different purposes, for compression efficiency, for bit-rate adaptivity, for error resilience, as well as for other features such as ahead-of-time transmission of pre-stored commercials in real-time applications.

3.3.4 Stream Switching with H.264/AVC

Concepts

Bitstream switching provides a simple, yet powerful means to support bit-rate adaptivity in wireless streaming environments. In this case, the transmitter streaming server stores the same content encoded with different versions in terms of rate and quality. Assume that for each source unit s_n several versions $m = 1, \ldots, M$ can be generated. The versions are represented by individual data units $\mathcal{P}_{n,m}$. Versions represent the encoding of the same video frame at different quality-rate levels. The reconstructed version of each source unit is denoted as $\hat{s}_{n,m}$. We restrict ourselves in the following to the practical case where the spatial and temporal resolution as well as the frame dependency structure for each version is of identical structure. Generalization to different structures for each version is straightforward.

Figure 3.16 shows the concept of bitstream switching. In Figure 3.16a) the concept of provisioning multiple versions and switching between the two is shown. These two versions result from encoding of the same original video sequence with for example two different quantization parameters or two different target bit-rates. However, it is important that switching cannot take place at arbitrary pictures as in this case a mismatch between the prediction chain at the encoder and the decoder would occur as shown in Figure 3.16b). Therefore, each of these versions must include means to randomly switch into it referred to as switching units. IDR pictures provide this feature as shown in Figure 3.16c), but they are also costly in terms of compression efficiency (for an analysis of bitstream switching for streaming, see [XZ05]).

SP-picture concept in H.264/AVC

The Switching–Predictive (SP)-picture concept in H.264/AVC [KK03] addresses the compression efficiency issue of IDR pictures for the purpose of stream switching. Figure 3.17 shows the concept for two versions: In this case, the streaming server not only stores different versions of the same content, but also Secondary SP (SSP)-pictures as well as Switching–Intra (SI)-pictures. As long as the bit-rate does not change, efficient primary SP-pictures are transmitted at the pre-selected possible switching points. If switching becomes necessary, one can rely on SSP or SI pictures.

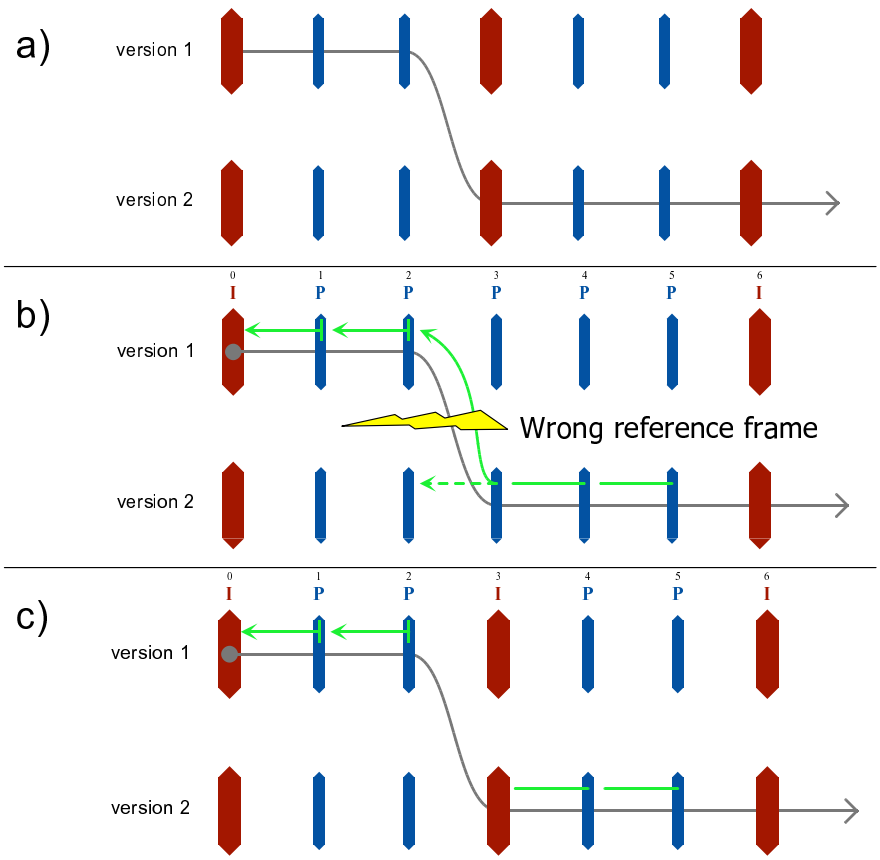


Figure 3.16: Bitstream Switching: a) Concept of provisioning two streams and switching between the two; b) Mismatch problem when switching arbitrarily; c) IDR frame switching

As SP frames can be identically reproduced by SSP or SI pictures mismatch free decoding for all following predictive frames is ensured. An extension to more than two versions is straightforward, but is omitted here for the sake of clarity.

The SP-picture concept allows applying predictive coding even in case of different reference signals. This is accomplished by performing the MCP process in the transform domain rather than in the spatial domain. The reference frame is quantized with a finer quantizer than the one for the original frame before it is forwarded to the reference frame buffer. The resulting primary SP-pictures are placed in the encoded bitstream in regular intervals. Primary SP-pictures are slightly less compression-efficient than regular P-picture, but significantly more efficient than regular IDR-pictures. The major benefit results from the fact that the quantized reference signal can be generated mismatch-free using any other prediction signal.

Note that within the SP picture concept in H.264/AVC, two different quantization parameters exist: QPSP and QPSP2. The residual error signal, which is inserted in the primary bitstream, is quantized with QPSP. Hence, the rate-distortion performance of the primary bitstream will greatly depend on the choice of QPSP. The parameter QPSP2 is used to quantize the reconstructed image and therefore will have implications on the quality of the prediction signal and the reconstructed image. The quality of the prediction signal will finally also influence the residual error signal in such a way that better prediction signals give smaller residual error signals and therefore require less bit-rate for entropy coding. To generate the reference signal without any previous dependencies, SI-pictures can also be used, which are only slightly less inefficient than common I-pictures, but can also be used for adaptive error resilience purposes. For more details on this unique feature

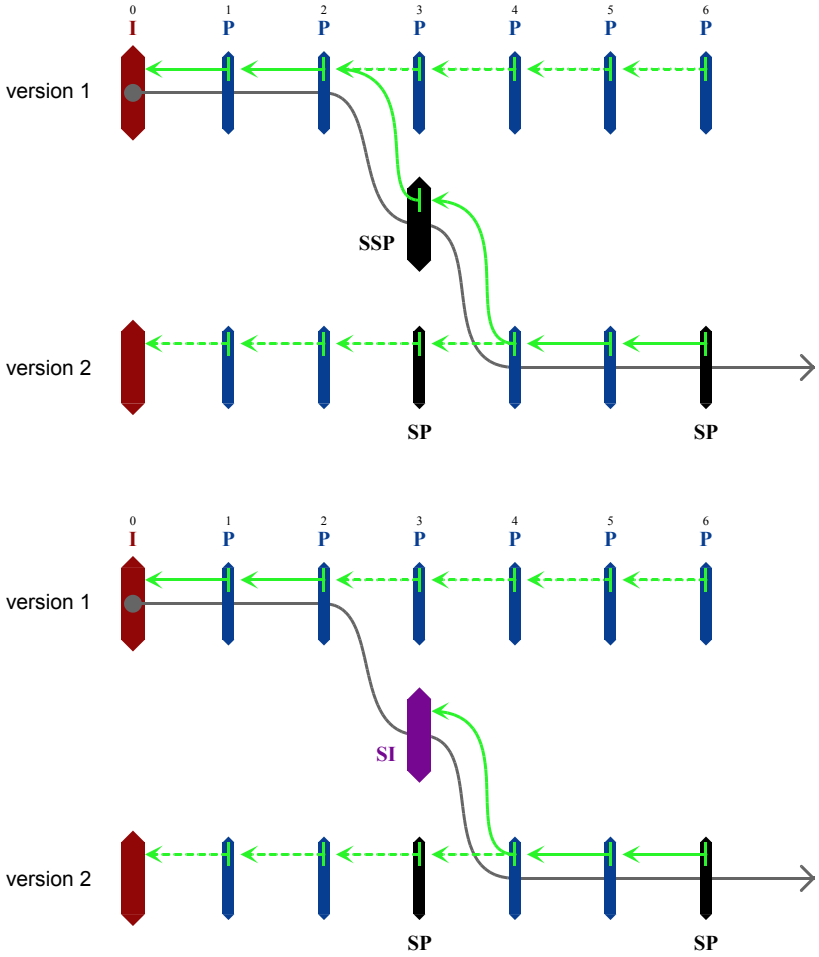


Figure 3.17: Bitstream switching with SP- and SI-pictures in H.264/AVC.

within H.264/AVC the interested reader is referred to [KK03].

Also note that some preliminary work on bitstream switching using the SP picture concept for congested links has been presented in [SG05] and [THG05] . The availability of different versions will be integrated in the formalization in section 3.6 and will be applied also in a streaming system design in chapter 9.

An Optimized Encoder for SSP/SI-Pictures

An encoder realization for generating primary SP-pictures is already included in the H.264/AVC test model software. In addition, we have developed an optimized encoder for SSP-pictures, as well as for SI-pictures. The main features are provided in the following; for details we refer to [Wal04]. The respective encoder structure for SSP-pictures is shown in Figure 3.18. Here, lower-case letters, e.g., l, refer to quantized signals, while capital letters, e.g., L, refer to non-quantized signals. Furthermore, signals in the transform domain are indicated by the letter ”l”, while signals in the pixel domain are indicated by the letter ”f”. The individual meaning of a signal, e.g., $pred$ for ”predicted”, can be concluded from its index.

According to Figure 3.18, we obtain the SSP-picture for switching from source stream 1 to target stream 2 by extracting and combining information from the encoding of both versions. The

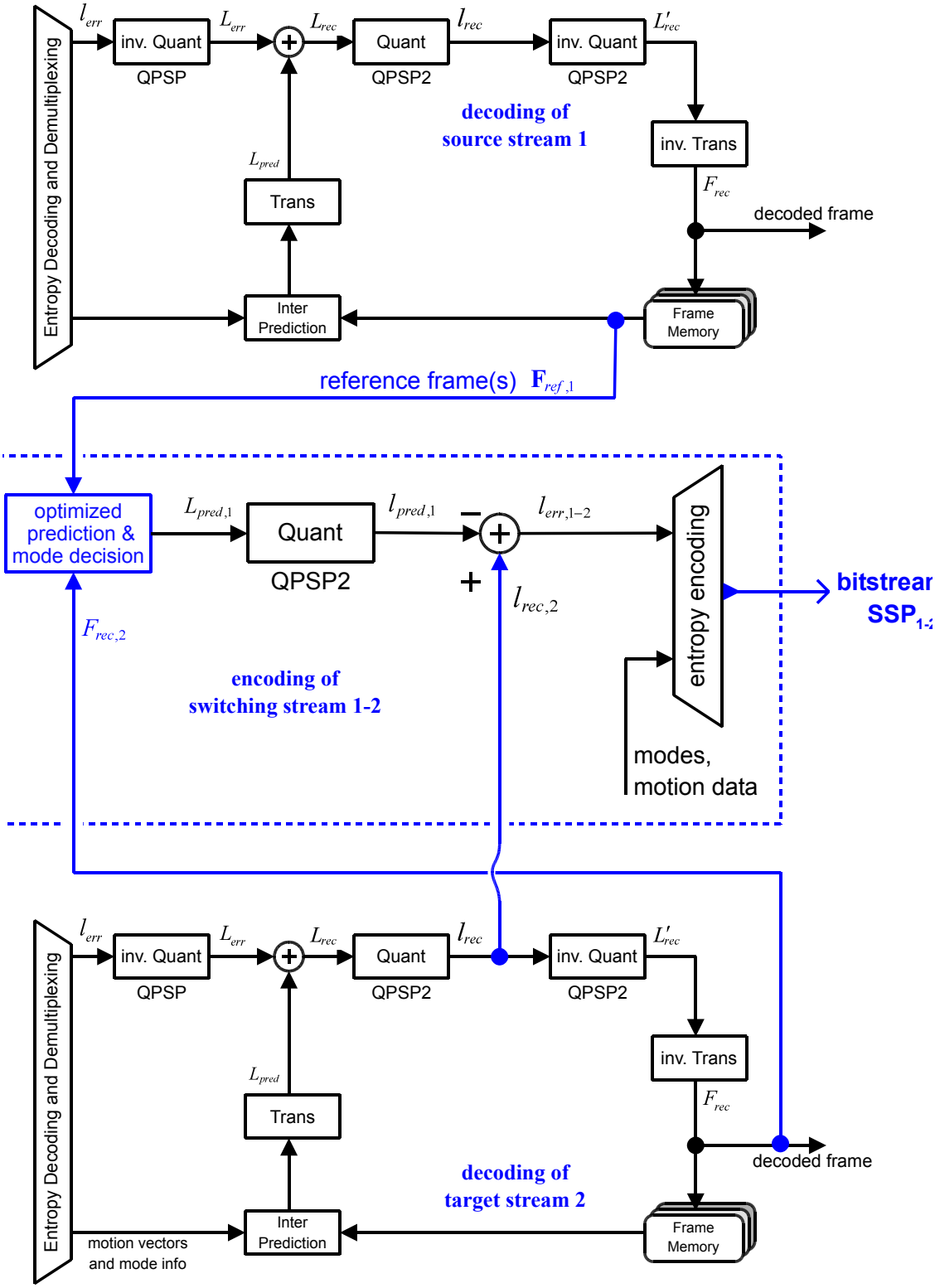


Figure 3.18: Optimized secondary SP-picture encoder.

encoding process for the secondary representations depends on the signal $l_{\text{rec},2}$ that is generated in the encoding and decoding process of the target primary SP-picture. We use the decoding process of target stream 2 for exporting $l_{\text{rec},2}$ as shown in Figure 3.18. Secondary SP (SSP) picture encoding also requires the prediction signal $L_{\text{pred},1}$. In our implementation, $L_{\text{pred},1}$ is generated using all reference frames $\mathbf{F}_{\text{ref},1}$, which are available by decoding source stream 1. For SI-pictures the same concept applies with the only difference that the prediction signal can be computed without any signals exported from stream 1.

It is also worth to mention that the straight-forward approach to simply use the prediction signal, motion vectors, and modes from encoding/decoding the primary source stream 1 is not efficient: The partition modes and the motion vectors chosen for encoding the primary SP pictures do not necessarily fit well for encoding the SSP and result in a suboptimal prediction signal with a large prediction error $l_{\text{err},1\to2}$. This implies that coding efficiency is low, as the residual has to be encoded without any further quantization. Hence, a prediction signal $L_{\text{pred},1}$ is required which minimizes the residual. Since no restrictions apply on $L_{\text{pred},1}$, we can optimize it by using all available reference frames $\mathbf{F}_{\text{ref},1}$. Classical operational rate-distortion optimization as used in the H.264/AVC test model and presented in subsection 3.2.5 is applied. However, the encoded SSP shall be identical to the primary SP-reconstruction of the target stream. The objective of the motion estimation and compensation must therefore be to match the reconstructed primary target frame $F_{\text{rec},2}$, rather than the original frame F_{orig}. With this modified mode selection we save up to 10% in bits for SSP-picture coding compared to the case when we use the prediction signal optimized to F_{orig}.

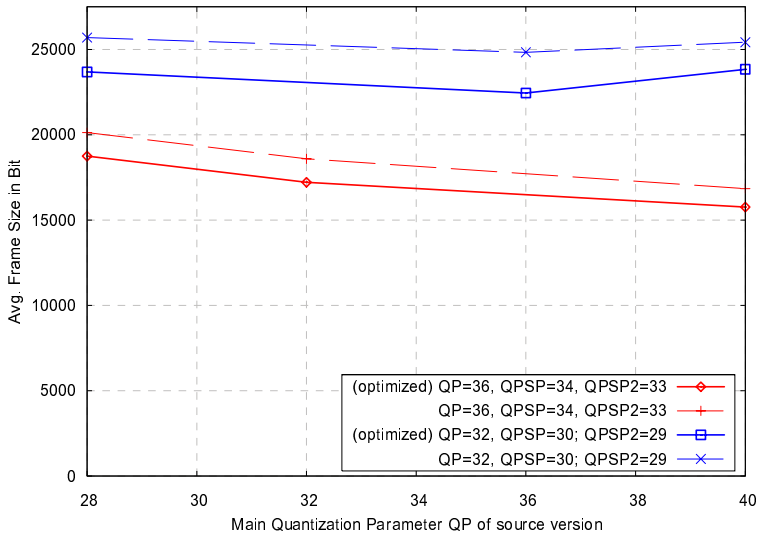


Figure 3.19: Average frame sizes for switching SP-frames over the quantization parameter QP.

In Figure 3.19 the average frame sizes for switching SP-frames are compared over the quantization parameter QP of the original signal for a typical test signal. The dashed curves show the SSP-frame sizes when the original source picture is used for motion compensated prediction in the operational rate-distortion optimization, whereas the solid curves were generated using the reconstructed primary SP-frame as the target picture for prediction. It is observed that savings of SSP bit-rate from 4% up to almost 10% are possible by applying this optimization. For more details on additional encoding results as well as the exact encoder implementation we refer to [SG05] and [Wal04].

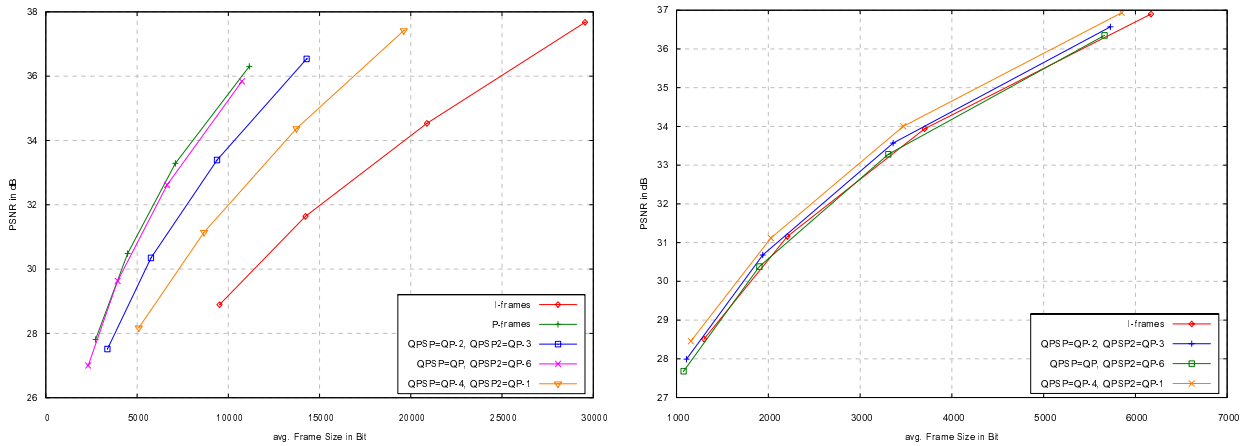
Experimental Results and Guidelines on Quantization Parameter Selection for SP-pictures

Following the brief introduction of the SP-picture concept, some selected performance results as well as guidelines for the use of SP-pictures are discussed. All coding results are generated for specifically constructed video sequence consisting of alternating scenes with high and low motion as well as high and low detail levels. The test sequence is encoded with a GOP size of 30 frames with the following structure: X1-B3-B4-P2-B6-B7-P5--B28-P26-B29-B30, where the first frame in each GOP X1 is either a (primary or secondary) SP- or an IDR-frame. In any case, the very first frame of the sequence is always an IDR-frame.

In a first set of experiments, we compare the rate-distortion performance of SP-frames encoded with different settings for the quantization parameters QPSP and QPSP2 to the standard frame types I and P. The source video sequence is encoded using four different quantization parameters[10] 28, 32, 36 and 40. These serve as the main quantizers QP for all I-, P-, and B-frames of the sequence. For the selection of the SP quantizers, three different rules are applied to illustrate the impact on primary and secondary SP-frame coding efficiency:

1. QPSP = QP-2 and QPSP2 = QP-3,
2. QPSP = QP and QPSP2 = QP-6,
3. QPSP = QP-4 and QPSP2 = QP-1.

Figure 3.20a) shows the rate distortion performance in terms of PSNR over the average frame size for the GOP boundary frames X with being either an I-, P-, or primary SP-picture. Note that the P-picture performance is only shown as a reference case.



(a) GOP Boundaries only (b) entire sequence

Figure 3.20: PSNR over average frame size in bits for different coding types of the GOP boundaries; (a) GOP boundary only (b) entire sequence.

Comparing I and P frames, I frames have significantly worse coding efficiency. However, note that the quality of I-frames for the same quantization parameter is consistently better by about 1 dB than that of P-frames. For SP-frame coding, especially for rule 2) from above it is obvious that primary SP frames can approximately achieve P-frame coding efficiency. Setting the quantizer QPSP2 to even lower values would further improve the coding efficiency of SP-frames but the

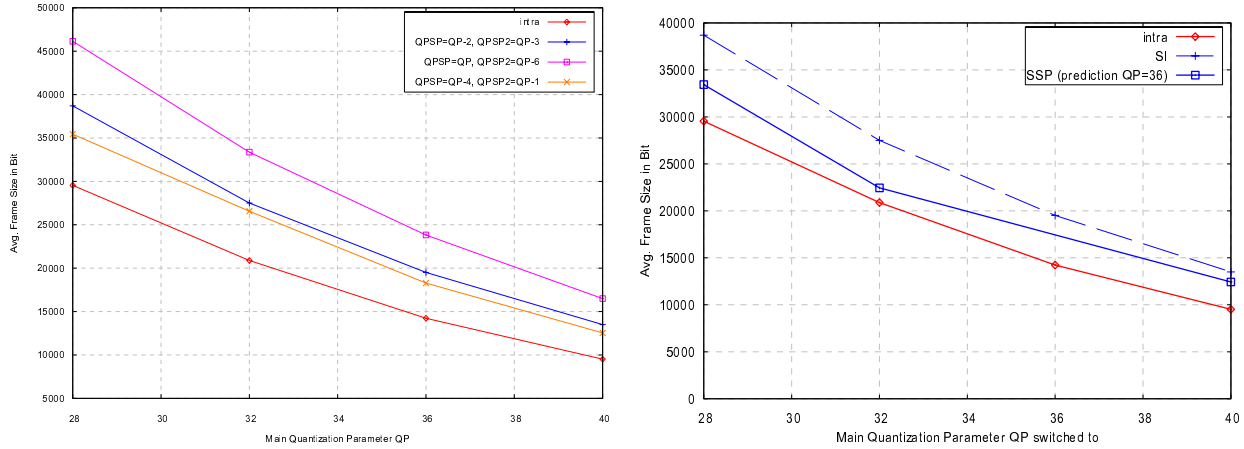
[10] For the applied test model JM3.6 the quantization parameter is 12 higher than for JM1.7 and is identical to the one of the final standard.

P-frame performance cannot be exceeded. Therefore, by observation from the graph, it is obvious that further reductions of QPSP2 will be of little benefits. Using higher values for the QPSP2 according to rule 1) and 3) degrades the SP-frame quality.

However, as the SP-frames are also used for reference, the selection of the QPSP strongly affects the rate-distortion performance of the entirely encoded sequence. This is shown Figure 3.20b) where rate distortion performance in terms of PSNR over the average frame size for all frames of the sequence is shown. One can observe that even though the coding efficiency of quantization rule 2) results in a similar performance of SP-frames and P-frames, such SP-frames are much worse reference frames and therefore lead to a degradation in coding efficiency for the complete stream. When applying one of the other two rules this effect is less prominent, they achieve better total rate-distortion performance and are therefore more suitable.

The performance of SP-frames is relevant for the regular transmission. The performance of the secondary representations is relevant for the switching case. As the reconstructed picture of primary and secondary representation is identical, for performance comparison of different schemes only the frame size of SSP- and SI-frames matter.

Figure 3.21 shows the resulting average frame size in bits of the secondary representation (switching frame) over the main quantization parameter QP of the version that is being switched to (target version). In Figure 3.21a) the average size of SI-frames using the three different QPSP and QPSP2-rules is compared and also the I-frame performance is shown. SI frames perform always worse than I-frames. We also observe that the frame size of the SI-frame is almost exclusively determined by the quantization parameter QPSP2: Higher values of QPSP2 result in smaller secondary frame sizes.



(a) SI Frame performance

(b) SSP frame performance

Figure 3.21: Average frame size in bits over main quantization parameter for different coding types of the GOP boundaries

In Figure 3.21b) the average size for SSP-frames is compared to that of I- and SI-frames. Rule 1) from above is applied for SSP frame quantization. All curves show only three points instead of four, as SSP-frames are generated for switching to a different version. Most of the aspects we already mentioned for SI-frames also hold for SSP-frames but note that SSP frames yield significantly lower frame sizes than SI-frames. However, with the quantization parameters chosen according to rule 1), the SSP-frame sizes are bigger than I-frame sizes. Note that rule 3) reduces the SSP frame sizes by approximately 5% to 10%, whereas applying rule 2) increases the frame sizes by approximately 20% and is therefore not shown. Also note that generally switching up from a lower quality to higher quality stream will require more bits than switching down to the

lower quality stream.

We will briefly summarize the most important results and provide some guidelines for the selection of the quantization parameters QPSP and QPSP2.

- The primary SP-frame rate-distortion performance is influenced by both QPSP and QPSP2. The size of a secondary representation is only related to QPSP2.
- SI-frames are always larger than regular I-frames, SSP-frames can potentially be smaller than I-frames, depending on the choice of quantization parameters and the actual video sequence.
- In order to achieve high coding efficiency for regular streams, it is important that the inserted SP-frames generate high quality reference frames, achieved by low QPSP.
- An estimation of ratio of secondary representations to primary SP-frames in a specific scenario is essential for the appropriate selection of the quantization parameters: If many secondaries frames are expected, QPSP2 should be maximized and rule 3) from above is appropriate. In case mainly primary SP-frames are transmitted, the primary stream coding efficiency should be optimized and rule 1) is more appropriate.

3.4 Scalable Video Coding

3.4.1 Background

As already outlined, scalable source coding refers to a source coder which simultaneously provides encodings of the same data source at different quality levels in one coded representation. Unlike for model sources or speech samples with usually only the distortion defining the quality levels, for video the quality can be changed in basically in three dimensions, namely the spatial resolution, the temporal resolution or frame rate, and the quantization distortion.

A huge amount of research has been conducted in the area of scalable video coding and, recently, new efforts within the Scalable Video Coding (SVC) group of MPEG have been quite far progressed to come up with a new scalable video coding standard. In the following we concentrate on some concepts and standard relevant developments, and introduce a FGS extensions of a test model version of H.264/AVC to show the potentials of scalable video in network environments. Motivated by the successes in embedded still image coding pioneered by Shapiro's EZW coding algorithm [Sha93b] and Said and Pearlman's SPIHT coder [SP96] extensions to video have been introduced. Rate–scalable, embedded video coding was first proposed by Taubman and Zakhor [TZ94] using 3–D subband coding. Based on this ground–breaking work, a number of embedded 3–D video coding algorithms such as in [Ohm94, CW99, KXP00, HW00] were proposed which combine 3–D subband coding with motion compensation. Many of these methods have been rediscovered and refined, and have been evaluated for SVC within MPEG. A detailed summary of different technologies is for example presented in [Tau03]. However, surprisingly enough, after considering many complex methods, the methods finally adopted in SVC are quite similar to those presented in the PTVC as introduced in 3.4.2.

After some first attempts in MPEG-2 and H.263 to add scalable extensions, in MPEG–4 version 2 scalability has been extended under the acronym FGS [Li01] to allow for a larger number of layers. To avoid additional complexity and motion drift effects for the error signal between the original signal and the reconstructed signal in the common MPEG–4 base layer an intra frame

coding method. Despite its simplicity the resulting low compression efficiency prohibited the use of the FGS in any practical application. Non–standard compliant extensions have been discussed in the literature where compression efficiency is increased at the expense of motion drift effects in the lower layers, see e.g. [WLZ01] or [vdSR02]. Along these lines we have introduced in [BSM$^+$01b, SB01, BSM$^+$01a] an extension to an early H.264 test model, namely TML5, which uses the MCP process of this test model, but a different coding for the texture.

3.4.2 Progressive Texture Video Coding

Scalable video coding basically has a fundamental problem in combination with MCP: The encoder incorporates a model of the decoder which assumes that all encoded information has been received and properly decoded. If the decoder receives just parts of the encoded bit–stream, the encoder basically can operate in one of three principles:

- it can apply multiple MCP stages in encoder an decoder resulting in high complexity and lower compression efficiency,
- the reference signals in the decoder and the decoder in the encoder are allowed to differ to some extent resulting in mismatches and degraded quality, or
- the enhancement layer ignores temporal redundancy resulting in significantly reduced compression efficiency.

As we opt for reasonable complexity decoders, the proposed PTVC avoids multiple decoding loops and is based on an efficient mixture of the latter two principles. The goal in the design of the PTVC is a minimum change of most of the features in H.264 but also to provide an embedded bit–stream and good compression efficiency.

The PTVC is based on the H.26L test model TML5.0 which has been modified to perform motion estimation and compensation only, i.e., the coding of transform coefficients has been disabled. TML5.0 supports only parts of the final H.264/AVC features, namely block based motion compensation with variable vector block sizes for each MB with vector blocks of square or rectangular shape, MCP with quarter–pel accuracy, backward–prediction only, and multiple reference frames.

Instead of the regular texture coding in TML5.0 with quantization, run–length coding and UVLC entropy coding, an embedded bit–plane coding is used to code I frames and the DFD resulting from the MCP process. The resulting packet for each frame consists of the combination of the TML5.0 control and motion information and the progressively coded texture bit–stream. Here, we only give a brief overview of the codec. For a detailed description of the PTVC we refer to [Buc00], [BSM$^+$01b] and [BSM$^+$01a]. To simplify the codec description and to focus on the main features we present a simplified version of the PTVC in the following. We omit the presentation of intra–frame coding, the loop–filter operation, and the prediction from multiple reference frames although all these features are included in the PTVC. The motion apparatus is identical to TML5.0 including a block based motion compensation with variable vector block sizes for each MB and quarter–pel motion vector accuracy.

The simplified encoder is depicted in Figure 3.22. We assume that the input video is pre–processed such that constant frame rate f_s is achieved and, therefore, the discrete time–index t specifies each frame. Assume that the reference frame buffer contains a video frame $\tilde{s}_{t-1}$. The video frame s_t at time t is processed as follows: the PTVC encoder performs motion estimation for s_t referencing $\tilde{s}_{t-1}$. The resulting motion vectors are used in the MCP process and the motion–compensated frame is subtracted from the input frame s_t. For each color component the resulting

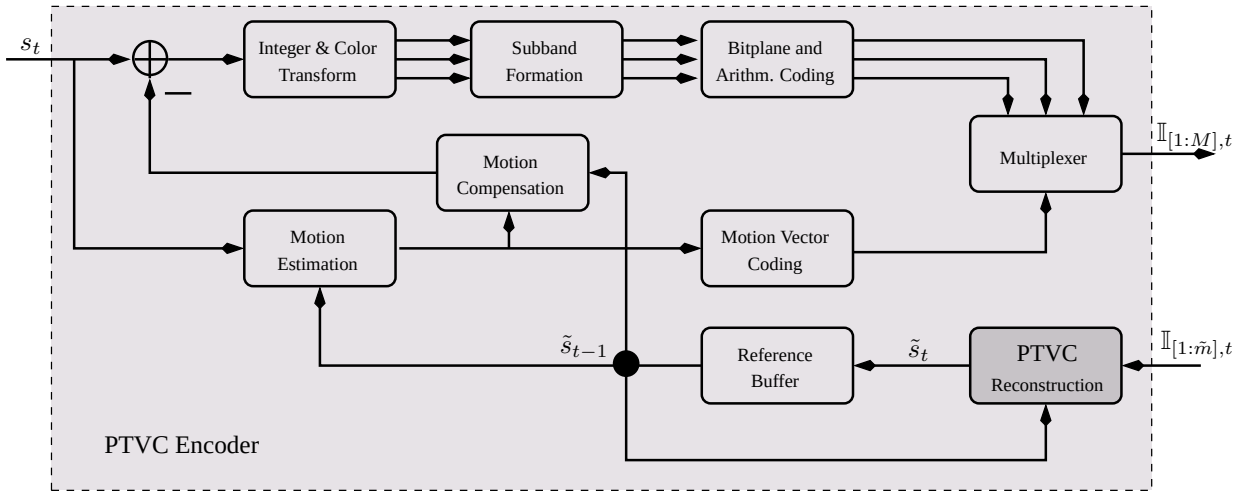


Figure 3.22: Simplified open loop PTVC encoder.

DFD is then encoded to obtain an embedded bit–stream for each frame. For this, the 4×4 integer transform from TML5.0 is applied and the coefficients of the entire frame are arranged into 16 subbands, such that, for example, all DC coefficients are arranged in the upper left subband.

The quantization and entropy coding applied to the DFD is similar to techniques used in SPIHT and JPEG–2000 based on bit-plane and context-based arithmetic coding. For each bit encountered in the bitplane representation of the absolute values of a given coefficient, a distinction is made between significance and refinement bits as shown in Figure 3.23. Significance bits are those bits indicating whether a given coefficient has already become significant, i.e., whether its Most Significant Bit (MSB) has already shown up in a given bitplane. Every bit below the MSB is interpreted as a so–called refinement bit. The remaining spatial correlations of the significance information within a subband are exploited by applying adaptive context–based arithmetic coding. Similar concepts were applied later in the CABAC entropy coding modes of H.264/AVC [MSW03]. In contrast to significance bits, refinement bits usually show only weak spatial correlations and could be considered as uniformly and independently distributed. The bitplane coding process is divided into three different scans of each bitplane. The first scan operates on non–singular significance information, that means, in this scan only those bits will be coded, which are significance bits and which have at least one significant neighbor. For this purpose, the significance coding routine and, if an MSB is encountered, the sign coding routine are used. In the second scan the refinement bits are coded. The third scan finally collects all remaining bits, which have not been coded, by either of the two preceding scans using the same coding routines as in the first pass. All bits of the coefficients are visited and examined in raster scan order within a given subband. The actual coding, however is performed according to the scan conditions. The processing of the subbands is performed in global zig-zag scan within each bit plane.

The context formation for encoding of significance bits and sign bits is done by mapping a neighborhood of the related coefficient to a reduced number of context states. Coding of significance bits is performed by using a context which depends on the significance state of the local 8-neighborhood. The 256 states are quantized to three final contexts which describe the activity of the neighbors (no, low, high activity). Sign bits are coded with a context depending on sign and significance states of the local 4-neighborhood. This routine is only invoked, when a coefficient indicates its significance for the first time. Refinement bits are coded with one single statistical

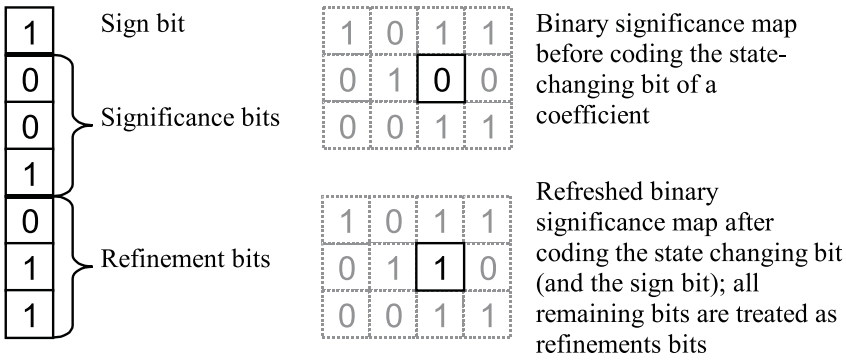


Figure 3.23: Bitplane representation of the absolute values of a coefficient.

model.

The control data and motion vectors are coded with UVLC just as in TML5.0. Data of a single encoded frame, namely control information, motion vectors, and the independently coded texture information for each color component are multiplexed as follows: The control and motion vector data is put in the beginning of the video stream. Then the texture information is added such that first R_Y bytes from the Y bitstream, one from U, and one from Y are appended. This is repeated until all symbols of all color components within the current frame have been distributed.

This ordering of the bit–stream results in an index sequence $\mathbb{I}_{[1:M],t}$ and progressive binary representation $\mathbb{B}_{[1:M],t}$ for each input frame s_t. $\mathbb{B}_{[1:M],t}$ consists of M prefixes $\mathbb{B}_{[1:m],t}$ which allows reconstructing s_t at M different rates. Whereas the first part containing the control and motion data usually has at least a few hundred bits, the texture part is completely progressive in the sense that any bit enhances the reconstruction quality. Therefore, all layers $m > 1$ can be as small as one bit. This presentation concludes the forward part of the encoder.

Since the decoder is part of the encoder as in any other hybrid video coding scheme we continue with presentation of the decoder before presenting the backward path in the encoder. The schematic decoder is shown in Figure 3.24. Assume that the reference frame buffer contains a video frame s'_{t-1} and a certain bit–sequence $\mathbb{B}_{[1:\breve{m}],t}$ is received where the received bit–sequence might be shorter than the original bit–sequence, i.e., $\breve{m} \leq M$. The bit–sequence is mapped to a received index sequence $\hat{\mathbb{I}}_{[1:\breve{m}],t}$ and is appropriately demultiplexed to obtain control and motion vector data as well as one progressive bit–stream for each color component. Reconstruction of the texture data is then performed in reverse order of encoding – arithmetic decoding, bit–plane reconstruction, subband decomposition and inverse integer transform. Texture decoding results in a reconstructed DFD at the decoder. The sum of the DFD and the motion compensated reference frame is the reconstructed video frame $\hat{s}_t$ at time t.

In a similar way as the decoder generates the reconstructed frame, the encoder produces the reference frame $\tilde{s}_t$. However, instead of using the indices of all layers $\mathbb{I}_{[1:M],t}$, the encoder might used just parts of it, namely the first $\tilde{m} \leq M$ parts. The sum of the reconstructed DFD and the motion compensated reference frame result in the new reference frame $\tilde{s}_t$ at time t and is forwarded to the reference frame buffer. The number of layers used for the generation of the reference signal $\tilde{m}$ are signaled in the bit–stream as part of the control data.

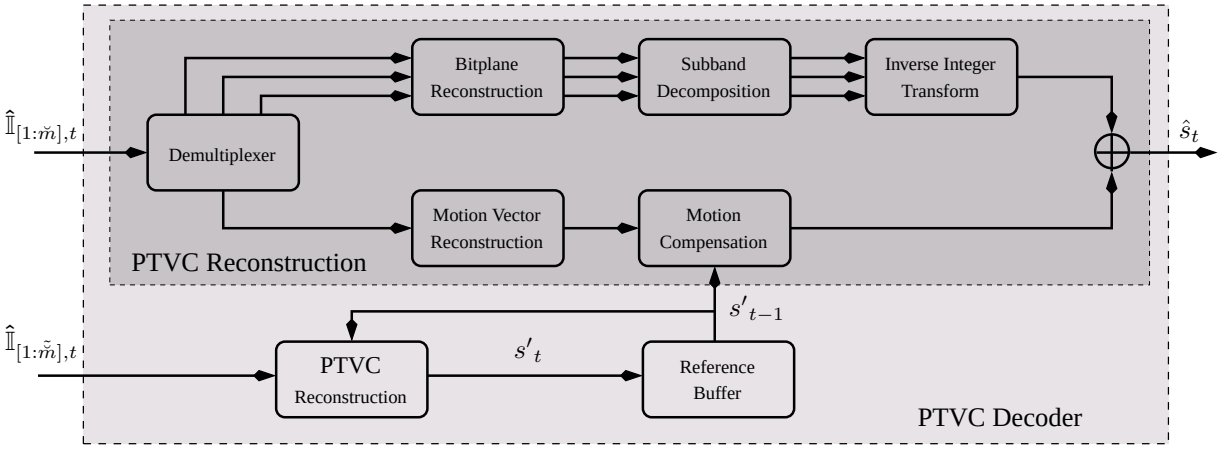


Figure 3.24: Schematic PTVC decoding algorithm.

The new reference frame at the decoder s'_t is in general not identical to the reconstructed frame $\hat{s}_t$. If the received number of layers $\breve{m}$ is at least as high as the signalled reference layer $\tilde{m}$, i.e., $\tilde{m} \leq \breve{m}$, then the reconstruction of the reference frame at the decoder is performed using the first $\tilde{m}$ parts just as done for the reconstructed and displayed frame $\hat{s}_t$. If the number of received layers $\breve{m}$ is below the signalled number of $\tilde{m}$, i.e., $\tilde{m} > \breve{m}$, then the decoded and displayed frame $\hat{s}_t$ and the reference frame at the decoder s'_t are identical. However, note that in this case the reference frame in encoder and decoder are different, i.e., $\tilde{s}_t \neq s'_t$, which results in a mismatch.

Although the encoder and the decoder have to reconstruct two frames at each time instant t, the same motion compensation signal can be used and the bit–plane reconstruction is for the signal reconstructed with lower number of layer is already included in the one for the higher number of layer. Therefore, the complexity of the second reconstruction is quite small.

3.4.3 Rate Control and Reference Frame Layer Selection

Similarly to regular video codecs, the PTVC has several options to be selected during the encoding and transmission process. However, due to the progressive nature of the coded signal, additional flexibility is given to the encoder. Basically, the encoder has to select the number of transmitted layers, $\bar{m} \leq M$, and the number of layers used for the reference signal, $\tilde{m} \leq M$ for each source frame s_t. Whereas the selection of the transmitted layers can be viewed as a rate control scheme, the selection of the reference frame layer is crucial for the trade–off between a high–quality reference signal ($\tilde{m}$ should be as high as possible) and a mismatch in case that the same reference frame can not be reconstructed at the receiver, i.e., $\breve{m} < \tilde{m}$ such that $\tilde{\breve{m}} \neq \tilde{m}$ ($\tilde{m}$ should be as low as possible to allow dropping of refinement information).

We apply consistent notation as introduced in subsection 3.2.5. We consider a sequence of N_s video frames s_t coded with the PTVC encoder resulting M layer and M rate–distortion pairs denoted as $r_{t,m}$ and $d_{t,m}$, respectively. Again we assume that this data is known by appropriate measurements at the encoder, and that both, the rates $r_{t,m}$ are monotonically increasing with the layer index m, and the distortions $d_{t,m}$ are monotonically decreasing with increasing rate $r_{t,m}$, and layer index m. Note the latter assumption on the distortions can only be guaranteed on average, but not for each s_t. As all layers $m > 1$ represent the progressively coded texture, we assume

bit–granularity for these layers, i.e.,

$$\forall_{m=2,\ldots,M} \quad r_{t,m} = r_{t,m-1} + 1.$$

The maximum allowed rate of the control and motion data, r_{mv}, is fixed before the encoding process. The encoding process guarantees that the rate of layer, $r_{t,1}$, which contains control and motion vector data is below r_{mv} if r_{mv} is reasonably selected. Usually, we use for r_{mv} a value of about 25% of the expected transmission rate. In the unlikely case that in a first encoding attempt $r_{t,1}$ exceeds r_{mv}, the motion estimation process is repeated by excluding more and more "bit-rate costly" MCP modes until $r_{t,1} \leq r_{\text{mv}}$. In case that the first layer cannot be reconstructed, we assume that previous frame concealment is applied, i.e., s_t is reconstructed with $\tilde{s}_{t-1}$. $\tilde{s}_0$ is assumed to be an all grey frame. For completeness, we add a rate–distortion pair for $m = 0$ with $r_{t,0} \triangleq 0$, and the distortion $d_{t,0}$ is defined by the distortion when reconstructing s_t with $\tilde{s}_{t-1}$.

For the selection of the reference frame layer we distinguish two different modes. In a first mode we assume that the encoder exactly knows the available number of bits for the transmission of source s_t, specified by some number $r_{c,t}$. In this mode, referred to as *known decoding layer* mode, it is obvious that the encoder chooses the reference frame layer $\tilde{m}_t$ such that reference signal $\tilde{s}_t$ is as good as possible, i.e.,

$$\forall_t \quad \tilde{m}_t = \arg \max_{m=1,\ldots,M} r_{t,m} \quad \text{such that} \quad r_{t,m} \leq r_{c,t}. \tag{3.24}$$

Then, the PTVC basically operates like any other single layer coder. The reconstructed and displayed frame at the decoder, $\hat{s}_t$, the reference frame at the encoder, $\tilde{s}_t$, and the reference frame at the decoder, s'_t, are identical.

In the *unknown decoding layer* mode, the encoder is not aware of the channel rate $r_{c,t-1}$, and consequently the highest layer the decoder can access is not a priori known to the encoder. Therefore, the encoder specifies the reference frame layer $\tilde{m}_t$ to an arbitrary number, possibly taking into account the tradeoff between high–quality reference frames and reference frame mismatch with resulting drift effects between encoder and decoder. The performance trade–off will be discussed in further details in subsection 3.5.3.

For common hybrid video coder, as discussed in subsection 3.2.5, the rate is controlled by the selection of an appropriate QP for each video frame s_t. For the PTVC, like for any other embedded source coder based on bitplane coding, the rate can directly be controlled by selecting the rate $r_{t,m}$ for each s_t. To be consistent with the previous notation, the encoder has to select an optimal transmission layer $\bar{m}_t$. For a low–delay mode, again a CBR rate control is applied. Assuming again a constant transmission bit-rate R and a constant frame rate f_s, the channel rate for each frame is constant and determined as $r_{c,} \triangleq R/f_s$. With the previously discussed assumption $r_{\text{mv}} \leq r_{c,t}$, the optimum transmission layer is determined as

$$\forall_{t=1,\ldots,N_s} \quad \bar{m}_t^* = \arg \max_{m=1,\ldots,M} r_{t,m} \quad \text{such that} \quad r_{m,t} = r_{c,t}. \tag{3.25}$$

For scenarios as discussed for the VBR rate control in subsection 3.2.5, we are only interested that some average rate constraint is maintained. Whereas for hybrid video coders the average rate is controlled by the QP, for the PTVC we propose the direct application of the Lagrangian formulation according to (3.17) whereby the average rate is controlled by the Lagrangian parameter λ_{PTVC}. Therefore, when using a VBR rate control, we select the optimal transmission layer $\bar{m}_t^*$ for each video frame s_t as

$$\forall_{t=1,\ldots,N_s} \quad \bar{m}_t^* = \arg \max_{\bar{m}=1,\ldots,M} \left(d_{t,\bar{m}} + \lambda_{\text{PTVC}} r_{t,\bar{m}} \right). \tag{3.26}$$

The transmission layer selection according to (3.26) is performed in a greedy way meaning that the results of encoding process of frame s_t are used for the selection of transmission layer for frame s_{t+1}. In case that the channel transmission rate is unknown at encoding time, e.g. when offline encoding is performed, then the procedure according to (3.26) is carried out for the selection of the optimal reference frame layer $\tilde{m}_t^*$ during the encoding process.

Note that in case that the encoding including the selection of the reference layer has been performed offline, regardless if the reference layers where chosen arbitrary or with the procedure according to (3.26), then for the selection of the transmission layer according to (3.26) additional effort is necessary for optimized performance: Namely, if the transmission layer for a certain video frame s_t is chosen such that $\bar{m}_t^* < \tilde{m}_t$, then the distortion for frame s_{t+1}, $d_{t+1,m}$, has to be recomputed for all m as the drift has not been considered in the offline distortion computation. As the drift effect propagates, this computation has to be repeated for all s_i with $i > t$.

3.4.4 Group-of-Picture Interleaving

For applications with relaxed delay constraints it is desirable to generate a progressive bit–stream not just for each source frame s_t, but also for a Group-of-Picture (GOP) consisting of N_{GOP} video frames. For the PTVC, this is accomplished by interleaving the individual embedded bit–streams $\mathbb{B}_{[1:M_t],t}$ for all $t = 1, \ldots, N_{\text{GOP}}$ to obtain a single embedded bit–stream $\mathbb{B}_{[1:M_{\text{GOP}}]}$ with $M_{\text{GOP}} \triangleq \sum_{t=1}^{N_{\text{GOP}}} M_t$ for an entire GOP. In the following we treat a GOP as an independent sequence for which the first frame is an I frame and the remaining frames are predictively coded.

For the decoder to be able to reconstruct the individual bit–streams $\mathbb{B}_{[1:\breve{m}_t],t}$ we introduce the interleaving weight $w_{\text{il},t}$ for each frame in the GOP whereby $w_{\text{il},t}$ can be any positive integer. The N_{GOP} and the $w_{\text{il},t}$ for each frame are transmitted in a UVLC coded form as a small header for each GOP. With the $w_{\text{il},t}$ and the received bit–stream $\mathbb{B}_{[1:\breve{m}_{\text{GOP}}]}$ available, the decoder reconstructs the bit–stream for each individual frame $\mathbb{B}_{[1:\breve{m}_t],t}$ by

1. starting with video frame $t = 1$,
2. taking the next $w_{\text{il},t}$ bytes from $\mathbb{B}_{[1:\breve{m}_{\text{GOP}}]}$,
3. appending these bytes to $\mathbb{B}_{[1:\breve{m}_t],t}$,
4. increasing the frame number t by one unless t is equal to N_{GOP}, then set $t = 1$,
5. going back to 2 until the end of $\mathbb{B}_{[1:\breve{m}_{\text{GOP}}]}$ is reached.

Basically, the encoder can choose the interleaving weights $w_{\text{il},t}$ arbitrarily. We define two modes of operation for the encoder. In a fixed mode the encoder only selects the interleaving weight of the first frame, the I-frame in our case, whereas the interleaving weights for all other frames are chosen to 1, i.e., $\forall_{t=2,\ldots,N_{\text{GOP}}} \quad w_{\text{il},t} = 1$.

In the optimized mode, the encoder first performs the rate control according to (3.26) to obtain the optimal transmission layer $\bar{m}_t^*$, then selects an arbitrary interleaving weight for the first frame, $w_{\text{il},1}$, and finally chooses the interleaving weights for the remaining frames as

$$\forall_{t=2,\ldots,N_{\text{GOP}}} \quad w_{\text{il},t} = \left\lceil \frac{r_{t,\bar{m}_t^*} w_{\text{il},1}}{r_{t,\bar{m}_1^*}} \right\rceil . \tag{3.27}$$

In case of offline encoding with unknown transmission rate at encoding time, the selection of the interleaving weights is not performed based on the optimal transmission layer $\bar{m}_t^*$, but on the

reference frame layer $\tilde{m}_t$ selected in the encoding process. With this information, the entire GOP can be interleaved right after encoding and the resulting progressive bit–stream for the entire GOP, $\mathbb{B}_{[1:M_{\text{GOP}}]}$, can be stored at the transmitter. In addition, with this predefined bit–stream structure, the rate–distortion pairs for the entire GOP, denoted as r_m and d_m with $m = 1, \ldots, M_{\text{GOP}}$, can be precomputed. Thereby, the drift effects have to be taken into account for all rate–distortion pairs, for which at least one frame is decoded based on a reference frame which does not match the reference frame in the encoding process.

3.5 Performance of Video Coders

In the following we will give a brief insight into different coding and encoding technologies. The focus is less on absolute values but rather in the comparison different technologies. All simulations are carried out using the QCIF test sequence *Foreman* (30 Hz, 300 frames, 176×144 pels) at a constant frame rate of $f_s = 10$ Hz and for an IPPP. . . -coded sequence. Unless stated otherwise, an intra period of $P_I = 100$ is applied.

3.5.1 Variable Bit-rate Coding

In this section we compare the performance of the PTVC to well-known standard reference video codecs. We will use VBR rate control according to (3.22) for the standard codecs and (3.26) for the PTVC. Figure 3.25 shows the performance in terms of average $\overline{\text{PSNR}}$ over the bit-rate R for QCIF, Foreman with frame rate $f_s = 10$ fps. Consistently from MPEG-4 simple profile (SP) to advanced simple profile (ASP) over TML5.0 the quality is increasing to JM1.7 which is almost identical to the performance of the final standard. Note that the performance of JM1.7 is such that the bit-rate is reduced to less than 60% compared to MPEG-4 SP. The PTVC outperforms all MPEG-4 codecs but is inferior to TML5.0 due to the modified texture coding. However, overall the PTVC approach is good enough for further investigations.

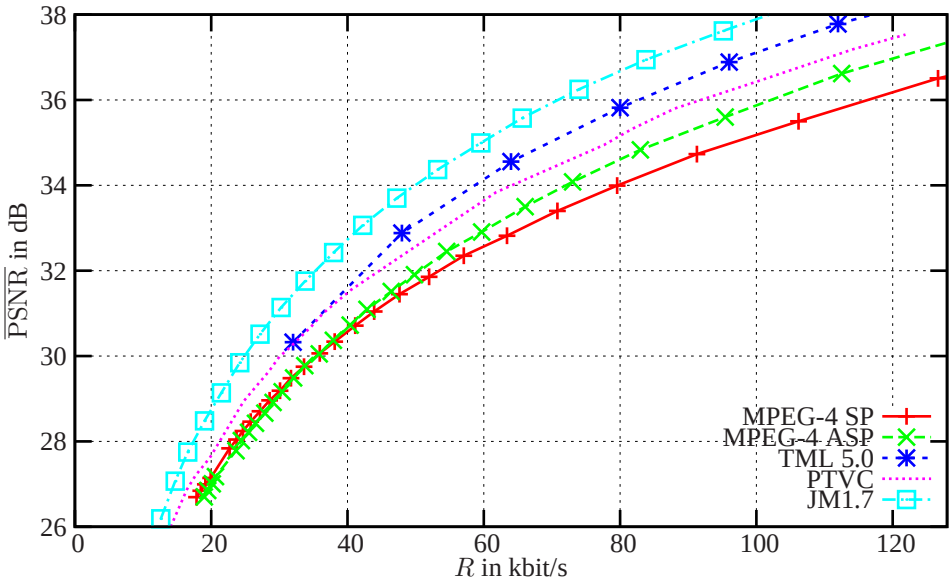


Figure 3.25: Performance of different video coding standards using VBR encoding for QCIF, Foreman with frame rate $f_s = 10$ fps.

3.5.2 Low-Delay Constant Bit-rate Coding

In this section we compare the performance of the PTVC to well-known standard reference video codecs for CBR rate control according to (3.21) for the standard codecs and (3.25) for the PTVC. Figure 3.26 shows the performance in terms of average PSNR, $\overline{\mathrm{PSNR}}$, over the bit-rate R for QCIF, Foreman with frame rate $f_s = 10$ fps compared to the VBR approach. In any case the losses due to the rate control are obvious. However these losses are more severe for the JM1.7 than for the PTVC. Therefore, for the PTVC the use of the CBR rate control is sufficient. Further investigations on the improvement of the PTVC VBR rate control are necessary.

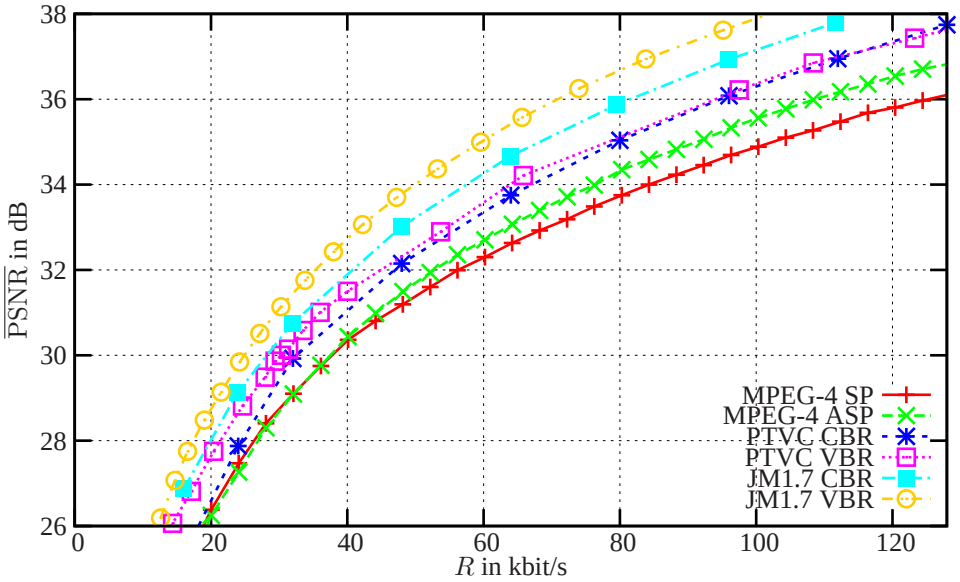


Figure 3.26: Performance of different video coding standards using VBR encoding for QCIF, Foreman with frame rate $f_s = 10$ fps.

3.5.3 Unknown Transmission Bit-rate

In a third set of experiments we investigate the scenario in which the transmission bit-rate r_t is not known prior to encoding. Standard hybrid coders fail to transmit video if the coding rate exceeds the transmission bit-rate. For the PTVC like for any scalable coder this is different. However, the encoder has to select the index $\tilde{m}$ to generate appropriate reference frames. In the sequel we apply CBR rate control for ease of implementation and due to only marginal loss compared to VBR. We define a feedback bit-rate r_f in a sense such that the reference frames are reconstructed by $\mathbb{I}_{\tilde{m}}$ with $\tilde{m} = r_f/f_s$. If the transmission bit-rate is identical to the feedback bit-rate, i.e., $r_t = r_f$, we have the same performance as discussed for CBR rate control (see Figure 3.26). Figure 3.27 shows the performance in terms of average PSNR, $\overline{\mathrm{PSNR}}$, over the bit-rate R for QCIF, Foreman with frame rate $f_s = 10$ fps for different cases including the matching case with $r_t = r_f$.

In contrast to the matching case, if the transmission bit-rate exceeds the feedback bit-rate, i.e., $r_t \geq r_f$, the decoder generates reference frames according to $\hat{\mathbb{I}}_{\tilde{m}} = \mathbb{I}_{\tilde{m}}$. This case is related to the successive refinement idea, i.e., a good quality is gradually enhanced. Despite the reference frames in encoder and decoder are identical, the latter part of the received bit-stream which the decoder can access for representation, is not used to generate the reference frames such that the reference

frames are in average of lower quality than the decoded frames. Figure 3.27 for $r_f = 32$ kbit/s over the transmission bit-rate r_t shows a typical behaviour. Note that virtually any transmission bit-rate r_t can be supported by this scheme. A very slow increase of quality with increasing transmission bit-rate is visible. Similar performance is reported by the MPEG-4 FGS approach [Li01] as also shown in Figure 3.27. Although the base layer has higher quality, the PTVC cannot perform better for higher bit-rates. The reason is due to the low quality reference frames and the almost "white" residuum.

In contrast, if the channel can not support the feedback bit-rate, i.e., $r_t < r_f$, the decoder generates reference frames according to $\hat{\mathbb{I}}_{\tilde{\tilde{m}}} = \mathbb{I}_{r_t/f_s}$. This case is more related to the graceful degradation case. Then, the reference frames in encoder and decoder differ as $\tilde{\tilde{m}} \neq \tilde{m}$. Therefore, encoder and decoder predict from different reference frames and drift is introduced into the MCP process. In Figure 3.27 the performance for $r_f = \{32, 48, 64, 80, 96, 112, 128\}$ kbit/s is shown. Obviously, this drift results in an immediate decrease of the quality even if the transmission bit-rate r_t is only slightly below the feedback bit-rate r_f. In this case only the frequent introduction of intra frames to completely remove the drift as initially introduced in [BSM+01b] can provide satisfying quality. This mismatch can only be circumvented by addition intra information.

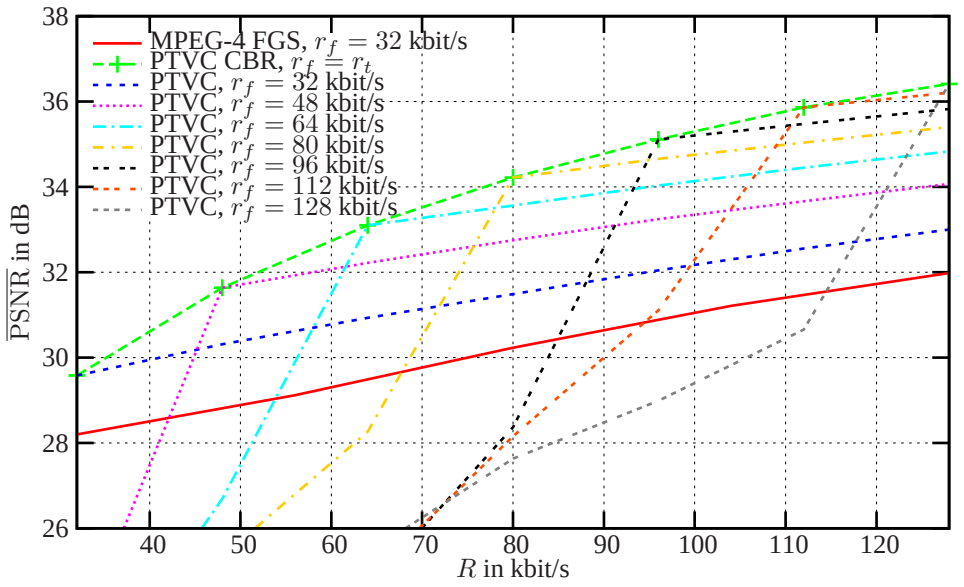


Figure 3.27: Performance over transmission bit-rate for the PTVC with different feedback bit-rates, QCIF, Foreman with frame rate $f_s = 10$ fps.

3.5.4 GOP Transmission Rate

As observed from Figure 3.27, in case of a mismatch and the transmission rate below the feedback rate, $r_t < r_f$, updates in terms intra frames are necessary. Obviously, the more frequent intra frames are introduced, the better is the compensation of mismatches, but also the worse is the compression efficiency. Therefore, we we conduct a simulation, for which we allow the modification of the intra frame frequency P_I. Figure 3.28 shows the performance in terms of $\overline{\text{PSNR}}$ over transmission bit-rate R for the PTVC with different feedback bit-rates and different GOP sizes P_I, QCIF, Foreman with frame rate $f_s = 10$ fps. It is observed that the loss of compression efficiency for low GOP sizes such as $N_{\text{GOP}} = 5$ are in general difficult to compensate and they are only reasonable if

a high variance of the transmission rate and the feedback rate can be expected. However, in general relatively high feedback rates with relatively low intra frequencies, e.g. $N_{\text{GOP}} = 20$ and $r_f = 64$ or $r_f = 96$ kbit/s can provide very good quality over a wide range of transmission bit-rates. A main conclusions from these is results is also that for the purpose of compensating the mismatch only, the introduction of entire intra frames limits the performance quite significantly.

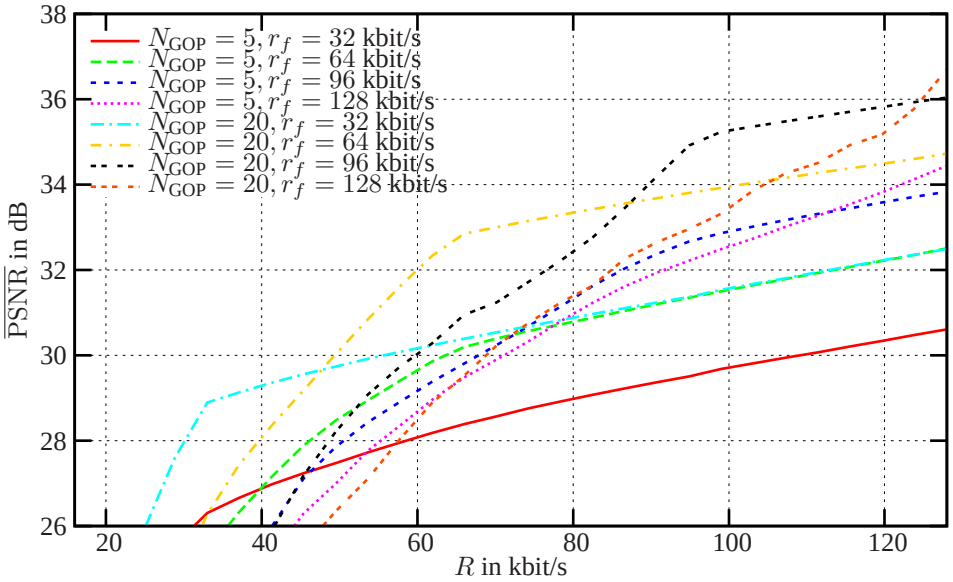


Figure 3.28: Performance over transmission bit-rate for the PTVC with different feedback bit-rates and different GOP sizes, QCIF, Foreman with frame rate $f_s = 10$ fps.

3.6 Formalization of Pre-Encoded Data

3.6.1 Abstraction of Progressively Coded Sources

Progressively coded sources will be used quite frequently in the remainder of this work. Therefore, for simplified and general presentation of different algorithms it is reasonable and convenient to abstract the presentation of a progressively coded source realization s as well as its packetization to a dependent sequence of packets. Assume that this source realization s has assigned a certain operational rate–distortion characteristic, or equivalently some generic rate–quality function $\mathcal{Q}(R)$ where R denotes the total amount of bits used for representation to obtain quality $\mathcal{Q}(R)$. Although we will almost exclusively pick the PSNR or the MSE as fidelity criterion, we continue with the generic expression. We further assume that for any source s the rate–quality function $\mathcal{Q}(R)$ is monotonically non–decreasing with increasing rate R[11].

Usually, the transmission over realistic channels requires an appropriate packetization of any sources, specifically of progressively coded source. Therefore, we asssume that before transmission the source is packetized in M entities, whereby the size (in bits) of each individual layer m is

[11]This assumption is not necessarily true for any source realization and any good source coder if the set of reconstruction points for source coder realization of higher rates or equivalently higher layers does not include the set of reconstruction points of all lower layers, e.g. in case of MSVQ. However, for higher dimensional sources such as still images or video frames the probability that this happens vanishes.

denoted as k_m. For convenience we define the sum up to layer m for all $m = 1, \ldots, M$ as

$$K_m \triangleq \sum_{i=0}^{m} k_m, \tag{3.28}$$

and $K_0 \triangleq 0$ by definition. The packetization might be determined by the underlying packet network or might be an optimization parameter as will be shown in section 5.4. An important parameter in the optimization of transmission systems is the definition of how important a certain transport unit or packet is for the reconstruction quality at the receiver. Therefore, we define the importance, $\mathcal{I}_m$, of a certain packet $\mathcal{P}_m$ containing layer m, as the amount by which the quality at the receiver increases if this packet is correctly decoded. In case of the packetization of a progressively coded source with rate–quality function $\mathcal{Q}(R)$ the importance for all layer $m = 1, \ldots, M$ is easily defined as

$$\mathcal{I}_m \triangleq \mathcal{Q}\left(K_m\right) - \mathcal{Q}\left(K_{m-1}\right). \tag{3.29}$$

Note that with this definition, the quality $\mathcal{Q}_m$, achievable by decoding up to layer m is easily computed as

$$\mathcal{Q}_m \triangleq \mathcal{Q}_0 + \sum_{i=1}^{m} \mathcal{I}_m. \tag{3.30}$$

. For notational convenience we define $\boldsymbol{\mathcal{I}} \triangleq \{\mathcal{I}_1, \ldots, \mathcal{I}_M\}$.

Assume now packetized transmission of each layer: The observed channel for layer m at a receiver is defined as $\mathcal{C}_m \triangleq \mathbf{1}\{\text{layer } m \text{ available}\}$. Hence, combination of a certain observed channel sequence $\boldsymbol{\mathcal{C}} = \{\mathcal{C}_1, \ldots, \mathcal{C}_M\}$, with (3.29) yields the following expression for the (actual) received quality:

$$\mathcal{Q}(\boldsymbol{\mathcal{I}}, \boldsymbol{\mathcal{C}}) \triangleq \mathcal{Q}_0 + \sum_{m=1}^{m} \mathcal{I}_m \mathcal{C}_m \prod_{m=1}^{m} \mathcal{C}_m. \tag{3.31}$$

More important than the actually received quality in (3.31) for system design and optimization is the expected quality. The channel behavior sequence $\hat{\boldsymbol{\mathcal{C}}} \triangleq \{\hat{\mathcal{C}}_1, \ldots, \hat{\mathcal{C}}_M\}$, is in general random, with $\hat{\mathcal{C}}_m \in \{0, 1\}$ the random variable indicating whether layer m is received successfully ($\hat{\mathcal{C}}_m = 1$), or lost ($\hat{\mathcal{C}}_m = 0$). The received quality, denoted as $\mathcal{Q}(\boldsymbol{\mathcal{I}}, \hat{\boldsymbol{\mathcal{C}}})$ is non-deterministic and depends on the randomness of the channel and the specific realization $\hat{\boldsymbol{\mathcal{C}}}$. A suited single measure to estimate the performance of a system is the expected quality defined as

$$E\left\{\mathcal{Q}(\boldsymbol{\mathcal{I}}, \hat{\boldsymbol{\mathcal{C}}})\right\} \triangleq \sum_{\boldsymbol{\mathcal{C}} \in \{0,1\}^N} \mathcal{Q}(\boldsymbol{\mathcal{I}}, \boldsymbol{\mathcal{C}}) \Pr\{\hat{\boldsymbol{\mathcal{C}}} = \boldsymbol{\mathcal{C}}\} \tag{3.32}$$

$$= \mathcal{Q}_0 + \sum_{m=1}^{M} \mathcal{I}_m \Pr\{\hat{\mathcal{C}}_m = 1\}. \tag{3.33}$$

Important for several rate allocation algorithms to work well is the assumption of a convex rate–quality function $\mathcal{Q}(R)$. However, it usually sufficient that the packetized version of the progressively coded source is convex in the quality–rate function which can be equivalently expressed as $\mathcal{I}_m \geq \mathcal{I}_{m+1}$ for all $m = 1, \ldots, M-1$. Monotonicity is expressed that for all layer $m = 1, \ldots, M$ the importance $\mathcal{I}_m$ is non–negative, i.e., $\mathcal{I}_m \geq 0$.

3.6.2 General Framework for Transmitting Pre-Encoded Data

Progressive coding allows easy integration of video data into packet networks with varying QoS. However, also for non-progressive media, a formalized description of packetized media is quite desirable for different reasons. We introduce a formalized description of the encoding, transmission and decoding process of a specific source following the definitions in [CM06]. However, some extensions and modifications are necessary to address all our considered use cases. Despite we will focus on video transmission in the following, the presented concepts are applicable to any multimedia data. The formalized description represents an extension of the abstraction for packetized progressively coded sources by additionally introducing timing, partial dependencies, and different encoded versions of one and the same source signal.

Such formalized descriptions of the encoded multimedia are required for efficient media streaming algorithms being able to make good decisions during the transmission process [CM06]. The description relies on the concept of data units, directed acyclic graphs and dependencies, quality versions, loss concealment, timing models, and the definition of the importance of data units.

Data Units

Following the presentation in chapter 2, *source* units s_n, $n = 1, \ldots, N$, (i.e., video frames) are encoded and mapped one-to-one to *data units* $\mathcal{P}_n$ (i.e., packets). Any advanced packetization modes, such as flexible macroblock ordering, slice structured coding, or packet interleaving schemes could be considered according to [CM06] as any source structure can be represented as an directed acyclic graph. However, for the moment we restrict ourselves to video where each video frame is transported in a separate packet. To specify decoding dependencies among data units, we write $n' \prec n$ if $\mathcal{P}_{n'}$ is required to decode $\mathcal{P}_n$. Furthermore, each source unit (and hence each data unit) has assigned a DTS $\tau_{\text{DTS},n}$ representing the latest time instant the data unit $\mathcal{P}_n$ must be decoded to be useful. The decoding time is relative to $\tau_{\text{DTS},1}$, which is assumed to be 0 without loss of generality. Data unit indices are ordered with increasing DTS $\tau_{\text{DTS},n}$.

3.6.3 Dependencies and Versions

According to [CM06], regardless of how many media objects there are in a multimedia presentation, and regardless of what algorithms are used for encoding and packetizing those media objects, the result is a set of data units for the presentation which can be represented as a directed acyclic graph. Each node of the graph corresponds to a data unit, and each edge of the graph directed from data unit n' to data unit n corresponds to a dependence of data unit n on data unit n'. That is to say, in order for data unit n to be decoded, data unit n' must also be decoded. This induces a partial order between data units, for which we write $n' \prec n$ if n is an ancestor of n (or equivalently if n is a descendant of n'). Thus, if a set of data units is received by the client, only those data units whose ancestors have all been also received can be decoded. The symbol $\prec$ is used to indicate a dependency of a data unit from another. Furthermore, the symbol $\prec\!\!\prec$ indicates a *direct* dependency. Finally, we introduce the operator $\preceq$. The expression $i \preceq n$ indicating a dependency $i \prec n$ or an equality $i = n$. The upper part of Figure 3.29 shows an example of possible frame dependencies and the corresponding graph.

As an extension to the framework in [CM06], we assume that for each source unit s_n several versions $m = 1, \ldots, M$ can be generated. The versions are represented by individual data units $\mathcal{P}_{n,m}$. Versions represent the encoding of the same video frame at different quality-rate levels. The reconstructed version of each source unit is denoted as $\hat{s}_{n,m}$. We restrict ourselves in the

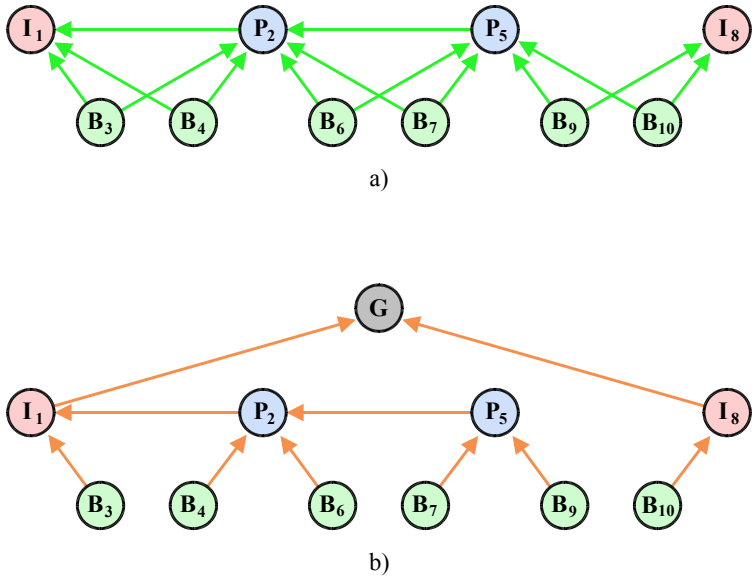


Figure 3.29: Frame dependencies and concealment graph.

following to the practical case where the spatial and temporal resolution as well as the graph for each version is of identical structure. Again, generalization to different structures for each version is straightforward. Furthermore, switching data units are integrated in each version. Switching data units may be realized by IDR-frames or by the use of SP pictures.

When transmitting a pre-encoded stream to a client, a server may select an appropriate version vector $\mathbf{m} = \{m_n\}_{n=1}^{N}$, with m_n the version chosen for each s_n. Hence, with this definition any arbitrary stream-switching strategy is possible, since different versions may be transmitted for each successive data unit. However, for our strategy we apply restrictions on version vector elements to avoid the problem of reference frame mismatches: Since switching is only allowed at I- or SP-picture positions, versions can only change at these positions as well.

Loss Concealment

Assume now that we operate in an environment where not necessarily all data units are received at the media decoder, either as data units are lost or the media server decides not transmit them or the data units are dropped at the receiver due to timing or buffer problems. In this case, concealment has to be applied for any representation of a missing data unit. In the remainder we apply the common ”freeze-picture” concealment, i.e., missing data units are represented by the timely nearest available source unit. In addition, only direct or indirect ancestors can be used concealing source units. Note, that the Presentation Time Stamp (PTS) of the concealing source unit can be in the future for example for B-frames in video. The index of the first candidate to conceal source unit s_n is denoted by the concealment index $c\,(n)$ and is defined as:

$$c\,(n) \triangleq \begin{cases} \{i \in \{1,\ldots,N\} | (i \prec n) \wedge \arg(\min_i(\tau_{\text{PTS},n} - \tau_{\text{PTS},i})\} & \text{if } i \text{ exists} \\ 0 & \text{otherwise} \end{cases}. \tag{3.34}$$

If there is no preceding source unit, e.g., for IDR-pictures, we assume that the lost source unit is concealed with a standard representation, e.g., a grey image (denoted as $c\,(n) = 0$).

In case of consecutive data unit loss, concealment is applied recursively. Assume that $c\,(n) = i$. If data unit $\mathcal{P}_i$ is also lost, the algorithm uses source unit s_j to conceal s_i, i.e., $c\,(i) = j$. To avoid

any inconvenient recursive notation we simply use $j \vdash n$ to express the fact that source unit s_n is eventually concealed with unit s_j. The resulting concealment dependencies can also be expressed by a directed graph. The lower part of Figure 3.29 shows an concealment graph corresponding to the frame dependencies in Figure 3.29a).

Quality

Based on these previous definitions, we can define a quality measure $\mathcal{Q}(s, \hat{s})$ reflecting the rewards/costs when representing s by $\hat{s}$. The definitions generalize the metrics introduced in section 2.1.2. To be more specific, let us define the reconstruction quality for this source unit

$$\mathcal{Q}_{m,n} \triangleq q(s_n, \mathcal{Q}^{(m)}(s_n)) \tag{3.35}$$

where $q(s, \hat{s})$ measures the rewards/costs when representing s by $\hat{s}$. The quality $q(.)$ might be the PSNR, or the MSE or any other distortion measure. The total average quality up to source unit n for a sequence of size N_s is therefore defined as

$$\overline{\mathcal{Q}}_{m,n}(N_s) \triangleq \frac{1}{N_s} \sum_{i=1}^{n} \mathcal{Q}_{m,n}, \tag{3.36}$$

and the total average quality is defined as $\overline{\mathcal{Q}}_m \triangleq \overline{\mathcal{Q}}_{m,N}(N)$.

Furthermore, these definitions also allow to define the concealment quality $Q\left(s_{n,m}, \hat{s}_i\right)$, if source unit n is represented with source unit i, $c\left(n\right)$ as

$$Q\left(s_{n,m}, \hat{s}_i\right) \triangleq q(s_n, \mathcal{Q}^{(m)}(s_i)). \tag{3.37}$$

Importance Definition

To allow making good decisions at the transmitter, it is essential that some decision making entity can quantify the importance of a data unit for the overall reception quality. Making use of the introduced definitions and by the abstraction of the encoding, transmission, and decoding process the so-called *importance* of each data unit $\mathcal{P}_{n,v}$ may be defined as the amount by which the quality at the receiver increases if the data unit is correctly decoded. The importance is defined as

$$\mathcal{I}_{n,v} \triangleq \frac{1}{N} \left(Q\left(s_n, \hat{s}_{n,v}\right) - Q\left(s_n, \hat{s}_{c(n),v}\right) + \sum_{\substack{i=n+1 \\ n \vdash i}}^{N} \left[Q\left(s_i, \hat{s}_{n,v}\right) - Q\left(s_i, \hat{s}_{c(n),v}\right) \right] \right). \tag{3.38}$$

The importance definition takes into consideration the quality of data unit $\mathcal{P}_{n,v}$, the chosen concealment strategy, as well as the dependency and concealment graph. In other words, the importance quantifies the improvement in quality, if the source unit contained in $\mathcal{P}_{n,v}$ is displayed instead of the concealment source unit $s_{c(n),m}$ for this unit, as well as for all other source units for which s_n is eventually used for concealment. Note that this definition is quite similar to the importance definition for different layer in (3.29), but in the earlier case the dependency was inherently given by the layer structuring.

Received and Expected Quality

The end–to–end performance of a media transmission system strongly depends on the versions chosen (expressed by the version vector $\mathbf{m}$), and the amount and importance of packets not available at the decoder. To be more specific, we briefly recap the channel definitions: The observed channel behavior for data unit $\mathcal{P}_{n,v}$ at a receiver is defined as as $\mathcal{C}_n \triangleq \mathbf{1}\{\text{data unit } \mathcal{P}_{n,v} \text{ available}\}$. Hence, combination of a certain observed channel sequence $\boldsymbol{\mathcal{C}} = \{\mathcal{C}_1, \ldots, \mathcal{C}_N\}$, with (3.38) and the concealment strategy as introduced above yields the following expression for the (actual) received quality:

$$\mathcal{Q}(\boldsymbol{\mathcal{C}}, \mathbf{m}) \quad = \quad \mathcal{Q}_0 + \sum_{n=1}^{N} \mathcal{I}_{n,m_n} \mathcal{C}_n \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \mathcal{C}_m. \tag{3.39}$$

Here, $\mathcal{Q}_0 \triangleq \frac{1}{N} \sum_{n=1}^{N} Q\left(s_n, \hat{s}_0\right)$ denotes the minimum quality, if instead of the original sequence all pictures are presented as grey. This is obviously only a hypothetical case, but it is necessary to have a comprehensive framework. For a proof on the derivation of (3.39), we refer to appendix A.1. Note that in this proof as well as in the further description in this section we drop the version index m for ease of exposition. In summary, in order to benefit from data unit $\mathcal{P}_n$, it is necessary that all preceding data units $\mathcal{P}_m$ are also available at the receiver.

Within this framework, the importance of each data unit and version is quite easily computed during the encoding process. As a consequence, (3.39) significantly simplifies the simulation of video transmission systems, as the achievable quality at the simulated media clients can be determined via linear combination of the channel vector and the importance of the selected versions of each data unit. Any decoding of erroneous video streams is thus not necessary anymore if a real decoder would make use of the same error concealment which may lead to significantly more efficient simulation environments.

However, more important is the practical importance of (3.39) for system optimization: the importance can be used advantageously at the transmitter for simple computation of the expected quality (at the receiver). Due to the random channel behavior sequence $\hat{\boldsymbol{\mathcal{C}}}$ also the received quality, $\mathcal{Q}(\hat{\boldsymbol{\mathcal{C}}})$, is a random variable. For certain channel realizations we obtain a good quality, whereas for others the received quality is much worse.

In many cases one is interested in a single measure to compare the different transmission strategies. The most obvious and suitable measure is the expected quality $E\left\{\mathcal{Q}(\hat{\boldsymbol{\mathcal{C}}})\right\}$. The following equation provides a definition of the expected received quality as well as a simplified method to derive it:

$$\begin{aligned} E\left\{\mathcal{Q}(\hat{\boldsymbol{\mathcal{C}}})\right\} \quad &\triangleq \quad \sum_{\boldsymbol{\mathcal{C}} \in \{0,1\}^N} \mathcal{Q}(\boldsymbol{\mathcal{C}}) \Pr\{\hat{\boldsymbol{\mathcal{C}}} = \boldsymbol{\mathcal{C}}\} \\ &= \quad \mathcal{Q}_0 + \sum_{n=1}^{N} \mathcal{I}_n \Pr\{\hat{\mathcal{C}}_n = 1 \,\Big|\, \forall_{k \prec n} \hat{\mathcal{C}}_k = 1\} \\ &\qquad \cdot \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \Pr\{\hat{\mathcal{C}}_m = 1 \,\Big|\, \forall_{k \prec m} \hat{\mathcal{C}}_k = 1\} \\ &= \quad \mathcal{Q}_0 + \sum_{n=1}^{N} \mathcal{I}_n \Pr\{\hat{\mathcal{C}}_n = 1 \wedge \forall_{k \prec n} \hat{\mathcal{C}}_k = 1\} \end{aligned} \tag{3.40}$$

Note that the expectation in this case is only over the channel statistics $\hat{\mathcal{C}}$. For a proof of the various equalities (3.40), we refer to appendix A.2.

3.6.4 Summary: Abstract Representation of Pre-encoded Video Data

With these preliminaries a beneficial abstraction of streamed media data beyond the concepts introduced in [CM06] is available. For channels which exhibit data unit losses, it is sufficient to know the number of encoded source versions V, the initial quality $\mathcal{Q}_0$, and the following metrics for each data unit $n = 1, \ldots, N$ and each version $m = 1, \ldots, M$:

- the importance $\mathcal{I}_{n,m}$,
- the data unit size $R_{n,m}$ in bytes,
- the decoding time stamp $\tau_{\text{DTS},n}$, and
- the dependencies expressed by the index of the directly preceding data unit(s) of $\mathcal{P}_n$.

Furthermore, in case that SP-pictures are used for each version m, the data unit size $R_{n,m\to m'}$ of the SSP-picture when switching to version m', and the SI-picture size are required [Wal04, SLW06]. This abstract description can be used to efficiently simulate video streaming over lossy channels by the application of (3.39) and is used for generating results in chapter 6. The application for the optimization of transmission schedules by applying (3.40) is shown 9 as well in [LJSB04].

3.7 Summary

In this chapter source and video coding tools have been introduced. The following observations and findings are of major importance:

- For the easy integration of source coders in networks, it is desirable that source coders are rate-adaptive, i.e., can support the generation of different quality-rate levels. Even preferably is the availability of progressive source coders that generate a bitstream containing several or many quality-levels.
- For simple κ-dimensional random variables many *good* progressive source coders exist, i.e., the p-th power distortion decays exponentially with the rate R.
- For video codecs to be applicable in network environments, both compression efficiency and network integration are of major importance. This chapter introduced the major H.264/AVC concepts. Several network integration and error resilience features have been developed in the course of this work and have been adopted as part of the H.264/AVC standard.
- The specifications of video codecs in 3GPP is quite compact, in general only an external specification with a profile/level compliance point a referenced. This leaves significant flexibility in efficient deployments: Encoder settings, e.g. bit-rate control, error resilience, or motion estimation processes, are generally completely left open in standardization work and are therefore investigated in this work.

- Bit-rate adaptivity is an important feature for the use of video codecs in wireless environments. For small scale variations, playout buffering or temporal scalability can be used. However, to adapt to larger scale variations, bitstream switching is a preferable solution. In particular, the Switching–Predictive (SP)-picture concept of H.264/AVC provides an efficient realization for bitstream switching.

- We have implemented an optimized Switching–Predictive (SP)-picture concept specifically for bitstream switching. Optimizations for encoding the primary and especially the secondary representations have been proposed and verified.

- Scalable video coding is even more suitable for easy network integration, but the MCP process in hybrid video codecs does not allow scalability without sacrificing compression efficiency. Scalable/progressive video codecs are in general not *good*. However, the designed and developed PTVC, a scalable extension of an interim version of the H.264 test model, provides a breakthrough in progressive video coding. By the use of drift-compensating means progressively coded video streams can be generated for easy network integration.

- The rate–distortion performance of video codecs is not only determined by the standard features, but also by the encoder implementation: for example by the use of VBR encoded video instead of CBR encoded video, significant benefits can be achieved. These features have been integrated in the H.264 test model and the PTVC. The performance has been shown by extensive simulations.

- For the appropriate use of PTVC, it is quite important that the feedback rate of the MCP process operates closely to the actually transmission rate of the video. In case there is a mismatch between the transmission rate and the feedback rate, the compression efficiency is in general significantly degraded, or the drift is unacceptable. The introduction of I-frames for drift compensation can overcome the drift problem when transmitting over channels with unknown transmission bit-rate.

- For the purpose of rate–distortion optimization in network environments, a formalized description of pre-encoded video is desirable. Therefore, the concept of *importance* of data units has been developed in this chapter which describes the importance of a data unit for the reconstruction quality of a sequence. The framework can be applied to progressively coded sources, scalable coded sources, and sources which can be represented by dependency graphs. Extensions to multiple versions have been developed. Furthermore, the application of this framework permits complexity-reduced simulation of packet-lossy video transmission.

4

Reliable Transmission Toolbox

4.1 Error Protection Preliminaries

4.1.1 Theoretical Background

In the following, we consider the block–fading AWGN channel as introduced in subsection 2.4.3 to present conceptual background on error protection schemes for mobile communication systems. The parameters describing the channel are listed in Table 2.3. We restrict ourselves to binary–input channels, but we also present bounds for the Gaussian input channel just for comparison purposes. The presentation basically follows the ideas presented in [SOW94], [BPS98], [Kno97], and [CT01].

To transmit messages across a channel to the receiver, a channel encoder maps messages on a sequence of channel symbols $\mathbf{x}$ of length n. Codes suitable for transmission in combination with BPSK modulation are almost exclusively constructed over the binary field $\mathbb{F}_2$[1]. The elements of $\mathbb{F}_2$ usually are represented by 0 and 1, but we commonly use the input symbols ± 1 to emphasize BPSK modulation as introduced in subsection 2.4.3. The messages to be transmitted are considered to be represented by a binary information word $\mathbf{u} \triangleq \{u_1, \ldots, u_k\}$ of length k with $u_t \in \{+1, -1\}$. The encoding of $\mathbf{u}$ is defined as the mapping of this sequence on a channel word $\mathbf{x} \in \mathbb{C} \subset \mathcal{S}_x^n$ of length n where $\mathbb{C}$ is the code book and $\mathcal{S}_x$ is the transmission alphabet set $\mathbb{F}_2$ with elements ± 1 in our case. For convenience, we write $\mathbf{x}(\mathbf{u})$ to indicate the code word assigned to the information sequence $\mathbf{u}$ and $\mathbf{u}(\mathbf{x})$ to indicate the information sequence assigned to code word $\mathbf{x}$. The rate of the code is defined as

$$r \triangleq \frac{k}{n} \tag{4.1}$$

and is measured in bits per channel use[2]. The channel symbols $\mathbf{x}$ of length n are transmitted over a channel $\mathcal{C}_H$ with symbol transition probability $f_{Y|X,H}\,(y|x,h)$ depending on an auxiliary random

[1] Also for other QAM and PSK modulation schemes binary codes are commonly used in practical systems. By the use of Bit–Interleaved Coded Modulation (BICM) or multilevel coding binary codes can at least practically provide the same performance as non–binary codes [CTB96].

[2] With the one–dimensional signal model, the unit "bits per channel use" is identical to the unit "bits per dimension". With the BPSK signal model in mind, we will in the sequel assume the code rate r to be dimensionless.

variable h which is viewed as a channel state in the following. In case of the block–fading AWGN channel, the channel state corresponds to the SNR. A decoder at the receiver decides for example based on a maximum-likelihood rule on the message $\hat{\mathbf{x}}$ with the detected word being from the set of transmitted word, i.e., $\hat{x} \in \mathcal{S}_{\hat{x}} = \mathcal{S}_x$. An inverse mapping $\mathbf{u}(\hat{\mathbf{x}})$ of $\hat{\mathbf{x}}$ to the information symbol space also allows to obtain the decoded information word $\hat{\mathbf{u}} \in \mathcal{S}_{\hat{u}}^k = \mathcal{S}_u^k$.

We consider in the following the case without any channel feedback in a sense that the code rate r is fixed and no power control is applied, i.e., $\gamma_f = 1$. Shannon's channel coding theorem [Sha48] establishes that for any channel code rate r less than the capacity of a channel with channel state h and the channel state h known to the transmitter, there exist channel codes whose probability of decoding error goes to zero as the block length grows to infinity. We concentrate on the message error probability in the following rather than on the symbol or bit error probability as most of the investigated applications are packet–based. Assuming maximum likelihood decoding, Gallager [Gal68] showed that the message error probability $P_B(h,r) = \Pr\left\{\hat{\mathbf{X}} \neq \mathbf{X}|H = h\right\}$ is a function of the code rate r and goes to zero exponentially fast as a function of the channel code block length n and is bounded by

$$P_B(r,h) \leq 2^{-n\underline{E}_{r,\text{gallager}}(r,h)} \tag{4.2}$$

with $\underline{E}_{r,\text{gallager}}(r,h)$ being the error exponent defined as

$$\underline{E}_{r,\text{gallager}}(r,h) \triangleq \max_{0\leq\rho\leq 1}[E_0(\rho,h) - \rho r] \tag{4.3}$$

and $E_0(\rho,h)$ the Gallager function which is defined for uniformly binary input symbols x with channel state h as

$$E_0(\rho,h) \triangleq \log_2\left\{\int_y \left[\frac{1}{2}f_{Y|X,H}\left(y|x=+1,h\right)^{\frac{1}{1+\rho}} + \frac{1}{2}f_{Y|X,H}\left(y|x=-1,h\right)^{\frac{1}{1+\rho}}\right]^{1+\rho} dy\right\}. \tag{4.4}$$

The Gallager function and consequently the error exponent can only be solved numerically for binary input channels, but is known to be monotonically decreasing and convex in r. Two points on the error exponent function are of major interest. The rate r which solves $\underline{E}_{r,\text{gallager}}(r,h) = 0$ is equivalent to the mutual information

$$R_I(h) \triangleq I(Y;X|H=h) = \left.\frac{\partial E_0(\rho,h)}{\partial\rho}\right|_{\rho=0}$$

between the input X and the output Y conditioned on a particular channel state h. This intuitively proves the forward part of the channel coding theorem [Sha48], namely if the code rate is chosen such that $r < R_I(h)$, according to (4.2) the error probability can be made arbitrarily small for sufficiently long code words n. The second point of interest, the *cutoff rate* is determined for uniformly binary input symbols x with channel state h as

$$R_0(h) \triangleq E_0(\rho=1,h) = 1 - \log_2\left(1+e^{-h}\right). \tag{4.5}$$

The cutoff rate has less theoretical justification than the mutual information. Still, this value was for a long time assumed to be the highest rate at which practical coding schemes can operate [VO79]. This has obviously changed with the invention of Turbo codes [BGT93], but for some codes and decoding schemes this measure is very useful as we will see later.

Figure 4.1 shows the capacity[3] R_C, the capacity for BPSK referred to as mutual information R_I and the cutoff rate R_0 for BPSK over the SNR h. The maximum difference at code rate $r = 0.5$ between the cutoff rate and the mutual information is about 2.5 dB.

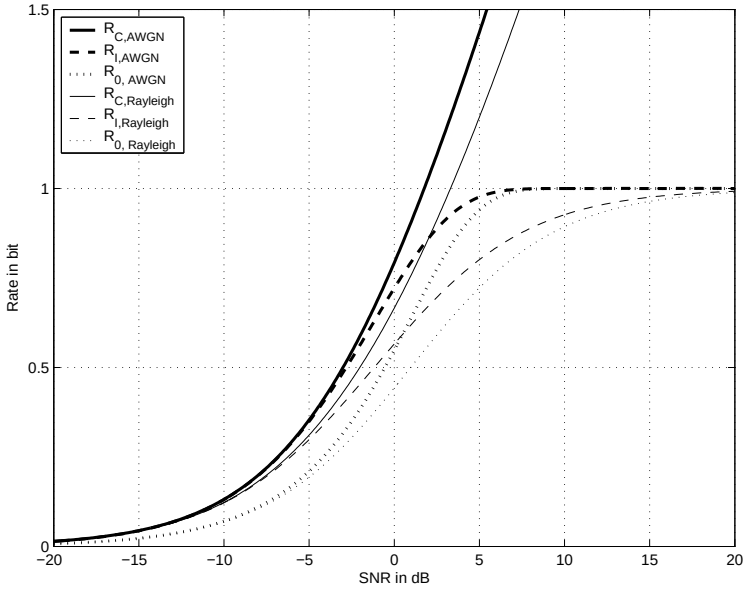


Figure 4.1: Capacity and Cutoff Rate over h for AWGN channel.

Note that the bound in (4.2) would result in error probabilities greater than 1 for $r > R_I(h)$ it is more appropriate to express the upper bound on the error probability in (4.2) as

$$P_B(h,r) \leq 2^{-n\underline{E}_{r,\text{gallager}}(r,h)} \mathbf{1}\left\{R_I(h) \geq r\right\} + \mathbf{1}\left\{R_I(h) \leq r\right\} \tag{4.6}$$

Gallager [Gal68] also showed that a lower bound on the error probability $P_B(h,r)$ tends exponentially to one with increasing n for all codes with code rate $R_I\,(h) \leq r$. With these observations the error probability can quite well be approximated as

$$P_B(h,r) \approx \mathbf{1}\left\{R_I(h) \leq r\right\}. \tag{4.7}$$

where the $\approx$ gets more and more accurate as $n \to \infty$.

In packet–radio systems it is often desired to perfectly detect transmission errors to avoid error propagation through layers which cannot be accomplished by the use of maximum–likelihood decoding unless an outer error detection code, e.g., Cyclic Redundancy Check (CRC) code, is applied. However, if this is undesired because of decreased coding efficiency, other decoding methods than maximum–likelihood might be applied with built–in error detection capabilities. This includes, but is not limited to bounded–distance decoding [LJ83] or sequential decoding algorithms as we will see later. In this case, the alphabet for the decoded information is extended by the erasure element ε indicating a decoding failure, i.e., $\mathcal{S}_{\hat{u}}^k = \mathcal{S}_u^k \cup \varepsilon$. In [CT01] the authors show that in the limit that $n \to \infty$ the probability of undetected errors can be made arbitrarily small.

[3]With the term capacity we refer in the following to the Shannon capacity which maximizes the mutual information by selecting an input distribution which satisfies the power constraint $\mathcal{E}\{x^2\} \leq 1$. For the AWGN or fading channel with CSI h known to the receiver this is achieved by the Gaussian input distribution.

The performance of codes and decoding schemes can be summarized, in a sense that for $n \to \infty$ there exist binary input codes which fulfill

$$\begin{aligned} \Pr\left(\text{error} \,|\, r < R_I(h)\right) &= 0, \\ \Pr\left(\text{error} \,|\, r \geq R_I(h)\right) &= 1, \\ \Pr\left(\text{undetected error}\right) &= 0. \end{aligned} \tag{4.8}$$

We refer to codes $\mathbb{C}$ fulfilling this property as *asymptotically* optimal.

An important observation for the performance of coding in mobile communications comes from the fact that both measures, the mutual information $R_I(H)$ and the cutoff rate $R_0(H)$, are random variables as the they depend on the random variable H. This means, that for a fixed code rate r that the mutual information[4] $R_I(H)$ is below this code rate. Therefore, the main source for the error probability in mobile communications is not due to the limited block–length according to (4.2), but by the random nature of the channel state H, i.e., the probability that the mutual information is below the chosen code rate r. With the outage probability

$$P_{\text{out}}(r, H) \triangleq \Pr\{R_I(H) < r\} \tag{4.9}$$

the average message error probability $\bar{P}_e(r, H)$ for a certain code with code rate r results in

$$\bar{P}_e(r, H) = \mathbb{E}_H\left\{P_B(r, H)\right\} \approx P_{\text{out}}(r, H), \tag{4.10}$$

where the $\approx$ is equivalently interpreted as in (4.7). However, it is also important that the code has also to be good for any state h such that $R_I(h) > r$. The existence of such codes was also shown in [CT01].

Let us now consider a system in which the channel code word $\mathbf{x}$ is spread not just over a single radio slot, but over F slots, each of length N_v. The code word size results in $n = N_v F$. Assuming the block–fading AWGN channel with statistically independent channel states H_f for the slots f, the total mutual information is the average over F radio slots, i.e.,

$$R_I(\mathbf{H}_F) \triangleq \frac{1}{F}\sum_{f=1}^{F} R_I(H_f). \tag{4.11}$$

Note that the mutual information $R_I(\mathbf{H}_F)$ still has random behavior. However, if the set of channel realizations is an ergodic sequence then in the limit $F \to \infty$, the mutual information is deterministic and equivalent to the conditional mutual information, i.e.,

$$R_I(\mathbf{H}) \triangleq \lim_{F\to\infty} R_I(\mathbf{H}_F) = \mathbb{E}_{\mathbf{H}}\left\{R_I(H)\right\} = I(Y; X|H). \tag{4.12}$$

The result in (4.12) is equivalent to the capacity of a perfectly interleaved fading channel as initially introduced in [Eri70]. The error and outage probabilities compute equivalently to (4.10) as

$$\bar{P}_e\left(r, \mathbf{H}_F\right) = \mathbb{E}_H\left\{P_B\left(r, \mathbf{H}_F\right)\right\} \approx P_{\text{out}}\left(r, \mathbf{H}_F\right) = \Pr\{R_I\left(\mathbf{H}_F\right) < r\}. \tag{4.13}$$

Similar arguments are made for the cutoff rate when replacing R_I by R_0 in (4.7), (4.11), and (4.12). The capacity for Gaussian input, the mutual information $R_I(\mathbf{H})$, and the cutoff rate $R_0(\mathbf{H})$ for iid Rayleigh fading channel is also shown in Figure 4.1.

[4] We continue with the discussion using the mutual information, but most of the arguments also hold for the cutoff rate.

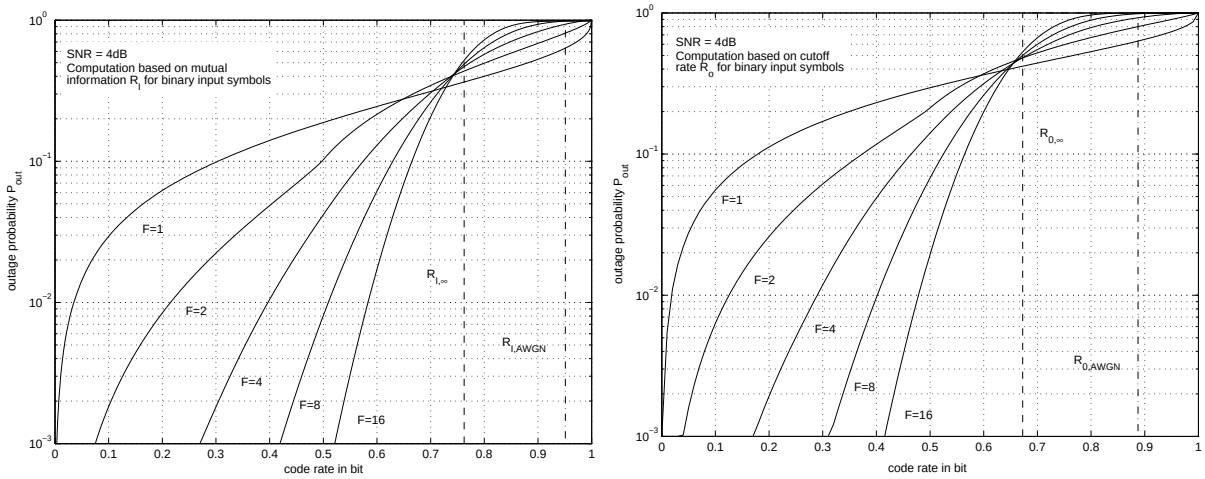


Figure 4.2: Outage Probability P_{out} over code rate r for average SNR $\mathbb{E}\{H\} = 4$ dB, different number of slots F, for $F \to \infty$ and for AWGN channel based on the mutual information (left–hand side) and the cutoff rate (right–hand side).

Similar to the single rate–distortion pairs and operational rate–distortion curves in source coding, the bounds on the performance of channel coding for a specific channel can be summarized by the pair of channel code rate r and error probability $\bar{P}_e(r, H)$. Figure 4.2 shows the outage probability P_{out} over the code rate r for average SNR $\mathbb{E}\{H\} = 4$ dB for different number of slots F as well as for $F \to \infty$ based on the mutual information (left–hand side) and the cutoff rate (right–hand side). The effects of channel coding with code rate r and diversity F based on the cutoff rate bounds over different numbers of radio slots are obvious: For the channel with only a single slot $F = 1$ and channel coding rate $r = 0.5$, the outage probability P_{out} is about 0.3, whereas for increased number of radio slots $F = 4$ and $F = 16$, the variance of the channel can be reduced and the outage probability decreases. However, this usually comes at the expense of additional delay and has therefore to be considered carefully for delay–critical applications. Current systems typically apply for example $F = 4$ as a compromise between diversity and delay resulting in a still significantly varying outage probability over the applied channel coding rate r. Since the channel coding rate r is directly proportional to the maximum supported application bit-rate, a trade–off between outage probability and bit-rate is necessary. This will be discussed in detail in chapter 8.

4.1.2 Rate-Compatible Punctured Channel Coding

In subsection 3.1.2 the benefits of a single source coder operating at different rates has been discussed. Similarly, for practical systems it is desired to use a single channel coding scheme which can operate at several or many rates without sacrificing significant performance losses in terms of error probability $\bar{P}_e(r, H)$. The channel coder should allow to adapt to varying conditions, for example to the actual channel state, or to some specific source rate. In [CT01], Caire and Tunetti present a framework on coding, decoding, and error detection for a specific scenario, namely to assess the throughput of ARQ protocols over the Gaussian collision channel. However, the framework presented is much more general and applies to scenarios investigated in this work. The authors also mention the analogy of the Gaussian collision channel and the block–fading AWGN channel.

A large variety of rates can be supported by rate–compatible puncturing. The binary information

word $\mathbf{u}$ of length k is encoded by mapping this sequence on a channel word $\mathbf{x} \in \mathbb{C}$ of length n; $\mathbb{C}$ is the code book. Let $\mathbb{C}_w$ for $w = 1, \ldots, N_w$ denote the punctured code of length n_w obtained by deleting the last $n - n_w$ symbols from $\mathbf{x}$. The code rate of the sub–code $\mathbb{C}_w$ is defined as

$$r_w \triangleq \frac{k}{n_w}.$$

Let us further assume without loss of generality that the individual codes are ordered such that $\forall w = 1, \ldots, N_w$ the code $\mathbb{C}_w$ is constructed such that it contains only code words which are binary extensions of the code words in $\mathbb{C}_w$ resulting in ordered code rates $r_w > r_{w-1}$.

Assume that transmitter and receiver have agreed to operate on channel code $\mathbb{C}_w$. The transmitter transmits the first n_w symbols from $\mathbf{x}$ denoted as $\mathbf{x}_{(0:n_w]}$. The decoder receives $\mathbf{y}_{(0:n_w]}$ and together with channel state information it generates the channel LLR for each symbol to obtain $\boldsymbol{\psi}_{(1:n_w]}$. As the decoder operates on code $\mathbb{C}$ rather than $\mathbb{C}_w$ expecting channel LLRs for n positions the receiver must fill up $\boldsymbol{\psi}_{(n_w+1:n]}$ with dummy symbols independent of the received signal. With the use of the LLR ψ as the generic decoder input and as the receiver has no information from the channel about these dummy positions, it is optimal to set these unknown positions to zero, i.e., $\boldsymbol{\psi}_{(n_w+1:n]} = \mathbf{0}$.

Bounds on the performance of punctured codes have been presented in [CT01]. Lemma 1, 2, and 3 confirm that there exist rate–compatible punctured codes being as good as non–punctured codes as long as the difference between the length between any two subsequent sub–codes $n_{w+1} - n_w$ is sufficiently large. Under this regime it shown that for all sub–codes $\mathbb{C}_w$, the conditions in (4.8) can be fulfilled. Punctured codes fulfilling this property are referred to *asymptotically optimal rate–compatible punctured codes*.

The interesting consequence from these results for the transmission over a block–fading channel has also been discussed in [CT01]. The existence of these asymptotically optimal rate–compatible punctured codes together with (4.11) shows, that the error probability as well as outage probability in (4.13) is not just an average error probability over an ensemble of individual codes $\mathbb{C}$ with different rates, where each code fulfills (4.8), but it can be approached by the use of an asymptotically optimal rate–compatible punctured code. In this case, sub–codes are designed such that they have integer multiple length of the transmission slot of the block–fading channel, i.e., $n_w = wN_v$. Note that specific symbol interleaving is not necessary but is integrated in the design of the code.

4.1.3 Unequal Error Protection

The single–user mobile channel with varying channel states modelled by the block–fading AWGN channel as introduced subsection 2.4.3 has some information–theoretic equivalence to the broadcast channel [Cov72] especially in case of delay restrictions. Whereas in the broadcast scenario the same information is transmitted to several users with different receiving conditions within the same bandwidth and a common power, in the block–fading AWGN the different channel states result from fading. Obviously, the single–user fading channel allows more methods to compensate the fading by the use of feedback and interleaving over F channel states. However, without the availability of feedback and $F < \infty$, the different channel states can be viewed as different receiving conditions.

For broadcast or multi–user channels in general, the channel is not described by a single measure such as the capacity, but by a capacity region describing the supportable rate combinations for all users with certain receiving conditions specified by the SNR. The capacity region for general broadcast channels is still unknown, but it is known that for almost all cases the simultaneous

transmission of users in frequency and time exceeds time–sharing[5] and that for special cases superposition approaches with stripping receivers are optimal [CT91]. Still, time–sharing is most attractive due to its simplicity and its almost optimal performance compared to optimal, but complex superposition approaches. Therefore, we will exclusively use time–sharing in the following.

Let us define M different "users", referred to as *levels* in the following. Each level m gets assigned a fraction ω_m of code word symbols $n_m = \omega_m n$. In our case of time–sharing, the sum of code word fractions must not exceed one, i.e., $\sum_{m=1}^{M} \omega_m \leq 1$. Then, each level m can apply its individual code rate r_m. With that, the number of binary symbols k_m transported by a certain level m is given as $k_m = r_m n_m$.

With the broadcast background, it might be decided that for example two levels should be supported, one which supports reception in $1 - P_{\text{out},1} = 50$ % of the time, and another one which supports reception in $1 - P_{\text{out},2} = 90$ % of the time. With a given channel state distribution, for example according to Figure 4.2, a certain number of available transmission slots F, and the channel coding rates asymptotically achievable are given as

$$\bigcup_{0 \leq \omega \leq 1} \left\{ r_1 \leq \omega R_I(h(P_{\text{out},1})) \quad \wedge \quad (r_2 \leq (1-\omega) R_I(h(P_{\text{out},2}))) \right\}, \tag{4.14}$$

where $h(P_{\text{out}})$ represents the minimum SNR necessary to support an outage probability below P_{out}, i.e., $h(P) \triangleq \min\{h : P \leq P_{\text{out}}(h)\}$. Again, the usage of the mutual information R_I in (4.14) is arbitrary and might be replaced by any other measure which describes the performance of a single–user code on this particular channel. If applied over multiple slots F, this scheme requires a specific interleaving, namely that the channel symbols for each level are distributed uniformly over all F slots proportionally to the fraction ω_m. A practical realization for block–fading channels and $\omega_m = 1/M$ is discussed in subsection 4.3.1.

Figure 4.3 shows the time–sharing rate region for two levels $M = 2$, average SNR $\mathbb{E}\{H\} = 4$ dB, and different number of slots F, based on the mutual information R_I. Different rate assignments for different transmission fractions ω are shown. Without going into detail, it is worth to mention that only for highly variant channels, i.e., $F \leq 4$ it seems to be interesting to use different protection for different levels. For example, for $F = 16$ and $\omega = 0.5$ the supported rate does not differ significantly meaning that with only sacrificing a small amount of bit-rate the outage probability is reduced significantly.

As the transmitter generally selects different code rates for each of the M different portions of the original data block, this approach is usually known under the acronym Unequal Error Protection (UEP). The availability of rate–compatible punctured codes[6] allows a huge flexibility in the selection of the rates at the transmitter, namely the number of levels M, as well as for each individual level m the number of code symbols n_m, or equivalently the fractions ω_m, and the individual number of information bits k_m, or equivalently the code rates r_m. Note that if the sum rate is to be maximized, in general UEP approaches do not yield any gains, but Equal Error Protection (EEP) is suggested to be used which is equivalent to a single level $M = 1$.

Unequal Error Protection for Delay–Limited Progressively Coded Sources Several researchers, e.g. [SZ98] and [CF99], have recognized the appropriateness of UEP when transmitting progressively coded sources over fading channels, usually in combination with a specific channel

[5] Time–sharing acts as a synonym for any orthogonal channel sharing, but it matches the defined channel model.

[6] Rate–compatibility is not necessary to support UEP. Still, we exclusively use rate–compatible puncturing in the following as the theoretical performance loss is zero, the loss in practical schemes negligible, and we have a unique code for different applications.

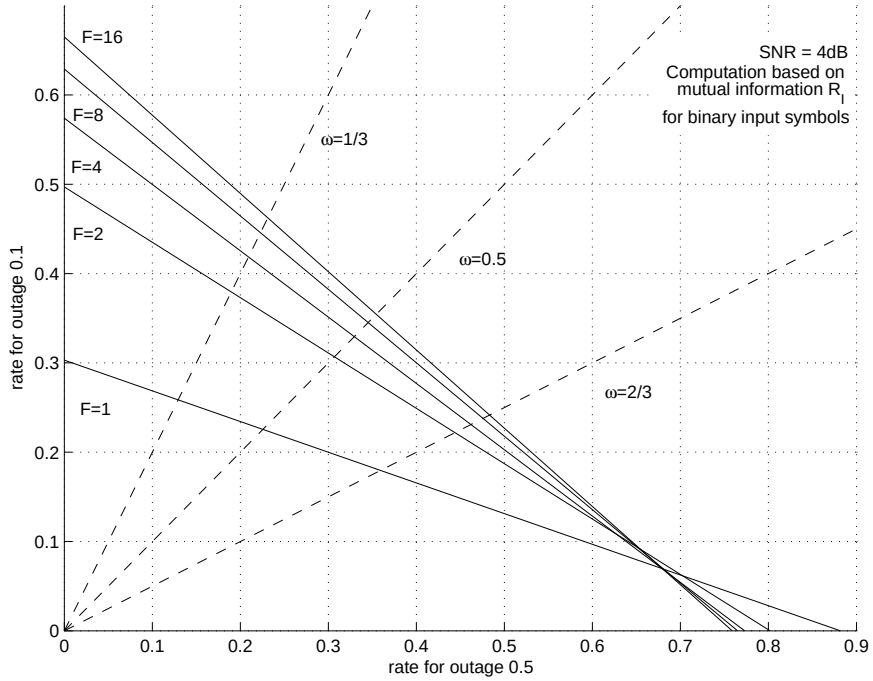


Figure 4.3: Rate region for two levels $M = 2$, SNR $\mathbb{E}\{H\} = 4$ dB, different number of slots F, based on the mutual information R_I and rate assignments for different transmission fractions ω.

coding method in mind. Replacing a specific channel code by asymptotically optimal channel code operating at capacity, mutual information, or cutoff rate allows to estimate the expected quality when transmitting a progressively coded source over a channel with unknown channel state H distributed according to some probability distribution P_H. The message is divided into M levels. Each level m, $m = 1, \ldots, M$ gets assigned k_m data symbols and n_m redundancy symbols. The channel symbols are distributed equally such that all layers observe the same channel state[7] h.

Assuming asymptotically optimally codes operating at the mutual information R_I, the entire layer m can be decoded as long as the experienced channel state h is such that the corresponding mutual information, $R_I(h)$, is larger equal than the code rate r_m assigned to the corresponding UEP level[8]. Therefore, at the decoder all layers which have sufficient redundancy can be reconstructed.

If we are interested in an expected quality for this progressively coded source as specified in (3.33) we need to obtain an appropriate expression for the probability that a certain layer m can contribute to the overall quality at the receiver, $P_m \triangleq \Pr\{\hat{\mathcal{C}}_m = 1 \wedge \forall_{i \prec m} \hat{\mathcal{C}}_i = 1\}$. Assuming a progressively coded source in a sense that after packetization the data in layer m can only be decoded if all data in the previous layers $1, \ldots, m-1$ are decoded, this probability results in

$$P_m \;=\; \min_{i=1,\ldots,m} (1 - P_{\text{out}}(H, r_i)) = \left(1 - P_{\text{out}}(H, \max_{i=1,\ldots,m} r_i)\right). \tag{4.15}$$

By combining (3.29) and (4.15) in (3.40) the expected received quality for a source with rate–quality function $\mathcal{Q}(R)$, a packetization of the source resulting in a vector $\mathbf{k} = \{k_1, \ldots, k_M\}$, and a

[7] We restrict ourselves to the notation with a single channel state for ease of exposition. However, the $F = 1$ case is easily generalized by replacing the channel state h by a sequence of channel states, $\mathbf{h}$.

[8] Although layer is assigned to source coding and level to channel coding, in the combination of layered coding and UEPwe use the terms ”level” and ”layer” synonymously.

code rate vector $\mathbf{r}$ computes as

$$\mathbb{E}\left\{\hat{\mathcal{Q}}(\mathbf{k},\mathbf{r})\right\} = \sum_{m=1}^{M} \left(\mathcal{Q}\left(K_m\right) - \mathcal{Q}\left(K_{m-1}\right)\right) \left(1 - P_{\text{out}}(H, \max_{i=1,\ldots,m} r_i)\right). \tag{4.16}$$

with K_m the accumulated bit-rate up to layer m as defined in (3.28). Appropriate selection of bit-rate allocations K_m and code rates r_m for each layer based on the expected quality in (4.16) is subject of the rate-distortion optimization algorithms in chapter 5.

4.1.4 Channel State Information and Power Control

Until now, we have completely ignored that the mobile systems under investigation usually provide feedback messages. On the physical layer, information is commonly exchanged to adapt the transmit power adapted to the current channel state. Fast power control and physical layer feedback in general has gained significant importance in 3G systems and the literature on this topic is extensive, on both theoretical and practical issues. We only restrict ourselves on a relatively simple, but meaningful example to exploit video transmission in this environment. In [GV97] and [CTB99], power allocation schemes for fading channels have been presented which exactly fit into the framework of the mobile channel model based on the block–fading AWGN channel. We will briefly summarize the main results for a specific scenario.

In contrast to the previous cases we now assume that the transmitter has perfect knowledge on the channel gain α_f and therefore of the channel state h_f for the upcoming transmission slot f. We distinguish two scenarios: In the constant–power transmission mode the transmission power for each slot is fixed, i.e., $\gamma_f = 1$, but the code rate r_f can be adapted to compensate the channel, i.e., $r_f = R_I(h_f)$. In this case the average rate supported by this channel is also given by the average mutual information according to (4.12). Therefore, the feedback and the channel knowledge cannot enhance the average mutual information of this system. However, in delay–limited systems this measure is less important, we are more interested on the performance if only a limited amount of channel state realizations are available, i.e., $F < \infty$. Whereas in case of no CSI, the transmission with a certain rate results in an outage, in case of the availability of the CSI, the channel coding rate is selected such that the transmission is reliable. This has the obvious advantage that the transmitter can assure with high probability that this information is correctly received at the receiver. Therefore, the outage probability P_{out} according to (4.13) has not the meaning of a data loss probability, but $1 - P_{\text{out}}$ represents the average cumulative distribution function (cdf) of the supported rate for a certain channel over F channel slots.

In the second scenario investigated in this work, we again assume that the transmitter is aware of the channel gain α_f for the next transmission slot, but only a long–term power constraint has to be fulfilled by $\mathbb{E}\{\gamma_f\} = 1$. In addition, we assume that the distribution p_α of the channel gain α is iid and that the transmitter has knowledge of the this distribution, or equivalently of the distribution of the SNR h. The maximum average rate with optimal power control is obtained by

$$R_I^{(\text{pc})}(H) = \max_{\gamma} \mathbb{E}_H\left\{R_I(H\gamma)\right\}, \tag{4.17}$$

whereby the maximization is over the power allocation function $\gamma = \gamma(H)$ such that $\mathbb{E}\{\gamma\} = 1$. In general, with $R_I'(h) \triangleq (\partial R_I(h))/(\partial h)$ an unconstrained solution for the power allocation function results in [9] $\gamma(h,\lambda) = [\{\gamma \in \mathbb{R} : R_I'(\gamma h) = \lambda/h\}]_+$, where λ is the solution to the constrained

[9] $[x]_+ \triangleq \max(x, 0)$

equation $\mathbb{E}_H\{\gamma(H,\lambda)\} = 1$. Again, the mutual information can be replaced by any other measure which describes the code performance. For the Shannon capacity R_C the power allocation results in the well–known water–filling solution $\gamma(h,\lambda) = [1/\lambda - 1/h]_+$ [CT91], for binary input symbols the power allocation function $\gamma(h,\lambda)$ cannot be not explicitly given. However, for binary input symbols and maximizing the cutoff rate, the optimal power allocation yields

$$\gamma(h,\lambda) = \left[\frac{1}{h}\log\left(\frac{h}{\lambda\log(2)} - 1\right)\right]_+ . \tag{4.18}$$

Figure 4.4 shows the capacity, the mutual information, and the cutoff rate for perfect channel state information at the transmitter and applying power control according to (4.17). Significant gains are obtained compared to the system without power control for the block–fading AWGN channel as the available power is only distributed to good channel realizations, one ”rides on the peaks”. Then, in case of a good channel realization higher data rate can be supported which increases the overall system performance. However, note that due to the resulting VBR channel, applications requiring CBR and/or relying on deadlines might suffer from this strategies. For progressive source transmission over such systems, outage can be completely avoided but the amount of transmitted information within the delay–budget may be small due to adverse channel conditions. The integration of real–time video transmission in a power-controlled environment is discussed in more detail in chapter 8.

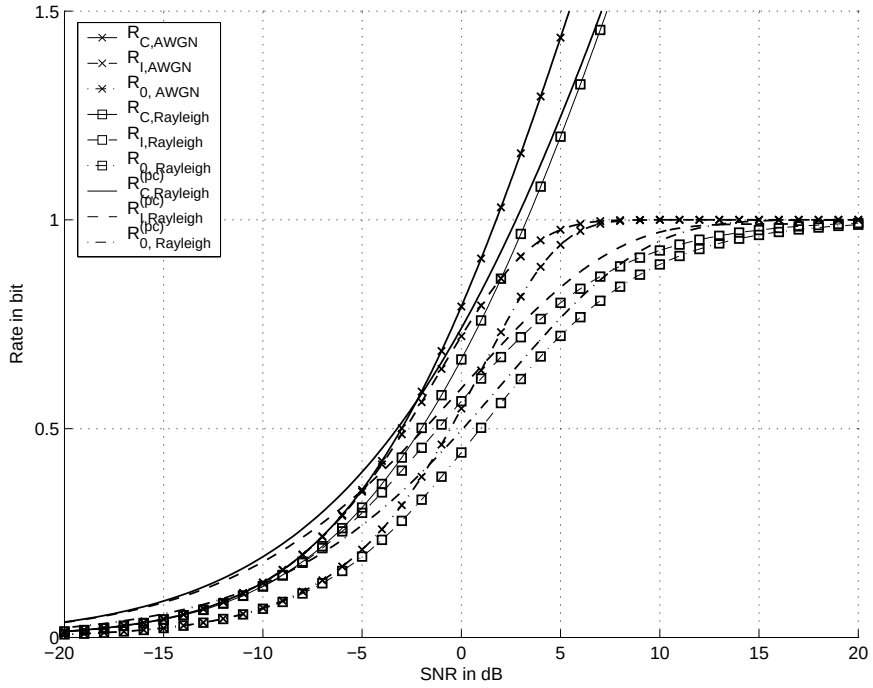


Figure 4.4: Capacity, mutual information and cutoff rate for perfect channel state information at the transmitter and applying power control according to (4.17).

4.1.5 Automatic Repeat Request Schemes

Feedback on the physical layer allows to provide the transmitter with CSI. The improvements in the supported rates for the block–fading AWGN channel have been shown in Figure 4.4. However, the

availability of CSI in form of the channel gain in future transmission slots at the transmitter requires some specific system properties as discussed in subsection 2.4.3 and is not feasible for many system designs. However, feedback on the link layer is widely exploited by the use of retransmission protocols, usually combined with FEC. These combined error protection protocols are usually referred to as *hybrid Automatic Repeat reQuest (ARQ)* protocols. In the following we assume a simple stop–and–wait retransmission protocol with a binary feedback for each transmitted slot, namely positive ACKnowledgement (ACK) and Negative AcKnowledgement (NAK). We ignore any implementation details like sequence numbering or erroneous feedback links as this low–rate feedback information is assumed to be perfectly received at the transmitter.

We will briefly describe three hybrid ARQ schemes which basically can be distinguished by their simplicity–performance tradeoff. The protocols are embedded in the block–fading AWGN framework with transmission slots denoted as f where each slot can transport N_v channel symbols. Therefore, we restrict ourselves to the case that retransmissions must also use N_v symbols. As we mainly deal with real–time transmission in this work, we restrict the maximum amount of available transmission slots for each source message $\mathbf{u}$ to a fixed number F to bound the maximum delay.

The simplest form of a hybrid ARQ scheme – referred to as *ARQI* in the remainder – provides F transmission attempts for one and the same message $\mathbf{u}$ of length k. Assume that this message is encoded to a channel symbol vector $\mathbf{x} \in \mathbb{C}$ of length $n = N_v$ and transmitted on radio slot $f = 1$ to the receiver. The receiver determines an estimate of the transmitted information message $\mathbf{u}$ by processing the channel LLR $\boldsymbol{\psi}_1$ of the received signal $\mathbf{y}$ based on code $\mathbb{C}$. If decoding is successful, the receiver sends an ACK to the transmitter on the feedback channel and the transmission of this message stops. In contrast, if the receiver detects an error, a NAK is sent. In this case, the transmitter repeats the same code word $\mathbf{x}$ on slot $f = 2$ and decoding is based on the channel LLR of the received signal $\boldsymbol{\psi}_2$. The process continues until either the code word is correctly received or the number of slots for this particular slot is consumed, i.e., $f = F$. If transmission stops after exactly the f-th transmission attempt, the effective code rate for the current code word is $r = k/(fN_v)$. If successful decoding does not occur within F slots then this message becomes useless and is declared as an outage.

The same protocol with only receiver modifications leads to a more powerful system, denoted as *ARQI+* in the following. Instead of ignoring the received signals of the previous transmission attempts $\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{f-1}$ in the f-th decoding attempt, maximum ratio combining of all received signals is performed. The receiver decodes the code $\mathbb{C}$ using the sum of the channel LLR

$$\boldsymbol{\psi}_f^+ \triangleq \sum_{i=1}^{f} \boldsymbol{\psi}_i,$$

which results in an optimum diversity effect by the application of maximum–ratio combination of all previously received transmission slots. The additional complexity in the system comes from the fact that the receiver must store the sum LLR $\boldsymbol{\psi}^+$.

From the channel coding point–of–view, ARQI type retransmission schemes can be viewed as a repetition code. Obviously, with the introduction of better codes, efficiency can be increased significantly. In so called ARQ type 2 protocols – referred to as *ARQII* in the following – incremental redundancy is exploited. For this purpose, we use rate–compatible punctured codes as introduced in subsection 4.1.2. Assume that the message $\mathbf{u}$ of length k bit is encoded to a channel code vector $\mathbf{x} \in \mathbb{C}$ of length $n = N_w N_v$ where N_w is a given integer. The codeword is divided into N_w sub–codes, each of length N_v, such that it can be modulated and transmitted in a separate radio slot. The code book $\mathbb{C}_w$, $w = 1, \ldots, N_w$ denotes the punctured code of length $N_w N_v$ obtained from $\mathbb{C}$ by deleting the last $(N_w - w)N_v$ symbols. Assume that this code word is stored at the transmitter.

On the first slot $f = 1$ the first N_v symbols are transmitted to the receiver. The receiver decodes the code $\mathbb{C}_1$ by processing the channel LLR of the received signal $\boldsymbol{\psi}_1$. If decoding is successful, an ACK is fed back to the transmitter and transmission for this message stops. In case of a failure and the reception of a NAK at the transmitter, not the same code word as in case of ARQI is sent on slot $f = 2$ but the remaining N_v symbols of $\mathbb{C}_2$. Now the receiver decodes code $\mathbb{C}_2$ by using the channel LLR of the received signal blocks $\{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2\}$. The process continues until either the code word is correctly received or the number of slots for this particular slot is consumed, i.e., $f = F$. It is assumed that sufficient sub–codes are available, i.e., $N_w \geq F$. The code rate after exactly the f-th attempt is again given as $r = k/(fN_v)$.

For the analysis of these ARQ schemes we assume in the following asymptotically optimal codes according to (4.8), and for the protocols ARQII we specifically use asymptotically optimal rate–compatible punctured codes. The presentation is again based on the mutual information R_I but can be replaced by any other measure describing the code performance. We are interested in the throughput η of the different ARQ protocols which depends on the applied code rate r and the number of available transmission slots F for each message as

$$\eta^{(F)}(r) \triangleq \frac{r}{\bar{F}}, \tag{4.19}$$

where $\bar{F}$ specifies the average amount of slots necessary to transmit a message with code rate r over a channel with iid channel states H having available a maximum of F transmission slots for each message. The throughput according to (4.19) is measured in terms of number of bits per channel symbol and transmission slot.

We continue with the derivation of the denominator for different ARQ protocols. In [CT01] it was shown that the probability $p_d^{(f)}(r)$ of successfully decoding a message coded with rate r after exactly f transmitted slots is given by

$$p_d^{(f)}(r) = p_o^{(f-1)}(r) - p_o^{(f)}(r), \tag{4.20}$$

where $p_o^{(f)}(r)$ denotes the probability that a message with code rate r cannot be decoded when transmitted over f radio slots. This probability obviously not only depends on the code rate r and the number of radio slots f, but also on the ARQ protocol in use as

$$p_o^{(f)}(r) = \begin{cases} \left(\Pr\left\{R_I(H) \leq r\right\}\right)^f & \text{for ARQI,} \\ \Pr\left\{R_I\left(\sum_{i=1}^f H_i\right) \leq r\right\} & \text{for ARQI+,} \\ \Pr\left\{\sum_{i=1}^f R_I(H_i) \leq r\right\} & \text{for ARQII,} \end{cases} \tag{4.21}$$

with the definition $p_o^{(0)}(r) \triangleq 1$. Interestingly, $p_o^{(f)}(r)$ serves again as an information outage probability. For the ARQII case with the application of appropriate rate–compatible punctured codes, the outage probability is equivalent to the outage probability as defined in (4.13). However, for the ARQI type protocols, instead of using optimal codes, low–performance repetition coding is applied.

We can basically distinguish two different operation modes. In *persistent acknowledged mode* the transmission of a single message $\mathbf{u}$ is performed until it is correctly received at the receiver, i.e., $F \to \infty$. In this case the residual error probability obviously goes to zero and the throughput $\eta_\infty(r)$ results in

$$\eta_\infty(r) = \lim_{F\to\infty} \eta^{(F)}(r) = \frac{r}{\sum_{f=1}^\infty f p_d^{(f)}(r)} = \frac{r}{\sum_{f=0}^\infty p_o^{(f)}(r)}. \tag{4.22}$$

Note that the average delay measured in radio slots results is equivalent to the average number of radio slots necessary to transmit a message coded with code rate r as $\sum_{f=0}^{\infty} p_o^{(f)}(r)$. In addition, for ARQII it is possible and also useful to have initial code rates r, i.e., the code rate for the first transmission attempt, being greater than 1. This is as with the code combination after receiving a certain amount f of radio slots we obtain a code with overall code rate $r_f = r/f$ being below 1 which possibly allows decoding of the transmitted message.

Figure 4.5 shows different results assuming asymptotically optimal codes operating at the cutoff rate R_0 for fully persistent transmission. Figure 4.5(a) shows the throughput $\eta_\infty(r)$ over the initial code rate r for different protocols as well as different average SNR $\mathbb{E}\{H\}$. For ARQI and ARQI+ a clear global optimum for $r < 1$ is obvious where the gains using ARQI+ are observed in terms of higher throughput. For the ARQII no initial code rate $r \in [0;1]$ exists which globally maximizes the overall throughput is found, but $r \to \infty$ maximizes the overall throughput. The optimization suggests to transmit a single code word with infinite block length k over $F \to \infty$ transmission slots and the code rate corresponds to $R_0(\mathbf{H})$, the cutoff rate of the fully interleaved Rayleigh fading channel. Depending on the average SNR a local maximum $r < 1$ for initial code rates $r \in [0;1]$ might exist.

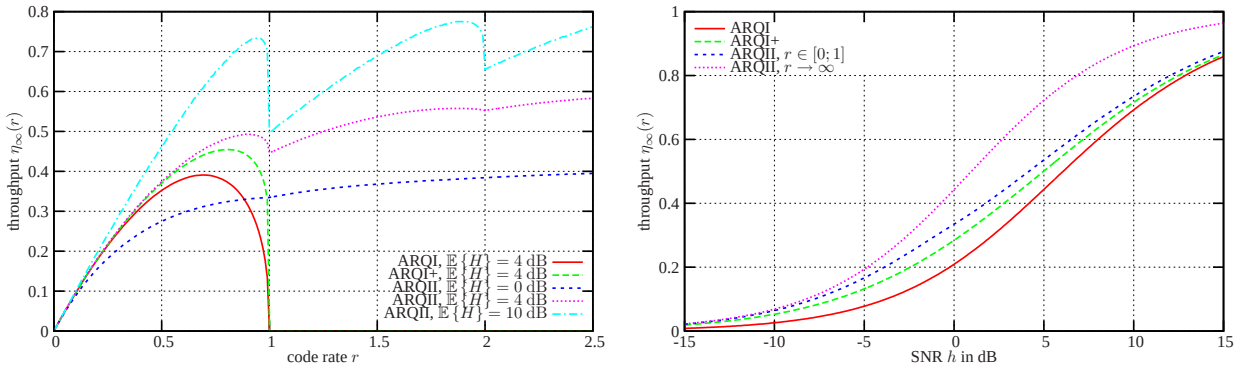


Figure 4.5: Performance of asymptotically optimal codes operating at the cutoff rate R_0 for fully persistent transmission and different protocols: (a) Throughput $\eta_\infty(r)$ over the initial code rate r for different average SNR $\mathbb{E}\{H\}$. (b) Throughput $\eta_\infty(r)$ over average SNR $\mathbb{E}\{H\}$ for optimal initial code rates $r \in [0;1]$ and $r \to \infty$ for ARQII.

Figure 4.5(b) compares the throughput $\eta_\infty(r)$ over the average SNR $\mathbb{E}\{H\}$ for optimal initial code rates $r \in [0;1]$. In addition, the performance for ARQII with $r \to \infty$ is shown which corresponds to the cutoff rate of the fully interleaved Rayleigh fading channel. The advantages of ARQI+ compared to ARQI can be signficant, ARQII with initial code rates $r \in [0;1]$ still provides gains, but less significant. However, the restriction to initial code rates $r \in [0;1]$ degrades the performance of ARQII significantly, but obviously keeps the delay within reasonable ranges. For all protocols and for average SNR greater than 0 dB the average delay does not exceed 2.5 radio slots.

The transmission of individual messages may still take a long time, and therefore, the persistent mode is not suited for real–time applications with delay limits. In case of delay constraints we assume the message transmission is stopped after $F < \infty$ radio slots regardless whether the message was received successfully or not. In the latter case, if the transmission of the message failed this results in an error. Then again for a particular channel H and a maximum amount of slots F available for the transmission of a certain message, the performance of the ARQ protocol is specified

by an error–rate pair where in this case the code rate is not explicitly given, but parameterizes the throughput[10] $\eta^{(F)}(r)$ and the residual error probability $\bar{P}_e(r, H)$. Both, the throughput and the error probability depend on the ARQ scheme in use. The residual error probability is directly obtained as $\bar{P}_e(r, H) = p_o^{(F)}(r)$. The computation of the throughput according to (4.19) results in

$$\eta^{(F)}(r) = \frac{r}{\sum_{f=1}^{F} f p_d^{(f)}(r) + \sum_{f=F+1}^{\infty} p_d^{(f)}(r)} = \frac{r}{\sum_{f=0}^{F-1} p_o^{(f)}(r)}. \tag{4.23}$$

Figure 4.6 shows the outage probability $p_o^{(F)}(r)$ over the normalized code rate r/F for different ARQ protocols and different number of maximum transmission slot F assuming asymptotically good codes operating at the cutoff rate for average SNR $\mathbb{E}\{H\} = 4\,\mathrm{dB}$. Obviously, for $F = 1$ all protocols behave identical to conventional FEC and the outage probability corresponds to $F = 1$ in Figure 4.2. With the possibility to use more than one slot for a single message coded with code rate r the performance changes depending on the protocol in use. For ARQI and ARQI+ the code rate obviously cannot exceed 1 such that only for normalized code rate below $1/F$ reliable transmission is possible. For all cases ARQI+ outperforms ARQI, the difference can be quite significant. For low initial code rates or low number of slots, the performance of ARQII does not yield significant gains when compared to ARQI+, only for the less practical case with initial code rates $r > 1$ and significant amount of independent channel realizations, i.e., $F > 4$, ARQII outperforms ARQI+. Note also that the ARQII protocol performance when plotted over the normalized code rate r/F is identical to the performance in Figure 4.2 with FEC only over code rate r. From this perspective, the use of more complicated ARQ protocols is not immediately justifiable.

Figure 4.7 gives insight into this behaviour showing the outage probability $p_o^{(F)}(r)$ over the throughput $\eta^{(F)}(r)$ for FEC, different ARQ protocols and different number of maximum transmission slot F assuming asymptotically good codes operating at the cutoff rate for average SNR $\mathbb{E}\{H\} = 4\,\mathrm{dB}$. For $F = 1$ the performance is independent of the protocol in use. However, with the use of more slots, in this case shown for $F = 4$ the differences are obvious. Whereas in case of FEC the throughput is constant as the same message is always transmitted in exactly F radio slots, in case of using an ARQ protocol the transmission of this message can be terminated earlier and a new message can be transmitted. Therefore, the throughput of the system depends not only on the probability that the message can be decoded after F radio slots but also after any $f < F$ radio slots. The performance for different ARQ protocols reveals the superiority of ARQII over ARQI+ which itself outperforms ARQI. The importance of the selection of an appropriate initial code rate r is again obvious. For example, different initial code rates might result in the same throughput, but in different outage probabilities. Again, for low outage probabilities the performance of ARQI+ and ARQII is only slightly different whereas ARQI performs significantly worse. More details of how to exploit ARQ protocols for delay–limited applications will be discussed and shown in chapter 8.

4.2 Channel Coding and Decoding Methods

4.2.1 Overview

The performance bounds presented in section 4.1 are based on random coding arguments. Based on this arguments it can be concluded that there exist at least one code which is as good as the

[10] The term ”throughput” might be misleading at this point as it does not distinguish whether the data received is correct or erroneous. Still, we stick with the terminology, but the index F indicates that only a limited amount of retransmissions, namely $F - 1$, can be applied.

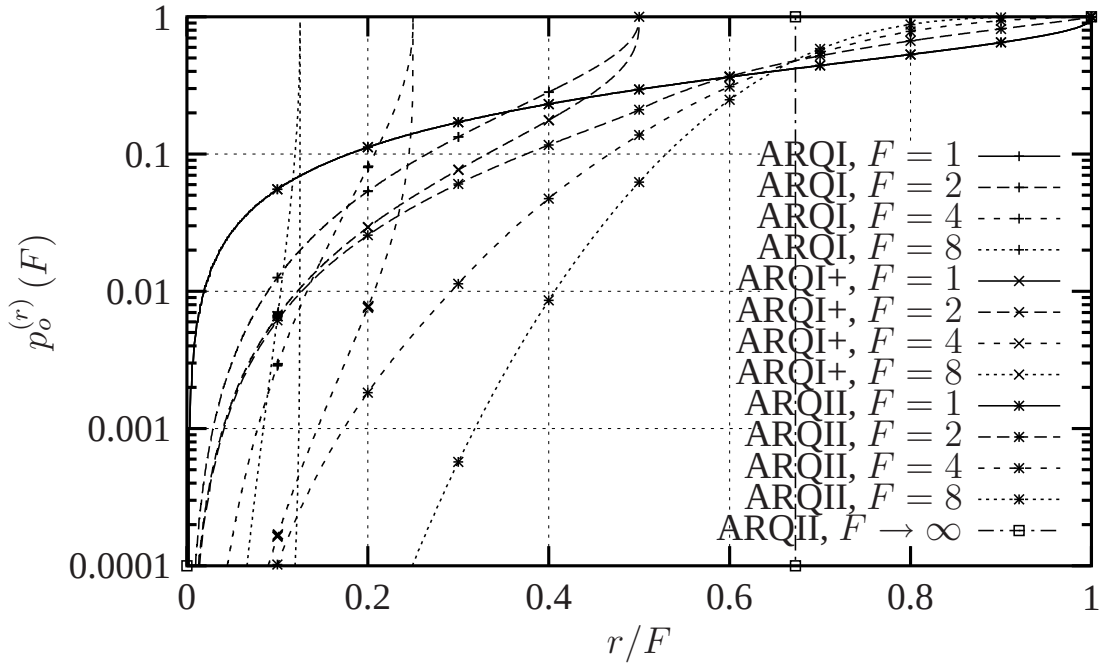


Figure 4.6: Outage probability $p_o^{(F)}(r)$ over the normalized code rate r/F for different ARQ protocols and different number of maximum transmission slot F assuming asymptotically good codes operating at the cutoff rate for average SNR $\mathbb{E}\{H\} = 4\,\mathrm{dB}$.

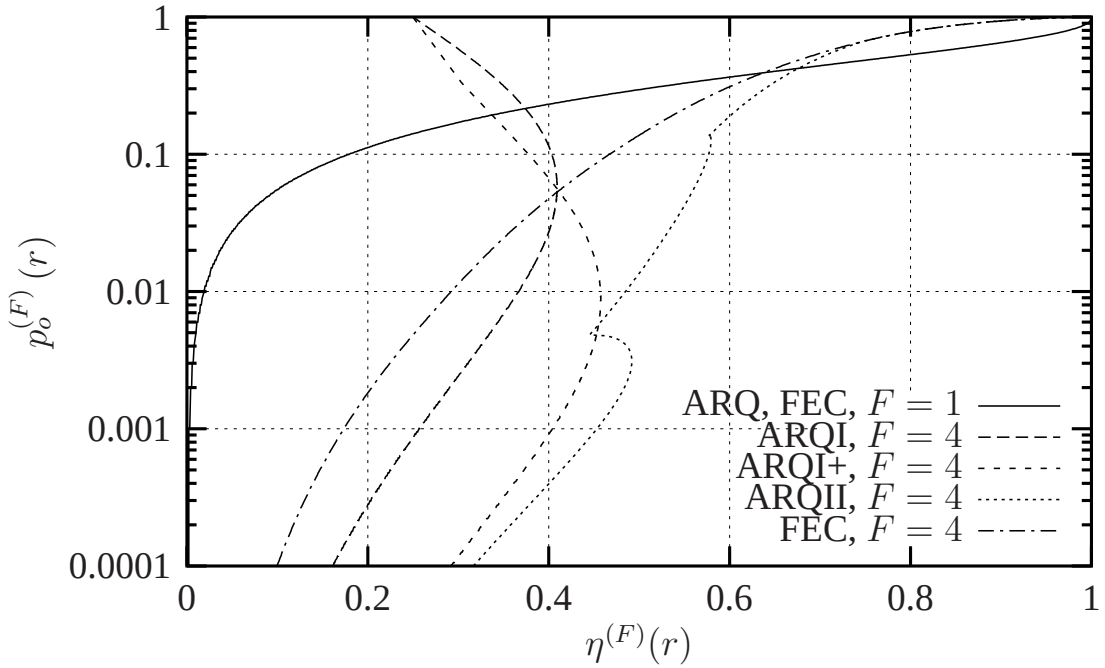


Figure 4.7: Outage probability $p_o^{(F)}(r)$ over throughput $\eta^{(F)}(r)$ for FEC, different ARQ protocols and different number of maximum transmission slot F assuming asymptotically good codes operating at the cutoff rate for average SNR $\mathbb{E}\{H\} = 4\,\mathrm{dB}$.

average code. The problem therefore is not in finding good codes, many codes are good, but in the implementation of the encoder and the decoder for a particular code. Assuming information words $\mathbf{u}$ of length k and code words $\mathbf{x}$ length n a random binary code would require storage capacity for a mapping table in encoder and decoder with 2^k entries storing, each entry requiring, $k+n$ bits. In addition, when using a maximum–likelihood decision rule, the metric would have to be computed for 2^k code words, whereby each of it requires n individual metric computations for received channel symbol. As n should be as large as possible for good performance according to (4.2) and for reasonable code rates, e.g, with $r = 1/2$, it is immediately obvious that implementing a random coding is completely infeasible[11]. Hence, in practical systems codes which possess a certain structure are required to obtain feasible complexity in the encoding and decoding process.

All practical codes represent a subspace of $\mathbb{F}_2^n$ and are linear such that linear algebra methods might be applied in the encoding and decoding process. For practical linear codes $\mathbb{C}$, the union bound provides a good measure [VO79, Pro95] on the message error probability depending on the channel state h as

$$P_B(\mathbb{C}, h) = \Pr\left\{\hat{\mathbf{X}} \neq \mathbf{X}\right\} \leq \sum_{d=d_{\text{free}}}^{\infty} a_d \cdot P_d(h). \tag{4.24}$$

with d_{free} the free distance of the code, $\{a_d\}_{d=d_{\text{free}}}^{\infty}$ the distance spectrum of the code, and $P_d(h)$ the pairwise error probability of two code words at Hamming distance d.

Note that the free distance d_{free} and the distance spectrum $\{a_d\}$ only depend on the channel code $\mathbb{C}$. Without power control, i.e., $\gamma = 1$, P_d exclusively depends on the Hamming distance d_f on each slot and the channel state H_f according to (2.46) at each of the different code word symbol as

$$P_{d,\mathbf{H}} = \mathbb{E}_{\mathbf{H}}\left\{\frac{1}{2}\operatorname{erfc}\left(\sqrt{\sum_{f=1}^{F} d_f H_f}\right)\right\} \leq \begin{cases} \frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{d\mathcal{E}_s}{N_0}}\right) & \text{for AWGN,} \\ \frac{1}{2}\left(\frac{1}{1+\frac{\mathcal{E}_s}{N_0}}\right)^d & \text{for Rayleigh.} \end{cases} \tag{4.25}$$

Note that "Rayleigh" refers to the case that the fading amplitudes affecting the d different symbols are iid Rayleigh with unit mean. For the block–fading AWGN channel the error probability also decreases exponentially similar to the iid Rayleigh fading case, but the Hamming distance is not symbol–based, but block–based [BPS98], i.e., it decreases with the amount of different blocks irrespective of the number of different symbols in each block. Note that for good channels, the error probability is dominated by the free distance of the code, but for worse channels also the distance spectrum a_d at low d is of relevance.

4.2.2 Convolutional Codes

Overview

The well–known benefits of convolutional codes – a time–invariant trellis structure that facilitates decoding – has led to an extensive use of trellis-based coding and decoding techniques in modern communication systems. A brief introduction is provided – for more details we refer to, e.g. [LJ83]. In what follows we describe a rate $1/c$ convolutional encoder as a device which generates the c–tuple $\mathbf{x}_t = (x_{1,t}, \ldots, x_{c,t})$ of code bits at time t given the information bit sequence $u_1, \ldots, u_t$ where

[11] For example, a random code with rate $1/2$ and $n = 500$ as the minimum typical block length in wireless communications would require about 10^{350} computations for each code word about the same number of storage capacity in bits. Mind that the number of atoms in the universe is estimated to be of order 10^{80}.

$x_{i,t}, u_t \in \mathbb{F}_2$, $t \geq 0$. Associating a particular sequence $\{u_t\}_{t=1}^{\infty}$ the series

$$u(D) = \sum_{t=1}^{\infty} u_t D^{t-1},$$

the code sequence $\{\mathbf{x}_t\}_{t=1}^{\infty}$ for time instant t results as the coefficients of the series

$$x_i(D) = u(D)g_i(D) \tag{4.26}$$

with $g_i(D)$ denoting the generator polynomials in $\mathbb{F}_2$ and all arithmetic operations performed in $\mathbb{F}_2$. The memory of the code μ is defined as $\mu \triangleq \max_i \deg g_i(D)$ such that any information bit u_t at time t influences $\mu + 1$ output tuples, namely $\{\mathbf{x}_t, \ldots, \mathbf{x}_{t+\mu}\}$. The encoding using polynomial generators can be implemented by the use of feed–forward encoders. The same code–words but a different mapping can be obtained by for example using rational generators

$$\hat{g}_i(D) \triangleq \frac{g_i(D)}{g_1(D)},$$

which result in a systematic code, but require a recursive implementation of the encoders. In general, systematic convolutional codes with comparable high free distance require rational generator functions. However, feed–forward systematic codes might also be generated by letting $g_1(D) = 1$ and using polynomial functions for $g_2(D), \ldots, g_c(D)$.

Block structure with $k < \infty$ rather than the initially provided desired semi–infinite structure is desired for most packet–based systems. In case that we want to obtain a block consisting of k information bits there exist basically three different techniques to obtain block structure for convolutional codes: (i) stopping the generation of code symbols after k information symbols, (ii) appending μ deterministic bits to the k information bits referred to as *termination*, or (iii) using tail–biting structure by appending the first μ bits of $\mathbf{u}$ as termination sequence. For (i) and (iii) the number of transmission symbols per information message $\mathbf{u}$ of length k results in resulting in $N_x = ck$. The first choice (i) results lower protection of the last information bits in $\mathbf{u}$, which is avoided by (iii), but results in the necessity of up to 2^{μ} more complex decoder structures and gets infeasible especially with increasing μ [Wei04]. In most practical systems block structure is achieved by termination which results in balanced protection, but slightly increased effective code rate as

$$r_{\text{term}} = \frac{k}{N_x} \quad \text{with } N_x = (k+\mu)c - \mu\mathbf{1}\left\{\text{systematic feedforward code}\right\}. \tag{4.27}$$

If the memory is much smaller than the number of information bits, i.e., $\mu \ll k$, the effective rate is almost identical to $1/c$, for other cases the ineffectiveness due to termination can be significant. For systematic feedforward encoders the deterministic bits appended to $\mathbf{u}$ need not be transmitted.

Due to extensive use of convolutional codes in different systems, there exist many optimal and sub–optimal decoding algorithms which can be differentiated with respect to their performance–complexity tradeoff. One of the reasons for the popularity of convolutional codes is the existence of manageable maximum–likelihood and sequence Maximum A Posteriori (MAP) decoding algorithms based on the Viterbi algorithm [Vit67]. In addition, the performance of convolutional codes for maximum likelihood decoding has been studied in [Vit67] and extended in [VO79]. Despite the possible representation of block codes with the use of termination, the performance of convolutional codes is not appropriately described by the block error probability according to (4.2), especially the the exponential decrease of the error probability with the block length n is not valid. To describe the performance of convolutional codes, Viterbi and Omura [VO79] used the *event*

error probability P_e which is defined as the probability that any another path has higher metric than the correct one. Employing the union bound and given that there are a_d incorrect paths at Hamming distance d from the correct path over an unmerged segment, the event error probability is upper–bounded as

$$P_e \leq \sum_{d=d_{\text{free}}}^{\infty} a_d \cdot P_d, \tag{4.28}$$

with P_d the pairwise error probability according to (4.25). Similarly to block codes, the performance of convolutional codes mainly depends on the free distance of the code, and to some extent on the multiplicity of error events at low distances. Heller [Hel68] introduced an upper bound on the minimum free distance of a rate $1/c$ convolutional code as

$$d_{\text{free}} \leq \min_{i \geq 1} \left\lfloor \frac{2^{i-1}}{2^i}(\mu + i)c \right\rfloor. \tag{4.29}$$

Good codes are found by computer search over all possible generators with a certain memory μ and a certain number of output symbols c. For tables and details see [Pro95] and references therein. It is obvious that the free distance and the code performance increases with increased number output symbols c, but the influence of the memory μ in the code performance is not immediately obvious. For a certain channel given by its h and with $r = 1/c, \quad 0 \leq r \leq R_0$, it is shown [VO79] that the event error probability strongly depends on the memory of the code as

$$P_e \leq \frac{2^{-(\mu+1)cR_0}}{1 - 2^{1-cR_0}}. \tag{4.30}$$

Even tighter bounds have been provided by in [VO79] using the error exponent according to (4.3). The importance of the memory on the event error probability for convolutional codes is obvious from (4.29) as well as from (4.30): The free distance basically increases linearly with the memory and the event error probability decreases exponentially with the memory of the code.

Representations of Convolutional Codes

There exist basically three different methods to describe convolutional codes, namely a state diagram, a tree diagram, and a trellis diagram. Especially the latter two are used in decoding algorithms and will therefore be introduced briefly in the following. For sake of clarity we restrict ourselves to rate $1/c$ convolutional codes with encoder having μ memory elements though the extension to general k/c codes is not significantly different. In a straight–forward manner a convolutional code is represented as a *tree diagram*. For binary input symbols from each node $\mathcal{N}$ there originate 2 branches ending in 2 successor nodes. The tree diagram of a convolutional code with $(c = 2, k = 1, \mu = 2)$ is shown in Figure 4.8. Each node $\mathcal{N}$ at depth t depends on the first t bits $\mathbf{u}_{(0:t]}$ of input sequence $\mathbf{u}$ such that we express $\mathcal{N}(\mathbf{u}_{(0:t]})$. The number of nodes grows exponentially with the depth of the tree t.

For convolutional codes it is observed that a certain structure in the tree repeats itself after μ stages. This redundancy is removed by merging all nodes at a certain tree depth t with the same properties to a single state $\pi_t \in \{1, \ldots, 2^\mu\}$ expressing the state of the convolutional encoder. For convenience, let us define the state $\pi(\mathbf{u}_{(0:t]})$ as the state of the convolutional encoder after encoding the information sequence $\mathbf{u}$ up to information symbol u_t. For completeness we include the initial state $\pi(\mathbf{u}_{(0:0]}) \triangleq \pi_0$ with π_0 representing the initial state of the encoder prior to encoding u_1. In case of termination the information sequence is extended with μ known bits such that

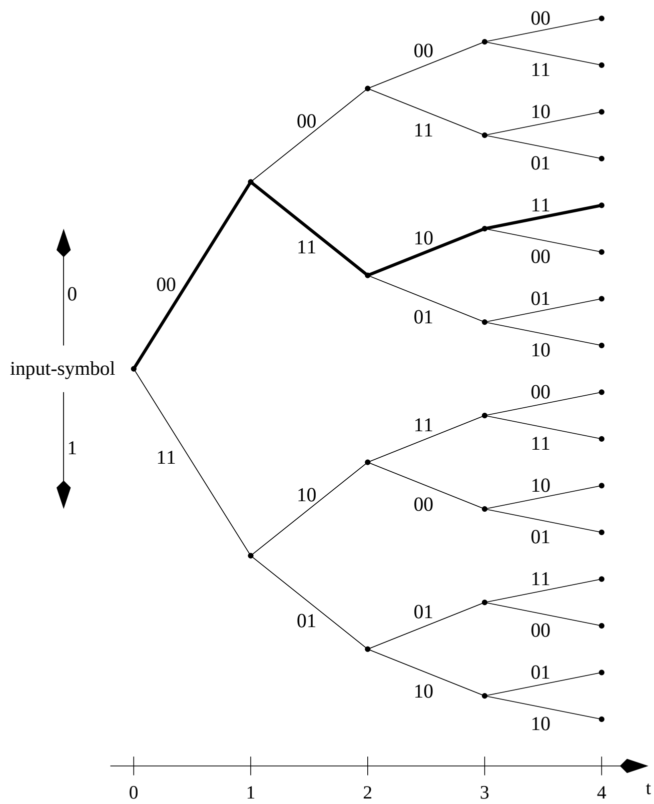


Figure 4.8: Tree representation of $(c = 2, k = 1, \mu = 2)$ convolutional code.

without loss of generality $\pi_{\text{term}} \triangleq \pi(\mathbf{u}_{(0:k+\mu]}) = \pi_0$. Depending on the state $\pi(\mathbf{u}_{(0:t-1]})$ and the information bit at position t, u_t, we obtain the state at position t, $\pi(\mathbf{u}_{(0:t]})$, as well as the output tuple $\mathbf{x}_t(\mathbf{u}) = \mathbf{x}(\pi(\mathbf{u}_{(0:t-1]}), \pi(\mathbf{u}_{(0:t]}))$ according to (4.26). The encoding of an information sequence $\mathbf{u}$ results in a state sequence $\{\pi_t(\mathbf{u})\}_{t=0}^{k+\mu}$. The encoder can be interpreted as a finite state machine with 2^μ states and a description is obtained by a *trellis diagram*. The trellis representation of a convolutional code with $(c = 2, k = 1, \mu = 2)$ is shown in Figure 4.9. Note that only depending on the state $\pi_{t-1}(\mathbf{u})$ and the information bit at position t, u_t, we obtain the successor state at position t, $\pi_t(\mathbf{u}) = \pi_s(\pi_{t-1}(\mathbf{u}), u_t)$, as well as as the output tuple $\mathbf{x}_t(\mathbf{u}) \triangleq \mathbf{x}(\pi_{t-1}(\mathbf{u}), u_t)$ according to (4.26). Also obvious from this representation is that the state sequence is uniquely defined in the reverse direction $\pi_t(\mathbf{u}) = \pi_p(\pi_{t+1}(\mathbf{u}), u_{t+1})$.

Optimal Sequence–Based Decoding Algorithms for Convolutional Codes

As already mentioned there exist several decoding algorithms for convolutional codes. Maximum likelihood, sequence MAP and symbol–based MAP decoding is commonly implemented by the exploitation the trellis structure of convolutional codes. Based on the observed channel output $\mathbf{y}$, the sequence MAP decoder according to (2.38)[12] searches for the sequence fulfilling

$$\hat{\mathbf{u}} \;=\; \arg\max_{\mathbf{u}\in\mathcal{S}_u^k} \Pr\left(\mathbf{X} = \mathbf{x}(\mathbf{u})\,|\mathbf{Y} = \mathbf{y}\right).$$

Assuming statistically independent information symbols $u_t, t = 1, \ldots, k$, a discrete memoryless channel, and a terminated convolutional code such that $\mathbf{u}$ is extended by μ information symbols,

[12]We ignore the dependency on the channel realization h in the following.

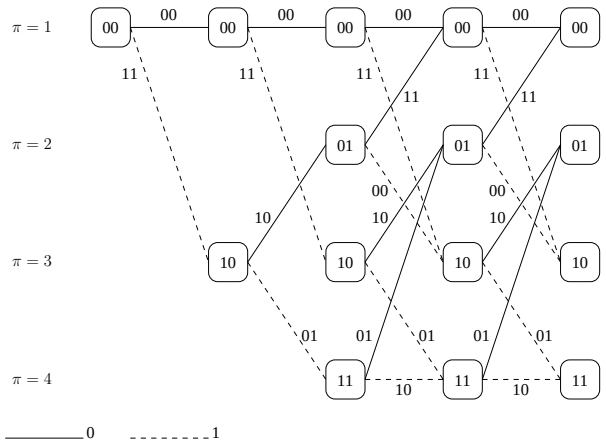


Figure 4.9: Trellis representation of $(c = 2, k = 1, \mu = 2)$ convolutional code.

and the a priori probability for the symbols returning into the terminating state $\pi_{k+\mu}$ is one 1 for all information symbols $u_t : \forall t = k+1, \ldots, k+\mu$, then the sequence MAP decision can be expressed as

$$\begin{aligned}
\hat{\mathbf{u}} &= \arg\max_{\mathbf{u}\in\mathcal{S}_u^k} \left[\sum_{i=1}^{N_x} \log \Pr\{y_i | x_i(\mathbf{u})\} + \log \Pr\{\mathbf{X} = \mathbf{x}(\mathbf{u})\} \right] \\
&= \arg\max_{\mathbf{u}\in\mathcal{S}_u^k} \left[\sum_{t=1}^{k+\mu} \left(\sum_{i=1}^{c} \log\left(\Pr\{y_{t,i} | x_{t,i}(\mathbf{u})\}\right) + \log\left(\Pr\{u_t\}\right) \right) \right] \\
&= \arg\max_{\mathbf{u}\in\mathcal{S}_u^k} \left[\sum_{t=1}^{k+\mu} \lambda\left(\boldsymbol{\psi}_t, \mathbf{x}_t(\mathbf{u}), L_t, u\right) \right]
\end{aligned} \tag{4.31}$$

where the metric increment is in general a function of the channel LLR $\boldsymbol{\psi}$, a transmitted vector $\mathbf{x}$, an a priori LLR L, and the bit u. For sequence MAP decoding the metric λ_{MAP} can be computed as

$$\lambda_{\text{MAP}}(\boldsymbol{\psi}, \mathbf{x}, L, u) \triangleq \sum_{i=1}^{c} \psi_i \cdot x_i + L \cdot u, \tag{4.32}$$

and L specifies the a priori knowledge on bit u. With the assumption of equiprobable input symbols u we have $L = 0$ and the sequence MAP decoding becomes a maximum–likelihood decoding strategy with metric increment $\lambda_{\text{ML}}(\boldsymbol{\psi}, \mathbf{x}) = \lambda_{\text{MAP}}(\boldsymbol{\psi}, \mathbf{x}, 0, u)$. For completeness we continue with a general metric λ being replaced by λ_{MAP} or λ_{ML} for sequence MAP or maximum–likelihood decoding, respectively. A straight–forward brute force decoder would basically determine all cumulative metrics for all possible $\mathbf{u} \in \mathcal{S}_u^k$ and select the one with the highest metric. Note that this decoding is not restricted to convolutional codes, but can be applied to any code.

In the following we will present the Viterbi algorithm as a specific realization of a dynamic programming problem. The possibly complicated presentation is motivated as we will use a very similar algorithm for other purposes in section 5.3 and section 5.4. The analogy of this algorithm with the Viterbi algorithm will be obvious.

Let us define a cumulative metric $\Lambda_t(\mathbf{u})$ for each information symbol position t as

$$\Lambda_t(\mathbf{u}) \triangleq \sum_{j=1}^{t} \lambda(\boldsymbol{\psi}_j, \mathbf{x}_t(\mathbf{u}), L_j, u_j) \tag{4.33}$$

Then, the problem in (4.31) of finding the optimal decoded sequence$\hat{\mathbf{u}}$ reduces to

$$\hat{\mathbf{u}} = \arg\max_{\mathbf{u}\in\mathcal{S}_u^k} \Lambda_t(\mathbf{u}) \tag{4.34}$$

with $t = k + \mu$. We introduce a recursive presentation for the metric $\Lambda_t(\mathbf{u})$ as well as for the encoder state $\pi_t(\mathbf{u})$ as

$$\begin{aligned} \Lambda_t(\mathbf{u}) &= \Lambda_t(\mathbf{u}) + \lambda(\boldsymbol{\psi}_t, \mathbf{x}(\pi_{t-1}(\mathbf{u}), u_t), L_t, u_t), \\ \pi_t(\mathbf{u}) &= \pi_s(\pi_{t-1}(\mathbf{u}), u_t), \end{aligned} \tag{4.35}$$

with $\Lambda_0(\mathbf{u}) \triangleq 0$ and $\pi_0(\mathbf{u}) \triangleq \pi_0$. Note, that for any $\mathbf{u}$, $\Lambda_t(\mathbf{u})$ and $\pi_t(\mathbf{u})$ do not depend on $\mathbf{u}_{(t:k+\mu]}$. With that, we can solve (4.31) as follows. Let $\hat{\mathbf{u}}_{(0:t],\pi}$ be the solution for the minimization problem in (4.34) and let $\Lambda^*_{t,\pi}$ with $\pi = \pi_t(\hat{\mathbf{u}})$ denote the optimal value of the objective function in (4.34). Therefore, $\hat{\mathbf{u}}_{(0:t],\pi}$ is the sequence of decoded bits fulfilling the MAP or maximum–likelihood criterion in (4.34) for an information bit length t and a termination state π. Then the following holds.

1. $\Lambda^*_{t,\pi}$ satisfies the dynamic programming equation

$$\Lambda^*_{t,\pi} = \max_{u\in\{\pm 1\}} \left(\Lambda^*_{t-1,\pi_p(u)} + \lambda(\boldsymbol{\psi}_t, \mathbf{x}(\pi_p(\pi, u), u), L, u) \right). \tag{4.36}$$

2. The information bit $\hat{u}_t(\pi)$ denotes the source bit achieving the minimum in (4.36).

3. The subsequence $\hat{\mathbf{u}}_{(0:t]}$ solves (4.34) for position t and termination state π.

This is obvious with the recursions in (4.35) and as for any $\mathbf{u}$, $\Lambda_t(\mathbf{u})$ and $\pi_t(\mathbf{u})$ only depends on $\mathbf{u}_{(0:t]}$. The key observation in (4.36) is that to obtain the optimum solution for $\hat{u}$ in terms of maximizing the a posteriori probability it is sufficient for each stage t and each possible state π to only look at the optimum subsequence $\hat{\mathbf{u}}_{(0:t],\pi}$. The operation in (4.36) is usually referred to as *add–compare–select* operation. To find the optimum decoded sequence $\hat{\mathbf{u}}$ it is sufficient to compute $\Lambda^*_{t,\pi}$ recursively for each position t and each state π based on the trellis representation. However, this still requires 2^μ add–compare–select operations for each information bit meaning that the complexity of the decoding algorithm grows exponentially with the memory μ of the code. The same holds for the storage requirements of the code.

Rate–Compatible Convolutional Codes

With regular $1/c$–convolutional codes only a very restricted set of code rates at or below $1/2$ can be supported. However, it has already been discussed that for different purposes, higher rate codes and flexibility by the use of rate–compatible codes is desired. Rate–compatible puncturing for convolutional codes has been introduced in [Hag88] with the so–called Rate–Compatible Punctured Convolutional (RCPC) codes. In this case, for each group of ω_p information bits only those code bits of the original encoder are transmitted that correspond to a 1 in the $c \times \omega_p$ binary puncturing

matrix $\mathcal{P}_i$ with $i = 1, \ldots, c\omega_p$ [Hag88]. $\mathcal{P}_i$ contains $n_i \triangleq \omega_p + i$ 1s such that the rate of the code is $r = \omega_p/n_i$ if we ignore termination overhead for the moment. With the definition of appropriate puncturing matrices $\mathcal{P}_i$, $i = 1, \ldots, (c-1)\omega_p$ in complete $\omega_p(c-1)$ different channel coding rates are available by this punctured code. Note that $i = 0$ corresponds to uncoded transmission if we assume systematic convolutional codes. For convenience we define the discrete set of available code rates for a practical convolutional code as $\mathcal{S}_r$.

Assuming maximum–likelihood decoding and employing the union bound similar to (4.28) for regular $1/c$–convolutional codes and generalizing a_d as the number of incorrect paths for ω_p successive nodes, i.e., inside one puncturing period, we obtain a generalization on the upper bound for the event error probability as

$$P_B(\mathbb{C}, h) \leq \frac{1}{\omega_p} \sum_{d=d_{\text{free}}}^{\infty} a_d \cdot P_d. \tag{4.37}$$

Good non–systematic RCPC codes with high d_{free} and also low multiplicity a_d have been presented in [Hag88]. We have extended this work to systematic recursive convolutional codes in [HS99].

4.2.3 Sequential Decoding of Convolutional Codes

The optimal algorithms minimizing the sequence and symbol error probability, the Viterbi and the BCJR algorithms both have exponential complexity and storage constraints with the memory μ of the code making them impracticable for convolutional codes with high memory. However, as indicated in (4.29) and (4.30), the performance of the convolutional code increases significantly with higher memory resulting in an obvious contradiction. Even prior to the invention of the Viterbi algorithm, Wozencraft [WR61] and Fano [Fan63b] introduced a sequential decoding algorithm which searches for the most probable path through a tree by examining only one path at a time. In [Mas72] Massey showed that the decoding of variable length codes and the sequential decoding of convolutional codes are two instances of the same underlying problem. In both cases the decoder operates on a tree with paths of different lengths. In case of the variable length decoder this tree represents the codewords of the variable length code, while in case of the sequential decoder the tree represents the partially explored code tree. The goal of both decoders is to find the path through the tree which given the observed sequence $\mathbf{y}$ was most likely transmitted. Using this analogy Massey proved that the Fano metric proposed by Fano on intuitive grounds is exactly the metric that minimizes error probability and, therefore, should be used for path selection with sequential decoding [Mas72]. The main difference between the Viterbi algorithm and the sequential decoding algorithms is that paths of different length have to be compared which required a metric adaptation in (4.32). We will briefly introduce the background of the Fano metric as it is used to derive a reliability vector for the decoded path of a sequential decoder in section 4.4.

Decoding of Variable–Length Codes and the Fano Metric

Let us assume a set of messages $\mathcal{U}$ denoted by integers $u = 1, 2, \ldots$. Each message $u \in \mathcal{U}$ is mapped to a variable length code word $\mathbf{x}(u) = \{x_1(u), x_2(u), \ldots, x_{l\{\mathbf{x}(u)\}}(u)\}$ of length $l\{\mathbf{x}(u)\}$. The code symbols $x_t(u)$, $t = 1, \ldots, N_u$, belong to some alphabet $\mathcal{S}_x$[13] and the maximum codeword length is denoted by $N_{\mathcal{U}}$, i.e., $N_{\mathcal{U}} = \max_{u \in \mathcal{U}} l\{\mathbf{x}(u)\}$. The set $\mathcal{T}_x = \{\mathbf{x}(u) \,|u \in \mathcal{U}\}$ of all codewords defines a VLC characterized by prefix–free code words and can be represented by a tree.

[13] In the sequel we will again restrict ourselves to binary code words with $\mathcal{S}_x = \mathbb{F}_2$

Assume now that the code words are transmitted over a discrete memoryless channel. Based on the observed channel output sequence $\mathbf{y} = \mathbf{y}_{(0:N_{\mathcal{U}}]}$, the goal of the VLC decoder is the estimation of the transmitted variable length codeword $\mathbf{x}(u)$ and its assigned message u in such a way that the error probability is minimized. This results in a MAP decision rule as

$$\hat{u} = \arg\left(\max_{u\in\mathcal{U}} \Pr\{u, \mathbf{y}\}\right). \tag{4.38}$$

Massey showed in [Mas72] by the use of random tail extension of shorter code words that the probability $\Pr(u, \mathbf{y})$ yields

$$\Pr(u, \mathbf{y}) = C(\mathbf{y}) \Pr\{u\} \cdot \exp\left(\sum_{t=1}^{l\{\mathbf{x}(u)\}} \lambda_{\text{Fan}}(y_t, x_t(u))\right), \tag{4.39}$$

with $C(\mathbf{y})$ being a constant which only depends only on the received sequence $\mathbf{y}$ and $\Pr\{u\}$ the a priori probability of message u. The metric increment $\lambda_{\text{Fan}}(y, x)$ is given by

$$\lambda_{\text{Fan}}(y, x) = \log\left(\frac{\Pr\{Y = y|X = x\}}{\Pr\{y\}}\right). \tag{4.40}$$

For information messages u with $\Pr\{u\} \propto \exp\left(-l\left\{\mathbf{x}(u)\right\}\right)$, (4.39) is referred to as *Fano metric* as it was first proposed by Fano [Fan63b] to be used for sequential decoding of convolutional codes. The application will be discussed in the following.

Consider an information sequence $\mathbf{u}$, with $\mathbf{u} = (u_1, \ldots, u_k)$ and the corresponding code word, encoded with a convolutional code with mother code rate $1/c$ and termination, is denoted as $\mathbf{x}(\mathbf{u})$. Note that after encoding, all code words $\mathbf{x}(\mathbf{u})$ have the same length N_x. The transmission over a discrete memoryless channel with transition probability $\Pr\{Y = y|X = x\}$ results in a received sequence $\mathbf{y}$, being represented by the channel LLR $\boldsymbol{\psi}$ in the following.

In contrast to maximum–likelihood or MAP decoding algorithms sequential decoding algorithms do not take into account all code words with full length. Instead, the decoding is based on the construction of a VLC tree which decides for the most likely code word in the tree. The recursive generation of the VLC tree $\mathcal{T}$ is discussed later, it basically distinguishes different sequential decoding algorithms. For a given tree, the decision on the most likely code word is equivalent to the decoding of a VLC. Consider for the moment that we have given any information bit sub–tree $\mathcal{U} \triangleq \mathcal{T}_u \subseteq \{\pm 1\}^k$ containing any number of subsequences $\mathbf{u}$ in form of a VLC. The length of an information word is denoted as $l\{\mathbf{u}\}$. By encoding all sequences in $\mathcal{U}$ with a convolutional encoder we obtain a code symbol symbol tree $\mathcal{T}_x \subseteq \mathcal{S}_x^{N_x}$. Note that $\mathcal{U}$ and $\mathcal{T}_x$ are equivalent and can be transformed into each other by the encoding prescription according to (4.26).

Then, the joint probability for a sequence $\mathbf{y}$ and any VLC information sequence $\mathbf{u} \in \mathcal{U}$ based on the information symbol $\mathcal{U}$ results in a similar expression as the sequence MAP decision rule according to (4.31), namely

$$\begin{aligned}\hat{\mathbf{u}} &= \arg\max_{\mathbf{u}\in\mathcal{U}} \Pr\{\mathbf{u}, \mathbf{y}\,|\mathcal{U}\} \\ &= \arg\max_{\mathbf{u}\in\mathcal{U}} \left(\sum_{t=1}^{l\{\mathbf{u}\}} \lambda_{\text{VLC}}\left(\boldsymbol{\psi}_t, \mathbf{x}_t(\mathbf{u}), L_t, u_t\right)\right),\end{aligned} \tag{4.41}$$

where the metric increment λ_{VLC} is in general a function of the channel LLR $\boldsymbol{\psi}$, the code symbol vector $\mathbf{x}$, an a priori LLR L, and the information bit u itself. In [WRH96] it was shown that the

metric increment λ_{VLC} for a convolutional code with code rate $1/c$ is expressed as

$$\lambda_{\text{VLC}}\left(\boldsymbol{\psi}, \mathbf{x}, L, u\right) = c\log(2) - \sum_{i=1}^{c} \log\left(1 + \exp\left(-\psi_i x_i\right)\right) - \log\left(1 + \exp\left(-L \cdot u\right)\right). \tag{4.42}$$

Again, equivalently to (4.33), we introduce a cumulative metric $\Lambda_t(\mathbf{u})$ for each information symbol position t as

$$\Lambda_t(\mathbf{u}) \;\triangleq\; \sum_{j=1}^{t} \lambda_{\text{VLC}}(\boldsymbol{\psi}_j, \mathbf{x}_j(\mathbf{u}), L_j, u_j) \tag{4.43}$$

Then, the problem in (4.41) of finding the optimal VLC codeword $\hat{\mathbf{u}}$ reduces to

$$\hat{\mathbf{u}} = \arg\max_{\mathbf{u}\in\mathcal{U}} \Lambda_t(\mathbf{u}) \tag{4.44}$$

with $t = l\left\{\mathbf{u}\right\}$. For convenience, we drop the length information $l\left\{\mathbf{u}\right\}$ in case that the metric of the entire code word is the measure of interest, i.e., $\Lambda(\mathbf{u}) \triangleq \Lambda_{l\{\mathbf{u}\}}(\mathbf{u})$. It is worth to note that for code words of equal length the decision in (4.44) is equivalent to the sequence MAP decision in (4.34). For details on the transformation steps we refer to [Don99]. In addition, let us emphasize the joint probability $\Pr\{\mathbf{u}, \mathbf{y}\}$ is expressed as

$$\Pr\{\mathbf{u}, \mathbf{y}\} = C(\mathbf{y}) \cdot \exp\left(\Lambda(\mathbf{u})\right). \tag{4.45}$$

Finally, a recursive presentation for the metric $\Lambda_t(\mathbf{u})$ in (4.43) as well as for the encoder state $\pi_t(\mathbf{u})$ is given by replacing λ with λ_{VLC} in (4.35).

In what follows, we briefly review the theory of sequential decoding algorithms. The task of every algorithm for decoding convolutional codes consists in searching the code tree for the path that was most likely transmitted given the received sequence. The principle idea of all sequential decoding algorithms is to restrict this search and to explore only a selection of all paths — the most promising paths — in the decoding process. This selection is based on a suitable metric and is justified intuitively by the argument, that if a path seems not to be promising at a certain node in the code tree, all of the paths stemming from it can be discarded without any essential loss in comparison to maximum likelihood decoding.

Since at any time instant typically only the most promising path is extended, a sub–tree of the entire code tree is explored during the decoding process requiring a metric that enables us to compare paths of different lengths. The procedure as defined in (4.41) fulfills these requirements. The most prominent sequential decoding algorithms are the *stack* and the *Fano* algorithms, for a detailed description of these algorithms we refer to [LJ83]. However, we will introduce the algorithms briefly in the following as the algorithms developed in section 4.4 rely on sequential decoding. All sequential decoding algorithms operate in a similar way: in each step they recursively generate a decoding VLC tree $\mathcal{U}$ and also recursively compute the cumulative metric $\Lambda_t(\mathbf{u})$ for the new tree according to (4.43).

Stack Algorithm

The stack algorithm derives its name from the fact that its decoding process is based on an ordered list of partially extended paths and corresponding metrics. The tree $\mathcal{U}_i$ is extended in each step by

appending ± 1 to the code word with the highest metric $\hat{\mathbf{u}}_i$, i.e., for each $i = 1, 2, \ldots$

$$\begin{aligned} \Lambda(\{\hat{\mathbf{u}}_{i-1}, \pm 1\}) &= \Lambda(\hat{\mathbf{u}}_{i-1}) + \lambda_{\text{VLC}}(\boldsymbol{\psi}_{l\{\hat{\mathbf{u}}_{i-1}\}+1}, \mathbf{x}(\pi_{l\{\hat{\mathbf{u}}_{i-1}\}}(\hat{\mathbf{u}}_{i-1}), \pm 1), L_{l\{\hat{\mathbf{u}}_{i-1}\}+1}, \pm 1), \\ \mathcal{U}_i &= (\mathcal{U}_{i-1} \cap \hat{\mathbf{u}}_{i-1}) \cup \{\hat{\mathbf{u}}_{i-1}, \pm 1\}, \\ \hat{\mathbf{u}}_i &= \arg\max_{\mathbf{u} \in \mathcal{U}_i} \Lambda(\mathbf{u}), \end{aligned} \tag{4.46}$$

with initial tree $\mathcal{U}_0 = \emptyset$, $\hat{\mathbf{u}}_0 = \{\}$, and $\Lambda(\hat{\mathbf{u}}_0) = 0$. After this initialization, the stack decoder carries out repeatedly the operation in (4.46). The operations are accomplished by removing the top path $\hat{\mathbf{u}}_{i-1}$ of an ordered list, computing the metric of the two new code words, inserting the two new code words $\{\hat{\mathbf{u}}_{i-1}, \pm 1\}$ into the ordered list according to the derived metrics $\Lambda(\{\hat{\mathbf{u}}_{i-1}, \pm 1\})$ such that finally the paths with the highest metric, $\hat{\mathbf{u}}_i$, appears at the top of the list. Note that the repeated operation according to (4.46) is very similar to the *add–compare–select* operation of the Viterbi algorithm consisting of two additions and comparison operations. For convolutional codes, this procedure repeats until the top path has reached the end of the code tree unless the decoding is stopped by an external interrupt before. For details on an efficient implementation using a linked list and efficient sorting algorithms we refer to [Don99]. In a practical system the stack size is naturally limited. Hence, when all stack entries are already filled, the paths with the lowest metric value that cannot be accommodated in the stack are simply dropped and, therefore, lost in the decoding process.

Fano Algorithm

For good performance, the Stack decoder requires a significant amount of storage capacity to keep track of the ordered list of paths. In contrast, the Fano algorithm [Fan63b] considers only one distinct path $\hat{\mathbf{u}}$ at any time instant such that basically only memory for a single path is required. Therefore, it is sufficient to specify the position of the Fano decoder by a single index $t = l\{\hat{\mathbf{u}}\}$. The Fano decoder explores a sequence of nodes in the code tree, moving from the current node $\mathcal{N}_t \triangleq \mathcal{N}(\hat{\mathbf{u}})$ to one of its successors $\mathcal{N}_t^{(u)} \triangleq \mathcal{N}(\{\hat{\mathbf{u}}, u\})$ with $u = \pm 1$ or its direct predecessor $\mathcal{N}_{t-1} \triangleq \mathcal{N}(\hat{\mathbf{u}}_{(0:l\{\hat{u}\}-1]})$, but never jumps. The decision, whether the decoder moves forward or backward in the code tree is based on the comparison of the metric of the current best path, $\Lambda_t \triangleq \Lambda(\hat{\mathbf{u}})$ with a threshold value $\mathcal{T}_{\text{Fan}}$. Essentially, all paths $\mathbf{u}$ with metric $\Lambda(\mathbf{u})$ better than $\mathcal{T}_{\text{Fan}}$, i.e., $\Lambda(\mathbf{u}) > \mathcal{T}_{\text{Fan}}$, have to be examined. Therefore, in order to minimize the number of the paths necessary to be examined, the threshold is updated in discrete steps $\pm\Delta_{\text{Fan}}$.

Figure 4.10 sketches a flow chart of the Fano algorithm[14]. After initialization ($\hat{\mathbf{u}} = \{\}$, $t = 0$ $\Lambda_t = 0$, $\mathcal{T}_{\text{Fan}} = 0$) or during the decoding process the decoder is in a certain node $\mathcal{N}_t$. The metrics $\Lambda_t^{(u)}$ with $u = \pm 1$ are computed according to (4.46) identically as for the Stack algorithm, and u^* is chosen as $u^* = \arg\max_{u=\pm 1} \Lambda_t^{(u)}$. If $\Lambda_t^{(u^*)} \geq \mathcal{T}_{\text{Fan}}$ the decoder operates just as the stack decoder, namely moves forward by updating the decoded vector as $\hat{\mathbf{u}} = \{\hat{\mathbf{u}}, u^*\}$. Otherwise, it is checked if continuing with the other information bit $-u^*$ or even moving backward is appropriate, or if the threshold has to be lowered. Backward moving, i.e., $\hat{\mathbf{u}} = \hat{\mathbf{u}}_{(0:t-1]}$, is selected if the threshold $\mathcal{T}_{\text{Fan}}$ is lowered that much that it is below the metric of the previous node, Λ_{t-1}. Backward moving is continued as long as on the way backward all paths forward have been checked and the metric of the previous node is greater equal than the current threshold $\mathcal{T}_{\text{Fan}}$. If on the way backwards other forward directions have not yet been checked, forward moving into this direction is attempted. If threshold loosening is chosen, i.e., $\mathcal{T}_{\text{Fan}} = \mathcal{T}_{\text{Fan}} - \Delta_{\text{Fan}}$, a regular forward operation according

[14] We ignore for the moment the shaded boxes which describe the modifications for computing path reliability as discussed in section 4.4.

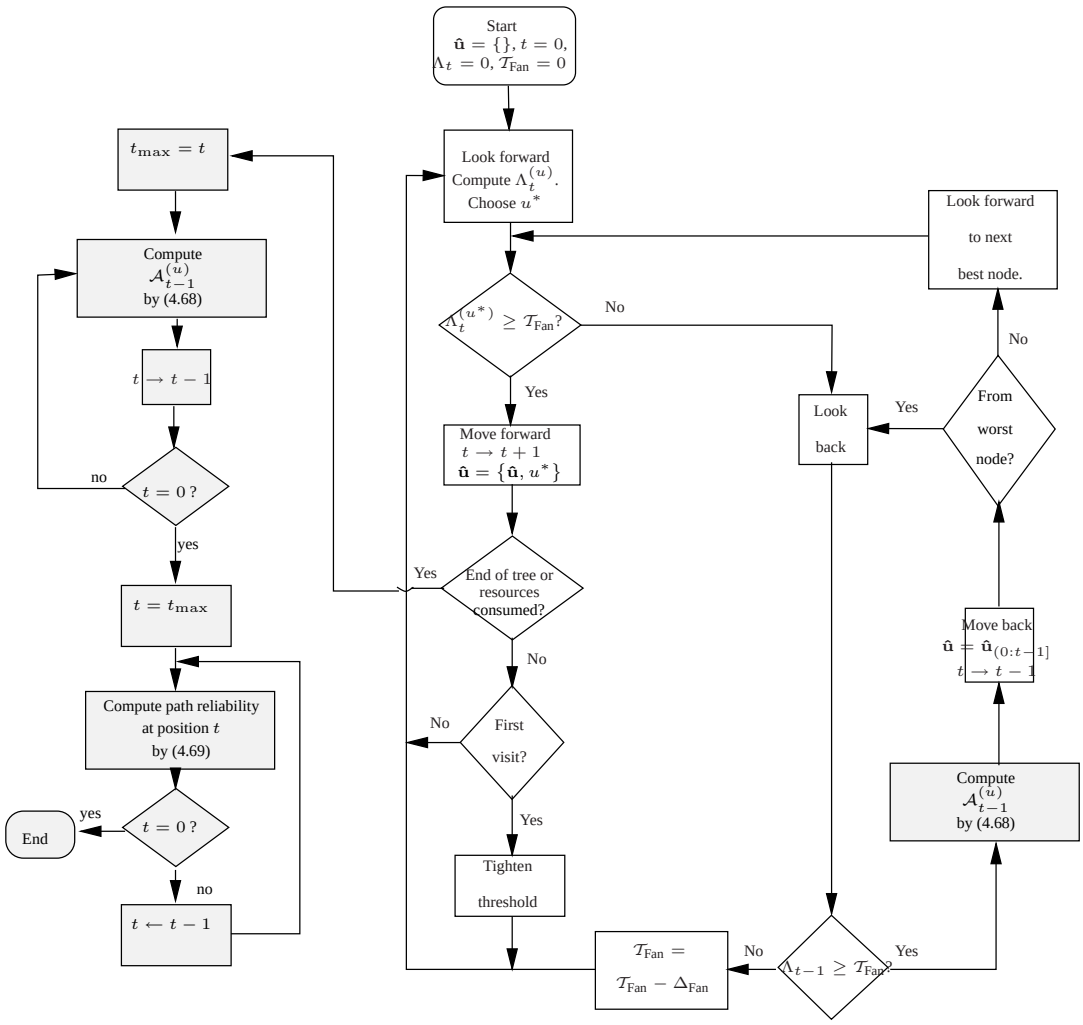


Figure 4.10: Flow chart of the Fano algorithm.

to (4.46) with metric computation and selection of best paths in forward direction based on the current values $\hat{\mathbf{u}}$ and $\mathcal{T}_{\text{Fan}}$ is invoked. Again, only this *add–compare–select* operation is of major interest for the computational costs of a Fano decoder, all other operations basically only result in simple comparisons and threshold adaptations. Again, on the way forward it is checked if the end of the code tree is reached and only *add–compare–select* operations according to (4.46) are counted. The residual operations necessary in the decoding algorithm basically only consist of very few comparisons, a trade–off between storage and number of operations is possible. On the way forward, threshold tightening is applied, i.e., the threshold $\mathcal{T}_{\text{Fan}}$ is increased by the largest possible positive integer multiple of Δ_{Fan} such that $\mathcal{T}_{\text{Fan}}$ does not exceed Λ_t. However, to prevent the decoder to be stuck in a dead–lock, this threshold tightening is only allowed if this node $\mathcal{N}_t$ is visited the first time [LJ83]. This event is detected by comparing the metric of the current path, Λ_t, with the actual threshold, $\mathcal{T}_{\text{Fan}}$ as $\Lambda_t < \mathcal{T}_{\text{Fan}} + \Delta_{\text{Fan}}$. To detect a first–time visit, Δ_{Fan} must be selected such that it is greater than the maximum increment, i.e.,

$$\Delta_{\text{Fan}} > \max_{\boldsymbol{\psi}} \lambda_{\text{VLC}}(\boldsymbol{\psi}, \mathbf{1}, |L|, 1) = c\log(2) - \log\left(1 + \exp\left(-|L|\right)\right).$$

For $1/n$–codes and equiprobable input symbols with $|L| = 0$ we obtain $\Delta_{\text{Fan}} > (c-1)\log(2)$, whereby equality is sufficient in practical systems. Therefore, we define the minimum increment

as $\Delta_{\text{Fan,min}} = (c-1)\log(2)$. For more details on the Fano algorithm we refer to [LJ83, AM91], an optimized implementation of the Fano algorithm is presented in [Don99].

Computational Complexity

In contrast to maximum–likelihood Viterbi decoding with a constant number of *add–compare–select* operation $N_{\text{op}} = 2^{\mu}$ per information bit, for sequential decoding algorithms the number of computations to decode a certain information word $\mathbf{u}$ is a random variable.

In [LJ83, Sav66, JB67, Jel69, VO79], it shown that the average[15] number of path extensions per information bit, N_{op}, equivalent to the *add–compare–select* operation in (4.46), is approximately Pareto distributed for sufficiently high number of operations[16], i.e.,

$$\Pr\{N_{\text{op}} > \eta\} \approx A_{\text{seq}}\eta^{-\rho}, \quad 0 < \rho < \infty, \tag{4.47}$$

where A_{seq} is a constant depending on the applied sequential decoding algorithm, and the Pareto exponent ρ is related to the Gallager function according to (4.4) and the code rate r as $\rho \triangleq E_0(\rho, h)/r$. An important observation from (4.47) is that the probability distribution $\Pr\{N_{\text{op}} > \eta\}$ is independent of the code memory μ. Hence, the computational behavior in terms of *add–compare–select* operations is independent of the code memory. The expected number of operations results in

$$\mathbb{E}\{N_{\text{op}}\} = A_{\text{seq}} \lim_{\eta\to\infty} \frac{\rho}{1-\rho}\left(\eta^{1-\rho} - 1\right), \tag{4.48}$$

and for the expected number to be finite, i.e., $\mathbb{E}\{N_{\text{op}}\} < \infty$, we require that $\rho > 1$. From (4.4) and the definition of ρ, it is obvious that $\rho \to \infty$ as $r \to 0$ and $\rho \to 0$ as $r \to R_I$. For a bounded mean of the number of *add–compare–select* operations according to (4.48) $\rho > 1$ and therefore a code rate $r < E_0(1, h)$ is necessary. Note that according to (4.5), the left hand side is equivalent to the cutoff rate R_0 which therefore provides a practical bound for convolutional codes with sequential decoding. For code rates $R_0 \leq r \leq R_C$, sequential decoding becomes impractical. The asymptotic behavior of probability distribution for the number of operations, $\Pr\{N_{\text{op}} > \eta\}$, for different ρ as well as the practical decoding range is shown in Figure 4.11. In [VO79] it is shown, that R_0 is not only a practical bound for sequential decoding, but also for many other decoding schemes.

We will verify this workload behavior for practical codes. For this purpose, the probability distribution for the number of operations, $\Pr\{N_{\text{op}} > \eta\}$, has been measured for a specific systematic feedforward $1/2$–convolutional code with $\mu = 96$ and information block length $k = 512$ and appropriate termination. The code rate r is computed according to (4.27) and simulations are performed at the SNR, such that the cutoff rate $R_0(h)$ according to (4.5) results in the code rate r, i.e., $R_0(h) = r$ and $h = -\log\left(2^{1-r} - 1\right) \equiv -1.06$ dB. Figure 4.12 shows the the probability distribution $\Pr\{N_{\text{op}} > \eta\}$ averaged over all information bit positions for the stack algorithm as well as for the Fano algorithm with different threshold increments Δ_{Fan}. The best computational distribution is obtained by the stack algorithm. For the Fano algorithms, a threshold increment Δ_{Fan} of about five times the minimum threshold increment $\Delta_{\text{Fan,min}}$ provides the best computational distribution. Furthermore, it is obvious that for sufficiently large η the distribution decreases exponentially with $\rho = 1$ meaning that practical codes with sufficiently large memory behave as predicted by random coding arguments according to (4.47). In real–time applications, the computational resources per information bit, N_{op}, in decoders are in general limited to some $N_{\text{op,max}}$ if a certain bit-rate has to be supported. Therefore, for a given number of maximum operations $N_{\text{op,max}}$, it might happen with

[15] this means that averaging is performed over the ensemble of all convolutional codes.

[16] more precisely, it is shown that $\lim_{\eta\to\infty}\left(\Pr\{N_{\text{op}} > \eta\} \cdot \eta^{-\rho}\right) = A_{\text{seq}}$.

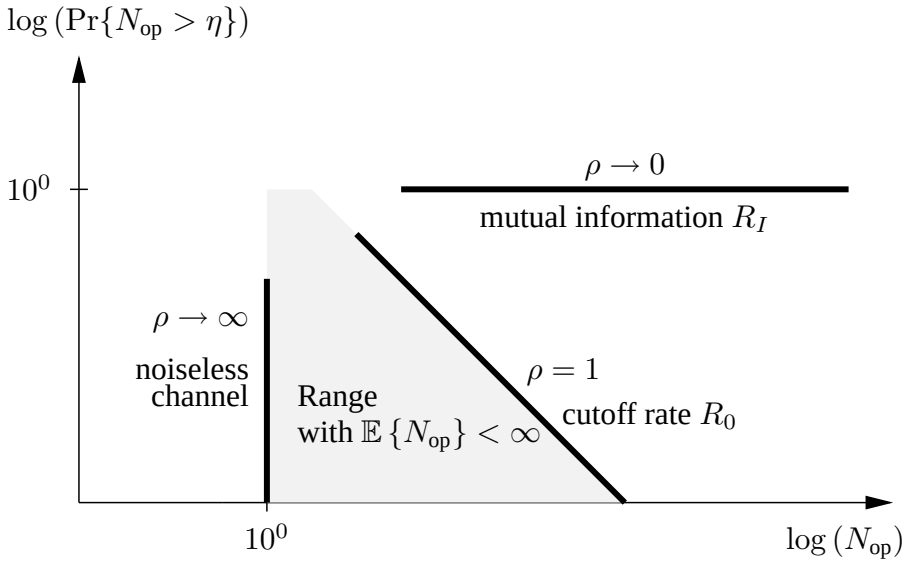


Figure 4.11: Asymptotic behavior of probability distribution for the number of operations, $\Pr\{N_{\text{op}} > \eta\}$, for different ρ together with practical decoding range.

probability $\Pr\{N_{\text{op}} > N_{\text{op,max}}\}$ for sequential decoders that the decoding process can not be finalized in time. Commonly, this information block is then declared as an erasure. From figure 4.11 for example, it can be observed that if we restrict the total amount of operations per information bit to $N_{\text{op,max}} = 64$, the erasure probability ε for the stack decoder is about 0.022, whereas for the optimum Fano the erasure probability is about 0.038.

Performance Comparison of Codes and Decoding Algorithms

Convolutional codes with moderate memory in the range of $\mu = 4, \ldots, 8$ are widely used in mobile communications systems, usually in combination with maximum–likelihood decoding based on the Viterbi algorithm. Sequential decoding algorithms have been ignored in practical systems mainly as the Viterbi algorithm can provide better performance at the reasonable complexity for these moderate–memory codes. In addition, non–systematic codes are used due to their superior performance compared to systematic feedforward codes. Figure 4.13(a) for example shows the block error probability for systematic feedforward and non–systematic convolutional codes, both with memory $\mu = 5$ and code rate $r = 1/2$ over the SNR per information bit, h/r_{term}. This presentation has been selected to compensate small differences in the termination overhead for the information block of size $k = 512$. The nonsystematic code outperforms significantly the systematic code, regardless of the decoding algorithms. In terms of decoding algorithms, the Viterbi algorithm outperforms the stack algorithm which itself is better than the Fano algorithms. Both, stack and Fano algorithms operate with $N_{\text{op,max}} = 2^5$ such that it compares to the Viterbi add–compare-select operations necessary for decoding a $\mu = 5$ code. Whereas in case of maximum–likelihood decoding the decoding process is always terminated, in case of sequential decoding an erasure might occur due the excess of the necessary operation. The presentation treats both errors equivalently by summing the error rate, P_B, and the erasure rate, ε, in the presentation in figure 4.13(a). Whereas ε is zero in case of maximum–likelihood decoding, for sequential decoding the percentage of detected errors, $\varepsilon/(P_B + \varepsilon)$, is not necessarily zero as shown in figure 4.13(b). For the moderate memory codes with $\mu = 5$, the detection probability is very high for low SNR,

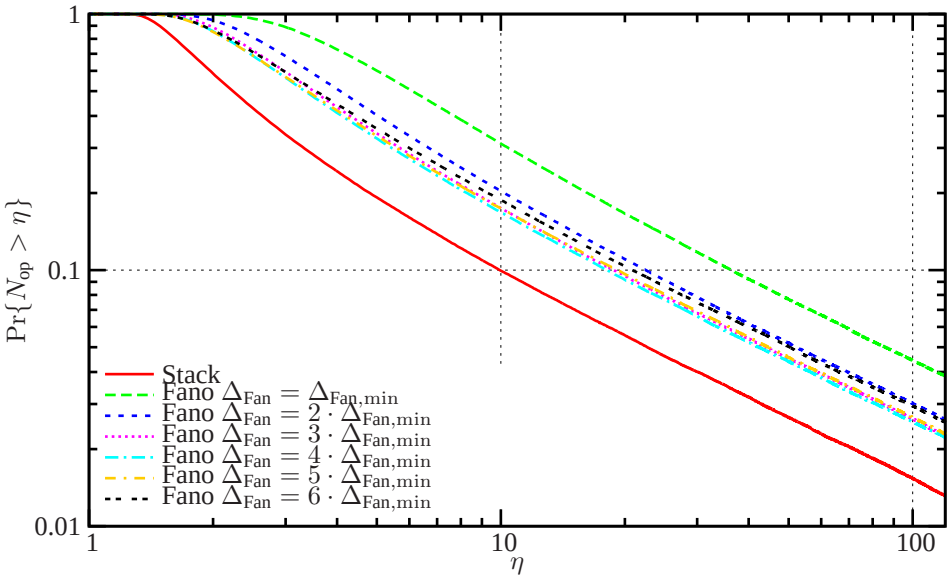


Figure 4.12: Probability distribution $\Pr\{N_{\text{op}} > \eta\}$ averaged over all information bit positions for the stack algorithm as well as for the Fano algorithm with different threshold increments Δ_{Fan}.

but almost vanishes for higher SNR. Also shown in figure 4.13(a) is the bound for asymptotically optimal rate $1/2$–codes operating at the cutoff rate R_0 and the mutual information R_I. All memory $\mu = 5$ codes, regardless of the applied decoding, have non–negligible error rates at this bound.

The whole situation changes when increasing the memory of the convolutional codes. Figure 4.13(a) and (b) also show the performance of rate $1/2$ convolutional codes with memory $\mu = 23$, 31, and 96. All three codes have systematic feedforward generators, but cannot be decoded with maximum–likelihood decoders due to the exponential complexity with the memory. However, sequential decoding of these higher memory with $N_{\text{op,max}} = 2^5$ outperforms the best memory–$\mu = 5$ performance. In any case, the stack algorithm performs slightly better than the Fano algorithm. With increasing memory, the performance in terms of error rates increases, but somehow saturates from $\mu = 31$ to $\mu = 96$. However, when looking at figure 4.13(b), the advantage of the memory $\mu = 96$ code is immediately obvious as the percentage of undetected errors is 0 for all channel states whereas for the $\mu = 31$ code as well as the $\mu = 23$ code the decoder still decides on wrong paths, the free distance of these code is not sufficient to guarantee virtually error–free performance. In addition, the memory $\mu = 96$ code has block erasure rate below 1% for SNR above the cutoff rate. Still, if the channel state is below the one proposed by the cutoff rate, the error rate increases relatively fast. Therefore, we have shown that this $\mu = 96$ code with code rate $1/2$ approximately performs like an asymptotically optimal code operating at the cutoff rate as defined in (4.8).

In the next experiment we are interested in the extension of this code performance to obtain practical rate–compatible convolutional codes which also perform asymptotically optimal at the cutoff rate. For this purpose a memory $\mu = 96$ code with code rate $1/3$ has been used to generate a family of rates. Although the literature provide many good high–memory codes for sequential decoding, good puncturing tables for higher memories are not known, only memories up to $\mu = 6$

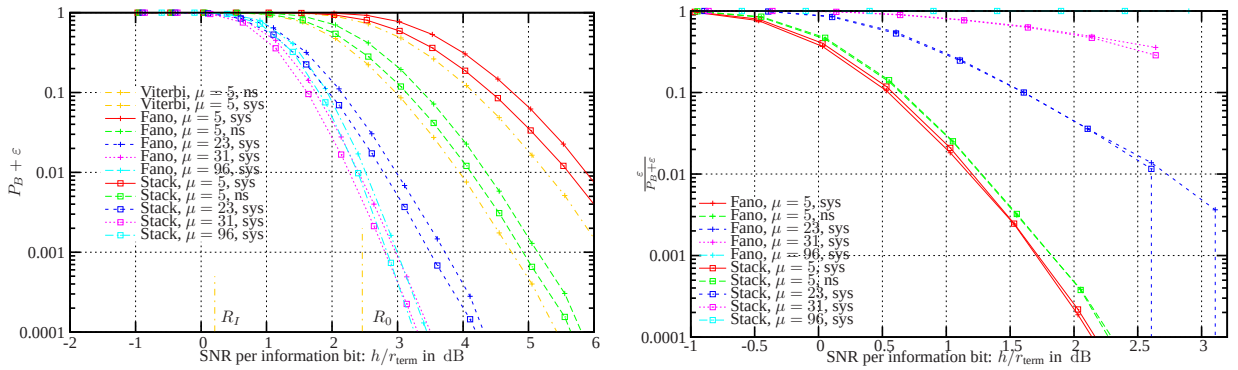


Figure 4.13: Detected and undetected block error rates for convolutional codes with different memory μ, systematic and non–systematic generator matrices, and different decoding strategies over the SNR per information bit: (a) Sum of detected and undetected block error probability, (b) Percentage of detected errors.

as for example provided in [Hag88]. The criteria of finding good codes is not completely obvious[17]. Therefore, we have decided to use regular puncturing matrices for high–memory convolutional codes, in our case exclusively memory $\mu = 96$ codes. *Regular* means, that in case of adding or removing a bit from the puncturing table to obtain a lower or higher rate, respectively, the position is chosen such that the distance between two non–punctured positions is minimized. In case of ambiguity, the position is chosen randomly. Regular puncturing can be well justified as long as the puncturing period ω_p is significantly smaller than the memory of the code, i.e., $\omega_p \ll \mu$ [Den98].

Figure 4.14 shows the performance in terms of erasure rate ε over SNR h of a rate–compatible punctured convolutional code with memory $\mu = 96$ and mother code rate $1/3$. The information block is again fixed to $k = 512$. Also shown for each code is the SNR necessary to operate at the cutoff rate assuming rate r (long line) and r_{term} (short line). It is observed that these punctured convolutional codes provide different error protection for different channel states. It is also observed that the undetected error probability P_B is virtually zero for all code rates greater equal $8/10$. In addition, one can see that the performance of the punctured codes is very similar to the results in Figure 4.13 for the Fano decoder with memory 96. The error rate at an SNR corresponding to the cutoff rate r is about $2 - 3\%$, at the effective code rate r_{term} it is higher at about 10%. Again, if the channel state is below the one proposed by the cutoff rate, the error rate increases relatively fast. Therefore, we can conclude that also punctured convolutional codes with high memory such as $\mu = 96$ and code rates below $r = 0.8$ performs like an asymptotically optimal code operating at the cutoff rate as defined in (4.8). We will use this performance approximation in the course of this work to simplify rate allocation procedures.

[17] Good sequential codes are found based on good distance profiles, i.e., on a path forward the distance increases as fast as possible. However, in case of puncturing one has to take into consideration the code properties for different starting points in the puncturing table. For the criteria according to (4.37) the error probability is the measure of interest and averaging over all positions in the puncturing matrix is valid. However, in case of sequential decoding, one has to relate the specific code properties to a distribution of the computational effort. There also exist bounds that relate parameters of specific codes to the performance with sequential decoding [CC78b]. However, neither the computation of these bounds is trivial nor averaging over all starting points in the puncturing matrix does necessarily provide an appropriate measure.

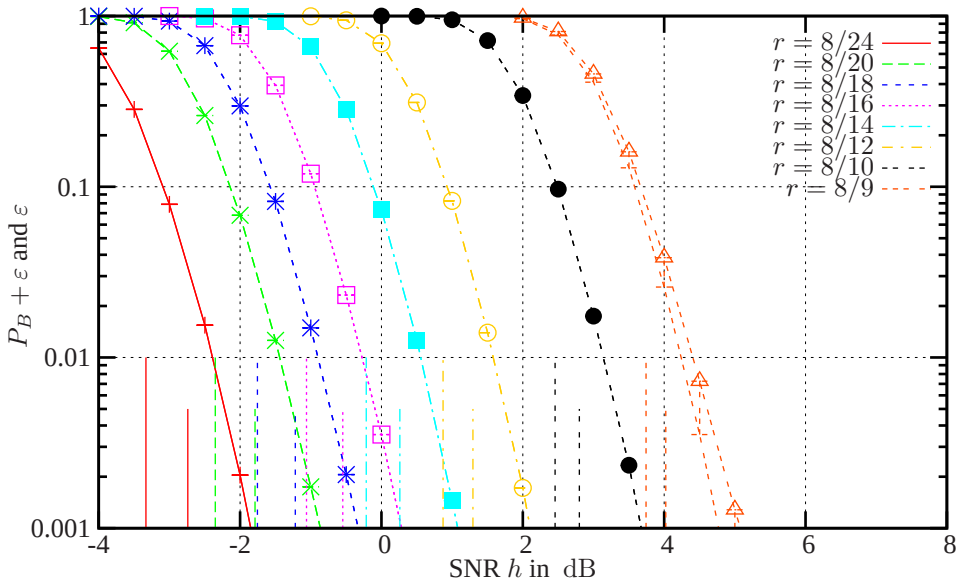


Figure 4.14: Total error rate $P_B + \varepsilon$ and detected error rate ε over SNR h of a rate–compatible punctured convolutional code with memory $\mu = 96$, mother code rate $1/3$, information block size $k = 512$ and different code rates r generated by puncturing; also shown for each code the SNR necessary to operate at the cutoff rate assuming rate r (longer line) and r_{term} (short line).

4.3 Convolutional Codes in Systems with Fading

4.3.1 System Design and Radio Access for Block Fading Channels

Convolutional codes are widely used in nowadays mobile systems. We have seen that the outage probability especially for block–fading channels can be decreased significantly at least theoretically by the use of more than one transmission slot F when transmitting over a block–fading as long as the delay is acceptable. However, the bounds do not provide any guidelines on how to distribute the channel symbols n among the F transmission slots. For good performance, neighboring code symbols of convolutional codes should experience some diversity, the trellis structure of convolutional codes wants to avoid that code bits contributing to pathes with small distances observe the same fading realization [Kno97]. In order to avoid this, the channel code word $\mathbf{n}$ is not transmitted directly over the channel but rather distributed over the F radio slots by a so–called interleaver.

We have developed an optimized combined puncturing and interleaving scheme for low-memory convolutional codes when applied over block-fading and erasure channels [Ohl98]. This was recognized [HSX00] in the standardization process for the channel coding of the Adaptive Multi–Rate (AMR) when used in GSM, that the mapping can be optimized for block–fading channels. The basic idea is that the interleaving of radio slots is optimized such that by the erasure of a whole slot, the remaining code is still as good as possible. For further details we refer to [Ohl98] and [HSX00].

However, in this work we focus on high-memory codes and in this case the optimized puncturing is neither of advantage nor can the optimization be applied in a straight–forward manner. A large variety of rates are supported by rate-compatible puncturing. In a punctured convolutional codes for each group of ω_p information bits only those code bits of the original encoder are transmitted

that correspond to a 1 in the $c \times \omega_p$ binary puncturing matrix $\mathcal{P}_i$ with $i = 0, \ldots, (c-1)\omega_p$ [Hag88]. $\mathcal{P}_i$ contains $n_i \triangleq \omega_p + i$ 1s such that the rate of the code is $r = \omega_p / n_i$. With the definition of appropriate puncturing matrices $\mathcal{P}_i$, $i = 1, \ldots, (c-1)\omega_p$ in complete $\omega_p(c-1)$ different channel coding rates are available by this punctured code. Note that $i = 0$ corresponds to uncoded transmission as we use systematic CCs. We define the set of available code rates as $\mathcal{S}_r$.

In order to incorporate sufficient interleaving in the system the channel code word $\mathbf{z}$ is not transmitted directly over the channel but rather distributed over several radio slots. $\Pi(\cdot, \mathcal{P})$ defines the mapping of index pair (i,t), with $t = 1, \ldots, b$ and $i = 1, \ldots, c$, either to the radio slot index pair (f,v), with $f = 1, \ldots, F$ and $v = 1, \ldots, N_v$, or to a punctured code bit depending on the puncturing matrix $\mathcal{P}$. With this mapping, we define the LLR corresponding to the transmitted channel symbol $z_t^{(i)}$ as

$$\psi_t^{(i)} \triangleq \begin{cases} \psi(y_{f,v}) = 4\alpha_f \frac{E_s}{N_0} y_{f,v} & \text{if } \mathcal{P}(i, t \mod \omega_p) = 1 \\ 0 & \text{if } \mathcal{P}(i, t \mod \omega_p) = 0 \end{cases} . \tag{4.49}$$

In the remainder of this work we assume a simple mapping from (i,t) to (f,v) as follows. We introduce a mapping Π_1 from (i,t) to a one-dimensional sequence $j = 1, \ldots, N_v F$ such that $\forall t_1 = 1, \ldots, b, \forall i_1 = 1, \ldots, c$ with $\mathcal{P}(i_1, t_1) = 1$ and $\forall t_2 = 1, \ldots, b, \forall i_2 = 1, \ldots, c$ with $\mathcal{P}(i_2, t_2) = 1$ we have $\Pi_1(i_1, t_1) < \Pi_1(i_2, t_2)$ if $t_1 < t_2$ or if $t_1 = t_2$ and $i_1 < i_2$. If $\mathcal{P}(i,t) = 0$ the code symbol $z_t^{(i)}$ is punctured. The mapping from $\Pi_1(\cdot)$ to the symbol pair (f,v) is defined as $f = \Pi_1(i,t) \mod F$ and $v = \lceil \Pi_1(i,t)/F \rceil$.

Figure 4.15 summarizes the channel coding and radio access scheme considered in the remainder of the work. The information message $\mathbf{u}$ of length k is encoded by the $1/c$ convolutional encoder to obtain channel code vector $\mathbf{z}$. Then, by appropriate puncturing and interleaving with $\Pi(\cdot, \mathcal{P})$ as well as by proper signal mapping, $\mathbf{z}$ is mapped on the transmitted signal $\mathbf{x}$, which is spread and transmitted over the F radio slots. The received signal $\mathbf{y}$ is de-interleaved, de-punctured and processed such that we obtain the LLR of the received signal $\boldsymbol{\psi}$, which is passed to the channel decoder. In case of ARQ type 2 schemes, exactly the same procedure is applied except that transmission can be stopped, if the message is correctly received before all F radio slots are used.

4.3.2 Outage Probabilities for FEC

Convolutional codes are used in wireless systems for simple forward error correction. Obviously, we cannot provide a comprehensive discussion on the performance of all different codes. However, we discuss at least the influence of different code and system parameters for a specific example. Figure 4.16 shows the combined block error and erasure rate $P_B + \varepsilon$ over the SNR per information bit, h/r_{term} for different convolutional codes with memory $\mu = 4$ and $\mu = 96$, mother code rate $1/2$, information block size $k = 512$, different interleaving schemes and different number of independent radio slots, F. In addition, also for each number of slots the outage probability for asymptotically optimal codes operating at the cutoff rate R_I is shown.

For any code in use the performance obviously increases with increasing number of radio slots, diversity gains are apparent. All codes result in similar slope as the SNR grows which mainly depends on the diversity degree, i.e., on the number of slots F, not on the code itself. In any case, the $\mu = 4$ codes perform worse than the cutoff rate bound. For $F = 1$ and $F = 2$ the optimized interleaving is identical to the regular interleaving. Only for $F = 4$ and $F = 8$ the optimized interleaving according to [Ohl98] and [HSX00] provides gains between 0.2 and 0.5 dB. Although the gains are not impressive, it is worth to note that they come for free. However, one can also observe that with the use of higher memory codes, the observed gains for the AWGN

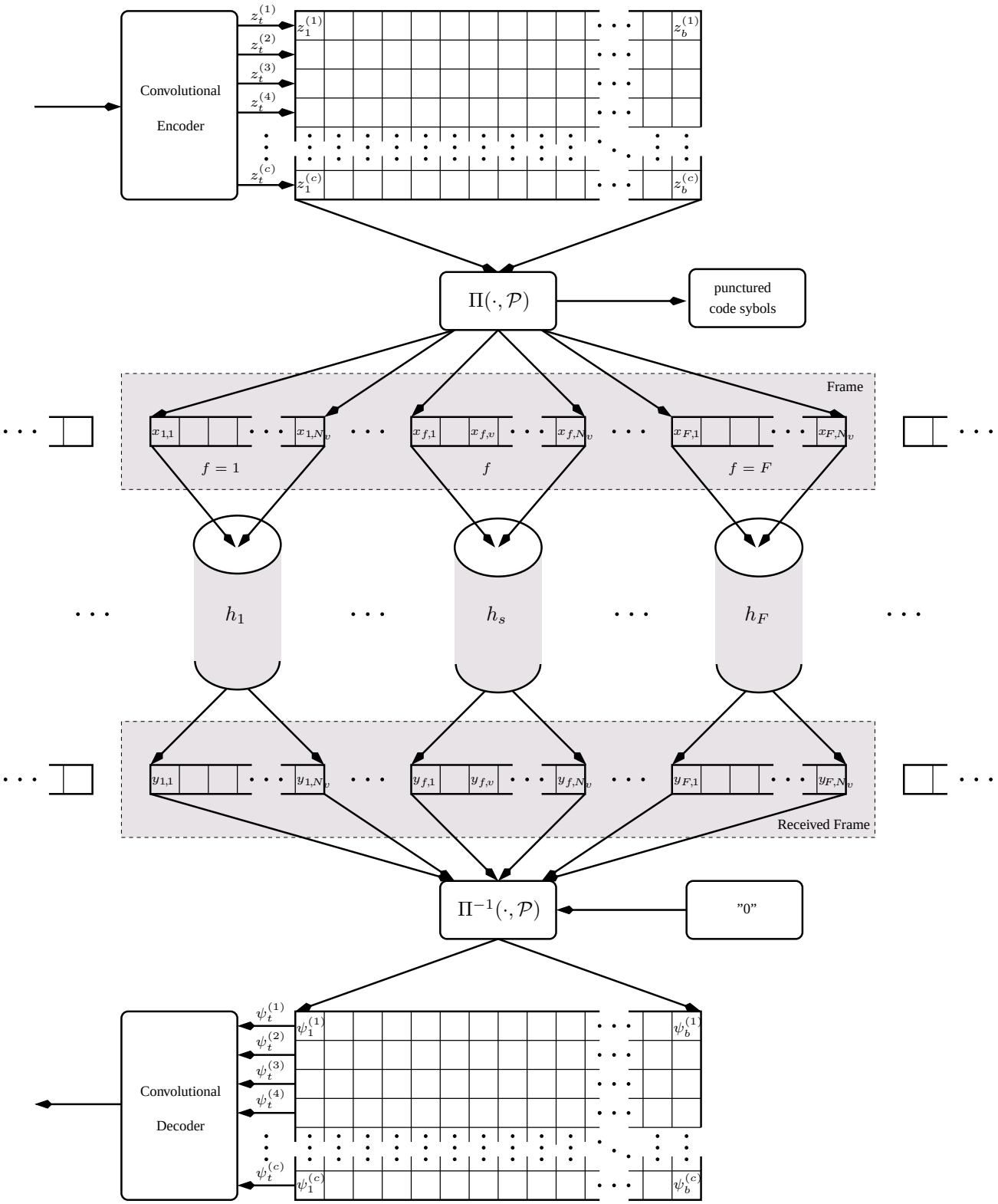


Figure 4.15: Radio access and channel coding scheme.

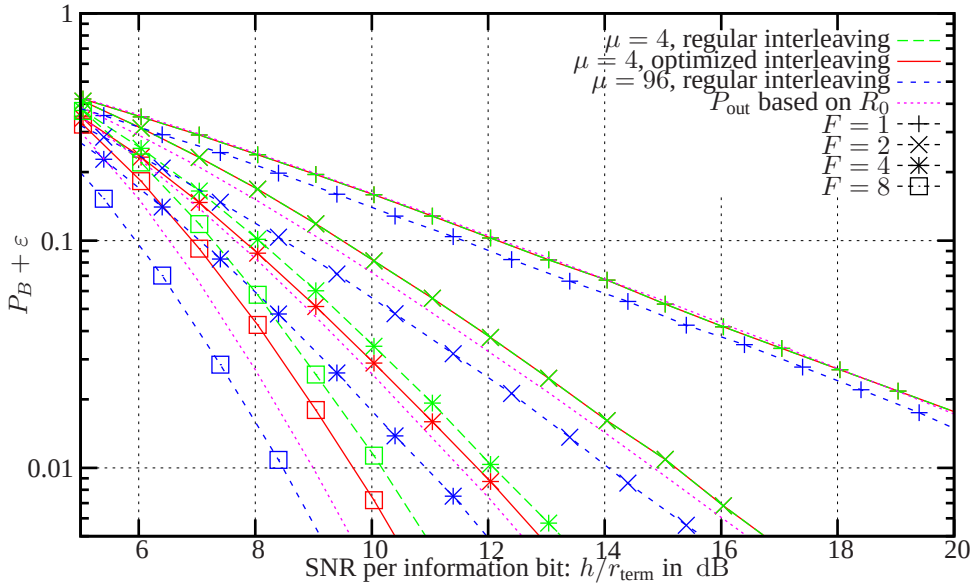


Figure 4.16: Combined block error and erasure rate $P_B + \varepsilon$ over the SNR per information bit, h/r_{term} for different convolutional codes with memory $\mu = 4$ and $\mu = 96$, mother code rate $1/2$, information block size $k = 512$, different interleaving schemes and different number of independent radio slots, F; also shown for each number of slots is the outage probability for asymptotically optimal codes operating at the cutoff rate R_I.

channel are also present when transmitting over a block–fading AWGN channel. For the high memory codes, only regular interleaving has been applied. The high memory codes with $\mu = 96$ and Fano decoding with $N_{\text{op}} = 16$ operations comparable to Viterbi decoding of memory $\mu = 4$ codes results in performance gains of $0.5 - 1.5$ dB depending on the number of slots F. In any case, the performance of the higher-memory codes outperforms the cutoff rate bound. Due to the good performance and the manageable complexity of high–memory codes and sequential decoding algorithms we will concentrate on these codes in the following and will use asymptotically optimal codes at the cutoff rate for performance estimations.

4.4 Regressive UEP and the Far End Error Decoder

4.4.1 System Overview

Motivation and Problem Formulation

In subsection 3.1.3 the extension from scalable source coding to progressive source coding has been presented as a major breakthrough to simplify the design of multimedia communication systems. In this case, virtually each additional bit received from the progressively coded bit–stream can enhance the quality. Investigations for example in [MM00] have shown that the reconstruction quality of efficient progressive source coding schemes such as SPIHT is severely affected by residual errors. Due to the properties of progressive coding any data after the first decoding error cannot be interpreted any more. In fact, even if the data following the decoding error is correct, usually the reconstruction quality decreases due to effects like synchronization loss and wrong interpretation of the entropy coded data. Thus, it is vital in progressive coding to deliver only error–free data

to the source decoder, i.e., to *detect* and *localize* the first error in the decoded block. Hence, it is obvious that for highly efficient progressive source coding algorithms no longer the average bit error probability is the most important parameter. Instead it is essential to delay the first decoding error as far to the end of the block as possible, i.e., to maximize the error–free part of the decoded data block.

Conventional UEP schemes, e.g. in [Hag88, SZ98], apply block coding or convolutional coding with termination. In addition, as error detection is essential as indicated in [MM00] an outer error detection code – usually a CRC of length N_{CRC} – is applied to each individual level m. Hence, in conventional schemes, e.g. [SZ98] only all error–free levels until the first erroneous level have been detected are delivered to the source decoder. Low error probability as well as low detection failure probability are desired, i.e., the reliability of the delivered data should be as high as possible. This requires on the one hand codes with low error probability directly asking for codes with memory μ as high as possible. In addition, for high reliability of the delivered data, the size of the error detection code, $\mu + N_{\text{CRC}}$, should be as high as possible. Unfortunately, termination and error detection adversely affect efficiency as an overhead of $\mu + N_{\text{CRC}}$ information bits for each individual level is necessary.

Assuming that we apply the same code rate r to each level and using k/M bits for each level, the effective code rate is decreased due to this overhead as

$$\frac{r_{\text{UEP}}}{r} = \frac{1}{1 + (\mu + N_{\text{CRC}})\frac{M}{k}}.$$

i.e., for the same amount of channel symbols available only kr_{UEP}/r bits compared to k for single layer approach without termination and error detection can be transmitted.

Therefore, conventional UEP schemes with individual coding of levels can be viewed as the channel–coding pendant to scalable source coding with a limited amount of layers, the equivalence to progressive source coding in the channel coding area is not obvious. For this purpose we introduce a coding scheme referred to as *Regressive Unequal Error Protection (UEP)* which includes an UEP scheme with many different rate–scales based on convolutional codes. Thereby, rather than using termination and error detection for individual level, we apply a single convolutional code over the entire levels, support UEP by puncturing with regressive redundancy, and provide decoding algorithms which are suited for this specific encoding. We define the objective of the decoding algorithm and discuss an optimum trellis–based decoding algorithm. Finally, we present in detail a low–complexity decoding algorithm based on sequential decoding, the so–called Far End Error Decoding (FEED) algorithm.

To formalize this idea, we introduce a system concept based on punctured convolutional codes for regressive UEP together with an innovative decoding concept in a sense that the decoder

1. estimates the most likely transmitted sequence based on the channel observations and a priori information as discussed in section 4.2,

2. computes a reliability of all information subsequences of this most likely sequence where each sub–path starts at the first symbol,

3. decides based on these reliabilities the error–free part of the decoded data block.

Encoding Principle

The encoding principle basically relies on convolutional codes with UEP where the code rate increases over the information sequence block [18]. Assume that an information sequence $\mathbf{u}$ of length k bit is encoded with a rate $1/c$–convolutional code, termination is applied optionally. A regressive redundancy profile is implemented applying puncturing with puncturing period ω_p to the encoded block before transmitting it. Therefore, we divide the information sequence $\mathbf{u}$ in $M = k/\omega_p$ level with $k_m = \omega_p$. We define a channel code vector $\mathbf{n} = \{n_1, n_2, \ldots, n_M\}$ which denotes the amount of channel symbols assigned to each level. The code rate of level m is therefore defined as

$$r_m = \omega_p / n_m.$$

For notational convenience we define the channel code vector up to level m as

$$\mathbf{n}_m \triangleq \{n_1, n_2, \ldots, n_m\}$$

and the sum over all components of vector $\mathbf{n}_m$ as

$$N_m \triangleq \sum_{i=1}^{m} n_i$$

and $N_0 \triangleq 0$.

In the following we assume that the receiver has perfect knowledge of the channel coding vector $\mathbf{n}$. After transmitting the code symbols for this information sequence $\mathbf{u}$, puncturing is reverted at the receiver by inserting 0's in the channel LLR for all positions which have been punctured at transmitter side. Therefore, we assume in the following decoding process that we operate on $1/c$ convolutional codes.

Decoding Principle

A crucial role in the determination of error–free part of the data block plays the reliability $\mathcal{R}$ of a subsequence $\mathbf{u}_{(0:t]}$ of an information word $\mathbf{u}$ where $0 \leq t \leq k$. The reliability expresses the probability that this information subsequence $\mathbf{u}_{(0:t]}$ is correct, i.e., all symbols in this information subsequence are correct, taking into account the entire received sequence $\mathbf{y}$. To be more specific, we denote by $\mathcal{S}(\mathbf{u}_{(t:t']})$ the set of all information words $\mathbf{u}$ that share a common subsequence $\mathbf{u}_{(t:t']}$ with $0 \leq t < t' \leq k$, i.e., $\mathcal{S}(\mathbf{u}_{(t:t']}) \triangleq \{\mathbf{u}' \in \mathcal{S} | \forall t < \tau \leq t'\, u_\tau = {u'}_\tau\}$. The reliability is defined as the probability of this sub–path $\mathbf{u}_{(0:t]}$ given the entire received vector $\mathbf{y}$, or equivalently the channel LLR $\boldsymbol{\psi}$ for this data block, i.e.

$$\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]}) \triangleq \Pr\{\mathbf{u}_{(0:t]} | \boldsymbol{\psi}\} = \sum\nolimits_{\mathbf{u}' \in \mathcal{S}(\mathbf{u}_{(0:t]})} \Pr\{\mathbf{u}' | \boldsymbol{\psi}\}. \tag{4.50}$$

For convenience, we define the reliability vector of all possible subblocks in $\mathbf{u}$ as

$$\mathcal{R}_{\mathbf{u}} \triangleq \left\{\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:0]}), \mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:1]}), \ldots, \mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:k]})\right\}. \tag{4.51}$$

[18] Note that in general any other principle to support a decrease of the average channel information in terms of ψ might be applied to support some unequal protection. For example, the transmit power of the transmitted signals might be reduced for later symbols in the encoded block. We have investigated another approach where the average SNR towards the end of the block is achieved by multilevel coding [Sch00].

As the set $\mathcal{S}(\mathbf{u}_{(0:t']})$ is a subset of the $\mathcal{S}(\mathbf{u}_{(0:t]})$ for $t < t'$ and with the definition of the reliability according to (4.51), it is obvious that the reliability vector is monotonically increasing with decreasing subblock length, i.e.,

$$\forall t < t' \quad \mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]}) \geq \mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t']}).$$

With the provision of this reliability for the maximum–likelihood sequence $\hat{u}$, an intermediate simple interface between the channel decoder and the progressive source decoder uses this reliability vector $\mathcal{R}_{\boldsymbol{\psi}}(\hat{\mathbf{u}})$ to provide the source decoder with this a shortened information sequence $\mathbf{u}^*$ that is error-free with some specified reliability $\mathcal{R}_0$. This interface is formalized by the definition of a shortening function $\mathcal{F}_s(\mathbf{u}, \mathcal{R}_{\mathbf{u}}, \mathcal{R}_0)$ for a bit sequence $\mathbf{u}$, the corresponding reliability vector $\mathcal{R}_{\mathbf{u}}$ and the reliability threshold $\mathcal{R}_0$ as

$$\mathcal{F}_s(\mathbf{u}, \mathcal{R}_{\mathbf{u}}, \mathcal{R}_0) \triangleq \mathbf{u}_{(0:t']} \quad \text{with} \quad t' = \max_{0 \leq t \leq k} \left\{ \mathbf{u}_{(0:t]} \,\middle|\, \mathcal{R}_{\psi}(\mathbf{u}_{(0:t]}) \geq \mathcal{R}_0 \right\}, \tag{4.52}$$

i.e., the longest subblock of $\mathbf{u}$ with $\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]}) \geq \mathcal{R}_0$. The first step, the estimation of an appropriate sequence $\hat{u}$, has been discussed in detail in subsection 4.2.2 and 4.2.3. The third step, the shortening function, is trivial and completely defined by (4.52). This leaves the computation of the path reliability. We will in the following focus exclusively on $1/c$–convolutional codes, for general discussion of the detection of a Markov source we refer to [WSH01]. We present two algorithms to compute the reliability, one based on a trellis representation equivalently to the BCJR algorithm [BCJR74] and one based on sequential decoding algorithms. For the latter we discuss extensions of both prominent sequential decoding algorithms, the stack algorithm and the Fano algorithm.

4.4.2 Optimum Trellis-Based Solution

The computation of the path reliability $\mathcal{R}_{\psi}(\mathbf{u}_{(0:t]})$ for any t according to (4.50) is closely related to symbol–by–symbol APP algorithm. The APP algorithm is also referred to as BCJR algorithm acknowledging the authors having proposed this algorithms in [BCJR74] for decoding of convolutional codes. In [BCJR74] the algorithm is proposed for estimating the APP for a Markov source observed through a Discrete Memoryless Channel (DMC). This general problem is then specialized to the symbol–by–symbol APP estimation by trellis–based decoding of convolutional codes. In [WSH01] we have also presented the generalized algorithm for computing the path reliability of a Markov source. In here, we exclusively concentrate on the computation for convolutional codes.

Consider an information sequence $\mathbf{u}$, with $\mathbf{u} = (u_1, \ldots, u_k)$ and the corresponding code word, encoded with a convolutional code with mother code rate $1/c$ denoted as $\mathbf{x}$ and length N_x. Assume that the decoder operates on the trellis as defined in subsection 4.2.2 with N_π states, state transitions $\pi_t(\mathbf{u}) = \pi_s(\pi_{t-1}(\mathbf{u}), u_t)$, corresponding output tuple $\mathbf{x}_t(\mathbf{u}) \triangleq \mathbf{x}(\pi_{t-1}(\mathbf{u}), u_t)$, and state transitions in the reverse direction as $\pi_t(\mathbf{u}) = \pi_p(\pi_{t+1}(\mathbf{u}), u_{t+1})$. Further assume that encoding starts in π_0, generates a code sequence $\mathbf{x}$ and ends in a single state π_{term} in case of terminated code sequences, or in a set of states for unterminated encoding. In any case we denote the information bit length as k, but in case of terminated sequences, the information word includes μ known information bits for termination.

The symbols x are transmitted over a discrete memoryless channel with transition probability $\Pr\{Y = y | X = x, H = h\}$ with h representing the channel state. Therefore, given an information subsequence $\mathbf{u}_{(0:t]}$ and a corresponding encoded sequence $\mathbf{x}_{(0:t]} = \{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$ with $\mathbf{x}_t \equiv \mathbf{x}_t(\mathbf{u})$, the DMC produces an output sequence $\mathbf{y}_{(0:t]} = \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$ based on the transmission of each

single symbol $x_{t,i}$ to obtain $y_{t,i}$. Note that the received symbols $y_{t,i}$ can directly be represented by the channel LLR $\boldsymbol{\psi}$ according to (2.48). Puncturing might be applied, but can be compensated by the insertion zeros for the channel LLR at appropriate positions and we restrict ourselves to $1/c$ convolutional codes. The goal of the reliability estimator is the estimation of the path reliability $\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]})$ for the decoded sub–path $\mathbf{u}_{(0:t]}, 1 \leq t \leq u$, assuming the knowledge of the entire received sequence $\boldsymbol{\psi}$.

In the following we derive a recursive algorithm for computing $\mathcal{R}(\mathbf{u})$ that is closely related to the BCJR algorithm. Similar to [BCJR74], we define the probability of trellis state π for the backward recursion as

$$\beta_t(\pi) \triangleq \Pr\{\mathbf{y}_{(t:k]} | \pi_t = \pi\} \tag{4.53}$$

and the joint probability of an observed sequence $\mathbf{y}$ and a transmitted sequence $\mathbf{x}$ as

$$\gamma(\mathbf{y}, \mathbf{x}) \triangleq \Pr\{\mathbf{y}, \mathbf{x}\}. \tag{4.54}$$

The definitions in (4.53) and (4.54) are general for any Markov source. For $1/c$ convolutional codes we replace $\mathbf{y}$ by $\boldsymbol{\psi}$ and the definition in (4.54) is related to a branch metric defined as

$$\begin{aligned} \gamma(\boldsymbol{\psi}, \mathbf{x}, L, u) &\triangleq \prod_{i=1}^{c} \Pr\{\psi_i | x_i\} \cdot \Pr\{u\} \\ &= \text{const} \cdot \exp\left(\frac{1}{2}\left(\sum\nolimits_{i=1}^{c} \psi_i \cdot y_i + L \cdot u\right)\right). \end{aligned} \tag{4.55}$$

where const is constant for all possible combinations. From [BCJR74] we obtain a backward–recursive representation for the computation of $\beta_t(\pi)$ for all $t = 1, 2, \ldots, k-1$ as

$$\beta_t(\pi(\mathbf{u}_{(0:t]})) = \sum_{u=\pm 1} \gamma(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\mathbf{u}_{(0:t]}), u), L_t, u) \cdot \beta_{t+1}(\pi_s(\pi_t(\mathbf{u}_{(0:t]}), u)) \tag{4.56}$$

with the boundary conditions $\beta_k(\pi_{\text{term}}) = 1$ and $\beta_k(\pi') = 0$ if $\pi' \neq \pi_{\text{term}}$ for terminated convolutional encoding, or $\forall_\pi \, \beta_k(\pi) = 1/N_\pi$ for unterminated encoding. Then, the following proposition allows to compute the reliability according to (4.50).

Proposition 1 *The reliability for any information sub–sequence $\mathbf{u}_{(0:t]}$, $t = 0, 1, \ldots, k$ given the entire observed channel sequence $\boldsymbol{\psi} = \boldsymbol{\psi}_{(0:k]}$ is given as*

$$\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]}) = \frac{\beta_t(\pi(\mathbf{u}_{(0:t]})) \prod_{\tau=1}^{t} \gamma(\boldsymbol{\psi}_\tau, \mathbf{x}_\tau, L_\tau, u_\tau)}{\beta_0(\pi_0)}, \tag{4.57}$$

with $\beta_t(\pi)$ according to (4.53), $\beta_0(\pi_0) = \Pr\{\mathbf{y}\}$ the probability of the received sequence, and $\gamma(\boldsymbol{\psi}, \mathbf{x}_\tau, L_\tau, u_\tau)$ according to (4.55).

Outline of proof. With (4.53) and (4.54) the joint probability of $\mathbf{u}_{(0:t]}$ and $\boldsymbol{\psi}$ can be expressed as

$$\begin{aligned} \Pr\{\mathbf{u}_{(0:t]}, \boldsymbol{\psi}\} &= \Pr\{\boldsymbol{\psi}_{(t:k]} | \pi(\mathbf{u}_{(0:t]})\} \cdot \Pr\{\mathbf{u}_{(0:t]}, \boldsymbol{\psi}_{(0:t]}\} \\ &= \beta_t(\pi(\mathbf{u}_{(0:t]})) \prod\nolimits_{\tau=1}^{t} \gamma_\tau\left(\pi(\mathbf{u}_{(0:\tau]}), \pi(\mathbf{u}_{(0:\tau-1]})\right). \end{aligned}$$

The conditional probability $\Pr\{\mathbf{u}_{(0:t]} | \boldsymbol{\psi}\} = \Pr\{\mathbf{u}_{(0:t]}, \boldsymbol{\psi}\} / \Pr\{\boldsymbol{\psi}\}$ can be derived by observing that $\Pr\{\boldsymbol{\psi}\} = \sum_{\pi=1}^{N_\pi} \Pr\{\boldsymbol{\psi}_{(0:k]}, \pi_0 = \pi\} = \beta_0(\pi_0)$ where the last equality follows from the fact

that encoding is started in the initial state π_0. Thus, using $\beta_0(\pi_0)$ obtained by k recursions of (4.56), the reliability of a specific information sub–sequence $\mathbf{u}_{(0:t]}$ is exactly the expression given in Proposition 1. ■

Therefore, we can formulate an algorithm to compute the reliability vector $\mathcal{R}_{\mathbf{u}}$ as follows:

1. Initialize the recursion: for terminated convolutional encoding, $\beta_k(\pi_{\text{term}}) = 1$ and $\beta_k(\pi') = 0$ if $\pi' \neq \pi_{\text{term}}$, and for unterminated encoding $\forall_\pi\, \beta_k(\pi) = 1/N_\pi$.
2. for all $\pi = 1, \ldots, N_\pi$ and all $t = 1, \ldots, k$, recursively compute $\beta_t(\pi)$ according to (4.56) setting the constant $\text{const} = 1$.
3. With the recursion completed, we obtain $\beta_0(\pi_0)$.
4. Then, for all $t = 0, 1, \ldots, k$ compute the path reliability $\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{u}_{(0:t]})$ according to (4.57).

In general, the reliability vector is computed for the maximum–likelihood path $\hat{\mathbf{u}}$ according to (4.31). Then, the reliability vector $\mathcal{R}_{\hat{\mathbf{u}}}$ is computed and shortening is applied according to (4.52). The major complexity in the algorithm comes from the computation of all $\beta_t(\pi)$ in step 2, i.e., the complexity is proportional to the number of states $N_\pi = 2^\mu$. Therefore, for higher memories, this algorithm becomes impracticable and reduced complexity algorithms are required and will be presented in the following.

4.4.3 Complexity-Reduced Computation of Reliability

The algorithm in subsection 4.4.2 to compute the path reliability is subject to the same principle constraints as the BCJR algorithm, namely the computational complexity and the memory requirements increase exponentially with the memory μ of the convolutional code. Still, it is desired to have codes with as high memory as possible as the free distance of the code decreases linearly and the event error probability decreases exponentially with the memory of the code according the discussion in subsection 4.2.2. However, in this case other than maximum–likelihood or MAP decoding algorithms have to be applied. Again we focus on sequential decoding algorithms. In this section we show how we can employ suboptimal decoding strategies and obtain a reliability vector $\mathcal{R}_{\hat{\mathbf{u}}}$ for the decoded path $\hat{\mathbf{u}}$.

Path Reliability for Variable Length Codes

Recall the VLC detection principle as introduced in subsection 4.2.3: Each message $u \in \mathcal{U}$ is mapped to a variable length code word $\mathbf{x}(u)$ of variable length $l\,\{\mathbf{x}(u)\}$. The code symbols $x_t(u)$, $t = 1, \ldots, N_u$, belong to some alphabet $\mathcal{S}_x$ and the maximum codeword length is denoted by $N_{\mathcal{U}}$. The set $\mathcal{T}_x = \{\mathbf{x}(u)\,|u \in \mathcal{U}\,\}$ of all codewords defines a VLC. The code words are transmitted over a discrete memoryless channel and we observe the channel output sequence $\mathbf{y} = \mathbf{y}_{(0:N_{\mathcal{U}}]}$. The VLC decoder according to (4.41) estimates the transmitted variable length codeword $\mathbf{x}(\hat{u})$ and its assigned message $\hat{u}$ in such a way that the error probability is minimized. Equivalently to the definition of the path reliability in (4.50) we obtain the reliability $\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}')$ for any subsequence $\mathbf{x}'$ with $\mathbf{x}'$ a prefix of a valid code word in the VLC code tree $\mathcal{T}_x$ as

$$\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}') \;\triangleq\; \Pr\{\mathbf{x}'|\mathbf{y},\mathcal{U}\} \tag{4.58}$$

where the condition on $\mathcal{U}$ denotes the VLC structure of the code. To be more specific, let us define $\mathcal{U}(\mathbf{x}')$ as a subset of $\mathcal{U}$ which contains all messages u with the same prefix $\mathbf{x}'$, i.e., $\mathcal{U}(\mathbf{x}') \triangleq$

$\left\{u \in \mathcal{U} \left| \forall_{0<t\leq l\{\mathbf{x}'\}} \quad x_t(u) = {x'}_t \right.\right\}$. Then, with (4.39), the reliability of this specific sequence $\mathbf{x}'$ is computed as

$$\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}') = \frac{\sum\limits_{u\in\mathcal{U}(\mathbf{x}')} \Pr\{u,\mathbf{y}\}}{\sum\limits_{u\in\mathcal{U}} \Pr\{u,\mathbf{y}\}} = \frac{\sum\limits_{u\in\mathcal{U}(\mathbf{x}')} \exp(\Lambda(u))}{\sum\limits_{u\in\mathcal{U}} \exp(\Lambda(u))} \tag{4.59}$$

with $\Lambda(u) = \sum_{t=1}^{l\{\mathbf{x}(u)\}} \lambda_{\text{Fan}}(y_t, x_t(u))$ and $\lambda_{\text{Fan}}(y_t, x_t(u))$ according to (4.40). Assume that we target to compute the path reliability vector for any $\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}(u))$ for any code word $\mathbf{x}(u)$ given the metric $\Lambda(u)$ for each code word $\mathbf{x}(u)$. The following proposition is helpful for this purpose.

Proposition 2 *The reliability $\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}(u))$ for any code word $\mathbf{x}(u)$ is recursively computed as*

$$\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}_{(0:t]}) = \mathcal{R}_{\mathbf{y},\mathcal{U}}(\{\mathbf{x}_{(0:t]}, +1\}) + \mathcal{R}_{\mathbf{y},\mathcal{U}}(\{\mathbf{x}_{(0:t]}, -1\}), \tag{4.60}$$

with boundary conditions according to (4.59) for all code words $\mathbf{x}(u)$.

The proof is obvious from (4.59). Therefore, given the metric $\Lambda(u)$ for each code word $\mathbf{x}(u)$, we can formulate the following algorithm to compute the path reliability vector $\mathcal{R}\mathbf{y},\mathcal{U}(\mathbf{x}(u))$ for any code word $\mathbf{x}(u)$.

1. Compute the denominator of (4.59) by summing the exponentials of all codeword metrics $\Lambda(u)$.
2. Compute the reliability $\mathcal{R}_{\mathbf{y},\mathcal{U}}(\mathbf{x}(u))$ for all code words $\mathbf{x}(u)$ by dividing the exponentials of the corresponding metrics by the denominator of step 1.
3. Compute the reliabilities for each prefix using $\mathbf{x}(u)$ for each t recursively according to (4.60).

Note that with the provision of the code word metrics $\Lambda(u)$ the computation of the path reliability is a pure post–processing step without modifications in the computation of the code word metric.

Reliability for Incomplete Sequential Decoding of Convolutional Codes

This principle of path reliabilities will now be exploited to generate useful cross–layer information for the case that any sequential decoding algorithm was not able to finalize the decoding process within a certain amount of provided computational resources. Consider an information sequence $\mathbf{u}$ of length k is encoded with a $1/c$ convolutional code resulting in a code word $\mathbf{x}(\mathbf{u})$. The transmission over a discrete memoryless channel with transition probability $\Pr\{Y = y|X = x\}$ results in a received sequence $\mathbf{y}$, being represented by the channel LLR $\boldsymbol{\psi}$. Sequential decoders repeatedly carry out *add–compare–select* operations and tree extensions according to (4.46). As discussed in subsection 4.2.3 in case of limited resources for the decoder – in our case in the following expressed by the maximum number of *add–compare–select* operations per information bit, $N_{\text{op,max}}$ resulting in $kN_{\text{op,max}}$ operations per information block – decoding might not be successful and results in an erasure. Despite incomplete the decoding process terminated after $kN_{\text{op,max}}$ operations provides a partially explored VLC code tree $\mathcal{U} = \mathcal{U}_{kN_{\text{op,max}}}$, a metric $\Lambda(\mathbf{u})$ for all $\mathbf{u} \in \mathcal{U}$, and a most promising information subsequence $\hat{\mathbf{u}} = \hat{\mathbf{u}}_{N_{k\text{op,max}}}$ according to (4.46). Note that in general the length of this sequence $\hat{\mathbf{u}}$ is lower than the initial information sequence length, i.e., $l\{\hat{\mathbf{u}}\} < k$ as otherwise the decoding algorithm would have terminated regularly.

In case of incomplete decoding, the computation of the path reliability can basically be added to sequential decoding algorithms. This is due to the fact that the metrics required to compute the path

reliability are already computed in the decoding process of any sequential decoder. The reliability computation is a post–processing operation performed in case of incomplete decoding. Equivalently to VLC case, we are interested in the reliability $\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}')$ for any subsequence $\mathbf{u}'$ with $\mathbf{u}'$ being a prefix of any information sequence $\mathbf{u} \in \mathcal{U}$. For that, we define $\mathcal{U}(\mathbf{u}')$ as a subset of $\mathcal{U}$ which contains all messages $\mathbf{u}$ with the same prefix $\mathbf{u}'$, i.e., $\mathcal{U}(\mathbf{u}') \triangleq \left\{\mathbf{u} \in \mathcal{U} \,\middle|\, \forall_{0<t\leq l\{\mathbf{u}'\}} \quad u_t = {u'}_t\right\}$. Then, the reliability $\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}')$ is obtained as

$$\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}') = \frac{\sum\limits_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp(\Lambda(\mathbf{u}))}{\sum\limits_{\mathbf{u}\in\mathcal{U}} \exp(\Lambda(\mathbf{u}))} \tag{4.61}$$

$$= \overbrace{\exp\left(\sum_{t=1}^{l\{\mathbf{u}'\}} \lambda_{\mathrm{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\mathbf{u}'_{(0:t-1]}), {u'}_t), L_t, {u'}_t)\right)}^{\text{common information subsequence}} \cdot \frac{\sum\limits_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp\left(\sum\limits_{t=l\{\mathbf{u}'\}+1}^{l\{\mathbf{u}\}} \lambda_{\mathrm{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\mathbf{u}_{(0:t-1]}), u_t), L_t, u_t)\right)}{\sum\limits_{\mathbf{u}\in\mathcal{U}} \exp\left(\sum\limits_{t=1}^{l\{\mathbf{u}\}} \lambda_{\mathrm{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\mathbf{u}_{(0:t-1]}), u_t), L_t, u_t)\right)}, \tag{4.62}$$

with $\pi(\mathbf{u}_{(0:0]}) = \pi_0$ the initial encoder state. Then, the reliability vector $\mathcal{R}_{\mathbf{u},\mathcal{U}}$ for a certain code word $\mathbf{u} \in \mathcal{U}$ computes as

$$\mathcal{R}_{\mathbf{u},\mathcal{U}} \triangleq \left\{\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}_{(0:1]}), \mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}_{(0:2]}), \ldots, \mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}_{(0:l\{\mathbf{u}\}]})\right\}.$$

Hence, if all the metrics of terminal nodes that have been visited during the sequential decoding process are saved, the path reliability for any of these paths is easily calculated by the procedure outlined previously. In the following we shall show that, in principle, for the most prominent sequential decoding algorithms – the Stack and the Fano algorithm – these metrics are stored during the decoding process or can be saved without any essential increase in storage consumption or computational complexity. We restrict ourselves to the computation of the reliability vector of the most promising path $\hat{\mathbf{u}}$ at the time the decoding process is stopped.

To simplify or enable the computation of the reliability vector, let us introduce an abbreviation for the numerator in (4.62) as

$$\mathcal{B}_{\mathcal{U}}(\mathbf{u}') \triangleq \sum_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp\left(\sum_{t=l\{\mathbf{u}'\}+1}^{l\{\mathbf{u}\}} \lambda_{\mathrm{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\mathbf{u}_{(0:t-1]}), u_t), L_t, u_t)\right). \tag{4.63}$$

Note that $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ is very similar to the backward recursion value $\beta_t(\pi)$ defined in (4.53) except that it does not consider all code words of the convolutional code, but only those investigated in the sequential decoding process contained in $\mathcal{U}$. We refer to $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ in the following as cumulative backward value. With this definition and with (4.43) the computation of the reliability $\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}')$ can be rewritten as

$$\mathcal{R}_{\boldsymbol{\psi},\mathcal{U}}(\mathbf{u}') = \exp\left(\Lambda(\mathbf{u}')\right) \cdot \frac{\mathcal{B}_{\mathcal{U}}(\mathbf{u}')}{\mathcal{B}_{\mathcal{U}}},$$

with $\mathcal{B}_{\mathcal{U}} \triangleq \mathcal{B}_{\mathcal{U}}(\{\})$ the cumulative backward value for the empty vector. Note that the cumulative backward for any code word $\mathbf{u} \in \mathcal{U}$ results in $\mathcal{B}_{\mathcal{U}}(\mathbf{u}) = 1$ which is obvious from the definition of

$\mathcal{B}_{\mathcal{U}}(\mathbf{u})$ in (4.63). In a similar manner as for the reliability according to (4.60) and the backward recursion $\beta_t(\pi)$ in (4.56), the cumulative backward value $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ can also be computed recursively as follows.

Proposition 3 *For any subsequence* $\mathbf{u}'$ *being a prefix of any information sequence* $\mathbf{u} \in \mathcal{U}$*, the cumulative backward value* $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ *defined in (4.63) is recursively computed as*

$$\begin{aligned}\mathcal{B}_{\mathcal{U}}(\mathbf{u}') &= \mathcal{B}_{\mathcal{U}}(\{\mathbf{u}',+1\}) \exp\left(\lambda_{VLC}(\boldsymbol{\psi}_{l\{\mathbf{u}'\}+1}, \mathbf{x}(\pi(\mathbf{u}'),+1), L_{l\{\mathbf{u}'\}+1},+1)\right) \\ &+ \mathcal{B}_{\mathcal{U}}(\{\mathbf{u}',-1\}) \exp\left(\lambda_{VLC}(\boldsymbol{\psi}_{l\{\mathbf{u}'\}+1}, \mathbf{x}(\pi(\mathbf{u}'),-1), L_{l\{\mathbf{u}'\}+1},-1)\right). \end{aligned} \quad (4.64)$$

Proof. To ease the exposition let us define some auxiliary value

$$\hat{\lambda}(\mathbf{u}) \triangleq \lambda_{\mathrm{VLC}}(\boldsymbol{\psi}_{l\{\mathbf{u}\}}, \mathbf{x}(\pi(\mathbf{u})), L_{l\{\mathbf{u}\}}, u_{l\{\mathbf{u}\}}). \quad (4.65)$$

Then, the definition of $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ in (4.63) can be rewritten as follows

$$\begin{aligned}\mathcal{B}_{\mathcal{U}}(\mathbf{u}') &= \sum_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp\left(\sum_{t=l\{\mathbf{u}'\}+1}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &= \sum_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp\left(\hat{\lambda}(\mathbf{u}_{(0:l\{\mathbf{u}'\}+1]}) + \sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &= \sum_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp\left(\hat{\lambda}(\mathbf{u}_{(0:l\{\mathbf{u}'\}+1]})\right) \cdot \exp\left(\sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &= \sum_{\mathbf{u}\in\mathcal{U}(\{\mathbf{u}',+1\})} \exp\left(\hat{\lambda}(\mathbf{u}_{(0:l\{\mathbf{u}'\}+1]})\right) \cdot \exp\left(\sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &+ \sum_{\mathbf{u}\in\mathcal{U}(\{\mathbf{u}',-1\})} \exp\left(\hat{\lambda}(\mathbf{u}_{(0:l\{\mathbf{u}'\}+1]})\right) \cdot \exp\left(\sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &= \exp\left(\hat{\lambda}(\{\mathbf{u}',+1\})\right) \cdot \sum_{\mathbf{u}\in\mathcal{U}(\{\mathbf{u}',+1\})} \exp\left(\sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &+ \exp\left(\hat{\lambda}(\{\mathbf{u}',-1\})\right) \cdot \sum_{\mathbf{u}\in\mathcal{U}(\{\mathbf{u}',-1\})} \exp\left(\sum_{t=l\{\mathbf{u}'\}+2}^{l\{\mathbf{u}\}} \hat{\lambda}(\mathbf{u}_{(0:t]})\right) \\ &= \exp\left(\hat{\lambda}(\{\mathbf{u}',+1\})\right) \mathcal{B}_{\mathcal{U}}(\{\mathbf{u}',+1\}) + \exp\left(\hat{\lambda}(\{\mathbf{u}',-1\})\right) \mathcal{B}_{\mathcal{U}}(\{\mathbf{u}',-1\}), \end{aligned} \quad (4.66)$$

which proves (4.64) using the definitions in (4.63) and (4.65). ■

Reliability Computation for Stack Algorithm

For the stack algorithm as presented in subsection 4.2.3 the computation of the reliability vector is trivial as the metric $\Lambda(u)$ for all paths in the stacks is directly known. Therefore, the algorithm in

subsection 4.4.3 in combination with proposition 2 can directly be applied in this context by just replacing $\mathbf{x}(u)$ with $\mathbf{u}$. After the completion of the stack algorithms we have access to a partially explored VLC code tree $\mathcal{U}$, a metric $\Lambda(\mathbf{u})$ for all $\mathbf{u} \in \mathcal{U}$, and a most promising information subsequence $\hat{\mathbf{u}}$. To compute the reliability vector $\mathcal{R}_{\hat{\mathbf{u}},\mathcal{U}}$, we propose the following post–processing algorithm.

1. set $t = l\{\hat{\mathbf{u}}\}$.
2. set $\mathbf{u}' = \hat{\mathbf{u}}_{(0:t]}$.
3. compute the numerator of (4.61) by summing the exponential metrics $\Lambda(\mathbf{u})$ for all $\mathbf{u} \in \mathcal{U}(\mathbf{u}')$ and write in $\Lambda(\mathbf{u}')$. Delete all code words $\mathbf{u} \in \mathcal{U}(\mathbf{u}')$ from the stack to obtain a new code tree $\mathcal{U}$, i.e.,
$$\begin{aligned} \Lambda(\mathbf{u}') &= \sum_{\mathbf{u}\in\mathcal{U}(\mathbf{u}')} \exp(\Lambda(\mathbf{u})) \\ \mathcal{U} &= (\mathcal{U} \cap \mathcal{U}(\mathbf{u}')) \cup \mathbf{u}' \end{aligned} \tag{4.67}$$
4. store this metric in some auxiliary value, i.e., $\mathcal{A}_t = \Lambda(\mathbf{u}')$.
5. set $t = t - 1$.
6. if $t \geq 0$ go to 2.
7. for all information sub–sequences of $\hat{\mathbf{u}}$, compute the reliability $\mathcal{R}_{\psi,\mathcal{U}}(\hat{\mathbf{u}}_{(0:t]})$ by dividing the stored values $\mathcal{A}(\mathbf{u}')$ from 4 with the auxiliary value $\mathcal{A}(\{\})$ for the root node, i.e.,
$$\forall t = 0, \ldots, l\{\mathbf{u}\} \quad \mathcal{R}_{\psi,\mathcal{U}}(\hat{\mathbf{u}}_{(0:t]}) = \frac{\mathcal{A}_t}{\mathcal{A}_0}.$$

Note that the exact solution can only be obtained in case that the stack size is sufficiently large to store all investigated paths. Otherwise, some of the paths that are explored during the decoding process are dropped and not considered in the reliability computation. Still, as we are interested in the decoding of the path that was most likely transmitted, in a reasonably well designed system this path should not drop from the stack. By the same argument the paths that are the most likely alternatives to the actually decoded path should also not drop from the stack. Therefore, paths that are lost during the decoding process do not contribute significantly to the outcome of the reliability computation.

Reliability Computation for Fano Algorithm

In the following we shall demonstrate how the Fano algorithm can be supplemented with the computation of the reliability vector, although by its decoding principle it only stores the metric of a single path. The actual algorithm for computing these reliabilities is somewhat more intricate in comparison to the stack algorithm as it requires some additional effort already during the decoding operation of the Fano algorithm. On the other hand, as we shall see, this ensures that every terminal node visited anytime during the decoding process will be considered and, hence, leads to the optimum reliability result considering the available information. For this purpose we will use the result from proposition 3. In the decoding algorithm, the quantity $\mathcal{B}_{\mathcal{U}}(\mathbf{u}')$ has a very natural

interpretation, it collects the metric increments of all paths that leave node $\mathcal{N}(\mathbf{u}')$ and have already been explored during the decoding process.

It is important to mention that in contrast to the trellis–based reliability computation as well as for the solution based on the stack algorithm, the reliability for the Fano solution can only be determined for the most likely path $\hat{\mathbf{u}}$ at the time the decoding operation is terminated. This is as the Fano algorithm only stores one path at any time instant. However, the computation of this reliability vector for this specific vector comes at almost no additional computational cost. We extend the conventional Fano algorithm as already presented in subsection 4.2.3 and by the shaded boxes in figure 4.10.

For each node $\mathcal{N}(\hat{\mathbf{u}}_{(0:t]})$ with $t = 0, \ldots, l\,\{\hat{\mathbf{u}}\}$ in the tree it is not sufficient to store just $\mathcal{B}_{\mathcal{U}}(\hat{\mathbf{u}}_{(0:t]})$, but the backward recursive value of both extended paths, $\mathcal{B}_{\mathcal{U}}(\{\hat{\mathbf{u}}_{(0:t]}, \pm 1\})$. For this purpose we define two additional auxiliary values, $\mathcal{A}_t^{(u)}$ with $u = \pm 1$, for each depth t of the currently most promising path $\hat{\mathbf{u}}$ containing the cumulative backward value for the common information subsequence $\hat{\mathbf{u}}_{(0:t]}$ extended by $u = +1$ and $u = +1$, i.e.

$$\forall_{u=\pm 1} \quad \mathcal{A}_t^{(u)} \;\triangleq\; \mathcal{B}_{\mathcal{U}}(\{\hat{\mathbf{u}}_{(0:t]}, u\}). \tag{4.68}$$

While going forward in the decoding tree, these two auxiliary values are initiated when computing the metrics for the successor nodes in (4.46) for the actual best path $\hat{\mathbf{u}}_{i-1}$ as

$$\forall_{u=\pm 1} \quad \mathcal{A}_{t+1}^{(u)} \;=\; \lambda_{\text{VLC}}(\boldsymbol{\psi}_{l\{\hat{\mathbf{u}}_{i-1}\}+1}, \mathbf{x}(\pi_{l\{\hat{\mathbf{u}}_{i-1}\}}(\hat{\mathbf{u}}_{i-1}), u). \tag{4.69}$$

The most important extension is introduced when going backward in the tree as the already computed backward recursive values for all paths in the current subset have to be stored. When returning from t to $t-1$ we apply proposition 3 to preserve all information that is necessary to compute the reliability vector. Therefore, we weight and sum up both auxiliary values $\mathcal{A}_t^{(u)}$ of the succeeding node and store it in the corresponding auxiliary value of the current node, i.e.

$$\begin{aligned} \mathcal{A}_{t-1}^{(u)} \;&=\; \mathcal{A}_t^{(+1)} \exp\left(\lambda_{\text{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\hat{\mathbf{u}}_{(0:t-1]}), +1), L_t, +1)\right) \\ &+\; \mathcal{A}_t^{(-1)} \exp\left(\lambda_{\text{VLC}}(\boldsymbol{\psi}_t, \mathbf{x}(\pi(\hat{\mathbf{u}}_{(0:t-1]}), -1), L_t, -1)\right). \end{aligned} \tag{4.70}$$

Due to the nature of the Fano algorithm, it is obvious that these auxiliary values are deleted just as the metrics Λ_t when going backward and then again forward. When decoding is terminated as the number of operations i exceed the maximum number of operations $N_{\text{op,max}}$ the auxiliary values $\mathcal{A}_t^{(u)}$ in the decoded information subsequence $\hat{\mathbf{u}}$ include all information to compute the reliability for each node in the decoded path. For this purpose, (4.70) is carried out recursively starting at node $\mathcal{N}(\hat{\mathbf{u}})$ until the root node $\mathcal{N}(\{\})$ is reached. From the definition of $\mathcal{A}_t^{(u)}$ in (4.68) it is obvious that the sum of the two auxiliary values results in the denominator of (4.64), i.e., $\mathcal{B}_{\mathcal{U}} = \mathcal{A}_0^{(+1)} + \mathcal{A}_0^{(-1)}$. Therefore, after summing all auxiliary values in the backward recursion, the reliability vector $\mathcal{R}_{\hat{\mathbf{u}}}$ finally computes as

$$\forall t = 0, \ldots, l\,\{\hat{\mathbf{u}}\} \quad \mathcal{R}_{\psi,\mathcal{U}}(\hat{\mathbf{u}}_{(0:t]}) \;=\; \exp(\Lambda_t) \cdot \frac{\mathcal{A}_t^{(+1)} + \mathcal{A}_t^{(-1)}}{\mathcal{A}_0^{(+1)} + \mathcal{A}_0^{(-1)}}. \tag{4.71}$$

The implementation in the logarithmic domain is straightforward when applying the rules for additions and multiplications according to [RVH95] and is recommended to be used to avoid numerical problems and to save complexity.

4.4.4 Performance Comparison for Progressively Coded Sources

To assess and measure the performance of different systems for our considered scenario standard evaluation metrics such as bit or block error probability are not applicable. For this purpose we have chosen a very specific scenario which compares CRC–based systems with the optimum trellis–based FEED solution as well as suboptimal sequential algorithms under similar complexity constraints. Note that in case the FEED principle is implemented based on the sequential decoding, the number of operations performed per decoded bit can be chosen arbitrarily, leading to a scalable computational complexity in contrast to the predefined complexity of the trellis–based approach.

Consider the following exemplary system for performance comparison, the parameters are shown in Table 4.1 and Figure 4.17. We assume that a progressively encoded source message of length k is to be encoded and transmitted over channel with a priori unknown statistics. In the experiments we assume BPSK modulated signals over an AWGN channel specified by the SNR h. For the FEED–based system – indicated as (1) in table 4.1 – an information sequence $\mathbf{u}$ of length $k = 512$ bit is encoded such that the channel coding vector results in $N = 2k = 1024$, i.e., the effective overall code rate is $1/2$. For this purpose, a $1/3$–convolutional code is applied, once a systematic recursive code with $\mu = 4$, once with a systematic code with memory $\mu = 96$. Termination is not applied for both cases. A regressive redundancy profile is implemented applying rate–compatible puncturing with puncturing period $\omega_p = 8$ to the encoded block before transmitting it. Therefore, we divide the information sequence $\mathbf{u}$ in $M = k/\omega_p = 64$ layer with $k_m = \omega_p$. Note that this layer definition for the level only applies for the encoder, not necessarily for the decoder, where basically each bit represents a separate layer. The channel coding vector $\mathbf{n}$ is monotonically decreasing.

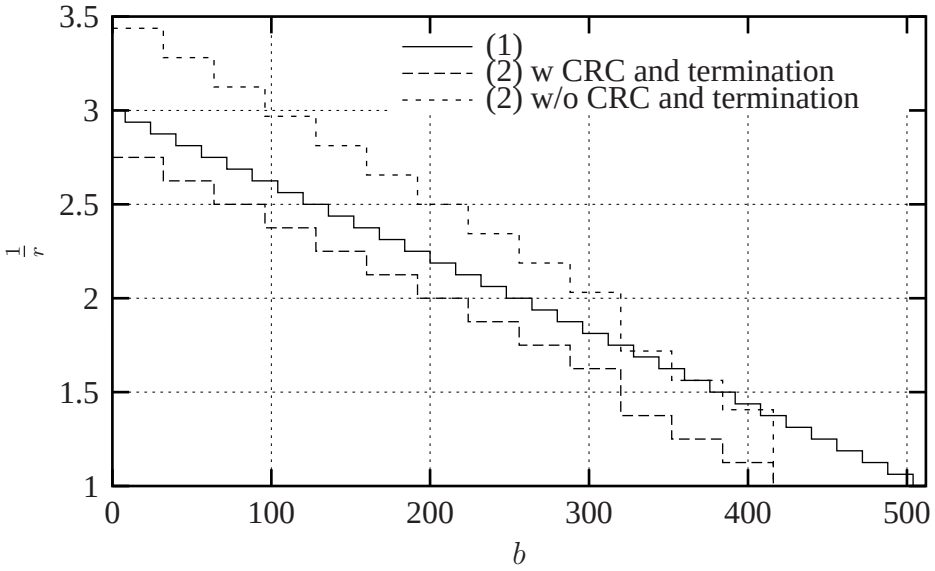


Figure 4.17: Redundancy profile $1/r$ over information size k for FEED–based systems (1), RCPC and CRC based system with $M = 13$ (2). The redundancy including CRC and termination as well as excluding them is shown.

In a first experiment we are interested in the quality of the reliability produced by the FEED–based decoder. For this purpose the following experiment has been conducted. For different channel SNRs and different FEED decoding algorithms we decode the received signal $\boldsymbol{\psi}$ to obtain a

decoded vector $\hat{\mathbf{u}}$ according to the algorithms presented in subsection 4.4.2 and section 4.4, compute the reliability vector $\mathcal{R}_{\hat{\mathbf{u}}}$ of the decoded vector $\hat{\mathbf{u}}$ and shorten the decoded path to obtain a certain requested reliability $\mathcal{R}_{\min}$. Then, we compare the obtained vector the $\mathcal{F}_s(\hat{\mathbf{u}}, \mathcal{R}_{\hat{\mathbf{u}}}, \mathcal{R}_{\min})$ and the equivalent subsequence of the transmitted information sequence, $\mathcal{F}_s(\mathbf{u}, \mathcal{R}_{\hat{\mathbf{u}}}, \mathcal{R}_{\min})$ and determine the probability $P_c(\mathcal{R}_{\min})$ that a subsequence shortened to provide reliability $\mathcal{R}_{\min}$ is correct as

$$P_c(\mathcal{R}_{\min}) \triangleq \Pr\{\mathcal{F}_s(\hat{\mathbf{u}}, \mathcal{R}_{\hat{\mathbf{u}}}, \mathcal{R}_{\min}) = \mathcal{F}_s(\mathbf{u}, \mathcal{R}_{\hat{\mathbf{u}}}, \mathcal{R}_{\min})\}.$$

Table 4.1: Code Parameters for different regressive UEP schemes.

Index	M	k	$k_m = k/M$	$\mathbf{n}$	N_{CRC}
(1)	64	64	8	see Figure 4.17	0
(2)	416	13	32	see Figure 4.17	4
(3)	416	8	52	$\{8/23, 8/21, 8/19, 8/17, 8/15, 8/13, 8/11, 8/9\}$	8
(4)	416	8	56	$\{8/23, 8/21, 8/19, 8/17, 8/15, 8/13, 8/11, 8/9\}$	4
(5)	512	8	64	$\{8/19, 8/18, 8/16, 8/15, 8/14, 8/12, 8/11, 8/9\}$	4
(6)	500	5	100	$\{8/20, 8/17, 8/14, 8/12, 8/10\}$	8

Figure 4.18 shows the simulation result for this probability $P_c(\mathcal{R}_{\min})$ over the requested minimum reliability $\mathcal{R}_{\min}$ for different FEED–based decoding algorithm and SNR h of -2.5 dB and h of -0.5 dB. For the trellis–based decoding algorithm (TBF), it is apparent, that $P_c(\mathcal{R}_{\min}) \approx \mathcal{R}_{\min}$ which justifies the appropriateness of the reliability definition according to (4.50). Similarly important, the trellis–based decoding algorithm provides an accurate information on the correctness of shortened subsequences. For the FEED algorithms based on sequential decoding $N_{\text{op}} = 16$ add–compare–select operations have been used to be similar to the complexity of the trellis–based decoding algorithm with $N_{\text{op}} \approx 2^{\mu}|_{\mu=4}$. Although the reliability produced by the sequential algorithms does not provide the same accurateness as for the optimum solution, the information is still quite accurate. The stack–based FEED (SBF) algorithm is slightly too optimistic in the assessment of the decoded and shortened path. In contrast, the Fano–based FEED (FBF) is too pessimistic in the assessment. However, the deviation is less than a factor of 3 and therefore, the computation of the reliability for sequential decoding algorithms as presented in section 4.4 is also justified. The benefits of sacrificing accurateness and using codes with higher memories will be made obvious by evaluating the amount of reliable data using different coding and decoding algorithms. Therefore, we define the average decoded information length for specific requested minimum reliability $\mathcal{R}_{\min}$ as

$$\overline{k}\left(\mathcal{R}_{\min}\right) \triangleq \mathbb{E}\left\{l\left\{\mathcal{F}_s(\hat{\mathbf{u}}, \mathcal{R}_{\hat{\mathbf{u}}}, \mathcal{R}_{\min})\right\}\right\}.$$

For comparison purpose, we have also specified several conventional systems which are based on an outer CRC code of length $N_{\text{CRC}} = 4$ or 8 as well as an inner RCPC code with memory $\mu = 4$ according to [HS99]. The specifications of five different systems are shown in Table 4.1 and partly also in Figure 4.17. Due to the overhead of the CRC and the termination, one has to sacrifice either some later part of the information sequence to obtain a similar redundancy profile (compare e.g. in (2), (3), (4), and (6)), or on has to reduce the code rate of the individual levels (compare (5)). The performance of a specific RCPC+CRC has been determined by simulation. For each code specified by a certain length k, a certain code rate r, a certain CRC length N_{CRC}, as well as

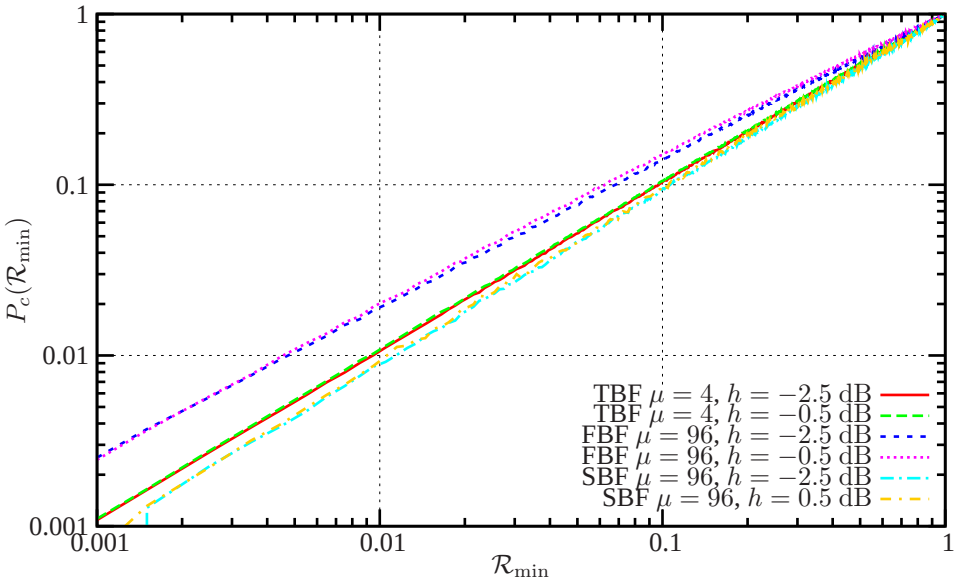


Figure 4.18: Simulated approximation of probability $P_c(\mathcal{R}_{\min})$ over the requested minimum reliability $\mathcal{R}_{\min}$ for different FEED–based decoding algorithm and SNR $h = -2.5$ dB and $h = -0.5$ dB.

a certain channel state h, the detected error probability $p_d(k, r, N_{\text{CRC}}, h)$ and the undetected error probability $p_u(k, r, N_{\text{CRC}}, h)$ has been determined.

In case that the decoding process did not detect any error up to level m,

1. the probability $P_{c,m}$ that the subsequence $\hat{\mathbf{u}}_{(0:mk/M]}$ is correct is derived as

$$P_{c,m} = \prod_{i=1}^{m} \frac{1 - p_d(k_i, r_i, N_{\text{CRC}}, h) - p_u(k_i, r_i, N_{\text{CRC}}, h)}{1 - p_d(k_i, r_i, N_{\text{CRC}}, h)},$$

   and

2. if maximally decoding up to level m, the average accepted amount of data yields

$$\overline{k}_m = \sum_{i=1}^{m} k_i P_{c,i}.$$

Note that the last term is equivalent to the performance estimation when transmitting a progressively coded source with UEP and using a quality function $\mathcal{Q}(R) = R$, i.e., the average amount of decoded bits is measured.

With these preliminaries, we can directly compare the FEED–based systems with systems relying on conventional RCPC codes together with error detection based on CRC. Assuming that we request a certain probability of correctness P_c for a subsequence to be accepted and forwarded to source decoder. Then we are interested on the average decoded amount of the initial information word $\overline{k}$ which can least be transmitted and decoded with this reliability. For the FEED–based system, the computation of the reliability and the application of the shortening function is applied. Thereby, the decoder can use the connections shown in figure 4.18 to obtain a mapping of the decoder reliability $\mathcal{R}_{\min}$ and the actual $P_c(\mathcal{R}_{\min})$ which is general at least approximately a one–to–one mapping. For the conventional systems, however, the decoder can only accept those levels

without any error detected by the CRC and, in addition, which provide sufficient reliability even if no CRC errors are detected, i.e., $P_{c,m} \geq P_c$.

In general, we are interested in a relatively high reliability $P_c > 90\%$ of the decoded subsequence, but obviously the exact value depends on the application. Thus, figure 4.19 shows the probability of an incorrect subsequence, $1 - P_c$, over the average decoded information, $\overline{k}$ for different systems and decoding schemes for SNR h of -2.5 dB and SNR h of -0.5 dB. Note that the two parameters P_c and $\overline{k}$ are connected either by the reliability $\mathcal{R}_{\min}$ or the level index m. Therefore, for the conventional systems based on RCPC codes and CRC only the dots in figure are realizable whereas for the FEED–based systems basically any point on the line is a valid decoder operation point. All systems are scalable in a sense that with better channel more information can be decoded in average when requesting the same reliability. For the CRC–based system one has to decide already in the encoder design whether reliability or the amount of decoded information part is more important. In case of reliability being important, one should chose higher CRC–lengths, otherwise lower CRC–lengths N_{CRC} are preferable. However, with $N_{\text{CRC}} = 4$ sufficient reliability is hardly achieved for low SNR such that usually $N_{\text{CRC}} = 8$ is preferable. In addition, the benefits of finer granularity (compare (6) with $M = 5$ to (3) with $M = 8$) is usually advantageous, but also depends on the SNR of the block to be decoded.

When comparing the conventional system with the trellis–based FEED algorithms, the only obvious advantage is the finer granularity. However, if a system goes without CRC and instead only relies on the distance properties of the code to obtain a reliability measure, the system generally performs worse than conventional systems as seen from figure 4.19. The by far best results are obtained by a high–memory convolutional code together with sequential decoding algorithms. Compared to the other schemes at least twice the amount of average data can be decoded at the same reliability of the data for the low SNR. For better channels the difference is still remarkable. It is also worth to note that the stack–based FEED algorithm outperforms the Fano–based algorithm. However, considering the low complexity and storage requirement of the Fano algorithm the performance is still extraordinary. In addition, the figures show the virtual bound based on redundancy profile (1) assuming that only rates below the cutoff rate $R_0(h)$ can be decoded. The bound is exceeded significantly as long as the requested reliability is below very high values.

4.5 Forward Error Correction for Packet-Lossy Channels

4.5.1 Introduction and Framework

In many practical situations one has to deal with channels which loose entire link layer or application layer packets rather than having access to the physical layer for appropriate error protection schemes. Scenarios have been discussed and appropriate channel models have been introduced in subsection 2.4.4 and 2.4.5, respectively. For most scenarios the retransmission of lost packets is the method of choice. However, there exist situations when this is not applicable, for example if back channels are not available or retransmissions cannot be performed due to delay constraints of the application. One can hope that the application provides sufficient error resilience or forward error protection can be introduced. Although almost completely ignored for a long time, it is well known that packet erasure channels can be made reliable by the introduction of forward error correction. For real–time transmission within the Internet community the only existing methods rely on RFC2733 [RS99]. However, with the idea of multicasting and broadcasting information over the Internet as well as within existing wireless systems [JSL04] this research area has been revitalized, many concepts have been rediscovered, and new ideas have been introduced

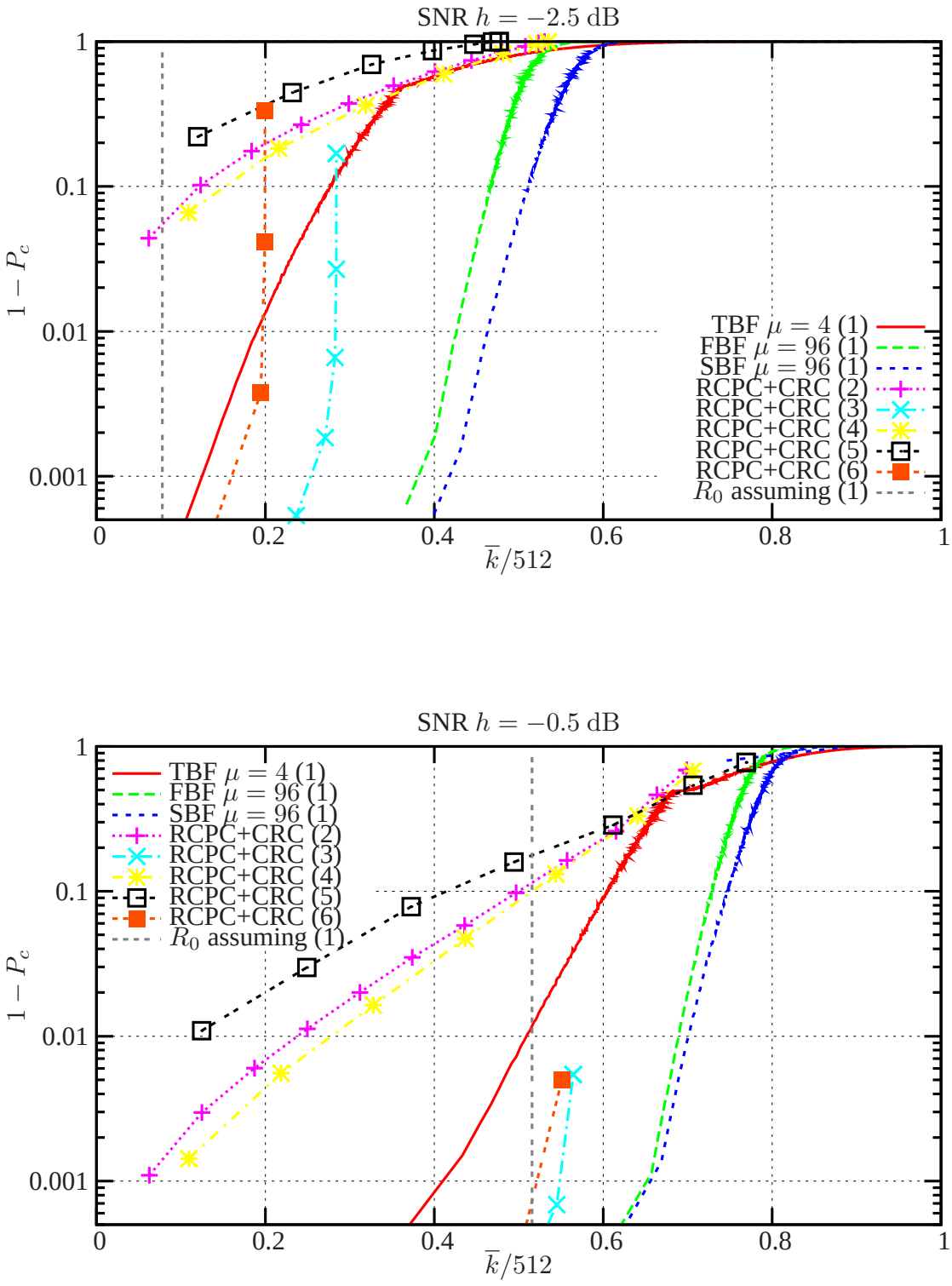


Figure 4.19: Probability of an incorrect subsequence, $1 - P_c$, over the average decoded information, $\overline{k}$ for different systems and decoding schemes for SNR h of -2.5 dB and SNR h of -0.5 dB.

in [LMSS01, LVG+02, LWSS07].

As the bits or symbols within one packet are all lost or correctly received, the application of bit-error correction FEC is obviously not appropriate. All concepts introducing FEC for packet lossy channels rely on the application of binary or symbol–based codes over individual bits or symbols, respectively, in a sequence of k consecutive packets. This concept can also be viewed as a block interleaver over successive link or application layer packets.

Assume the following framework: A sequence of k consecutive packets, each of length $l_{\mathcal{P}}$ bits or symbols, is expanded by $n-k$ parity packets such that in total we have n packets to transmit a message of $l_{\mathcal{P}}k$ symbols. The parity packets are formed by binary or symbol-based codes by applying the same channel code to the same position over all k information packets. In case of a linear code, the performance is exclusively determined by the binary or symbol–based code. With this background in mind we use the term symbol and packet erasure synonymously in the sequel.

From the Singleton bound [LJ83] we know that reconstruction of a length–k sequence of erased symbols is not feasible if the number of erased symbols, $\hat{h}$, exceeds the redundancy of the code, $n-k$. However, there exist so–called Maximum Distance Separable (MDS) codes which allow to reconstruct all k information packets if any k of the n packets are received. Reed-Solomon (RS) codes for instance are MDS codes, but are limited to block lengths $n \leq 2^{n_s}-1$, with n_s the symbol size of the code. Since usually the symbol size is selected to $n_s = 8$ bits corresponding to one byte, the maximum block length is restricted to $n \leq 255$. Furthermore, RS codes have significant encoding and decoding complexity. Recently, other codes operating with arbitrary long block lengths $k \to \infty$ and providing low encoding and decoding complexity have been introduced [Lub02, Sho06, LWSS07]. These practical codes allow encoding for arbitrary block lengths but do not meet the Singleton bound [LMSS01]. However, the inefficiency – defined by ratio of the average number of parity bits necessary to be received to reconstruct all k information symbols – is below 1.1 for these low–complexity codes [Lub02, Sho06]. Although we restrict ourselves in the remainder on RS codes and assume MDS, the framework is applicable for practical codes as well.

With the existence of codes with $n \to \infty$ fulfilling the Singleton bound it is obvious that the capacity of erasure channels with symbol or bit erasure probability ε is given as $R_C = 1-\varepsilon$. However, usually infinite block lengths are not feasible, especially in packet transmission. Commonly, the amount of combined packets is significantly restricted due to delay constraints which restricts the code length n. Similar to Gallager bound (4.2) for binary input channels, the limited block length n of codes when applied to packet or symbol erasure channels also results in an outage probability P_{out} which depends on the code block length n, the information length k, and the statistics of the erasure channel. The statistics of the channel are most suitably expressed by the block erasure probability $p_n\left(\hat{h}\right)$ expressing the probability that exactly $\hat{h}$ out of n symbols are erased where $0 \leq \hat{h} \leq n$. Note that $\sum_{\hat{h}=0}^{n} p_n\left(\hat{h}\right) = 1$. For example for a memoryless erasure channel with packet loss probability ε the block erasure probability computes as

$$p_n\left(\hat{h}\right) = \binom{n}{\hat{h}} \varepsilon^{\hat{h}} (1-\varepsilon)^{n-\hat{h}}. \tag{4.72}$$

If the channel is not memoryless, the computation is more difficult; for details we refer to [Lie99] and references therein.

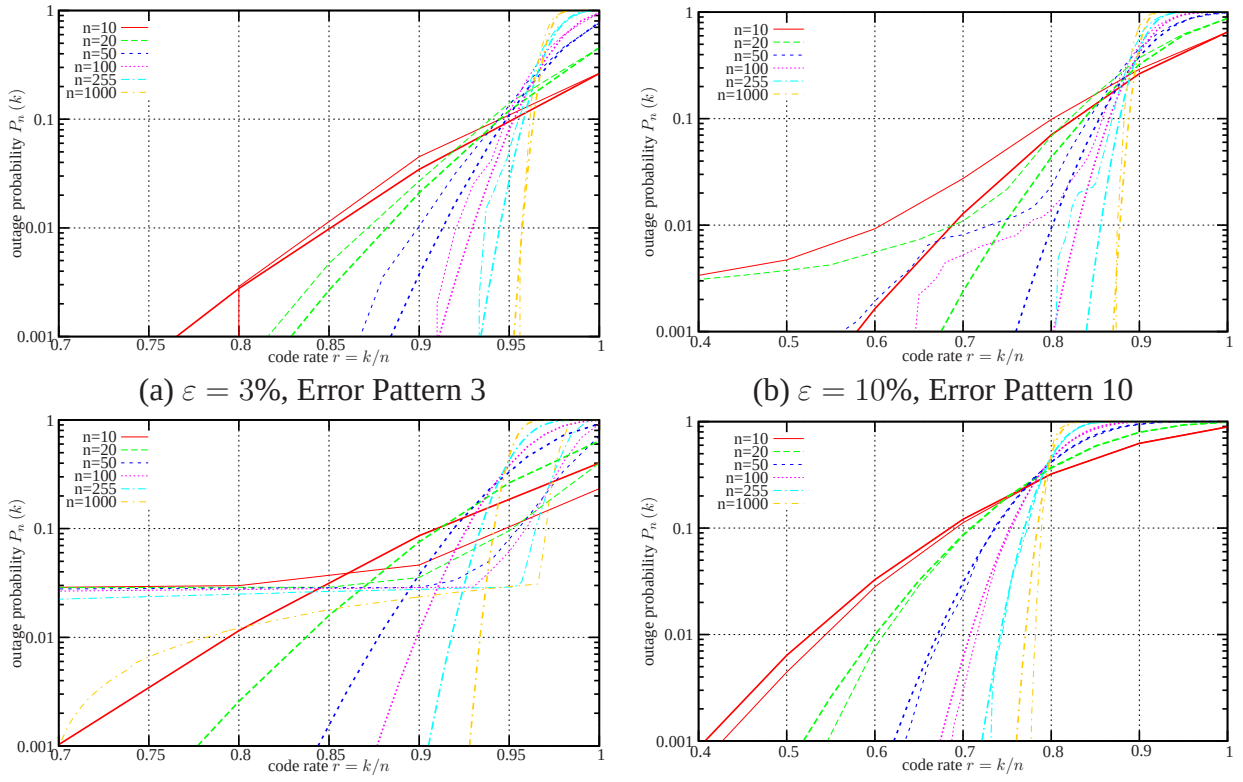
4.5.2 Code Performance

Assume that we apply a code with MDS properties and code parameters (n, k) to a channel with block erasure probability distribution $p_n\left(\hat{h}\right)$. Then, the outage probability for this code, $P_n(k)$, computes as the sum of the probability of all events which cannot be decoded by this specific code, i.e.

$$P_n(k) = \sum_{\hat{h}=k+1}^{n} p_n\left(\hat{h}\right). \tag{4.73}$$

Similarly to the description for the performance for the block–fading AWGN channel by a pair of a channel r and an outage probability $P_{\text{out}}(r, H)$ we obtain for the packet erasure channel and the application of an MDS code description $P_n(k)$ for a specific erasure channel.

Figure 4.20 shows the outage probability $P_n(k)$ over the code rate $r = k/n$ for an MDS code with different code length n and different channels. Each diagram shows the performance of a statistically independent packet erasure channel with different erasure probability $\varepsilon = 3, 5, 10, 20\%$ as well as the performance when applied over one of the Internet error patterns (3,5,10,20) as introduced in subsection 2.4.5.



(a) $\varepsilon = 3\%$, Error Pattern 3

(b) $\varepsilon = 10\%$, Error Pattern 10

(c) $\varepsilon = 5\%$, Error Pattern 5

(d) $\varepsilon = 20\%$, Error Pattern 20

Figure 4.20: Outage probability $P_n(k)$ over the code rate $r = k/n$ for an MDS code with different code length n and different channels. Each diagram shows the performance of a statistically independent packet erasure channel with different erasure probability $\varepsilon = 3, 5, 10, 20\%$ (thick lines) as well as the performance when applied over one of the Internet error patterns 3, 5, 10, and 20 (thin lines).

The outage probability over the code rate shows similar behavior as for good codes applied to the block–fading AWGN channel. The outage probability $P_n(k)$ decreases with decreasing code

rate $r = k/n$. An important parameter on the performance of the system is the code symbol length n which has an equivalent interpretation as the number of slots F in case of the AWGN channel: Better performance for increasing block length n can be traded with higher delay just as in case of the block–fading AWGN channel. The results also reflect that in some cases the loss process on the Internet is very close to statistically independent packet losses and FEC can be applied even for reasonable delays, see, e.g., pattern 3 and 20. Error pattern 10 already shows some bursty error behavior and requires longer block lengths to obtain outage probabilities below 1%. In case of error pattern 5, the application of FEC is almost useless unless we have codes with $n \approx 1000$.

A simple generic FEC scheme for packet losses has been standardized in RFC2733 [RS99]. It provides means to generate parity packets from a sequence of k information packets which can basically be of different length. All relevant information of missing RTP packets can be recovered. In addition, RFC2733 provides a simple decoding scheme based on XOR connections, for details we refer to the standard text. However, RFC2733 is limited to binary codes with code length $n \leq 24$ can be supported. For details the interested reader is referred to [RS99] or [Tse03].

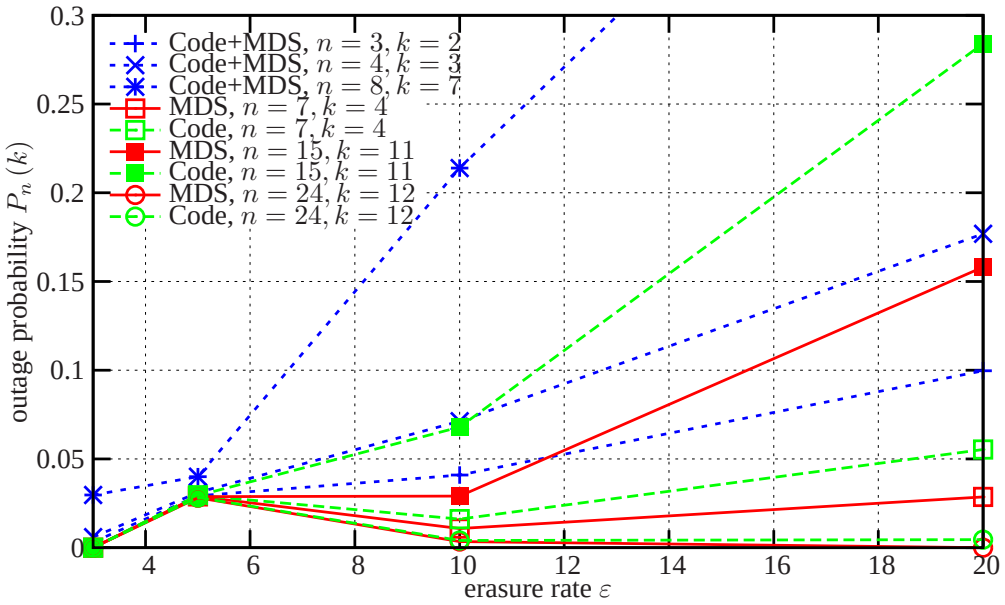


Figure 4.21: Outage probability $P_n(k)$ over the error file for different codes applying RFC 2733 as well as the MDS bounds for these code parameters.

To evaluate the performance of simple binary FEC for packet–lossy channels we have applied different codes to the transmission of an RTP stream over the Internet error patterns. Figure 4.21 shows the outage probability $P_n(k)$ over different error file labels for different codes applying RFC 2733. Also shown are the MDS bounds for the corresponding code parameters. The parity check codes with $k = n - 1$ are MDS codes and therefore, the performance is identical with the bounds. For all other binary codes[19] like the Hamming codes with $n = 7$, $k = 4$ and $n = 15$, $k = 11$ as well the extended Golay code with $n = 24$, $k = 12$, the code performs slightly worse than the MDS bounds. For error pattern 3, almost all codes, even very simple, can reduce the error rate significantly. For error pattern 10 and 20, parity check codes are not sufficient but the Hamming code with $n = 7$, $k = 4$ and the Golay code perform well to obtain residual error rate below 1%, but require about 50% redundancy. Note that some codes even worsen the performance,

[19] For more details on good binary codes the interested reader is for example referred to [LJ83].

the residual error rate $P_n(k)$ is above the original error rate ε. Finally, error pattern 5 shows the limitation of FEC in case of burst errors. Even relatively strong codes can only remove small amount of errors. The error bursts result in a high residual loss rate.

4.6 Summary

In this chapter error protection tools suitable for the transmission over wireless and packet-lossy channels have been introduced. The following observations and findings are of major importance:

- The existence of asymptotically *good* forward error correction codes providing also flexibility in terms of block lengths and code rates provide the option to apply forward error correction schemes in different system designs in an optimized manner. In addition, the performance can quite well be estimated by the use of theoretical bounds such as channel capacity or cutoff rate. This knowledge provides a powerful toolkit for system analysis, design and optimizations.

- The concepts of unequal error protection, automatic repeat request schemes, as well as power control and channel adaptation in combination with forward error correction provide a comprehensive toolkit for different error protection schemes. Whereas in case of infinite delay, the performance of the different schemes is quite similar and of less interest, we have shown in the initial part of this chapter that for delay-limited environments, the use of suitable schemes may have significant benefits.

- Specific forward error correction schemes have been introduced and discussed in greater detail. Special focus was on rate–compatible punctured convolutional codes, including those with high-memory. The performance of the codes has been analyzed. Furthermore, appropriate mobile radio access schemes and interleavers have been proposed.

- Different decoding methods for convolutional codes have been introduced and analyzed, with special focus on sequential decoding schemes based on stack and Fano algorithm. The performance of high-memory convolutional codes together with sequential decoding at the cutoff rate has been verified.

- By the observation that for progressive multimedia codecs any data after the first decoding error cannot be interpreted any more, a new error protection scheme has been introduced, namely regressive UEP together with path shortening at the receiver. In this case only the initial reliable information of a source block is delivered to the media decoder.

- Specifically, the Far End Error Decoding (FEED) has been developed in this work. FEED is an extension to conventional convolutional decoding algorithms in a sense that it allows to determine a path reliability. This path reliability is used to determine the useful and reliable initial part of a progressively coded source.

- An optimal path reliability based on a trellis solution has been developed. However, due to the complexity especially for higher memory convolutional codes, a major contribution of this chapter is the derivation of a path reliability computation based on stack and Fano sequential decoding. Performance results and comparisons have been provided.

- Finally, this chapter also introduces erasure protection schemes for the use over link layer or application layer lossy channels. Promising performance results for IP packet-loss channels have been obtained.

- In summary, this chapter provides the tools for error protection schemes in mobile and packet-loss environments. In combination with the system design tools, the application of these schemes and the optimization together with media data and service requirements will allow to provide suitable and well-designed video communication systems. Some specific realizations will be discussed in the next chapters.

5

Progressive Source Coding and Unequal Error Protection

5.1 System Overview

In the following, we will consider a system consisting of a scalable or progressive source coder in combination with an Unequal Error Protection (UEP) channel coding scheme. To be more specific, the source coder is realized by a κ–dimensional progressive source coder as specified in subsection 3.1.3 producing M decodable approximations of the source S. The generated index sequence $\mathbf{i}_{[1:m]} = i_1, \ldots, i_m$ is mapped to an embedded binary representation of length k_m for each layer m.

Before transmitting these binary code words over a noisy channel, an individual channel code is applied to each layer with code rate $r_m = k_m/n_m$ with n_m representing the total number of binary channel coded symbols available for layer m. As in general different code rates are allocated to each individual layer this results in an UEP scheme. We assume that the total number of channel symbols, $n \triangleq \sum_{m=1}^{M} n_m$, is constrained. Therefore, we can equivalently express the fraction of channel symbols available for layer m divided by the total amount of channel symbols as $\omega_m = n_m/n$. Note that we can normalize the overall rate constraint, n, for a source coder with dimension κ by an constraint on the the overall transmission rate $R = n/\kappa$ with unit bits per source dimension.

Figure 5.1 shows a block diagram of the system under consideration. After transmitting over a channel with channel state h each individual layer is channel decoded separately at the receiver. The channel decoding may result in a reconstructed index for the layer j_m or it may just report that the decoding was not successful. Based on the reconstructed layer indices j_m a source is reconstructed based on the received sequence of layers $\hat{s}_{j_1,j_2,\ldots,j_M}$.

As already discussed for the UEP scheme in subsection 4.4.4, it is neither straightforward nor trivial to determine appropriate system parameters. Among others an appropriate selection of source and channel coding rates depends on the application, the performance measure, the source characteristics, the channel characteristics, the receiver structure as well other practical constraints. To be more specific we are interested in the following question: Given certain channel characteristics, e.g., by some channel states or channel state distributions, as well resource constraints, e.g., by the total amount of channel symbols, n, or a total transmission rate R, what is an appropriate

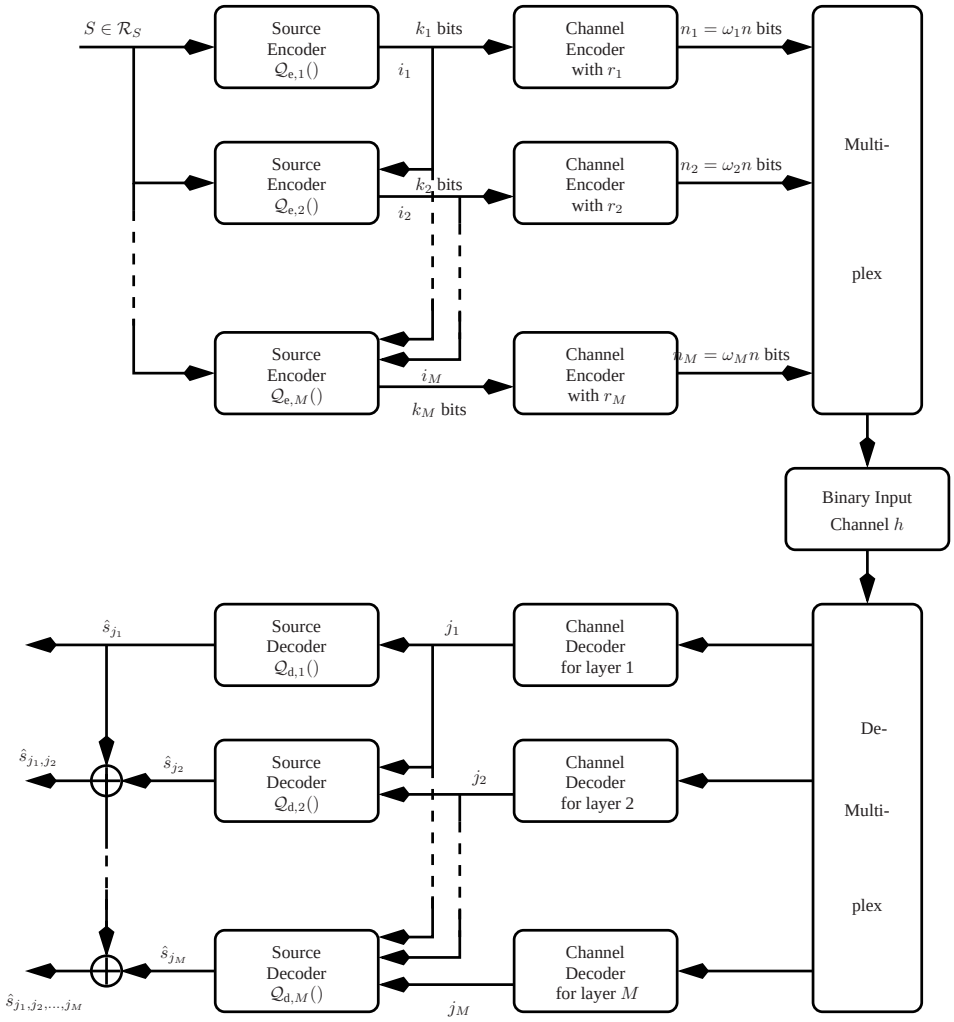


Figure 5.1: Multi-layer Coding and Transmission System.

number of layers M and what are appropriate channel coding rates r_m for a progressively coded source. This question cannot be answered generically, so we restrict ourselves in the following to some representative scenarios which either provide theoretical background or have significant practical use cases.

5.2 Information-Theoretic Considerations

5.2.1 Problem Formulation

Assume that we like to transmit a source, for example an image, over a noisy channel, for example an AWGN channel. We insist that this source is decodable at different intermediate rates. This might be targeted as someone likes to decode a partial glimpse earlier for example to stop the transmission of an undesired image when browsing a data base. We ask ourselves what is the performance penalty with respect to the full resolution when providing this feature. Although this metric might not be suitable for all questions it allows us to investigate another application with exactly the same system setup and evaluation tools: We consider using multiple layers for the source coder and UEP to partially decode the source received over a noisy channel. This addresses

the need that the lower layers in a progressively coded source have higher significance than the higher layers.

To be more specific, let S be a κ–dimensional random variable with probability density function f_S over $\mathcal{R}_S$, a closed bounded subset of $\mathbb{R}^\kappa$ with nonempty interior. The source produces a κ–dimensional vector s which is encoded using a progressive vector quantizer according to (3.11) such that for each $m = 1, \ldots, M$, the quantizer $\mathcal{Q}_\mathrm{e}$ outputs a bit index i_m of length $k_m = \kappa\omega_m r_m R$. Each index i_m is channel encoded, sent over a binary symmetric input channel, channel decoded and received as an index j_m of the same size k_m. The set of the received indices $\mathcal{J}_m$ is assumed to be identical to the transmitted index set $\mathcal{I}_m$ for each m. Note that due to maximum–likelihood channel decoding and no other means for error detection an erasure symbol is excluded from all $\mathcal{J}_m$ for the discussions in this section.

We assume that the encoder and the channel decoder are aware of the channel state. The source decoder simply reconstructs the reproduction value q_{j_m} of the decoded index j_m. Therefore, after transmission and channel decoding we reconstruct the received index sequence $\mathbf{j}_{[1:M]}$ to obtain a reconstruction $\hat{S}^{(M)} \triangleq q_{\mathbf{j}_{[1:M]}}$.

Note that the decoded layer indices j_m are in general not deterministic, but they strongly depend on the observed channel as well as on the applied channel coding rate r_m. Therefore, we address the randomness of the decoded layer indices by writing J_m. As we fully decode the entire source regardless whether errors have occured or not, the measure of interest in our case is the expected p-th power distortion at the receiver defined as

$$D = \mathbb{E}_{S,\mathbf{J}_{[1:M]}} \left\{ \left\| S - q_{\mathbf{J}_{[1:M]}} \right\|^p \right\}, \tag{5.1}$$

where the expectation is taken over both the source S and the channel expressed by the random channel indices $\mathbf{J}_{[1:M]}$. Depending on the overall rate per source component, R, we are interested in

- a bounded expression of the distortion for a given assignment of code rates $\{r_m\}_{m=1}^M$ and a distribution of the fraction of code symbols assigned to each layer, $\{\omega_m\}_{m=1}^M$;
- in code rates $\{r_m^*\}_{m=1}^M$ minimizing this distortion for a given number of layer M given the fractions of code symbols assigned to each layer, $\{\omega_m\}_{m=1}^M$;
- in a gain or penalty when introducing layered coding in terms of distortion and/or rate.

To obtain a reasonable expression for the measure interest D according to (5.1) depending on R, $\{r_m\}_{m=1}^M$, as well as $\{\omega_m\}_{m=1}^M$, we introduce some further reasonable assumptions on the source and channel coding schemes. We will concentrate on the main results in here rather than going into too many details. For a comprehensive discussion with proofs on inner and outer bounds or a single layer scheme, i.e., $M = 1$, we refer to [ZM94] and [HZ97]. For details on the multi-layer scheme, i.e. $M > 1$, we refer to [Sto01, SZ01, SZ04].

5.2.2 Assumptions

According to (3.10) the p-th power distortion of optimal vector quantizers as well as practical source decoders decays to zero as $\hat{\beta}_\kappa 2^{-pR}$ for some constant $\hat{\beta}_\kappa$ that depends on the source and the vector dimension. With (3.13) we even know that for progressive coding we have the same decay rate. The only distortion penalty paid for progressive coding is in the constant $\hat{\beta}_\kappa$, not the decay rate 2^{-pR}. Obviously, this only holds as R is sufficiently large which is in the following

indicated by '$\simeq$'[1]. Therefore, we assume in the remainder when only decoding the first layer m representations we obtain a noiseless distortion according to (3.13) repeated in the following as

$$D^{(m)}(\mathcal{Q}) = \mathbb{E}\left\{|S - \mathcal{Q}^{(m)}(S)|^p\right\} \simeq 2^{-pR^{(m)}+O(1)}.$$

The M individual channel encoders add $n_m - k_m$ bits of redundancy to the original quantization index of size k_m, respectively. At the receiver side we assume that maximum–likelihood decoding is performed. According to (4.2) it is known that an upper bound on the minimum probability of decoding error P_B goes to zero exponentially fast as a function of the channel code block length n. Similarly, Gallager [Gal68] also showed that a lower bound on the minimum error probability P_B tends exponentially to zero. Therefore, we define a *reliability function* of a binary input channel with channel code rate r as

$$E_r(r) = \limsup_{n\to\infty} \frac{-\log_2 P_B}{n}, \tag{5.2}$$

where P_B is the minimum probability of decoding error taken over all channel codes with a given block length n and rate r. We equivalently write (5.2) as $P_B \simeq 2^{-nE_r(r)}$. The reliability function $E_r(r)$ cannot be explicitly determined. However, there exist upper and lower bounds, $\overline{E}_r(r)$ and $\underline{E}_r(r)$, respectively such that for any r we have $\underline{E}_r(r) \leq E_r(r) \leq \overline{E}_r(r)$. For example, the error exponent, $\underline{E}_{r,\text{gallager}}(r,h)$ defined in (4.3) represents a lower bound on the reliability function. For any binary symmetric input channel, uniquely specified by

$$\delta_{\text{bsic}} = \int_y \sqrt{f_{Y|X,H}\left(y|x=-1,h\right) f_{Y|X,H}\left(y|x=+1,h\right)},$$

there exist even tighter upper and lower bounds than the one presented by Gallager [Gal68]. For further details the interested reader is for example referred to McEliece and Omura [MO77], Viterbi and Omura [VO79], or Litsyn [Lit99].

Figure 5.2 shows different bounds on the error exponent for a binary symmetric channel with crossover probability $\varepsilon = 0.02$ and resulting $\delta_{\text{bsic}} = 2\sqrt{\varepsilon(1-\varepsilon)} = 0.14$. Note that zero–rate exponent computes as $E_0 = 1/2\log\left(1/\delta\right)$ and the cutoff rate is generalized to $R_0 = 1 - \log(1 + \delta_{\text{bsic}})$ which conforms to (4.5) for the AWGN δ_{bsic} with $\delta_{\text{bsic}} = e^{-h}$.

To prove the lower bound on the distortion we further have to assume random index assignment on each layer m as initially introduced in [HZ97] for $M = 1$. Furthermore, as each index i_m is actually a function of the rate R, the diameters for the decision regions $\operatorname{diam}\left(\mathcal{R}_{\mathbf{i}_{[1:m]}}\right)$ are also functions of R. As R increases one would expect these cell diameters to shrink to zero, and also to shrink relative to cells from previous layers. We assume that the ration of the diameter $\operatorname{diam}\left(\mathcal{R}_{\mathbf{i}_{[1:m]}}\right)$ at the m-th stage and the diameter at the $(m-1)$-th stage, $\operatorname{diam}\left(\mathcal{R}_{\mathbf{i}_{[1:m-1]}}\right)$, decrease at least exponentially fast in terms of the excess rate at the m-th stage, i.e.,

$$\frac{\operatorname{diam}\left(\mathcal{R}_{\mathbf{i}_{[1:m]}}\right)}{\operatorname{diam}\left(\mathcal{R}_{\mathbf{i}_{[1:m-1]}}\right)} \lesssim 2^{-\omega_m r_m R} \tag{5.3}$$

where $\lesssim$ expresses that this relationship only holds as R is sufficiently large.

Finally, the probability of an incorrectly decoded index on layer m, $p_{e,m}$, is directly obtained as

$$p_{e,m} \triangleq \Pr\{J_m \neq i_m | I_m = i_m\} \simeq 2^{-\omega_m \kappa R E_r(r_m)}. \tag{5.4}$$

[1] if one wants to be more accurate, $D \simeq 2^{-\alpha R + O(1)}$ would have to be expressed as $\lim_{R\to\infty} D \cdot 2^{\alpha R} = \text{const}$ and $O(1)$ expresses that this term is independent from R.

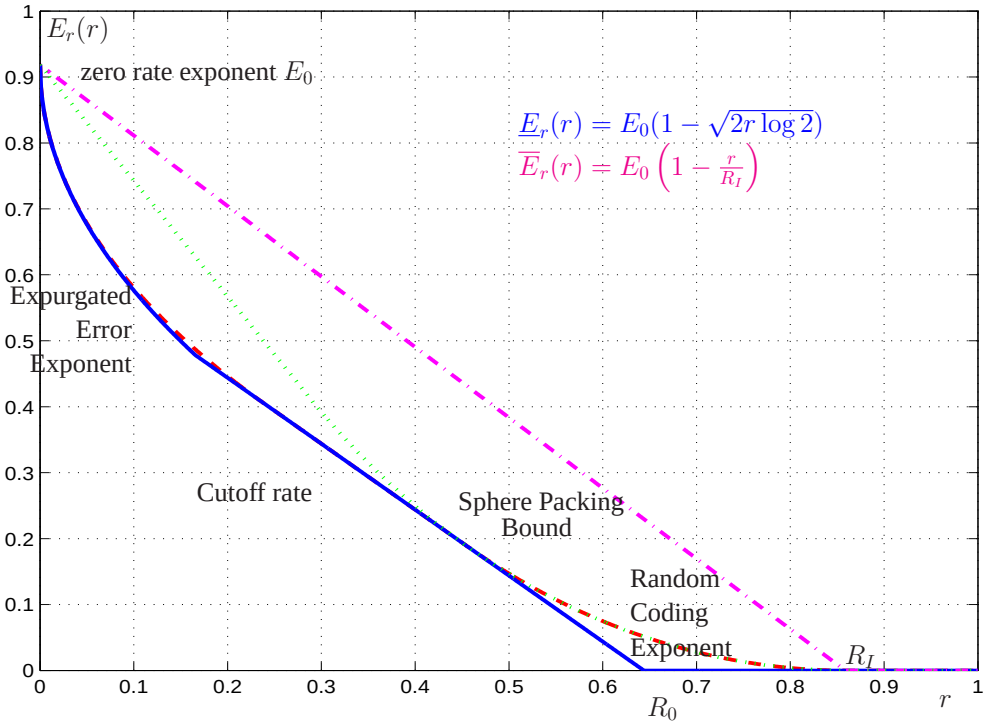


Figure 5.2: Upper and lower bounds on the error exponent $E_r(r)$ over the channel coding rate r for a binary symmetric channel with crossover probability $\varepsilon = 0.02$.

5.2.3 Distortion for Fixed Channel Coding Rates

With these preliminaries and assumptions the average p-th power distortion, $D(\mathbf{r}_M)$, of an M-layer, κ-dimensional progressive vector quantization system with $\omega_M = \{\omega_m\}_{m=1}^M$, channel code rates $\mathbf{r}_M = \{r_m\}_{m=1}^M$, channel reliability function $E_r(r)$, and transmission rate R, is determined as

$$D(\mathbf{r}_M) \simeq 2^{-pR\sum_{m=1}^M \omega_m r_m + O(1)} + \sum_{m=1}^{M} 2^{-R\left(\omega_m \kappa E_r(r_m) + p\sum_{i=1}^{m-1}\omega_i r_i\right) + O(1)}, \tag{5.5}$$

where $O(1)$ is independent of the overall rate R.

We briefly outline the idea of proof for (5.5), for more details we refer to [HZ97] and [SZ04]. Basically, one can show that an upper bound of the distortion is equivalent to (5.5) except that the error exponent, E_r, is replaced by any lower bound of the error exponent, $\underline{E}_r$. In addition, the assumption (5.3) is used. In a similar manner, an upper bound on the distortion can be found with the only difference that the error exponent, E_r, is replaced by any upper bound of the error exponent, $\overline{E}_r$. For this to accomplish, random index assignment [HZ97] is assumed. In addition, there are basically two proof techniques. In a first approach we assume that all errors can be detected at the receiver. A consequence of this assumption is that the output of the channel decoder is either the correct symbol or we know that an error occurred. If an error occurred we only use the error–free information for reconstruction and it is assumed that the perfect error detection system provides a lower–bound to the distortion. However, the last assumption is only a conjecture, a proof is not obvious. Nevertheless, this approach seems very reasonable for many practical systems. A different approach follows the idea in [HZ97] for proving the lower bound can show that full decoding with random index assignment is in the range of the source distortion of the last non–erroneous layer and it is greater equal than this distortion. With the upper and lower bounds on

the distortions which only differ in the different terms for the error exponent, (5.5) is immediately obvious.

5.2.4 Optimized Channel Coding Rates

With (5.5) we obtain the average p-th power distortion $D(\mathbf{r}_M)$ for a fixed set of channel code rates $\mathbf{r}_M$. Obviously, we are interested in channel coding rates $\mathbf{r}^*{}_M$ which minimize the distortion in (5.5), i.e.,

$$\mathbf{r}^*{}_M = \arg\min_{\mathbf{r}_M} D(\mathbf{r}_M). \tag{5.6}$$

A characterization of the optimal set of channel code rates is given in the following. The channel code rates $r_1^*, \ldots, r_M^*$ minimize the average p-th power distortion of an M-layer, κ–dimensional progressive vector quantization system with transmission rate fractions $\omega_1, \ldots, \omega_M$, and transmission rate R, if and only if

$$E_r(r_m^*) = \frac{p}{\kappa\omega_m} \sum_{i=m}^{M} \omega_i r_i^*, \quad \text{for } m = 1, \ldots, M, \tag{5.7}$$

if the overall rate per source dimension, R, is sufficiently large, i.e., $R \to \infty$. With this result, the minimum average p-th power distortion $D_{\min}$ of a progressive coding system results in

$$D_{\min} \simeq 2^{-pR\sum_{m=1}^{M} \omega_m r_m^* + O(1)}. \tag{5.8}$$

We provide an outline of the proof for (5.7) and (5.8) in the following.

Outline of proof. Let us assume that the rate is sufficiently large, i.e., $R \to \infty$, such that '$\simeq$' in (5.5) can be replaced by an equality and we ignore the constant $O(1)$ independent of R. Then, let D^* be the distortion in (5.5), minimized by choosing $r_i = r_i^*$ for all i. Then by inserting (5.7) in (5.5), we obtain

$$\begin{aligned} D^* \cdot 2^{pR\sum_{n=1}^{m-1} \omega_n r_n^*} &= 2^{-pr\sum_{n=m}^{M} \omega_n r_n^*} + 2^{-R\kappa\omega_m E_r(r_m^*)} + \sum_{j=m+1}^{M} 2^{-R\left[\omega_j \kappa E_r(r_j^*) + p\sum_{n=m}^{j-1} \omega_n r_n^*\right]} \\ &\quad + \sum_{j=1}^{m-1} 2^{-R\left[\omega_j \kappa E_r(r_j^*) - p\sum_{i=j}^{m-1} \omega_i r_i^*\right]}. \end{aligned} \tag{5.9}$$

The last summation in (5.9) does not depend on r_m^*. Thus, for all m, changing the value of r_m^* alone cannot reduce the value of the first term in (5.9)

$$2^{-pR\sum_{n=m}^{M} \omega_n r_n^*} + 2^{-\kappa R\omega_m E_r(r_m^*)} + \sum_{j=m+1}^{M} 2^{-R\left[\omega_j \kappa E_r(r_j^*) + p\sum_{n=m}^{j-1} \omega_n r_n^*\right]}. \tag{5.10}$$

Then we use induction on m. First, suppose $m = M$. Then, by (5.10), changing r_M^* cannot reduce

$$2^{-pR\omega_M r_M^*} + 2^{-\kappa R\omega_M E_r(r_M^*)}. \tag{5.11}$$

Since E_r is monotone decreasing, this minimality is achieved by equating the two exponents in (5.11), which forces the same decay rate, as $R \to \infty$ [HZ97]. Thus, the optimized channel code rate r_M^* satisfies

$$pr_M^* = \kappa E_r(r_M^*) \tag{5.12}$$

which establishes (5.7) for case $m = M$. Now assume (5.7) is true for $M, M-1, \ldots, m+1$. The induction hypothesis gives

$$\omega_j \kappa E_r(r_j^*) = p \sum_{i=j}^{M} \omega_i r_i^* \quad \text{for all } j \geq m+1 \tag{5.13}$$

and therefore for all $j \geq m+1$,

$$\omega_j \kappa E_r(r_j^*) + p \sum_{i=m}^{j-1} \omega_i r_i^* = p \sum_{i=m}^{M} \omega_i r_i^*. \tag{5.14}$$

Thus, combining (5.13) and (5.14) implies that, as $R \to \infty$, changing the value of r_m^* cannot reduce the value of

$$(M-m+1)2^{-pR\sum_{n=m}^{M}\omega_n r_n^*} + 2^{-\kappa R \omega_m E_r(r_m^*)}. \tag{5.15}$$

This minimality is achieved by equating the two exponents in (5.15), which yields

$$p \sum_{n=m}^{M} \omega_n r_n^* = \kappa \omega_m E_r(r_m^*) \tag{5.16}$$

and establishes (5.7) and therefore concludes the induction argument.

To obtain the result in (5.8) the optimized channel coding rates $\mathbf{r}^*{}_M$ as obtained in (5.7) are inserted in (5.5) and the minimum distortion is

$$\begin{aligned}
D^* &= D\left(\mathbf{r}^*{}_M\right) \\
&= 2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)} + \sum_{m=1}^{M} 2^{-R\left(\omega_m \kappa E_r(r_m^*) + p\sum_{i=1}^{m-1}\omega_i r_i^*\right) + O(1)} \\
&= 2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)} + \sum_{m=1}^{M} 2^{-R\left(\omega_m \kappa \frac{p}{\kappa\omega_m}\sum_{i=m}^{M}\omega_i r_i^* + p\sum_{i=1}^{m-1}\omega_i r_i^*\right) + O(1)} \\
&= 2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)} + \sum_{m=1}^{M} 2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)} \\
&= (M+1)2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)} \qquad (5.17) \\
&= 2^{-pR\sum_{m=1}^{M}\omega_m r_m^* + O(1)}. \qquad (5.18)
\end{aligned}$$

■

The inherent formulation in (5.7) implies a method of computing the optimal channel code rates of a progressive source coder in the presence of channel noise, by recursively solving the following equations:

$$\begin{aligned}
E_r\left(r_M^*\right) &= \frac{p}{\kappa} r_M^* \\
E_r\left(r_m^*\right) &= \frac{\omega_{m+1}}{\omega_m} E_r\left(r_{m+1}^*\right) + \frac{p}{\kappa} r_m^* \quad \text{for } m = M-1, \ldots, 1.
\end{aligned} \tag{5.19}$$

This recursive solution approach is sketched in Figure 5.3a). Since E_r is monotonically decreasing, a solution to (5.19) exists if

$$E_r\left(r_m^*\right) - \frac{\omega_{m+1}}{\omega_m} E_r\left(r_{m+1}^*\right) - \frac{p}{\kappa} r_m^*,$$

is positive at $r_m^* = 0$ for each m, that is if $\frac{\omega_{m+1}}{\omega_m} \leq \frac{E_0}{E_r\left(r_{m+1}^*\right)}$. Since $E_r(r_{m+1}^*) \leq E_0$ for all m, a sufficient condition for a solution to (5.19) to exist is that $\omega_{m+1} \leq \omega_m$ for all m.

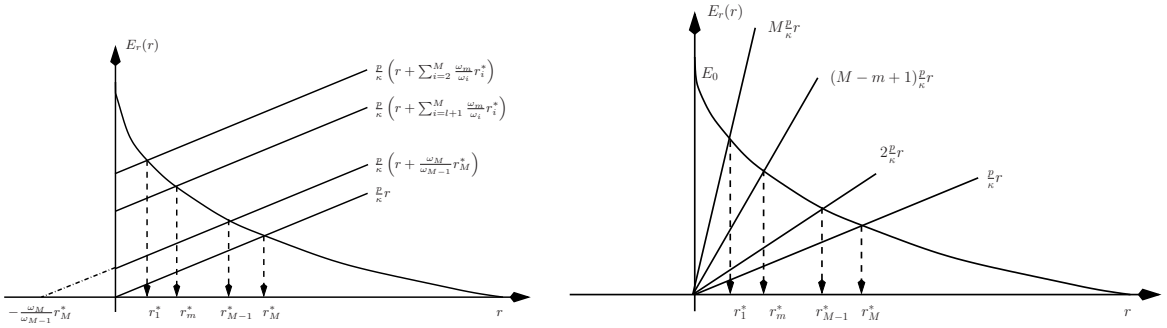


Figure 5.3: Implicit Solution to optimal channel coding rates: (a) Fixed transmission block sizes $\kappa\omega_m R$ for each layer; (b) Equal information part length k/M for each layer.

5.2.5 Equal Transmission Block Size

Let us focus on a specific but practically interesting case, namely that the transmission block size for each individual layer is equal, i.e., $n_m = n/M$ for all $m = 1, \ldots, M$. This is equivalently expressed as $\omega_m = 1/M$, and therefore, it is obvious that a solution to (5.19) exists. The recursive solution in (5.19) also implies that the optimized channel code rates r_m^* increase in m meaning that low resolution layers in the progressive source coder use more redundancy than higher resolution layers. This is also obvious from Figure 5.3a). The distortion in this case results in

$$D^* \simeq 2^{-\frac{pR}{M} \sum_{m=1}^{M} r_m^* + O(1)}. \tag{5.20}$$

Tight upper and lower bounds and exact expressions on the error exponent $E_r(r)$ are only known for specific code rates. Therefore, an exact solution on the optimized code rates in (5.7) as well as the minimum distortion is in general infeasible. However, to get an idea on the code rates, one can use the upper and lower bounds as shown in Figure 5.2. Figure 5.4 shows the channel coding rates using upper and lower bounds of the error exponent instead of the exact error exponent in (5.7) over layer index m for a binary symmetric channel different bit error rates ε as well as $p = 2$, $\kappa = 20$, $R = 20$, and any[2] $M \leq 50$.

From Figure 5.4 it is obvious that an UEP scheme with decreasing redundancy over layers should be applied. The optimized channel coding rates obviously depend on the crossover probability of the binary symmetric channel ε. Note that the optimized channel coding rates r_m^* minimizing (5.7) with the exact error exponent E_r do not necessarily lie in between the channel coding rates minimizing the upper bound and the channel coding rates minimizing the lower bounds.

[2]The presentation with $M - m$ on the x-axis allows the use of Figure 5.4 for any M.

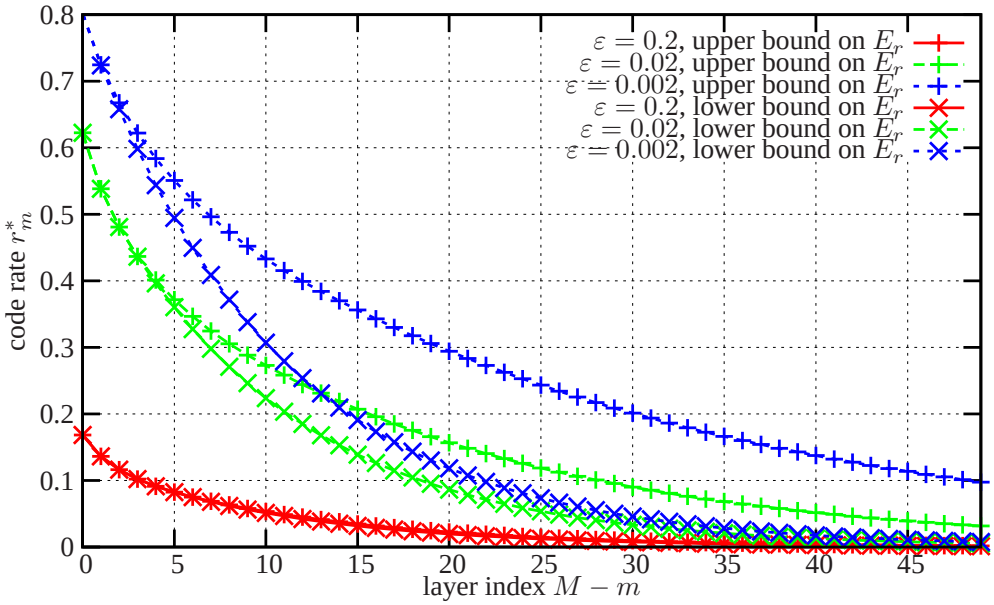


Figure 5.4: Optimized channel coding rates using upper and lower bounds on the error exponent over generalized layer index $M - m$ for a binary symmetric channel different bit error rates ε as well as $p = 2$, $\kappa = 20$, $R = 20$, and any $M \leq 50$.

In [SZ04] the performance for a high number of layers is also discussed, i.e., $M \to \infty$, for this specific case $\omega_m = 1/M$. For the result to be reasonable it has also to be assumed that the rate is significantly higher than the number of layers, $R \gg M$, such that high resolution in each layer can be assumed. For this case the minimum distortion approximately results in

$$D^* \approx 2^{-\frac{\kappa R}{M} E_0} = \delta_{\text{bsic}}^{\frac{pR}{2M}}. \tag{5.21}$$

This result can basically be shown by using (5.20) and the relatively loose analytic lower and upper bound on the error exponent shown in Figure 5.2. It is worth to note that for the binary symmetric channel with crossover probability ε the minimum distortion according to (5.21) results in $D^* \approx \varepsilon^{pR/M}$. This means that the distortion increases by introducing more layers M for the same rate constraints R.

5.2.6 Equal Information Part Length

In a second specific scenario not the numbers of channel symbols are equal for each layer m, but the number of source bits in each layer, k_m. We assume that the entire information word of length k is equally divided such that $k_m = k/M$. Therefore, we obtain the optimized channel coding rates by specifying $\omega_m = \frac{k}{R\kappa r_m^* M}$ and inserting this value in (5.7). Thus, the implicit solution for the optimized channel coding rates results in

$$E_r\left(r_m^*\right) \quad = \quad \frac{p}{\kappa} r_m^* (M - m + 1) \quad \text{for } m = 1, \ldots, M. \tag{5.22}$$

This construction principle is sketched in Figure 5.3b), which is no more recursive. The optimized channel coding rates can be computed independently for each layer m. As expected, the

optimized channel code rates r_m^* again increase with m resulting in better protection for the lower layers. The corresponding minimum distortion yields

$$D^* \simeq 2^{-\frac{pR}{\sum_{m=1}^{M}\frac{1}{r_m^*}}+O(1)}. \tag{5.23}$$

An interesting insight is obtained if we determine the total overall rate R_M necessary to support the same expected distortion D^* as would be necessary for a noiseless transmission k/κ, i.e.,

$$\frac{\kappa R_M}{k} = \frac{\sum_{m=1}^{M}\frac{1}{r_m^*}}{M}. \tag{5.24}$$

Figure 5.5 shows the rate increase $\frac{\kappa R_M}{k}$ for protecting with one or multiple layers for a binary-symmetric channel with bit error rate[3] $\varepsilon = 0.0024$. The upper and lower bounds of the error exponent are used to obtain the optimal channel coding rates according to (5.22). It is immediately observed that the rate increase when introducing layered transmission may be significant. Compared to a single layer case $M = 1$ the rate increased for layered transmission is significant. For example, $M = 10$ would already require between 3 to 4 times of the source rate which is about 1.5 to factor 2 compared to the case for single layer transmission. Therefore, for constant rate R and even optimized rate allocation, progressive transmission and unequal error protection does not come for free.

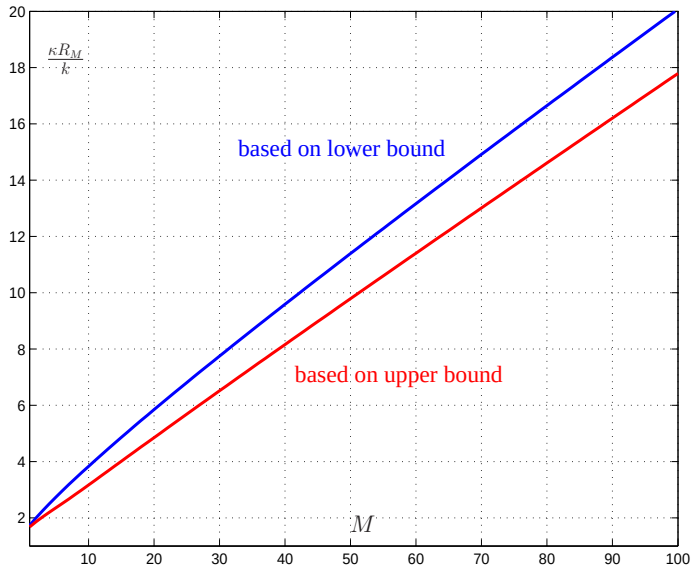


Figure 5.5: Rate Increase compared to source rate $\frac{\kappa R_M}{k}$ over the applied number of layers M for equal number of source bits in each layer.

5.2.7 One Practical Example

To investigate the relevance of the theory derived in the section, we have performed a simulation experiment, for which the optimized rate allocation according to (5.22) has been used. Again a binary-symmetric channel with bit error rate $\varepsilon = 0.0024$ has been used. To realize a practical

[3] This corresponds to a binary input AWGN with hard decision at SNR of 6 dB.

UEP scheme, punctured convolutional codes with memory $\mu = 4$ have been used. In addition, a uniformly distributed source was quantized using a multistage vector quantizer of dimension $\kappa = 4$. A total rate of $R = 20$ bit per dimension had been applied. Two schemes have been assessed, one referred to as Equal Error Protection (EEP), one referred to as Unequal Error Protection (UEP). For the EEP case, the layer is set to $M = 1$ and the number of source bits, k_m, is 44. For the UEP case, the layer is set to $M = 4$ and the number of source bits, k_m, is 8. Maximum-Likelihood decoding has been used for channel decoding.

Table 5.1: Average distortion for EEP and UEP cases based on simulations and bounds.

$\varepsilon = 0.0024$ dB	EEP $M = 1$	UEP $M = 4$
simulation	$D_{\text{sim}} = 2.1 \cdot 10^{-6}$	$D_{\text{sim}} = 4.7 \cdot 10^{-6}$
upper bound	$\overline{D} = 2.4 \cdot 10^{-7}$	$\overline{D} = 1.1 \cdot 10^{-5}$
lower bound	$\underline{D} = 2.4 \cdot 10^{-7}$	$\underline{D} = 8.3 \cdot 10^{-6}$

The results of this experiment are shown in Table 5.1 and Figure 5.6. From Table 5.1, it is observed that for UEP the simulated distortion, D_{sim} is between the upper and lower bounds, $\overline{D}$ and $\underline{D}$, respectively. For EEP, upper and lower bounds are identical and the real distortion is much worse. The EEP system has better average distortion than the UEP system. Furthermore, Figure 5.6 shows the distribution of the distortion for each decoding event. Due to the higher source rate that is available for EEP compared to UEP (44 bit instead of 24 bit), lower distortions can be achieved more often. The only advantage for UEP is that very high distortions are less frequent.

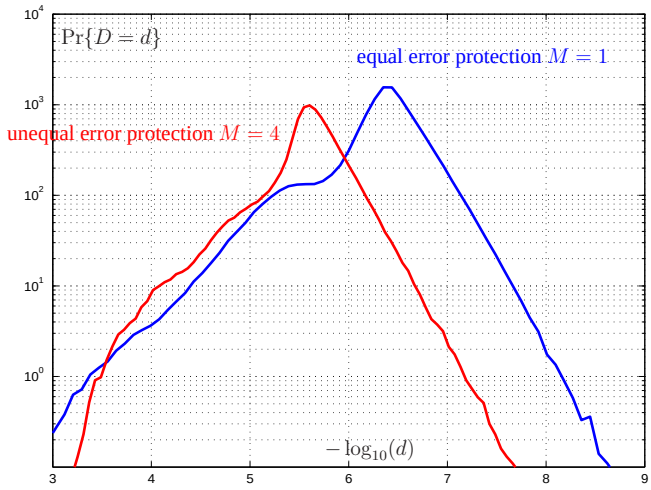


Figure 5.6: Distribution of distortion d for EEP and UEP case.

5.2.8 Discussion

We will in the following discuss some conclusions when designing progressive transmission systems. In addition, although the results presented in this section are based on specific assumptions and restrictions as introduced in 5.2.2, they also include very general system design principles which have previously been shown with less restrictive assumptions.

- From (5.5) it is obvious that for a given set of channel coding rates, the overall distortion is the sum of the quantization distortion, $2^{-pR\sum_{m=1}^{M}\omega_m r_m + O(1)}$ and the influence of the channel, the number of layers, M, the fraction assigned to each layer, ω_M, as well as on the applied channel coding rates $\mathbf{r}_M$. The distortion is basically determined by the lowest decrease factor in the exponent, i.e.,

$$\min_m \left(\omega_m \kappa E_r\left(r_m\right) + p \sum_{i=1}^{m-1} \omega_i r_i \right).$$

- The minimum distortion according to (5.8) shows that there is a distortion penalty: The exponential distortion decrease is lower than in case of noiseless transmission. Whereas for noiseless transmission the exponent decreases with the distortion power factor p with the overall rate R, for noisy and layered transmission the exponent is smaller, namely $p\sum_{m=1}^{M}\omega_m r_m^*$. This quantity basically corresponds to the mean of the optimal channel coding rates r_m weighted by the fraction of channel symbols ω_m on each layer.

- The results obtained in [ZM94] and [HZ97] for single layer transmission basically correspond to the results in (5.8) and (5.7) when inserting $M = 1$ and $\omega_1 = 1$. Therefore, from (5.7) as well from the illustration in Figure 5.3 it is obvious that for a transmission scheme with multiple layers M, only the optimum code rate for the highest layer, r_M^* is identical to the code rate of the single layer transmission. For all other layers, the channel code rates are lower. Therefore, the penalty term $\sum_{m=1}^{M}\omega_m r_m^*$ is always lower or equal when compared to the term for single layer transmission, $r_{M=1}^*$. Hence, at the same overall rate R, optimized transmission with multiple layers, is always inferior than single layer transmission under the assumptions in subsection 5.2.2. These results can also be verified for the specific cases as presented in subsection 5.2.5 and subsection 5.2.6 as well as for the practical example in subsection 5.2.7.

- Regardless of the number of layers in use, the best performance is terms of minimum distortion for a constant overall rate R is achieved when the optimum code rates are as high as possible. The highest possible code rate, the mutual information R_I is only achieved, if the slope of the lines in Figure 5.3 goes to zero. In this case, the crossing point results in a selection of a code rate such that $E_r(r^*) = 0$. However, this slope only tends to zero, if the dimension of the source goes to infinity, i.e., $\kappa \to \infty$. Therefore, for a given overall rate R the best performance in terms of minimum distortion can be achieved by using an infinitely large dimension in the source coding algorithm, as well as an infinitely long channel code length. In this case the distortion decreases exponentially with the overall rate R multiplied by R_I and all layers are basically protected by the same channel coding rates. This result can be viewed as a special case of the famous and well–known *separation principle* first manifested by Shannon [Sha48, Sha93a]: It states that a source signal can be optimally transmitted over a noisy channel by first applying source coding at the rate–distortion bound. Then, this coded version is channel coded with code rate corresponding to the capacity of the channel. It is also appealing that with this principle source coding and channel coding can be designed independently: the source coding algorithm is only adapted to the source signal, where as the channel coding algorithm matches the channel properties.

In summary, we conclude that a penalty for layered or progressive transmission has to be accepted in the overall rate–distortion performance. The analysis in this section mainly dealt with the distortion sacrifice due to channel coding with reduced block length. The penalty for layered

source coding was not addressed as it basically is less severe only affecting some constant term. Note that the parameter on the source dimension κ is of negligible influence on the source coding performance as the overall rate grows to infinity. The parameter κ mainly determines the length of the channel coding vectors. We obtain equivalent performance bounds by quantizing κ consecutive source samples, each with a scalar quantizer with rate rR, but grouping the resulting $rR\kappa$ bits to form the information part of a single channel code word.

However, we also know that for practical source coding schemes there exists an additional performance loss for layered source coding, especially when considering video transmission. Therefore, under these initial assumptions and with the measure of interest according to (5.1), a progressive transmission system comprising of layered or progressive source coding combined with optimized unequal error protection is not useful.

However, especially the assumption that the channel state for the transmission of each individual message is ergodic does not hold for delay–constraint applications when transmitted over mobile channels with non–perfect power control. Even with sufficiently large block length n_m for each layer m the transmitter cannot expect that the message can be transmitted with arbitrarily low error probability due to unpredictable channel states. Other system constraints which do for example not allow online encoding of each individual message to the actual transmission conditions also can benefit from progressive and layered source coding as will be shown in chapter 8 and 9.

5.3 Unequal Erasure Protection

5.3.1 Framework and Definitions

The equivalence of channel coding applied to packet erasure channels and the mobile channel modelled by the block–fading AWGN channels immediately asks to apply Unequal Error Protection (UEP) approaches as introduced in subsection 4.1.3 also to packet erasure channels. As not errors are considered, but packet erasures, we refer to this coding scheme as Unequal Erasure Protection (UXP). Ground–breaking work in the field has been presented in [ABE+96] under the acronym Priority Encoded Transmission (PET). The combination of UXP with progressive source coders has been introduced in [MRL00] for the transmission of SPIHT coded images which is slightly different to the PET approach as it allows a fragmentation of the total message at any symbol boundary. This framework serves also for the basis of the UXP consideration [LWPW04] within the AVT and will be briefly introduced in the following. The application to PTVC coded video is discussed in section 6.3.2.

Figure 5.7 shows the coding and decoding framework. The message is divided into M levels such that each layer includes one symbol of length n_s bits of each of the n transmission packets. The length of each transmission packet results in $N_H + M$ symbols. Each level m, $m = 1, \ldots, M$ gets assigned k_m data symbols and $n - k_m$ redundancy symbols. In packet–switched networks it is also common that each packet requires some header information of length N_H symbols. Assuming a total rate–constraint of N_{total}, the parameters are restricted as

$$nn_s(M + N_H) \leq N_{\text{total}}. \tag{5.25}$$

The transmission framework can also be viewed as block interleaver which is a commonly applied tool to spread burst errors. Assuming MDS codes, the entire layer can be decoded as long as the number of lost packets is less or equal to $n - k_m$. At the decoder all layers which have sufficient redundancy can be reconstructed.

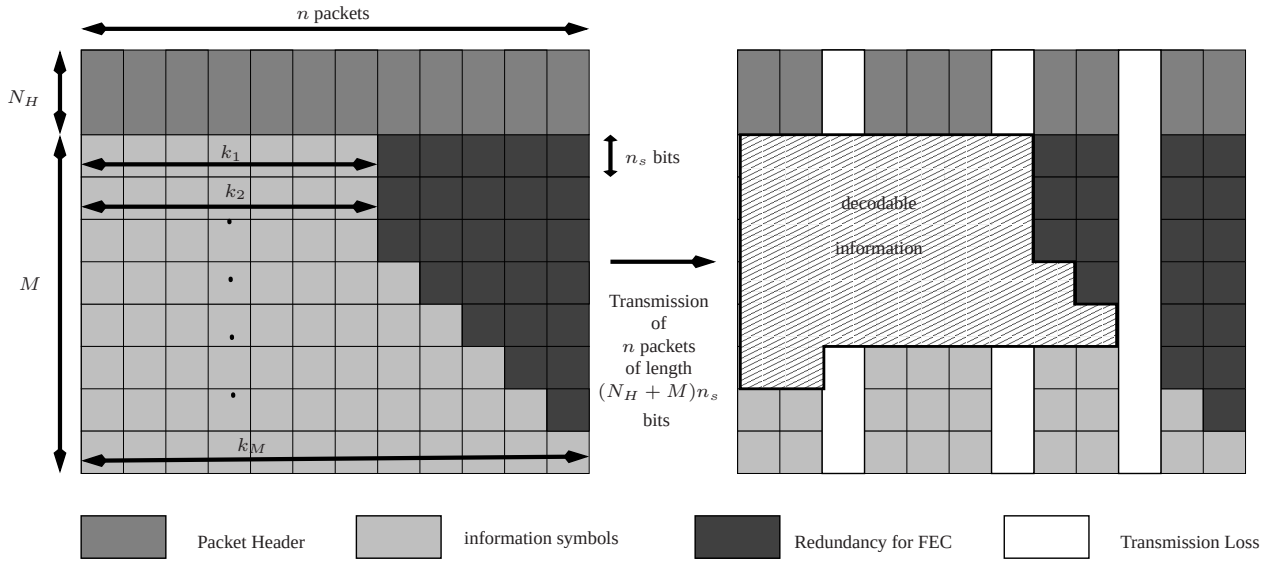


Figure 5.7: Coding and decoding framework for unequal erasure protection of progressively coded messages.

If we are interested in the expected quality for this progressively coded source as specified in (3.33) we need to obtain an appropriate expression for the probability that a certain layer m can contribute to the overall quality at the receiver, $P_m \triangleq \Pr\{\hat{\mathcal{C}}_m = 1 \wedge \forall_{i \prec m} \hat{\mathcal{C}}_i = 1\}$. Assuming a progressively coded source in a sense that after packetization the data in layer m can only be decoded if all data in the previous layers $1, \ldots, m-1$ can be decoded, i.e., $\mathcal{P}_m \prec \mathcal{P}_{m-1}$ for all $m = 2, \ldots, M$, this probability results in

$$P_m \;=\; \min_{i=1,\ldots,m} P_n\left(k_i\right) = P_n\left(\min_{i=1,\ldots,m} k_i\right). \tag{5.26}$$

By combining (3.29) and (5.26) in (3.40) the expected quality at the receiver, $\mathbb{E}\left\{\hat{\mathcal{Q}}(\mathbf{k})\right\}$, for a source with rate–quality function $\mathcal{Q}(R)$ and a packetization of the source resulting in a vector $\mathbf{k} = \{k_1, \ldots, k_M\}$ is given by

$$\mathbb{E}\left\{\hat{\mathcal{Q}}(\mathbf{k})\right\} = \sum_{m=1}^{M} \left(\mathcal{Q}\left(K_m\right) - \mathcal{Q}\left(K_{m-1}\right)\right) P_n\left(\min_{i=1,\ldots,m} k_i\right). \tag{5.27}$$

Note that if we apply systematic channel codes we might be able to reconstruct some additional symbols in the first layer which does not satisfy the PET condition anymore. This is also illustrated in Figure 5.7. However, as the contribution to the expected quality is in general very low, we neglect this part in the computation of the expectation for sake of clarity.

It is worth to note that from a different perspective this combined source–channel coding system can also be viewed as multiple description coding [WWZ80]. Multiple description coding provides a framework for the transmission of packetized source data over a channel where each packet observes the same loss statistics. In fact, progressive source coding together with UXP is considered as the most practical and efficient multiple description approach if the number of descriptions M is larger than $4 - 5$.

5.3.2 Optimized Rate Allocation

Given a certain quality–rate function $\mathcal{Q}(R)$ the expected quality in (5.27) is mainly determined by the number of levels as well as the selected source coding vector $\mathbf{k}$ under the rate constraint (5.25). In the following we assume that the number of levels, M, as well as the packet length, n, are fixed due to some transmission constraints. Then, we are interested in a channel symbol vector which maximizes some expected received quality $\overline{\mathcal{Q}}(\mathbf{k}) \triangleq \mathbb{E}\left\{\hat{\mathcal{Q}}(\mathbf{k})\right\}$, i.e.,

$$\mathbf{k}^* = \arg\max_{\mathbf{k}\in\mathcal{S}_k^M} \overline{\mathcal{Q}}(\mathbf{k}). \tag{5.28}$$

Although the execution of (5.28) seems to be rather trivial, for reasonably small M as well as a reasonable amount of source bits, a brute force search over all possible combinations gets prohibitively complex as $|\mathcal{S}_k|^M$ combinations of source vectors have to be evaluated.

An overview of approaches to lower complexity solution has been provided by Dumitrescu et al. [DWW04]. In this work, different approaches [MRL00, PR99, SHX02] are summarized with respect to their complexity as well as their generality. Among others, it was shown in [DWW04] that our approach as initially presented in [SB01] provides an optimal solution with minimum complexity, if the quality–rate function $\mathcal{Q}(R)$ is convex. Therefore, we will briefly introduce the algorithm, for further details we refer to [SB01] and [DWW04].

From [SB01, Lemma 2] it is derived that the optimal source coding vector $\mathbf{k}^*$ maximizing the expected quality in (5.28) is monotonically increasing, i.e.,

$$k^*{}_m \geq k^*{}_{m-1} \quad \forall m = 2, \ldots, M.$$

In the following we will present the optimized rate allocation as an analogy to the Viterbi algorithm. Therefore, equivalently to the cumulative metric in (4.33), we introduce a cumulative expected quality up to layer m as

$$\overline{\mathcal{Q}}_m(\mathbf{k}) \triangleq \mathcal{Q}_0 + \sum_{i=1}^{m} \left(\mathcal{Q}\left(K_i\right) - \mathcal{Q}\left(K_{i-1}\right)\right) P_n\left(k_i\right). \tag{5.29}$$

and a cumulative representation of the source coding rate as $K_m(\mathbf{k}) \triangleq \sum_{i=1}^{m} k_i$. Note that the term $\mathcal{Q}\left(K_i\right) - \mathcal{Q}\left(K_{i-1}\right)$ corresponds to the importance $\mathcal{I}_i$ of layer i, such that we can write

$$\overline{\mathcal{Q}}_m(\mathbf{k}) \triangleq \mathcal{Q}_0 + \sum_{i=1}^{m} \mathcal{I}_i P_n\left(k_i\right), \tag{5.30}$$

Assume for the moment that the total source rate is constrained to K_{tot}, i.e., $K_M(\mathbf{k}) = K_{\text{tot}}$. Then, with $\mathcal{Q}(R)$ being convex, result of Lemma 3 in [SB01], and the definition of the cumulative expected quality in (5.29), we can reduce the problem defined in (5.28) with source rate constraint K_{tot} to

$$\max_{\mathbf{k}} \overline{\mathcal{Q}}_m(\mathbf{k}) \quad \text{subject to} \quad K_m(\mathbf{k}) = K, \tag{5.31}$$

with $m = M$ and $K = K_{\text{tot}}$.

Similar to the recursions in (4.35) for the Viterbi decoder, we specify for $m = 1, \ldots, M$

$$\begin{aligned} \overline{\mathcal{Q}}_m(\mathbf{k}) &= \overline{\mathcal{Q}}_{m-1}(\mathbf{k}) + \mathcal{I}_m P_n\left(k_m\right) \\ K_m(\mathbf{k}) &= K_{m-1}(\mathbf{k}) + k_m, \end{aligned} \tag{5.32}$$

with $\overline{\mathcal{Q}}_0(\mathbf{k}) = \mathcal{Q}_0$ and $k_0(\mathbf{k}) = 0$.

Similar for the metrics for the Viterbi decoder, neither the source coding vector $\mathbf{k}$ nor $\overline{\mathcal{Q}}_m(\mathbf{k})$ nor $K_m(\mathbf{k})$ do depend on $k_{m+1}, \ldots, k_M$. Therefore, the solution of the minimization problem in (5.31) needs to be determined only over a subsequence of source coding vectors $\mathbf{k}_m \triangleq \{k_1, \ldots, k_m\}$.

With these preliminaries we are able to determine an optimized rate allocation. Let $\mathbf{k}_m^*$ be the solution for the optimization problem in (5.31) and let $\overline{\mathcal{Q}}_{m,K}^*$ with $K = \sum_{i=1}^{m} k_i^*$ denote the optimum value of the objective function in (5.31). Then, $\mathbf{k}_m^*$ is the subsequence of source code vectors achieving the maximum in (5.31) for a layer index m and a rate constraint K. Then the following results are observed.

1. $\overline{\mathcal{Q}}_{m,K}^*$ satisfies the dynamic programming equation

$$\overline{\mathcal{Q}}_{m,K}^* = \max_{k:1\le k\le n} \left(\overline{\mathcal{Q}}_{m,N-k}^* + \mathcal{I}_m P_n\left(k_m\right) \right). \tag{5.33}$$

2. The source code vector $k_{m,K}^*$ achieves the maximum in (5.33).

3. The subsequence $\mathbf{k}_{m+1}^*$ solves (5.31) for layer index $m+1$ and channel coding constraint $K + k_{m,K}^*$.

4. With the optimal source code vector being monotonically increasing and convex distortion the number of possible channel symbols can be further reduced as

$$\overline{\mathcal{Q}}_{m,K}^* = \max_{k:1\le k\le n \,\wedge\, k\ge k_{m-1}^*(K-k)} \left(\overline{\mathcal{Q}}_{m,N-k}^* + \left(\mathcal{Q}\left(K_m\right) - \mathcal{Q}\left(K_{m-1}\right)\right) P_n\left(k_m\right) \right). \tag{5.34}$$

   where $k_{m-1,K-k}^*$ is the optimal source code for the previous layer whereby $K_{0,0}^* = 0$.

5. The unconstrained optimization is solved by

$$\overline{\mathcal{Q}}_{\max} = \max_{N:M\le K\le Mn} \overline{\mathcal{Q}}_{m,K}^*. \tag{5.35}$$

6. Finally, the optimal source code vector $\mathbf{k}$ is the vector achieving $\overline{\mathcal{Q}}_{\max}$ in (5.35).

For the performance of this algorithm in comparison to other algorithms for progressive image transmission we refer the interested reader to [DWW04]. We will apply this algorithm for the transmission of PTVC coded video over packet–lossy channels in section 6.3.2.

5.4 Progressive Image Transmission over Fading Channels

5.4.1 System Description

In the following, we apply a regressive redundancy system with a FEED–based decoding algorithm to transmit progressively coded sources with an assigned quality–rate function $\mathcal{Q}(R)$ over a channel with a priori unknown channel state. The system setup has already been used for the performance evaluation in subsection 4.4.4. We assume that the source message is packetized into packets, each of length $k = \omega_p$ with ω_p the puncturing period of the convolutional code in use.

Figure 5.8 shows the considered system: An image source s is encoded using the progressive image encode SPIHT. This results in an encoded bit stream $\mathbf{u}$ and an importance $\mathcal{I}_m$ for each virtual

layer m. Using a rate allocation (see subsection 5.4.2) an optimized channel symbol vector $\mathbf{n}^*$ is obtained. This vector is used to encode the source message $\mathbf{u}$ with a high-memory convolutional code together with the application of UEP according to the principles introduced in section 4.4. After transmission, the FEED decoder generates an estimate of the received sequence, $\hat{\mathbf{u}}$ as well as a reliability vector for this message, $\mathcal{R}_{\psi,\mathcal{U}}(\hat{\mathbf{u}})$. By use of a shortening function $\mathcal{F}_s(\hat{\mathbf{u}}, \mathcal{R}, \mathcal{R}_0)$ according to (4.52) with minimum reliability $\mathcal{R}_0$ we obtain shortened reliable vector $\hat{\mathbf{u}}_{(0:\tau]}$, which is then forwarded to the SPIHT decoder to reconstruct source at the receiver, $\hat{s}$.

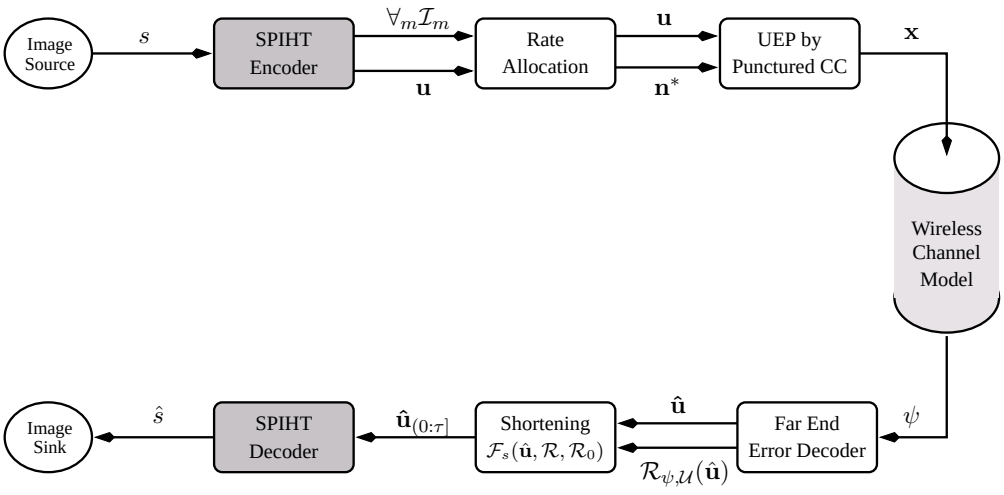


Figure 5.8: Progressive image transmission system using UEP and FEED.

The regressive redundancy profile is implemented by applying puncturing to a rate $1/c$ convolutional code such that we have access to a set of code rates in $\mathcal{S}_r \triangleq \{\omega_p/(\omega_p + i)\}_{i=0}^{\omega_p(c-1)}$ with minimum code rate $r_{\min} = \omega_p/n_{\max} = \omega_p/(\omega_p c)$ and maximum code rate $r_{\max} = \omega_p/n_{\min} = \omega_p/\omega_p = 1$. The channel coding symbol can be selected from the set $\mathcal{S}_n = \{\omega_p, \omega_p + 1, \ldots, c\omega_p\}$. We possibly include $n = 0$ in the set to indicate that a certain message/layer is not transmitted at all. With a fixed packetization of equal length k for each layer m the rate allocation on each level is determined by the channel symbol vector $\mathbf{n}$. For notational convenience we define the channel coding vector up to layer m as $\mathbf{n}_m = \{n_1, \ldots, n_m\}$ and the sum over all components of vector $\mathbf{n}_M$ as $N_m(\mathbf{n}_M) \triangleq \sum_{i=1}^{m} n_i$ with $m = 1, \ldots, M$ and $N_0(\mathbf{n}_M) = 0$ by definition.

5.4.2 Optimized Rate Allocation

We are in the following interested in finding an appropriate channel symbol vector given a certain quality–rate function $\mathcal{Q}(R)$, or equivalently the importance $\mathcal{I}_m = \mathcal{Q}(mk) - \mathcal{Q}((m-1)k)$ for each layer $m = 1, \ldots, M$ given a total amount of channel symbols N_{tot}. To be more specific, we are interested in a channel symbol vector which maximizes the expected received quality $\overline{\mathcal{Q}}(\mathbf{n}) \triangleq \mathbb{E}_H\left\{\hat{\mathcal{Q}}(\mathbf{n})\right\}$, i.e.,

$$\mathbf{n}^* = \arg\max_{\mathbf{n} \in \mathcal{S}_n^M} \overline{\mathcal{Q}}(\mathbf{n}), \tag{5.36}$$

subject to $N_M(\mathbf{n}^*) \leq N_{\text{tot}}$. In [CF99] for example, an appropriate rate allocation has been discussed for a given set of error probabilities $p_d(k, r)$ for RCPC codes assuming a sufficiently long CRC and perfect error detection. The channel state was assumed to be constant, but losses occur

due to the imperfectness of the code. In our approach [SW01] as we apply high–memory convolutional codes and FEED decoding. Therefore, we assume asymptotically perfect channel codes operating at the cutoff rate, but losses occur due to the variable channel states and we use the expression in (4.15). The decoding probability when applying a channel symbol vector n is defined as

$$p_d(n) \triangleq \mathbb{E}_H \left\{ \mathbf{1} \left\{ R_0(H) < \frac{k}{n} \right\} \right\}. \tag{5.37}$$

Therefore, for the expected quality $\overline{\mathcal{Q}}(\mathbf{n})$ we can use a slightly modified version of (4.16) as

$$\overline{\mathcal{Q}}(\mathbf{n}) = \mathcal{Q}_0 + \sum_{m=1}^{M} \mathcal{I}_m p_d \left(\min_{i=1,\ldots,m} n_i \right). \tag{5.38}$$

Although the evalution of the equation (5.38) seems to be rather simple for reasonably small $k = \omega_p$ as well as a reasonable amount of source bits, Mk, a brute force search over all possible combinations is again complex as $|\mathcal{S}_r|^M$ combinations of channel symbol vectors have to be evaluated.

Fortunately, if the quality–rate function $\mathcal{Q}(R)$ or at least the packetized version expressed by the importance $\mathcal{I}_m$ is monotonically increasing and convex it can be shown that the maximization problem in (5.36) can be solved exactly by a framework based on dynamic programming as already introduced in subsection 4.2.2 for maximum likelihood decoding of convolutional codes and in subsection 5.3.2 for the UXP case. Furthermore, as the detected error probability $p_d\,(n)$ is monotonically decreasing in n, the search can be simplified: It is easily shown by exchange arguments that the optimal channel code vector $\mathbf{n}^*$ which maximizes the expected quality in (5.38) is monotonically decreasing, i.e.,

$$\forall_{m=2,\ldots,M} \quad {n^*}_m \leq {n^*}_{m-1}. \tag{5.39}$$

In the following we will present the optimized rate allocation as an analogy to the Viterbi algorithm. Therefore, equivalently to the cumulative metric in (4.33), we introduce a cumulative expected quality up to layer m

$$\overline{\mathcal{Q}}_m(\mathbf{n}) \triangleq \mathcal{Q}_0 + \sum_{i=1}^{m} \mathcal{I}_i p_d\,(n_i)\,. \tag{5.40}$$

Then, together with the monotonicity of p_d in (5.39), the problem in (5.36) subject to the total rate constraint is reduced to

$$\max_{\mathbf{n}} \overline{\mathcal{Q}}_m(\mathbf{n}) \quad \text{subject to} \quad N_m(\mathbf{n}) \leq N, \tag{5.41}$$

for $m = M$ and $N = N_{\text{tot}}$. Similar to the recursions in (4.35) for the Viterbi decoder, we define for $m = 1, \ldots, M$

$$\overline{\mathcal{Q}}_m(\mathbf{n}_M) \triangleq \overline{\mathcal{Q}}_{m-1}(\mathbf{n}_M) + \mathcal{I}_m p_d(n_m), \tag{5.42}$$

$$N_m(\mathbf{n}) \triangleq N_{m-1}(\mathbf{n}) + n_m, \tag{5.43}$$

with $\overline{\mathcal{Q}}_0(\mathbf{n}_M) = \mathcal{Q}_0$ and $N_0(\mathbf{n}_M) = 0$. Again aligned to the metrics for the Viterbi decoder, neither the channel code vector $\mathbf{n}$ nor $\overline{\mathcal{Q}}_m(\mathbf{n}_M)$ nor $N_m(\mathbf{n})$ do depend on $n_{m+1}, \ldots, n_M$. Therefore, the solution of the minimization problem in (5.41) needs to be determined only over a subsequence of channel coding symbols $\mathbf{n}_m$.

With these preliminaries we are able to determine an optimized rate allocation. Let $\mathbf{n}_m^*$ be the solution for the optimization problem in (5.41) and let $\overline{\mathcal{Q}}_{m,N}^*$ with $N = \sum_{i=1}^{m} n_i^*$ denote the optimum value of the objective function in (5.41). Therefore, $\mathbf{n}_m^*$ is the subsequence of channel codes achieving the maximum in (5.41) for a layer index m and a rate constraint N. Then the following results are observed.

1. $\overline{\mathcal{Q}}_{m,N}^*$ satisfies the dynamic programming equation

$$\overline{\mathcal{Q}}_{m,N}^* = \max_{n \in \mathcal{S}_n} \left(\overline{\mathcal{Q}}_{m,N-n}^* + \mathcal{I}_m p_d(n) \right). \tag{5.44}$$

2. The channel symbol $n_{m,N}^*$ achieves the maximum in (5.44).

3. The subsequence $\mathbf{n}_{m+1}^*$ solves (5.41) for layer index $m+1$ and channel coding constraint $N + n_{m,N}^*$.

4. With the optimal channel symbol vector being monotonically increasing we can further reduce the number of possible channel symbols as

$$\overline{\mathcal{Q}}_{m,N}^* = \min_{n \in \mathcal{S}_n \wedge n \wedge\, n \leq n_{m-1,N-n}^*} \left(\overline{\mathcal{Q}}_{m,N-n}^* + \mathcal{I}_m p_d(n) \right), \tag{5.45}$$

   where $n_{m-1,N-n}^*$ is the optimal channel code for the previous layer and $n_{0,0}^* = n_{\max}$.

5. Finally the maximum expected quality is found by as $\overline{\mathcal{Q}}_{\max} = \overline{\mathcal{Q}}_{M,N}^*$ and the optimal channel symbol vector $\mathbf{n}^*$ is the vector achieving $\overline{\mathcal{Q}}_{\max}$.

Modifications of this algorithms are easily possible to obtain for example a minimum overall distortion or a maximum average rate. Furthermore, the algorithm can be modified such that source rate allocation is included or excluded in the optimization process. If the channel symbol $n = 0$ is excluded from set $\mathcal{S}_n$, then all layers $m = 1, \ldots, M$ are transmitted. We refer to this optimization as *channel–optimized rate allocation*. In case of inclusion of $n = 0$, the number of transmitted layers, $\bar{M} \triangleq \{\arg\max_{m=1,\ldots,M} n_m \wedge n_m \neq 0\}$ might be different to M. The basic idea is to let M be as large as possible, in our case $M = N_{\text{tot}}/k$ and leave the selection of the number of transmitted layers to the rate allocation process. We refer to this optimization as *source–channel–optimized rate allocation*. In recent years several researchers have focussed on optimized rate allocation algorithms for the same or similar purposes relying on the principles introduced in [CF99, SAR00, SW01], overviews and further reduction on complexity sacrifying some amount of suboptimalty are presented for example in [DWW04] and [SHX04a].

5.4.3 Selected Experimental Results

In [SZ98] a product code is presented for the protection of SPIHT coded images transmitted over a channel with memory. A similar approach has been proposed in [SAR00]. Both systems use an embedded source code and a product channel code. The row code of the product code is a concatenation of an outer CRC code and an inner RCPC code, while its column code is a systematic Reed–Solomon (RS) code. Both systems use equal error protection along the rows and unequal error protection along the columns. Optimized rate allocation schemes for these scenarios have been presented, among others, in [DWW04, SHX04a, SHX04b] as well as references therein.

In the following, we concentrate on the simulation scenario as proposed in [SZ98]. Simulations were performed for BPSK transmission over a flat–fading Rayleigh channel with a one–dimensional signal–model. The channel is characterized by the average SNR of $\mathbb{E}\{H\}$ of $10\,\mathrm{dB}$ and the normalized Doppler spread of $W_{\mathrm{Dop}} = 10^{-5}$. For the modelling of the correlated fading the method as described in [Jak74] is used. At the decoder, hard–decision decoding is assumed. SPIHT coded images (512×512, grey-scale) were transmitted with a total bit budget of 0.25 bit per pixel (bpp) which results in a total amount of bits $N_{\mathrm{tot}} = 65536$ for source and channel coding. In the simulations at least $N_{\mathcal{C}} = 3000$ independent experiments have been carried out, each with test image 'Lenna'. Several coding schemes have been investigated and the best performance was achieved by a product code with UEP. Additionally, results for a product code with EEP have been reported.

We compare ourselves to the system by using the same source, the same source coder, the same number of total transmission bits, N_{tot}, and the same channel model. We replace the error protection scheme by a FEED–based system with a systematic convolutional code with memory $\mu = 96$, mother code rate $c = 1/7$, and puncturing period $\omega_p = 32$. In addition, as the product code used in [SZ98] includes an inherent interleaver to combat the deep fades, we have added for the same purpose a quadratic block interleaver Π of dimension 256 to spread the generated sequence of encoded layers similar to the radio access scheme presented in Figure 4.15.

The decoding in [SZ98] relies on a hard–decision decoding, so for fair comparison we also adopt this scheme. The signal model according to (2.45) is slightly modified such that

$$y_{f,\Pi(v)} = 1 - 2 \cdot \mathbf{1}\left\{\left(\alpha_{f,v} x_{f,\Pi(v)} + \nu_{f,v}\right) \leq 0\right\},$$

with $\alpha_{f,v}$ the correlated fading amplitudes with normalized power $\mathbb{E}\left\{|\alpha_f|^2\right\} = 1$ and noise samples $\nu_{f,v}$ with variance $\frac{N_0}{2\mathcal{E}_s}$. One could assume that the decoder has at least knowledge on the average bit error rate

$$\mathbb{E}\{\varepsilon\} = \mathbb{E}_\alpha\left\{\frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{\alpha^2\mathcal{E}_s}{N_0}}\right)\right\}.$$

Then, the unique interface to any channel decoder is maintained by computing the channel LLR as for this average bit error probability $\mathbb{E}\{\varepsilon\}$ [HOP96] as

$$\psi_{f,v} = \log_2 \frac{\mathbb{E}\{\varepsilon\}}{1 - \mathbb{E}\{\varepsilon\}} y_{f,v}. \tag{5.46}$$

Note that the performance of some channel decoders such as sequential decoders or maximum a posteriori decoders could be enhanced adapting the channel LLR to the expected bit error rate of each slot ε_f. However, as this knowledge is unnecessary for Viterbi decoding in [SZ98] we also dispense with this additional information. For our specific case with $\mathbb{E}\{H\}$ of $10\,\mathrm{dB}$, the expected bit error rate results in $\mathbb{E}\{\varepsilon\} = 0.023$.

In the following, we use two performance measures as defined in subsection 2.1.2 for video transmission, but as $N_s = 1$ we skip the index for the frame number t. We use the expected PSNR, $\overline{\mathrm{PSNR}} = \mathbb{E}_{\hat{\mathcal{C}}}\left\{\overline{\mathrm{PSNR}}_{\hat{\mathcal{C}}}\right\}$, as defined in (2.12) as well as the PSNR of the expected MSE, $\mathrm{PSNR}_{\overline{\mathrm{mse}}} = \mathrm{PSNR}\left(\mathbb{E}_{\hat{\mathcal{C}}}\left\{d\left(\hat{\mathcal{C}}\right)\right\}\right)$. Both measures are used as objective function in the rate allocation procedure as well as to report the results. FEED–based decoding is performed using the Fano algorithm with different amount of operations per information bit, N_{op}. Shortening on the output of the FEED decoder is applied with the reliability threshold $\mathcal{R}_0 = 0.1$. Although this threshold is rather low, it was observed that this value provides the best overall performance.

Rate Allocation

For the rate allocation we have to solve the problem stated in (5.41). With the knowledge of the individual importance $\mathcal{I}_m$ as well as the constraints on the channel coding rates with $n_{\max} = 224$ we require an appropriate modelling on the probabilities $p_d(n)$ as defined in (5.37). More precisely, the modelling in (5.37) requires a distribution of the channel states. In combination with the block interleaver, the correlated fading channel can be viewed as a block fading channel not with varying SNR but with varying bit error rate ε_f for the transmission of each individual image f. We are interested in the distribution of the bit error probability ε for this specific channel assuming a block length of $N_{\text{tot}} = 65536$. This distribution – obtained by simulations [HSWD00] – is used together with the cutoff rate for the binary symmetric channel, $R_0(\varepsilon) = 1 - \log_2(1 + 2\sqrt{\varepsilon(1-\varepsilon)})$ to compute the probability $p_d(n)$ in (5.37). With these preliminaries we perform the rate allocation according to subsection 5.4.1.

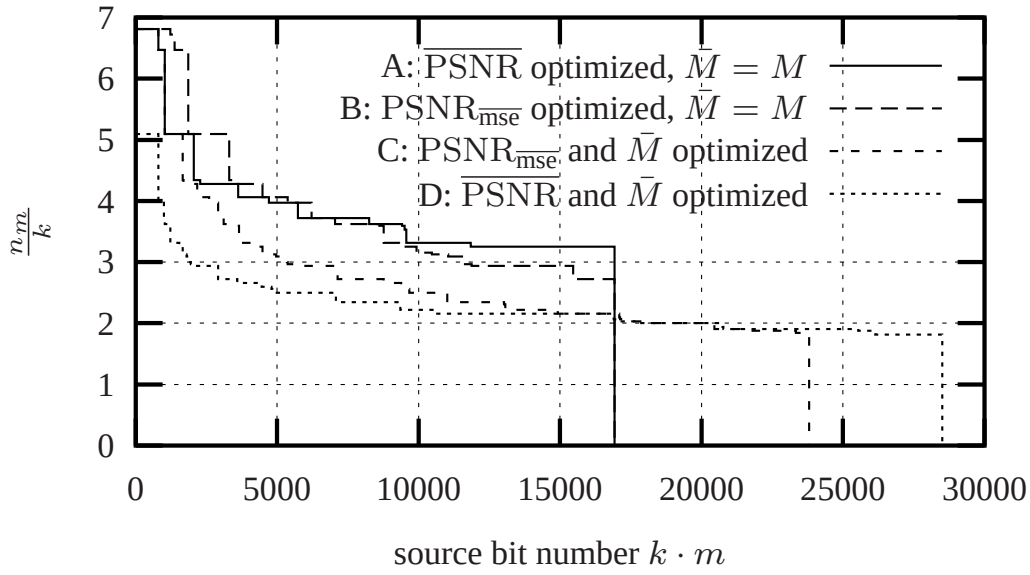


Figure 5.9: Rate allocation for different scenarios assuming progressive transmission of SPIHT coded 'Lenna' image over a channel with SNR $\mathbb{E}\{H\} = 10\,\text{dB}$, Doppler frequency $W_{\text{Dop}} = 10^{-5}$, hard–decision decoding and assuming FEED–based decoding.

Figure 5.9 shows results of the rate allocation procedure for four different scenarios. The inverse code rate n_m/k is shown over the source rate mk. For scenario A and B the source bit-rate mk has been fixed such that it is equal to the product code (PC) with UEP in [SZ98] and error–free transmission for both systems results in identical quality. This is accomplished by disallowing $n = 0$ in the rate allocation and restricting M such that the same source rate Mk is obtained as for rate allocation. The optimization in case A is based on the maximization of $\overline{\text{PSNR}}$. For scenario B, the optimization is based on minimizing the average distortion. Similarly, in scenario C and D the optimization is based on the average PSNR and the expected distortion, respectively. However, here the total number of transmitted layers $\bar{M}$ and therefore the overall source rate $\bar{M}k$ can be adjusted by adding $n = 0$ in the rate allocation. In general, the optimization based on the expected distortion tends to protect the earlier bits better than the one based on the average PSNR. This is obvious as for the expected distortion the decoding of a low bit-rate image contributes much more to the overall degradation than in the case of the average PSNR.

Simulation Results

Simulations have been carried out for all rate allocation schemes in Figure 5.9. Additionally, the receiver complexity was varied by providing a different amount of operations per information bit, N_{op}, for the FEED–based channel coder. The complexity for the product code was assumed to be mainly determined by the Viterbi decoder for the convolutional code with memory $\mu = 6$.

Table 5.2: Performance results for different coding schemes and work loads compared to bounds and existing solutions [SZ98].

Scheme	N_{op}	$\overline{\text{PSNR}}$	$\text{PSNR}_{\overline{\text{mse}}}$
PC EEP [SZ98], $M = 19648$	> 64	28.23 dB	28.05 dB
PC UEP [SZ98], $M = 16928$	> 64	28.19 dB	28.15 dB
FEED A $k\bar{M} = 16928$	bound	28.37 dB	28.16 dB
$\text{PSNR}_{\overline{\text{mse}}}$ optimized	16	27.85 dB	27.53 dB
FEED B $k\bar{M} = 16928$	bound	28.38 dB	28.14 dB
$\overline{\text{PSNR}}$ optimized	16	27.82 dB	27.41 dB
FEED C $k\bar{M} = 23808$	bound	29.25 dB	28.70 dB
	4	28.78 dB	28.25 dB
$\text{PSNR}_{\overline{\text{mse}}}$ optimized	16	28.84 dB	28.34 dB
	64	28.87 dB	28.37 dB
FEED D, $k\bar{M} = 28512$	bound	29.54 dB	28.11 dB
	4	29.25 dB	28.59 dB
$\overline{\text{PSNR}}$ optimized	16	29.32 dB	28.61 dB
	64	29.36 dB	28.76 dB

The results for the average PSNR, $\overline{\text{PSNR}}$, and the PSNR of the average distortion, $\text{PSNR}_{\overline{\text{mse}}}$, are shown in Table 5.2. Also included are the results for the product code as presented in [SZ98] for EEP and UEP[4]. For fixed source rate the PC UEP outperforms the FEED approach by about $0.2 - 0.6$ dB for both measures. This is because the sequential decoder is more sensitive to a burst of bit errors than the product code. This behavior is explained in more detail in [HSWD00].

For the adapted source rate schemes C and D we can see significant gains of the proposed system when compared to the product code. For equal complexity, gains up to 1.2 dB are possible. An additional effect is the complexity scalability of our receiver. If we reduce the number of operations per information bit to about 10 % of the product code, the gain to the high complexity receiver is reduced by at most 0.2 dB. This means that the proposed FEED system is capable to trade receiver complexity and decoded quality. This novel property is very interesting for applications where different types of receivers are present in the system. Further discussion and detailed results are presented in [HSWD00].

[4]In [SZ98] only the average PSNR is given. However, the average PSNR was computed different as a different $\mathcal{Q}_0$ has been used. In [SZ98] the failure of a transmission was more penalized by setting $\mathcal{Q}_0 = 0$ whereas in our case concealment with a grey image has been assumed resulting in $\mathcal{Q}_0 = 14.2$ dB. We have adjusted the results in [SZ98] to obtain comparable measures.

5.5 Summary

In this chapter several schemes which combine progressively coded source and unequal error protection have been introduced. The following observations and findings are of major importance:

- A transmission system incorporating a progressively coded source together with unequal error protection has been introduced. The transmission is flexible in a sense that the number of layers, M, the source bits for each layer k_m, and the channel coding rate for each layer r_m can be chosen independently.

- The transmission system also includes a single layer case with equal error protection by using $M = 1$. It also includes practical cases for which the amount of source bits for each layer is equal, or that the number of channel symbols n_m are the same for each layer.

- Information-theoretic considerations have been used to assess such a system when applied on a binary-input symmetric channel, e.g. a binary symmetric channel or an AWGN channel with BPSK modulation. A progressive vector quantizer according to (3.11) is combined with channel codes with error probability performing at the Gallager bound according to (4.2). By applying high-resolution assumptions and techniques (see [ZM94] and [HZ97] for the usage in single layer systems only), a distortion function for a multi-layer system with M layers with fixed channel coding rates $\mathbf{r}_M$ has been derived in (5.5).

- The optimized code rates $\mathbf{r}^*{}_M$ minimizing the distortion in (5.5) have been derived in (5.7) and the corresponding minimum distortion is provided in (5.8).

- For the specific case with equal transmission block length $n_m = n/M$ for each layer a recursive method to derive the optimal code rates has been proposed. From this it is obvious that for a multi-layer system unequal error protection with increasing code rates minimizes the end-to-end distortion. However, it was also shown in (5.20) and (5.21) that for identical overall transmission rate there is a penalty introducing layered transmission. To obtain the same end-to-end distortion, the rate for a layered system with M layers needs to be increased roughly by a factor M to obtain the same performance as for the single layer case.

- For the specific case with equal source block length $k_m = k/M$ for each layer the optimal code rates can be determined directly by (5.22). In (5.24) and Figure 5.5, the rate increase to obtain the same end-to-end distortion when introducing layers has been shown. The results on the penalty when using multistage transmission and unequal error protection are also supported by a simple practical experiment. A more detailed discussion on the findings is provided in subsection 5.2.8.

- In section 5.3 a framework for Unequal Erasure Protection (UXP) in combination with progressively coded sources based on Reed-Solomon codes has been introduced. In combination with progressively coded sources, this combined source–channel coding system can also be viewed as a multiple description coding system [WWZ80].

- Given a certain quality–rate function $\mathcal{Q}(R)$, the expected quality for such a system taking into account the packet loss rate of the channel has been derived in (5.27). The quality is mainly determined by the number of levels as well as the selected source coding vector $\mathbf{k}$ under some rate constraint (5.25).

- In subsection 5.3.2, an optimized source coding vector $\mathbf{k}^*$ minimizing the distortion in (5.27) is derived. Under some non-restrictive assumptions, it can be shown that this optimization can be solved by a dynamic programming approach in (5.33).

- The constrained optimization for a fixed number of layers can be extended by an unconstrained optimization according to (5.35). In this case not only the code rates of each layer are determined, but also the optimum number of layers is selected.

- In section 5.4 a system for progressive image transmission over fading systems is introduced. The system uses the Far End Error Decoding (FEED) decoding algorithm as proposed in section 4.4 together with unequal error protection and shortening.

- For appropriate configuration of the redundancy profile, a rate allocation is formulated in (5.36) that searches for the appropriate code rate, or equivalently the number of channel symbols n_m for each layer by the application of puncturing. Note that each layer corresponds to a fixed size puncturing period.

- The optimized channel symbol vector $\mathbf{n}^*$ minimizing the distortion in (5.38) is derived. Under some non-restrictive assumptions, it can be shown that this optimization can again be solved by a dynamic programming approach in (5.44).

- Again, this algorithm can be modified to include source rate allocation by the inclusion of the channel symbol $n = 0$ to the set $\mathcal{S}_n$. We refer to this optimization as *source–channel–optimized rate allocation* whereas for the case for which $n = 0$ is excluded from set $\mathcal{S}_n$, the only the channel coding rates are optimized. This is referred to as *channel–optimized rate allocation*.

- To verify the performance of the FEED algorithm as well as the optimized rate allocation, a progressive image transmission system including SPIHT has been setup. The exact same conditions as proposed in [SZ98] have been used. For fixed source rate the reference system based on a product code UEP slightly outperforms the FEED approach. However, by the use of source–channel–optimized rate allocation the FEED-based algorithm outperforms the reference system significantly at the same decoding complexity.

- In the same set of experiments it is also shown that if we reduce decoder complexity to about 10 % of reference system, the FEED is still significantly better than the reference system. This means that the proposed FEED system is capable to trade receiver complexity and decoded quality. This novel property is very interesting for applications where different types of receivers are present in the system, for example for broadcast systems with a heterogeneous receiver population.

6

Video Transmission in Packet-Lossy Environment

In this chapter we will discuss video transmission in packet–lossy environment mainly over generic or Internet channels. Nevertheless the methods introduced here are applicable to mobile scenarios as well and the most promising features will be used in some system designs for mobile video transport in chapters 7, 8 and 9. Specific focus is on the use of H.264/AVC and the PTVC.

6.1 H.264/AVC-Compliant End-to-End Systems

6.1.1 Formalization of H.264/AVC Packetized Video

By the use of slices and slice groups as introduced in subsection 3.3.2, H.264/AVC provides a flexible and efficient syntax to map the N_{MB} MBs of each frame t of the image sequence to individual NAL units. According to the RTP payload specification [WHS+05] each NAL unit is carried in a single RTP packet[1]. The encoding of s_t results in one or several RTP packets again being abstracted by a data unit $\mathcal{P}_i$ with sequence number i. Therefore, each macroblock $b = 1, \ldots, N_{\text{MB}}$ of source unit s_t is assigned to a certain data unit $\mathcal{P}_i$ specified by the mapping $\mathcal{M}_{b,t} = i$. We recall the definition of the greatest index i corresponding to source s_t as i_t. The considered video transmission system is shown in Figure 6.1. Assume that the data units $\mathcal{P}_i$ are transmitted over a channel which either delivers the data unit $\mathcal{P}_i$ correctly, indicated as $\mathcal{C}_i = 1$, or the data unit is lost, i.e., $\mathcal{C}_i = 0$. A data unit is also assumed to be lost, if it is received after its DTS has expired, i.e., $\mathcal{C}_i = \mathbf{1}\{\tau_{r,i} \leq \tau_{\text{DTS},i}\}$. We do not consider highly complex concepts with multiple deadlines [Gha96, KRG03] in which late data units are processed by the decoder to at least update the reference buffer resulting in reduced long–term error propagation.

[1] Additionally, [WHS+05] also specifies Single–Time Aggregation Units (STAPs) and Multiple–Time Aggregation Units (MTAPs) to allow the aggregation of individual NAL units of the same frame or even different frames, respectively, into a single RTP packet. Furthermore, Fragmentation Units (FUs) are defined which allow to fragment an NAL unit into multiple RTP packets. Whereas MTAPs are unacceptable in low–delay applications, STAPs and FUs only provide hooks for network friendliness and are basically included in the presented framework.

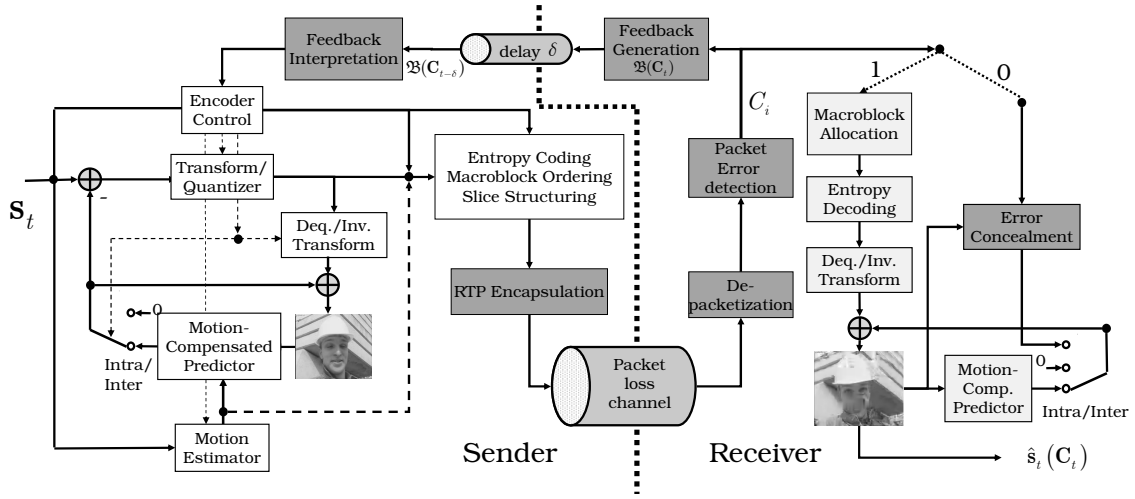


Figure 6.1: Video transmission system in packet loss environment.

At the receiver, due to the coding restriction of slices and slice groups as well as with the information in RTP and slice headers, the decoder is able to reconstruct the information of each correctly received NAL unit and its encapsulated slice. The decoded MBs are then distributed according to the mapping $\mathcal{M}$ in the frame. For all MBs positions, for which no data has been received, appropriate error concealment has to be invoked before the frame is forwarded to the reference and the display buffer. The decoded source $\hat{s}_t$ obviously depends on the channel behaviour affecting data units $\mathcal{P}_{i_{t-1}+1}, \ldots, \mathcal{P}_{i_t}$ corresponding the current frame s_t, but due to the predictive coding and error propagation in general also on the channel behaviour all previous data units, $\mathcal{C}_t \triangleq \mathcal{C}_{[1:i_t]}$. This dependency is expressed as $\hat{s}_t(\mathcal{C}_t)$.

Due to the bi–directional nature of conversational applications a low–delay, low bit-rate, error–free feedback channel from the receiver to the transmitter can be assumed as indicated in Figure 6.1. This feedback link allows sending some back channel messages $\mathcal{B}(\mathcal{C}_t)$. In our framework we model the feedback link as error–free, but the feedback message delay is normalized to the frame rate such that $\mathcal{B}(\mathcal{C}_{t-\delta})$ expresses a δ frames delayed version of $\mathcal{B}(\mathcal{C}_t)$ with $\delta = 0, 1, 2, \ldots$. The exploitation of this feedback link by using different messages types is discussed later in this section.

6.1.2 Error Concealment

Error concealment is a *non–normative* feature in any video decoder. There exists a vast literature dealing with different levels of error concealment, some of them very simple, some of them very complex. The basic idea is that the decoder should generate a representation for the lost area which matches perceptually as well as possible the lost information without knowing the lost information itself. For this, the surrounding area in the spatial and temporal domain is exploited. Usually, macroblock based error concealment is used starting from reliable neighbours and recursively concealing more and more macroblocks by also using already concealed macroblocks. In the H.264/AVC test model software two types of error concealment algorithms have been introduced [VHW01, WHV+02, SHW03], one exploiting spatial information only suitable mainly for intra frames and one exploiting temporal information. Both algorithms provide good performance with manageable complexity.

The spatial concealment method is based on weighted pixel value averaging [SSD98]: In this

case each pixel value in a macroblock is concealed by the weighted sum of the closest boundary pixels of the available adjacent macroblocks with the weight being relative to the inverse distance between the pixel to be concealed and the boundary pixel. The temporal concealment uses temporally and spatially surrounding motion vectors to perform motion compensation [CCW97]. The selection of the appropriate motion vectors is based on the boundary–matching techniques [LRL93]. Especially for reasonably good subjective quality, it has been recognized [Zha04a], that it is important to select the appropriate error concealment technique, spatial or temporal. We have extended the error concealment in the test model software by using the macroblock mode information of the reliable neighbours and concealed neighbors to decide whether spatial error concealment or temporal error concealment is more suitable. We refer to this error concealment as Adaptive temporal and spatial Error Concealment (AEC) in the remainder.

Figure 6.2 shows the performance of different error concealment algorithms. From left to right, an example frame is shown when using Previous Frame Concealment (PFC), spatial only, temporal only, and AEC. PFC replaces the missing information by the information at the same location in the temporally preceding frame. Hence, it shows artifacts in the global motion part of the background as well as in the foreground. Spatial error concealment based on weighted pixel averaging smoothes the erroneously decoded image and removes strange block artifacts, but also many details. Temporal concealment relying on motion vector reconstruction with boundary–matching–based techniques keeps details, but results in strange artifacts in uncovered areas. Finally, the AEC keeps many details but also avoids strange block artifacts and is therefore very appropriate with feasible complexity. In the remainder of this work we will use either simple PFC or AEC.

PFC Spatial only Temporal only AEC

Figure 6.2: Performance of different error concealment strategies: PFC, spatial concealment only, temporal concealment only, AEC.

6.1.3 Performance of Basic Error Resilience Tools

The error resilience tools in H.264/AVC have already been introduced in section 3.3.2. In the following we will briefly discuss the performance of basic error resilience tools. In order to assess error resilience tools and to acknowledged the importance of video transmission over packet–lossy channels the H.264/AVC standardization process has adopted a set of common test conditions for IP based transmission [Wen01]. Anchor video sequences, appropriate bit-rates and evaluation criteria are specified. Initially, we will restrict ourselves to a small but representative selection of the common Internet test conditions. We use almost exclusively the QCIF test sequence *foreman*. The first 10 seconds are encoded at a frame rate of $f_s = 7.5$ fps. The video is encoded with VBR rate control according to (3.22) by applying a single quantization parameter[2] $q = 12, 14, \ldots, 30$

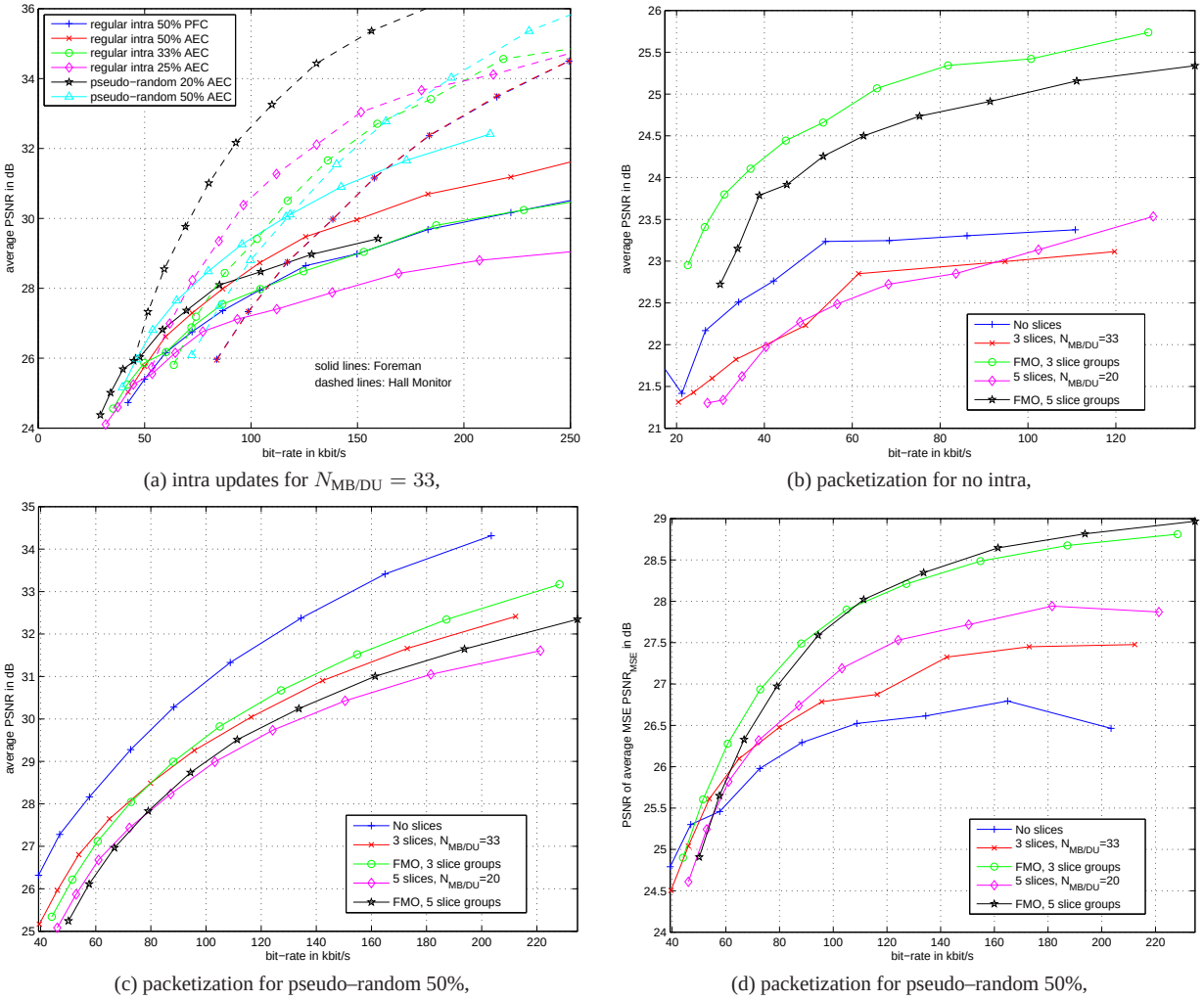
[2]In the remainder of this section we make use of test model JM1.7 that uses QP value mappings of 12 lower than the final standard.

to the entire sequence. The resulting total bit rate R includes a 40 byte IP/UDP/RTP header for each transmitted data unit. As performance measure the average luminance PSNR, $\overline{\mathrm{PSNR}}$, is chosen as suggested in [Wen01] where the average is taken over all encoded frames. For all experiments at least 6000 frames have been transmitted resulting in at $N_{\mathcal{C}} = 80$ channel realizations to obtain sufficient statistics. This is much more as suggested in [Wen01]. Initially we concentrate exclusively on channel error pattern 10 as introduced in subsection 2.4.5.

In the presence of errors it has been recognized that the introduction of more frequent non–predictively coded image parts is of major importance. In early work on the latter subject, e.g., [ZK99, HM92, LV96], it has been proposed to introduce intra–coded MBs, regularly, randomly or preferably in a certain pseudo–random update pattern. Figure 6.3a) shows the performance of regular intra updates as well as pseudo–random with different update ratios in for test error pattern 10, test sequence *foreman* and *hall monitor*, and slice packetization with $N_{\mathrm{MB/DU}} = 33$. From the results for different intra update ratios it is obvious that removing error propagation is essential in error–prone video transmission. An appropriate intra–update ratio can increase the overall quality significantly. Also the placement of intra updates is important, pseudo–random patterns perform better than regular patterns. Also immediately apparent is that the optimal update ratio depends on different parameters, namely the sequence characteristics and the transmission bit-rate as seen from Figure 6.3a), but obviously also on the channel characteristics. Simple adaptive schemes are presented for example in [LV96], but rate–distortion optimized intra updates are introduced in subsection 6.1.4.

In addition, the objective performance of different error concealment strategies are assessed: the results show that already for moderate movement as present in *foreman* a significant gain for AEC compared to the PFC is visible in terms of objective quality as already indicated subjectively in Figure 6.2. A second class of common error resilience tools proposes the introduction of slices, and the concept has been extended by FMO in H.264/AVC to provide the possibility of mapping MBs to data units in an arbitrary manner. In the following, whenever we refer to FMO we exclusively use checkerboard patterns. Then, for the same test scenario – test sequence *foreman*, error pattern 10, AEC – Figure 6.3b) shows the average PSNR, $\overline{\mathrm{PSNR}}$, over the bit-rate for different packetization schemes without applying any other error resilience features, i.e., no intra updates. The overall performance is very low. However, whereas for slice structured coding no benefits compared to the system without slice structuring are noticeable, the benefits of FMO are obvious. Note that a higher number of slice groups – 5 compared to 3 slice groups – does not provide higher quality as more packetization overhead is required and less prediction options are accessible. The same set of curves are shown in Figure 6.3c) except that the pseudo–random intra update ratio is set to the best value of Figure 6.3a), namely 50%. First of all it is noticed that all schemes provide significantly better performance than schemes without any intra updates. However, in contrast the scheme without other error resilience, any packetization performs worse than the scheme without packetization. Two reasons can be given for this behaviour: First of all, the coding without slices provide better compression efficiency[3]. The second and more severe reason results from the fact that as the same packet loss rate is observed for each scheme, regardless of the packet size and packet frequency, it is more likely that a frame is corrupted than for the case without packetization where the loss of a data unit loss results in the loss of an entire frame. However, without packetization also many more frames with good reconstruction quality are present, especially if a high intra update ratio is used. Note that these objective results also match quite well the subjective observations as the rare loss of entire frames is most of the time less disturbing than more frequent

[3]This behaviour can be observed from the graph as each curve shows the dots for the same quantizers. For the scheme without packetization the dots are further left than the ones with packetization.



(a) intra updates for $N_{\text{MB/DU}} = 33$,

(b) packetization for no intra,

(c) packetization for pseudo–random 50%,

(d) packetization for pseudo–random 50%,

Figure 6.3: PSNR over bit-rate performance of different error resilience tools in H.264/AVC for test error pattern 10 for QCIF test sequence *foreman* at a frame rate of $f_s = 7.5$ fps (a) average PSNR, $\overline{\text{PSNR}}$,for different update modes, different intra update ratios, and different error concealment strategies compared to test sequence *Hall Monitor* at $f_s = 15$ fps with slice structured coding and $N_{\text{MB/DU}} = 33$, (b) average PSNR, $\overline{\text{PSNR}}$,for different packetization schemes and no additional intra updates. (c) average PSNR, $\overline{\text{PSNR}}$,for different packetization schemes and 50% pseudo–random intra updates. (d) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$ for different packetization schemes and 50% random intra updates.

slightly distorted frames with block artifacts. Nevertheless, for the same set of curves as shown in Figure 6.3c) we also present the results based on the PSNR of the average MSE, $\text{PSNR}_{\overline{\text{mse}}}$, in Figure 6.3c). This measure takes into account more appropriately the problems of single highly distorted frames. In this case any kind of packetization is superior to the case where a single frame represents an entire data unit with FMO outperforming regular slice structured coding. The optimum number of slice groups depends on the bit-rate. An appropriate selection of tools such as intra updates as well as packetization is necessary, depending on the application quality measure, the sequence characteristics, the channel characteristics and possibly other constraints. Some method for appropriate selections are discussed in the following.

6.1.4 Operational Encoder Control in Error–Prone Environment

Framework and Mode Selection Process

The tools for increased error resilience in H.264/AVC just as in any other hybrid video coding, in particular those to limit error propagation, do not significantly differ from the ones used for compression efficiency. Features like multi–frame prediction or intra coding of individual MBs are not primarily error resilience tools. They are mainly used to increase coding efficiency in error–free environments providing a design freedom left to the video encoder. This also means that bad decisions at the encoder can lead to poor results in coding efficiency or error resiliency or both. For compression efficiency operational encoder control based on Lagrangian multiplier techniques has been proposed in (3.19) to select motion vectors, reference frames, and macroblock modes. Obviously, it is contradictory, if the same decision criteria are used to obtain good selections for compression efficiency and error resilience.

Therefore, it has been proposed [ZRR00, CSK00, WFSG00] to modify the selection of the coding modes according to (3.19) to take into account the influence of the lossy channel. When encoding macroblock b with a certain coding mode m_b, it is suggested to replace the encoding distortion $d_{b,m}$ by the p–th power decoder distortion[4]

$$\tilde{d}_{b,m}(\mathcal{C}_t) \triangleq ||\mathbf{s}_{b,t} - \hat{\mathbf{s}}_{b,t}(\mathcal{C}_t, m)||^p \,, \tag{6.1}$$

which obviously depends on the reconstructed pixel values $\hat{s}_t(\mathcal{C}_t, m)$ and therefore also on the channel behaviour $\mathcal{C}_t$ and the selected coding mode m.

In general, the channel behaviour is not deterministic and the realization, $\mathcal{C}_t$, observed by the decoder is unknown to the encoder which does not allow to determine the decoder distortion in (6.1) at the encoder. However, we can assume that the encoder might at least have some knowledge on the statistics of the random channel behaviour, denoted as $\hat{\mathcal{C}}_t$. In an RTP environment, RTCP for example uses this channel to send receiver reports on the experienced loss and delay statistics which allows the encoder to incorporate this statistics in the encoding process. Assume that the statistics on loss process are perfectly known to encoder, i.e., $\mathcal{B}(\mathcal{C}_t) = \hat{\mathcal{C}}_t$ and the loss process is stationary such that it is independent of the delay. Then, the encoder is at least able to compute the expected distortion

$$\overline{d}_{b,m} \triangleq \mathbb{E}_{\hat{\mathcal{C}}_t}\left\{\tilde{d}_{b,m}(\hat{\mathcal{C}}_t)\right\} = \mathbb{E}_{\hat{\mathcal{C}}_t}\left\{\left|\left|\mathbf{s}_{b,t} - \hat{\mathbf{s}}_{b,t}(\hat{\mathcal{C}}_t, m)\right|\right|^p\right\}. \tag{6.2}$$

It is suggested that for the selection of the macroblock mode using the MSE with $p = 2$ the expected distortion in (6.2) is applied as distortion measure instead of the encoding distortion, i.e.

$$\forall_b \quad {m^*}_b = \arg\min_{m\in\mathcal{O}} \left(\overline{d}_{b,m} + \lambda'_{\mathcal{O},b} r_{b,m}\right), \tag{6.3}$$

where the option set $\mathcal{O}$ is not altered compared to 3.19. In [SWK02] a modification of the mode selection is suggested as follows. Assume that the considered macroblock b has expected loss probability $p_{\mathrm{MB},b}$. Then, with similar techniques as used in [WG01] to obtain an indication on the selection of the Lagrangian parameter for error–free transmission, it is shown that the Lagrangian parameter for error–prone transmission should be selected as

$$\lambda'_{\mathcal{O},b} = (1 - p_{\mathrm{MB},b})\alpha(\hat{\mathcal{C}}_{t-1})\lambda_{\mathcal{O}},$$

[4]Generalization to other distortion measures is easily possible.

with $\alpha(\hat{\mathcal{C}}_{t-1}) \leq 1$ some value depending on the channel statistics of the previous frame and equal to 1 for error–free transmission. Data units usually include entire slices that are independently encoded. Furthermore, the error concealment for the current MB is independent of the mode selection for the current MB. In addition, assuming that the error concealment is independent of the mode selection of previously coded MBs in this frame[5], then the mode selection in (6.3) for macroblock b is equivalent to

$$\begin{aligned} m^*{}_b &= \arg\min_{m\in\mathcal{O}} \left((1-p_{\text{MB},b})\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1}) + p_{\text{MB},b}\overline{d}_{0,b}(\hat{\mathcal{C}}_{t-1}) + (1-p_{\text{MB},b})\alpha(\hat{\mathcal{C}}_{t-1})\lambda_{\mathcal{O}} r_{b,m} \right) \\ &= \arg\min_{m\in\mathcal{O}} \left((1-p_{\text{MB},b})\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1}) + (1-p_{\text{MB},b})\alpha(\hat{\mathcal{C}}_{t-1})\lambda_{\mathcal{O}} r_{b,m} \right) \\ &= \arg\min_{m\in\mathcal{O}} \left(\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1}) + \alpha(\hat{\mathcal{C}}_{t-1})\lambda_{\mathcal{O}} r_{b,m} \right), \end{aligned} \tag{6.4}$$

where $\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1})$ expresses the expected distortion if this MB b is correctly received and the distortion $\overline{d}_b(\hat{\mathcal{C}}_{t-1})$ if this MB is lost. $\overline{d}_b(\hat{\mathcal{C}}_{t-1})$ is independent of the selected mode as the concealment algorithm is obviously independent of the mode decision. In [SWK02] we further showed that the Lagrange parameter could be adapted by selecting $\alpha(\hat{\mathcal{C}}_{t-1})$ depending on the channel statistics and mode decisions in previous frames. However, the benefits are neglible and for simplicity we propose $\alpha(\hat{\mathcal{C}}_{t-1})$ such that the macroblock mode for macroblock b is selected as

$$m^*{}_b = \arg\min_{m\in\mathcal{O}} \left(\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1}) + \lambda_{\mathcal{O}} r_{b,m} \right). \tag{6.5}$$

The computation of the expected distortion $\overline{d}_{b,m}(\hat{\mathcal{C}}_t)$ and $\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1})$ at the encoder is discussed later in this subsection.

A similar procedure might be applied also for the decision on reference frames and motion vectors. The selection of motion vectors based on the expected distortion has been proposed in [YR03]. However, the gains applying this procedure are marginal if it is combined with error resilient MB selection. Therefore, in the remainder we exclusively apply a scheme in which motion vectors and reference frames are obtained based on the encoding distortion. In addition, in case of using multiple reference frames we have proposed a restriction on the reference frames in combination with error–resilient intra macroblock updates [SWK02]. In this case, the area in the reference frames in the rate–distortion optimized reference frame and motion vector selection is restricted such that no pixels are used for prediction which have been intra refreshed due to error resilience reasons in later frames. This restriction can be used in combination with any intra update mode and avoids the anewed appearance of already disappeared erroneous areas. For details of this algorithm and performance results, we refer to [SWK02].

Estimation of Expected Decoder Pixel Distortion

The expected decoder distortion in (6.2) computes as the sum of the individual expected pixel distortions[6] $\mathbb{E}_{\hat{\mathcal{C}}_t}\{|s-\tilde{s}(\mathcal{C}_t)|^p\}$. The estimate of the squared expected pixel distortion, i.e., $p=2$, in packet loss environment has been addressed in several contributions. For example, in [CSK00], [KKK02] and [WFSG00] methods to estimate the distortion introduced due the transmission errors and the resulting error propagation have been proposed. In all these proposals the quantization

[5] This restriction can be removed if also the channel statistics of previous data units of the same frame are considered. However, we neglect this as the implementation is simplified and the error concealment can be applied as post–processing.

[6] For notational convenience we drop the indices on the position of the pixel.

noise and the distortion introduced by the transmission errors, the so–called drift noise, are linearly combined. As the encoder needs to keep track of an estimated pixel distortion, additional complexity and memory is required in the encoder – dependent on the actual method chosen. The addition is approximately one–time the decoder complexity as for each pixel the drift noise has to be computed and stored.

The most recognized method, the so–called Recursive Optimal per–Pixel Estimate (ROPE) algorithm [ZRR00], however, provides an accurate estimation for baseline H.263 and MPEG–4 simple profile in combination with simple temporal error concealment by keeping track of the first and second moment of the decoded pixel value $\tilde{s}\left(\mathcal{C}_t\right)$, $\mathbb{E}\left\{\tilde{s}\left(\mathcal{C}_t\right)\right\}$ and $\mathbb{E}\left\{\tilde{s}^2\left(\mathcal{C}_t\right)\right\}$, respectively. As two moments for each pixel have to be tracked in the encoder the additional complexity of ROPE is approximately twice the complexity of the decoder. However, the extension of the ROPE algorithm to H.264/AVC is not straight–forward. Especially deblocking filter, quarter–pel motion accuracy, and multiple reference frames but also the AEC require taking into account expectations of products of pixels at different positions to obtain an accurate estimation which makes the ROPE infeasible or at least prohibitively complex in this case.

Therefore, a powerful and generic yet complex method has been introduced into the JVT test model to estimate the expected decoder distortion [SWK02]. We have chosen to apply a Monte Carlo like method as discussed in subsection 2.1.2. Similar to (2.14), an estimate of the decoder distortion, $\overline{d}_{b,m}$, in (6.2) is obtained as

$$\overline{d}_{b,m}^{(N_\mathcal{C})} \triangleq \frac{1}{N_\mathcal{C}} \sum_{n=1}^{N_\mathcal{C}} \tilde{d}_{b,m}(\mathcal{C}_{n,t}) = \frac{1}{N_\mathcal{C}} \sum_{n=1}^{N_\mathcal{C}} ||\mathbf{s}_{b,t} - \hat{\mathbf{s}}_{b,t}(\mathcal{C}_{n,t}, m)||^p \,. \tag{6.6}$$

with $\mathcal{C}_{n,t}, n = 1, \ldots, N_\mathcal{C}$ representing $N_\mathcal{C}$ independent realizations of the random channel $\hat{\mathcal{C}}_{n,t}$. An interpretation of (6.6) leads to a simple solution to estimate the expected pixel distortion $\overline{d}_{b,m}$. More details on the implementation and the performance of this algorithm, referred to as Multiple–Decoder Distortion Estimation (MDDE), are discussed in the following.

Implementation of Mode Selection Algorithm based on MDDE

For the modified mode decision according to (6.5) an estimation of $\overline{d}_{1,b,m}(\hat{\mathcal{C}}_{t-1})$ rather than $\overline{d}_{b,m}(\hat{\mathcal{C}}_t)$ is necessary. In the encoder $N_\mathcal{C}$ independent copies $\mathcal{C}_{n,t}$ of the random channel as well as of the decoder are operated, each with individual reference frames $\hat{\mathbf{s}}_{n,t}$. The following implementation of the encoding process including the mode selection for packet lossy channels is proposed:

for each frame $t = 1, \ldots,$

- for each MB $b = 1, \ldots,$
  - for each coding mode $m \in \mathcal{O}$,
    - encode MB b with coding mode m,
    - for each channel–decoder pair $n = 1, \ldots, N_\mathcal{C}$,
      - Decode this MB based on possibly erroneous reference frames $\hat{\mathbf{s}}_{n,t-1}$, $\ldots$, $\hat{\mathbf{s}}_{n,t-N_{\text{ref}}}$ to obtain $\hat{\mathbf{s}}_{n,b,t}(m)$,
      - compute

$$\tilde{d}_{1,b,m}(\mathcal{C}_{n,t}) \triangleq ||\mathbf{s}_{b,t} - \hat{\mathbf{s}}_{n,b,t}(m)||^2 \,,$$

∗ compute the expected distortion for this mode m as

$$\overline{d}_{1,b,m}^{(N_\mathcal{C})} \triangleq \frac{1}{N_\mathcal{C}} \sum_{n=1}^{N_\mathcal{C}} \tilde{d}_{1,b,m}(\mathcal{C}_{n,t}),$$

– select the optimized mode ${m^*}_b$ equivalently to (6.5) as

$${m^*}_b = \arg\min_{m \in \mathcal{O}} \left(\overline{d}_{1,b,m}^{(N_\mathcal{C})} + \lambda_\mathcal{O} r_{b,m} \right), \tag{6.7}$$

– encode macroblock b in frame t with mode ${m^*}_b$,

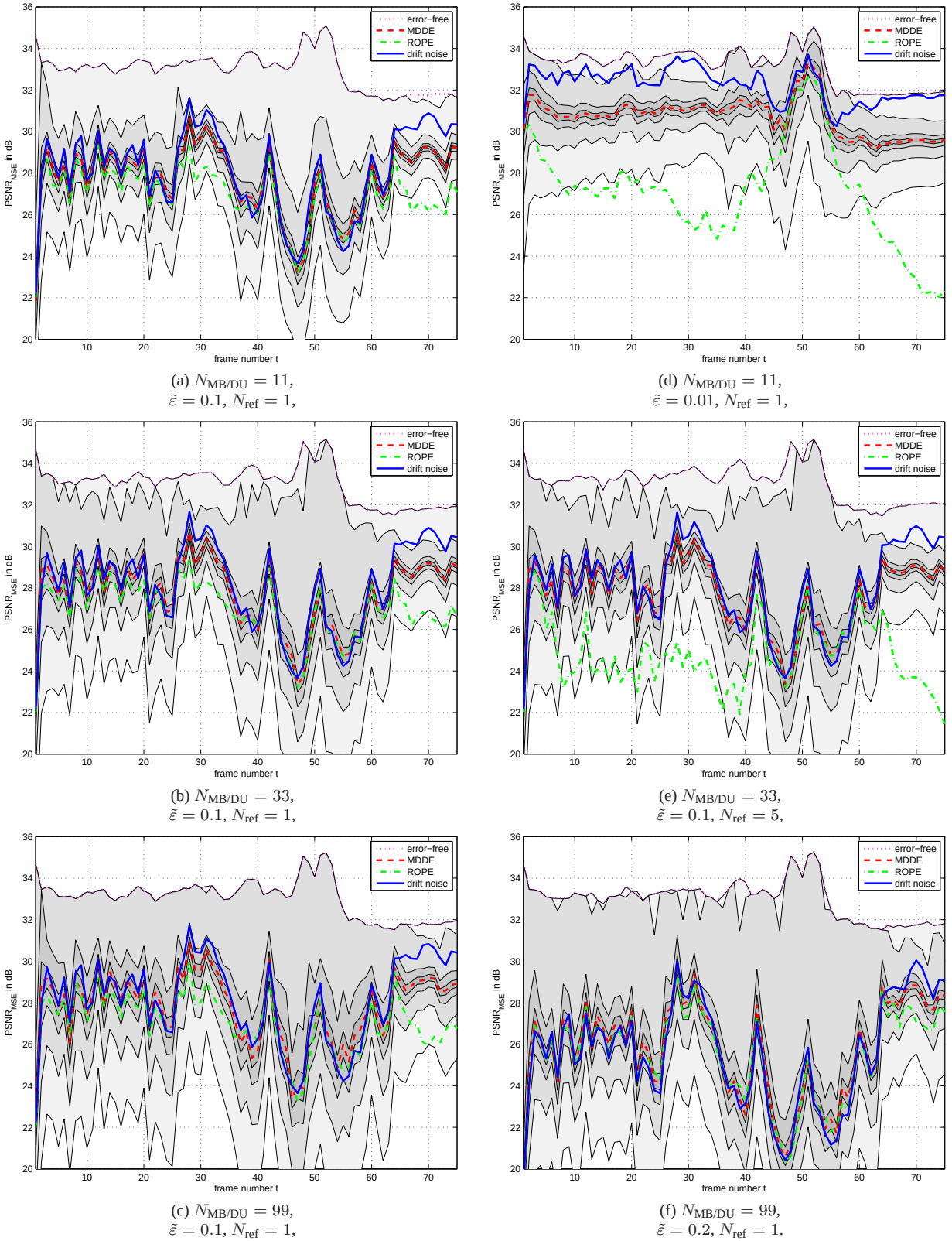
- generate the appropriate data units for this frame, $\mathcal{P}_{i_{t-1}+1}, \ldots, \mathcal{P}_{i_t}$,
- for each channel–decoder pair $n = 1, \ldots, N_\mathcal{C}$
  - draw a channel realization $\mathcal{C}_{n,i_{t-1}+1}, \ldots, \mathcal{C}_{n,i_t}$ for this specific pair n,
  - apply this channel realization to all data units $\mathcal{P}_{i_{t-1}+1}, \ldots, \mathcal{P}_{i_t}$,
  - decode this frame and apply error concealment based on reference frames $\hat{\mathbf{s}}_{n,t-1}$, …, $\hat{\mathbf{s}}_{n,t-N_{\text{ref}}}$ to generate new reference frame $\hat{\mathbf{s}}_{n,t}$.

Obviously, the most complex part of the encoding is the repeated decoding in the mode selection process as well as the storage for the reference frames of each individual channel-decoder pair. The additional complexity compared to mode selection without taking into account the channel statistics is approximately $N_\mathcal{C}$–times the decoder complexity in the encoder. Further complexity reductions are possible, but out-of-scope of this work.

Performance Results

In a first set of experiments we are interested in the quality of the expected decoder distortion $\overline{d}$ in the encoder when using different strategies for the computation within JM1.7 test model. To make ROPE feasible we only use the nearest full–pel motion vector and we ignore the loop filter operation for the recursive estimation of first and second order of the pixel expectation. For more details on the ROPE implementation for H.264/AVC we refer to [Xu02]. Furthermore, the approach proposed in [KKK02] measuring the expected distortion as the sum of quantization noise and drift noise is evaluated. We selected the presentation in Figure 6.4: Together with the error–free PSNR, the PSNR of expected distortion for each frame, $\text{PSNR}_{\overline{\text{mse}}}$, over frame number t for ROPE [ZRR00, Xu02], drift noise [KKK02], and MDDE with $N_\mathcal{C} = 500$, for test sequence *foreman* with QP $q = 20$ and PFC. The mode selection is applied according to the MDDE approach with $N_\mathcal{C} = 500$ which is an accurate estimation of the actual distortion. However, note that this strategy might hide effects caused by the propagation of the deviation to the actual distortion, but we neglect these effects in the following. Also shown for MDDE approach are the 90% reliability ranges of the $\text{PSNR}_{\overline{\text{mse}}}$ for $N_\mathcal{C} = 1, 10, 100, 1000$ from light grey to darker grey. The reliability is computed according to the bounds introduced in (2.19) which assumes a Gaussian process of the resulting distortions and therefore provides a worst case assumption. Different parameters are selected for the different sub-figures (a)-(f), namely the number of MBs per data unit $N_{\text{MB/DU}}$, the expected loss $\tilde{\varepsilon}$ assuming statistically independent packet losses, and the number of reference frames N_{ref}.

It is observed that for higher expected error rates $\tilde{\varepsilon} \geq 0.1$ and single reference frames the full–pel ROPE and the drift noise approach provide quite accurate estimates of the actual distortion.



(a) $N_{\text{MB/DU}} = 11$, $\tilde{\varepsilon} = 0.1$, $N_{\text{ref}} = 1$,

(b) $N_{\text{MB/DU}} = 33$, $\tilde{\varepsilon} = 0.1$, $N_{\text{ref}} = 1$,

(c) $N_{\text{MB/DU}} = 99$, $\tilde{\varepsilon} = 0.1$, $N_{\text{ref}} = 1$,

(d) $N_{\text{MB/DU}} = 11$, $\tilde{\varepsilon} = 0.01$, $N_{\text{ref}} = 1$,

(e) $N_{\text{MB/DU}} = 33$, $\tilde{\varepsilon} = 0.1$, $N_{\text{ref}} = 5$,

(f) $N_{\text{MB/DU}} = 99$, $\tilde{\varepsilon} = 0.2$, $N_{\text{ref}} = 1$.

Figure 6.4: Error–free PSNR and PSNR of expected distortion for each frame, $\text{PSNR}_{\overline{\text{mse}}}$, over frame number t for ROPE [ZRR00, Xu02], drift noise [KKK02], and MDDE with $N_{\mathcal{C}} = 500$, different parameters number of macroblock per data unit $N_{\text{MB/DU}}$, expected loss $\tilde{\varepsilon}$, and number of reference frames N_{ref} for test sequence *foreman* with QP $q = 20$ and PFC. Also shown for MDDE are the 90% reliability ranges of the PSNR of the average $\text{PSNR}_{\overline{\text{mse}}}$ for $N_{\mathcal{C}} = 1, 10, 100, 1000$ from light grey to darker grey.

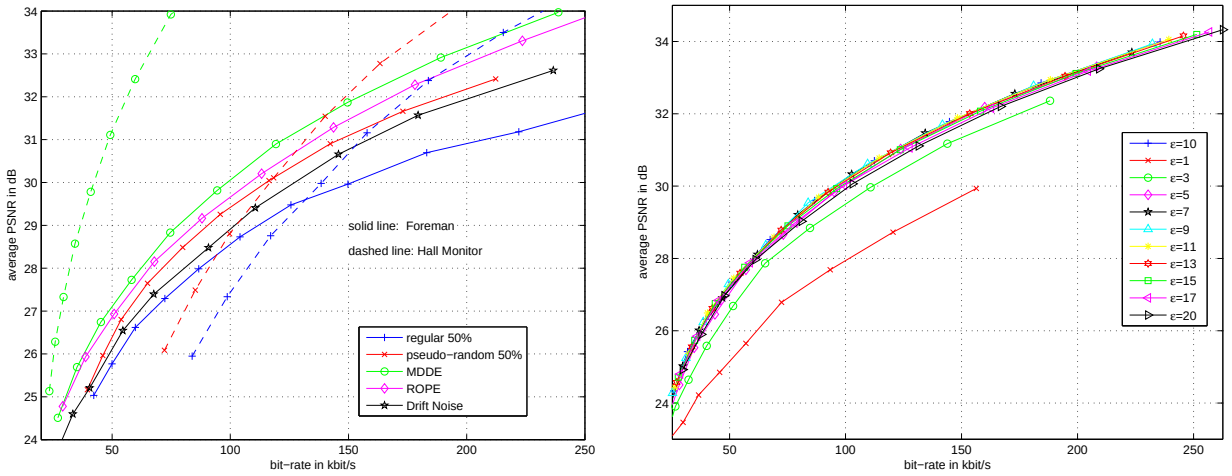
However, for some frame numbers the estimation is very inaccurate especially for the last frames where sub–pixel accurrate global motion dominates the sequence activity. Whereas ROPE tends to overestimate the distortion, the drift noise approach usually underestimates the real distortion. For low error rates, e.g. $\tilde{\varepsilon} = 0.01$ in Figure 6.4d), and for multiple reference frames, e.g. $N_{\text{ref}} = 5$ in Figure 6.4e), the estimation for ROPE and drift noise is very inaccurate. Therefore, to obtain a reasonably good quality of the expected distortion, consistent over a wide parameter range, we apply the MDDE approach in the remainder of this work.

To obtain an indication of the MDDE performance, we compute the variance of the expected distortion for each frame in the encoder for $N_{\mathcal{C}} = 500$ channel–decoder pairs in the encoder. Then, with the bounds $\text{PSNR}^{(N_{\mathcal{C}})}_{\overline{\text{mse}},l,\beta,t}$ and $\text{PSNR}^{(N_{\mathcal{C}})}_{\overline{\text{mse}},u,\beta,t}$ introduced in (2.19), we can provide a range in which at least 90% of the realizations for the actual estimated distortion falls, depending on the number of channel–decoder pairs, $N_{\mathcal{C}} = 500$. For each set of parameters in Figure 6.4 the 90% reliability range, i.e., $\beta = 0.9$, is shown for $N_{\mathcal{C}} = 1, 10, 100, 1000$ from light grey to darker grey. In any case, the upper bound, $\text{PSNR}^{(N_{\mathcal{C}})}_{\overline{\text{mse}},u,\beta,t}$, is limited to the error–free distortion. Obviously, the reliability range decreases with increasing number of channel–decoder pairs, but it adversely affects the complexity of the encoder. The results are more accurate if packets are smaller, e.g. $N_{\text{MB/DU}} = 11$, especially when compared to the case where the entire frame is transported in a single data unit. However, note that this does not mean that we advocate for smaller packet sizes for better rate–distortion performance. In terms of performance–complexity tradeoff, $N_{\mathcal{C}} = 100$ seems to be a good compromise and is chosen in the remainder of this work whenever we attempt to estimate the decoder distortion in the encoder.

Although these results provide an indication that MDDE with $N_{\mathcal{C}} = 100$ outperforms ROPE and drift noise, we are also interested in the rate–distortion performance when using different algorithms for intra updates. Figure 6.5 compares the optimized MDDE approach to other methods for different parameter settings for the same parameter settings as used for the results in Figure 6.3, i.e., QCIF test sequence *foreman* encoded at frame rate $f_s = 7.5$ fps with VBR rate control, transmitted over test error pattern 10, and with AEC in the decoder. For the estimation of the expected decoder distortion we assume statistically independent packet losses with $\tilde{\varepsilon} = 10\%$ as well as PFC in the decoder. Although this is not consistent with the actual decoder error concealment, an encoder can in general not rely that AEC is used in the decoder such that we apply PFC as worst–case assumption. For MDDE, we apply $N_{\mathcal{C}} = 100$ independent channel–decoder pairs.

Figure 6.5a) shows the average $\overline{\text{PSNR}}$ over the bit-rate for different MB mode selection strategies for $N_{\text{MB/DU}} = 33$. For the *foreman* sequence is observed that the channel and content–adaptive mode selection modes MDDE and ROPE outperform the best regular and pseudo–random intra update strategies, drift noise approach performs slightly worse than optimized pseudo–random updates. It is also observed that the quality of the estimated decoder distortion significantly influences the rate distortion performance. The gains of the MDDE compared the best regular intra updates are in the range of $1 - 2.5$ dB, depending on the bit-rate. The gains can also be expressed that for the same quality the bit-rate can be decreased by about 30%. Our ROPE implementation generally overestimates the decoder distortion as loop filter and fractional–pel MCP are ignored. Therefore, compared to MDDE for the same QP bit-rate and average PSNR are higher, but the tradeoff is not optimal. The drift noise estimation method generates similar bit-rates as MDDE for the same QP, but the average PSNR is lower. Therefore, we conjecture that the placement of intra macroblocks is sub–optimal. The results indicate that an adaptation of the ROPE algorithm to H.264/AVC is a promising way to obtain similar results as with the MDDE for future practical implementations, but for our research we stick with MDDE. It is also observed that although for the *foreman* sequence the random intra updates with 50% still perform reasonably well, for the sequence *Hall Monitor*

the performance is significantly degraded compared to the optimized solution: The required bit rate for pseudo–random updates is about three times as high as for the optimized case at the same quality.



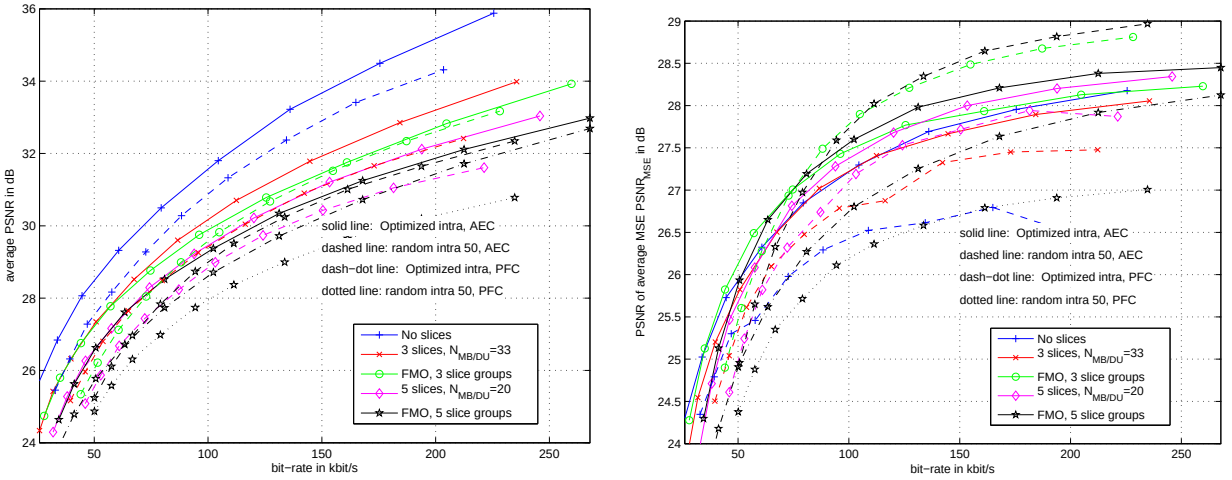
(a) intra update for $N_{\text{MB/DU}} = 33$

(b) intra update for $N_{\text{MB/DU}} = 33$

Figure 6.5: PSNR over bit-rate performance of optimized mode selection schemes for *foreman* and error pattern 10: (a) Average $\overline{\text{PSNR}}$ for different MB mode selection strategies, rate–distortion optimized with distortion estimation according to MDDE, ROPE, and drift noise, as well as regular and pseudo–random updated with ratio 50% for $N_{\text{MB/DU}} = 33$; (b) Average PSNR, $\overline{\text{PSNR}}$, for MDDE and different expected loss rates, $\tilde{\varepsilon}$, and $N_{\text{MB/DU}} = 33$.

For the results in Figure 6.5a) we assumed that the channel statistics or at least the average packet loss rate ε is known at the encoder and is used in the estimation process, i.e., $\tilde{\varepsilon} = \varepsilon$. However, in general one can not guarantee that the loss rate is known in the encoder exactly as feedback messages might be delayed, statistics might not be sufficient, etc. The influence of the accurateness of this expected loss rate is investigated, the results are presented in Figure 6.5b) showing the average $\overline{\text{PSNR}}$ for MDDE and different expected loss rates, $\tilde{\varepsilon}$, and $N_{\text{MB/DU}} = 33$. Estimation errors in the loss probability result in negligible performance loss as long as they are in the range of half or double the real loss rates. Optimality is achieved with expected loss rates slightly lower than the actual loss rate. This is due to correlation in the loss process, application of AEC in the real decoder, as well as greedy approach in the mode selection process. However, the exact loss rate in the encoder provides a sufficiently good performance such that we use $\tilde{\varepsilon} = \varepsilon$ in the remainder.

Finally, we investigate the combination of optimized macroblock mode selection and different packetization schemes in Figure 6.5c) and d) with the same parameters as for pseudo–random intra updates with ratio 50% in Figure 6.3c) and d), respectively. For comparison, the corresponding results for pseudo–random intra updates are also shown in Figure 6.5a) and b). Figure 6.5a) shows the average $\overline{\text{PSNR}}$ over the bit-rate. It is observed that for any packetization schemes, the MDDE approach outperforms the pseudo–random intra updates. The results based on the average PSNR propose to omit any advanced packetization schemes, the reasons are identical to those discussed for pseudo–random intra updates. In addition, it is observed that for the same number of packets per frame, slice structured coding outperforms FMO. A closer investigation reveals that the average PSNR for both packetization schemes are almost identical, but the bit-rate for FMO is higher. This comes from the fact that in case of FMO less prediction within one frame is possible compared to slice structured coding as less MBs in the neighborhood can be used for prediction.



(a) average $\overline{\text{PSNR}}$, (b) PSNR of average.

Figure 6.6: PSNR over bit-rate performance of optimized mode selection schemes for *foreman* and error pattern 10: (a) average $\overline{\text{PSNR}}$ for different packetization schemes and optimized intra updates with MDDE; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$, for different packetization schemes and optimized intra updates with MDDE .

Similarly to Figure 6.3d), we also show the PSNR of the average MSE, $\text{PSNR}_{\overline{\text{mse}}}$, for optimized intra compared to pseudo–random intra updates in Figure 6.5b). The results are surprising, but explained in the following: First of all it is observed that the performance for different packetization schemes does not differ significantly if optimized mode selection is applied. Depending on the bit-rate, either no slice structuring or FMO with a higher number of slice groups – 5 in our case – performs best. However, it also observed that for bit-rates higher than about 80 kbit/s, FMO with pseudo–random intra updates is superior than optimized intra updates. A closer look reveals that for the same quantization parameter, both, the bit-rate and the PSNR, $\text{PSNR}_{\overline{\text{mse}}}$, are in favour of the pseudo–random intra updates which is explained as follows: The higher bit-rates indicate that for optimized intra more intra information is inserted than for pseudo–random with 50%. In general, more intra updates are favorable in terms of PSNR as they remove error propagation, which initially contradicts the result of Figure 6.5d). However, for high bit-rates the contribution of the quantization noise to the overall MSE vanishes, the performance is exclusively determined by instantaneous losses and error propagation. The latter contribution is also of less relevance as for both cases error propagation is usually stopped immediately due to the high amount of intra information. Therefore, the contribution of instantaneous losses, mainly determined by the error concealment algorithm, has major influence on the performance. For both intra update modes, the same error concealment algorithm, namely AEC, is applied. However, whereas in case of pseudo–random intra updates the AEC still can perform temporal error concealment as at least some of the neighbors are coded in inter mode, in case of optimized intra almost exclusively spatial error concealment has to be used which results in significantly higher distortion though subjectively still acceptable. This effect is also confirmed in Figure 6.5d) as for PFC, optimized intra outperforms random intra updates significantly. This results also show the potential and the necessity of good error concealment in combination with FMO.

In summary, from the results it is obvious that error resilience tools – if applied appropriately – can significantly increase the performance of packet–lossy video transmission. Improvements are achieved by the introduction of packetized video, in particular FMO, as well as by the use of AEC, especially if the measure of interest is the average MSE. However, most important is the limitation

of error propagation by the use of intra updates which is most powerful if combined with optimized mode selection schemes.

6.1.5 Interactive Error Control

Background

Removing and limiting drift and error propagation is the key for operating video in packet-lossy environments. In the following we will introduce methods which completely eliminate error propagation by the use of feedback channels.

The availability of the fast feedback channel in RTP connections especially for conversational applications has led to different standardization activities, e.g. [OWS+04], and research activities in recent years. Assume that in contrast to the previous scenario where only the statistics of the channel process, $\hat{\mathcal{C}}_t$, are known to the encoder, that in case of timely feedback we can even assume that a δ–frame delayed version $\mathcal{C}_{t-\delta}$ of the loss process experienced at the receiver is known at the encoder. This characteristic can be conveyed from the decoder to the encoder by acknowledging correctly received data units, sending not–acknowledge messages for missing slices or both types of messages.

In less time–critical applications such as streaming or download, one could obviously decide to retransmit lost data units if the transmitter has stored the data units using TCP or as for example proposed in [RLM+04] in RTP environments. In low–delay applications the retransmitted data units, especially in end–to–end connections, would however arrive too late to be useful at the decoder as discussed in chapter 2. For the case of online encoding this is different: The observed and possibly delayed receiver channel realization, $\mathcal{C}_{t-\delta}$, can still be useful to the encoder though the erroneous frame has already been decoded and concealed at the decoder. The basic goal of these approaches is to reduce, limit, or even completely avoid error propagation by integrating the information in the encoding process.

The exploitation of the observed channel at the encoder has been introduced in [SFG97] and [GF99] under the acronym *Error Tracking* for standards such as MPEG-2, H.261 or H.263 version 1, but has been limited by the reduced syntax capabilities of these video standards. When receiving the information that as certain data unit has not been received correctly at the decoder (typically including the coded representation of several or all macroblocks of a certain frame $\mathbf{s}_{t-\delta}$), the encoder attempts to track the error to obtain an estimate of the decoded frame $\hat{\mathbf{s}}_{t-1}$ serving as reference for the frame to be encoded, $\mathbf{s}_t$. Appropriate actions after having tracked the error are for example presented in [WFSG00, SFG97, GF99, Wad89, WZ98]. However, all these concepts is in common that error propagation in frame $\hat{\mathbf{s}}_t$ is only removed if frames $\hat{\mathbf{s}}_{t-\delta+1}, \ldots, \hat{\mathbf{s}}_{t-1}$ have been received at the decoder without any error.

This promising performance when exploiting decoder state information at the encoder has been recognized by standardization bodies, and the problem of continuing error propagation has been addressed by extending the syntax of existing standards. In MPEG-4 [MPG, version 2] a tool to stop temporal error propagation has been introduced under the acronym New Prediction (NEWPRED) [NT96, FNI96, TKI97]. Similarly, in H.263+ Annex N [H26b, Annex N] Reference Picture Selection (RPS) for each Group-of-Blocks (GOB) is specified. If combined with slice structured mode as specified in H.263+ Annex K [H26b, Annex K] as well as Independent Segment Decoding (ISD) as specified in H.263+ Annex R [H26b, Annex R] the same NEWPRED techniques can be applied within the H.263 codec family.

NEWPRED obviously relies on rather quick feedback messages and online encoding, but also on the possibility that the encoder can choose other reference frames than the timely preceding.

NEWPRED allows to completely eliminate error propagation in frame $\hat{\mathbf{s}}_t$ even if additional errors have occurred for the transmission of frames $\hat{\mathbf{s}}_{t-\delta+1}, \ldots, \hat{\mathbf{s}}_{t-1}$. Different encoder operation modes have been discussed in the literature [FNI96]. Basically two modes can be distinguished: In one mode only acknowledged areas are used as reference and in another mode the operation is only altered when the encoder receives information that the decoder misses some data units.

Framework within H.264/AVC

Whereas the initial NEWPRED approaches have been introduced exclusively for error resilience, in H.263++ Annex U [H26b, Annex U] and especially in H.264/AVC the extended syntax allowing to select MB modes and reference frames on MB or even sub–MB basis permits incorporating methods for reduced or limited error propagation in a straight–forward manner [WFSG00]. We introduce different operation modes when incorporating decoder state information in the encoding process.

Assume that at the encoder each generated data unit $\mathcal{P}_i$ is assigned a decoder state $\mathcal{C}_{\text{enc},i} \in \{\text{ACK}, \text{NAK}, \text{OAK}\}$ whereby $\mathcal{C}_{\text{enc},n} = \text{ACK}$ refers to the case that data unit $\mathcal{P}_i$ is known to be correctly received at the decoder, $\mathcal{C}_{\text{enc},i} = \text{NAK}$ reflects that data unit $\mathcal{P}_i$ is known to be missing at the decoder, and $\mathcal{C}_{\text{enc},i} = \text{OAK}$ reflects that for data unit $\mathcal{P}_i$ the acknowledgement message is still outstanding and it is not known whether this data unit will be received correctly.

With feedback messages conveying the observed channel state at the receiver, i.e., $\mathcal{B}(\mathcal{C}_t) = \mathcal{C}_t$, and a back channel which delays the back channel messages by δ frames, i.e., we assume in the remainder that for the encoding of frame s_t, the encoder is aware of the following information:

$$\mathcal{C}_{\text{enc},i} = \begin{cases} \text{ACK} & \text{if } \tau_{\text{PTS},i} \leq \tau_{s,t-\delta} \text{ and } \mathcal{C}_i = 1, \\ \text{NAK} & \text{if } \tau_{\text{PTS},i} \leq \tau_{s,t-\delta} \text{ and } \mathcal{C}_i = 0, \\ \text{OAK} & \text{otherwise.} \end{cases} \tag{6.8}$$

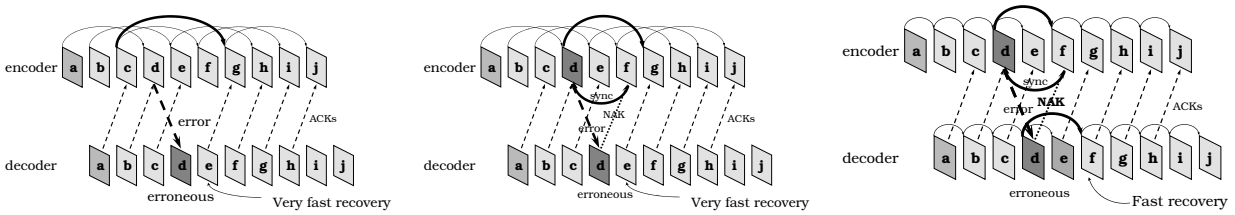
This is equivalent that received data units are acknowledged with delay δ, missing data units are reported with delay δ, and for data units sent just recently no decoder status information is available yet.

The information about the decoder state $\mathcal{C}_{\text{enc},i}$ can be integrated in a modified rate–distortion optimized operational encoder control similar as what has been discussed in subsection 6.1.4. In this case the MB mode ${m^*}_b$ is selected from a modified set of options, $\hat{\mathcal{O}}$, with a modified distortion $\hat{d}_{b,m}$ for each selected option m as

$$\forall_b \quad {m^*}_b = \arg\min_{m \in \hat{\mathcal{O}}} \left(\hat{d}_{b,m} + \lambda_{\mathcal{O}} r_{b,m} \right). \tag{6.9}$$

We distinguished four different operation modes which differ only by the set of coding options available to the encoder in the encoding process, $\hat{\mathcal{O}}$, and the applied distortion metric, $\hat{d}_{b,m}$. The encoder's reaction to delayed ACK and NAK messages for three different feedback modes is shown in Figure 6.7 assuming that frame (d) is lost and the feedback delay is $\delta = 2$.

Feedback Mode 1: Acknowledged Reference Area Only Figure 6.7a) shows this operation mode: Only the decoded representation of data units $\mathcal{P}_i$ which have been positively acknowledged at the encoder, i.e., $\mathcal{C}_{\text{enc},i} = \text{ACK}$, are allowed to be referenced in the encoding process. In the context of operational encoder control this is formalized by applying the encoding distortion in (6.9), i.e., $\hat{d}_{b,m} = d_{b,m}$, as well as a set of encoding options which is restricted to acknowledged areas only, i.e., $\hat{\mathcal{O}} = \mathcal{O}_{\text{ACK},t}$. Note that the restricted option set $\mathcal{O}_{\text{ACK},t}$ depends on the frame to be



(a) Feedback Mode 1
Acknowledged Reference Area Only

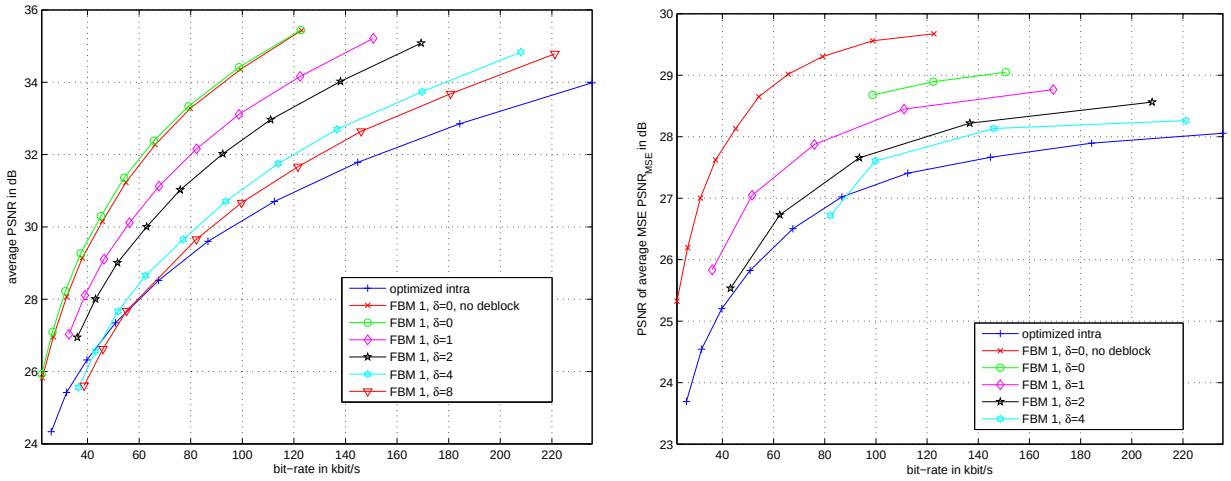
(b) Feedback Mode 2
Synchronized Reference Frames

(c) Feedback Mode 3
Limited Error Propagation

Figure 6.7: Operation of different feedback modes.

encoded and is applied to both, the motion estimation as well as in the reference frame selection. Obviously, if no reference area is available, the option set is restricted to intra modes only, or if no satisfying match is found in the accessible reference area, intra coding is applied.

This mode does not prevent the appearance of an error within a single frame but error propagation and reference frame mismatch between encoder is completely avoided. This is even independent of the decoder's error concealment as long as correctly decoded pixels are not altered by the error concealment algorithm at the receiver. However, exactly this problem is present in the test model JM1.7 used in this work as the deblocking filter operation in the motion compensation loop is applied over slice boundaries. To guarantee complete removal of encoder and decoder mismatch one has either to restrict the reference area significantly, or the deblocking filter must have the capability to be disabled at slice boundaries. Due to this problem, the adaptive deblocking filter mode had been introduced in the final standard of H.264/AVC.



(a) Average $\overline{\text{PSNR}}$

(b) PSNR of average MSE.

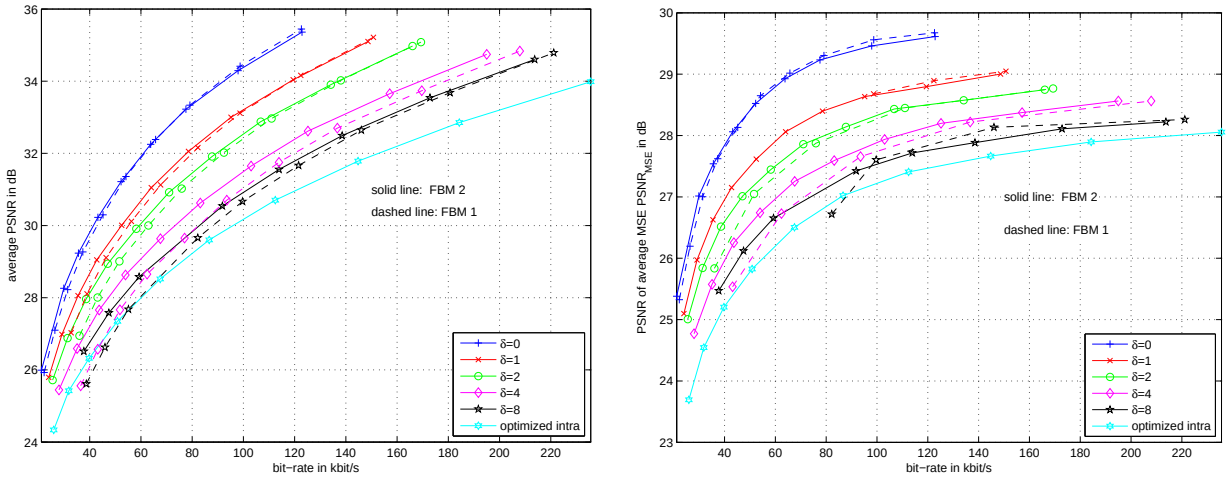
Figure 6.8: PSNR over bit-rate performance for feedback mode 1 with different feedback delays compared to optimized mode selection scheme for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$: (a) Average PSNR, $\overline{\text{PSNR}}$,; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$.

Figure 6.8 shows the performance in terms of average $\overline{\text{PSNR}}$ in (a) and PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$ in (b) for feedback mode 1 with different feedback delays δ compared to optimized mode selection scheme for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$. The number of reference frames is $N_{\text{ref}} = 5$, except for $\delta = 8$ with $N_{\text{ref}} = 10$. If not stated otherwise, the in–loop deblocking filter is used and the mismatch is accepted. The results show that for any delay this system with feedback outperforms the best system without any feedback. For small feedback delays, the gains are significant and for the same average PSNR the bit-rate is less than 50% compared to the

forward only mode. With increasing delay the gains are reduced, but compared with the highly complex mode decision without feedback this method is still very attractive. Obviously, the high-delay results are strongly sequence dependent but for other sequences similar results have been verified. The performance advantages are also verified when using the average MSE as the measure of interest. The figure also shows that the influence of loop filter mismatch ("no deblock") is less significant than the loss of coding efficiency when turning off the loop filter for the entire sequence. As a consequence, disabling the the loop filter in the final standard of H.264/AVC should be considered very carefully when operated in such environments.

Feedback Mode 2: Synchronized Reference Frames The loop–filter problem is avoided if the encoder synchronizes its reference frames to the reference frames of the decoder. This is accomplished by using exactly the same decoding process for the generation of the reference frames in the encoder and the decoder. The reference frames are synchronized as shown in Figure 6.7b). The important difference is that not only positively acknowledged data units are allowed to be referenced, but also a concealed version of data units with decoder state $\mathcal{C}_{\text{enc},i} = \text{NAK}$. The reference frame buffer is updated by a decoder with delayed channel observation $\mathcal{C}_{t-\delta}$. This is formalized by again applying the encoding distortion in (6.9), i.e., $\hat{d}_{b,m} = d_{b,m}$ but the restricted reference area and the option set in this case also include concealed image parts, $\hat{\mathcal{O}} = \mathcal{O}_{\text{NAK},t}$.

The critical operation in this mode is the error concealment as it is a non–normative feature in H.264/AVC. Therefore, only if the encoder is exactly aware of the decoder's error concealment, this mode can provide benefits compared to feedback mode 1, otherwise is is not expected that encoder and decoder reference frame mismatches can be avoided.



(a) Average $\overline{\text{PSNR}}$

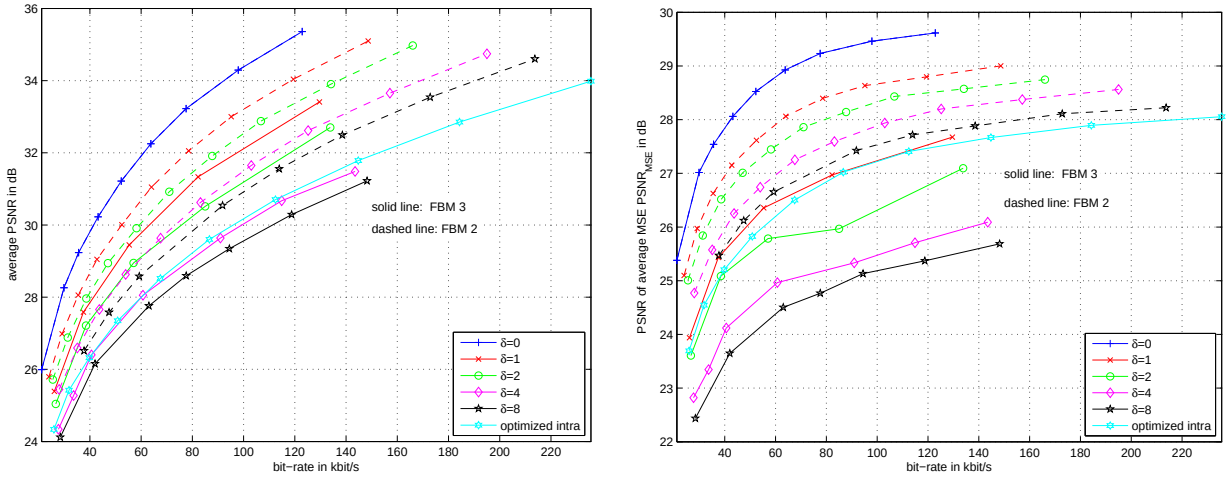
(b) PSNR of average MSE.

Figure 6.9: PSNR over bit-rate performance for feedback mode 2 with different feedback delays compared to optimized mode selection scheme and feedback mode 1 for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$: (a) Average PSNR, $\overline{\text{PSNR}}$,; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$.

Figure 6.9 shows the performance in terms of average $\overline{\text{PSNR}}$ in (a) and PSNR of average MSE in (b) for feedback mode 2 with different feedback delays δ compared to optimized mode selection scheme and feedback mode 1 for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$. Again, the number of reference frames is $N_{\text{ref}} = 5$, except for $\delta = 8$ with $N_{\text{ref}} = 10$. The results for feedback mode 2 show similar results as for feedback mode 1, the performance for increasing delay decreases in a similar way as it does for the feedback mode 1 only. For low bit-rates and

delays $\delta < N_{\text{ref}} - 1$, the gains are only visible when using feedback mode 2 instead feedback mode 1 for lower bit-rates as referencing concealed areas is preferred over intra coding by the rate–distortion optimization. For higher bit-rates this advantage vanishes as the intra mode is preferred anyways over the selection of ”bad” reference areas. For delay $\delta = 4$ with $N_{\text{ref}} = 5$, i.e., only a single reference frame is available at the encoder, the gains of feedback mode 2 are more obvious as in case of feedback mode 1 in case of a lost slice the encoder basically is forced to use intra coding. However, in summary well-designed systems with $\delta < N_{\text{ref}} - 1$, the gains combined with the disadvantage of necessary normative error concealment for feedback mode 2 makes the use of feedback mode 1 the preferred one in practicable systems. Still due to implementation advantages we prefer to use feedback mode 2 in the remainder of this work. The differences to feedback mode 1 are negligible.

Feedback Mode 3: Regular Prediction with Limited Error Propagation Feedback mode 1 and 2 are mainly suitable in case of higher error rates. However, if the error rates are low or even negligible, the performance is significantly restricted by the longer prediction chains due to the feedback delay. Therefore, in feedback mode 3 as shown in Figure 6.7c) it is proposed to only alter the prediction in the encoder in case of the reception to a NAK. Again, the encoding distortion in (6.9) is applied, i.e., $\hat{d}_{b,m} = d_{b,m}$ but the reference area and the option set in this case are altered only in case of the reaction of a NAK to already acknowledged image parts, i.e., $\hat{\mathcal{O}} = \mathcal{O}'_{\text{NAK},t}$, or, as applied in our case to acknowledged and concealed image parts, $\hat{\mathcal{O}} = \mathcal{O}'_{\text{NAK},t}$. Reference areas which are possibly corrupted by error propagation are also excluded for future reference frames. This mode obviously performs well in case of lower error rates. However, for higher error rates error propagation still occurs quite frequently.



(a) Average $\overline{\text{PSNR}}$

(b) PSNR of average MSE.

Figure 6.10: PSNR over bit-rate performance for feedback mode 2 with different feedback delays compared to optimized mode selection scheme and feedback mode 1 for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$: (a) Average PSNR, $\overline{\text{PSNR}}$,; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$.

Figure 6.10 shows the performance in terms of average PSNR $\overline{\text{PSNR}}$ in (a) and PSNR of average MSE in (b) for feedback mode 3 with different feedback delays δ compared to optimized mode selection scheme and feedback mode 2 for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 33$. Again, the number of reference frames is $N_{\text{ref}} = 5$, except for $\delta = 8$ with $N_{\text{ref}} = 10$. Note that feedback mode 2 and feedback mode 3 are identical for zero feedback delay. However, surprisingly

for increasing delay feedback mode 3 performs significantly worse than feedback mode 2 for both quality measures, but especially for the average MSE. The error propagation degrades the overall quality much more significantly and can not compensate the gains for compression efficiency. Obviously, the performance depends on the sequence characteristics and especially on the error rates. For lower error rate it is will shown later, that the differences between feedback mode 2 and 3 are less significant but in general feedback mode 2 is preferable over feedback mode 3 also from the subjective performance point-of-view.

Feedback Mode 4: Unrestricted Reference Areas with Expected Distortion Update For completeness we present an even more powerful feedback mode, which extends feedback mode 3 to address the error propagation with increased amount of intra updates. We also discuss its drawbacks and justify why it is not used in the remainder of this work. In [WFSG00] and [ZRR00] techniques have been proposed which combine the error–resilient mode selection with available decoder state information in the encoder. In this case the set of encoding options is not altered, i.e., $\hat{\mathcal{O}} = \mathcal{O}$, but only the computation of the distortion is altered. Only for all data units with outstanding acknowledgement at the encoder, i.e., $\mathcal{C}_{\text{enc},i} = \text{OAK}$, the randomness of the observed channel state is considered, for all other data units the observed channel state is no more random. The expected distortion in this case computes as

$$\hat{d}_{b,m} = \mathbb{E}_{\{\forall_{\left\{i\middle|\mathcal{C}_{\text{enc},i}=\text{OAK}\right\}}\hat{\mathcal{C}}_i\}}\left\{\tilde{d}_{b,m}\left(\forall_{\{i|\mathcal{C}_{\text{enc},i}\neq\text{OAK}\}}\mathcal{C}_i, \forall_{\{i|\mathcal{C}_{\text{enc},i}=\text{OAK}\}}\hat{\mathcal{C}}_i\right)\right\}. \tag{6.10}$$

Compared to feedback mode 1 and 2, this method is especially beneficial, if the feedback is significantly delayed. Compared to feedback mode 3, it reduces the unsatisfying performance in case of error propagation. Note that for $\delta \to \infty$ this mode turns into the mode selection without feedback at all, and for $\delta = 0$ this mode is identical to feedback mode 2 and feedback mode 3. However, whenever the encoder gets information on the state of a certain data unit at the decoder, in case of MDDE method all reference frames in the encoder's decoders, and in case of ROPE the first and second order moments have to be re–computed. Thus, the computational, storage, and implementation complexity is significantly increased [ZRR00] which is the reason why we omit this method in the remainder and almost exclusively concentrate on feedback mode 2.

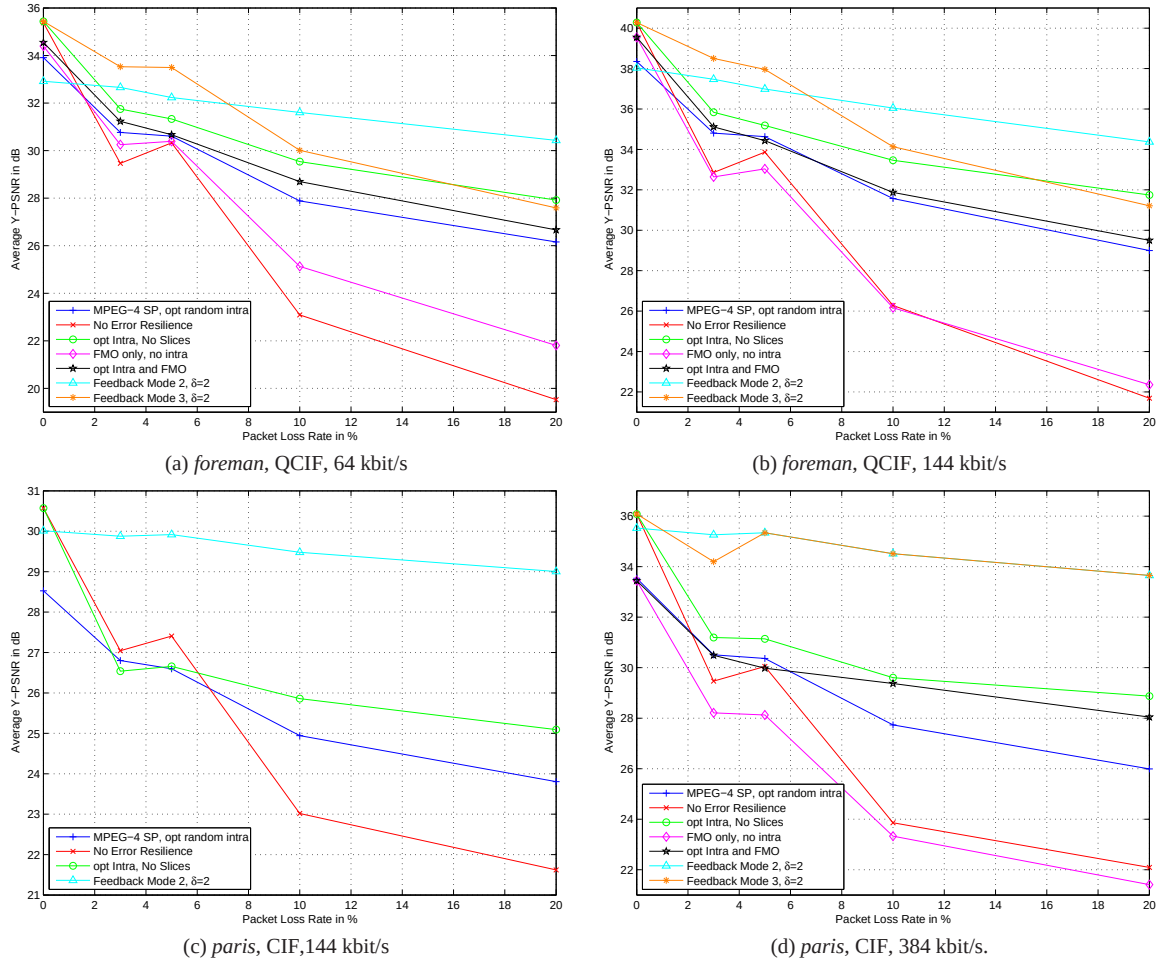
6.1.6 Performance Results for Test Conditions

To verify the findings of the previous results also for other error rates, bit-rates, and test sequences we have applied selected error resilience tools to the test cases as suggested in [Wen01]. We extend the previous performance evaluation to zero loss probability as well as four error patterns 3, 5, 10, and 20 as introduced in subsection 2.4.5 as well as to the CIF test sequence *paris* encoded at frame rate $f_s = 15$ fps.

We exclusively use the average $\overline{\text{PSNR}}$ to report the results as suggested in [Wen01]. VBR rate control according to (3.22) is applied, i.e., we select only a single parameter for the entire sequence, the bit-rate is not exactly matched. We apply weighted time–sharing between the quantizer for which the resulting bit-rate $\overline{R}$ just exceeds the target bit-rate R, $\overline{q}$, and the quantizer, $\underline{q}$, for which the resulting bit-rate, $\underline{R}$, falls just below the target bit-rate, R. The average PSNR of the two experiments are denoted as $\overline{\overline{\text{PSNR}}}$ and $\overline{\underline{\text{PSNR}}}$, respectively. The time-sharing idea justifies that for the performance measure the weighted average PSNR,

$$\overline{\text{PSNR}} = \frac{\overline{\overline{\text{PSNR}}} - \overline{\underline{\text{PSNR}}}}{\overline{R} - \underline{R}}\left(\overline{R} - R\right) + \overline{\underline{\text{PSNR}}},$$

is applied.



(a) *foreman*, QCIF, 64 kbit/s

(b) *foreman*, QCIF, 144 kbit/s

(c) *paris*, CIF,144 kbit/s

(d) *paris*, CIF, 384 kbit/s.

Figure 6.11: Average $\overline{\mathrm{PSNR}}$ over packet error rate for *foreman*, QCIF, with frame rate $f_s = 7.5$ fps and *paris*, CIF, with frame rate $f_s = 15$ fps both for different bit-rates and different error resilience tools in H.264/AVC compared to MPEG–4 with optimized random intra updates. H.264/AVC uses no error resilience and channel optimized intra updates without slice structuring as well as with FMO checkerboard pattern with 2 packets per frame, and feedback mode 2 with $\delta = 2$.

Figure 6.11 shows the average $\overline{\mathrm{PSNR}}$ over packet error rate[7] for *foreman* and *paris* for different bit-rates and different error resilience tools in H.264/AVC compared to MPEG–4 simple profile with optimized ratio of random intra updates. The results are consistent for both sequences and all bit-rates, and also verify the results previously presented for *foreman* and error pattern 10 only. The significance of the difference between different schemes is mainly explained by different sequence characteristics. It is observed that for error–free transmission omitting any error resilience tools obviously results in best performance. The performance gains in terms of compression efficiency of H.264/AVC over MPEG-4 simple profile is also visible. If feedback mode 2, in our case with feedback delay $\delta = 2$, we have to sacrifice some compression efficiency as the prediction signal is in general worse as it is further in the past. This does not apply for feedback mode 3.

[7]The labels on the abscissa specify the corresponding error pattern rather than the exact packet loss rates. Note that the 5% error file is burstier than the others resulting in somewhat unexpected results.

However, the performance in error–free transmission environment is less relevant for our investigations. With increasing error rates it is obvious that any kind of error resilience feature in general improves the performance. Thereby, it is again recognized that means reducing the error propagation are much more important than packetization modes such as FMO, in our case with checkerboard pattern and two packets per frame. Again, with the average PSNR as the measure of interest, the best performance without any feedback is obtained using channel–adapted rate–distortion optimized mode selection according to (6.4) with each packet containing an entire source frame. Additional significant performance improvements can be achieved by the introduction of decoder feedback information. Thereby, for lower error rates feedback mode 3 outperforms feedback mode 2, but feedback mode 2 provides very consistent results over a large range of error rates.

From the results as well as subjective observations it can be concluded that avoiding error propagation is the most important issue in error–prone video transmission. If no feedback is available, an increased percentage of intra macroblocks, selected by channel–adapted optimization schemes perform best. Whenever feedback is available, it is suggested that only those areas are used for prediction for which the encoder is sure that the decoder has exactly the same reference area.

6.2 End-to-End Forward Error Correction for Low-Delay Applications

The use of FEC schemes for packet erasure channels has already been introduced in section 4.5. In the following we will discuss the application of end–to–end forward correction schemes for low–delay conversational applications over packet lossy channels. The use of generic end–to–end FEC based on RFC2733 [RS99] in the considered video transmission scheme is shown Figure 6.12. The channel code is applied over n RTP packets which are reconstructed at the receiver if channel decoding is successful.

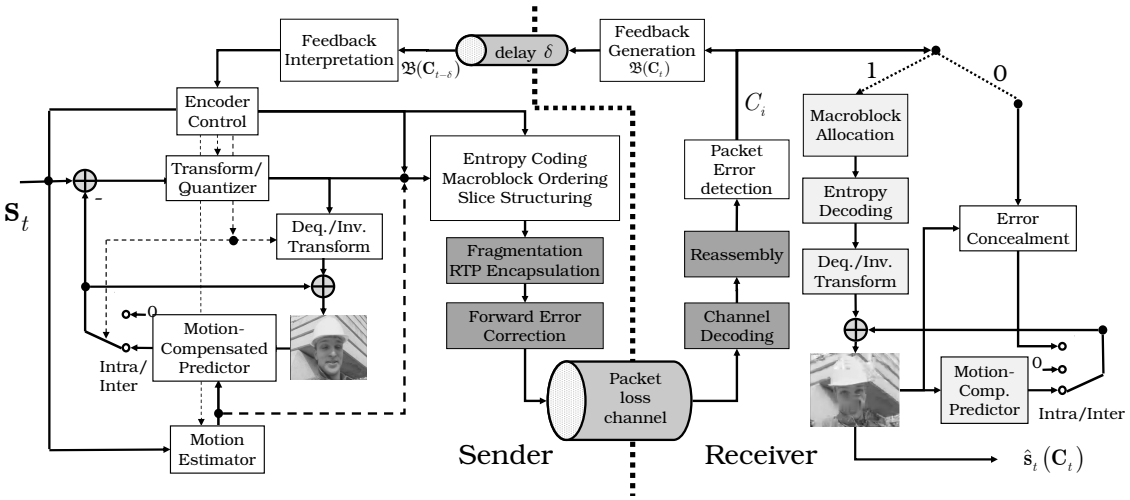


Figure 6.12: Video transmission system in packet loss environment.

The encoder generally has to await k RTP packets before the $n-k$ parity packets can be generated. At the receiver, all n RTP packets have to be received before decoding is started. However, as shown in section 4.5, the performance of the code strongly depends on the block length of the

code, n: The block length should be as large as possible resulting in a delay of in general n RTP packets.

Three different modes to generate and combine $k = 3$ information packets are shown in Figure 6.13. Figure 6.13a) proposes to apply the channel code over several source units whereby each source unit is contained in a single data unit. However, the drawback of this approach is obvious as possibly one has to wait for the parity packets resulting in intolerable delay for conversational applications. Therefore, other means to generate more than one data unit from a single source unit are required. Figure 6.13b) proposes to introduce slice structured coding to obtain several data units for each source unit. The drawback of this mode is also obvious as the coding efficiency with the introduction of slices decreases significantly. In addition, in both modes the data units generally do not have the same length resulting in additional overhead when applying RFC2733.

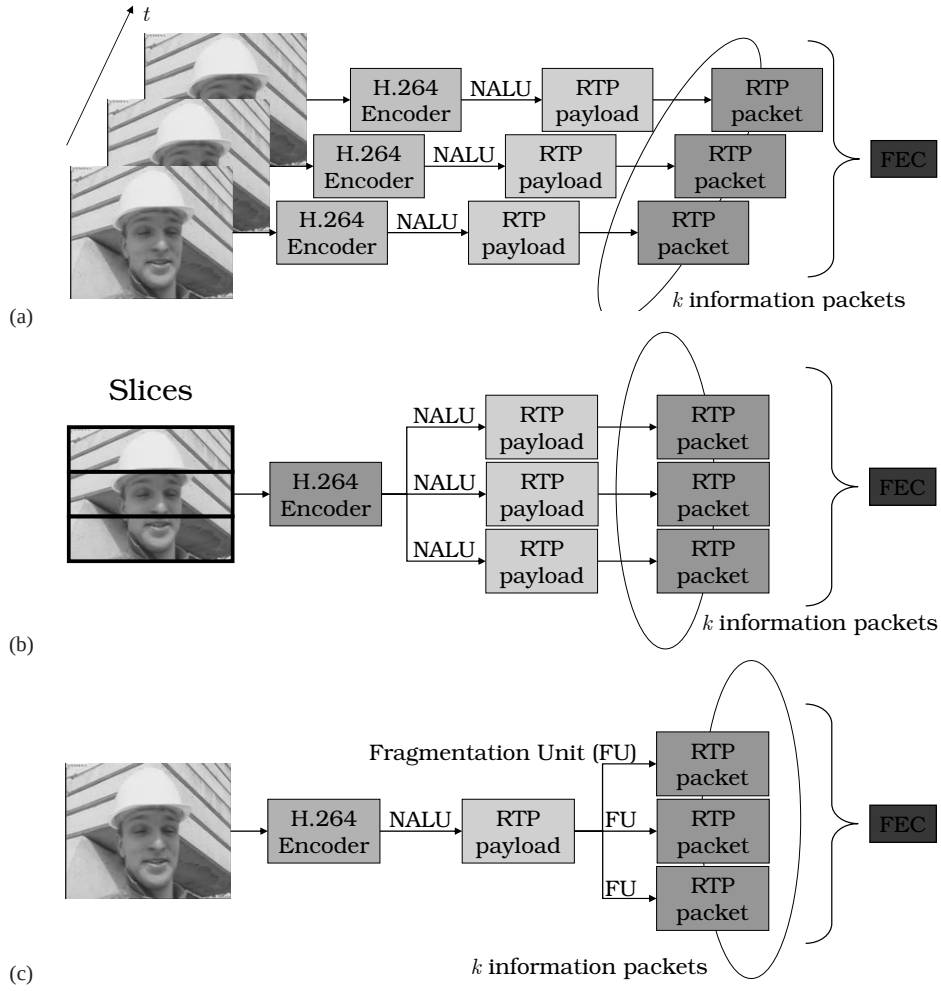


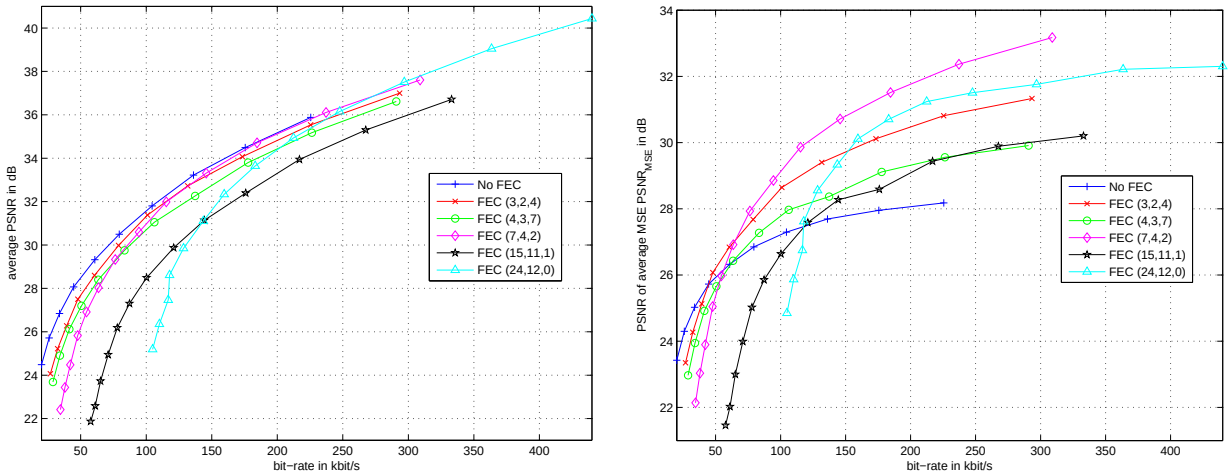
Figure 6.13: Modes to generate and combine $k = 3$ information packets for generic FEC: (a) multiple frames, (b) multiple slices, (c) fragmentation units.

Therefore, we have introduced an advanced packetization mode in the RTP payload specification for H.264 [WHS+05] which allows fragmentation of NAL units to several RTP packets. The principle is shown in Figure 6.13c): Each source unit can be divided into k fragmentation units of basically equal length with at most one byte difference. Then, generic FEC as introduced in section 4.5 is applied based on RFC2733. The compression efficiency is not restricted as we abandon any slice structured coding and solely can rely on the FEC. Obviously, any other error

resilience features such as rate–distortion optimized mode selection or the exploitation of feedback can be combined with FEC. Selected performance results are presented in the following.

Figure 6.14 shows the PSNR over bit-rate performance for different codes and optimized mode selection scheme indicated as $(n, k, \tilde{\varepsilon})$ for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 99$. All the overhead for fragmentation units, IP overhead, etc. is taken into account. It is observed that for the average $\overline{\text{PSNR}}$ the system without FEC performs best except for very high bit-rates. The performance of the simple Hamming code with $n = 7$ is rather good. For larger block sizes n, the performance of the code is mainly limited by the header overhead, especially when applying the Golay code with $n = 24$ is used.

However, with the average MSE as the measure of interest, the performance of FEC is significantly superior than systems not using FEC. Due to the reduced error rates, the percentage of lost frames decreases significantly. The best performance is achieved by the Hamming code with $n = 7$. A similar performance is achieved by the Golay code, but with the advantage that one can omit any other error resilience tools.



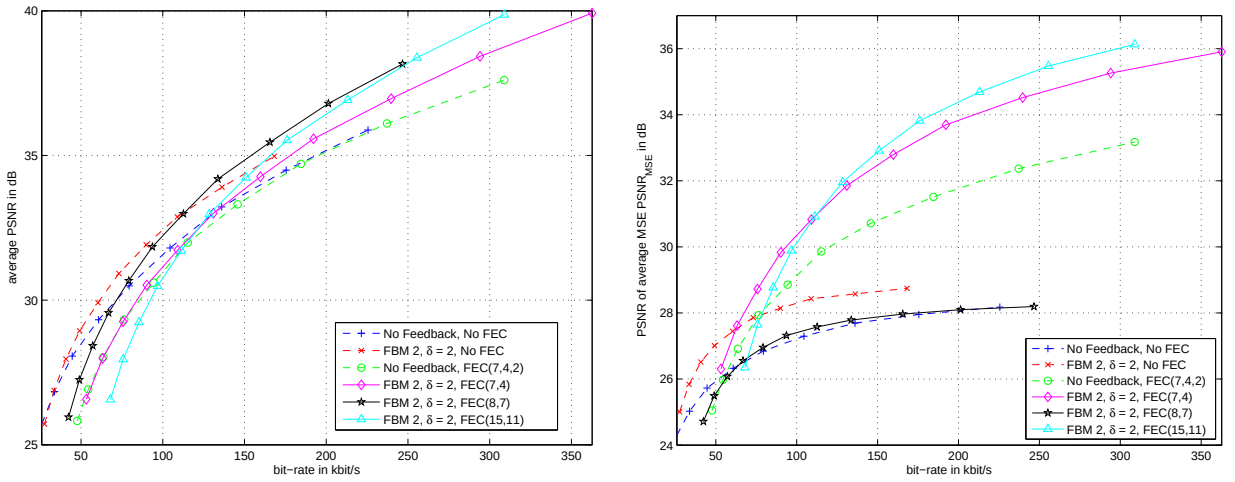
(a) average $\overline{\text{PSNR}}$

(b) PSNR of average MSE $\text{PSNR}_{\overline{\text{mse}}}$

Figure 6.14: PSNR over bit-rate performance for different codes and optimized mode selection scheme indicated as $(n, k, \tilde{\varepsilon})$ for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 99$: (a) Average $\overline{\text{PSNR}}$; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$.

The performance of the combination of FEC and feedback mode 2 for different channel codes is shown in Figure 6.15 compared to several other modes presented previously. Again it is observed that the introduction of feedback provides significant performance gains. For the average PSNR again no FEC or very high–rate codes perform best, but the difference of the different mode is not that significant. For the average MSE and higher bit-rates it is again suggested to use a stronger FEC even if combined with feedback information.

It is concluded that for higher error rates the application of even simple end–to–end forward error correction schemes can improve the performance significantly even for low–delay applications. The increased amount of overhead due to headers as well as parity packets is compensated by significantly lower error rates for the video application. It is also suggested that one could basically omit any error resilience tools in the application and mainly rely on end–to–end forward error correction. Beyond this, even more flexible frame–wise adaptive error correction schemes may be applied. This is not discussed in further detail for packet–lossy channels but in chapter 8 we introduce such advanced schemes relying on very similar principles for the transmission of video over wireless channels.



(a) average $\overline{\text{PSNR}}$ (b) PSNR of average MSE $\text{PSNR}_{\overline{\text{mse}}}$

Figure 6.15: PSNR over bit-rate performance for different codes with feedback mode 2 indicated as (n, k) compared to no FEC as well as for *foreman*, error pattern 10, AEC, and $N_{\text{MB/DU}} = 99$: (a) Average $\overline{\text{PSNR}}$; (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$.

6.3 FEC for Moderate-Delay Applications

6.3.1 Forward Error Correction with H.264/AVC

Up to now we have only investigated the transmission of low–delay video over error–prone channel. This obviously limits the application of source and channel coding methods that add additional delay, especially FEC schemes. For less time–critical applications such as video streaming or multimedia broadcast applications additional delay can be tolerated without sacrificing user perception. There exist a huge amount of possibilities to exploit delay in end–to–end video streaming applications such as FEC, selective retransmissions, video streaming over TCP, etc. For details, references, and optimization we refer the interested reader for example to [CM06].

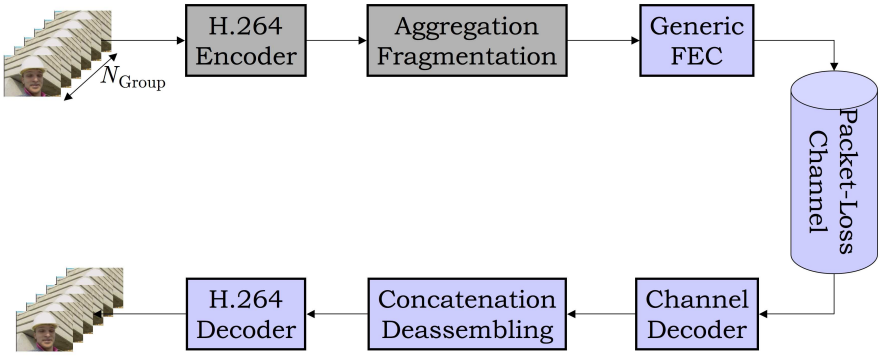
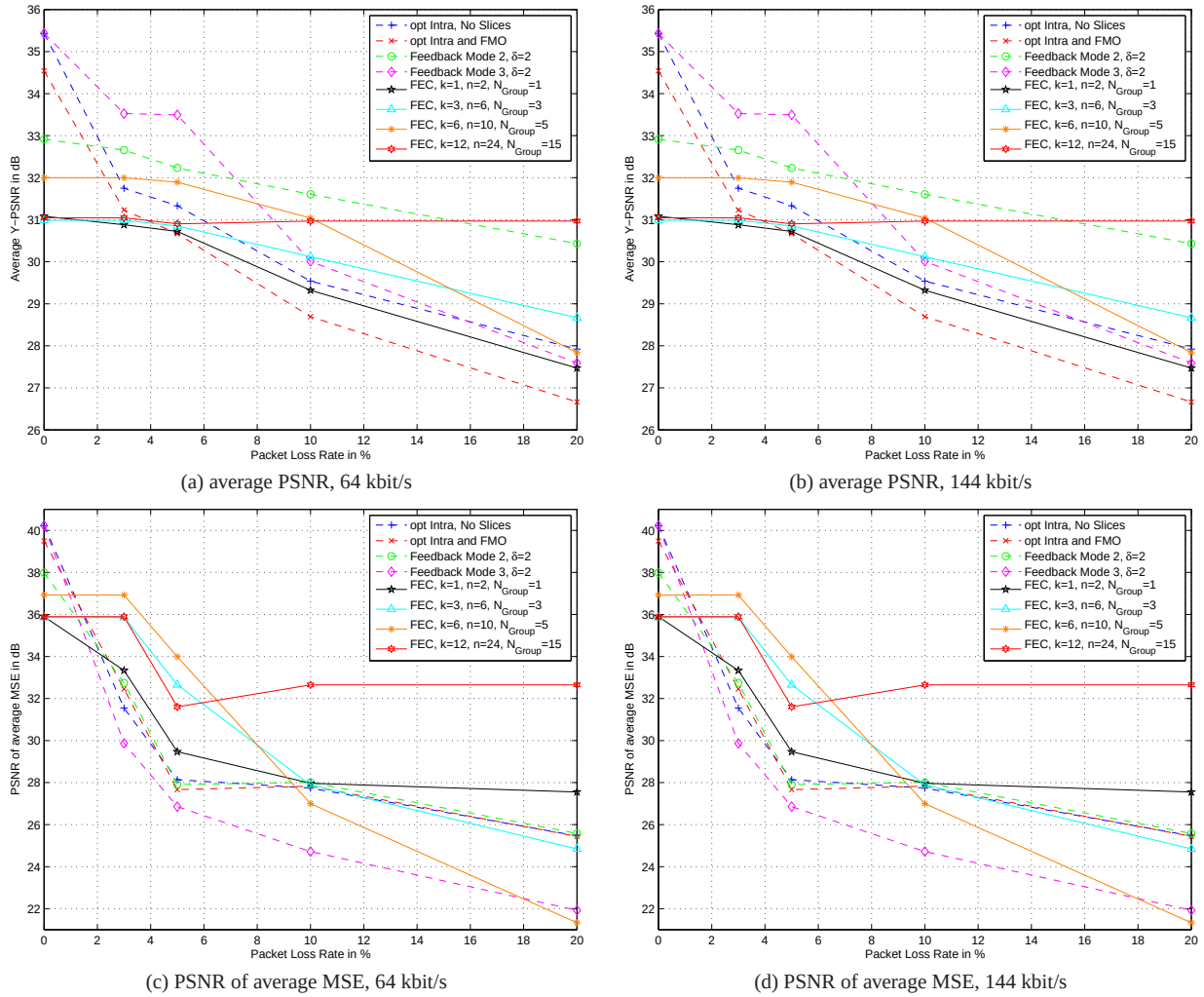


Figure 6.16: Video transmission system for moderate–delay applications over a packet–lossy channel applying H.264, frame aggregation, fragmentation and generic FEC.

In following we will introduce a very simple yet powerful system based on generic FEC according to [RS99]. The system shows the potential of FEC in packet-lossy environments and also serves as a benchmark for the more advanced system introduced in subsection 6.3.2. An overview of the system is shown in Figure 6.16: A standard–compliant H.264/AVC encoder is used which allows to add error resilience features, namely distortion–optimized mode selection.

Each frame is packetized into a single NAL unit. However, rather than transmitting each frame separately in an RTP packet, we apply MTAP rules according to the RTP payload specification for H.264 [WHS+05] for the aggregation of up to N_{Group} frames to form a group–of–frames. The size of the entire group–of–frames, B_{Group}, is determined by the size of each single NAL unit, b_n, and some overhead H_{MTAP} to form the MTAP. The resulting single packet is then further processed by applying fragmentation according to the RTP payload specification as already introduced in section 6.2 for a single NAL unit[8]. The fragmentation is again applied such the the generated RTP packets do have almost equal size. The output of the fragmentation unit is a sequence of k RTP packets to be forwarded to the generic FEC function which itself produces $n-k$ parity packets to form a binary (n,k) code. In case that the not all packets can be recovered it is assumed that the entire N_{Group} packets are dropped. The delay of the system is mainly determined by the algorithmic delay of the aggregation of N_{Group} frames resulting in an encoding delay of $\Delta T^{(\mathcal{Q}_e)} = N_{\text{Group}}/f_s$.



(a) average PSNR, 64 kbit/s

(b) average PSNR, 144 kbit/s

(c) PSNR of average MSE, 64 kbit/s

(d) PSNR of average MSE, 144 kbit/s

Figure 6.17: Average PSNR, $\overline{\text{PSNR}}$,as well as the PSNR of the average MSE over packet error rate for *foreman*, QCIF, with frame rate $f_s = 7.5$ fps for different bit-rates for simple generic FEC with different group–of–frame sizes, N_{Group}, optimized mode selection compared to good other different error resilience tools (dashed lines).

[8]This mode is disallowed in the current draft of the RTP payload specification, but might be reconsidered in future updates or extensions.

The following simulation results are again based on the test conditions [Wen01], we exclusively use QCIF test sequence *foreman*. Figure 6.17 shows the performance in terms of average $\overline{\text{PSNR}}$ as well as the PSNR of the average MSE over packet error rate for *foreman*, QCIF, with frame rate $f_s = 7.5$ fps for different bit-rates and simple generic FEC with different group–of–frame sizes, N_{Group}. No additional error resilience tools are applied. Basically, the system streams a pre–encoded video which has been produced for compression efficiency only. The parameters, N_{Group} and n have been selected such that the amount of RTP packets per video frame is 2 except for $N_{\text{Group}} = 15$ where $n = 24$ is selected as this is virtually the highest value supported by RFC2733[9]. Different channel codes have been selected at code rate $r = k/n = 1/2$ or slightly higher. The code $(n = 2, k = 1)$ obviously only repeats the packets, the $(n = 6, k = 3)$ and $(n = 10, k = 6)$ are reconstructed by shortening the Hamming codes $(n = 7, k = 4)$ $(n = 15, k = 11)$, respectively.

The relatively low code rates of around 0.5 obviously significantly reduce the coding efficiency as seen for error rate 0 in Figure 6.17 for all bit-rates and performance measures. For error–prone transmission, let us first consider the case with the average PSNR as the measure of interest according to Figure 6.17a) and (b). With increasing error rates we observe that the performance of the generic FEC schemes increases significantly compared to previously presented error resilience tools. For lower error rates as well as for low delays, i.e., low n, the performance still cannot exceed the performance of conventional error resilience schemes for the codes with $r = 0.5$. The higher rate $(n = 10, k = 6)$–code performs well also for low error rates, but its performance degrades for higher error rates as the case for any other schemes except for the extended Golay code with $(n = 24, k = 12)$. With $N_{\text{Group}} = 15$, i.e., an encoding delay of 2 seconds, the performance of the scheme is almost independent for the considered code rates and even outperforms feedback mode 2 for higher error rates. Note that even the simple repetition code $(n = 2, k = 1)$ performs surprisingly well and outperforms FMO.

If the PSNR is replaced the average MSE as measure of interest (see Figure 6.17 c) and d)), the advantages of applying FEC over conventional error resilience is even more apparent. For all bit-rates and all error patterns, the reduction of the packet loss probability results in the phenomenon that all FEC schemes with code rates at about 0.5 outperform other error resilience tools including feedback modes. Only for the more bursty error pattern, we observe the drawbacks of an FEC-based solution. This effect had already been observed in Figure 4.20c). To summarize, the avoidance of transmission errors provides constant quality and avoids artifacts from packet losses. Therefore, from this result it is obvious that one should use FEC even in packet–loss networks, especially if the applications allows at least moderate delay, artifacts from packet–loss are unacceptable, and pre-encoded streams are distributed.

6.3.2 PTVC with Unequal Erasure Protection

From the previous results we have seen that FEC schemes can improve the performance of moderate–delay application significantly. However, the simple setup in subsection 6.3.1 has two disadvantages. First, channel adaptivity has not been considered, i.e., the channel code rate is selected to match the expected loss rate. This effect has been observed as codes with different code rates perform differently for different error rates. Secondly, the system suffers from the fact that in case of the failure of recovery usually an entire group of video frames is lost, and error propagation occurs in the presented system. Therefore, we will in the following address the problem of channel adaptivity and graceful degradation in case of losses.

[9]For simplicity, the group–of–frame size N_{Group} has been chosen as a multiple of 75 since the sequence encoded sequence consists of 75 frames. This restriction can obviously be removed in practical systems.

Theoretically, the problem of graceful degradation or successive refinement with the loss or reception of equally important packets has been introduced as multiple description coding [WWZ80]. However, the approaches taken within multiple description coding are not directly applicable for modern video codecs with highly complex prediction mechanisms resulting in catastrophic error propagation in case of packet losses. A well known practical approach for Multiple Description Coding (MDC) video relies on progressive coding combined with UXP as introduced in section 5.3. Early work on this subject for video transmission has for example been introduced by McCanne et al. [MVJ97] as well as by Horn et al. [HSLG99]. We extend this work based on the already introduced PTVC together with an UXP approach as presented in section 5.3 for transmitting pre–encoded video over packet–lossy channels with low to moderate delays.

Figure 6.18 shows an overview of the proposed system. The video sequence is encoded with the PTVC with fixed GOP size N_{GOP} and GOP interleaving as introduced in subsection 3.4.4. Together with the interleaved GOPs we also generate a quality–rate function $\mathcal{Q}(R)$ for the embedded bitstream for each GOP. We provide both, the average PSNR as well as the PSNR of the averaged MSE. The algorithmic encoding delay results in $\Delta T^{(\mathcal{Q}_e)} = N_{\text{GOP}}/f_s$.

At the time the decoder requests the streaming of the specific sequence, or a multicast/broadcast session is initiated, UXP based on RS codes is applied to the embedded bit–stream as introduced in subsection 5.3. Optimized rate allocation is performed for a given transmission bit-rate R whereby we assume that the transmitter is aware of the expected packet loss rate $\tilde{\varepsilon}$ with the assumption that the generated packets are lost statistically independent. The generated packets are encapsulated into a RTP/UDP/IP packet with header size $N_{\text{Header}} = 40$ bytes and finally routed to the receiver. There, channel decoding is performed based on the correctly received packets as shown in Figure 5.7. After GOP de–interleaving the individual embedded bit–streams for each of the N_{GOP} frames within this GOP are forwarded to the PTVC decoder. Although within one GOP mismatches between the encoder's reference buffer and the decoder buffer might occur, with the start of a new GOP and the reception of an I–frame, the drift is removed. In [Sto02] we have extended this framework by the introduction of modified predictive frames being generated according to a feedback message. However, this scheme requires online encoding and only provides marginal gains such that we omit the presentation in further detail.

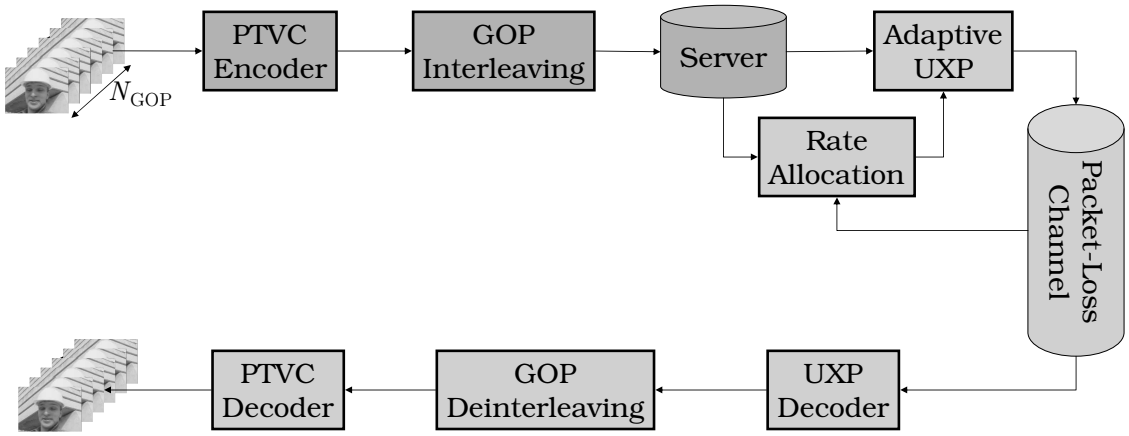
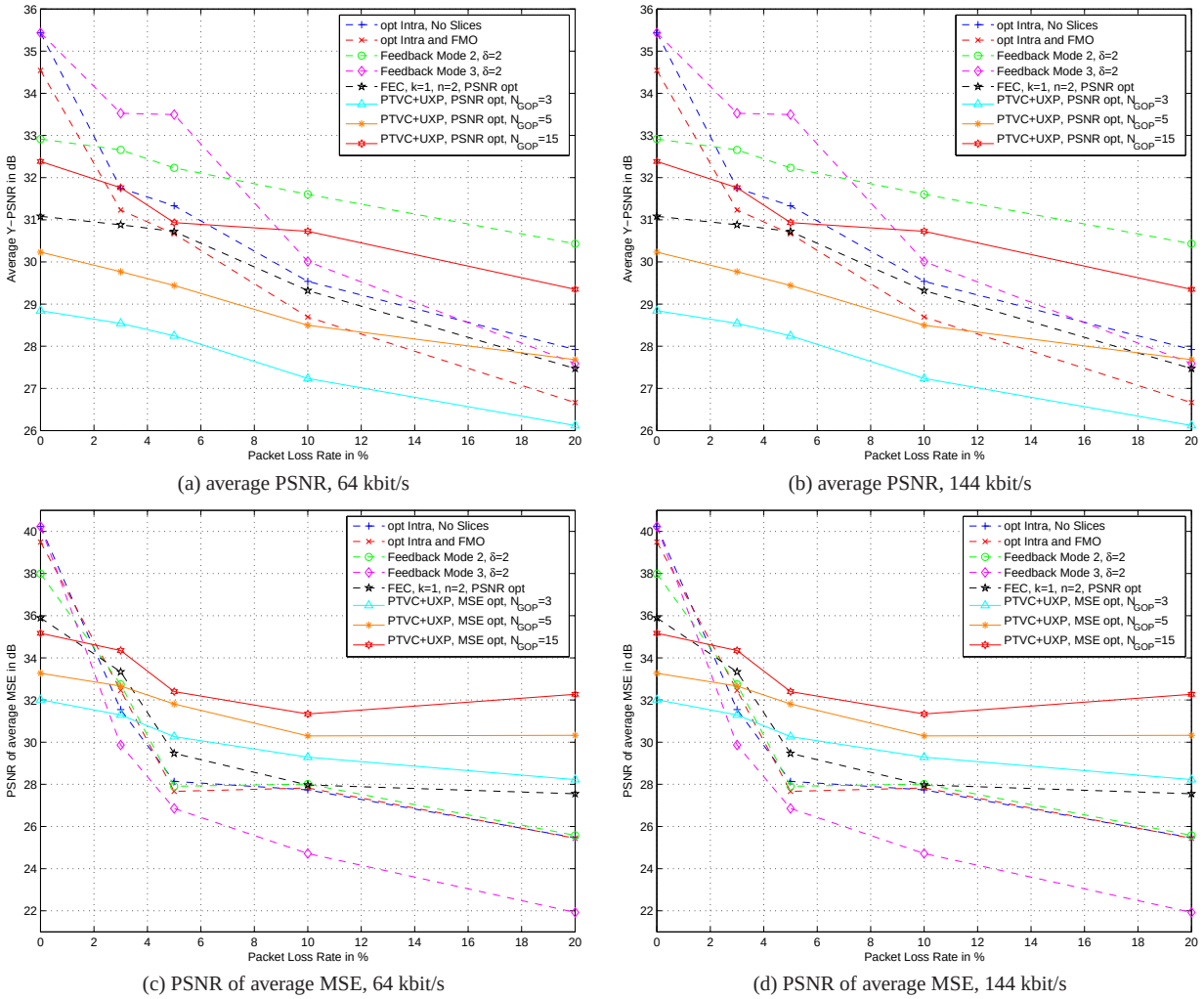


Figure 6.18: System Diagram: PTVC combined with UXP.

Simulation results are again based on the common test conditions [Wen01]. We exclusively use QCIF test sequence *foreman* and restrict ourselves to schemes where the number of transmission packets N_{packet} has been selected such that we have exactly two transmission packets for each frame. This seems to be a good tradeoff between erasure protection capability and IP/UDP/RTP

overhead N_H. Also, we select the feedback bit-rate for the PTVC to be at 80% of the residual bit-rate R_{residual} which is the bit-rate available for the video and the FEC excluding the overhead of the transport protocol. For an even more advanced system one could adapt the number of packets per frame as well as the feedback bit-rate to the erasure statistics of the channel. However, for simplicity and also as usually one wants to avoid to encode streams for different channel conditions we exclude this optimization in the following. We compare our MDC approach based on the combination of PTVC and UXP to schemes without FEC as well as the simple generic FEC scheme as presented in subsection 6.3.1.



(a) average PSNR, 64 kbit/s

(b) average PSNR, 144 kbit/s

(c) PSNR of average MSE, 64 kbit/s

(d) PSNR of average MSE, 144 kbit/s

Figure 6.19: PSNR over packet error rate for *foreman*, QCIF, with frame rate $f_s = 7.5$ fps for different bit-rates and different error resilience tools: PTVC with UXP for different GOP sizes, H.264/AVC with FEC for different group–of–frame sizes N_{Group}, and FMO, different feedback approaches. H.264/AVC uses no error resilience, FMO checkerboard pattern with 2 packets per frame.

Figure 6.19 shows the performance in terms of average PSNR, $\overline{\text{PSNR}}$, as well as the PSNR of the average MSE, $\text{PSNR}_{\overline{\text{mse}}}$ over packet error rate for *foreman*, QCIF, with frame rate $f_s = 7.5$ fps for different bit-rates and PTVC combined UXP for different gop sizes N_{GOP} compared to H.264/AVC with FEC as well as FMO. In all cases we have two packets per video frame except for the generic FEC case with code $(n = 24, k = 12)$ due to the restrictions in RFC2733.

Basically it is observed that the UXP approach cannot provide better quality than the simple FEC

schemes. The reason for this is mainly the lower compression efficiency of the PTVC compared to H.264/AVC as well as the necessity to introduce more frequent I–frames to remove drift artifacts. We have seen that approaches without allowing drift by lowering the feedback bit-rate perform even worse. Still, for low bit-rates with channel–adaptive rate allocation as well as well adjusted gop size N_{GOP}, the performance is similar to that of the H.264/AVC with FEC. For the PSNR being the measure of interest, larger GOP sizes with $N_{\text{GOP}} = 15$ are preferable. However, also note that for higher bit-rates the loss in compression efficiency is even higher resulting in overall less performance of the UXP approach. With the MSE being the measure of interest it is observed that especially for higher bit-rates and smaller delays with a lower number of packets per GOP, n, the UXP schemes outperform the simple FEC schemes. In this case the multiple description principle can provide benefits, whereas for longer delays simple FEC with high compression efficiency outperforms the PTVC+UXP if the MSE is the measure of interest. Only for specific cases with relatively low amount of packets available for a GOP, the combination of PTVC and UXP can provide benefits over simple FEC schemes.

It is also worth to mention that the simple FEC schemes could easily be further enhanced by applying adaptive channel coding, or by using better codes, e.g. RS codes with MDS property instead of simple binary codes. Therefore, only if the coding efficiency of scalable or progressive coding schemes can be enhanced significantly and we have a specific scenario mind with relatively low delay, then the application of scalable coding with UXP may outperform simple FEC schemes with non–scalable coders in packet-loss environments.

6.4 Summary: Major Observations for Video Transmission over Packet Loss Channels

In this chapter we have introduced different tools and combinations of tools to successfully transmit video over packet–lossy channels. Although the amount of tests was limited to only selected video sequences, selected bit-rates and only a several packet loss patterns, we are confident that the obtained results provide sufficient indications for the following conclusions.

- The transmission of video is mainly disturbed by errors due to missing information, but also to a high degree due to error propagation as a consequence of the predictive encoding. Both artifacts should be combatted when designing a video transmission system for packet–lossy channels. As subjective tests are infeasible to quantify the quality of packet-lossy video transmission, *we propose and also have used two different objective measures: The average Peak Signal–to–Noise Ratio (PSNR) as well as the PSNR of the Mean Square Error (MSE)*. This approach provides a good indication on the subjective quality, and using both measures allows to detect deficiencies of one or the other measure for certain cases.

- We have *designed, developed and implemented a toolbox for error resilient video transmission based in H.264/AVC*: This toolbox includes among others, slice-structured coding, Flexible Macroblock Ordering (FMO), random and channel-adaptive intra updates, interactive error control, as well as advanced error concealment in the decoder. Specifically, *we have developed a generic scheme for the estimation of the expected decoder pixel distortion in the encoder taking into channel conditions*. This estimation has been used in a rate-distortion optimized mode selection for optimized and standard-compliant video encoding for packet lossy channels. Furthermore, *we have developed several interactive error control schemes*

that make use of the multiple reference frame concept in H.264/AVC and rate-distortion optimized mode selection in the test model encoder.

- When designing a system it is important to understand the application constraints in terms of acceptable delay, available information, as well as online or offline coding schemes. In any case a compression–efficient video standard is essential in networked video, as the saved video bit-rate can be used by error resilience or lower layer tools to avoid errors or limit the effects or errors.

- In general it is observed that the avoidance of errors can significantly improve the performance. This may most easily be accomplished by the application of retransmissions. However, if retransmissions are not feasible due to delay or protocol restrictions, the application of FEC can be very helpful, as errors can be completely avoided. This results in stable and consistent video quality over the entire sequence.

- Multiple description video schemes can be achieved by the use of a progressive video codec and UXP. In our case this is realized by the combination of the PTVC and a Reed-Solomon code. In addition, it is essential to apply an appropriate rate allocation scheme. However, multiple description video schemes outperform simple FEC schemes with single layer video codecs only in specific application constraints, usually for lower delay applications. It is essential that the compression efficiency of the applied progressive source coder is not significantly smaller than the one of non-scalable coders.

- We have also shown that even for low–delay conversational applications the application of simple erasure FEC schemes can be helpful especially for higher packet loss rates.

- However, for most low-delay conversational applications, the use of FEC or retransmission schemes is impossible. In this case losses cannot be avoided, but is is most important to limit error propagation.

- The most effective approach to stop error propagation is to exploit feedback messages from the decoder and combine it with multiple reference frame in modern video codecs. Different feedback modes have been discussed being more or less suitable for different error rates, feedback delays, as well as performance measures.

- Without the availability of feedback messages, channel–optimized mode selection schemes perform well and should be used by encoders on packet-loss environments: Note that such extensions can be introduced in an H.264/AVC compliant manner by only modifying the mode selection process in the video encoder. Optimized mode selection schemes have been derived and applied.

- Other error resilience tools have been investigated: advanced packetization schemes such as slice-structuring or FMO together with appropriate error concealment methods might be applied, but the results in this chapter cannot provide sufficient evidence for usefulness of these approaches in packet loss environments.

- In any of the discussed case it is helpful to have indication on the expected channel conditions at the transmitter. This knowledge allows to appropriately select error resilience tools and can be integrated in optimized mode selection schemes in the video encoder as well as in the selection of appropriate error resilience tools.

In this chapter we have only investigated end–to–end solutions assuming that packet losses cannot be avoided. For wireless transmission schemes the situation is different. The packet loss rate can be controlled more easily, many more adaptations can be performed over different layers. The remaining three chapters are dedicated to this subjects.

7

Video Services in UMTS-like Environments

7.1 System Overview

Mobile handheld devices are nowadays equipped with color displays and cameras. They also have sufficient processing power which allows presentation, recording, encoding and decoding of video sequences. In addition, emerging and mobile systems provide sufficient bit-rates to support video communication services. Nevertheless, bit-rate will always be a scarce resource in wireless transmission environments due to physical bandwidth and power limitations and thus, efficient video compression is required.

The 3GPP architecture nowadays supports a significant amount of packet-switched services. They are typically defined end-to-end between two terminals, or a terminal and a central server. The virtual connection is logically split into core and radio access network. Figure 7.1 shows a high-level architecture and the corresponding network nodes and protocol layers. Of major interest is obviously the radio-related part, as it is most critical in terms of bit-rate, resource consumption, and reliability.

Although any IP-based application can be considered as a packet-switched service, 3GPP restricts itself to a specific set of multimedia services that are supported by the network architecture. This greatly simplifies interoperability and allows efficient and high-quality integration of real-time multimedia services. Among others, this is achieved by the provision of a certain network QoS, which means that a suitable radio (access) bearer with clearly defined characteristics and functionality is set up across the air interface. The bearer definition includes all aspects to enable the provision of a contracted QoS like control signaling, user plane transport, and QoS management functionality. On top of such radio access bearers, end-to-end multimedia services including video are supported, e.g. conversational services, streaming services, download services as well as broadcast and multicast services. As already highlighted in subsection 2.1.5, different applications generally have different service constraints, which leads to the selection of different codec settings, delivery protocols, and radio bearer settings to support the required end-to-end QoS.

According to subsection 2.1.5, a service may have certain QoS requirements that must be met to satisfy the user. To achieve this, especially the radio bearer must provide adequate means at the

physical, MAC, or RLC layer, as shown in Figure 7.1. Furthermore, QoS can also be provided end-to-end within the application layer. In this section we omit any cross-layer design aspects, but investigate the appropriate selection of different tools in the radio layer and the application layer for different applications. The design follows a strict layer separation, which is commonly preferred in complex 3GPP network architectures.

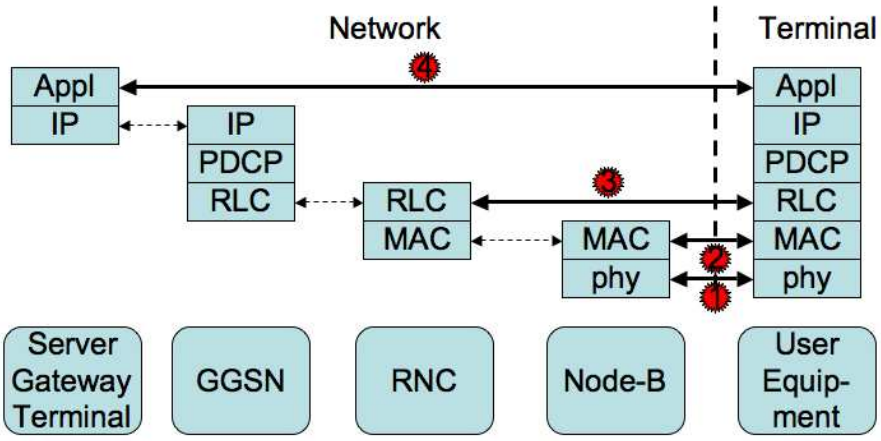


Figure 7.1: Conversation video transmission in RTP-based mobile environment.

Specifically, the robustness and the suitability of the H.264/AVC design for mobile video applications are discussed. Basically, the same system diagram as shown in Figure 6.1 also applies when modeling the transmission over a mobile packet-switched channel. Transmission modes or specific bearers in wireless systems can be established to fulfill different QoS requests. H.264/AVC provides on top of compression efficiency means to be integrated easily into existing and emerging packet networks. Furthermore, H.264/AVC is a general purpose codec and can be used in different applications under different service constraints. The codec is designed such that it interfaces very well with packet-based networks such as RTP/IP.

Although shared radio channels such as EGPRS or HSDPA may exploit radio resources more efficiently, dedicated channel modes are used for conversational as well as other low–delay applications as it is essential to provide bearers with a certain QoS in terms of guaranteed bit-rate, delay, as well as error rate. Hence, we will in the following concentrate on UMTS dedicated channels that can guarantee a certain bit-rate and in case of UMTS also a fixed error rates by the use of fast power control or diversity techniques such as multiple antenna systems in combination with forward error correction. As already discussed in section 2.3.1, two different bearer modes are supported, Unacknowledged Mode (UM) and Acknowledged Mode (AM). The properties of UMTS dedicated channels for the transmission of packet–based applications have already been introduced in subsection 2.4.4. In contrast to the discussion in chapter 6 for the transmission over wired Internet backbones, four major differences apply in the characteristics when transmitting over UMTS dedicated channels:

1. The RTP/IP-packet length influences the loss rates as it is more likely that a longer packet is hit by bit-errors or is delivered by an erroneous link layer packet as discussed in subsection 2.4.4.

2. In contrast to the backbone channel when transmitting over dedicated mobile links the delay of the IP-packet is also influenced by its length due to the transmission delay. Therefore, for low-delay applications a CBR-like rate control as introduced in subsection 3.2.5 is essential.

3. The end–to–end principle is no more completely fulfilled. In entry gateways to the mobile system, e.g., in support nodes, the IP-packets might be modified by applying header compression. Therefore, the costly header overhead can be reduced for mobile transmission systems.

4. Finally, the mobile link layer can provide different QoS in terms of reliability and delay by the use of power control, FEC, or retransmission protocols.

The influence of these additional aspects for video transmission systems will be discussed in the following. In the remainder of this chapter, we will ignore any influence of a core network as well as the connection of the communication partner to the network. We assume that the communication partner is connected to the entry gateways to the mobile network by a perfect connection. The additional influences can in general be ignored [RSL+01] but for stringent delay considerations they may be taken into account [G1196].

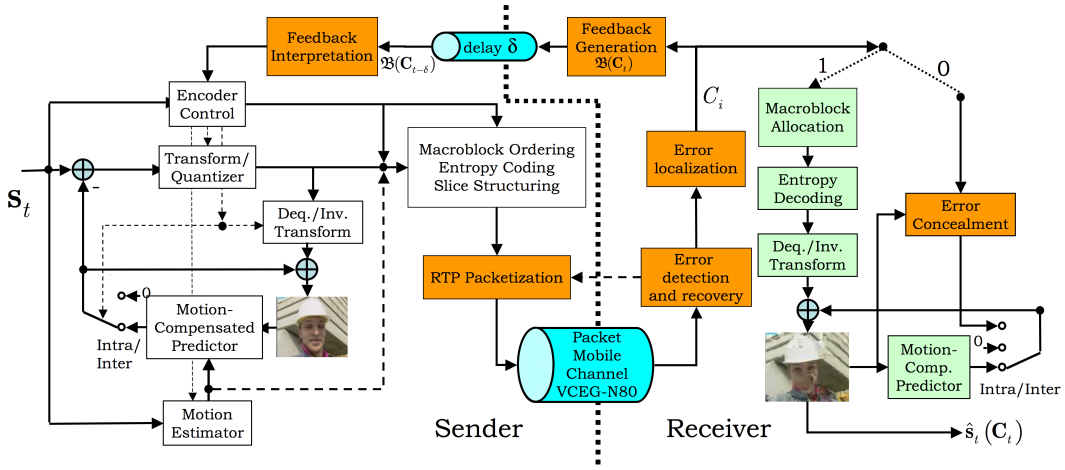


Figure 7.2: Conversation video transmission in RTP-based mobile environment.

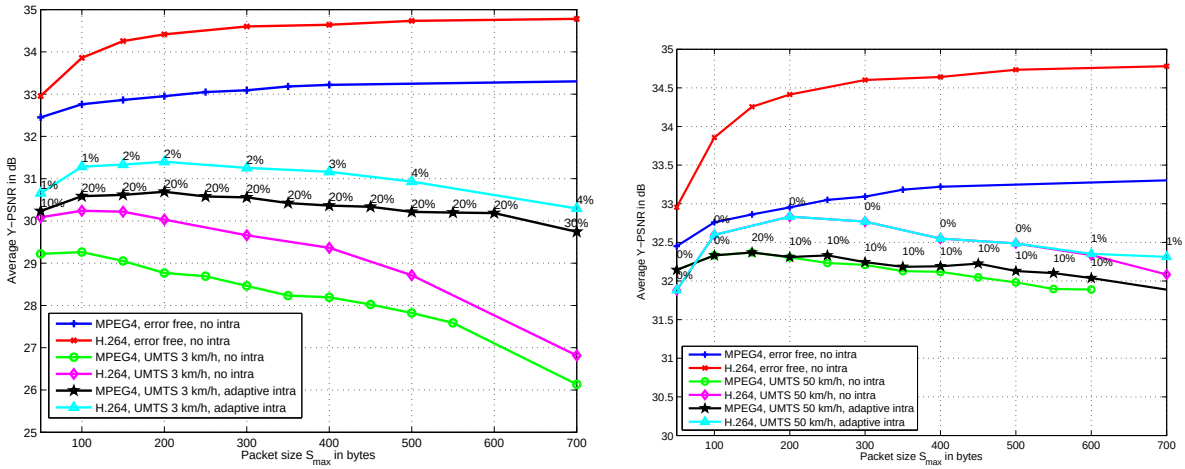
Figure 7.2 shows the use of a hybrid video codec such as H.264/AVC in an RTP-based conversational environment in UMTS. For the following simulations the tools as introduced in chapter 3 and also applied in chapter 6, specifically in section 6.1, are now applied also to mobile communication environments. The detailed setup and parameters are presented in the following. The following simulations are based on [RSL+01].

7.2 Experimental Results for Conversational Video

The experimental results based on the test conditions in [RSL+01] show the influence of the selection of different error resilience and error concealment features for the quality of the decoded video for MPEG-4 as well as H.264/AVC. The bearer cases (3) and (4) according to Table 2.4 are used. As performance measure the average luminance $\overline{\text{PSNR}}$ is chosen as suggested in [RSL+01]. The average is taken over at least 200 transmission and decoding runs. Protocol overheads are appropriately taken into account according to [RSL+01]. Results are shown for the QCIF test sequence

foreman of length 10 seconds coded at a constant frame rate of 7.5 fps with a single intra frame in the beginning and only P-frames afterwards. Encoding is performed using the packetization mode according to Figure 3.14b) with different $S_{\max}$.

Figure 7.3 shows results for the $\overline{\mathrm{PSNR}}$ over the packet size $S_{\max}$ in bytes for different mobile speeds for MPEG-4 as well as H.264/AVC. For reference, also the error-free performance for both codecs with different packetization modes is shown. Depending on the packetization mode, H.264/AVC outperforms MPEG-4 between $0.5 - 1.7$ dB for this specific mobile test case. The degradation due to the use of shorter video packets is obvious from the results. A degradation of 1.8 dB is observed for $S_{\max} = 50$ and H.264/AVC as the advanced prediction modes of this codecs are significantly restricted. The degradation for MPEG-4 is less, but still significant. For increased slice length, the degradation is less significant. For the case without any intra updates



(a) Bearer Case (3): UMTS 3 km/h

(b) Bearer Case (4): UMTS 50 km/h

Figure 7.3: $\overline{\mathrm{PSNR}}$ over the slice length $S_{\max}$ in bytes for different mobile speeds for MPEG-4 as well as H.264/AVC. For reference, also the error-free performance for both codecs with different packetization modes is shown.

and without video packetization, the performance degrades significantly for both codecs in the range of $5 - 7$ dB for slow mobile speeds when compared to the error-free case. The introduction of slices is beneficial as can be seen. For low speeds, packet sizes in the range of 100 bytes are preferable, for higher speeds resulting in lower error rates about 150-200 bytes seem to be more appropriate. However, the degradation compared to the error-free mode without packetization is still more than 4 dB. When using intra updates the performance can be improved significantly. For MPEG-4 encoding in Figure 7.3, random intra refresh is applied. The best-performing random intra refresh ratio in percentage is shown for each simulation point. Overall, longer packet sizes are tolerable with intra updates. However, the appropriate intra update ratio strongly depends on the packet size and the channel behavior as observed from Figure 7.3. In addition, also sequence characteristics and bit-rates influence the appropriate percentage of intra updates. For H.264/AVC, the performance for channel-adaptive intra refresh is shown. In this case, the mode selection assumes a channel with statistically independent video packet losses $\tilde{\varepsilon}$, whereby $\tilde{\varepsilon}$ corresponds to the likelihood, that a packet of size $S_{\max}$ is lost. The expected video packet loss rate p in percentage is indicated next to each simulated point. The combination of slices with channel-adaptive intra updates outperforms the use of a single tool when transmitting over wireless transmission channels. To obtain an estimation of the loss rate $\tilde{\varepsilon}$ at the receiver, a feedback channel for example based on RTCP receiver reports can be used in practical systems.

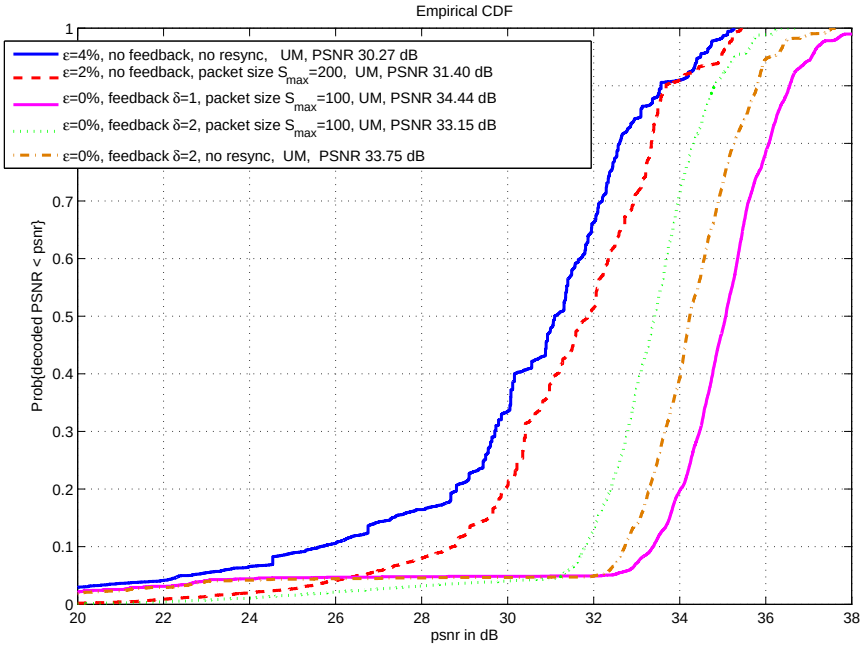


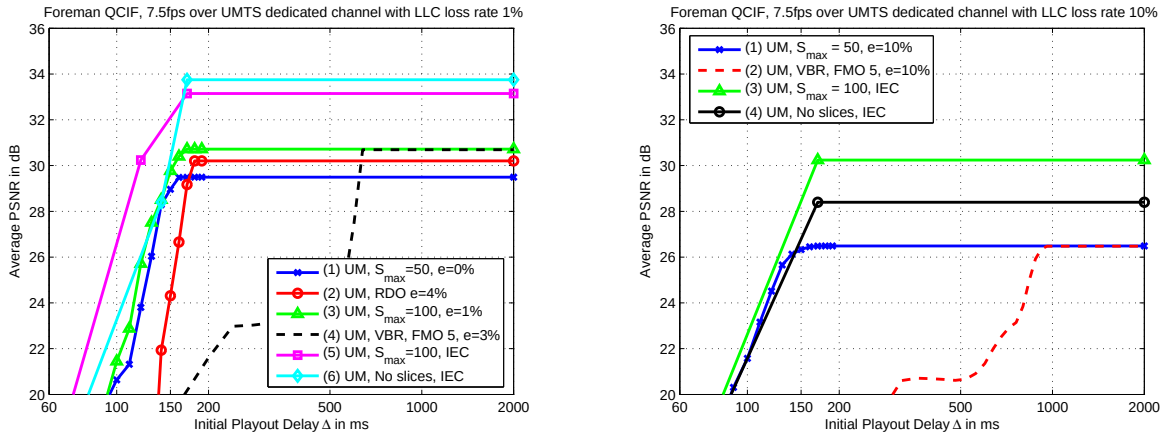
Figure 7.4: Distribution of decoded PSNR for each frame for different transmission modes and bearer case (3) according to Table 2.4.

In a further set of experimental results a system which exploits multiple reference frames and fast network feedback is investigated based on Interactive Error Control (IEC) as introduced in section 6.1.5. We restrict our simulation results to feedback mode 2 with five reference frames. Figure 7.4 shows the cdf of the decoded PSNR for each video frame for different error-resilience strategies, namely for channel-optimized intra updates (indicated by $\tilde{\varepsilon} > 0$ in the legend) with slice structuring (referred to as packet size in bytes in the legend) and IEC (referred to as feedback) with feedback mode 2 with and without slice structuring for delay $\delta = 1$ and $\delta = 2$ frames. This corresponds to a round trip time of about 150 ms and 250 ms, respectively. The legend also shows the assumed loss probability $\tilde{\varepsilon}$ for the channel-optimized video encoding, the feedback delay δ, the transport mode, and the $\overline{\mathrm{PSNR}}$ for each case. Only bearer case (3) according to Table 2.4 is used.

Specifically, for $\delta = 2$ and slice-structured coding, the results indicate that channel-optimized intra refresh with slice structuring and feedback mode 2 perform very similar. The feedback mode might still be beneficial in this case as the complex estimation of the decoder distortion is not necessary for the feedback case. However, more interesting is the case with feedback and no slice structuring: A single frame packetization mode provides significantly higher $\overline{\mathrm{PSNR}}$. This is initially surprising as the packet loss rate is still much lower when multiple slices are used as can be observed from the cdf: The probability of bad frames (PSNR below 22 dB) is higher with no packetization. However, in case of no errors the increased coding efficiency when not using slices provides many frames with significantly higher PSNR than for slice structuring. As we avoid error propagation, the correctly received frames are indeed error-free, which is not the case without feedback.

In a further set of simulations as shown in Figure 7.5 , we present the results in $\overline{\mathrm{PSNR}}$ over the initial playout delay at the decoder, Δ, for different modes. Thereby, for the delay components as introduced in subsection 2.1.4 only the encoder buffer delay $\Delta T^{(\mathcal{B}_\mathrm{e})}$ and the transmission delay $\delta^{(\mathcal{C})}$ on the physical link is considered, i.e., $\Delta \triangleq \Delta T^{(\mathcal{B}_\mathrm{e})} + \delta^{(\mathcal{C})}$. Additional processing delay as well as

transmission delays on the backbone networks may add in practical systems. Figure 7.5a) shows



(a) Bearer (3) with 1 % LLC loss rate

(b) Bearer (1) with 10 % LLC loss rate

Figure 7.5: Average PSNR for different low-delay video transport systems over initial playout delay Δ for UMTS dedicated channel with link layer error rates of 1% and 10% (see Table 2.4, Bearer (3) and (1), respectively).

the performance for bearer case (3) according to Table 2.4, whereby all curves (1)-(6) use the UM. The system designs according to curve (1)-(4) do not make use of any fast feedback channel, but it is assumed that the encoder has knowledge of a link layer loss rate of about 1%. Therefore, these modes may also be applied in video conferencing with multiple participants. Specifically, in curve (1), (2), and (3) CBR encoding with bit-rates 50, 60, and 52 kbits/s, respectively, is applied to match the bit-rate of the channel taking into account the overhead. Curve (1) relies on slices of maximum size $S_{\max} = 50$ bytes only, no additional intra updates to remove error propagation is introduced. Curve (2) in contrast neglects slices, but uses optimized intra up-dates with $p = 4\%$, curve (3) uses a combination of the two features with $S_{\max}$ =100 bytes and $p = 1\%$. The transmission adds a delay of about 170 ms for the entire frame. For initial delays lower than 170 ms, NAL units are lost due to late arrival.

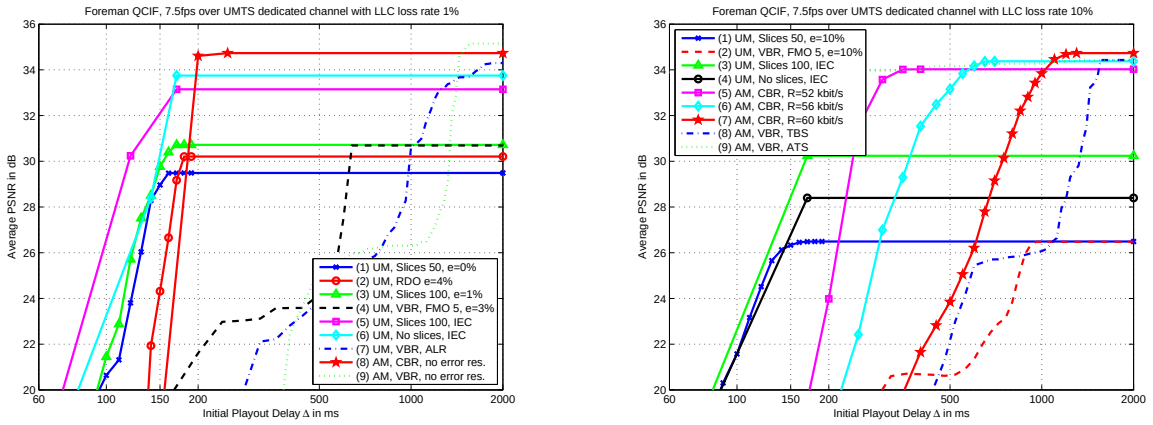
For initial play-out delays above this threshold, only losses due to link errors occur. If small initial play-out delay is less essential, a similar performance can be achieved by VBR encoding combined with FMO with 5 slice groups in checkerboard pattern as well as optimized intra with $p = 3\%$ as shown in curve (4). However, the VBR encoding causes problems for low-delay applications in mobile bottleneck links, and therefore, a CBR-like rate control is essential. Systems according to curves (5) and (6) assume the availability of a feedback channel from the receiver to the transmitter, which is capable to report the loss or acknowledgement of NAL units: IEC with feedback mode 2 with $d = 2$ is applied. For the slice mode with $S_{\max} = 100$ bytes as shown in curve (5) significant gains can be observed for delays suitable for video telephony applications, but due to the avoided error propagation it is even preferable to abandon slices and only rely on IEC as shown in curve (6). This is in line with the findings in Figure 7.4. The average PSNR is about 3 dB better than the best mode not exploiting any feedback.

In summary, the application of advanced error resilience schemes in standard-based environments can improve the video quality significantly. Therefore, if feedback information is available and completely lost frames are tolerable to some extent, it is better to exchange slice structured coding to obtain better compression efficiency. In case of available feedback, the simple IEC system design without requiring the expected decoder distortion and slice structuring outperforms many

highly-sophisticated error resilience schemes as long as the delay of the feedback is reasonably low.

7.3 Experimental Results for Moderate-Delay Applications

For applications for which short delay is less essential other modes may enhance the quality over UMTS networks. Therefore, in Figure 7.6 we have added some system configurations which generally add higher delays. Specifically, Figure 7.6a) shows in addition to the modes in Figure7.5 a) the performance for bearer case (3) according to Table 2.4 using the UM for curve (7) and the acknowledged mode (AM) for curves (8) and (9). For the system according to curve (7) the feedback is exploited for application layer retransmission of RTP packets and VBR encoding is used. It is obvious that this mode is not suitable for low-delay applications, but if end-to-end delay is of less relevance, it provides better performance than any other scheme relying on methods in the video layer. In addition, it is important to note that no error resilience is necessary in this case. Finally, curve (8) and (9) show the performance of CBR encoded video and VBR encoded video, respectively, with matching bit-rates for the acknowledged mode. The performance of the CBR mode is excellent even for lower delays, but at least 200ms of initial play-out delay must be accepted which makes the applicability for conversational modes critical, but not infeasible, if the system supports fast retransmissions. It is also observed that for VBR encoding low-delay applications cannot be well supported, but if initial playout delays of a few seconds can be accepted, VBR encoding with acknowledged mode on the link layer provides the best overall performance.



(a) Bearer (3) with 1 % LLC loss rate

(b) Bearer (1) with 10 % LLC loss rate

Figure 7.6: Average PSNR for different advanced video transport systems over initial playout delay Δ for UMTS dedicated channel with link layer error rates of 1% and 10% (see Table 2.4, Bearer (3) and (1), respectively).

For the UMTS bearer case (3) with 10% link layer loss rate, advanced transport system enhances the overall system performance significantly. Figure 7.6b) shows that significantly better performance can be achieved by the use of the acknowledged mode, but only for initial play-out delays well over 300 ms according to curve (5) with CBR and bit-rate 52 kbit/s. Interestingly, if the initial play-out delay is increased, one can also support higher bit-rates resulting in higher quality. This behaviour has been exploited in the HRD specification of H.264 where it was recognized that an encoded stream is contained not just by one but many leaky buckets [RCCR03]. Finally, the

systems according to curve (8) and (9) show the performance for VBR encoded video in case of Timestamp-Based Streaming (TBS) and Ahead-of-Time Streaming (ATS) over the AM mode. It is interesting that with ATS low playout delays can be achieved, but obviously this requires that the data cannot be generated online. In addition, in practical systems some kind of start-up delay might occur due to TCP-like congestion control. It is also worth to note that the performance of video over the 10% link layer loss bearer does not differ significantly from the 1% one if the initial play-out delay constraints are not stringent.

7.4 System Design Guidelines

The obtained results allow comparing different options for different applications. A summary of proposed video and transport features for the video test sequence Foreman over a 64 kbps UMTS link with RLC-PDU loss rates of 1% and 10% is provided in Table 7.1. For more details we refer to [Sto06]. In addition to the discussed schemes, we have also added guidelines and results for the case of broadcast applications. Details on the system design for broadcast applications, especially the use of erasure based FEC are discussed in [Sto06] and [SSW+08].

Table 7.1: Proposed video and transport features for different applications with performance in terms of delay and average PSNR for different RLC-PDU loss rates and QCIF video sequence Foreman coded at 7.5 fps.

Video Application	Video Features	Transport Features	1% RLC-PDU loss rate		10 % RLC-PDU loss rate	
64 kbit/s UMTS transmission scenario			Delay	PSNR	Delay	PSNR
Download-and-Play On-demand streaming	VBR, no error res., playout buffering	ATS, AM on RLC	> 1.5 sec	35.2 dB	> 1 sec > 10 sec	34.4 dB 35.1 dB
Live streaming	CBR/VBR, no error res., playout buffering	TBS, AM on RLC	> 250 ms	34.7 dB	> 400 ms > 1.5s	34.0 dB 34.7 dB
Broadcast	VBR, regular IDR, no other error resilience	FEC, long TTI, fragmentation	> 5 sec	34.5 dB	> 5 sec	32.0 dB
Conferencing	CBR, Intra Updates, Slices	UM	> 150 ms	30.7 dB	> 150 ms	26.5 dB
Telephony	CBR, IEC, no slices	UM	> 150 ms	33.7 dB	> 150 ms	30.2 dB

For download-and-play as well as on-demand streaming applications with initial playout delays beyond one to two seconds, the video should mainly be encoded for compression efficiency, that is, relatively relaxed variable bit-rate (VBR) rate control and no explicit error resilience features. The reliability should be provided in the link layer by using Acknowledged Mode (AM) on the radio link layer. The resulting delay jitter can be compensated with playout buffering. Ahead-of-time streaming (ATS) can be applied if the receiver buffer has sufficient size. For higher error rates, the quality even scales with the initial playout delay as the jitter can be better compensated for larger delays. For live applications, timestamp-based streaming (TBS) must be applied. For lower requested delays in range of 250–500 ms, CBR-like rate control is also preferable. Video error resilience is still not essential as the AM provides sufficient QoS.

For broadcast applications without any feedback, it is proposed to apply extensions of the FEC. This can be accomplished by longer TTIs in the physical layer and/or application layer FEC.

In addition, we suggest using regular IDR frames for random access and error resilience, but a relatively relaxed rate control. For low RLC-PDU loss rates, the FEC-based scheme is almost as good as the one relying on feedback mechanisms. However, for higher loss rates the acknowledged mode with RLC layer retransmission outperforms application layer FEC by about $2-3$ dB. In any case, the application layer FEC adds delay.

For low-delay applications, the video encoder should apply CBR-like rate control and the transport and link layer basically must operate in UM without any retransmission or deeply interleaved FEC schemes. The video application itself must take care to provide sufficient robustness. In case that feedback is not available or only limited to reporting statistics, for example, in conferencing applications, more frequent intra-MB updates based on robust mode decision as well as slice-structured coding are proposed.

However, compared to the acknowledged mode significant degradations in the video quality must be accepted, especially if the RLC-PDU loss rates are high. Therefore, the physical layer must provide sufficient QoS to support these applications. For video telephony, the fast feedback channel can be exploited for interactive error control (IEC). No additional means of error resilience are necessary. In this case and for low loss rates, the achieved video quality is significantly better, about 3 dB, when compared to video error resilience without feedback. The degradation compared to reliable download-and-play applications is only about 1.5 dB.

The system design for video communication in emerging multi-user systems such as HSDPA and MBMS require new design principles. We skip the details in this work but refer to our work in [JSL04], [LJSB04], [Sto06], [LGSW07] and [SSW$^+$08].

7.5 Summary

In this chapter, system design guidelines for video services in UMTS have been introduced. The following observations and findings are of major importance:

- Depending on the application, video services have certain QoS requirements. To fulfill these requirements, preferably radio bearers should provide sufficient QoS, but also end-to-end features in the application layer are necessary in certain service environments.
- H.264/AVC is a general purpose codec and can be used in different applications under different service constraints. The codec is designed such that it interfaces very well with packet-switched mobile networks using for example RTP/IP.
- For download-and-play as well as on-demand streaming applications video compression efficiency and variable bit-rate (VBR) rate control is most essential. The sufficient QoS should be provided by the radio bearer.
- For live applications, CBR-like video rate control is preferable for dedicated channels. Video error resilience is not essential as the radio bearer can provide sufficient QoS.
- For broadcast applications without any feedback, extended FEC on the physical layer and/or application layer FEC is an appropriate choice. Video should be coded with regular IDR frames for random access and error resilience. Application layer FEC can trade performance and delay.
- For low-delay mobile video applications over dedicated channels, the video should apply strict CBR-like rate control. The radio bearer should operate in Unacknowledged Mode

(UM) without any retransmission. The video application itself must take care to provide sufficient robustness.

- In case that feedback is not available or only limited to reporting statistics as, for example, in conferencing applications, more frequent intra-MB updates based on robust mode decision as well as slice-structured coding should be used.
- For video telephony, a fast feedback channel shall be exploited for interactive error control (IEC). No additional means for error resilience are necessary. This results in surprisingly good performance without significantly sacrificing radio efficiency.

8

Mobile Conversational Video

8.1 Common Channel and Service Environment

The transmission of low delay video over mobile channels has proven to be the most challenging application according to the results in chapter 7. It is observed that error-free performance can basically not be guaranteed on the radio bearer. Therefore, either it is necessary to compensate the problems by high transmission power or the quality of the application is degraded. System design and cross-layer solutions improving the transmission of low-delay video over mobile fading channels are therefore highly desirable. To understand the potentials of such solutions this chapter is exclusively dedicated to combine many of the concepts, functions and interfaces introduced in the previous chapters to realize systems for low-delay mobile video transmission.

We basically follow the system design as initially introduced in Figure 2.1, but refine the system diagram to address mobile conversational video transmission. A refined system diagram is shown in Figure 8.1. Note that in this system, due to the delay constraints of conversational video, each video frame s_t is treated separately for the transmission. However, we obviously still use MCP video coding. The system as shown in Figure 8.1 consists of functional blocks as well as messages and interfaces which had been introduced to a large extent in all previous chapters. We briefly summarize the functions of each of the blocks and introduce the applied tools within each of the functional blocks with focus to mobile conversational video transmission.

To allow comparison of different schemes, we focus on a simple, but representative channel model based on subsection 2.4.3 and specifically the block-fading AWGN channel which is completely characterized by only the few parameters according to Table 2.3:

- The average SNR $\mathcal{E}_s/N_0$ is set to 4 dB for the remainder of this chapter as this value somehow represents a worst-case coverage case in mobile systems [Kno97].
- The channel gain α is exclusively modeled as Rayleigh distributed random variable independently chosen for each radio slot.
- The power control factor γ_f is set to 1 unless it is stated otherwise.
- The total number of symbols per radio slot, N_v, is selected to $\{1600, 3200, 4800, 6400\}$ binary symbols in the remainder.

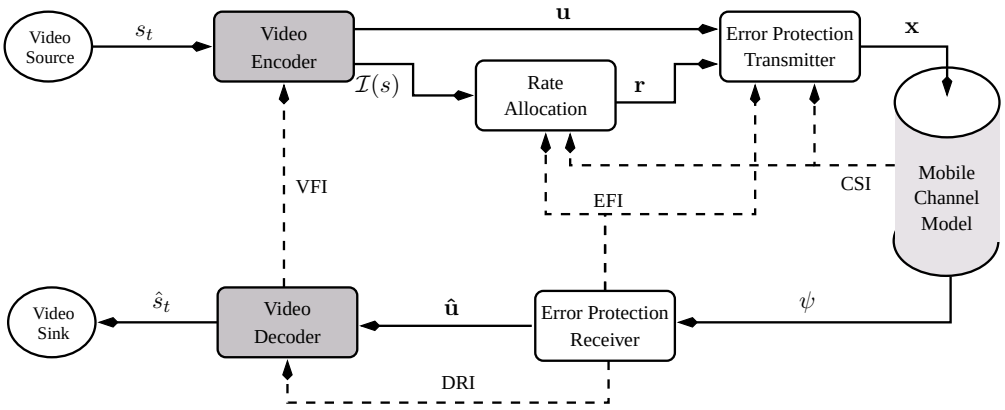


Figure 8.1: Generic System Design for Mobile Conversational Video Transmission.

- The TTI of the transmission slots, T_f, is set to 25 ms and therefore the maximum symbol rate after channel coding is determined as $\{64, 96, 128, 160\}$ kbit/s.

The video produces frames with constant frame rate f_s. For the setup in this chapter, exclusively a frame rate of $f_s = 10$ fps is applied. To address low-delay service requirements of mobile video transmission, the transmission of a single video frame s_t at time t can be spread at most over F radio slots. Therefore, interleaving or retransmission is always limited to a maximum of F slots. We also enforce for the video source strict rate control ensuring that a video frame does not spread over more than F slots. With the setting applied in here, each video frame is interleaved over $F = 4$ slots.

In terms of evaluation, we compare the sequence of original frames, s_t, to the sequence of received frame, $\hat{s}_t$. The metrics applied in this section follow the objective metrics introduced in section 2.1.2. Specifically, we apply

- the average channel PSNR, $\overline{\mathrm{PSNR}}_{\hat{\mathcal{C}}}$, as defined in (2.12) (we drop the channel index $\hat{\mathcal{C}}$ for the remainder of this section and write $\overline{\mathrm{PSNR}}$ only), and
- the PSNR of the average MSE, $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ as defined in (2.13).

To ensure sufficient statistics, at least $N_{\mathcal{C}} = 50$ independent channel simulations have been performed, where each sequence consists of 10 seconds video. With the applied frame rate of $f_s = 10$ fps, this results in a sequence length of $N_s = 100$ frames and in a total amount of $N_e = 5000$ video frames for each experiment. To obtain long test sequences the original sequence is looped appropriately. We report the confidence bounds as introduced in (2.17) and (2.19) for a confidence level of $\beta = 90\%$. To obtain insight in channel bandwidth gains and video quality improvements, for all cases we generally report the performance in terms of quality over total bit-rate for source and channel coding in kbit/s which is parametrized by the symbol size N_v and results in values of $\{64, 96, 128, 160\}$ kbit/s.

We now continue introducing the specific tools that are available for the functions and interfaces in Figure 8.1. Several combinations of the tools will be investigated by extensive simulations and comparisons and conclusions are provided. The following components from the video coding toolbox will be used and assessed in the introduced channel and service environment:

- single layer H.264/AVC video coding with CBR rate control according to chapter 3, (3.21),

- data partitioning in H.264/AVC as briefly introduced in chapter 3, and
- the Progressive Texture Video Coding (PTVC) as introduced in section 3.4.2.

In all cases, we use five reference frames in the video encoding process.

Video Feedback Information (VFI) from the video decoder in the receiver to the video encoder in the transmitter has proven to be very beneficial for conversational video applications in chapters 6 and 7. Therefore, we apply interactive error control (IEC) based on feedback mode 2 for H.264/AVC based video coding. In case the PTVC is used, then the VFI is exploited by transmitting some information on the decoded layer from the decoder to the open-loop encoder. The VFI may also be used to report long-term video frame loss statistics to the video encoder. This information is used for the purpose of channel–adaptive intra updates.

Furthermore, the following features from the error protection toolbox will be integrated in this transmission and service environment:

- Forward Error Correction (FEC) schemes as introduced in sections 4.2 and 4.3: Specifically, high memory punctured convolutional systematic codes with memory $\mu = 96$, mother code rate $c = 1/7$ and puncturing period $\omega_p = 32$ are applied. The set of accessible code rates is determined as $\mathcal{S}_r = \{\omega_p/(\omega_p + i)\}_{i=0}^{\omega_p(c-1)}$.
- Interleaving and channel access schemes as introduced in 4.3.1.
- Different Automatic Repeat reQuest (ARQ) schemes as introduced in section 4.1.5.
- Channel State Information and Power Control as introduced in section 4.1.4,
- Regressive UEP and Far End Error Decoding (FEED) as introduced in section 4.4.

It is worth mentioning that instead of high memory punctured convolutional systematic codes, other codes such as rate compatible Turbo codes [RM00] could easily be applied in this framework. The adaptation of the setup and as well as the optimization procedures are straightforward. Furthermore, as we have shown in chapter 4, the above error protection tools can be very well modeled by the use of bounding technologies. We have made use of these bounds to estimate the performance of punctured high-memory convolutional codes. As the real performance is very close to these bounds, the results presented in here are generic, and any channel code with the performance according to (4.8) can serve as an enabler to provide the required functionality.

The error-protection schemes include the exploitation of Error–Protection Feedback Information (EFI) messages for ARQ as well as Channel State Information (CSI) messages for power control schemes. Therefore, they are integrated into the error protection tools. However, those messages are also potentially used in some transmitter rate allocation schemes along with the Source Significance Information (SSI), expressed as the importance vector $\mathcal{I}$ in Figure 8.1. To perform the optimized rate allocation of source and channel coding rates, as well as for the purpose of power allocation and UEP profile design, different rate allocation schemes will be used for each video frame:

- a rate control for optimized source-channel rate allocation for single layer H.264/AVC and for data partitioning according to section 8.2.3,
- a *source–channel–optimized rate allocation* as introduced in section 5.4.2 for the purpose of transmitting PTVC encoded video with FEED,

- an optimized rate allocation for transmitting PTVC encoded video with ARQ schemes will be introduced in section 8.3.3,
- an optimized power-source rate allocation for transmitting PTVC encoded video with power control schemes will be introduced in section 8.3.4, and
- for performance estimation information-theoretic bounds are all based on assuming asymptotically good codes operating at the cutoff rate according to the presentation in (4.8).

Table 8.1: Common simulation parameters for transmission of H.264/AVC and PTVC coded video sequences over block fading channels using different error control schemes.

Description	Notation	Simulation parameter
video sequence		Foreman
number of transmitted frames	N_e	≥ 5000
spatial resolution		QCIF
number of reference frames		5
frame rate	f_s	10fps
PTVC I-frame ratio	R_{IP}	1
No. of decoder operations for Fano	C_{lim}	16
radio slots per frame	F	4
transmission block length	N_v	$\{1600, 3200, 4800, 6400\}$ channel symbols
puncturing period	ω_p	32
memory	μ	96
mother code rate	$1/c$	$1/7$
average channel SNR	$\mathcal{E}/N_0$	4 dB
confidence interval	β	90%
distribution of channel gain	p_α	iid Rayleigh
transmission time interval	T_f	25 ms
total bit-rate		$\{64, 96, 128, 160\}$ kbit/s

Table 8.1 summarizes all common simulation parameters. Furthermore, Figure 8.1 only shows a general high-level system diagram which hides some implementation details for specific system designs. The diagram will be refined in the course of this chapter with the introduction of specific source coding and error protection schemes. However, the restrictions in terms of service requirement as well as the channel characteristics are maintained throughout the rest of this chapter.

8.2 H.264/AVC-based Source-Channel Coding System

8.2.1 System Overview

In a first set of system design proposals, we apply the generic architecture in Figure 8.1 and specify some of the components in more detail. In this section, we concentrate exclusively on video coding schemes based on H.264/AVC. Furthermore, for the purpose of error protection tools, we restrict ourselves to FEC tools only. The updated system diagram is shown in Figure 8.2.

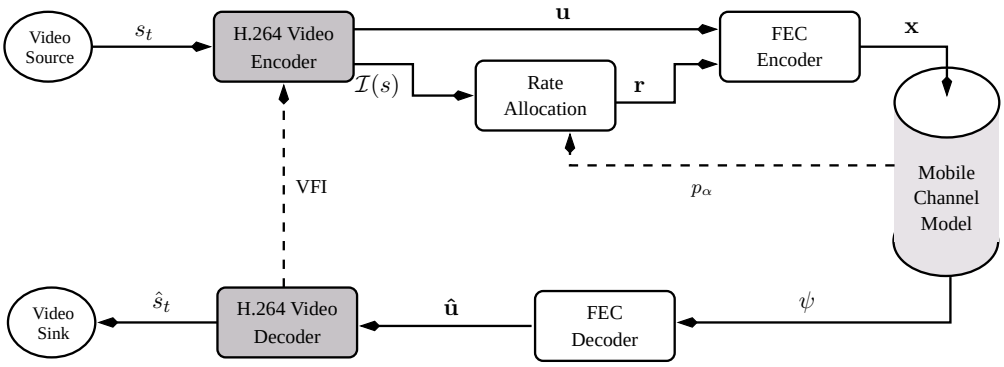


Figure 8.2: H.264/AVC-based System for Mobile Conversational Video Transmission.

For the case of H.264/AVC we will discuss two modes in further detail: One mode relies on single layer coding and EEP, the second mode relies on data partitioning and UEP. As from an implementation and optimization point of view, the single layer coding and EEP mode can be treated as a special case of data partitioning, we will introduce the more general system in the following.

A system making use of data partitioning and UEP may provide benefits as in case of poor channel conditions, not necessarily all data will be lost, but at least the most important part can be decoded and displayed. This requires that the most important data is protected stronger against severe channel impairments than less important data – resulting in a typical UEP scheme. We have introduced data partitioning into the H.264/AVC standard [SM01]: Following this proposal and some refinements, H.264/AVC supports the partitioning of I-slices in two partitions, P-slices in three partitions, and B-slices in a single partition. In the remainder, we focus on P-slices, as they can be viewed as a superset of the other slice types. The partitioning is organized such that different syntax elements are assigned to different partitions. Without going into all details, partition A contains all control and header information as well as any data related to the MCP process. Whereas data partitioning in previous standards such as MPEG-4 and H.263 version 2 only distinguishes two partitions, in H.264/AVC the second partition is further split into intra-related information assigned to partition B and inter-related information assigned to partition C. This enhancement acknowledges the fact that in error-prone environments more frequent intra information is necessary to limit error propagation. This intra information is in general more important – especially for the subjective quality – than the inter information. For more details on data partitioning we refer the reader to [SM01, Wen03, Kar03, SB04].

The considered data partitioning system is shown in Figure 8.3: The video encoding process distributes the sequentially encoded frames $n = 1, \ldots, N$ and distributes the syntax elements among the partitions. Each partition is separately entropy coded resulting in three partitions A, B and C and a partition size vector, $\mathbf{k}(q) \triangleq \{k_{\mathrm{A}}(q), k_{\mathrm{B}}(q), k_{\mathrm{C}}(q)\}$. The partition size depends on the applied quantization parameter q. In H.264/AVC, each partition may be transmitted in a separate NAL unit, but can be concatenated using fragmentation unit packets as specified in the RTP payload format for H.264 [WHS$^+$05]. This concatenation results in a partly "embedded" bit–stream for each H.264/AVC video frame.

Then, FEC is applied using channel coding rates $\mathbf{r} \triangleq \{r_{\mathrm{A}}, r_{\mathrm{B}}, r_{\mathrm{C}}\} \in \mathcal{S}_r^3$ for each of the partition. The encoded video frame is FEC-encoded, interleaved, and transmitted over the mobile channel. We use the regressive UEP scheme as introduced in section 4.4 and already applied in section 5.4 for SPIHT image transmission. The receiver decodes the received channel code word using the

FEED algorithm according to section 4.4 providing also error detection and code word shortening. Finally, the bit-stream is depacketized and only entirely correct partitions are forwarded to the H.264/AVC decoder.

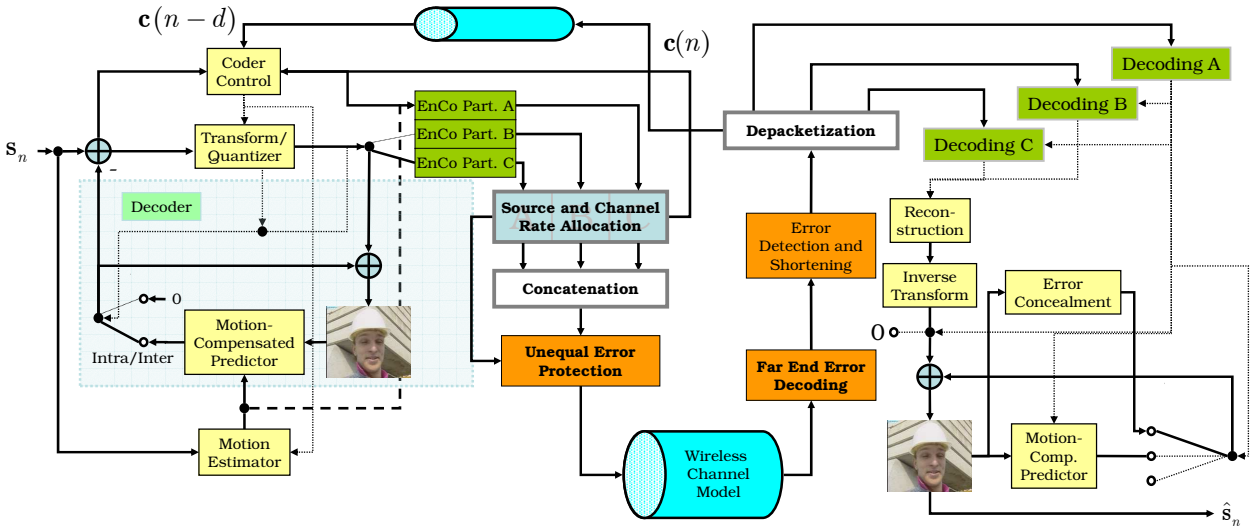


Figure 8.3: Mobile conversational video system with UEP and data partitioning.

The video decoder processes the received data partitions as follows: In case of the loss of partition A regular error concealment is carried out, otherwise the header and motion information is decoded and used in the motion compensation process. If partition C is available the Displaced Frame Difference (DFD) is reconstructed and added to the motion-compensated frame. If partition B is also available, the intra macroblocks are added to the reconstructed frame. However, if partition B is lost, temporal error concealment is applied. Note that the system using a single layer codec with EEP corresponds to the special case of data partitioning by assuming that all data of a slice are contained in data partition A, and by setting $k_{\mathrm{B}}(q) = 0$ and $k_{\mathrm{C}}(q) = 0$.

The loss of partitions or entire frames and the subsequent concealment causes a display problem for missing frame. In addition, the mismatch in the reference buffers at the encoder and the decoder also cause error propagation. To compensate this basically two effective counter-measures have previously been discussed in this work:

1. Without the availability of VFI, channel-adaptive mode selection schemes perform well and should be used by encoders in packet-loss environments. In this case the amount of intra-coded information is increased to compensate the error propagation taking into account the expected channel conditions.

2. The most effective approach to stop error propagation is to exploit fast VFI and combine it with IEC. If applied in this chapter we restrict ourselves to feedback mode 2 (synchronized reference frames), for which we assume that the encoder includes exactly the same algorithm as the decoder to reconstruct the reference buffers in case of losses. For the feedback delay from encoder to decoder we assume that the encoder has a δ-frames delayed version. The actual delay value depends on the system implementation and influences the performance of the IEC system. Different δ values are used in the following to show the influence of this system parameter.

The extension of these two error resilience modes for data partitioning is straightforward with the use of the Multiple–Decoder Distortion Estimation (MDDE)-scheme as introduced in subsection 6.1.4, and with feedback mode 2 as introduced in subsection 6.1.5. For implementation details we refer to [Kar03]. In the remainder of this section, no slice structured coding or any other packetization mode within a single frame is applied taking into account the results in chapters 6 and 7.

What remains open in the system design is the appropriate selection of the QP q for each frame as well as the channel coding rate vector $\mathbf{r}$ for each of the partitions in case of data partitioning or for the single frame in case of using H.264/AVC single layer. The video encoder decides for each video frame to be encoded and transmitted next on the quantization parameter q as well as the channel coding rate vector $\mathbf{r}$. The total amount of source and channel coding symbols, $N_{\text{tot}}(q, \mathbf{r})$ must not exceed the total number of available channel symbols $N_{\max} \triangleq N_v F$, i.e.,

$$N_{\text{tot}}(q, \mathbf{r}) \triangleq \frac{k_{\text{A}}(q)}{r_{\text{A}}} + \frac{k_{\text{B}}(q)}{r_{\text{B}}} + \frac{k_{\text{C}}(q)}{r_{\text{C}}} \leq N_{\max}. \tag{8.1}$$

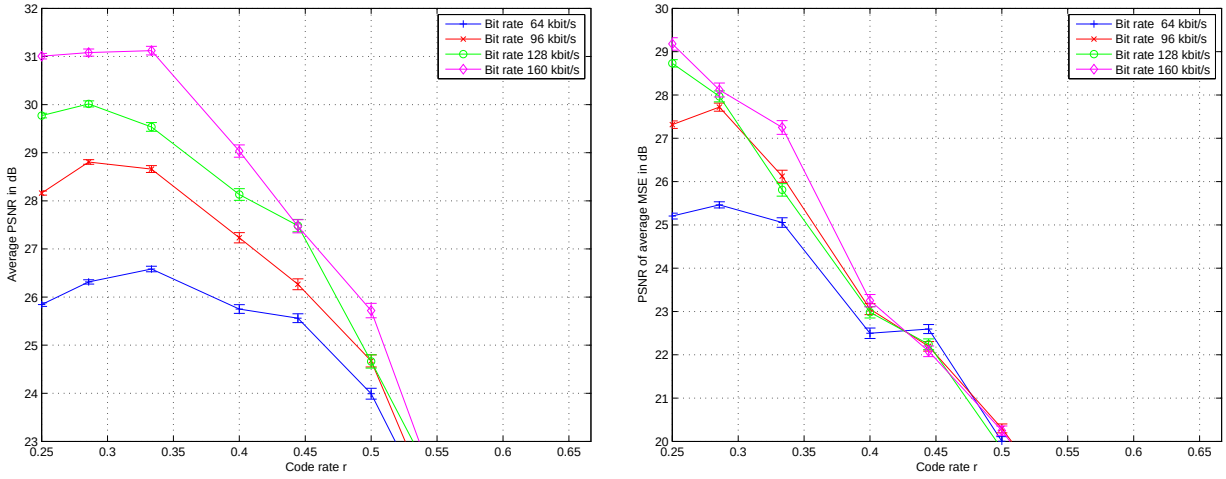
Note that the size of the partitions $\mathbf{k}(q)$ is directly controlled by the applied QP q. Due to discrete channel coding rates, determined by the puncturing patterns, as well as by the required signaling overhead and the termination overhead, the true N_{tot} is slightly different than presented. Our implementation takes care of these issues, but for the sake of clarity in the presentation we stick with the model described in (8.1).

The quantization parameter and channel coding rate selection is part of the rate allocation procedure. We assume that the rate allocation has access to some SSI as well as to the expected channel conditions, expressed by the distribution, p_α, of the channel gains. It is obvious that due to the interleaving over F slots the packet loss probability depends on the applied channel coding rate r. This follows exactly the discussion in chapter 4: Figure 4.2 shows the outage probability P_{out} over code rate r for different number of slots, F. Assuming that we use the cutoff rate curves for performance measurements, then for an average SNR of 4 dB and interleaving over $F = 4$ radio slots, we observe the expected outage probability for a certain partition in case we apply a certain channel coding vector. Since the channel coding rate r is directly proportional to the maximum supported video bitrate, which itself determines the encoding quality, there exists an obvious trade-off between residual losses and encoding quality. The selection of q for each frame as well as the channel coding rate vector $\mathbf{r}$ should therefore maximize some expected quality. This optimization will be addressed in the following sections along with some simulation results.

8.2.2 Constant Code Rate FEC and H.264 Single Layer System

In our initial system design we select straight-forward H.264/AVC based single layer video coding and EEP to fulfill the service requirements, especially to maintain a low delay. For error protection, an FEC with a fixed code rate $r \in \mathcal{S}_r$ is applied and the resulting code word is interleaved over $F = 4$ radio slots. Therefore, for each video frame at most $rN_{\max}$ bits are available. We apply an H.264/AVC video encoder with a strict CBR rate control according to (3.21) to ensure that the number of bits for each frame does not exceed $rN_{\max}$. As we expect that error-free transmission cannot be guaranteed, we may apply error resilience features during the encoding process, specifically channel-optimized mode selection with a configurable expected packet loss rate $\tilde{\varepsilon}$ as well as IEC with feedback mode 2. By the use of the VFI, the video encoder may have access to the $\tilde{\varepsilon}$. Note that this loss rate is directly determined as $\tilde{\varepsilon} = P_{\text{out}}(r)$.

Performance Results without Error Resilience Tools



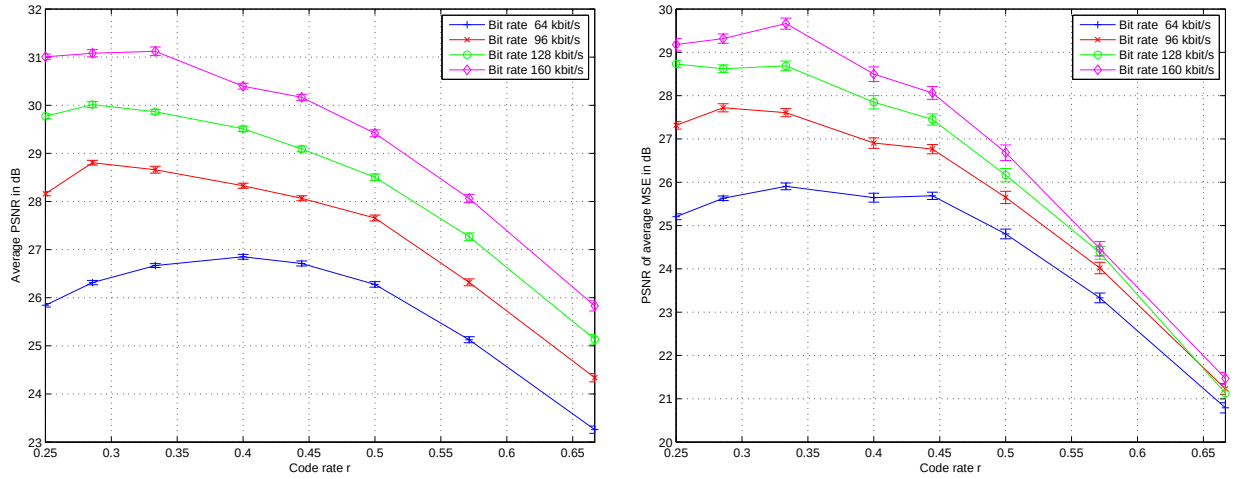
(a) $\overline{\text{PSNR}}$ for $\tilde{\varepsilon} = 0$ (b) $\text{PSNR}_{\overline{\text{mse}}}$ for $\tilde{\varepsilon} = 0$

Figure 8.4: PSNR over code rate r performance without any error resiliency tools for sequence *foreman*.

In Figure 8.4 the performance of this system for different code rates, different bit-rates, and different metrics is shown. In all cases also the error bars for confidence levels $\beta = 90\%$ are indicated within the results. From the figure it is obvious that for different modes, different bit-rates, and different performance metrics, there always exists an optimized channel coding rate r^*. The best code rates for all bit-rates are in the range between 0.25 and 0.33. These code rates are lower if the MSE is applied as performance criteria (see Figure 8.4b) than for the case when the PSNR is applied (see Figure 8.4a). Generally, with increasing bit-rate, the optimized code rate decreases. This is obvious as the additional bit-rate is preferably used for additional error protection to avoid error propagation than to improve the quality of the encoded video. It is also observed that in case the MSE being the criteria of interest, then increasing the bit-rate does only marginally improve the performance. The quality is determined by the distortion resulting from losses.

Performance Results with Channel-Optimized Mode selection

For the results in Figure 8.5, the system has been extended such that for each code rate, a channel–optimized mode selection adapted to the loss rate $\tilde{\varepsilon} = P_{\text{out}}(r)$ at the corresponding code rate has been applied. Compared to the results in Figure 8.4, for higher code rates the quality is always significantly higher. However, it is also observed that the best code rates r^* are still almost as low as for the case without error resilience. From a first comparison between the two graphs, it is further observed that the performance at the optimum rate is not significantly larger than without any error resilience. This indicates that the separation of source and channel coding may provide the good results even in case of delay-limited mobile video services. A detailed comparison of the results for the best code rates is provided in Figure 8.7.



(a) $\overline{\mathrm{PSNR}}$ for $\tilde{\varepsilon} = P_{\mathrm{out}}(r)$ (b) $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ for $\tilde{\varepsilon} = P_{\mathrm{out}}(r)$

Figure 8.5: PSNR over code rate r performance for channel-optimized mode selection adapted to the channel loss rate for sequence *foreman*.

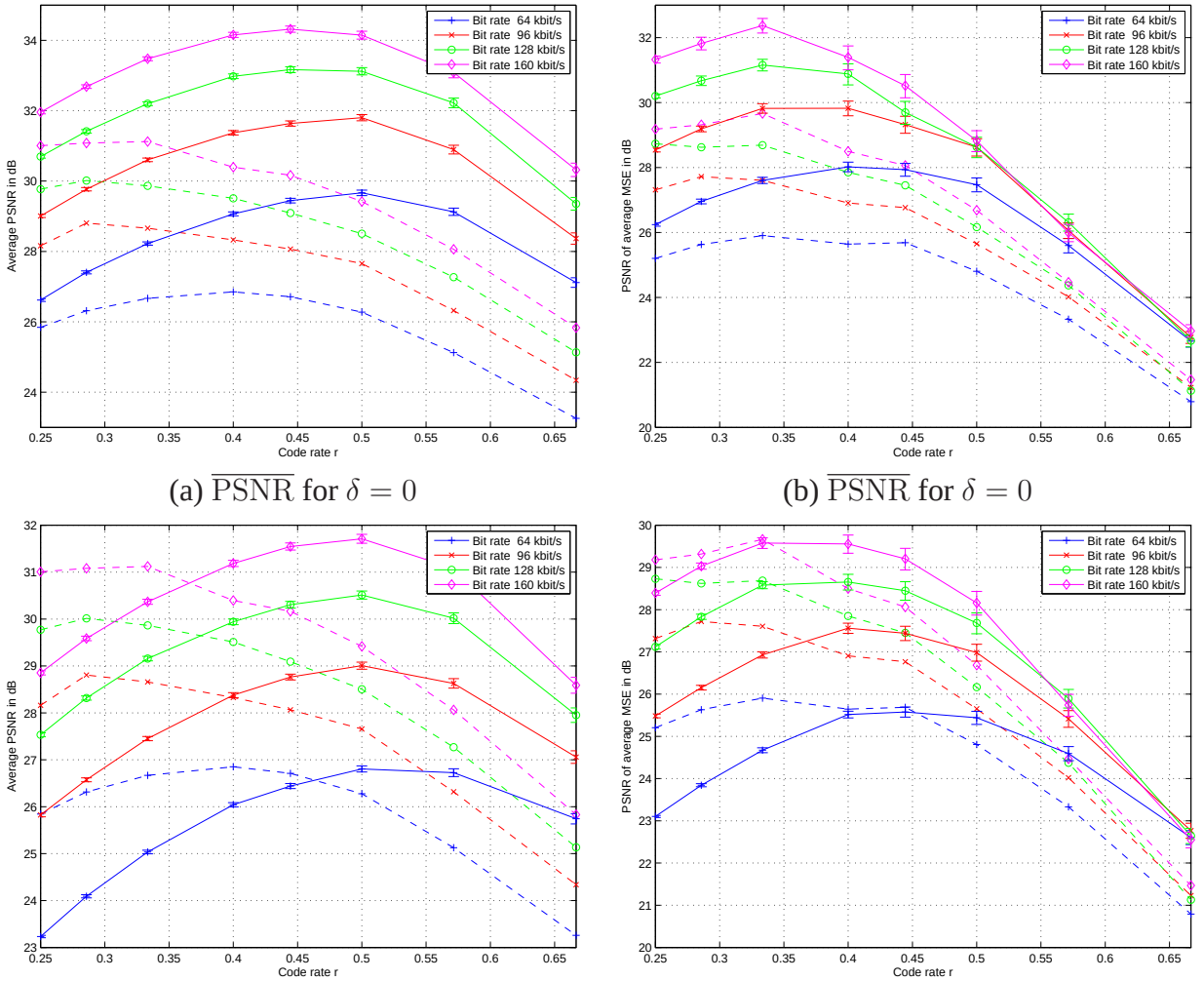
Performance Results with Interactive Error Control

For the results in Figure 8.6, the system has been extended by IEC. Two feedback delay modes have been investigated, one with instantaneous feedback and $\delta = 0$ in Figure 8.6 a) and b) and one with $\delta = 2$ in Figure 8.6 c) and d). The diagram also shows for comparison the corresponding performance for channel-optimized mode selection in dashed lines. In all cases for IEC, it is observed that the optimized code rate r^* is higher than for the case without feedback. This indicates that the due to the avoidance of error propagation, higher frame drop probabilities can be tolerated. Note also that the difference between using the PSNR or the MSE as selection criteria provides significantly different results in the code rate selection for IEC: Whereas for the PSNR criteria consequently a code rate of $r^* = 0.5$ should be chosen, for the case that the MSE being the measure of interest, the best code rates are around $r^* = 0.4$.

Comparison of Different Modes for Optimized Configuration

Figure 8.7 finally compares the best system configurations, i.e., using the optimized r^*, in terms of PSNR over the total bit-rate in kbit/s for the case without any error resilience, channel-optimized mode selection, and IEC with feedback delay $\delta = 0$ and $\delta = 2$. Figure 8.7 summarizes already reported results from Figures 8.4, 8.5 and 8.6. For the optimized channel coding rate it is observed, that only IEC with delay $\delta = 0$ differentiates in terms of performance from the other modes. Surprisingly, the mode without any error resilience performs basically as well as the channel-optimized mode selection. Therefore, despite the channel cannot provide arbitrarily low error rates for reasonable code rates, the separation of source and channel coding still provides very attractive results. Consequently, the appropriate rate allocation between video rates and channel code rates is more important than the provisioning of error resilience.

When compared to the IEC modes, for feedback delay $\delta = 2$, small gains compared to a mode without feedback can be achieved. However, if the delay of the feedback can be reduced or the feedback is basically instantaneous, then the IEC based mode can provide significant gains as can be seen for the case of $\delta = 0$. The gains in average PSNR are roughly $2 - 2.5$ dB and the bitrate may be almost reduced by a factor of two when compared to any of the other three modes. The



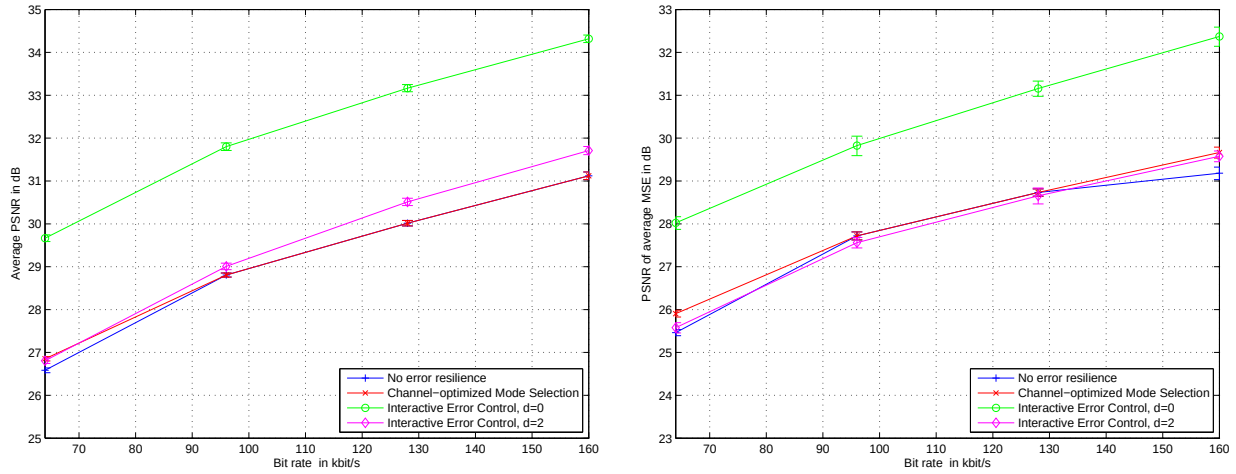
(a) $\overline{\mathrm{PSNR}}$ for $\delta = 0$

(b) $\overline{\mathrm{PSNR}}$ for $\delta = 0$

(c) $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ for $\delta = 2$

(d) $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ for $\delta = 2$

Figure 8.6: PSNR over code rate r performance for interactive error control (IEC), feedback mode 2, with different feedback delay δ for sequence *foreman* compared to channel-optimized mode selection (dashed lines).



(a) Average $\overline{\mathrm{PSNR}}$. (b) PSNR of MSE, $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$.

Figure 8.7: PSNR over bit rate performance for different modes with optimized configurations for sequence *foreman*.

results are almost identical, if the MSE is used as the measure of interest.

Therefore, in the remainder of this chapter as we focus on mobile conversational video, we assume the availability of a VFI channel. As the IEC-based transmission approaches provide the best performance all over, we will focus on those and integrate IEC in all further system designs. To understand the influence of the feedback delay, we apply two different delay modes, $\delta = 0$, i.e., the video encoder knows instantaneously, if and what packets had been received at the decoder for the previously delivered frame, and $\delta = 2$ for which the receiver information is available at the receiver within about $100 - 150$ ms.

8.2.3 Source-Adaptive Code Rate Selection

System Overview and Rate Allocation

The system and procedures introduced in the previous subsection applied a fixed channel code rate for the entire sequence. The optimized code rate was chosen *after* comparing the different results. This is in general practically no suitable procedure as the selection may be different depending on the sequence and the channel statistics. Secondly, this approach ignores the fact that the channel code rate may be even chosen adaptively for each video frame, depending on the local characteristics. Therefore, we proposed to apply an operational greedy rate allocation procedure as introduced in the following.

Furthermore, in this system we will introduce data partitioning as a further mean to react to varying channel conditions. Therefore, the system diagram in Figure 8.3 is still valid. The detailed procedure on the rate allocation for the data partitioning case is presented in the following, and the rate allocation for the single layer case can be viewed as subset of the one presented.

Recall that for each of the three partitions A, B and C, we have a partition size vector $\mathbf{k}(q)$ and assigned channel coding rates $\mathbf{r} = \{r_\mathrm{A}, r_\mathrm{B}, r_\mathrm{C}\}$. Furthermore, as already mentioned, we restrict ourselves to the IEC mode with $\delta = 0$ and $\delta = 2$. The partition size vector $\mathbf{k}(q)$ and the code rate vector $\mathbf{r}$ have to fulfill the rate constraint in (8.1) for each video frame. The quantization parameter q as well as the code rate vector $\mathbf{r}$ should be selected such that some quality criteria is maximized.

The measure of interest defined in the optimization process is the expected quality $\mathbb{E}\{\mathcal{Q}(q,\mathbf{r})\}$ that can be derived as a special case from (3.32) as

$$\begin{aligned}\mathbb{E}\{\mathcal{Q}(q,\mathbf{r})\} &= p_0(\mathbf{r})\mathcal{Q}_0 + p_A(\mathbf{r})\mathcal{Q}_{\text{A}}(q) + p_{\text{A,C}}(\mathbf{r})\mathcal{Q}_{\text{A,C}}(q) \\ &\quad + p_{\text{B,C}}(\mathbf{r})\mathcal{Q}_{\text{B,C}}(q) + p_{\text{A,B,C}}(\mathbf{r})\mathcal{Q}_{\text{A,B,C}}(q),\end{aligned} \tag{8.2}$$

where for both, the event probability $p_i(\mathbf{r})$ and the quality $\mathcal{Q}_i(q)$ the index i denotes the correctly decoded partitions. Note that non-reasonable combinations are excluded taking into account the dependencies of partitions. For example decoding partition B or C without partition A cannot increase the quality compared to $\mathcal{Q}_0$. The encoder is able to estimate the quality $\mathcal{Q}_i(q)$ by applying the appropriate error concealment. Note also that these quality terms depend on q except for $\mathcal{Q}_0$ referring to the quality in the case the entire frame is lost.

The probability that a certain event i occurs, depends on the applied channel coding vector $\mathbf{r}$ for different partitions and the resulting outage probabilities $P_{\text{out}}(r_m)$. Due to the dependency of partitions B and C on A, it is obvious that the only reasonable channel code rate vectors $\mathbf{r}$, must fulfill $r_{\text{A}} \leq r_{\text{B}}$ and $r_{\text{A}} \leq r_{\text{C}}$. Although we have supposed that in general partition B is more important than partition C, it is not obvious that this is always the case, since partition B and C can be decoded independently. Therefore, we consider two cases, namely $r_{\text{B}} \leq r_{\text{C}}$ and $r_{\text{B}} > r_{\text{C}}$. Due to the interleaving and the access to the same channel realization for all channel encoded partitions it is obvious that if a partition with channel coding rate r_m cannot be decoded, any partition with $r_{m'} \geq r_m$ cannot be decoded either. With these preliminaries, the event probabilities result in

	$r_{\text{B}} \leq r_{\text{C}}$	$r_{\text{B}} > r_{\text{C}}$
$p_0(\mathbf{r})$	$P_{\text{out}}(r_{\text{A}})$	$P_{\text{out}}(r_{\text{A}})$
$p_{\text{A}}(\mathbf{r})$	$P_{\text{out}}(r_{\text{B}}) - P_{\text{out}}(r_{\text{A}})$	$P_{\text{out}}(r_{\text{C}}) - P_{\text{out}}(r_{\text{A}})$
$p_{\text{A,B}}(\mathbf{r})$	0	$P_{\text{out}}(r_{\text{B}}) - P_{\text{out}}(r_{\text{C}})$
$p_{\text{A,C}}(\mathbf{r})$	$P_{\text{out}}(r_{\text{C}}) - P_{\text{out}}(r_{\text{B}})$	0
$p_{\text{A,B,C}}(\mathbf{r})$	$1 - P_{\text{out}}(r_{\text{C}})$	$1 - P_{\text{out}}(r_{\text{B}})$.

(8.3)

Note that by making use of (8.3), the equation in (8.2) can be transferred into an additive form according to (3.31) by defining an importance $\mathcal{I}_m$ for each partition and multiplying this importance with $P_{\text{out}}(r_m)$ with $m \in \{\text{A},\text{B},\text{C}\}$.

The encoder then selects the QP q^* from the set of available quantizers $\mathcal{S}_q$ and the channel coding rates $\mathbf{r}^*$ such that the expected quality is maximized, i.e.,

$$\{q^*, \mathbf{r}^*\} = \arg \max_{\{q \in \mathcal{S}_q,\, \mathbf{r} \in \mathcal{S}_r^3\}} \mathbb{E}\{\mathcal{Q}(q,\mathbf{r})\}, \tag{8.4}$$

subject to the rate constraint in (8.1).

Although the search may be reduced to a linear or quadratic complexity order, a brute force search strategy is easily feasible since the number of layers and options is low. Basically, for each $q \in \mathcal{S}_q$, the optimal combination of channel coding rate $\mathbf{r}^*(q)$ is sought, excluding impossible combinations. In addition, it is assumed that $\mathbb{E}\{\mathcal{Q}(q,\mathbf{r}^*(q))\}$ has only one global maximum over q and that for given r_{A} and r_{B} the smallest $r_{\text{C}} \in \mathcal{R}$ is used which fulfills the rate constraint in (8.1).

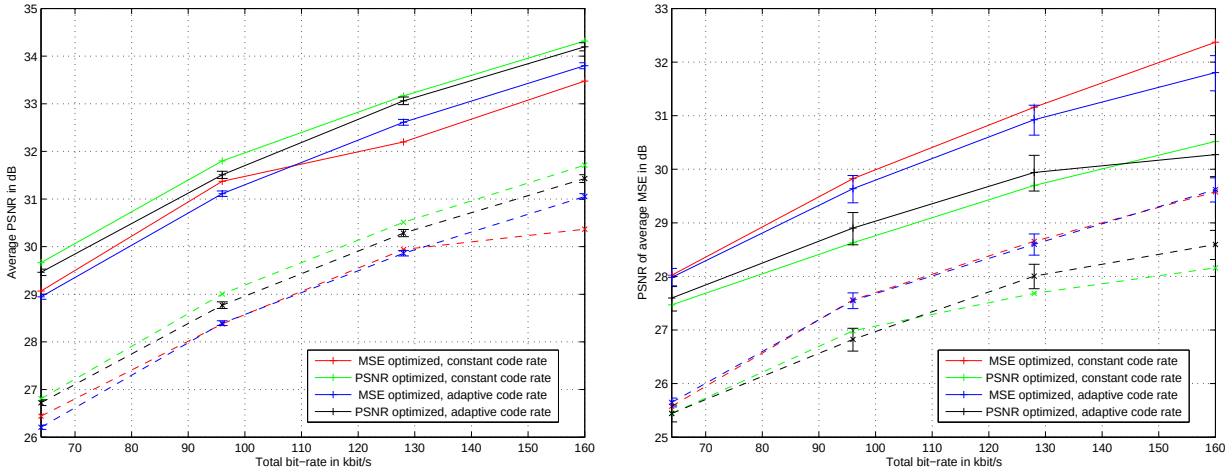
Feedback and Adaptive Coderate

In a first system design making use of the introduced rate allocation concept, we restrict ourselves to a single layer system using H.264/AVC baseline and not employing data partitioning. In this

case (8.4) reduces to

$$\begin{aligned}\{q^*, r^*\} &= \arg \max_{\{q\in\mathcal{S}_q, r\in\mathcal{S}_r\}} P_{\text{out}}(r)\mathcal{Q}_0 + (1 - P_{\text{out}}(r))\mathcal{Q}_{\text{FF}}(q) \\ &= \arg \min_{\{q\in\mathcal{S}_q, r\in\mathcal{S}_r\}} P_{\text{out}}(r))\mathcal{I}_{\text{FF}}(q), \end{aligned} \tag{8.5}$$

subject to the simplified rate constraint in $k_{\text{FF}}(q)/r \leq N_{\max}$, where r is the applied code rate for the EEP, $\mathcal{Q}_{\text{FF}}(q) \triangleq \mathcal{Q}_0 + \mathcal{I}_{\text{FF}}(q)$ is the quality for obtaining the full frame quantized with quantization parameter q, $\mathcal{I}_{\text{FF}}(q)$ the corresponding importance and $k_{\text{FF}}(q)$ is the corresponding number of bits. As a quality metric, typically either the PSNR or the MSE may be applied, generally resulting in different rate allocation as will be seen below.



(a) Average $\overline{\text{PSNR}}$ (b) PSNR of average MSE, $\text{PSNR}_{\overline{\text{mse}}}$

Figure 8.8: PSNR performance over the total bit-rate for feedback delay $\delta = 0$ (solid lines) feedback delay $\delta = 2$ (dashed lines).

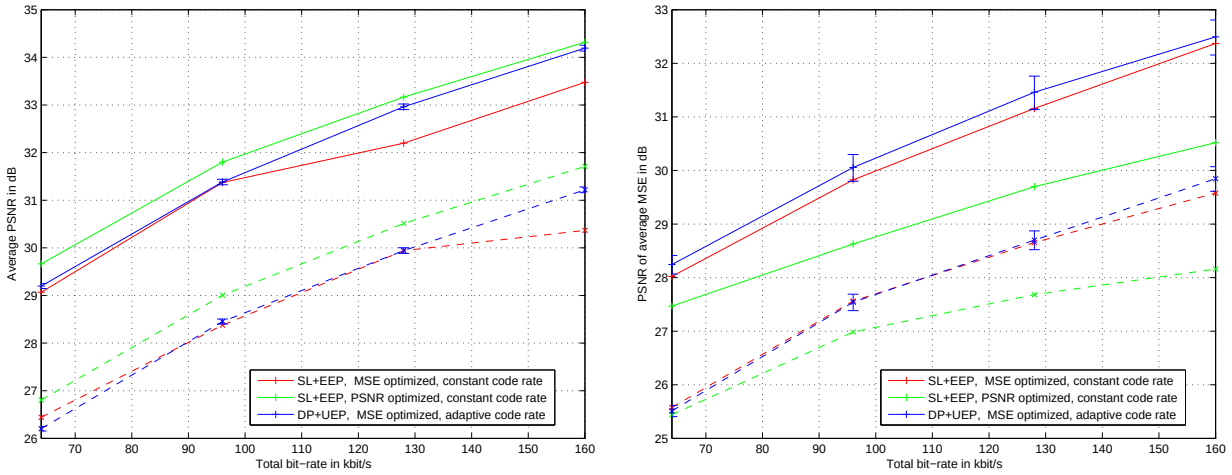
In Figure 8.8 the PSNR performance over the total bit-rate is shown: Figure 8.8a) shows the performance for the average $\overline{\text{PSNR}}$ and Figure 8.8b) the performance for the PSNR of the average MSE, $\text{PSNR}_{\overline{\text{mse}}}$. In both figures, four curves for feedback delay $\delta = 0$ (solid lines) and four curves for feedback delay $\delta = 2$ (dashed lines) are shown. The green and the red curves are duplicated from Figure 8.7 showing the globally best constant code rate when optimizing for maximum PSNR (green curve) and minimizing MSE (red curve). The black and blue curves show the performance for the optimization as presented in this subsection, one for minimizing the MSE, one for maximizing the PSNR. The results are consistent for the different delays, so we focus on $\delta = 0$ in the further discussion. The performance of the global optimization after simulation and the local optimization is almost equally good for all cases but the global post-simulation optimizations slightly outperforms the adaptive optimization in here. This is initially surprising, but bear in mind that the global optimization is generally not operational, so the local optimization is a good alternative to achieve the the same performance as possible with post-optimization.

Another reason that leads to the weaker performance of the adaptive selections as presented results from the fact that the optimization in (8.5) does not take into account the consequences of a quantization parameter selection for the MCP process: The quality of the prediction signal influences the coding performance of the subsequent frames. Some more analysis on this aspect will be provided below in comparison to data partitioning. A final remark on this figure deals with the confidence intervals regarding these simulations. Despite a significant amount of frames have

been simulated, the confidence intervals for the $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$ metric in Figure 8.8b) are rather large. This results from the fact, that for the simulation of this scenario, basically two random variables contribute to the statistics and influence the rate allocation, namely the source statistics and the channel realizations.

Data Partitioning and UEP

In the system design discussed in the following, we further enhance the previous system by integrating data partitioning. Due to the potentially different importance of the different data partitions the optimization in (8.4) may result in an UEP assignment of the channel coding rates. In this case it is essential to apply this local optimization since a global assignment of channel coding rates to individual partitions is impossible. This is due to the fact that the sizes of the data partition change for each frame, and therefore a global rate constraint cannot be maintained without significantly sacrificing bit-rate.



(a) Average $\overline{\mathrm{PSNR}}$ (b) PSNR of average MSE, $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$

Figure 8.9: PSNR performance over the total bit-rate for data partitioning with UEP compared to single layer modes for feedback delay $\delta = 0$ (solid lines) feedback delay $\delta = 2$ (dashed lines).

In Figure 8.9 the PSNR performance over the total bit-rate is shown: Figure a) shows the performance for the average $\overline{\mathrm{PSNR}}$ and Figure b) the performance for the PSNR of the average MSE, $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$. In both figures, three curves for feedback delay $\delta = 0$ (solid lines) and three curves for feedback delay $\delta = 2$ (dashed lines) are shown. The green and the red curves are copied from Figure 8.7 showing the globally best constant code rate when optimizing for maximum PSNR (green curve) and minimizing MSE (red curve). The blue curves show the performance for the optimization as presented in this subsection, one for minimizing the MSE. Note that we do not show the results for the case of PSNR optimized data partitioning for a very simple reason: In this case the difference between the importances of individual partitions is so insignificant that it always results in an EEP assignment. Furthermore, as the compression efficiency of data partitioning is slightly inferior than for single layer coding, the performance is always in favor of single layer coding.

However, for MSE optimized encoding, there is a clear benefit when using data partitioning: Especially the MSE performance in Figure 8.8b) shows that data partitioning can improve the overall performance compared to single layer coding. Data partitioning with UEP, optimized rate allocation and IEC can provide overall very good performance and is a candidate system for low-delay mobile video communication services.

More Insight – Code Rate and Quantization Parameter Selection

In the following we will provide some more insight on the results as provided in the previous two subsections, especially focussing on the selection of the optimized parameters $\{q^*, \mathbf{r}^*\}$ for the different system designs.

In Figure 8.10 we show the quantization parameter over the total bit-rate for feedback delay $\delta = 0$ (solid lines) and feedback delay $\delta = 2$ (dashed lines). Note that a lower quantization parameter[1] results in higher quality of the encoded signal. The quantization parameter for delay $\delta = 2$ is by about 4 quantization parameter steps higher than for the case of $\delta = 0$. This is due to the lower correlation between the predicted frame and the predication frames which is also the main reason for the lower efficiency when the feedback delay increases. However, the tendencies are similar for both delays, and therefore we focus on $\delta = 0$ (solid lines) in the remainder. The figure reveals that PSNR optimization always selects lower QPs as the encoding quality is more important for this metric. More insight on this follows in Figure 8.11. Furthermore, the adaptive code rate selection always results in a slightly higher QP than the constant CR with post-optimization. This indicates that the optimization in (8.4) tends to select the QP q^* higher, possibly ignoring that a better reference signal may provide overall better quality as indicated by the slightly better performance of the constant code rate cases.

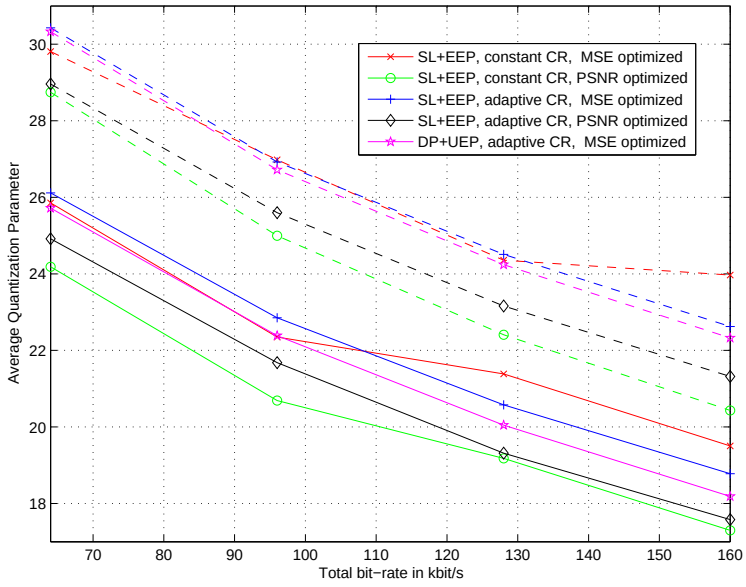
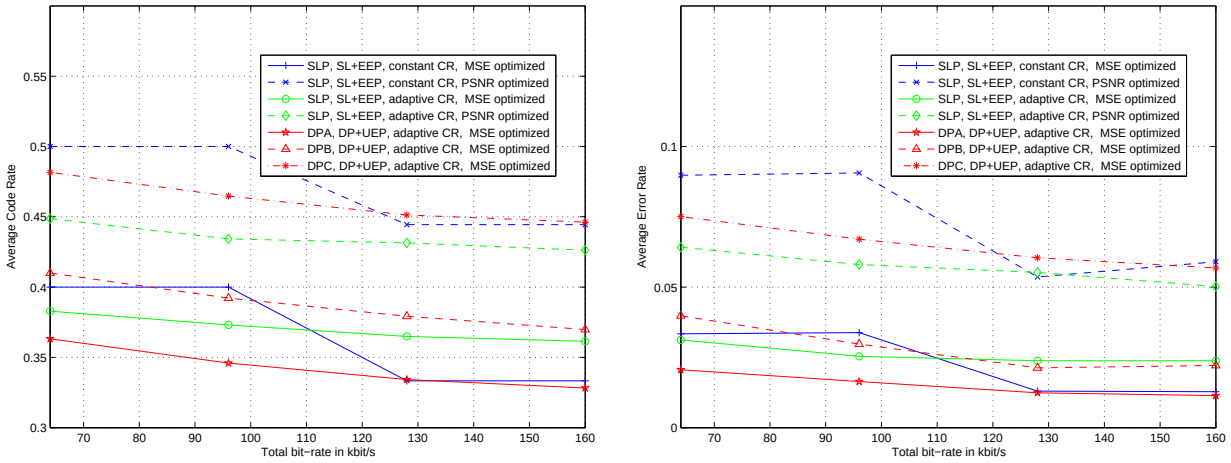


Figure 8.10: Average quantization parameter over the total bit-rate for feedback delay $\delta = 0$ (solid lines) feedback delay $\delta = 2$ (dashed lines); for sequence *foreman* and different system configuratons.

In Figure 8.11a) the code rate allocation over the total bit-rate for feedback delay $\delta = 0$ is shown. For constant non-adaptive coding, referred to as constant code rate, the globally best code rate is shown. For the adaptive code rate assignment the average code rate is presented. For the data partitioning case, the average code rate of all three partitions is shown. The figure is in line with the previous observations that lower code rates are assigned if MSE is applied, and if the

[1]Test model JM1.7 was used for the purpose of these simulations. In this interim model, the quantization parameter is by 12 lower than in the final standard. For comparing the reported QP with the one in the final standard, please add 12.

adaptive code rate is applied. Furthermore, it is observed that the allocation results in a clear UEP assignment for data partitioning. It is also obvious that partition B is more important than partition C, the assigned code rate is always lower. One main reason for this is that intra modes are mainly selected during encoding in case it is difficult to predict the frame based on the frames in the reference buffers. This also means that a temporal concealment will not work well and the importance is in general higher for B partitions and they are protected stronger. The consequences of the rate allocation can be observed in Figure 8.11b) where the resulting error rates are shown. Typically, the rate allocation asks for loss rates of $1-3\%$ for the case that MSE is applied as criteria, and in case of data partitioning, if it is partition A or B. For the PSNR criteria, error rates between $6-9\%$ result in the best performance. The partition C in case of data partitioning may be lost with higher probability between $6-7\%$, even if the MSE criteria is applied. In summary it is observed that data partitioning in the proposed environment is quite powerful, but it requires that the rate allocation is done appropriately and adaptively such that the different importance of each partition is reflected.



(a) Average optimized channel coding rates (b) Average error rates

Figure 8.11: a) Average optimized channel coding rates and b) average error rates over the total bit-rate for feedback delay $\delta = 0$, both for sequence *foreman* and different system configuratons.

8.3 PTVC-based Source-Channel Coding System

8.3.1 System Overview

Single-layer coding or data partitioning does not permit the adaptation to the actual transmission conditions. Once a frame has been encoded, adaptation to the transmission conditions is almost impossible. Therefore, we look for video coding schemes that allow rate adaptation even after some early parts of the video frame have already been transmitted. The advantage of using the PTVC allows an easy and accurate adaptation of the video quality to the actual experienced channel conditions by using the progressive texture part of the PTVC for instantaneous rate adaptation. Some initial ideas and results have already been introduced in [SJW02b], an extension of this work is presented in the following. We will combine the error protection schemes presented in chapter 4, the PTVC presented in section 3.4.2 and different feedback strategies. The principles

of the framework are applicable to any kind of real-time video transmission. However, for the remainder of this section we will exclusively focus on conversational video applications.

Figure 8.12 shows the general PTVC-based system diagram which will be refined in the sequel of this section when we introduce different error protection schemes in more detail. All the service requirements and channel conditions as introduced in section 8.1 hold also for the PTVC-based system and therefore also allow comparison with the systems introduced in section 8.2. We assume that the long-term channel statistics p_α are perfectly known at the transmitter and that we have access to an instantaneous and error-free low bit-rate channel in both directions. This channel is used to transport feedback information (ACK, NAK, or decoder reference layer $\tilde{\check{m}}$) in the backward direction and channel code rate allocations in the forward direction. In a practical video communication system the even more delay-sensitive audio channel might be used to transport these messages. In the remainder we assume that the feedback is instantaneous and all IEC modes operate with $\delta = 0$.

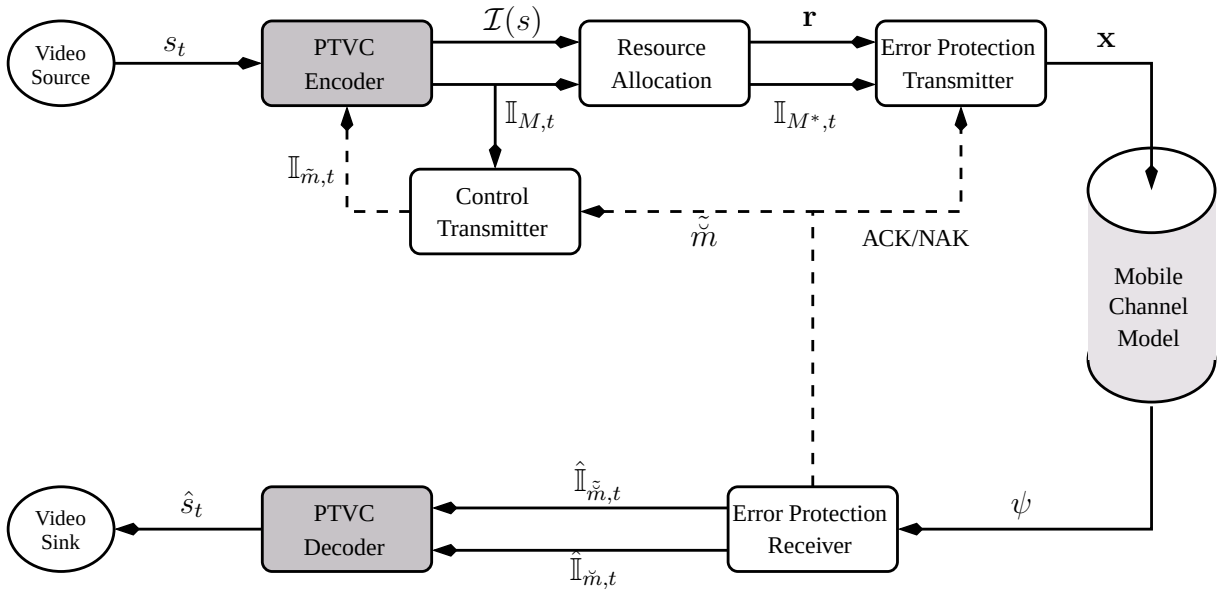


Figure 8.12: General Wireless Video Transmission System.

Following the PTVC video encoding process as introduced in section 3.4.2, for each frame the prediction mode is chosen by the transmitter control indicating whether this frame is predicted from the images in the reference frame buffer (P-frame) or purely intra coded (I-frame). In our case we may introduce a regular intra-frame update every P_I-th frame. I-frames get assigned R_{IP} times more data rate than P-frames. However, for consistent low delay and as we make use of the feedback channel, we set $R_{IP} = 1$, i.e., the I-frame has the same size as the P-frame. I-frames are used only once in the beginning of the sequence or as a result of the feedback reporting and video encoder rate control decisions. For each video frame s, the encoding process generates an embedded bit-stream $\mathbb{I}_M$ and layered importance vector[2] $\mathcal{I}(s)$. These parameters are forwarded to a rate allocation process which selects an appropriate number of layers M^* to be transmitted and the corresponding channel code rate vector $\mathbf{r}$. Optimized rate allocation algorithms for different error protection schemes are discussed.

The transmitter outputs an appropriate transmission signal vector $\mathbf{x}$ and at the receiver we obtain

[2]In the remainder we assume that the importance vector includes the base quality $\mathcal{Q}_0$.

the LLR of the received signal ψ. The LLR is passed to the receiver of the error protection system. In general, the error protection decoder generates two versions of the decoded index $\hat{\mathbb{I}}$. $\hat{\mathbb{I}}_{\breve{m}}$ is used to reproduce the source data $\hat{s}$ whereas $\hat{\mathbb{I}}_{\tilde{\breve{m}}}$ is used to produce a new frame for the reference buffer. In addition, we will investigate several optional feedback strategies. The amount of reliably decoded bits $\tilde{\breve{m}}$ is conveyed from the receiver to the transmitter such that an appropriate reference frame at the encoder can be generated using the index $\mathbb{I}_{\tilde{\breve{m}}}$. This system in combination with a FEC-based error protection using UEP and FEED is presented in subsection 8.3.2. Additionally, the error protection scheme itself might exploit the feedback in an ARQ scheme. Systems with different ARQ strategies are presented in subsection 8.3.3. Finally, in subsection 8.3.4, we introduce error protection schemes, that rely on the knowledge of the exact channel state at the transmitter and therefore, can adapt the code rate, and possibly even the transmit power for each transmitted video frame.

In order to perform appropriate rate allocation at the transmitter and to estimate the performance of the systems, we are interested in a quality estimation procedure for each of the systems. Therefore, let $\breve{M} \triangleq \min\{m : \hat{\mathbb{I}}_m \neq \mathbb{I}_m\} - 1$ be a random variable specifying the last layer for which the source coder index $\mathbb{I}_m$ was correctly decoded after transmission. We assume the decoder knows the value of $\breve{M}$ by perfect error detection and that the probability of undetected errors is zero. Then, $\mathcal{Q}^{(\breve{M})}(s)$ is used as an estimate of s. Recalling the definitions in (3.30) and (3.33), the expected quality for a progressively encoded video frame s can be given as

$$\mathbb{E}\left\{\mathcal{Q}_{\breve{M}}(s)\right\} = \sum_{m=0}^{M} \mathcal{Q}_m(s) \cdot \Pr\{\breve{M} = m\} \tag{8.6}$$

$$= \mathcal{Q}_0(s) + \sum_{m=1}^{M} \mathcal{I}_m(s) \cdot \Pr\{\breve{M} \geq m\}, \tag{8.7}$$

where $\Pr\{\breve{M} = m\}$ denotes the probability that exactly layer m is decoded and $\Pr\{\breve{M} \geq m\}$ denotes the probability that at least layer m is decoded. Note that the expectation is over the channel statistics. The error probability $\Pr\{\breve{M} = m\}$ depends on the channel statistics and the error protection scheme. We will provide appropriate expressions for an UEP scheme in section 8.3.2, several ARQ schemes in 8.3.3, and schemes with CSI in subsection 8.3.4.

8.3.2 Forward Error Correction

System Overview

In a first system design we integrate the system as introduced in section 5.4 for SPIHT in the video transmission system shown in Figure 8.12. The basic idea is that by the use of the PTVC together with the regressive UEP and the FEED algorithm, the transmission adapts automatically to the experienced transmission conditions over a short amount of time. The refined system is shown in Figure 8.13: After the generation of a video frame s and PTVC encoding we obtain the index $\mathbb{I}_M$ and the importance vector $\mathcal{I}(s)$, which is passed to the rate allocation process. The rate allocation process generates an optimized channel coding vector $\mathbf{n}^*$ of length M^* and the shortened information message $\mathbb{I}_{M^*}$ of length K_{M^*}. For rate allocation, we reuse exactly the algorithm as introduced in subsection 5.4.2. For an assigned channel symbol vector, $\mathbf{n}_m = \{n_1, \ldots, n_m\}$, the probability $\Pr\{\breve{M} \geq m\}$ that decoded layer $\breve{M}$ is at least m corresponds to $p_d(n_m)$ according to (5.37). Specifically we make use of *source–channel–optimized rate allocation* such that based on

the importance vector $\mathcal{I}(s)$ and the channel statistics the rate allocation provides both, rate control and channel code rate assignment.

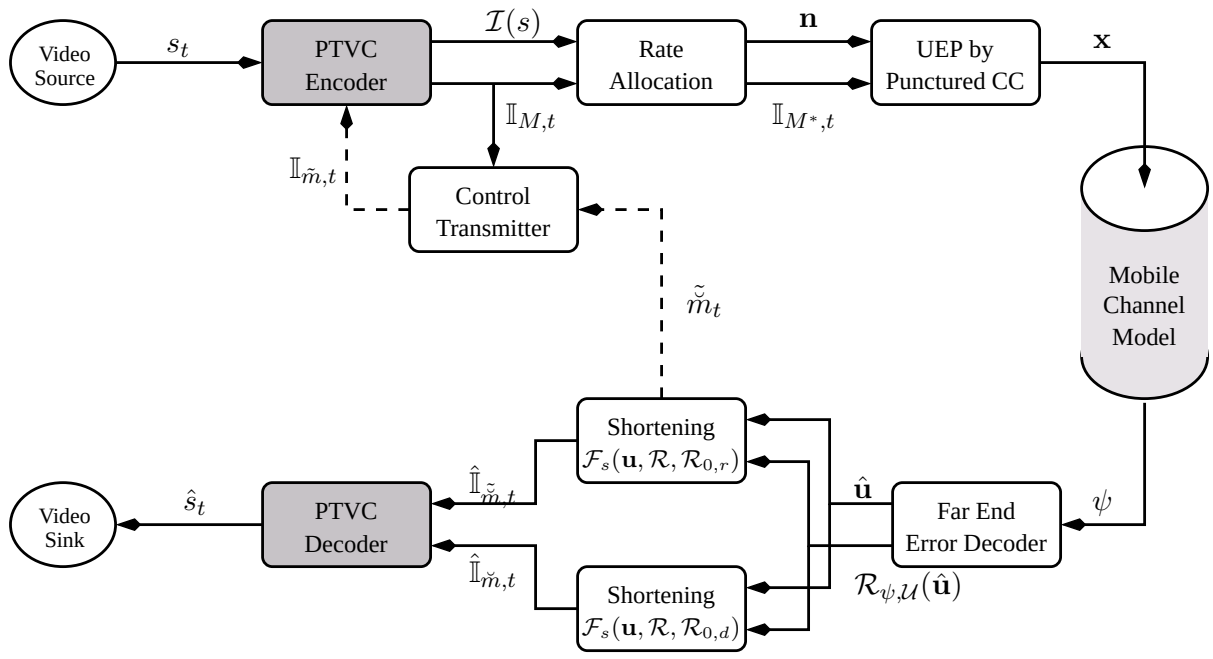


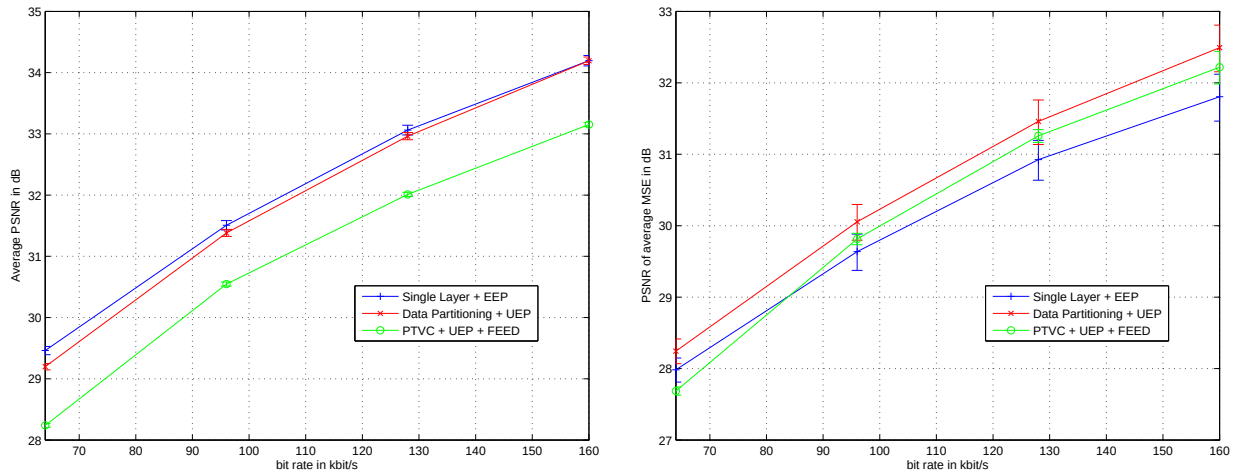
Figure 8.13: Wireless transmission system with unequal error protection.

The generated bit index is channel encoded, appropriate puncturing and interleaving over F slots is applied. The resulting transmission signal vector $\mathbf{x}$ is then transmitted over F radio slots and at the receiver after appropriate signal processing we obtain the LLR ψ of the received signal. The FEED algorithm processes ψ to obtain a decoded binary sequence $\hat{\mathbf{u}}$ and the corresponding reliability vector $\mathcal{R}_{\psi,\mathcal{U}}(\hat{\mathbf{u}})$. The information sequence is shortened with the decoding reliability threshold $\mathcal{R}_{0,d}$ to obtain a received index $\hat{\mathbb{I}}_{\check{m}}$. This index is used to decode and represent the frame $\hat{s}$ at the receiver. Additionally, the information sequence is also shortened with a reference reliability threshold $\mathcal{R}_{0,r}$ to obtain a received reference index $\hat{\mathbb{I}}_{\tilde{\check{m}}}$ which is used to obtain a reference frame at the receiver. The decoded reference layer $\tilde{\check{m}}$ is conveyed to the transmitter through a VFI channel. This information is then used to produce a reference frame in the PTVC encoder by setting $\tilde{m} = \tilde{\check{m}}$. The reference frame at the transmitter side is then produced by decoding $\mathbb{I}_{\tilde{m}}$. In general we use $\mathcal{R}_{0,r} \geq \mathcal{R}_{0,d}$ as the reference frames should be more reliable than the presented frames to avoid error propagation and drift problems. The UEP is implemented by applying rate-compatible puncturing as presented in subsection 4.3.

Experimental Results

To verify and evaluate the proposed system the H.264/AVC-based simulation environment as introduced in section 8.2 has been extended. The common simulation parameters have been summarized in Table 8.1. All simulations are carried out using the QCIF test sequence *foreman* at a constant frame rate of $f_s = 10$fps. To fulfill the minimum delay requirement and to have constant frame rate I-frames get assigned the same number of bits, i.e. $R_{IP} = 1$. We again use the systematic convolutional code with memory $\mu = 96$, mother code rate $1/c = 1/7$ and puncturing period $\omega_p = 32$. The computational limit for the sequential decoder is set to $C_{\text{lim}} = 16$. In the following

experiments the rate allocation for the PTVC and UEP scheme targets to minimize the MSE for all cases.

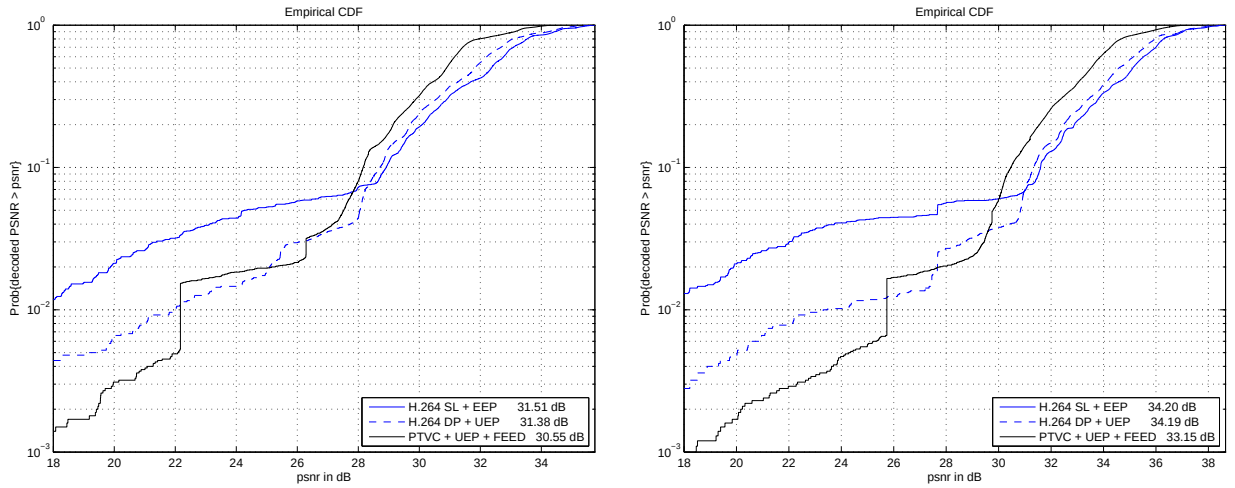


(a) $\overline{\mathrm{PSNR}}$ (b) $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$

Figure 8.14: PSNR over total bit-rate performance for UEP and FEED for sequence *foreman* compared to H.264/AVC schemes.

In Figure 8.14 the PSNR over total bit-rate performance for UEP and FEED for sequence *foreman* is shown (green curve) compared to H.264/AVC based schemes as introduced in section 8.2. The reference systems are all based on operationally implementable adaptive source rate selection according to subsection 8.2.3. The blue curve shows the single layer H.264/AVC with EEP, and the red curve the data partitioning scheme with UEP. Figure 8.14a) shows the performance in terms of average PSNR: In this case the H.264/AVC based systems outperform the proposed system significantly by about 1dB, or about $10-15\%$ in terms of bit-rate. This is due to two main reasons: First, the proposed system is optimized towards minimum MSE distortion, and secondly the compression efficiency of the PTVC is lower. In Figure 8.14b) the performance in terms of PSNR of the average MSE is shown: In this case, the proposed system outperforms the single layer performance with EEP and performs almost as good as data partitioning with UEP.

To provide some more insight into the results, and also some justification for the proposed PTVC and UEP system, we show the distribution of the PSNR of each individual frame in Figure 8.15 with different total bit-rate of 96 kbit/s in a) and 160 kbit/s in b). In both cases similiar observations can be made, so we focus on Figure b): it is observed that the probability of low image quality below 25dB is significantly lower for the proposed system then for the two H.264/AVC-based systems. This also shows that the loss of entire frames is prevented by the use of the PTVC and UEP. However, due to the lower coding efficiency of PTVC compared to H.264/AVC for error-free transmission (high PSNR values above 30dB are less likely), the overall performance is lower. Therefore, the average PSNR as well as the PSNR of the MSE as shown in Figure 8.14 may still hide some important aspects. Hence, if more consistent video quality is of higher importance than high average PSNR, the proposed PTVC and UEP with FEED at the receiver is an attractive alternative, especially compared to the single layer H.264/AVC with EEP system.



(a) Total bit-rate 96 kbit/s (b) Total bit-rate 160 kbit/s

Figure 8.15: Distribution of PSNR for UEP and FEED, different bit-rates and sequence *foreman* compared to H.264/AVC systems; legend shows also average PSNR.

8.3.3 Non-Persistent ARQ Schemes

System Overview

For the adaptation to varying transmission conditions, the use of UEP is mainly reasonable if no feedback channel for the error protection scheme is available. However, in conversational applications, fast feedback channels are very common, as for example audio needs to be transmitted in both directions. In this case, retransmission protocols are very suitable to adapt to varying transmission conditions. Retransmission protocols based on Automatic Repeat reQuest (ARQ) have been introduced in subsection 4.1.5. For low-delay applications, non-persistent ARQ schemes need to be applied, and in addition, as we transmit over bit-rate limited channels, the application of ARQ results in a variable bit-rate channel. Therefore, applications that can react to the variable bit-rate, are highly preferable in resource and delay-limited ARQ schemes – the combination with the PTVC provides a promising system concept.

As discussed in subsection 4.1.5, ARQ schemes are generally combined with FEC. Different schemes, referred to as ARQI, ARQI+, and ARQII, result in different performance, but may have impact on the complexity. Hence, we present in the following systems with different protocols and show the performance in combination with delay-limited mobile conversational video employing the PTVC. The basic idea is as follows: The PTVC encoder generates an embedded bitstream for each video frame. Then this embedded video frame is transmitted in pieces starting from the beginning using an ARQ protocol. If no retransmission is necessary, the second slot can transmit the new data from the embedded bitstream, otherwise the retransmission is carried out. The amount of transmitted data for this specific frame determines the quality at the receiver as well as the prediction signal in the PTVC encoder for the next frame. A more detailed description is the system follows.

Figure 8.16 shows the proposed system: By encoding of s the PTVC encoder generates a binary index $\mathbb{I}_M$ and the corresponding layered importance vector $\mathcal{I}(s)$, which is stored at the transmitter side. Based on this importance vector and the available number of transmission slots F to transmit this message the rate allocation process derives an optimized channel coding rate $r^*_{m=1}$ for the first transmission attempt in slot $f = 1$. The optimization process is discussed later in this subsection.

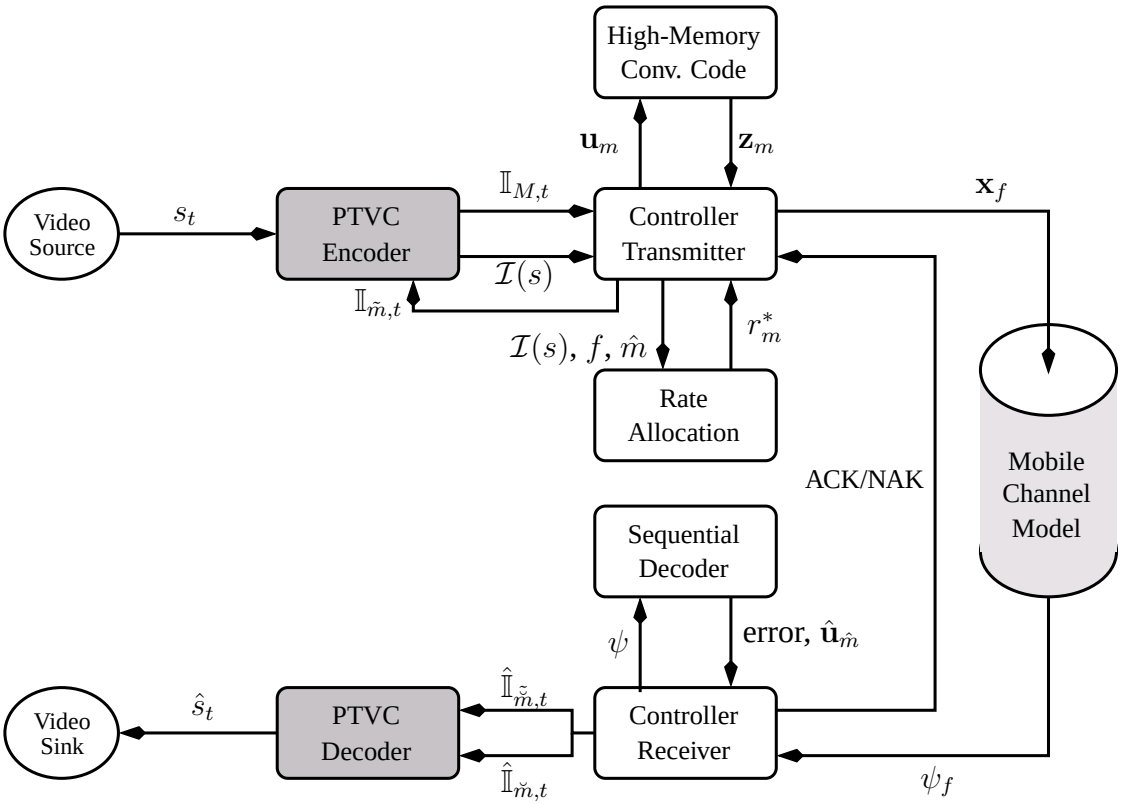


Figure 8.16: Wireless transmission system with automatic repeat request.

The transmitter control selects $k_{m=1} = N_v r_{m=1}$ information bits from the progressively coded index $\mathbb{I}_{M,t}$ to generate the information message $\mathbf{u}_{m=1}$. This message is encoded by the channel code – in our case a high-memory punctured convolutional code – to obtain the code symbol vector $\mathbf{z}_{f=1}$. By appropriate puncturing and mapping the transmitter generates the transmission message $\mathbf{x}_{f=1}$ of length N_v. At the receiver the LLR $\psi_{f=1}$ is generated and by inverse puncturing we obtain the LLR ψ which is passed to the decoder. The channel decoder – in our case a sequential decoder – either decodes the message $\hat{\mathbf{u}}_m$ or indicates a decoding failure. Note that in a practical system we cannot necessarily assume that no undetected errors occur, but the probability is very low and in general it can be assumed that $\hat{\mathbf{u}} = \mathbf{u}$.

In case of a detected error a NAK is sent to the transmitter and for ARQI+ and ARQII protocols $\psi_{f=1}$ is stored at the receiver. At the transmitter if we receive a NAK and we still can transmit some data within the delay budget, i.e., $f \leq F$, we transmit the additional redundancy of the first message according to the specified ARQ protocol, i.e., repetition of $\mathbf{x}_{f=1}$ in the case of ARQI and ARQI+ or transmission of the second subblock in the case of ARQII. At the receiver the LLR $\psi_{f=2}$ is processed according to the employed ARQ protocol to obtain a new ψ which is passed to the channel decoder. This procedure is repeated until decoding is successful and is stopped, if the delay constraint has expired for this source message, i.e., $f > F$. In the case of successful decoding an ACK is sent to the transmitter and the decoded information $\hat{\mathbf{u}}_{m=1}$ is stored at the receiver. If the transmitter receives an ACK, the transmission of the current information part $\mathbf{u}_{m=1}$ is stopped.

Then, a new rate allocation is performed based on the number of residual transmission slots $\tilde{f} = F - f + 1$ and $\mathcal{I}(s)$ to obtain the optimized code rate $r^*_{m=2}$, or, in general r^*_m. The next

$k_m = N_v r_m^*$ information bits are taken from the binary index $\mathbb{I}_M$ to obtain the information part $\mathbf{u}_m$ which is encoded, punctured, transmitted and processed at the receiver as already discussed.

If the receiver has processed the received LLR of the final radio slot $f = F$, no more information of the source data s_t is expected. Assume that the number of successfully transmitted information parts is denoted as $\breve{m}$. Therefore, at the receiver the decoded index $\hat{\mathbb{I}}_{\breve{m}}$ is obtained by the concatenation of the received information parts $\{\hat{\mathbf{u}}_{m=1}, \ldots, \hat{\mathbf{u}}_{\breve{m}}\}$. $\hat{\mathbb{I}}_{\breve{m}}$ is decoded to obtain $\hat{s}_t$. The same index is used to reconstruct the reference frame at the decoder, i.e., $\hat{\mathbb{I}}_{\tilde{\breve{m}}} = \hat{\mathbb{I}}_{\breve{m}}$. At the transmitter, if no more slots are available the transmission for the current source message is stopped. Additionally, an appropriate reference frame has to be generated. As the transmitter is aware of the number of layers $\breve{m}$ decoded at the receiver by the ACK/NAK feedback channel, the reference frame can be generated by setting $\tilde{m} = \breve{m}$. The transmitter generates a new reference frame by decoding $\mathbb{I}_{\tilde{m}}$. Then, the new frame s_{t+1} can be encoded and the processing and transmission of the subsequent binary index $\mathbb{I}_{M,t+1}$ starts.

This transmission and scheduling strategy ensures that only if the first bits of the encoded frame containing the most important data like motion vectors are received, the residual bits are attempted to be transmitted. The important data therefore has F radio slots to be conveyed to the receiver, whereas the less important information can only use the remaining slots $\tilde{f} < F$.

Rate Allocation: Selection of Channel Coding Rates

Remaining in the system description is the selection of the appropriate source rate and channel coding rate for each new transmission within the ARQ schemes. For a reasonable selection, we apply again a quality optimization. Therefore, we are interested in the expected quality for a given channel code rate vector $\{r_1, \ldots, r_F\}$ assigned in each transmission slot f. This serves to estimate the performance as well as to obtain the optimized channel coding rates by maximizing the expected quality.

According to (8.6), we require the probability $\Pr\{\breve{M} = m\}$, i.e., the probability that the random variable $\breve{M}$ referring to the the number of decoded layers takes on the value m. As this probability depends on the code rate vector $\mathbf{r}$ and the number of available transmission slots F we define

$$p_{\breve{M}}(m, F, \mathbf{r}) \triangleq \Pr\{\breve{M} = m\} = \sum_{j=0}^{F-m} p_c(m, F-j) p_o^{(j)}\left(r_{m+1}\right), \tag{8.8}$$

where $p_o^{(j)}(r)$ is the outage probability according to (4.8) and $p_c(m, f)$ is the probability that exactly m layers can be decoded within the next f transmission slots. This probability can be recursively computed as

$$p_c(m, f) \;=\; \sum_{j=0}^{f-(m-1)} p_c(m-1, f-j) p_d^{(j)}\left(r_m\right),$$

for $m \geq 1$, $f \geq m$ and initialization $\forall_{f=1,\ldots,F}\, p_c(0, f) = 0$ and $p_c(0, 0) = 1$.

With (8.6) and (8.8) the expected quality $\mathbb{E}\{\mathcal{Q}(F, \mathbf{r}, s)\}$ depending on the number of available transmission slots F, the code rate vector $\mathbf{r}$, and the actual source data s is given as

$$\mathbb{E}\{\mathcal{Q}(F, \mathbf{r}, s)\} = \sum_{m=0}^{M} \mathcal{Q}_m(s) \cdot p_{\breve{M}}(m, F, \mathbf{r}), \tag{8.9}$$

where $\mathcal{Q}_m(s)$ can be derived from the importance vector $\mathcal{I}(s)$ according to (3.30). We aim to maximize $\mathbb{E}\{\mathcal{Q}(F, \mathbf{r}, s)\}$ with respect to the code rate vector $\mathbf{r}$. However, in contrast to FEC where we do not know if a layer has been correctly decoded at the receiver in the case of ARQ schemes we can perform a new rate allocation before transmitting a new information part $\mathbf{u}_{\hat{m}}$ with the knowledge of the number of residual radio slots $\tilde{f}$. Therefore, we derive the expected quality assuming that the next layer to be transmitted is $\hat{m}$ and having $\tilde{f} = F - f + 1$ residual radio slots as

$$\mathbb{E}\left\{\mathcal{Q}(\tilde{f}, \hat{m}, \mathbf{r}_{[\hat{m}:M]}, s)\right\} = \sum_{m=0}^{\hat{m}-1} \mathcal{Q}_m(s) + \sum_{m=\hat{m}}^{M} \mathcal{Q}_m(s) \cdot p_{\breve{M}}(m, \tilde{f}, \mathbf{r}_{[\hat{m}:M]}), \tag{8.10}$$

with $\mathbf{r}_{[\hat{m}:M]} \triangleq \{r_{\hat{m}}, \ldots, r_M\}$.

For every new layer $\hat{m}$ to be transmitted we aim to solve the following code rate allocation

$$\mathbf{r}^*{}_{[\hat{m}:M]} = \arg \max_{\mathbf{r}_{[\hat{m}:M]} \in \mathcal{S}_r^{M-m+1}} \mathbb{E}\left\{\mathcal{Q}(\tilde{f}, \hat{m}, \mathbf{r}_{[\hat{m}:M]}, s)\right\} \tag{8.11}$$

with $\mathcal{S}_r$ the set of accessible channel code rates given by the puncturing tables. Brute force search can be prohibitively complex, especially for large $|\mathcal{S}_r|$ and a large number of layers. Unfortunately, as $p_c(m, f)$ is calculated recursively a complexity reduced algorithm similar to the UEP case cannot be established. However, as the future code rates $r_{\hat{m}+1}, \ldots, r_M$ can be allocated later we assume that the code rate for all layers will be same to find the optimized code rate $r^*_{\hat{m}}$, i.e.,

$$r^*_{\hat{m}} = \arg \max_{r \in \mathcal{S}_r} \mathbb{E}\left\{\mathcal{Q}(\tilde{f}, \hat{m}, \{r_{\hat{m}} = r, r_{\hat{m}+1} = r, \ldots, r_M = r\}, s)\right\}. \tag{8.12}$$

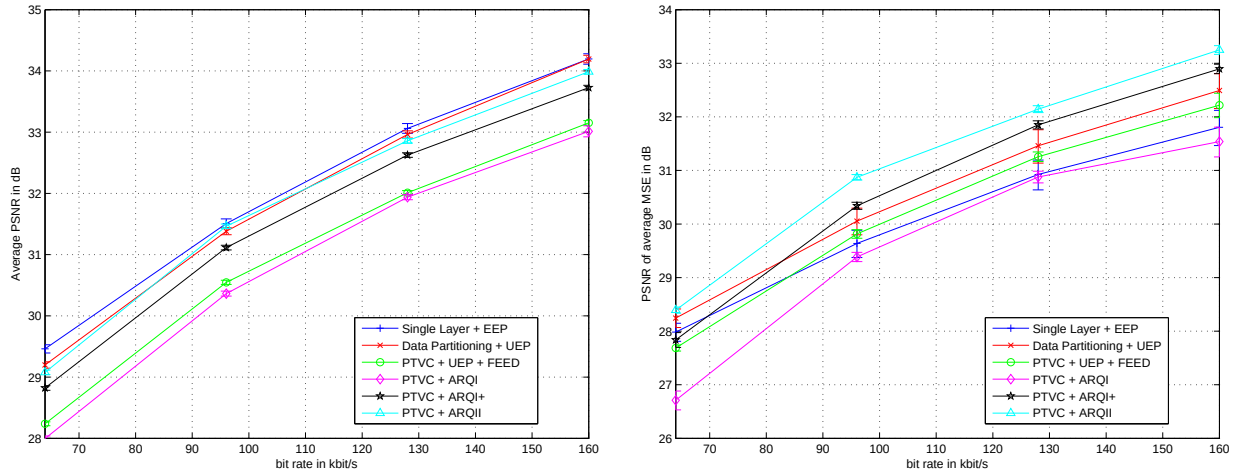
Experimental Results

The simulation environment as already used for the H.264/AVC-based transmission as well as for the UEP-based scheme in combination with PTVC has been further extended to integrated the ARQ schemes. We again use the systematic convolutional code with memory $\mu = 96$, mother code rate $1/c = 1/7$ and puncturing period $\omega_p = 32$. The computational limit for the sequential decoder is again set to $C_{\text{lim}} = 16$. In the following experiments the rate allocation for the PTVC and ARQ schemes targets to minimize the MSE for all cases.

In Figure 8.17 the PSNR over the total bit-rate performance for different ARQ systems for sequence *foreman* compared to PTVC with UEP and H.264/AVC-based systems is shown. The reference systems being single layer H.264/AVC (blue curve), data partitioning with UEP (red curve), and PTVC with UEP and FEED (green curve) are a copy from Figure 8.14 and therefore need no further explanation.

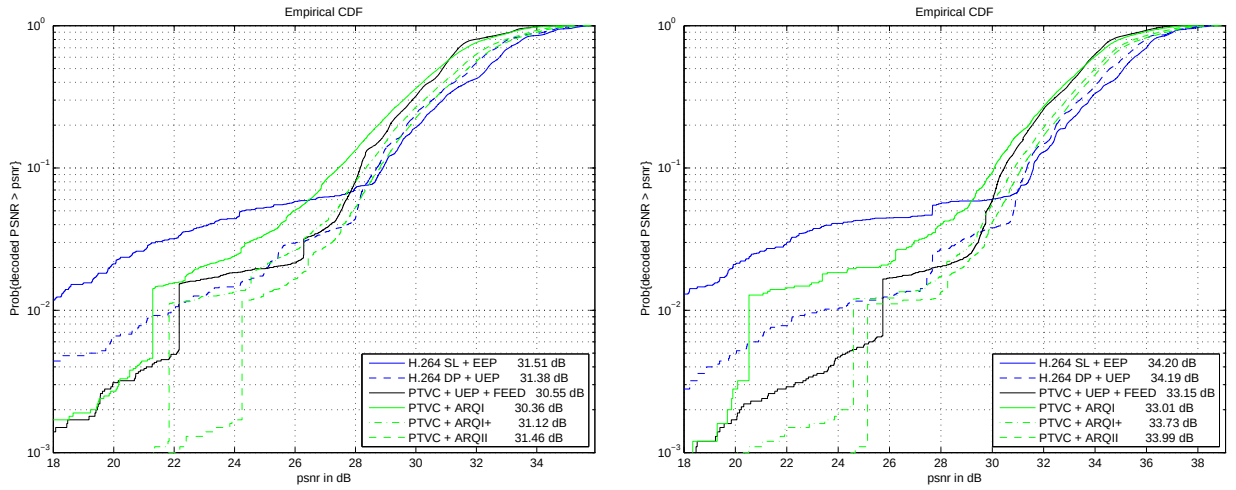
Figure 8.17a) shows the performance in terms of average PSNR: The performance of ARQI is not satisfying, it is even outperformed by the simple UEP approach. The ARQI+ scheme using diversity reception performs significantly better. The ARQII approach, though optimized towards MSE, performs very well and basically as good as the H.264/AVC based schemes. The good performance of the ARQ-based schemes together with PTVC are even more obvious when using the MSE as performance measure: Figure 8.17b) shows that ARQI+ and ARQII outperform all FEC-only based schemes. Especially ARQII provides about 1.5dB gain in PSNR and about $30 - 35$% in bit-rate compared to single layer H.264/AVC with EEP and about 0.8dB gain in PSNR and about $10 - 15$% in bit-rate compared to data partitioning with UEP.

More insight into the reasons for this performance is provided in Figure 8.18: The distribution of the PSNR for each individual frame with different total bit-rate of 96 kbit/s in a) and 160 kbit/s in b)



(a) $\overline{\mathrm{PSNR}}$ (b) $\mathrm{PSNR}_{\overline{\mathrm{mse}}}$

Figure 8.17: PSNR over total bit-rate performance for different ARQ systems for sequence *foreman* compared toPTVC with UEP and H.264/AVC-based systems.



(a) Total bit-rate 96 kbit/s (b) Total bit-rate 160 kbit/s

Figure 8.18: Distribution of PSNR for ARQ schemes and PTVC, different bit-rates and sequence *foreman* compared to PTVC with UEP and H.264/AVC-based systems.

are shown. The H.264/AVC-based systems (blue curves) as well as the PTVC combined with UEP system (black curves) are copied from Figure 8.15 and do not need further explanation. The green curves show the performance of the ARQ-based systems. We focus on Figure b) for 160 kbit/s: With ARQI+ and especially ARQII, low quality images below 25dB can basically be completely avoided which explains the extremely good performance for the MSE-based performance in Figure 8.17b). The consistent performance of the ARQI+ and ARQII based systems make these systems very attractive for mobile conversational services. The only requirement is the availability of a fast feedback channel and retransmission which is the case in emerging mobile radio systems. An optimized rate allocation can maximize the performance. In the combination with the PTVC the adaptation to the actual transmission conditions is feasible and the adaptation of the reference frames completely avoids mismatches and error propagation.

8.3.4 Channel-State Information at Transmitter

System Overview

In case of UEP as introduced in subsection 8.3.2 and in case of ARQ-based systems according to subsection 8.3.3, the actual channel conditions are not known during the rate allocation process. Therefore, in both schemes for many cases resources on the physical layer are wasted in a sense that data is transmitted that is of no use for the receiver. For example, in case of UEP all layers that are not decoded, are of no use for the receiver. Furthermore, the code rate for the initial layers may have been far too low. For ARQ systems, the total FEC code rate does usually also over-provision the channel, such that the allocation between source and channel coding rate is sub-optimal.

This waste of resources can be avoided, if the encoder has knowledge of the fading conditions in the up-coming transmission slots by using accurate and up-to-date Channel State Information (CSI). The availability of CSI information may be based on frequent measurements and the fast feedback of this information from the receiver to the transmitter. In TDD schemes, the information may be derived directly from the return channel. Schemes as introduced in subsection 4.1.4 make use of the availability of the CSI by adapting the code rate and/or the transmit power to the expected channel conditions. We distinguish two scenarios: In the constant–power transmission mode the transmission power for each slot is fixed, i.e., $\gamma_f = 1$, but the code rate r_f can be adapted to compensate the channel. In the power–controlled mode in addition to the knowlegde of the channel gain α_f for the next transmission slot, the transmitter only needs to fulfill the long–term power constraint $\mathbb{E}\{\gamma_f\} \leq 1$. However, similar as for the ARQ schemes, in case of code rate adaptation, it is essential that the source adapts to the transmission conditions. The combination with a progressive source coding algorithm, in particular the PTVC, provides therefore a promising concept and is introduced in more detail in the following.

Figure 8.19 shows the proposed system that extends the ARQ-based system introduced in Figure 8.18: By encoding of the video frame s_t, the PTVC encoder generates a binary index $\mathbb{I}_{M,t}$ and the corresponding layered importance vector $\mathcal{I}(s)$. Then, for each radio slot f, the power and rate allocation selects an appropriate channel code rate $r_f^* \in \mathcal{S}_r$ and, if applicable, an appropriate power control factor γ_f^*. Note that the allocation may choose to not transmit at all during this slot and select $\gamma_f^* = 0$. If this is the first slot within this message, i.e., $f = 1$, the transmitter control selects $k_{m=1} = N_v r_{m=1}^*$ information bits from the progressively coded index $\mathbb{I}_{M,t}$ to generate the information message $\mathbf{u}_{m=1}$. This message is encoded by the channel code with rate $r_{f=1}^*$ to obtain the code symbol vector $\mathbf{z}_{f=1}$. By appropriate puncturing and mapping the transmitter generates the transmission message $\mathbf{x}_{f=1}$ of length N_v. If appropriate, the selected power control is applied before the transmission message $\mathbf{x}_{f=1}$ is transmitted. At the receiver again the LLR

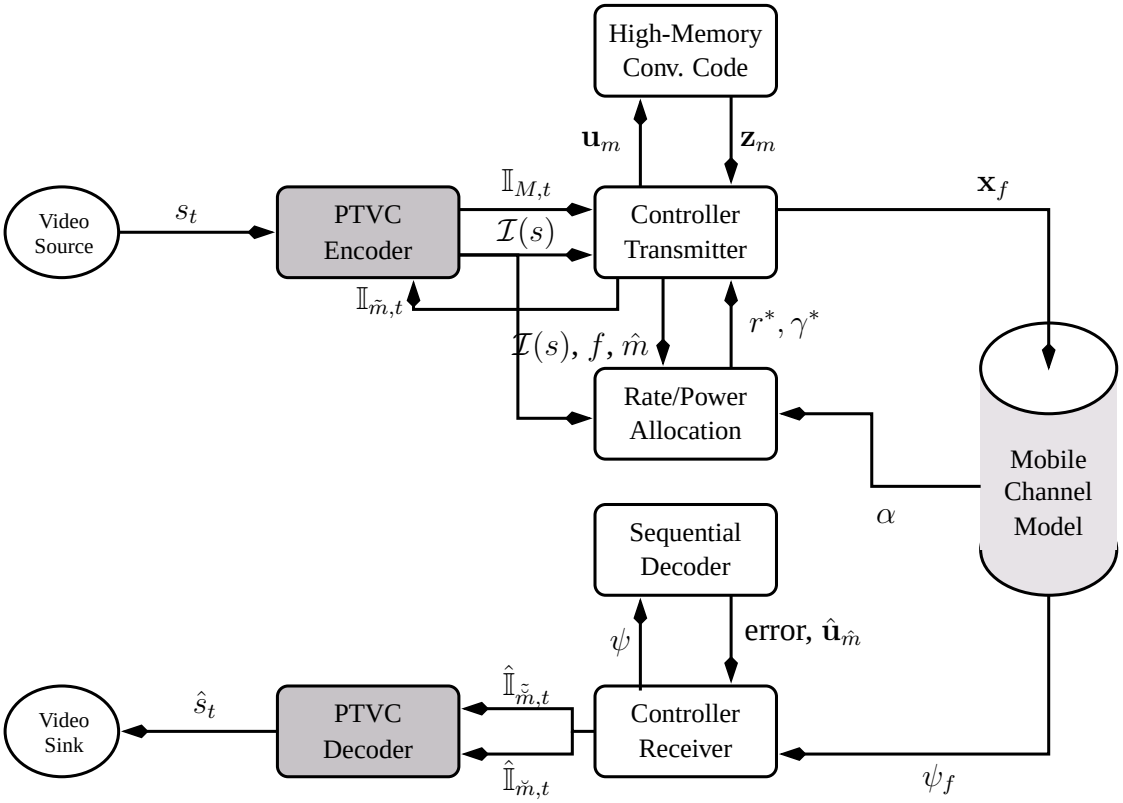


Figure 8.19: Mobile conversational video transmission system with channel–state information at transmitter.

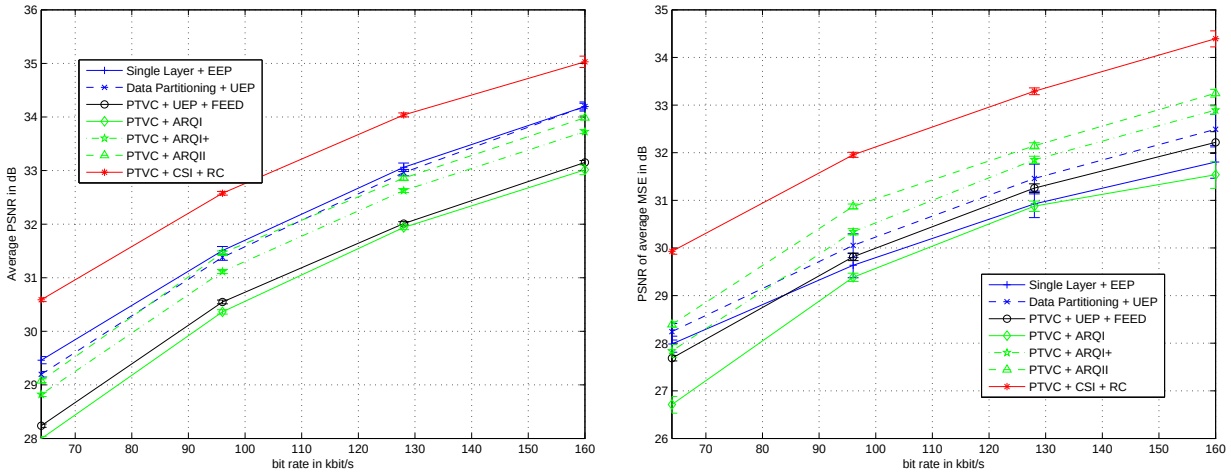
$\psi_{f=1}$ is generated and by inverse puncturing we obtain the LLR ψ which is passed to the channel decoder. The channel decoder either decodes the message $\hat{\mathbf{u}}_m$ or indicates a decoding failure. In this case, any ARQ scheme as proposed in subsection 8.3.3 may be applied. However, in the following we assume that the selected combination of $\{\gamma_f^*, r_f^*\}$ results in the desired performance and no errors have occurred. The received message $\hat{\mathbf{u}} = \mathbf{u}$ is stored in the receiver. With the transmission of the next radio slot f a new rate allocation is performed based on $\mathcal{I}(s)$ to obtain the optimized allocation rate $\{r_f^*, \gamma_f^*\}$. The next $k_m = N_v r_m^*$ information bits are taken from the binary index $\mathbb{I}_M$ to obtain the information part $\mathbf{u}_m$ which is again encoded, punctured, transmitted and processed at the receiver. Transmitter and receiver terminate the transmission and reception of the corresponding message as soon $f = F$ radio slots have been processed.

Assume that the number of successfully transmitted information parts is denoted as $\breve{m}$. At the receiver the decoded index $\hat{\mathbb{I}}_{\breve{m}}$ is obtained by the concatenation of the received information parts $\{\hat{\mathbf{u}}_{m=1}, \ldots, \hat{\mathbf{u}}_{\breve{m}}\}$. $\hat{\mathbb{I}}_{\breve{m}}$ is decoded to obtain $\hat{s}_t$. The same index is used to reconstruct the reference frame at the decoder and the encoder, i.e., $\hat{\mathbb{I}}_{\breve{\tilde{m}}} = \hat{\mathbb{I}}_{\tilde{m}} = \hat{\mathbb{I}}_{\breve{m}}$. Based on this, the transmitter generates a new reference frame by decoding $\mathbb{I}_{\tilde{m}}$. Then, the new frame s_{t+1} can be encoded and the processing and transmission of the subsequent binary index $\mathbb{I}_{M,t+1}$ starts.

This proposed scheme is generic for all following schemes, they will only differ in how the power and rate allocation are carried out. We differentiate a scheme with constant transmit power and channel code rate allocation only, a scheme which selects $\gamma_f^*, \{r_f^*\}$ to maximize the throughput as proposed in (4.18), and one scheme which also takes into account the importance $\mathcal{I}(s)$ for the power and rate allocation.

Channel State Information for Code Rate Control

In a first system design that exploits CSI, we maintain the constant power constraint with $\gamma = 1$ for all slots. Only the code rate is changed during the allocation process for each slot. Assume that the rate allocation has knowledge of the channel SNR h_f for the upcoming transmission slot f. Then it selects the code rate r_f for this slot to compensate the channel, i.e., $r_f \leq R_0(h_f)$. As only a discrete set of code rates is available, the actual code rate r_f is selected as the largest code rate in $\mathcal{S}_r$ that fulfills $r_f \leq R_0(h_f)$. If no code rate fulfills this criteria, $r_f = 0$ is selected, i.e., no message can be transmitted in this slot.



(a) $\overline{\text{PSNR}}$ (b) $\text{PSNR}_{\overline{\text{mse}}}$

Figure 8.20: PSNR over bit rate performance for CSI applied for rate control for sequence *foreman* compared to other modes.

In Figure 8.20 the PSNR over the total bit-rate performance for the PTVC with CSI for code rate control (red curve) for sequence *foreman* is shown compared to PTVC with UEP (black curve) and ARQ (green curves) as well as H.264/AVC-based systems (blue curves). Figure 8.20a) shows the performance in terms of average PSNR: The proposed system with CSI outperforms any other system by about 1dB which results in bit-rate savings of about 30%. With this system, even H.264/AVC based systems can be outperformed. When using the MSE as performance measure: (see Figure 8.20b) the performance gains are similarly impressive despite the rate allocation does not take into account any SSI. Especially compared to H.264/AVC based schemes, the bit-rate savings are in the range of 50%. Note that such a CSI-based scheme cannot be implemented easily with H.264/AVC, as the rate for each frame needs to be adapted before the first part is transmitted. The PTVC allows the adaptation during the transmission. Some more insight on the reason for the good performance of CSI-based schemes will be reported in Figure 8.23.

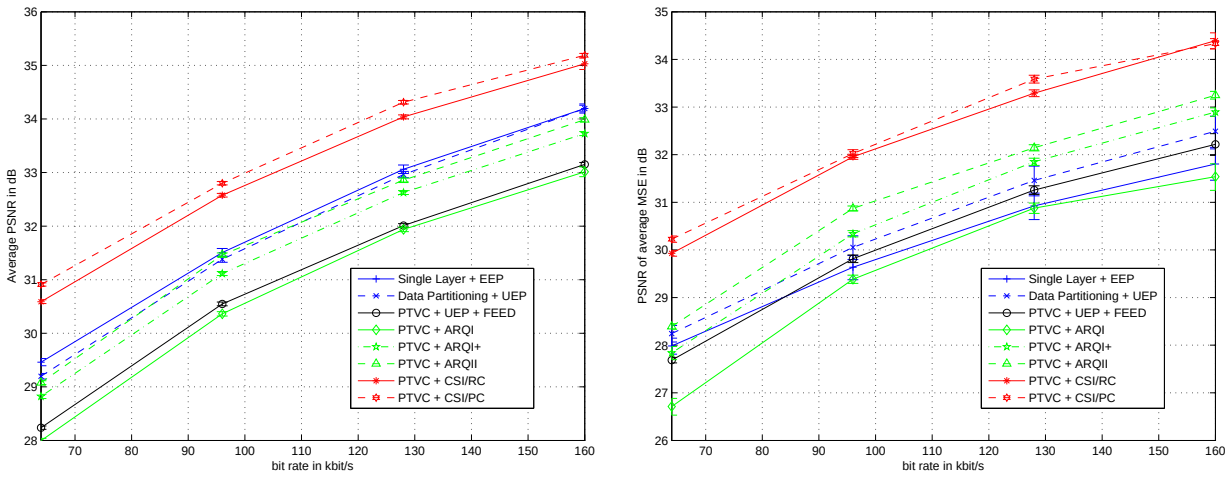
Channel State Information for Power Control

In a second scenario the power constraint is relaxed in a sense that only a long–term power constraint has to be fulfilled by $\mathbb{E}\{\gamma_f\} = 1$. This may be be well justifiable in a multi-user communication environment with shared resources such as in UMTS. Assume that the resource allocation has knowledge of the channel SNR h_f for the upcoming transmission slot f. Following (4.18) it

selects the power control factor as

$$\gamma_f^* = \left[\frac{1}{h_f} \log \left(\frac{h_f}{\lambda \log(2)} - 1 \right) \right]_+ , \tag{8.13}$$

where λ is the solution to the constrained equation $\mathbb{E}_H \{\gamma(H, \lambda)\} = 1$. We assume that the distribution p_α of the channel gain α or or equivalently of the distribution of the SNR h are known at the transmitter to compute λ. The code rate r_f^* is then selected as the largest code rate in $\mathcal{S}_r$ that fulfills $r_f^* \leq R_0(\gamma_f^* h_f)$. The power control or the code rate selection may result in $\gamma_f^* = 0$ or $r_f^* = 0$, i.e., no message can be transmitted in this slot.



(a) $\overline{\text{PSNR}}$ (b) $\text{PSNR}_{\overline{\text{mse}}}$

Figure 8.21: PSNR over bit rate performance for CSI applied for power control for sequence *foreman* compared to other modes.

In Figure 8.21 the PSNR over the total bit-rate performance for the PTVC with CSI for power control (red dashed curve) for sequence *foreman* is shown compared to the modes shown in Figure 8.20. For both metrics, the CSI with power control can improve the performance even more, but the gains compared to CSI with rate control are less significant by about at most 0.3dB in PSNR.

Combined Channel–State Information and Source–Significance Information

The previous two system proposals relying on CSI completely ignored any SSI in the selection of power control factors and channel code rates. In the constant power case, the inclusion of SSI is not possible, as the code rate is directly selected based on the channel state. However, in case of power allocation, the allocation may trade some power towards more important data. We acknowledge this by modifying the power allocation to take into account the operational importance vectors of the PTVC encoded frame.

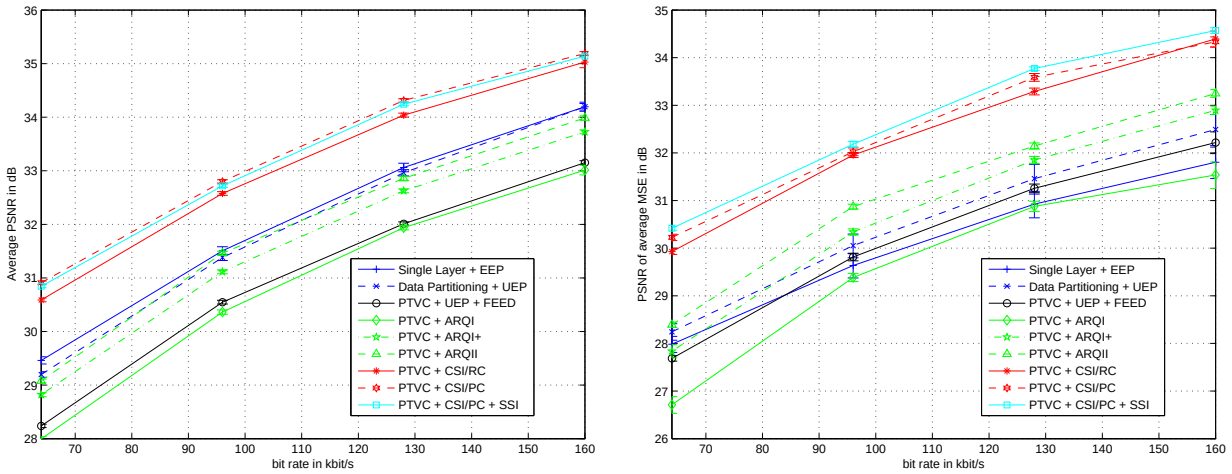
Assume a progressively coded video frame with a quality rate function $\mathcal{Q}(k)$, or equivalently expressed by the importance vector $\mathcal{I}$. A closed solution for the optimized power allocation taking into account this SSI is not feasible. However, an unconstrained optimization based on Lagrangian formulation may be applied to select the power control factor γ_f for each slot. The quality gain by applying a certain power control factor γ_f must be traded with the power consumption taking into account the overall power constraint of $\mathbb{E}\{\gamma_f\} = 1$.

For a given channel state h_f for this slot f, the application of a certain power allocation factor γ results in a support of a code rate $r = R_0(\gamma h_f)$ and a certain number of source bits of rN_v. This

amount of bits results in a quality improvement of $\mathcal{Q}(K_{f-1} + rN_v) - \mathcal{Q}(K_{f-1})$. Based on this quality improvement, the optimized power control factor γ_f^* is selected as

$$\gamma_f^* = \arg\max_{\gamma \in \mathbb{R}_0^+} \left\{ \mathcal{Q}\left(K_{f-1} + R_0(\gamma h_f) N_v\right) + \lambda\gamma \right\}. \tag{8.14}$$

The selection of the appropriate Lagrange parameter λ must be such that the average power constraint is maintained but this not obvious initially and may depend on the video sequence characteristics. However, in our scheme we propose to apply a Lagrange parameter suitable for the case without SSI and then gradually adapt λ to maintain the average power constraint of $\mathbb{E}\{\gamma_f\} = 1$. The update steps and frequency are implementation-specific and need to be carefully selected to avoid oscillations and instabilities - slow updates are preferred. In the following we provide some simulation results for which the MSE was used as criteria to select the power control factor.

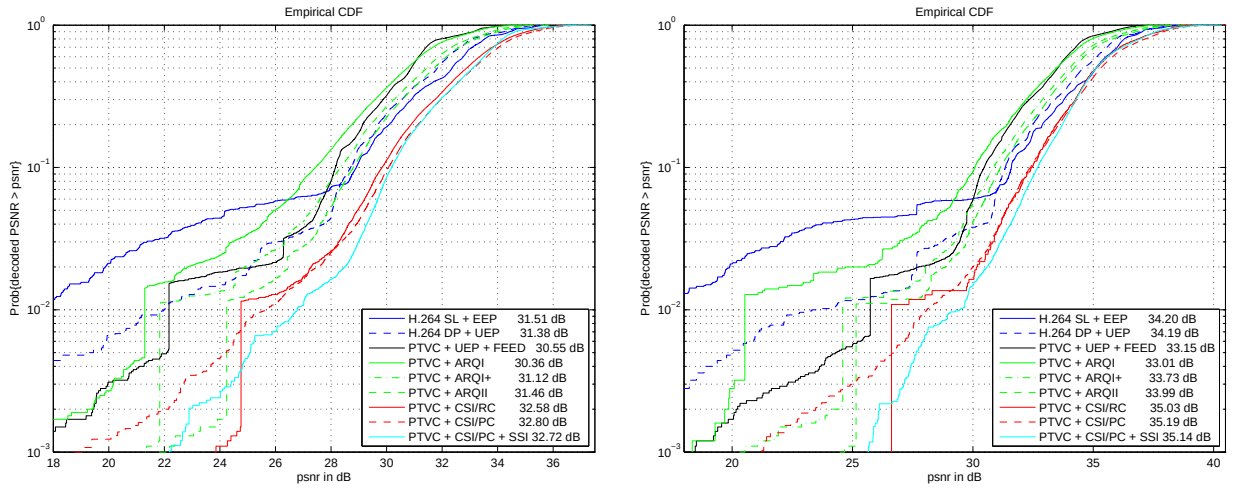


(a) $\overline{\text{PSNR}}$ (b) $\text{PSNR}_{\overline{\text{mse}}}$

Figure 8.22: PSNR over bit rate performance for combined CSI applied for power control and SSI for sequence *foreman* compared to other modes.

In Figure 8.22 the PSNR over the total bit-rate performance for the PTVC with CSI and SSI using power control (cyan curve) for sequence *foreman* is shown compared to the modes shown in Figure 8.21. It is observed in Figure 8.22a) that for this mode, the average PSNR is slightly degraded compared to the power control without SSI information. However, when using the MSE as performance criteria, the performance can be further enhanced by about 0.2dB. The reason for this improvement is discussed in more detail in Figure 8.23 which is also used to summarize the performance of different schemes for the transmission of low-delay conversational video applications.

The distribution of the PSNR for each individual frame is shown in Figure 8.23 with different total bit-rate of 96 kbit/s in a) and 160 kbit/s in b). The H.264/AVC-based systems (blue curves) as well as the PTVC combined with UEP (black curve) and ARQ (blue curves) are copied from Figure 8.18 and do not need further explanation. The red and cyan curves show the performance of the CSI-based systems. We focus on Figure b) for total bit-rate160 kbit/s: The system with fixed power constraint and code rate control only (solid red curve) shows that bad frames below 26dB are basically never present. This is due to the fact that in any case some information is transmitted. In contrast, for the case of power control, the probability of high PSNR frames is slightly higher, but due to exclusion of some slots for transmission, some low-quality frames are present as well.



(a) Total bit-rate 96 kbit/s (b) Total bit-rate 160 kbit/s

Figure 8.23: Distribution of PSNR for all PTVC modes with UEP, ARQ and CSI error protection schemes compared to H.264/AVC modes.

This deficiency can be eliminated with the use of SSI (cyan curves). In this case, the power control ensures a more consistent quality. Overall, the system applying power control with SSI provides an excellent quality and shows that a cross-layer design mobile conversational video system can provide high and consistent quality.

Based on these results, schemes that share resources for power and possibly also bit-rate among several or all users in a mobile system are promising candidates for emerging mobile systems. For example, HSDPA relies on this concept, but has not been fine-tuned to mobile conversational applications yet. In our work for streaming over HSDPA [LJSB04], we have shown that delay-limited applications can be well supported by such systems. Furthermore, we have provided some theoretical justifications, for the combination of SSI and CSI when sharing resources among different users in a multi-user environment: The concepts introduced in [MS02a, MS02b] show that by making use of SSI, a significant statistical multiplexing gain can be achieved. The realization of such concepts for mobile conversational video applications should be subject of future work. However, the tools and concepts introduced in this chapter are promising candidate components for such systems.

8.4 Summary

In this chapter different system and cross-layer designs for mobile conversational video applications have been introduced. The following observations and findings are of major importance:

- Mobile conversational video applications introduce challenges in the system design as the delay constraints as well as the variability and bandwidth limitation of the mobile radio channel requires careful selection of the source coding tools, the error protection tools, as well as the resource allocation. To obtain insight into good system designs, a simplified, but representative evaluation framework has been designed to compare different system designs.

- In terms of video coding, we have integrated H.264/AVC with single layer coding and data partitioning, as well as the PTVC. Specific error resilience schemes are also supported. For error protection, FEC schemes applying EEP and UEP, ARQ schemes in combination with

FEC, as well as power control have been integrated and investigated. All of the schemes allow adaptation to transmission conditions, especially if appropriate feedback messages are in place. The different schemes provide tradeoffs in terms of flexibility, performance and implementation complexity. Furthermore, it is essential, that the combination of tools is supported by appropriate resource allocation schemes to exploit the full potential of the system towards meeting the service requirements.

- For H.264/AVC single layer scheme, the combination with EEP has been investigated. In a simple system design, a constant channel code rate has been applied, and the use of different error resilience schemes has been investigated, namely, channel-optimized mode selection and IEC with different feedback delay. For all metrics and systems an optimized channel coding rate could be determined. If using this optimized rate, it turns out that video error resilience is of little relevance, and may be omitted entirely – the classical separation of source channel coding provides also a good system design for low-delay mobile conversational video transmission. However, the selection of the appropriate code rate is not necessarily feasible in the operation of a system. Hence, error resilience may be of importance for robust system design. Furthermore, if fast video feedback is available for IEC to compensate losses, higher channel code rates can be selected such that the overall performance is increased.

- Overall IEC-based systems with fast feedback provide the best option to avoid error propagation in variable transmission conditions. Therefore, the fast IEC approach has been adopted to the majority of our system designs.

- The H.264/AVC-based system has been extended by the implementation of a source-adaptive code rate selection. However, except for providing an operationally feasible system, no gains in terms of performance could be observed by adapting the code rates for each video frame when compared to the non-adaptive scheme with optimized code rate selection.

- In a further extension of the H.264/AVC-based system, data partitioning has been integrated. Due to the different importance of the partitions, generally an UEP scheme is the preferred FEC option. It was observed that data partitioning together with UEP provides a powerful scheme to adapt to varying transmission conditions in H.264/AVC-based mobile conversational video systems, specifically it avoids the loss of entire video frames. However, it is essential that the source and channel code rate allocation is done appropriately and adaptively such that the different importance of each partition is reflected. A suitable algorithm taking into account the expected quality has been proposed for quantization parameter and UEP-based channel code rate allocation.

- Significantly better adaptation to the varying channel conditions can be obtained by the use of progressively coded video frames. Therefore, the PTVC has been integrated in a mobile conversational video communication system including IEC-based drift elimination. The basic idea for all error protection schemes in combination with PTVC is to push the first error as far as possible out in the progressively coded bit-stream for each video frame.

- A combination with regressive UEP and the FEED-based decoding provides exactly this property and has been integrated in the system, together with the source–channel–optimized rate allocation as already introduced in subsection 5.4.2. The performance of such a system can compete with H.264/AVC-based systems. Especially, if more consistent video quality

is of higher importance than high average PSNR, the combination of PTVC and UEP with FEED at the receiver is an attractive option.

- The same principle, channel-adaptation as well as maximizing the amount of ”good” data for each video frame, can be accomplished by the use of non-persistent ARQ schemes. Especially with hybrid ARQ schemes of type 2, low quality video frames can be completely avoided. This consistent performance makes non-persistent ARQ systems very attractive for mobile conversational services in case a fast feedback channel for retransmission is available. A quality-optimizing channel code rate allocation can maximize the performance. Furthermore, for the combination with the PTVC the adaptation to the actual transmission conditions can be accomplished: the adaptation of the reference frames in the PTVC encoder can be realized in the transmitter without specific VFI, only based on the ARQ status messages. This procedures completely eliminate any error propagation problems.

- The radio resources can be exploited most efficiently if the transmitter is aware of the Channel State Information (CSI) before allocating the transmission resources code rate and power. By the use of these means, the combination of PTVC and CSI-based rate allocation schemes, any other system, including H.264/AVC-based systems can be outperformed by 1dB in PSNR and $30 - 50\%$ in bit-rate savings.

- Even more powerful schemes relax the constant transmit power regime for each slot to only an average constraint - such schemes are common nowadays in emerging mobile multi-user systems. By applying power control to each radio slot, some additional gains could be achieved. Furthermore, such a power allocation scheme can be modified to take into account SSI in terms of operational importance vectors of each PTVC encoded frame. We have introduced an SSI-aware power control scheme for which the results show the overall best results with very consistent quality. This system, combining PTVC with reference frame adaptation, powerful FEC, as well as SSI-aware power control provides an excellent quality and shows that by the application of cross-layer design mobile conversational video system can provide high and consistent performance.

9

Video Streaming over Variable Bit-rate Mobile Channels

9.1 Background

The transmission of streaming applications over mobile channels differs significantly from low-delay communications as discussed in section 7.2 and chapter 8. Relaxed delay constraints allow the use of advanced radio layer quality-of-service features to compensate short-term fluctuations in the channel quality. For example, variations within a few milliseconds due to short-term fading and interference can be compensated by the use of interleaving and forward error correction. Shadowing effects may still be compensated by the use of power control, rate adaptation and retransmission protocols but if the variations are too severe, for example at cell edge areas, or due to handover or changes in the overall system load, the transmission bit-rates generally need to change. The video must react to the modified channel conditions in an adaptive way to prevent congestions and losses. If the video is online encoded or transcoding is possible, and if the encoder has sufficient information on the updated transmission conditions, operational encoder control strategies for VBR channels can be applied [OR98]. The encoder rate control dynamically adapts to variable transmission bit-rates [HOK99].

However, online encoding or transcoding are generally too complex and not feasible for streaming servers which commonly have to serve several or many connections at the same time. Streaming of pre-encoded content is highly preferred. In this case, other means for video rate adaptation are necessary: In case of short-term channel bit-rate variations, playout buffering at the receiver can compensate for bit-rate fluctuations such that the display timeline is maintained.

We have developed a framework in [SJK04] which ensures the decoding of each video unit before exceeding its display deadline and, hence guarantees successful sequence presentation even if the media rate does not match the channel rate. This is accomplished by the use of a delay jitter buffer of the radio system to compensate the variable bit-rate channel. However, in addition to the delay jitter buffer, the video decoder usually also requires a buffering unit as has been shown in Figure 2.4 in case that the video is not fully constant bit-rate.

The work in [SJK04] rigorously proves that the separation of the dejitter buffer of the channel

and the decoder buffer is in general suboptimal for VBR video transmitted over VBR channel. The required initial playout delay for separate buffers is always at least as high as the for a joint dejitter and decoder buffer. For a given video-stream and a deterministic VBR channel, an exact expression of the minimum initial delay and the minimum required buffer can be calculated. However, in general the channel variatifons are non-deterministic. For this case, we were able to provide probabilistic statements in case that we observe a random behaviour of the channel bitrate: The probability of non-successful decoding for a given initial delay could be provided, similar to the experimental results in Figure 7.5 and Figure 7.6. The generic framework had been applied to a specific example tailored to mobile video streaming: We were able to derive bounds that allow to guarantee a certain quality-of-service even for random VBR channels in a wireless environment. Specifically, it has been shown by probabilistic statements that for UMTS-like channels the bit-rate variations due to link layer retransmissions can be well compensated by receiver buffering without adding significant additional delay. Simulation results validate the findings. In addition to receiver buffering in case of anticipated buffer underrun, techniques such as adaptive media playout [KSG04] enable a streaming media client, without the involvement of the server, to control the rate at which data is consumed by the playout process.

Nevertheless, if the reception conditions are too heterogeneous, playout buffering and adaptive media playout are not sufficient to compensate for bit-rate variations in wireless channels. In this case, rate adaptation of pre-encoded streams has to be performed by modifying the encoded bit-stream. Some means to perform this adaptation have already been introduced in chapter 3. Such an adaptation can be carried out at different instances in the network: At the streaming server, in intermediate routers, or at the entry gateway to the mobile access network. Different methods are for example discussed in [HKS96] and [KZ02]. The preferred instance may be the one closer to the air interface, as there exists more up-to-date channel state information about the expected transmission conditions which would allow making better decisions. However, in contrast, a streaming server usually has much more intelligence to react to variable bit-rates than intermediate routers or gateways: Routers usually only drop packets in case of congestion without taking into account their individual importance.

To address intelligent rate adaptation beyond simple packet dropping, we introduce a rate adaptation entity - referred to as *scheduler*. The scheduler has sufficient information and intelligence to be able to drop packets with respect to their relative importance. A formalized framework under the acronym rate-distortion optimized packet scheduling has been introduced in [CM06] and has been extended in section 3.6 of this work. We will use this framework to define more and less important packets in a stream. However, packets with different importance can only be obtained by appropriate encoding means as shown in chapter 3: Therefore, if variations on the transmission path are expected, it is beneficial to pre-encode media streams with appropriate packet dependencies, such that the importance of the packets in the stream can be easily differentiated by the network components.

Based on these discussions we propose an H.264/AVC streaming system for variable bit-rate mobile channels. This system relies on three different means to support bit-rate adaptivity, namely

1. receiver playout buffering to compensate small channel variations,
2. temporal scalability to generate packets of different importance, and
3. advanced bit-stream switching as introduced in section 3.3.4.

The objective of this chapter is to understand the performance and potentials of rate adaptation features for variable bit-rate switching in realistic scenarios. Therefore, the remainder of this chapter is structured as follows: A brief overview of an end-to-end wireless video streaming system is

provided in section 9.2. For the various features available in H.264/AVC to support temporal scalability and bit-stream switching we refer to chapter 3. Also, a framework for describing video streaming over arbitrary VBR channels is introduced. We will use a specific example of a VBR channel to explain the proposed streaming system in more detail. Furthermore, a statistical model to describe the influence of mobile links on packet transmission is introduced. As an example system we will consider a variable bit-rate EGPRS system according to the model introduced in section 2.4.4. We propose a relatively simple, yet sufficiently accurate description of the channel characteristics. In section 9.4, we integrate the developed concepts into an optimized decision making strategy for the selection of frames and versions in a mobile streaming scenario. Experimental results for H.264/AVC video streaming over EGPRS links demonstrate the applicability of our strategy in section 9.5.

9.2 System Overview

Figure 9.1 shows the considered mobile streaming system including an end–to–end connection between a media streaming server and a client. Assume that the client requests H.264/AVC pre–encoded data from the server to be streamed to the end user. The client buffers the incoming data and starts with decoding and presentation of the reconstructed video sequence after some initial delay. Once playback has started, a continuous presentation of the sequence should be guaranteed. However, in our investigated system neither the bit-rate, nor the delay is constant, and some data units may not even be available at the decoder. Therefore, the media streams stored at the server have to be not only compression efficient: It should also be possible to flexibly adapt their bit-rate to varying conditions on the wireless link.

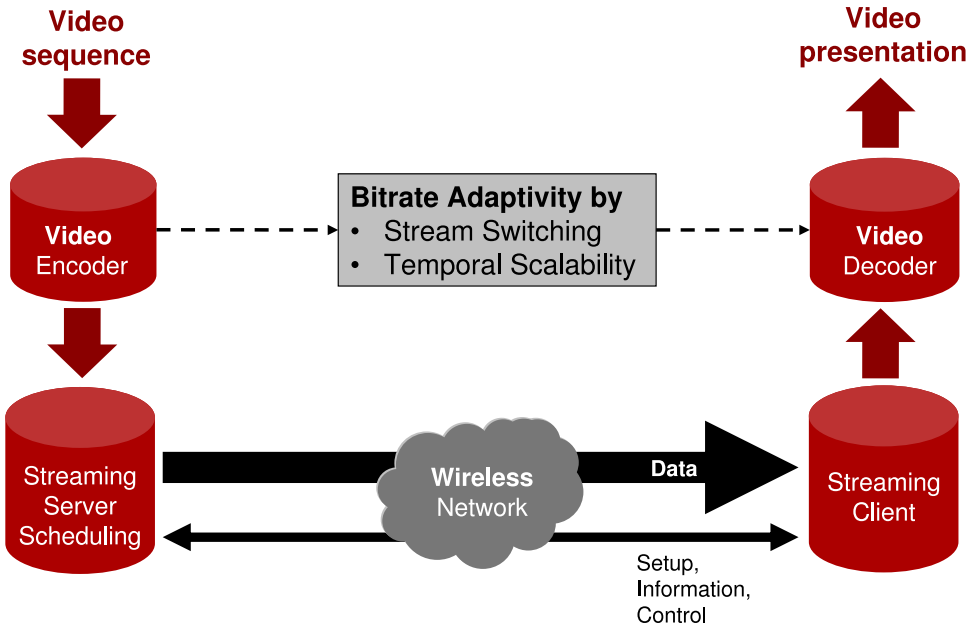


Figure 9.1: Mobile streaming system to support variable-bit-rate channels.

The flexible reference frame concept in H.264 as introduced in chapter 3 including generalized B-pictures provide a huge flexibility to generate different frame dependencies. Such subsequences can be exploited for temporal scalability and rate shaping of pre-encoded video. For example, for video including non-reference frames the rate can easily be adapted as dropping of non-reference

frames does not result in error propagation. The adaptation of bit-rates in time scales larger than the initial playout delay requires addition means. As the bit-rate on mobile links is precious, especially when compared to storage on servers, it should be ensured that the data-on-air is efficiently compressed. By the availability of sufficient buffer feedback as well as channel state information the streaming server has at least anticipation on the currently supported bit-rate. There exist many different means for end-to-end bit-rate estimation but relying on end-to-end TCP-like bit-rate estimation for mobile links does usually not work well. However, the streaming server and /or the client may poll the network on the actual rate conditions and may get appropriate feedback to adjust the bit-rate to the available one in the network.

Under these conditions bit-stream switching provides a simple but powerful concept to support bit-rate adaptivity. According to section 3.3.4, in this case the streaming server stores the multiple versions of the same content with different rate and quality. In addition, each version provides means to randomly switch into it. According to section 3.3.4, this can be accomplished by the use of frequent IDR, or more efficiently by the use of the switching-predictive (SP)-picture concept in H.264/AVC.

However, the availability of multiple options, in this case different forms of pre-encoded content, requires a decision making entity on which options to use. Therefore, the proposed streaming system in Figure 9.1 includes a central unit at the transmitter, referred to as *scheduler*: To make good decisions, this entity should have access to as much and as up-to-date information as possible. Beneficial information may be the importance, deadlines and size of certain media packets as well as the current channel states and the expected bit-rates. Based on this information the scheduler should decide on which data unit to be transmitted next. The scheduler attempts to optimize its decision which packets, as well as which versions, are to be transmitted at each transmission opportunity. The next sections are dedicated to formalize the available and accessible source and channel information as well as how to use this available information in the scheduling.

9.3 A Framework for the Description of Mobile Links and Source Representation

9.3.1 Media Encoding and Abstraction

To provide bit-rate adaptivity for the H.264/AVC pre-encoded streams the features introduced in chapter 3 are applied:

- Receiver playout buffering, such that small bit-rate variations can be compensated.
- Temporal scalability using X-B-B-P-B-B-P-.... structures, whereby the B-pictures are non-referenced and X denotes a switching point. This allows for fast rate adaptation at the sender side.
- Advanced bit-stream switching as introduced in section 3.3.4 to compensated large scale variations. Several versions need to be provided which cover the full range of expected channel conditions.

Each of the encoded media stream is hinted with metadata in the transmitter. The hinting process abstracts M encoded source versions by generating the base quality $\mathcal{Q}_0$, and for each data unit $n = 1, \ldots, N$ and each version $m = 1, \ldots, M$ it generates the following metadata:

- the importance $\mathcal{I}_{n,m}$,
- the data unit size $R_{n,m}$ in bytes,
- the decoding time stamp $\tau_{\mathrm{DTS},n}$, and
- the dependencies expressed by the index of the directly preceding data unit(s) of $\mathcal{P}_n$.

Furthermore, in case that SP-pictures are used for each version m, the data unit size $R_{n,m\to m'}$ of the SSP-picture when switching to version m', and the SI-picture size are required (see chapter 3). This abstract description is used in the following for the optimization of transmission schedules.

9.3.2 Abstract Channel Representation

An accurate mobile channel model as for example presented in subsection 2.4.4 is definitely helpful and necessary to obtain representative simulation results. However, it is obvious that such a model can never be comprehensive, nor can it be assumed that the detailed statistics parameters are known in advance. Nevertheless, it is obviously advantageous and desirable to include channel state information into decisions at the transmitter. Therefore, an abstraction of detailed expected performance of channel to a generic, meaningful and measurable information which is available at the sender unit for optimization purposes is highly desirable.

Specifically for our considered streaming system sufficient information for our scheduling entity may be some *a priori* information on the probability that the channel supports a certain data rate over a certain time interval. More precisely, we ask how likely it is that a certain amount of data has been sent out from the sender buffer within a certain amount of time . As in our case the sender and the receiver buffer are each other's complement and we assume the propagation delay to be negligible, the time the data leaves the sender buffer is equivalent to the time it is available at the receiver. To formalize this notion, we define the event that the channel is able to support some rate r (in bits) within a time interval t as $\mathcal{R}(r,t)$. However, it is not only sufficient to receive a certain rate by some time for the data to be useful at the receiver: Due to the expected dependencies in our media encoded stream it may be necessary that also some preceding data is sent out at the latest at some earlier time. Therefore, we generally require a joint probability distribution, $\Pr\{\bigcup_i \mathcal{R}(r_i,t_i)\,|\xi\}$, on the event $\mathcal{R}(r,t)$ which depends on the individual probability of the joint events, as well as on the current channel state ξ at time τ_a.

The access to an estimate of the single event success probability $\Pr\{\mathcal{R}(r,t)\}$ may be feasible. However, the estimation of the joint probability function is generally far too complex. Still, if we only have access to the single event success probabilities, the joint event success probability can at least be bounded by the product of the single success probabilities and the minimum of the single success probabilities, i.e.,

$$\prod_i \Pr\{\mathcal{R}(r_i,t_i)\} \le \Pr\left\{\bigcup_i \mathcal{R}(r_i,t_i)\right\} \le \min_i \{\Pr\{\mathcal{R}(r_i,t_i)\}\}. \tag{9.1}$$

Therefore, it is desirable to describe mobile channels by single event success probabilities $\Pr\{\mathcal{R}(r_i,t_i)\}$.

9.3.3 Simplified Description for EGPRS-like channels

The exact derivation of the single event success probability distribution for complex channel models is still too complicated and likely without practical relevance, as discussed previously.

Therefore, we attempt to obtain a simplified description for the single event success probability $\Pr\{\mathcal{R}(r,t)\}$ in case of EGPRS-like channels. Despite we will verify the channel model only for the specific system, we are confident that the proposed model is generic and can also be adapted to other mobile systems.

Recall that according to section 2.4.4, the transmission within each single state of the mobile system is represented by $\mathcal{W}_\xi = \{k_{\text{LLP},\xi}, T_{\text{TTI}}, \mathcal{L}_{\text{LLP},\xi}\}$. For the description of the loss process $\mathcal{L}_{\text{LLP}}$ in each state ξ it is sufficient to specify the loss probability $p_{\text{LLP},\xi}$. In the remainder of this chapter we will drop the index "LLP", as the channel model only operates on the link layer.

Let X_ξ be a random variable which describes the amount of data transmitted with a single link layer packet in state ξ, with $X_\xi \in \{0; k_\xi\}$. Furthermore, let $1 - p_\xi$ be the probability of successful packet reception ($X_\xi = k_\xi$), and p_ξ the probability of a packet loss ($X_\xi = 0$). The mean and variance of this process are $m_\xi = k_\xi(1 - p_\xi)$ and $\sigma_\xi^2 = k_\xi^2(1 - p_\xi)p_\xi$, respectively.

As we can expect that the provisioning of feedback and also the retransmissions on the link layer happen quite fast, the respective delay can be neglected. This is especially the case for scenarios where the channel propagation time of one packet is sufficiently smaller than the time interval between two consecutive higher layer data units. Moreover, in delayed feedback systems packet labeling allows reordering of received packets. Therefore, we can assume that the packet transmitted at time instant t will immediately be retransmitted at time instant $t + 1$ if it was lost. Then, for some channel state sequence $\xi_T = \{\xi_1, \ldots, \xi_T\}$, the sum rate $S(\xi_T)$ can be defined as

$$S(\xi_T) \triangleq \sum_{t=1}^{T} X_{\xi_t} = \sum_{\xi=1}^{N_\pi} \omega_\xi X_\xi, \tag{9.2}$$

with ω_ξ the frequency of state ξ in the sequence ξ_T. Note that $S(\xi_T)$ is a random variable itself. For sufficiently large T, it can be assumed that the sum rate $S(\xi_T)$ approaches a normal distribution due to the central limit theorem [Bil95]. In addition, if the frequency ω_ξ for each state is also sufficiently large, the distribution of the normalized sum rate can be characterized as a normal distribution[1], i.e.,

$$\frac{S(\xi_T) - Tm(\xi_T)}{\sigma(\xi_T)\sqrt{T}} \to \mathcal{N}(0,1), \tag{9.3}$$

with normalized mean

$$m(\xi_T) = \frac{1}{T}\sum_{\xi=1}^{N_\pi} \omega_\xi m_\xi \tag{9.4}$$

and normalized variance

$$\sigma^2(\xi_T) = \frac{1}{T}\sum_{\xi=1}^{N_\pi} \omega_\xi \sigma_\xi^2 \tag{9.5}$$

due to the central limit theorem and some extensions [Bil95].

However, in general the state sequence is also random and follows an underlying Markov model. Assuming that the actual state ξ is known, we are interested in the distribution of the sum rate $S_{T|\xi}$ after the transmission attempt of T link layer packets, i.e.,

$$S_{T|\xi} \triangleq \sum_{t=1}^{T} X_{\xi_t|\xi}. \tag{9.6}$$

[1]Throughout this work, $\mathcal{N}(m, \sigma^2)$ will denote the normal distribution with mean m and variance σ^2.

For T sufficiently large, a normal distribution of the sum rate is still justified. However, the derivation of the mean and the variance is not straightforward. Therefore, it is recommended to estimate those parameters $m_{T|\xi}$ and $\sigma^2_{T|\xi}$ depending on the number of link layer packets T and the initial state ξ. If the channel state, is not accessible, we denote the mean as m_T and the variance as σ^2_T. Figure 9.2 shows the normalized mean $m_{T|\xi}/T$ and m_T/T, as well as the normalized variance $\sigma^2_{T|\xi}/T$ and σ^2_T/T for the EGPRS parameters given in table 2.4.4. When comparing the different curves for the two parameters, it is obvious that additional simplifications and modeling might be performed. In a practical system, these parameters should be estimated in advance or are constantly updated during the transmission. In the following we will assume that the parameters $m_{T|\xi}$ and $\sigma^2_{T|\xi}$, or at least some estimate are available to the transmitter.

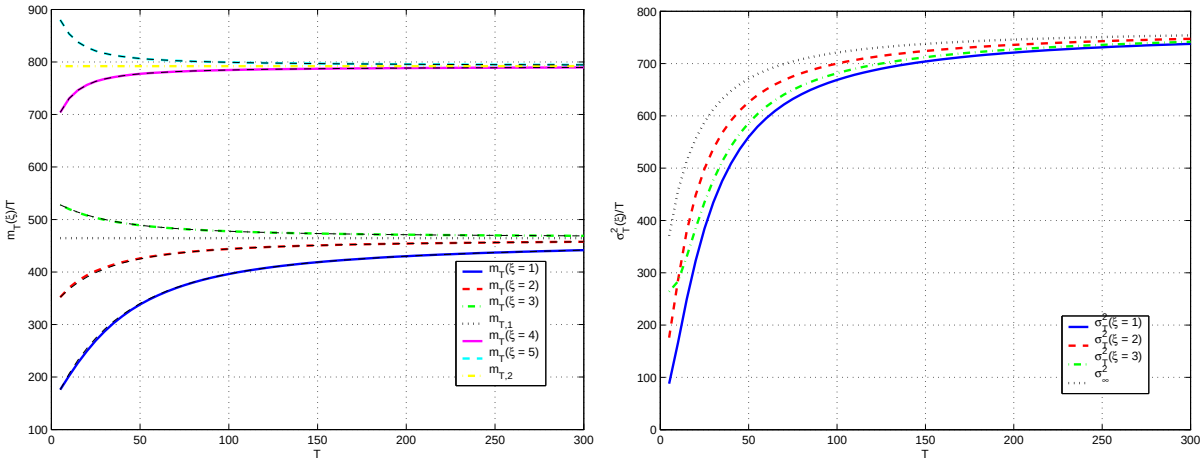


Figure 9.2: Normalized $m_{T|\xi}/T$ and m_T/T, as well as normalized variance $\sigma^2_{T|\xi}/T$ and σ^2_T/T vs. number of link layer packets T.

With knowledge of the mean and the variance for each T (and each initial state ξ), the probability of a certain sum rate is expressed as

$$\Pr\{S_T = s\} \quad = \quad \frac{1}{\sqrt{2\pi\sigma^2_T}} e^{-\frac{(s-m_T)^2}{2\sigma^2_T}}. \tag{9.7}$$

Hence, the single event success probability in case of knowledge of the channel state can be determined as

$$\Pr\left\{\mathcal{R}\left(r,t\right)\right\} = \Pr\{S_{\lfloor t/\tau_3 \rfloor} \geq r\} \quad = \quad \frac{1}{2}\operatorname{erfc}\left(\frac{r - m_{\lfloor t/\tau_3 \rfloor}}{2\sigma_{\lfloor t/\tau_3 \rfloor}}\right). \tag{9.8}$$

For ease of exposition, we will in the following only present the case where the channel state is not known. The extension to the case when the channel state is known, is straightforward.

9.4 Optimized Packet Scheduling and Bitstream Switching

9.4.1 Transmitter Assumptions

We consider a mobile video streaming system as introduced in section 9.2, with a central scheduling unit at the transmitter. The scheduler shall decide at each transmission opportunity which

data unit to transmit next out of the set of available data units, $\{\mathcal{P}_{n,m}\}$, with $n = 1, \ldots, N$ and $m = 1, \ldots, M$. To achieve good user experience, some principles for the selection of data units are obvious:

1. The scheduling algorithm should be able to react to varying channel conditions by bit-stream switching. Only if the channel conditions change too fast, additional reduction of the temporal resolution should be allowed.

2. Data units should be transmitted as close to the time instant as they are due at the receiver. Otherwise, bandwidth is wasted, which might result in expiration and consequently dropping of other earlier data units.

3. Nevertheless, it should be possible to transmit important data units earlier to guarantee their delivery even in bad channel conditions.

4. Version switching should preferably be accomplished with SP-frames rather than with SI-frames.

Previous work on this subject has for example been performed in [KZ02], which is an extension to well-known Early-Deadline First (EDF) scheduling [HKS96]. In [KZ02] the EDF scheduling is extended taking into account frame dependencies. In this work we formalize the concept of frame dependencies and frame importance, extend it to stream switching and introduce schedulers which attempt to optimize the sending order of data units in varying channel conditions. Before we present our proposed algorithm for optimized transmission scheduling and bit-stream switching, it is reasonable to address some assumptions. Those will be helpful for significantly reducing the amount of possible data units to be considered in the optimization process and therefore reduce the complexity of the scheduling process.

- Each data unit $\mathcal{P}_{n,m}$ is only transmitted once since we assume that the lower link layer retransmission protocol clears out all errors. Hence, a loss in our system only happens due to late-arrival at the media client.

- If the transmission of data unit $\mathcal{P}_{n,m}$ in version m has been attempted, all data units at the same position n in the video sequence, which resemble different versions $m' \neq m$, are removed from the set of data units considered for future transmissions.

- It is also assumed that the information on the successful reception or loss of a single data unit is immediately available at the transmitter. As a consequence, a status[2] γ_n can be assigned at the transmitter to each position n in the video sequence. If the transmission of a data unit $\mathcal{P}_{n,m}$ ($m = 1, \ldots, V$) has been attempted, the status takes on one of the following two final values:

  - $\gamma_n =$ ACK, if a data unit at position n is known to be correctly received, and
  - $\gamma_n =$ NAK, if a data unit at position n is known to be lost.

- Positions at which no data unit $\mathcal{P}_{n,m}$ of any version has been transmitted yet are assigned one of the two intermediate status values:

  - $\gamma_n =$ R (for READY), if transmission is possible, since all ancestors $\mathcal{P}_{n',m_{n'}}$ are available at the receiver (i.e., have $\gamma_{n'} =$ ACK).

[2] Note that the status is only indexed with the position n, but not with the version m.

  - γ_n = P (for PENDING) if transmission is not reasonable yet, since there are still some ancestors $\mathcal{P}_{n',m_{n'}}$ missing at the receiver (i.e., which have $\gamma_{n'}$ = R or $\gamma_{n'}$ = P).

- As a consequence, only data units with status γ_n = R are considered for transmission.

- Any data units $\mathcal{P}_{n,m}$ with expired deadline $T_n + \Delta > \tau_a$ (with Δ the initial playout delay and τ_a the actual time at the transmitter) are not transmitted and, together with all of their depending data units, are assigned γ_n = NAK. Note that this procedure is already quite intelligent, as in this case the channel is not blocked with non-useful data for the receiver.

- Switching positions in the video sequence are assigned two status values: one for SI-frames γ_n, and one for SP-frames $\tilde{\gamma}_n$. For SP-frames to be decodable, it is necessary and sufficient that the previous P-frame of any version is available. This does not apply for the case of switching with IDR-Frames only. However, we will only treat the SP/SI case in the following, the IDR-frame concept is a subset of the general framework.

9.4.2 Periodic Update of Side Information at the Transmitter

Any optimized scheduling strategy requires up-to-date side information on the state of the system in the decision process. Therefore, we will introduce the various update steps that need to be performed before each scheduling process starts. Upon initialization, the first position $n = 1$ in the video sequence, as well as all other switching positions which have an SI-frame available, are assigned γ_n = R. All other positions are initialized with γ_n = P. After each successful or non-successful completion of the transmission of a data unit $\mathcal{P}_{n,m_n}$ at actual time τ_a, a new transmission opportunity is available. Before deciding on which data unit to transmit next, the status values at all data unit positions in the transmitter are updated as follows:

1. All data unit positions n' for which the deadline has expired, i.e., where $T_{n'} + \Delta > \tau_a$, are assigned $\gamma_{n'}$ = NAK.

2. If the previous transmission of data unit $\mathcal{P}_{n,m_n}$ was successful, the corresponding status value is changed to γ_n = ACK.

3. If the previous transmission, however, was not successful, the corresponding status value is changed to γ_n = NAK.

4. All data unit positions $\tilde{n}$ for which at least one ancestor $\tilde{n} \prec n'$ has status $\gamma_{n'}$ = NAK are also assigned status $\gamma_{\tilde{n}}$ = NAK.

5. All data unit positions with status $\gamma_{n'}$ = P for which all ancestors have γ_n = ACK are switched to status $\gamma_{n'}$ = R.

6. At switching positions for which all ancestors of the SP-frame are now available at the receiver, the status is changed to $\tilde{\gamma}_n$ = ACK. In this case, either the SI-frame or the SP-frame (depending on the rate) for each version can be selected as a possible candidate for transmission.

After this update procedure has been performed, a new "good" data unit with γ_n = R is selected for transmission by the scheduler as described in the next subsection.

9.4.3 The Scheduling Process

The task of the scheduler is to determine an optimal transmission order and version of the data unit to be sent next. The data unit should be selected to maximize the expected overall user perception, expressed by some quality criterion. In particular, the scheduler decides at each transmission opportunity

- which data unit to transmit next,
- and in case of an SP-, SI-, or I-picture, which version to transmit next.

During this decision process the scheduler should take into account as much side information as possible, namely:

- the currently expected channel behavior,
- the actual time τ_a,
- the deadlines, the importance, the different versions of the data units, and
- the up-to-date status of different data units.

As already mentioned, the scheduler might decide to transmit more "important" data units earlier to guarantee their timely delivery with high probability, whereas other data units with very low importance might not be transmitted at all. Hence, we express the actual delivery order by the transmission schedule $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$, where π_k holds the index of the data unit (i.e., the position in the video sequence) to be transmitted at (temporal) position k. Furthermore, for each element in the transmission schedule, a version m_{π_k} is selected[3].

We propose to select the next data unit for transmission based on some utility function, for which we use the expected quality at the receiver $\mathcal{Q}(\boldsymbol{\pi}, \boldsymbol{m})$. This metric generally depends on all relevant source and channel information, such as rates, deadlines, importance vectors, etc. More specifically, the optimized schedule $\boldsymbol{\pi}_{\text{opt}}$ and the optimized version vector $\boldsymbol{m}_{\text{opt}}$ satisfy

$$(\boldsymbol{\pi}_{\text{opt}}, \boldsymbol{m}_{\text{opt}}) = \arg\max_{(\boldsymbol{\pi}, \boldsymbol{m})} \tilde{\mathcal{Q}}(\boldsymbol{\pi}, \boldsymbol{m}), \tag{9.9}$$

with

$$\tilde{\mathcal{Q}}(\boldsymbol{\pi}, \boldsymbol{m}) = \mathcal{Q}_0 + \sum_{\substack{n=1 \\ \gamma_n = \text{ACK}}}^{N} \mathcal{I}_{n,m_n} + \sum_{\substack{n=1 \\ \gamma_n = \text{NAK}}}^{N} 0 + \sum_{\substack{n=1 \\ \gamma_n \in \{\text{R,P}\}}}^{N} \mathcal{I}_{n,m_n} \tilde{P}_n(\boldsymbol{\pi}, \boldsymbol{m}) \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \tilde{P}_m(\boldsymbol{\pi}, \boldsymbol{m}).$$

Here, $\tilde{P}_n(\boldsymbol{\pi}, \boldsymbol{m})$ expresses the probability that data unit $\mathcal{P}_{n,m_n}$ will be received in time for this selection of $\boldsymbol{\pi}$ and $\boldsymbol{m}$. Note that for data units with $\gamma_n = \text{ACK}$, this probability is equal to 1, while for $\gamma_n = \text{NAK}$ it is equal to 0. In case $\gamma_n \in \{\text{R,P}\}$, the probability depends on the sum rate of the scheduled data units, the delivery deadline $T_n + \Delta$, the actual time τ_a, and the channel statistics $\mathcal{R}(r, t)$, and can be written as

$$\tilde{P}_n(\boldsymbol{\pi}, \boldsymbol{m}) = \Pr\left\{ \mathcal{R}\left(\sum_{k=1}^{\rho_n} R_{\pi_k, m_{\pi_k}}, T_n + \Delta - \tau_a \right) \middle| \bigcup_{\substack{m \prec n \\ \gamma_m \in \{\text{R,P}\}}} \mathcal{R}\left(\sum_{k=1}^{\rho_m} R_{\pi_k, m_{\pi_k}}, T_m + \Delta - \tau_a \right), \boldsymbol{\pi}, \boldsymbol{m} \right\}, \tag{9.10}$$

[3]The version vector $\mathbf{m} = \{m_n\}_{n=1}^{N}$ is ordered with respect to the position of the data units in the video sequence.

where ρ_n defines the (temporal) position of data unit $\mathcal{P}_{n,m_n}$ in the schedule $\boldsymbol{\pi}$.

Hence, when determining the expected quality according to (9.9), we acknowledge the fact that due to the dependencies in the video sequence not only the actual data unit must have been received, but also all of its predecessors. The above notation can be simplified by using the joint probability $P_n(\boldsymbol{\pi}, \boldsymbol{m})$ instead of the conditional probability $\tilde{P}_n(\boldsymbol{\pi}, \boldsymbol{m})$, i.e.,

$$P_n(\boldsymbol{\pi}, \boldsymbol{m}) = \Pr\left\{ \bigcup_{\substack{m \prec n \\ \gamma_m \in \{\mathrm{R,P}\}}} \mathcal{R}\left(\sum_{k=1}^{\rho_m} R_{\pi_k, m_{\pi_k}}, T_m + \Delta - \tau_a \right) \middle| \boldsymbol{\pi}, \boldsymbol{m} \right\}. \tag{9.11}$$

The optimal transmission schedule and version vector now have to satisfy

$$(\boldsymbol{\pi}_{\text{opt}}, \boldsymbol{m}_{\text{opt}}) = \arg\max_{(\boldsymbol{\pi}, \boldsymbol{m})} \mathcal{Q}(\boldsymbol{\pi}, \boldsymbol{m}), \tag{9.12}$$

with

$$\mathcal{Q}(\boldsymbol{\pi}, \boldsymbol{m}) \;\triangleq\; \sum_{\substack{n=1 \\ \gamma_n \in \{\mathrm{R,P}\}}}^{N} \mathcal{I}_{n,m_n} P_n(\boldsymbol{\pi}, \boldsymbol{m}). \tag{9.13}$$

Note that in (9.13), we have already considered the fact that only data units with status $\gamma_n = \mathrm{R}$ or $\gamma_n = \mathrm{P}$ can be part of the schedule $\boldsymbol{\pi}$. The version vector $\boldsymbol{m}$ remains identical to the previous definition.

With these preliminaries, the scheduler now operates by repeatedly executing the following steps, until there are no more ready or pending data units at the transmitter:

1. After successful or non-successful completion of the transmission of a data unit, the status of the data units in the transmission set is updated according to subsection 9.4.2.
2. Then, by combining this updated status information with (possibly new) channel state information, the scheduler determines $(\boldsymbol{\pi}_{\text{opt}}, \boldsymbol{m}_{\text{opt}})$ according to (9.12).
3. Finally, transmission of data unit $\mathcal{P}_{\pi_{\text{opt},1}, m_{\text{opt},\pi_{\text{opt},1}}}$ is initiated.

9.4.4 Implementation Aspects and Complexity Reduction

The number of possible combinations the scheduler has to compare in (9.12) is huge, since exchange of a single element in $\boldsymbol{\pi}$ even at some later position generally influences the expected quality and thus the selection of the next data unit. To find the optimized schedule and version vector, in principle, a brute-force search is necessary. Since this is far from being practically feasible, complexity reductions are essential. In the following we will discuss some simplifications first before we present our optimized scheduling algorithm.

Weighted Cumulative Importance

A major problem in the computation of the expected quality for a certain combination $\{\boldsymbol{\pi}, \boldsymbol{m}\}$ originates from the fact that all later parts in the transmission schedule can be re-ordered in many ways. However, only the data units with status $\gamma_n = \mathrm{R}$ are candidates for transmission, and typically this set is very small. We only want to find the best transmission order and version for

them at the actual transmission and scheduling opportunity. While the aforementioned later parts are not used, any modifications there usually affect the decision of the data unit to transmit next (i.e., the one we are actually interested in). Hence, we still have to consider the influence of the transmission of current data units on later frames: It is definitely not sufficient to replace the entire set of possible transmission schedules and version vectors in (9.12) by a set which only includes data units with $\gamma_n = \mathrm{R}$.

We take these dependencies into account by introducing the weighted cumulative importance of a certain data unit $\mathcal{P}_{n,m_n}$. For a given transmission schedule $\boldsymbol{\pi}$ and version vector $\boldsymbol{m}$ the weighted cumulative importance $\tilde{\mathcal{I}}_n(\boldsymbol{\pi}, \boldsymbol{m})$ is recursively defined as

$$\tilde{\mathcal{I}}_n(\boldsymbol{\pi}, \boldsymbol{m}) \triangleq \mathcal{I}_{n,m_n} + \sum_{\{k|\gamma_k=\mathrm{P} \wedge n \prec\prec k\}} w_k \tilde{\mathcal{I}}_k(\boldsymbol{\pi}, \boldsymbol{m})\, P_k(\boldsymbol{\pi}, \boldsymbol{m}). \tag{9.14}$$

Here, $n \prec\prec k$ means direct dependency (i.e., n is a direct ancestor of k), and w_n denotes the weight of a data unit at position n in the video sequence. In case of P-frames and B-frames we define for the sake of simplicity

$$w_n \triangleq \begin{cases} 1 & \text{if } n \text{ is a P-frame or SP-frame,} \\ 1/2 & \text{if } n \text{ is a B-frame.} \end{cases}. \tag{9.15}$$

This weighted cumulative importance has to be re-computed at each scheduling opportunity for all data units in the transmission set with $\gamma_n \in \{\mathrm{R,P}\}$.

Assume now that the version vector $\boldsymbol{m}$ is fixed. Let $\mathcal{S} = \{n|\gamma_n = \mathrm{R}\}$ be the set containing all data unit positions with status $\gamma_n = \mathrm{R}$. Correspondingly, $\overline{\mathcal{S}} = \{n|\gamma_n = \mathrm{P}\}$ contains all remaining data unit positions with $\gamma_n = \mathrm{P}$, which are still waiting for transmission. In the following we will only investigate schedules, which have elements from $\mathcal{S}$ in the first positions, before any remaining ones from $\overline{\mathcal{S}}$ are added. Such a schedule is denoted as $\{\boldsymbol{\pi}(\mathcal{S}), \boldsymbol{\pi}(\overline{\mathcal{S}})\}$. Furthermore, let us define the sum rate of all data units in $\mathcal{S}$ as

$$R_{\mathcal{S}} \triangleq \sum_{n \in \mathcal{S}} R_{n,m_n}. \tag{9.16}$$

Thus, the weighted cumulative importance according to (9.14) can be approximated by

$$\tilde{\mathcal{I}}_n\left(\mathcal{S}, \boldsymbol{\pi}(\overline{\mathcal{S}}), \boldsymbol{m}\right) = \mathcal{I}_{n,m_n} + \sum_{\substack{k \in \overline{\mathcal{S}} \\ n \prec k}} w_k \mathcal{I}_{k,m_k} \Pr\left\{ \bigcup_{\substack{j \in \overline{\mathcal{S}} \\ j \prec k}} \mathcal{R}\left(R_{\mathcal{S}} + \sum_{i=1}^{\rho_j(\overline{\mathcal{S}})} R_{\pi_i}, T_j + \Delta - \tau_a \right) \right\}, \tag{9.17}$$

with $\rho_j(\overline{\mathcal{S}})$ the (temporal) position of data unit $\mathcal{P}_{j,m_j}$ in $\boldsymbol{\pi}(\overline{\mathcal{S}})$.

To simplify the computation of the weighted cumulative importance, we apply the upper and lower bounds according to (9.1). Let us define

$$P_j(r, \boldsymbol{\pi}(\mathcal{S})) \triangleq \Pr\left\{ \mathcal{R}\left(r + \sum_{i=1}^{\rho_j(\mathcal{S})} R_{\pi_i}, T_j + \Delta - \tau_a \right) \right\}. \tag{9.18}$$

Hence, a lower bound on the weighted cumulative importance in (9.14) is obtained as

$$\underline{\mathcal{I}}_n\left(\mathcal{S}, \boldsymbol{\pi}(\overline{\mathcal{S}}), \boldsymbol{m}\right) = \mathcal{I}_{n,m_n} + \sum_{\substack{k \in \overline{\mathcal{S}} \\ n \prec\prec k}} w_k \underline{\mathcal{I}}_k\left(\mathcal{S}, \boldsymbol{\pi}(\overline{\mathcal{S}})\right) P_k(R_{\mathcal{S}}, \boldsymbol{\pi}(\overline{\mathcal{S}})), \tag{9.19}$$

which results in the following lower bound on the expected quality in (9.13)

$$\underline{\mathcal{Q}}(\{\boldsymbol{\pi}(\mathcal{S}),\boldsymbol{\pi}(\overline{\mathcal{S}})\},\boldsymbol{m}) \;=\; \sum_{n\in\mathcal{S}} \underline{\mathcal{I}}_n\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right) P_n(0,\boldsymbol{\pi}(\mathcal{S})). \tag{9.20}$$

To obtain an upper bound on the expected quality in (9.13) let us recursively define a local success probability for all $k \in \overline{\mathcal{S}}$ as

$$\hat{P}_k \triangleq \min\{P_k(R_{\mathcal{S}},\boldsymbol{\pi}(\overline{\mathcal{S}})), \forall_{m\in\overline{\mathcal{S}}|m\prec\prec k}\hat{P}_m\}. \tag{9.21}$$

Hence, an upper bound on the cumulative weighted importance in (9.14) can be given for any $k \in \overline{\mathcal{S}}$

$$\overline{\mathcal{I}}_k\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right) = \mathcal{I}_{k,m_k}\hat{P}_k + \sum_{\substack{m\in\overline{\mathcal{S}}\\ k\prec\prec m}} \overline{\mathcal{I}}_m\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right), \tag{9.22}$$

which results in the following upper bound on the expected quality in (9.13)

$$\overline{\mathcal{Q}}(\boldsymbol{\pi}(\mathcal{S}),\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}) \;=\; \sum_{n\in\mathcal{S}} \left[\mathcal{I}_{n,m_n}P_n(0,\boldsymbol{\pi}(\mathcal{S})) + \sum_{\substack{k\in\overline{\mathcal{S}}\\ n\prec\prec k}} \overline{\mathcal{I}}_k\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right)\right]. \tag{9.23}$$

A third method to at least estimate the quality using the weighted cumulative importance is to ignore the joint channel events. The corresponding estimate of the weighted cumulative importance results for all $n \in \mathcal{S}$ in

$$\hat{\mathcal{I}}_n\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right) \triangleq \sum_{\substack{k\in\overline{\mathcal{S}}\\ n\prec k}} \mathcal{I}_{k,m_k}P_k(R_{\mathcal{S}},\boldsymbol{\pi}(\overline{\mathcal{S}})), \tag{9.24}$$

which yields the following estimate on the expected quality in (9.13)

$$\hat{\mathcal{Q}}(\boldsymbol{\pi}(\mathcal{S}),\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}) \;=\; \sum_{n\in\mathcal{S}} \mathcal{I}_{n,m_n}P_n(0,\boldsymbol{\pi}(\mathcal{S})) + \hat{\mathcal{I}}_n\left(\mathcal{S},\boldsymbol{\pi}(\overline{\mathcal{S}}),\boldsymbol{m}\right). \tag{9.25}$$

The advantage of using a fixed later schedule $\boldsymbol{\pi}(\overline{\mathcal{S}})$ and the notion of cumulative weighted importance is that the latter can be pre-computed before different transmission schedules $\boldsymbol{\pi}(\mathcal{S})$ are tested. The specific implementation of this computation is not discussed in further detail here, but basically in most cases the computation starts in the leaves of the dependency graph and moves backwards to the data unit positions with status $\gamma_n = \text{R}$. Although none of the proposed modifications are optimal anymore due to the fixing of the later data unit positions in $\boldsymbol{\pi}(\overline{\mathcal{S}})$, they significantly reduce the complexity, especially, if the set $\mathcal{S}$ is kept small.

The number of selected data units for the set $\mathcal{S}$ is referred to as *Look-Ahead Units* and is denoted by $N_{\mathcal{P}}$ in the following. For the selection procedure, we have used two different modes:

1. In the so-called *ready selection* mode, we choose the first $N_{\mathcal{P}}$ data unit positions with status $\gamma_n = \text{R}$.

2. In the so-called *sequential selection*, we choose the next $N_{\mathcal{P}}$ data unit positions, regardless of whether their status is $\gamma_n = \text{R}$ or $\gamma_n = \text{P}$. In this case, the presented algorithm has to be modified slightly, as the status of data units changes due to scheduling. It is required to introduce an internal variable which keeps track of the status during the scheduling process.

Nevertheless, for both selection modes, the fixed part of the schedule $\boldsymbol{\pi}(\overline{\mathcal{S}})$ is sequentially filled with the remaining data units in the order of their index n.

These definitions and simplifications can be combined in order to develop a multi-stage scheduling algorithm for video streaming over VBR links. The respective flow diagram is depicted in Figure 9.3, and the various elements will be explained in the next subsections.

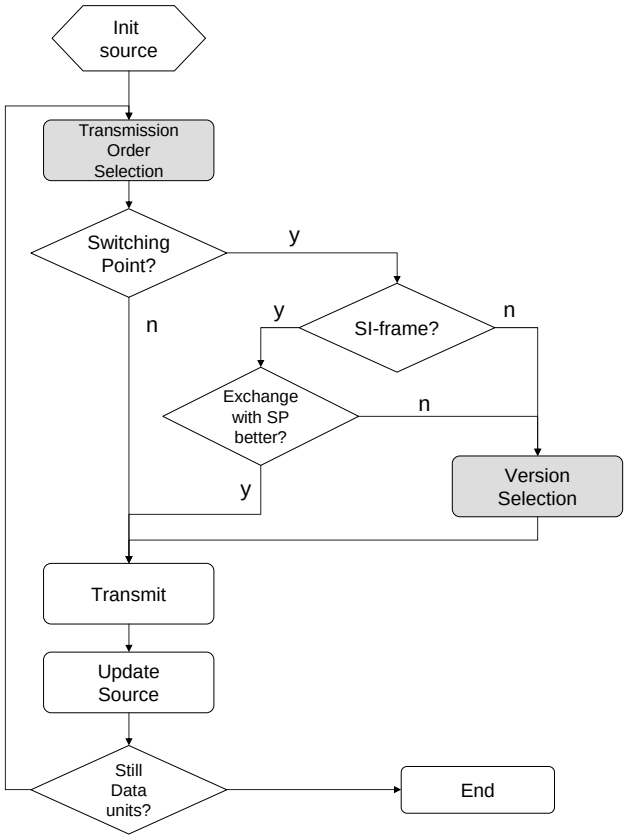


Figure 9.3: Flow diagram of the scheduling algorithm.

Separation of Transmission Order and Version Selection

Based on one of the quality metrics in (9.20), (9.23), and (9.25) we select the best transmission order $\boldsymbol{\pi}(\mathcal{S})$ for each scheduling opportunity. During this process we always operate within the same version, i.e., if a switching point is included in $\boldsymbol{\pi}(\mathcal{S})$, the same version as used for the previous group-of-pictures (GOP) is selected after the switching point. The case of version switching is discussed separately below.

Note that despite fixing the schedule $\boldsymbol{\pi}(\overline{\mathcal{S}})$ and only using a single version vector, all data unit positions up to the end N would need to be considered for the computation of the weighted cumulative importance. However, for IDR-frame switching it is easily shown that it is sufficient to only consider all data unit positions up to the next non-transmitted IDR-frame position. Though this is not completely accurate in case of SP/SI-frames for switching, we neglect this and use the same strategy for transmission order selection.

Data Unit Selection

In case a non-switching position n is selected, the version m_n of the data unit is unambiguous, as exactly for one version the predecessors are available at the decoder. The corresponding data unit $\mathcal{P}_{n,m_n}$ is then transmitted. In case a switching position is selected, the corresponding data unit, however, is not immediately transmitted. If the switching position requires an SP-frame or an IDR-frame, the version selection procedure as presented below is immediately invoked. If the switching position requires an SI-frame, an alternative proposal is made: The selected SI-frame is replaced with the corresponding SP-frame. If this proposal yields a better metric than the SI-frame, the alternative is selected. If this is not the case, the version selection below is invoked.

Version Selection

In addition to the local decisions on the transmission order, whenever a switching data unit position is due for transmission, the (possibly new) version is selected based on the following principles: Only a single data unit position, namely the switching position, is included in $\mathcal{S}$. However, instead of fixing the version vector $\boldsymbol{m}$, we evaluate the quality for all possible versions at this position. Our proposed algorithm allows to take into account not only the version selection of the next switching point, but a total of N_s switching points. Note that between two switching points the version is fixed for all data units to the version of the preceding switching point. Recursive computation of the quality in (9.20), (9.23), or (9.25) is then applied using the weighted cumulative importance. The number of considered switching points N_s is also referred to as *Look-Ahead Switching Points*.

Obviously, the more switching points N_s we take into consideration, the more complex the algorithm gets, but the performance also increases as more future data is taken into account for the decisions. In the next section, we will evaluate the performance with different system parameters and scheduling options for our proposed scenario and optimization strategy.

9.5 Experimental Results

9.5.1 Simulation Parameters

An exemplary set of simulations has been carried out using the following parameters: We have encoded $M = 4$ versions of a QCIF sequence of length $N = 2698$ frames with alternating speakers and sport scenes using H.264/AVC test model software JM8.2. The sequence has a length of about 90 seconds and contains sufficiently diverse content to yield representative results. We used a single QP for each version, namely[4] $q = 28, 32, 36, 40$, and a common frame rate of 30 fps, without any additional rate control algorithm. A GOP structure with X-B-B-P-B-B-P...-B-X whereby X represents a switching picture, i.e., an SP or IDR picture. The distance between two switching points is 1 s. The SP-pictures have ”IDR” property in the sense that referencing over SP-pictures is not permitted. In addition, we provide both, SSP- and SI-pictures, for switching such that it is ensured that a switching point can be used for both, switching and random access. The generated video sequences have been hinted and for all versions we provided the hinting data. The initial playout delay at the streaming client is set to be $\Delta = 1.5$ s. For all our simulations we use the framework as introduced in section 3.6 which includes a freeze-frame error concealment as soon as one packet in the dependency chain is lost.

[4] JM8.2. already has adjusted the QP to the value of the final standard.

The mobile link is modeled using the EGPRS channel model according to [CCCQ00] and section 2.4.4. We restrict ourselves to the more challenging scenarios with 8 and 15 users per cell according to Table 2.4.4. Changes among the two groups of channel states may happen statistically independent every 20 seconds. Each simulation point represents the algebraic mean over 200 independent channel realizations, hence the transmission of 300 minutes or 5 hours of video.

As reference system, we also encoded the same video sequence with the rate control provided in JM8.2 to obtain two single-rate bit-streams, one with 69 kbit/s and one with 96 kbit/s. Those two where used as they match quite well the expected channel transmission rates of the two groups. The same GOP structure is used, but I-pictures have been applied instead of SP-pictures. We also investigate the case where we only apply SI-frames, such that the performance is actually similar to IDR-picture switching.

In the following we will compare the achievable performance for

- different scheduler settings (i.e., in terms of different number of Look Ahead Units and Look Ahead Switching Points),
- different selection modes (i.e., sequential and ready selection mode),
- different methods to compute the weighted importance (i.e., lower bound, upper bound, and linear combination).

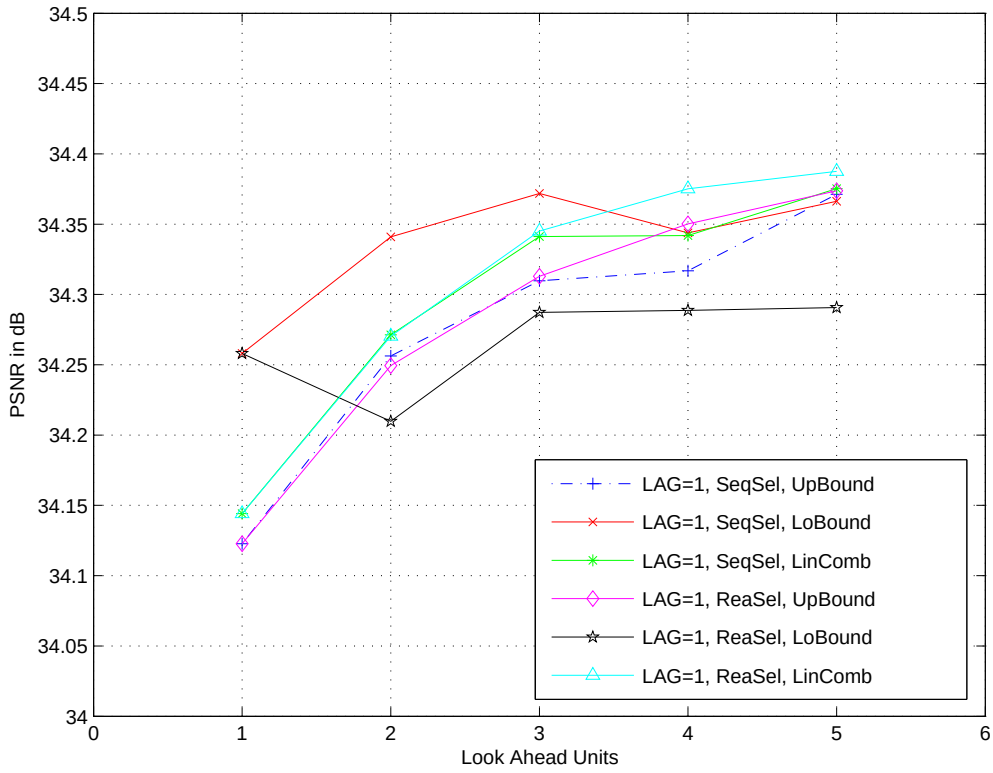


Figure 9.4: Average PSNR, $\overline{\mathrm{PSNR}}$, over Look-Ahead Units, $N_{\mathcal{P}}$, for one Look-Ahead Switching Points, N_s, and all selection modes and combining methods.

9.5.2 Local Decisions - Temporal Scalability

First, we will investigate the performance of the scheduling algorithm when making local decisions on the actual transmission order, as well as on which frames are to be dropped (temporal scalability). Bitstream switching is not considered in this first set of simulations. Figure 9.4 shows the average PSNR over the number of Look-Ahead Units $N_{\mathcal{P}}$ for one Look-Ahead Switching points, N_s, and all selection modes and combining methods. The channel with 8 users per cell has been used and the video encoded with CBR 96 kbit/s. As a general observation it can be concluded that the achievable quality increases with larger scheduling sets. Note that scheduling set size of one is identical to an EDF scheduling and compares our system to state-of-the-art performance [HKS96, KZ02]. The use optimized local decisions can not really provide significant gains in performance, the PSNR variations are in a range of at most 0.3 dB and are therefore rather marginal.

The inconsistencies for the *ready selection* mode for $N_{\mathcal{P}} = 2$ can be explained by the following additional observation from our simulations: This method tends to use too often not the next ready data unit, but the second next one in the set, which has locally higher weighted cumulative importance. Whereas this locally "optimal" decision provides the best metrics in this case, it leads to higher data data unit drop rates at the sender in the long term.

Since larger values of $N_{\mathcal{P}}$ also increase the complexity of the algorithm, we conclude from the results in Figure 9.4 that $N_{\mathcal{P}} = 3$ seems to be a good compromise for all scheduler options considered. We apply this setting for the remainder of the work unless mentioned otherwise.

9.5.3 Influence of Scheduler Options

In the next set of simulations, we introduce stream switching. In a first set of experiments, the influence of scheduler options is of interest. In general, both the selection mode and the combining method used in the scheduler have an influence on the overall performance when stream-switching is enabled. Therefore, Figure 9.5 shows the average PSNR, $\overline{\mathrm{PSNR}}$ over the number of Look-Ahead Switching Points N_s for different selection modes and combining methods (UpBound according to (9.23), LoBound according to (9.20), and LinCom according to (9.25)) and an EGPRS channel with 8 users. Obviously, if more switching points are taken into account, the performance increases for all strategies. Compared to not looking ahead (i.e., streaming the data as is and only allowing local decisions), a gain of up to 1.4 dB is now achievable.

While all strategies yield some gain, we have found out in a long series of tests that a scheduler using sequential selection with upper bound combining achieves the best and most consistent results. Therefore, we apply this mode in the remainder of this work.

9.5.4 System Performance

Now we will compare systems which use different streaming technologies, namely

- Constant Bit–Rate (CBR) streaming as a reference system,
- streaming with optional bit-stream switching using SI-pictures,
- streaming with optional bit-stream switching using SP-pictures.

In all cases, we add smart dropping in the sense that the transmitter is aware of expired deadlines at the receiver and does not attempt to transmit this data. This setup is quite suitable to show the potential performance gains achievable with stream switching.

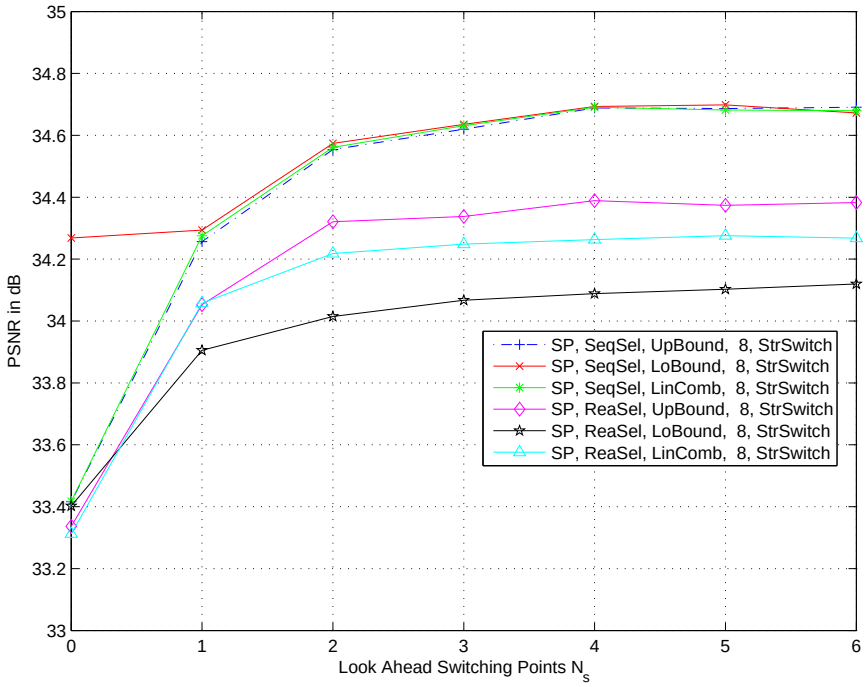


Figure 9.5: PSNR over Look-Ahead Switching Points N_s and all selection modes and combining methods (UpBound according to (9.23), LoBound according to (9.20), and LinCom according to (9.25)).

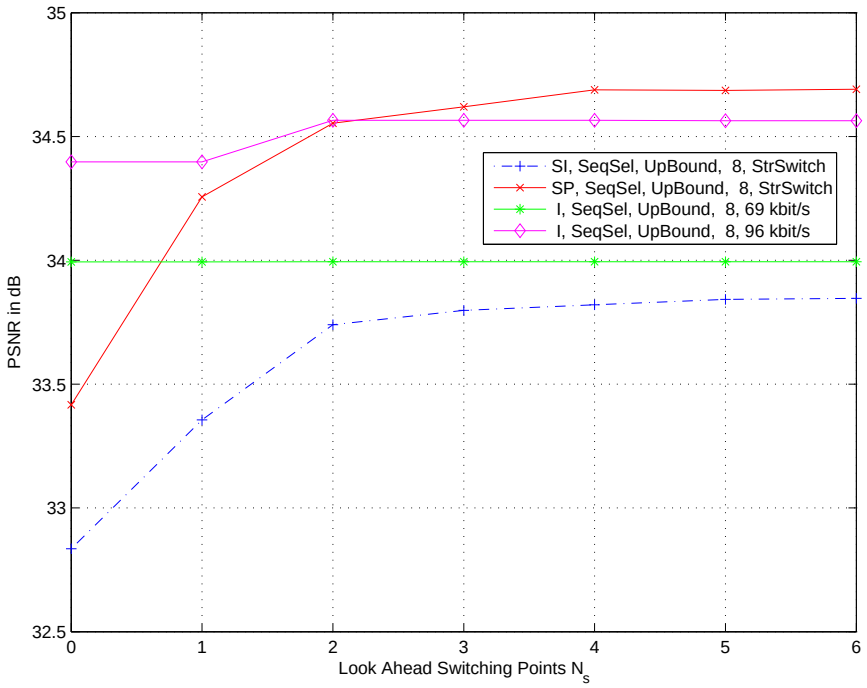


Figure 9.6: Average PSNR over Look-Ahead Switching Points N_s for different encoding strategies and a channel with 8 users.

Figure 9.6 shows the average PSNR over Look-Ahead Switching Points N_s for different encoding strategies and an EGPRS channel with 8 users. As can be observed, SP-picture switching clearly outperforms SI-picture switching by about 0.8 dB in PSNR. The gains are similar, if IDR-pictures would be used instead of SI-pictures (not shown here for the sake of conciseness). It is interesting to note that even in case of constant bit-rate streaming, the use of a scheduler at the transmitter provides some gains due to better local decisions, which result in larger scale temporal scalability. Furthermore, for the chosen scenario with only 8 users, bit-stream switching only outperforms CBR streaming, if at the scheduler at least two Look-Ahead Switching Points are considered.

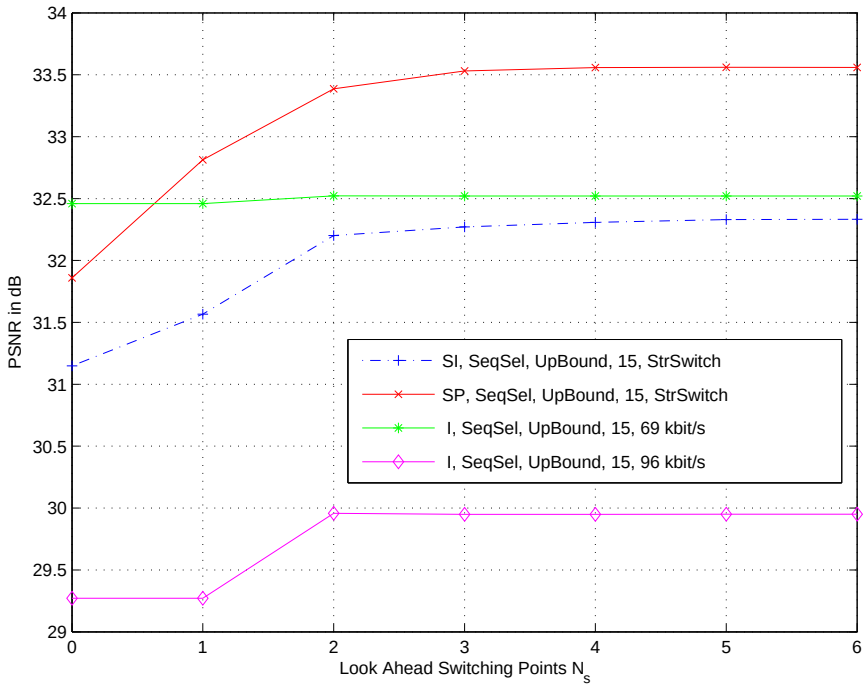


Figure 9.7: Average PSNR, $\overline{\text{PSNR}}$ over Look-Ahead Switching Points N_s for different encoding strategies and a channel with 15 users.

If we increase the system load at the transmitter by changing to a scenario with 15 users, the situation is different as shown in Figure 9.7: In this case the CBR stream with 96 kbit/s does not lead to an acceptable quality any more, and the 69 kbit/s stream seems more appropriate. The system which allows switching among 4 different streams, however, yields constant good quality and the overall degradation compared to the previous system with 8 users is only about 1 dB.

To provide further insight on what exactly happens in the system, Figure 9.8 depicts the respective data unit drop rates. If we allow a reasonable amount of Look-Ahead Switching Points to appropriately schedule the transmission for the future, the data unit drop rates significantly decreases in case bit-stream switching is used. As a consequence, the objective performance (in terms of PSNR) and also the subjective performance (in terms of viewer satisfaction) are largely enhanced.

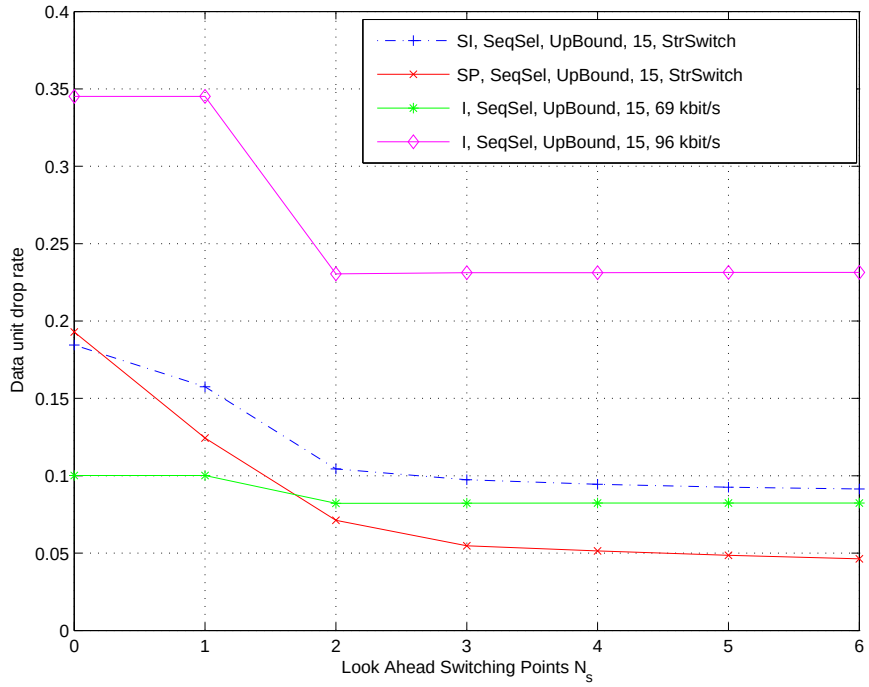


Figure 9.8: Data unit drop rate over Look-Ahead Switching Points N_s for different encoding strategies and a channel with 15 users.

9.6 Summary

In this chapter streaming methods for variable bit-rate mobile channels have been introduced and investigated in detail. The following observations and findings are of major importance:

- Online encoding or transcoding is generally too complex and not feasible for streaming servers which commonly have to serve several or many connections at the same time. Means for video rate adaptation for pre-encoded content are necessary.

- We have shown in [SJK04] by probabilistic statements that for UMTS-like channels with power control and retransmissions the bit-rate variations due to link layer retransmissions can be well compensated by receiver buffering without adding signicant additional delay. Playout buffering is an excellent mean to compensate smaller variations of bit-rates.

- In case of more heterogeneous reception conditions, playout buffering is not sufficient and the pre-encoded content needs to be modified by packet dropping and/or by bit-stream switching.

- We have proposed a video streaming system which relies on three major principles: receiver playout buffering, temporal scalability and advanced bit-stream switching. The objective of this chapter has been to understand the performance and potentials of rate adaptation features for variable bit-rate switching in realistic scenarios.

- A decision making entity, referred to as scheduler, is introduced at the transmitter. The scheduler should make optimized decisions taking into account as much and as up-to-date

information as possible. Beneficial information for the scheduler is the importance, deadlines and size of certain media packets as well as the current channel states and the expected bit-rates. Our scheduler decides for each transmission opportunity which data unit to be transmitted next.

- We have introduced a formalized description of the available and necessary source and channel information. The source information abstraction is based on section 3.6. For the channel abstraction the distribution of the supported bit-rate over some period of time is well approximated by a normal distribution.
- To optimize its decisions at the transmitter, the scheduler maintains the state of all data units during the transmission session. The status of individual data units may indicated acknowledged, lost, ready for transmission, or still pending. To generate this status, successful or unsuccessful delivery, dependencies and deadlines are taken into account. The status is updated frequently with the reception of any new information.
- The actual scheduling process decides at each transmission opportunity, which data unit to transmit next, possibly out of decoding order, and in case of an SP-, SI-, or I-picture, which version to transmit next. To make good decisions, the scheduler performs its selection such that the expected quality is maximized by this decision.
- The number of possible combinations the scheduler would have to compare in a brute–force implementation is completely intractable. For this purpose we have introduced a set of complexity reductions by fixing the transmission schedule for later data units and only optimizing the schedule locally. This allows to pre-compute a weighted-cumulative importance for all data units that are candidates for transmission. Further complexity reduction means are introduced and an implementation of the scheduler is provided. The computational complexity can be traded versus performance.
- Experimental results have been provided for an EGPRS-like channel using playout buffering, temporal scalability, as well as bit-stream switching based on the H.264 SP-picture concept.
- It is verified that by the use of optimized scheduling, temporal scalability can be optimized. However, the use of optimized local decisions can not really provide signicant gains in performance, the PSNR improvements are in a range of at most 0.3 dB and are therefore rather marginal. Only little complexity taking into account 3 candidates for transmission is considered to be sufficient.
- We investigated different schemes and found out in a long series of tests that a scheduler using sequential selection of data units with upper bound combining on the channel probability achieves the best and most consistent results.
- A system with 4 available streams has been evaluated for EGPRS-like transmission. By the use of SP-frame switching and optimized scheduling, the transmission of CBR encoded video without switching can be outperformed with moderate gains for medium system loads. Significant gains are observed in case the system load is higher, for example for EGPRS with 15 users in one sector.
- We emphasize that optimization procedures such as the scheduler as proposed in here are necessary within the transmitter to exploit the potentials of temporal scalability and bit-stream switching in realistic mobile transmission environments.

10

Summary and Outlook

This work has investigated system designs and optimizations for the application of video services in mobile communication systems. Special focus has been put on the cooperation and optimization of functions in different layers of a transmission system taking into account the service requirements of different video applications. Based on channel models that have been derived from state-of-the-art and emerging mobile communication systems, we have introduced suitable transmission and video coding methods. In particular, the H.264/AVC video coding standard as well as the Progressive Texture Video Coding (PTVC), a proprietary scalable extension of H264/AVC, have been intensively analyzed for their applicability in mobile and Internet communication environments. In doing so, innovative error protection tools, for example the Far End Error Decoding (FEED), have been designed which enable adaptation to the varying reception conditions in mobile communication environments. Furthermore, these tools enable cross-layer optimization with the proposed video coding tools. The selection of options for video coding and error protection as well as their cooperation has been supported by the development of quality-optimizing selection and rate allocation schemes. The work has been further supported by information-theoretic justifications, practically relevant system designs, as well as extensive simulation results. The provided data has shown the benefits of the proposed methods in comparison to existing systems and has quantified the expected gains. In the following, we will summarize the major findings of this work; the major own contributions are highlighted in *italics*.

Summary and Major Findings: Source and Video Coding Toolbox

In chapter 3, source and video coding tools have been introduced. The major contributions and findings can be summarized as follows:

- In order to ease integration of source coders in networks, it is desirable that source coders are rate-adaptive, i.e., can support the generation of different quality-rate levels. Progressive source coders that generate a bit-stream containing several or many quality-levels are advantageous. Especially for video codecs in network environments, both compression efficiency and network integration are of major importance. We have introduced several *H.264/AVC network integration and error resilience features* that have partly been developed in the course of this work and have been adopted as part of the H.264/AVC standard. However,

the specifications of video codecs in H.264/AVC and the integration in mobile communication standards leaves significant flexibility in efficient deployments: Encoder settings, e.g., bit rate control, error resilience, or motion estimation processes, are generally left open in standardization work and are therefore investigated in this work.

- Bitrate adaptivity is an important feature for the use of video codecs in mobile communication environments. For small scale variations, playout buffering or temporal scalability can be used. However, to adapt to larger scale variations, bit-stream switching is a preferable solution. In particular, the Switching–Predictive (SP)-picture concept of H.264 provides an efficient realization for bit-stream switching. We have implemented *an optimized SP-picture concept specifically for bit-stream switching*. Optimizations for encoding the primary and especially the secondary representations have been proposed and verified.

- Scalable video coding is even more suitable for easy network integration, but the MCP process in hybrid video codecs does not allow scalability without sacrificing compression efficiency. Scalable/progressive video codecs are in general not *good*. However, we have *designed and developed the Progressive Texture Video Coding (PTVC)*, a scalable extension of an interim version of the H.264 test model. The PTVC provides a breakthrough in progressive video coding. By the use of drift-compensating means progressively coded video streams can be generated in order to ease network integration. The rate–distortion performance of different video codecs is also determined by the encoder implementation. Enhancements for encoder implementations have been introduced and the performance of different video codecs has been compared.

- For the purpose of rate–distortion optimization in network environments, a *formalized description for transmitting pre-encoded video has been developed*. In particular, the concept of *importance* of data units has been introduced which describes the importance of a data unit for the reconstruction quality of a sequence. The framework can be applied to progressively coded sources, scalable coded sources, and sources which can be represented by dependency graphs. Extensions to multiple versions have been developed. Furthermore, the application of this framework permits complexity-reduced simulation of packet-lossy video transmission.

Summary and Major Findings: Transmission Toolbox for Multimedia Communication

In chapter 4, error protection tools suitable for the transmission over wireless and packet-lossy channels have been introduced:

- The existence of asymptotically good Forward Error Correction (FEC) codes with flexibility in terms of block lengths and code rate allows their usage in many different system environments and applications in mobile communication scenarios. The performance of asymptotically good FEC can quite well be estimated by the use of information-theoretic bounds such as channel capacity or cutoff rate. This knowledge provides a powerful toolkit for system analysis, design, and optimizations. Based on these bounds, *we have formalized and analyzed different FEC-based error protection schemes in mobile radio channels for delay-limited applications: rate-adaptive FEC codes, Unequal Error Protection (UEP), Automatic Repeat reQuest (ARQ) schemes, as well as power control and channel adaptation.*

- For the practical realization of FEC codes, we *make use of and have implemented rate–compatible punctured convolutional codes, in particular those with high memory*. Different

decoding methods for convolutional codes have been introduced and analyzed, with special focus on sequential decoding schemes based on stack and Fano algorithm. The performance of high-memory convolutional codes together with sequential decoding at the cutoff rate has been verified. The performance of the codes has been analyzed. Furthermore, appropriate mobile radio access schemes and interleavers have been proposed to be combined with channel codes. It is important to note that the findings in this work are not specific to the applied FEC codes, however, in order to verify the results by practical system designs, these codes have been chosen.

- A change of paradigm in the FEC decoding has been introduced based on the observation that for progressive multimedia codecs any data after the first decoding error cannot be interpreted: The first error within a source message should be as "late" as possible. Therefore, *we have developed and implemented a new error protection scheme, namely regressive UEP together with path shortening at the receiver*. In this case, only the initial reliable information of a source block is delivered to the media decoder. Specifically, the *Far End Error Decoding (FEED) has been developed in this work*. FEED is an extension to conventional convolutional decoding algorithms in a sense that it allows to determine the reliability of a path. This path reliability is used to determine the useful and reliable initial part of a progressively coded source. An optimal path reliability based on a trellis solution has been developed. However, due to the complexity especially for high memory convolutional codes, a major contribution of this chapter is the derivation of a path reliability computation based on stack and Fano sequential decoding. Performance results and comparisons have been provided.

- We also *introduced and developed erasure protection schemes* for the use over link layer or application layer packet loss channels. Promising performance results for IP-loss channels have been obtained showing that FEC can also provide significant gains if used above the physical layers.

Summary and Major Findings: Progressive Source Coding and Unequal Error Protection

In chapter 5, several schemes which combine progressively coded source and unequal error protection have been introduced:

- A transmission system incorporating a progressively coded source together with unequal error protection has been introduced. Information-theoretic considerations have been used to assess such a system when applied on a binary-input symmetric channel: A progressive vector quantizer is combined with channel codes with error probability performing at the Gallager bound. By applying high-resolution assumptions, *a distortion function for a multi-layer system with fixed channel coding rates has been derived*. Based on this the optimized code rates minimizing the end-to-end distortion can be obtained along with the minimum distortion itself.

- For the specific case with equal transmission block length a recursive method to derive the optimal code rates has been proposed. For the specific case with equal source block length for each layer the optimal code rates can be determined directly. Applying these rate allocation methods, *we were able to show that that for a multi-layer system unequal error protection minimizes the end-to-end distortion*. However, *it was also shown that layered transmission always introduces a penalty in terms of distortion or resource consumption*. To

obtain the same end-to-end distortion, the rate for a layered system with M layers needs to be increased roughly by a factor M in order to obtain the same performance as for the single layer case. The results on the penalty when using multistage transmission and unequal error protection are also supported by a practical experiment.

- We have introduced *a framework for Unequal Erasure Protection (UXP) in combination with progressively coded sources based on Reed-Solomon codes for the transmission over packet-lossy channels*. This combined source–channel coding system can also be viewed as a Multiple Description Coding (MDC) system. We have derived the expected quality for such a system taking into account the packet loss rate and the quality-rate function of the source. Furthermore, *an optimized source coding vector minimizing this distortion is derived and it is shown that this optimization can be solved by a dynamic programming approach*. The constrained optimization for a fixed number of layers can be extended by an unconstrained optimization. In this case, not only the code rates of each layer are determined but also the optimum number of layers is selected.

- We have also *introduced a complexity and quality-scalable system for progressive image transmission over fading systems that makes use of the Far End Error Decoding (FEED) decoding algorithm together with unequal error protection and shortening*. To obtain a redundancy profile, a rate allocation is formulated that searches for the appropriate code rates for each layer by the application of puncturing. The optimized channel code rate vector can again be obtained using a dynamic programming approach. The algorithm can be modified to include source rate allocation. We refer to this optimization as *source–channel–optimized rate allocation* whereas for the case with a fixed number of source coding layers only channel code rates are determined referred to as *channel–optimized rate allocation*.

- To verify the performance of the FEED algorithm as well as the optimized rate allocation, a progressive image transmission system including SPIHT has been setup. *If the source–channel–optimized rate allocation is used, the FEED-based algorithm outperforms the reference system significantly at the same decoding complexity*. Furthermore, it is shown that the proposed FEED system is capable to trade receiver complexity and decoded quality. This novel property is very interesting for applications where different types of receivers are present in the system, for example for broadcast systems with a heterogeneous receiver population.

Summary and Major Findings: Video Transmission in Packet Lossy Environment

In chapter 6, we have introduced different tools and combination of tools to successfully transmit video over packet–lossy channels:

- The transmission of video is not only disturbed by errors due to missing information but also to a high degree due to error propagation as a consequence of the predictive encoding. Both artifacts should be combatted when designing a video transmission system for packet–lossy channels. As subjective tests are infeasible to quantify the quality of packet-lossy video transmission, *we propose and also have used two different objective measures: The average Peak Signal–to–Noise Ratio (PSNR) as well as the PSNR of the Mean Square Error (MSE)*. This approach provides a good indication on the subjective quality, and using both measures allows to detect deficiencies of one or the other measure for certain cases. It is also observed that the availability of the expected channel conditions at the transmitter is beneficial to

appropriately select error resilience tools. We *have integrated the expected channel statistics in optimized mode selection schemes in the video encoder as well as in the selection of appropriate error resilience tools*.

- We have *designed, developed and implemented a toolbox for error resilient video transmission based on H.264/AVC*: This toolbox includes among others slice-structured coding, Flexible Macroblock Ordering (FMO), random and channel-adaptive intra updates, interactive error control, as well as advanced error concealment in the decoder. Specifically, *we have developed a generic scheme for the estimation of the expected decoder pixel distortion in the encoder taking into account the channel conditions*. This estimation has been used in a rate-distortion optimized mode selection for optimized and standard-compliant video encoding for packet lossy channels. Furthermore, *we have developed several interactive error control schemes* that make use of the multiple reference frame concept in H.264/AVC and rate-distortion optimized mode selection in the test model encoder.

- All H.264/AVC based error resilience tools *have been optimized and analyzed in a realistic packet loss environment for a set of formalized test conditions*. The conclusions for an H.264/AVC based approach are as follows: The most effective approach to stop error propagation is to exploit feedback messages from the decoder and combine it with multiple reference frames in modern video codecs using Interactive Error Control (IEC). Without the availability of feedback messages, channel–optimized mode selection schemes perform well and should be used by encoders in packet-lossy environments. Other error resilience tools have been investigated: advanced packetization schemes such as slice-structuring or FMO together with appropriate error concealment methods have been proposed by others, but the results in this chapter cannot provide sufficient evidence for the usefulness of these approaches in packet-lossy environments.

- For the design of a video system it is important to understand the application constraints in terms of acceptable delay, available information, as well as online or offline coding schemes. In general it is observed that the avoidance of errors can significantly improve the performance. Therefore, error resilience may not be provided by the video codec itself but by underlying error protection tools. An *erasure FEC framework has been implemented and combined with H.264/AVC video coding* for the transmission over packet lossy links. It is verified that errors can be avoided completely which results in stable and consistent video quality over the entire sequence.

- Furthermore, *a MDC video coding scheme has been realized by the use of the PTVC and an UXP scheme based on Reed-Solomon codes*. An appropriate rate allocation scheme has been implemented following the proposal in chapter 5. However, it is observed that multiple description video schemes outperform simple FEC schemes with single layer video codecs only under specific application constraints, usually for low delay applications.

Summary and Major Findings: Video Services in UMTS-like Environments

In chapter 7, system design guidelines for video services in UMTS-like environments have been introduced:

- We *have integrated the toolbox for H.264/AVC-based error resilient video transmission as introduced in chapter 6 in a UMTS-like video service environment*. The framework and test

conditions had been developed within the H.264/AVC standardization based on our contributions. The framework includes quality evaluation guidelines, test patterns, radio link layer QoS means such as retransmissions, as well as interfaces to map H.264/AVC Network Abstraction Layer (NAL) units to the packet-switched radio bearers.

- Based on the simulations and analysis using this framework, *we provide detailed system design guidelines on the usage of video features, application layer transport features, as well as radio link transport features for different applications*. For download-and-play as well as on-demand streaming applications, video compression efficiency and variable bitrate (VBR) rate control is most essential. The sufficient QoS should be provided by the radio bearer. For live applications, CBR-like video rate control is preferable for dedicated channels. Video error resilience is not essential as the radio bearer can provide sufficient QoS. For broadcast applications without any feedback, extended FEC on the physical layer and/or application layer FEC is an appropriate choice. Video should be coded with regular IDR frames for random access and error resilience. Application layer FEC can easily trade performance and delay.

- For low-delay mobile video applications over dedicated channels, the video should apply strict CBR-like rate control. The radio bearer should operate in unacknowledged mode (UM) without any retransmission. The video application itself must provide sufficient robustness. In case that feedback is not available or only limited to reporting statistics, for example, in conferencing applications, more frequent intra-MB updates based on robust mode decision as well as slice-structured coding should be used. *For video telephony, we recommend the exploitation of a fast feedback channel for interactive error control (IEC)*. No additional means of error resilience are necessary. This results in surprisingly good performance without sacrificing radio efficiency significantly.

Summary and Major Findings: Mobile Conversational Video

In chapter 8, system designs and optimization for mobile conversational video services have been introduced:

- Mobile conversational video applications introduce significant challenges in the system design, as the delay constraints as well as the variability and bandwidth limitation of the mobile radio channel require careful selection of the source coding tools, the error protection tools, as well as the resource allocation. To obtain insight into good system designs, *we have developed a simplified, but representative evaluation framework to compare different system designs*.

- Video coding and error protection tools in mobile video communication need to provide adaptation capabilities to varying transmission conditions. We have integrated different adaptive tools in mobile conversational video transmission systems: Specifically for video coding, *we have integrated H.264/AVC with single layer coding and data partitioning, as well as the PTVC*. Specific error resilience schemes are also supported: *For error protection, FEC schemes applying EEP and UEP, ARQ schemes in combination with FEC, as well as power control have been integrated and investigated*. The different integrated schemes provide tradeoffs in terms of flexibility, performance and implementation complexity. The combination of different tools is preferably supported by appropriate feedback messages as

well as by appropriate resource allocation algorithms to fully exploit the potentials of the proposed systems towards meeting the service requirements.

- For H.264/AVC single layer schemes, only the combination with EEP for each frame is reasonable. The simplest system design applies a constant channel code rate together with a CBR video rate control for the entire sequence. For such a system, *if the optimized channel code rate is used, it turns out that video error resilience is of little relevance, and may be omitted entirely - the classical separation of source channel coding provides also a good system design for low-delay mobile conversational video transmission.* Video error resilience such as channel-optimized mode selection are only of importance for robust system design, e.g., if a priori information on expected channel conditions at the transmitter is inaccurate. However, *we have shown by extensive simulations that overall IEC-based systems with fast feedback provide the best option to maximize performance and avoid error propagation in variable transmission conditions.* Therefore, fast IEC has been adopted to the majority of our system designs.

- The H.264/AVC-based system has been extended by the integration of data partitioning. We have developed a suitable quality-optimizing source and channel code rate selection algorithm for H.264/AVC data partitioning that generally results in an UEP assignment. *Source and channel code rate optimized H.264/AVC data partitioning together with UEP provides a powerful scheme to adapt to varying transmission conditions in H.264/AVC-based mobile conversational video systems.* Specifically it can avoid the loss of entire video frames.

- Better adaptation to the varying channel conditions can be obtained by the use of progressively coded video frames. Therefore, we have integrated the PTVC in mobile conversational video communication systems including IEC-based drift elimination. *We have followed the same basic principle for all error protection schemes in combination with PTVC, namely to push the first error as far as possible out in the progressively coded bit-stream for each video frame.* For the combination with regressive UEP, FEED-based decoding and a source–channel–optimized rate allocation, we have shown that such a PTVC-based system can compete with H.264/AVC-based systems, in particular if more consistent video quality is of higher importance than high average PSNR.

- If a fast feedback channel for fast retransmission is available for error protection, the same basic principle can be implemented even more efficiently by the use of non-persistent ARQ schemes, especially hybrid ARQ schemes of type 2. *We have developed quality-optimized ARQ-based systems for which the adaptation of the reference frames in the PTVC encoder can be realized in the transmitter without specific VFI but only based on the ARQ status messages.*

- The radio resources can be exploited most efficiently if the transmitter is aware of the Channel State Information (CSI) before allocating the transmission resources code rate and power. By the use of these means, *we have shown that the combination of PTVC and CSI-based error protection schemes outperform any other system, including H.264/AVC-based systems by* 1*dB in PSNR or* 30% *to* 50% *in bit-rate savings*.

- Even more powerful schemes are realized by relaxing the power restrictions to only an average power constraint. We have modified the power allocation scheme by taking into account Source Significance Information (SSI) of each PTVC encoded frame for the power allocation. *We have introduced an SSI-aware power control scheme for which the results show the*

overall best performance with very consistent quality. This system combining PTVC with reference frame adaptation, powerful FEC, as well as SSI-aware power control provides excellent quality and shows that by the application of cross-layer design mobile conversational video system can provide high and consistent performance.

Summary and Major Findings: Video Streaming over VBR Mobile Channels

In chapter 9 streaming methods for variable bit-rate mobile channels have been introduced and investigated in detail:

- Online encoding or transcoding is generally too complex and not feasible for streaming servers. Therefore, means for video rate adaptation for pre-encoded content are necessary. *We have proposed an H.264/AVC based video streaming system which relies on three major principles for rate adaptation*: receiver playout buffering, temporal scalability and advanced bit-stream switching using the SP-frame concept of H.264/AVC. We have shown by probabilistic statements that for UMTS-like channels with power control and retransmissions the bit-rate variations due to link layer retransmissions can be well compensated by receiver buffering without adding significant additional delay. Playout buffering is an excellent means to compensate smaller variations of bit rates. However, in case of more heterogeneous reception conditions, playout buffering is not sufficient and the pre-encoded content needs to be modified by packet dropping and/or by bit-stream switching.

- A key element in our rate-adaptive streaming system is the scheduler in the transmitter. The scheduler should make decisions taking into account as much and as up-to-date information as possible. Beneficial information for the scheduler are importance, deadlines, and size of certain media packets as well as the current channel state and the expected supported bit-rate. *We have developed the concept of a scheduler that decides for each transmission opportunity which data unit is transmitted next*. A formalized description of the available and necessary source and channel information has been provided. The source information abstraction relies on the framework developed in this work. For the channel abstraction the distribution of the supported bit-rate over some period of time is well approximated by a normal distribution. The scheduler makes decisions such that the expected quality is maximized.

- The implementation of an optimal scheduler is prohibitively complex and completely intractable. For this purpose, we have developed a set of complexity reductions that allow for pre-computing a weighted-cumulative importance for all data units which may possibly be transmitted at this transmission opportunity. Additional complexity reductions are introduced. *We have implemented a scheduler that allows to trade complexity and performance.*

- The *proposed streaming system, including the developed scheduler has been integrated in an experimental setup on top of an EGPRS-like channel* using playout buffering, temporal scalability, as well as bit-stream switching based on H.264 SP-picture concept. It is verified that by the use of optimized scheduling, temporal scalability can be optimized. Furthermore, a system with four available streams has been evaluated: By the use of SP-frame switching and optimized scheduling as proposed in this work, conventional state-of-the-art systems using CBR encoded streams can be outperformed. However, as a major finding in this work *we emphasize that optimization procedures such as the scheduler as proposed in here are essential to exploit the potentials of temporal scalability and bit-stream switching in realistic mobile transmission environments.*

Outlook

Based on the findings in this work, there are several interesting questions and open problems, which may be addressed in subsequent studies:

- This work addressed complexity issues in the transmission and receiver units. Especially handhelds nowadays are still complexity- and memory-restricted. Especially promising features such as IEC or advanced FEC schemes such as FEED require a detailed complexity analysis if integrated in handheld architectures.

- We have addressed the transmission of conversational and streaming applications in mobile communication environments. The focus in this work was mainly on the optimization of a single-user transmission scheme. However, in advanced mobile system designs such as HSDPA, resources will be shared which enables a significant amount of new options in the optimization of mobile video transmission. We have provided an initial analysis in section 8.3.4 for the case that the single user only needs to comply to an average power constraint and in [LJSB04] for multi-user video streaming over HSDPA. Those results show the significant potentials of multi-user optimized communication. Additional work is necessary.

- This work focused mainly on point-to-point transmission schemes, broadcast and multicast applications have not been treated explicitely. However, nowadays mobile broadcasting applications get increasingly popular [FA08] and the system design in this case offers many potentials for cross-layer design. For example, for Mobile TeleVision (TV) service, important aspects to be studied are statistical multiplexing, low channel change times, and application layer FEC [SSW+08].

- Video transmission over the Internet has also gained significant interest due to emerging Internet Protocol TeleVision (IPTV) services. In this case, quality must be comparable to or even better than existing TV distribution systems. This requires new architectures and system designs, including advanced application layer error protection schemes. We have presented some initial work in [SLW08] but additional work on cross-layer optimizations are required to provide highest efficiency and quality.

A

Outline of Proofs for Importance and Quality Definitions

In this Appendix some concepts and equations introduced in section 3.6 are proven.

A.1 Received Quality as Sum of Importances

In the following the outline of the proof for (3.39) is given.

Outline of proof. To prove (3.39) let us first define the following abbreviations:

$$\epsilon_n \triangleq \mathcal{C}_n \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \mathcal{C}_m \tag{A.1}$$

$$\hat{Q}_i(n) \triangleq \epsilon_n Q\left(s_i, \hat{s}_n\right) + \left(1-\epsilon_n\right) \hat{Q}_i(c\left(n\right)) \tag{A.2}$$

$$\hat{Q}_i(0) \triangleq Q\left(s_i, \hat{s}_0\right) \tag{A.3}$$

$$\mathcal{Q}_i \triangleq Q\left(s_i, \hat{s}_i\right) \tag{A.4}$$

In addition, we need the following relation

$$\forall_{i,n \vdash i}\ \left(1-\epsilon_i\right) \hat{Q}_i(n) = \left(\epsilon_n - \epsilon_i\right) Q\left(s_i, \hat{s}_n\right) + \left(1-\epsilon_n\right) \hat{Q}_i(c\left(n\right)), \tag{A.5}$$

which is easily shown by using $\forall_{i,n\vdash i}\,\epsilon_i\epsilon_n=\epsilon_i$ and by inserting in (A.2). Hence,

$$\begin{aligned}
\sum_{\substack{i=n+1\\ n\vdash i}}^{N}\hat{Q}_i(i) &= \sum_{\substack{i=n+1\\ n\vdash i}}^{N}\epsilon_i Q\left(s_i,\hat{s}_i\right)+\left(1-\epsilon_i\right)\hat{Q}_i(c\left(i\right))\\
&= \sum_{\substack{i=n+1\\ n\vdash i}}^{N}\epsilon_i Q\left(s_i,\hat{s}_i\right)+\left(\epsilon_{c(i)}-\epsilon_i\right)Q\left(s_i,\hat{s}_{c(i)}\right)+\left(1-\epsilon_{c(i)}\right)\hat{Q}_i(c\left(c\left(i\right)\right))\\
&= \sum_{\substack{i=n+1\\ n\vdash i}}^{N}\epsilon_i\left[\left(\mathcal{Q}_i-Q\left(s_i,\hat{s}_{c(i)}\right)\right)+\sum_{\substack{k=i+1\\ i\vdash k}}^{N}\left(Q\left(s_k,\hat{s}_i\right)-Q\left(s_k,\hat{s}_{c(i)}\right)\right)\right]\\
&\quad+\epsilon_n\sum_{\substack{k=n+1\\ n\vdash k}}^{N}\left(Q\left(s_k,\hat{s}_n\right)-\hat{Q}_k(c\left(n\right))\right)+\sum_{\substack{k=n+1\\ n\vdash k}}^{N}\hat{Q}_k(c\left(n\right)) \qquad \text{(A.6)}
\end{aligned}$$

The received quality according to Eq. (3.39) is then obtained as

$$\begin{aligned}
Q(\mathcal{C}) &\overset{(a)}{=} \frac{1}{N}\sum_{n=1}^{N}\hat{Q}_n(n)\\
&\overset{(b)}{=} \frac{1}{N}\sum_{\substack{n=1\\ c(n)=0}}^{N}\left[\hat{Q}_n(n)+\sum_{\substack{i=n+1\\ n\vdash i}}^{N}\hat{Q}_i(i)\right]\\
&\overset{(c)}{=} \frac{1}{N}\sum_{\substack{n=1\\ c(n)=0}}^{N}\left[\epsilon_n\mathcal{Q}_n+\left(1-\epsilon_n\right)\hat{Q}_n(c\left(n\right))+\sum_{\substack{i=n+1\\ n\vdash i}}^{N}\hat{Q}_i(i)\right]\\
&\overset{(d)}{=} \frac{1}{N}\sum_{\substack{n=1\\ c(n)=0}}^{N}\left[\epsilon_n\left(\mathcal{Q}_n-Q\left(s_n,\hat{s}_0\right)+\sum_{\substack{k=n+1\\ n\vdash k}}^{N}\left(Q\left(s_k,\hat{s}_n\right)-\hat{Q}_k(0)\right)\right)+Q\left(s_n,\hat{s}_0\right)\right.\\
&\qquad+\sum_{\substack{i=n+1\\ n\vdash i}}^{N}\epsilon_i\left(\left(\mathcal{Q}_i-Q\left(s_i,\hat{s}_{c(i)}\right)\right)+\sum_{\substack{k=i+1\\ i\vdash k}}^{N}\left(Q\left(s_k,\hat{s}_i\right)-Q\left(s_k,\hat{s}_{c(i)}\right)\right)\right)\\
&\qquad\left.+\sum_{\substack{i=n+1\\ n\vdash i}}^{N}\hat{Q}_i(c\left(n\right))\right]\\
&\overset{(e)}{=} \frac{1}{N}\sum_{n=1}^{N}Q\left(s_n,\hat{s}_0\right)+\frac{1}{N}\sum_{n=1}^{N}\epsilon_n\left(\mathcal{Q}_n-Q\left(s_n,\hat{s}_{c(n)}\right)+\sum_{\substack{k=n+1\\ n\vdash k}}^{N}\left(Q\left(s_k,\hat{s}_n\right)-Q\left(s_k,\hat{s}_{c(n)}\right)\right)\right)\\
&\overset{(f)}{=} \mathcal{Q}_0+\sum_{n=1}^{N}\epsilon_n\mathcal{I}_n. \qquad \text{(A.7)}
\end{aligned}$$

whereby

(a) holds applying the decoding and concealment strategy as introduced above,

(b) holds with the same argument as mentioned previously that each frame is either concealed with grey ($c\,(n) = 0$), or its concealment depends on a frame which is concealed by grey,

(c) can be shown by inserting (A.5),

(d) can be shown by inserting (A.6),

(e) results from simple reordering and the fact that the sets for the sums are mutually exclusive,

(f) is obvious with the definitions in (3.38) and (3.39).

■

A.2 Expected Quality as Weighted Sum of Importance

In the following the outline of the proof for (3.40) is given.

Outline of proof.

$$
\begin{aligned}
E\left\{Q(\hat{\mathcal{C}})\right\} &= \sum_{\mathcal{C}\in\{0,1\}^N} \mathcal{Q}(\mathcal{C})\Pr\{\hat{\mathcal{C}}=\mathcal{C}\} \\
&\overset{(a)}{=} \sum_{\mathcal{C}\in\{0,1\}^N} \left(\mathcal{Q}_0 + \sum_{n=1}^{N} I_n\mathcal{C}_n \prod_{\substack{m=1\\ m\prec n}}^{n-1} \mathcal{C}_m\right)\Pr\{\hat{\mathcal{C}}=\mathcal{C}\} \\
&\overset{(b)}{=} \sum_{\mathcal{C}\in\{0,1\}^N} \mathcal{Q}_0\Pr\{\hat{\mathcal{C}}=\mathcal{C}\} + \sum_{n=1}^{N}\sum_{\mathcal{C}\in\{0,1\}^N}\left(I_n\mathcal{C}_n \prod_{\substack{m=1\\ m\prec n}}^{n-1} \mathcal{C}_m\right)\Pr\{\hat{\mathcal{C}}=\mathcal{C}\} \\
&\overset{(c)}{=} \mathcal{Q}_0 + \sum_{n=1}^{N}\mathcal{I}_n \sum_{\mathcal{C}\in\{0,1\}^N}\left(\mathcal{C}_n \prod_{\substack{m=1\\ m\prec n}}^{n-1} \mathcal{C}_m\right) \\
&\qquad\qquad \cdot \prod_{i=1}^{N}\Pr\{\hat{\mathcal{C}}_i=\mathcal{C}_i \,\big|\, \hat{\mathcal{C}}_1=\mathcal{C}_1,\ldots,\hat{\mathcal{C}}_{i-1}=\mathcal{C}_{i-1}\} \\
&\overset{(d)}{=} \mathcal{Q}_0 + \sum_{n=1}^{N} I_n\Pr\{\hat{\mathcal{C}}_n=1 \,\big|\, \forall_{k\prec n}\hat{\mathcal{C}}_k=1\}\prod_{\substack{m=1\\ m\prec n}}^{n-1}\Pr\{\hat{\mathcal{C}}_m=1 \,\big|\, \forall_{k\prec m}\hat{\mathcal{C}}_k=1\}
\end{aligned}
$$

Here,

(a) results from inserting (3.39),

(b) holds as the sums are obviously exchangeable,

(c) assumes that the loss of data units only depends on past, but not on future channel states[1],

[1]Obviously, we can not assume that loss occurs statistically independent, as we might have strong correlations between data units or link layer units.

(d) can be shown by using the following relationship

$$\begin{aligned}
&\sum_{\mathcal{C}\in\{0,1\}^N} \left(\mathcal{C}_n \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \mathcal{C}_m \right) \prod_{i=1}^{N} \Pr\{\hat{\mathcal{C}}_i = \mathcal{C}_i \,\big|\, \hat{\mathcal{C}}_1 = \mathcal{C}_1, \ldots, \hat{\mathcal{C}}_{i-1} = \mathcal{C}_{i-1}\} \\
&\quad = \sum_{\{\mathcal{C}\in\{0,1\}^N | \mathcal{C}_n=1 \wedge \forall_{m \prec n} \mathcal{C}_m=1\}} \prod_{i=1}^{N} \Pr\{\hat{\mathcal{C}}_i = \mathcal{C}_i \,\big|\, \hat{\mathcal{C}}_1 = \mathcal{C}_1, \ldots, \hat{\mathcal{C}}_{i-1} = \mathcal{C}_{i-1}\} \\
&\quad = \Pr\{\hat{\mathcal{C}}_n = 1 \,\big|\, \forall_{k \prec n} \hat{\mathcal{C}}_k = 1\} \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \Pr\{\hat{\mathcal{C}}_m = 1 \,\big|\, \forall_{k \prec m} \hat{\mathcal{C}}_k = 1\} \qquad \text{(A.8)}
\end{aligned}$$

and by replacing the left-hand side of this relation by the right-hand side in (d).

■

B

Acronyms

1G first–generation

2G second–generation

3G third–generation

3GPP Third–Generation Partnership Project

AEC Adaptive temporal and spatial Error Concealment

ACK positive ACKnowledgement

ACR Absolute Category Rating

AM Acknowledged Mode

AMC Adaptive Coding and Modulation

AMPS Advanced Mobile Phone Service Telephony

AMR Adaptive Multi–Rate

APP a posteriori probability

ARQ Automatic Repeat reQuest

ASO Arbitrary Slice Ordering

ATS Ahead-of-Time Streaming

AUC Authentification Center

AVC Advanced Video Coding

AVT Audio/Video Transport

AWGN Additive White Gaussian Noise

BCJR Bahl–Cocke–Jelinek–Raviv

BER bit error rate

BF-AWGN Block Fading AWGN

BICM Bit–Interleaved Coded Modulation

bpp bit per pixel

BPSK Binary Phase Shift Keying

BSC Base Station Controller

BSS Base Station System

BTS Base Tranceiver Station

CABAC Context–Adaptive Binary Arithmetic Coding

CAVLC Context–Adaptive Variable Length Coding

CBR Constant Bit–Rate

cdf cumulative distribution function

CDMA Code Division Multiple Access

CIF Common Intermediate Format

CPB Coded Picture Buffer

CR Conditional Replenishment

CRC Cyclic Redundancy Check

CSD Circuit Switched Data

CSI Channel State Information

CS Coding Scheme

DCCP Dynamic Congestion Control Protocol [KHF04]

DCT Discrete Cosine Transform

DFD Displaced Frame Difference

DMC Discrete Memoryless Channel

DPB Decoded Picture Buffer

DRI Decoder Reliability Information

DTS Decoding Time Stamp

ECSD Enhanced CSD

ECSQ Entropy Constrained Scalar Quantization

ECVQ Entropy Constrained Vector Quantization

EDF Early-Deadline First

EDGE Enhanced Data Rates for GSM and TDMA/136 Evolution

EEP Equal Error Protection

EFI Error–Protection Feedback Information

EFR Enhanced Full Rate

EGPRS Enhanced GPRS

EIF Error Indication Flag

EIR Equipment Identity Register

EUDCH Enhanced Uplink Data CHannel

EZW Embedded Zerotree Wavelet

FDD Frequency–Division Duplex

FEC Forward Error Correction

FEED Far End Error Decoding

FGS Fine–Granular Scalability

FMO Flexible Macroblock Ordering

FR Full Rate

FTP File Transfer Protocol [PR85]

fps frames per second

FU Fragmentation Unit

GDR Gradual Decoding Refresh

GERAN GSM Radio Access Network

GGSN Gateway GPRS support node

GMSK Gaussian Minimum Shift Keying

GOB Group-of-Blocks

GOP Group-of-Picture

GPRS General Packet Radio Service

GSM Global System for Mobile Communication

HLR Home Location Register

HRD Hypothetical Reference Decoder

HSCSD High–Speed Circuit Switched Data

HSDPA High–Speed Downlink Packet Access

HTTP Hypertext Transfer Protocol [FGM+97]

IDR Instantaneous Decoder Refresh

IEC Interactive Error Control

IETF Internet Engineering Task Force

iid independent and identically distributed

IMS IP Multimedia Subsystem

IP Internet Protocol [Pos81]

IPTV Internet Protocol TeleVision

IS International Standard

ISD Independent Segment Decoding

ISDN Integrated Services Digital Network

ISI inter–symbol interference

ISO/IEC International Organization for Standardization/International Electrotechnical Commission

ITU-T International Telecommunications Union – Telecommunications Sector

JPEG Joint Photographic Expert Group

JVT Joint Video Team

LER LLC error rate

LLC Logical Link Control

LLR log–likelihood ratio

MAC Medium Access Control

MAP Maximum A Posteriori

MB Macroblock

MBMS Multimedia Broadcast/Multicast Service [MBM03, MBM05]

MCP Motion–Compensated Prediction

MCS Modulation and Coding Scheme

MCTF Motion–Compensated Temporal Filtering

MDC Multiple Description Coding

MDDE Multiple–Decoder Distortion Estimation

MDS Maximum Distance Separable

MIMO Multiple–Input Multiple–Output

MLWDF Modified Largest Weighted Delay First

MMS Multimedia Messaging Service

MPEG Moving Pictures Expert Group

MSC Mobile Switching Center

MSB Most Significant Bit

MSE Mean Square Error

MSVQ Multi–Stage Vector Quantization

MS Mobile Station

MTAP Multiple–Time Aggregation Unit

MTSI Multimedia Telephony over IMS [MTS07]

NAK Negative AcKnowledgement

NAL Network Abstraction Layer

NEWPRED New Prediction

NRI NAL Reference Identification

OSI Open Systems Interconnection [Zim80]

PCM Pulse Code Modulation

PDC Personal Digital Cellular

PDCP Packet Data Convergence Protocol

pdf probability density function

PDU Packet Data Unit

PET Priority Encoded Transmission

PFC Previous Frame Concealment

PPS Picture Parameter Set

psd power spectral density

PSK Phase Shift Keying

PSP primary SP

PSNR Peak Signal–to–Noise Ratio

PSC Packet–Switched Conversational [PSC02]

PSS Packet–Switched Streaming [PSS02a, PSS02b]

PSTN Public Switched Telephone Network

PTS Presentation Time Stamp

PTVC Progressive Texture Video Coding

QAM Quadrature Amplitude Modulation

QCIF Quarter Common Intermediate Format

QP Quantization Parameter

QPSK Quaternary Phase Shift Keying

QPSP Quantization Parameter for SP picture

QPSP2 Quantization Parameter for secondary SP picture

QoS Quality-of-Service

RCPC Rate–Compatible Punctured Convolutional

RGB Red–Green–Blue

RLC Radio Link Control

RMT Reliable Multicast Transmission

ROPE Recursive Optimal per–Pixel Estimate

RPS Reference Picture Selection

RS Reed–Solomon

RTCP Real–Time Control Protocol

RTP Real–Time Transport Protocol [SCFJ03]

RTSP Real–Time Streaming Protocol [SRL98]

RTT Round–Trip Time

RoHC Robust Header Compression [HJH$^+$01]

SAP Session Announcement Protocol [HPW00]

SDP Session Description Protocol [HJ98]

SDU Service Data Unit

SEI Supplemental Enhancement Information

SGSN Serving GPRS support node

SI Switching–Intra

SIP Session Initiation Protocol [HSSR99]

SMS Short Messaging Service

SNDCP Sub-Network Dependent Convergence Protocol

SNR Signal–to–Noise Ratio

SNIR Signal–to–Noise and Interference Ratio

SP Switching–Predictive

SPIHT Set Partitioning in Hierarchical Trees

SPS Sequence Parameter Set

SAD Sum of Absolute Differences

SSD Sum of Squared Differences

SSI Source Significance Information

SSP Secondary SP

STAP Single–Time Aggregation Unit

SVC Scalable Video Coding

TBS Timestamp-Based Streaming

TCP Transmission Control Protocol [Pos80a, SK91]

TDD Time–Division Duplex

TDMA Time Division Multiple Access

TML Test Model Long–term

TSVQ Tree–Structured Vector Quantization

TTI Transmission Time Interval

TV TeleVision

UDP User Datagram Protocol [Pos80b]

UEP Unequal Error Protection

ULP Unequal Level Protection

UM Unacknowledged Mode

UMTS Universal Mobile Telecommunications Systems

UTRAN UMTS Terrestrial Radio Access Network

UVLC Universal Variable Length Coding

UXP Unequal Erasure Protection

VBR Variable Bit–Rate

VBV Video Buffer Verifier

VCEG Video Coding Experts Group

VCL Video Coding Layer

VFI Video Feedback Information

VLC Variable Length Code

VLR Visiting Location Register

VQ Vector Quantization

WSSUS Wide–Sense Stationary Uncorrelated Scattering

WWW World-Wide Web

Bibliography

[ABE+96] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Transactions on Information Theory*, 42:1737–1744, November 1996.

[AM91] J.B. Andersen and S. Mohan. *Source and Channel Coding: An Algorithmic Approach*. Kluwer Academic Press, Dordrecht, The Netherlands, 1991.

[APS99] M. Allman, V. Paxson, and W. Stevens. TCP congestion control. Request for Comments (standard) 2581, Internet Engineering Task Force (IETF), April 1999.

[ASX+06] A. Arnold, T. Stockhammer, W. Xu, J. Afzal, and C. Buchner. Demonstration of MBMS video streaming over GERAN. In *Proceedings 3rd IEEE Consumer Communications and Networking Conference (CCNC)*, volume 2, pages 1297–1298, 8-10 Jan. 2006.

[ATGX06] J. Afzal, T. Stockhammer T., Gasiba, and W. Xu. System design options for video broadcasting over wireless networks. In *Proceedings 3rd IEEE Consumer Communications and Networking Conference (CCNC)*, volume 2, pages 938–943, 8-10 Jan. 2006.

[BCD+98] C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D. Newell, J. Ott, G. Sullivan, S. Wenger, and C. Zhu. RTP payload format for the 1998 version of ITU-T rec. H.263 video (H.263+). Request for Comments (standard) 2429, Internet Engineering Task Force (IETF), October 1998.

[BCJR74] L.R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, pages 284–287, March 1974.

[BFF+98] L. Berc, W. Fenner, R. Frederick, S. McCanne, and P. Stewart. RTP payload format for JPEG-compressed video. Request for Comments (standard) 2435, Internet Engineering Task Force (IETF), October 1998.

[BGGM80] A. Buzo, A.H. Gray, R.M. Gray, and J.D. Markel. Speech coding based upon vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(5):562–574, May 1980.

[BGT93] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error–correcting coding and decoding: Turbo-codes. In *Proceedings IEEE International Conference on Communications (ICC)*, pages 1064–1070, Geneva, Switzerland, May 1993.

[BHS04] P. Brun, G. Hauske, and T. Stockhammer. Subjective assessment of H.264-AVC video for low-bitrate multimedia messaging services. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages 1145–1148, 24-27 Oct. 2004.

[Bil95] P. Billingsley. *Probability and Measure*. John Wiley and Sons, New York, USA, 1995.

[BPS98] E. Biglieri, J. Proakis, and S. Shamai (Shitz). Fading channels: Information-theoretic and communication aspects. *IEEE Transactions on Information Theory*, 44(6):2619–2692, October 1998.

[BS00] M. Bystrom and T. Stockhammer. Modeling of operational distortion-rate characteristics for joint source-channel coding of video. In *Proceedings International Conference on Image Processing (ICIP)*, volume 1, pages 359–362vol.1, 10-13 Sept. 2000.

[BS04] M. Bystrom and T. Stockhammer. Dependent source and channel rate allocation for video transmission. *IEEE Transactions on Wireless Communications*, pages 258–268, January 2004.

[BSM+01a] C. Buchner, T. Stockhammer, D. Marpe, G. Blättermann, and G. Heising. Efficient fine granular scalable video coding. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages 997–1000, 7-10 Oct. 2001.

[BSM+01b] C. Buchner, T. Stockhammer, D. Marpe, G. Blättermann, and G. Heising. Progressive texture video coding. In *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 3, pages 1813–1816, 7-11 May 2001.

[CC78b] P.R. Chevillat and D.J. Costello. Analysis of sequential decoding for specific time–invariant convolutional codes. *IEEE Transactions on Information Theory*, 24:443–451, July 1978.

[CCCQ00] Jian Cai, Li Fung Chang, Kapil Chawla, and Xiaoxin Qiu. Providing differentiated services in EGPRS through packet scheduling. In *Proceedings IEEE Globecom*, November 2000.

[CCS02] Codec for circuit–switched multimedia telephony service; general description. 3GPP Technical Recommendation TR26.110, 3GPP, December 2002.

[CCW97] M.-J. Chen, L.-G. Chen, and R.-M. Weng. Error concealment of lost motion vectors with overlapped motion compensation. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(3):560–563, June 1997.

[CF99] V. Chande and N. Farvardin. Joint source-channel coding for progressive transmission of embedded source coders. In *Proceedings Data Compression Conference (DCC)*, Snowbird(UT), USA, April 1999.

[CLG89a] P.A. Chou, T. Lookabaugh, and R.M. Gray. Entropy–constrained vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(1):31–42, January 1989.

[CLG89b] P.A. Chou, T. Lookabaugh, and R.M. Gray. Optimal pruning with applications to tree–structured source coding and modelling. *IEEE Transactions on Information Theory*, 35(2):299–315, March 1989.

[CM06] P. A. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. *IEEE Transactions on Multimedia*, 8:390–404, April 2006.

[Cov72] T. Cover. Broadcast channels. *IEEE Transactions on Information Theory*, 18:2–14, January 1972.

[CSK00] G. Cote, S. Shirani, and F. Kossentini. Optimal mode selection and synchronization for robust video communications over error–prone networks. *IEEE Journal on Selected Areas in Communications*, 18(6):952–965, June 2000.

[CT91] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, USA, 1991.

[CT01] G. Caire and D. Tuninetti. The throughput of hybrid-ARQ protocols for the gaussian collision channel. *IEEE Transactions on Information Theory*, 47(5):1971–1988, July 2001.

[CTB96] G. Caire, G. Taricco, and E. Biglieri. Capacity of bit-interleaved channels. *Electronic Letters*, 32:1060–1061, June 1996.

[CTB99] G. Caire, G. Taricco, and E. Biglieri. Optimum power control over fading channels. *IEEE Transactions on Information Theory*, 45:1468–1489, July 1999.

[CW99] S. Choi and J. Woods. Motion compensated 3-D subband coding of video. *IEEE Transactions on Image Processing*, 8:155–167, February 1999.

[Due97] M.J. Duerst. The progressive transmission disadvantage. *IEEE Transactions on Information Theory*, 43(1):347–350, January 1997.

[DWW04] S. Dumitrescu, X. Wu, and Z. Whang. Globally optimal uneven error–protected packetization of scalable code streams. *IEEE Transactions on Multimedia*, 6(2):230–239, April 2004.

[EC91] W.H.R. Equitz and T.M. Cover. Successive refinement of information. *IEEE Transactions on Information Theory*, 39(2):269–275, March 1991.

[Eff98] M. Effros. Optimal modeling for complex system design. *IEEE Signal Processing Magazine*, 15(6):51–73, November 1998.

[Eff99] M. Effros. Distortion–rate bounds for fixed- and variable-rate multiresolution source codes. *IEEE Transactions on Information Theory*, 45(6):1887–1910, September 1999.

[Eri70] T. Ericson. A gaussian channel with slow fading. *IEEE Transactions on Information Theory*, 16(3):353–355, May 1970.

[EVB01] J. Eberspächer, H.-J. Vögel, and C. Bettstetter. *GSM – Switching, Services and Protocols*. John Wiley and Sons, New York, USA, 2nd edition, March 2001.

[Eve63] G. Everett. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operational Research*, 11:399–417, 1963.

[FA08] B. Furht and S. Ahson, editors. *Handbook of Mobile Broadcasting: DVB-H, DMB, ISDB-T and MediaFLO*. CRC Press, Taylor and Francis Group, March 2008.

[Fan63b] R.M. Fano. A heuristic discussion of probabilistic decoding. *IEEE Transactions on Information Theory*, 9:64–74, April 1963.

[FCC03] T. Friedman, R. Caceres, and A. Clark. RTP control protocol extended reports (RTCP XR). Request for Comments (standard) 3611, Internet Engineering Task Force (IETF), November 2003.

[FG03] M. Flierl and B. Girod. Generalized B–pictures and the draft H.264/AVC video–compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):587–597, July 2003.

[FGM+97] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext transfer protocol – HTTP1.1. Request for Comments (standard) 2068, Internet Engineering Task Force (IETF), January 1997.

[FNI96] S. Fukunaga, T. Nakai, and H. Inoue. Error resilient video coding by dynamic replacing of reference pictures. In *Proceedings IEEE Globecom*, London, United Kingdom, November 1996.

[Fow00] J. Fowler. QccPack: an open-source software library for quantization,compression, and coding. In *Proceedings Data Compression Conference (DCC)*, Snowbird(UT), USA, April 2000. software can be downloaded at http://qccpack.sourceforge.net.

[G1196] ITU–T. *One way transmission time*, number G.114, February 1996.

[Gal68] R. Gallager. *Information Theory and Reliable Communications*. John Wiley and Sons, New York, USA, 1968.

[Ger79] A. Gersho. Asymptotically optimal block quantization. *IEEE Transactions on Information Theory*, 25(4):373–380, July 1979.

[GF99] B. Girod and N. Färber. Feedback-based error control for mobile video transmission. *Proceeding of the IEEE*, 97:1707–1723, October 1999.

[Gha96] M. Ghanbari. Postprocessing of late cells for packet video. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(6):669–678, December 1996.

[Gir87] B. Girod. The effiency of motion–compensating prediction for hybrid coding of video sequences. *IEEE Journal on Selected Areas in Communications*, 5(7):1140–1154, August 1987.

[Gir92] B. Girod. Psychovisual aspects of image communications. *Signal Processing*, 28(3):239–251, September 1992.

[Gir93] B. Girod. Motion–compensating prediction with fractional–pel accuracy. *IEEE Transactions on Communications*, 41(4):604–612, April 1993.

[Gir00] B. Girod. Efficiency analysis of multi–hypothesis motion–compensating. *IEEE Transactions on Image Processing*, 9(2):173–183, February 2000.

[GN99] R.M. Gray and D.L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383, October 1999.

[Goo99] B. Goodman. Internet telephony and modem delay. *IEEE Transactions on Networks*, 13:8–17, 1999.

[GR75] F. Giorda and A. Racciu. Bandwidth reduction of video signals via shift vector transmission. *IEEE Transactions on Communications*, 23:1002–1004, September 1975.

[GS98] R. Gurin and H. Schulzrinne. Network quality of service. In I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, pages 81–137. Morgan Kaufman, San Francisco (CA), USA, 1998.

[GSAX06] T. Gasiba, T. Stockhammer, J. Afzal, and W. Xu. System design and advanced receiver techniques for MBMS broadcast services. In *Proceedings IEEE International Conference on Communications (ICC)*, volume 12, pages 5444–5450, June 2006.

[GSX06] T. Gasiba, T. Stockhammer, and W. Xu. Reliable and efficient download delivery with Raptor codes. In *Proceedings of 4th International Symposium on Turbo Codes 4th International Symposium on Turbo Codes & Related Topics*, Munich, Germany, June 2006.

[GV97] A. Goldsmith and P. Varaiya. Capacity of fading channels with channel side information. *IEEE Transactions on Information Theory*, 43:1968–1992, November 1997.

[H12] ITU–T. *Codec for Videoconferencing using Primary Digital Group Transmission*, number H.120. version 1, 1984; version 2, 1988.

[H26a] ITU–T. *Video Codec for Audiovisual Services at $p\times 64$ kbit/s*, number H.261. version 1, Nov. 1984; version 2, Mar 1988.

[H26b] ITU–T. *Video Coding for Low Bit Rate Communication*, number H.263. version 1, November 1995; version 2, January 1998; version 2, November 2000.

[H2603] ITU–T and ISO/IEC JTC 1. *Advanced Video Coding for Generic Audiovisual Services*, number H.264 – ISO/IEC 14496-10(AVC), 2003.

[H3295] ITU–T. *Terminal of Low Bit Rate Communication*, number H.324, November 1995.

[H3297] ITU–T. *Multimedia Telephone Terminals over Error Prone Channels*, number H.324/Annex C, September 1997.

[H3298] ITU–T. *Packet based multimedia communication systems*, number H.323, February 1998.

[Hag88] J. Hagenauer. Rate–compatible punctured convolutional codes (RCPC codes) and their applications. *IEEE Transactions on Communications*, 36(4):389–400, April 1988.

[Hag95] J. Hagenauer. Source controlled channel decoding. *IEEE Transactions on Communications*, 43(9):2449–2457, September 1995.

[Hat80] M. Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE Transactions on Vehicular Technology*, 29(3):317–325, 1980.

[Hel68] J.A. Heller. Short constraint length convolutional codes. Space Program Summary 37–54 171–174, Jet Propulsion Laboratory, Pasadena, USA, December 1968.

[HFGC98] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar. RTP payload format for MPEG1/MPEG2 video. Request for Comments (standard) 2250, Internet Engineering Task Force (IETF), January 1998.

[HHS03] G. Hauske, R. Hofmeier, and T. Stockhammer. Subjective image quality of low-rate and low-resolution video sequences. In *Proceedings of 8th International Workshop on Mobile Multimedia Communications (MoMuC)*, Munich, Germany, March 2003.

[HJ98] M. Handley and V. Jacobson. Session description protocol. Request for Comments (standard) 2327, Internet Engineering Task Force (IETF), April 1998.

[HJH+01] H. Hannu, L.-E. Jonsson, R. Hakenberg, T. Koren, K. Le, Z. Liu, A. Martensson, A. Miyazaki, K. Svanbro, T. Wiebke, T. Yoshimura, and H. Zheng. Robust header compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed. Request for Comments (standard) 3095, Internet Engineering Task Force (IETF), July 2001.

[HKS96] T. Hasegewa, T. Kato, and K. Suzuki. A video retrieval protocol with video data prefetch and packet retransmission considering play-out deadline. In *Proceedings IEEE International Conference on Image Processing*, pages 32–39, November 1996.

[HM92] P. Haskell and D. Messerschmitt. Resynchronization of motion-compensated video affected by ATM cell loss. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, pages 545–548, 1992.

[HOK99] C.Y. Hsu, A. Ortega, and M. Khansari. Rate control for robust video transmission over burst-error wireless channels. *IEEE Journal on Selected Areas in Communications*, 17(5):756–773, May 1999.

[HOP96] J. Hagenauer, E. Offer, and L. Papke. Iterative decoding of binary block and convolutional codes. *IEEE Transactions on Information Theory*, pages 429–445, March 1996.

[HPW00] M. Handley, C. Perkins, and E. Whelan. Session announcement protocol. Request for Comments (standard) 2947, Internet Engineering Task Force (IETF), October 2000.

[HS99] J. Hagenauer and T. Stockhammer. Channel coding and transmission aspects for wireless multimedia. *Proceeding of the IEEE*, 87(10):1764–1777, October 1999. Special Issue on Video Transmission for Mobile Multimedia Applications.

[HSLG99] U. Horn, K. Stuhlmüller, M. Link, and B. Girod. Robust Internet video transmission based on scalable coding and unequal error protection. *IEEE Transactions on Image Processing*, 15(1–2):77–94, September 1999.

[HSSR99] M. Handley, H. Schulzrinne, E. Scholer, and J. Rosenberg. SIP: Session initiation protocol. Request for Comments (standard) 2543, Internet Engineering Task Force (IETF), March 1999.

[HSWD00] J. Hagenauer, T. Stockhammer, C. Weiß, and A. Donner. Progressive source coding combined with regressive channel coding for varying channels. In *Proceedings of the 3rd ITG Conference Source and Channel Coding (SCC'00)*, pages 123–130, Munich, Germany, January 2000.

[HSWH99] J. Hagenauer, T. Stockhammer, B. Wimmer, and E. Hundt. Error robust multiplexing schemes. *Signal Processing, Special Issue of Image Communications on Error Resilient Video*, 14:585–597, January 1999.

[HSX00] T. Hindelang, T. Stockhammer, and W. Xu. Verfahren zu gemeinsamen interleaving und codepunktierung. DE1991015687, WO0062425, 171–174, Deutsches Patentamt, Munich, Germany, October 2000.

[HT02] H. Holma and A. Toskala. *WCDMA for UMTS*. John Wiley and Sons, New York, USA, 2002.

[HW00] S.-T. Hsiang and J. W. Woods. Embedded video coding using motion compensated 3–D subband/wavelet filter banks. In *Proceedings International Packet Video Workshop*, Sardinia, Italy, May 2000.

[HWG04] M.M. Hannuksela, Y.-K. Wang, and M. Gabbouj. Isolated regions in video coding. *IEEE Transactions on Multimedia*, 6(2):250–267, April 2004.

[HZ97] P. Hochwald and K. Zeger. Tradeoff between source and channel coding. *IEEE Transactions on Information Theory*, 43(5):1412–1424, September 1997.

[IPM02] IP multimedia call control protocol based on SIP and SDP. 3GPP Technical Specification TS24.229, 3GPP, December 2002.

[ISO00] ISO/IEC 138180-2. Information technology - generic coding of moving pictures and associated audio information: Video (MPEG-2/H.262), video buffering verifier. Annex C, 2000. 2nd Edition.

[IT97] ITU-T. Video coding for low bit rate communication, Hypothetical Reference Decoder. Annex B, September 1997.

[ITU82] ITU–T. *Methods for the Subjective Assessment of the Quality of Television Pictures*, number 500–2, 1982.

[Jac96] K. Jack. *Video Demystified: A Handbook for the Digital Engineer*. LLH Technology Publishing, Eagle Rock (VA), 1996.

[Jak74] W. C. Jakes. *Microwave Mobile Communications*. John Wiley and Sons, New York, USA, 1974.

[JB67] I.M. Jacobs and E.R. Berlekamp. A lower bound to the distribution of computation for sequential decoding. *IEEE Transactions on Information Theory*, 13:167–174, April 1967.

[Jel69] F. Jelinek. An upper bound on moments of sequential decoding effort. *IEEE Transactions on Information Theory*, 15:140–149, April 1969.

[JJ81] J.R. Jain and A.K. Jain. Displacement measurements and its applications to interframe image coding. *IEEE Transactions on Communications*, 29(12):1799–1808, December 1981.

[JLSX04] H. Jenkač, G. Liebl, T. Stockhammer, and W. Xu. Flexible outer Reed-Solomon coding on RLC layer for MBMS over GERAN. In *Proceedings IEEE Vehicular Technology Conference (VTC)*, volume 5, pages 2777–2781Vol.5, 17-19 May 2004.

[JLSX05] H. Jenkač, G. Liebl, T. Stockhammer, and W. Xu. Retransmission strategies for MBMS over GERAN. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pages 1773–1779Vol.3, 13-17 March 2005.

[JMSX05] H. Jenkač, T. Mayer, T. Stockhammer, and W. Xu. Soft decoding of LT-codes for wireless broadcast. In *Proceedings of IST Summit 2005*, Dresden, Germany, June 2005.

[JS00] W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proceedings International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Chapel Hill(NC), USA, June 2000.

[JS05] H. Jenkač and T. Stockhammer. Asynchronous media streaming over wireless broadcast channels. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 1318–1321, 6-8 July 2005.

[JSK03] H. Jenkač, T. Stockhammer, and G. Kuhn. On video streaming over variable bit-rate and wireless channels. In *Proceedings of the International Packet Video Workshop*, April 2003.

[JSL04] H. Jenkač, T. Stockhammer, and G. Liebl. H.264/AVC video transmission over MBMS in GERAN. In *Multimedia Signal Processing, 2004 IEEE 6th Workshop on*, pages 191–194, Siena, Italy, 29 Sept.-1 Oct. 2004.

[JSX05a] H. Jenkač, T. Stockhammer, and W. Xu. Cross-layer issues and forward error correction for wireless video broadcast. In *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, volume 2, pages 999–1003, 11-14 Sept. 2005.

[JSX05b] H. Jenkač, T. Stockhammer, and W. Xu. Permeable-layer receiver for reliable multicast transmission in wireless systems. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pages 1805–1811Vol.3, 13-17 March 2005.

[JSX06a] H. Jenkač, T. Stockhammer, and W. Xu. Asynchronous and reliable on-demand media broadcast. *IEEE Network Magazine, Special Issue on Multimedia over Wireless Broadband Networks*, 20(2):14–20, March 2006.

[JSX06b] H. Jenkač, T. Stockhammer, and W. Xu. Crosslayer assisted reliability design for wireless multimedia broadcast. *EURASIP Signal Processing Journal, Special Issue:*

Advances in Signal Processing-assisted Cross-Layer Designs, pages 1933–1949, August 2006.

[JSX06c] H. Jenkač, T. Stockhammer, and W. Xu. Reliable wireless broadcast with asynchronous access: Data carousels versus fountain codes. In *Proceedings of 15. IST Mobile & Wireless Communications Summit 2006*, Myconos, Greece, June 2006.

[JSXAS06] H. Jenkač, T. Stockhammer, W. Xu, and W. Abdel-Samad. Efficient video-on-demand services over mobile datacast channels. *Journal of Zhejiang University SCIENCE A, Special Issue for Selected Papers (Part I) of 15th International Packet Video Workshop (PV2006)*, 7(5):873–884, May 2006.

[KG04] M. Kalman and B. Girod. Modeling the delays of successively–transmitted Internet packets. In *Proceedings IEEE International Conference on Multimedia and Expo*, Taipeh, Taiwan, June 2004.

[KHF04] E. Kohler, M. Handley, and S. Floyd. Datagram congestion control protocol (DCCP). Internet Draft, Work in Progress draft-ietf-dccp-spec-06.txt, Internet Engineering Task Force (IETF), February 2004.

[KK03] M. Karczewicz and R. Kurceren. The SP and SI frames design for H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7), July 2003.

[KKK02] C.W. Kim, D.W. Kang, and I.S. Kwang. High-complexity mode decision for error prone environment. Doc. JVT-C101, Joint Video Team (JVT), Fairfax(VA), USA, May 2002.

[Kno97] R. Knopp. *Coding and multiple-access over fading channels*. PhD thesis, EPFL, Lausanne, Switzerland, 1997.

[Kos91] V. Koshelev. Hierarchical coding of discrete sources. *IEEE Transactions on Information Theory*, 39(2):269–275, March 1991.

[KRG03] M. Kalman, P. Ramanathan, and B. Girod. Rate–distortion optimized streaming with multiple deadlines. In *Proceedings IEEE International Conference on Image Processing*, Barcelona, Spain, September 2003.

[KSB95] F. Kossentini, M.J.T. Smith, and C.F. Barnes. Image coding using entropy–constrained residual vector quantization. *IEEE Transactions on Image Processing*, 4(2):1347–1357, February 1995.

[KSG04] M. Kalman, E.G. Steinbach, and B. Girod. Adaptive media playout for low–delay video streaming over error–prone channels. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(6):841–851, June 2004.

[KXP00] B.-J. Kim, Z. Xiong, and W. A. Pearlman. Low bit–rate scalable video coding with 3D set partitioning in hierarchical trees (3D SPIHT). *IEEE Transactions on Circuits and Systems for Video Technology*, 10:1374–1387, December 2000.

[KZ02] S.H. Kang and A. Zakhor. Packet scheduling algorithm for wireless video streaming. In *Proceedings International Packet Video Workshop*, Pittsburgh(PY), USA, April 2002.

[LB01] L. Lastras and T. Berger. All sources are nearly successively refinable. *IEEE Transactions on Information Theory*, 47(3):918–926, March 2001.

[Lee90] D.H. Lee. *Asymptotic Quantization Error and Cell–Conditioned Two–Stage Vector Quantization*. PhD thesis, University of Michigan, Ann Arbor, USA, December 1990.

[LGP+03] Z. Li, W. Gao, F. Pan, S. Ma, K.P. Lim, G. Feng, X. Lin, S. Rahardja, H. Lu, and Y. Lu. Adaptive rate control with HRD consideration. Doc. JVT–H014, Joint Video Team (JVT), Geneva, Switzerland, May 2003.

[LGSW07] M. Luby, T. Gasiba, T. Stockhammer, and M. Watson. Reliable multimedia download delivery in cellular broadcast networks. *IEEE Transactions on Broadcasting*, 53(1):235–246, 2007.

[Li01] W. Li. Overview of fine granular scalability in the MPEG–4 video standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):385–398, March 2001.

[Lit99] S. Litsyn. New upper bounds on error exponents. *IEEE Transactions on Information Theory*, 45(3):385–398, March 1999.

[LJ83] S. Lin and D.J. Costello Jr. *Error Control Coding – Fundamentals and Applications*. Prentice Hall Book Co., Englewood Cliffs, New Jersey, USA, 1983.

[LJL+03] P. List, A. Joch, J. Lainema, G. Bjøntegaard, and M. Karczewicz. Adaptive deblocking filter. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):614–619, July 2003.

[LJSB04] G. Liebl, H. Jenkač, T. Stockhammer, and C. Buchner. Radio link buffer management and scheduling for video streaming over wireless shared channels. In *Proceedings International Packet Video Workshop*, Irvine, CA, USA, December 2004.

[LJSB05a] G. Liebl, H. Jenkač, T. Stockhammer, and C. Buchner. Joint buffer management and scheduling for wireless video streaming. In *Proceedings of the 4th International Conference on Networking (ICN 2005)*, pages 882–891, La Reunion, France, April 2005.

[LJSB05b] G. Liebl, H. Jenkač, T. Stockhammer, and C. Buchner. Radio link buffer management and scheduling for wireless video streaming. *Telecommunication Systems,*, 30(1-3):255–277, November 2005. Springer Science & Business Media B.V.

[LLC02] M.-J. Lin, H. Luo, and L.F. Chang. A linux-based EGPRS real-time test bed software for wireless QoS and differentiated service studies. In *Proceedings of the IEEE International Conference on Communications (ICC)*, volume 2, pages 1039–1044, 2002.

[LMSS01] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielmann. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47:569–584, February 2001.

[LOR98] T. V. Lakshman, A. Ortega, and A. R. Reibman. VBR video: Trade–offs and potentials. *Proceeding of the IEEE*, 86(5):952–973, May 1998.

[LRL93] W.M. Lam, A.R. Reibman, and B. Liu. Recovery of lost or erroneously received motion vectors. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, pages 417–420, March 1993.

[LS02] P. Leelapornchai and T. Stockhammer. Progressive image transmission applying multipath routing in mobile ad hoc networks. In *Proceedings International Conference on Image Processing (ICIP)*, volume 1, pages 553–556, 22-25 Sept. 2002.

[LSB00] G. Liebl, T. Stockhammer, and F. Burkert. Modelling and simulation of wireless packet erasure channels. In *PR 10th Virginia Tech/MRPG Symposium on Wireless Personal Communication*, Blacksburg (VA), USA, May 2000.

[LSNB01] G. Liebl, T. Stockhammer, T. Nguyen, and F. Burkert. Erasure-resilient multimedia streaming over packet radio networks. In *Proceedings of the 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001)*, pages 328–333, Orlando(FL), USA, July 2001.

[LSS02a] G. Liebl, T. Stockhammer, and P. Strasser. Dynamic multiplexing of IP-streams onto shared cellular links. In *Proceedings of the 12th Virginia Tech/MPRG Symposium on Wireless Personal Communications*, Blacksburg(VA), USA, June 2002.

[LSS02b] G. Liebl, T. Stockhammer, and P. Strasser. Priority-based multiplexing of ip-streams onto shared cellular links. In *Proc. IEEE Vehicular Technology Conference (VTC)*, pages 1091–1095, Birmingham(AL), USA, May 2002.

[LSSL02] G. Liebl, T. Stockhammer, P. Strasser, and D. Liebermann. A real-time simulation environment for IP-traffic over cellular links. In *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002)*, pages 42–47, Orlando (FL), USA, July 2002.

[LSW03] A. Luthra, G.J. Sullivan, and T. Wiegand, editors. *Special Issue on the H.264/AVC Video Coding Standard*, volume 13. July 2003.

[LSWS06] G. Liebl, T. Schierl, T. Wiegand, and T. Stockhammer. Advanced wireless multiuser video streaming using the scalable video coding extensions of H.264/MPEG4-AVC. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 625–628, July 2006.

[Lub02] M. Luby. LT codes. In *PR 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[LV96] J. Liao and J. Villasenor. Adaptive intra update for video coding over noisy channels. In *Proceedings IEEE International Conference on Image Processing*, volume 3, pages 763–766, October 1996.

[LVG+02] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft. The use of forward error correction (FEC) in reliable multicast. Request for Comments (standard) 3453, Internet Engineering Task Force (IETF), December 2002.

[LWG+06] M. Luby, M. Watson, T. Gasiba, T. Stockhammer, and W. Xu. Raptor codes for reliable download delivery in wireless broadcast systems. In *Consumer Communications and Networking Conference, 2006. CCNC 2006. 2006 3rd IEEE*, volume 1, pages 192–197, 8-10 January 2006.

[LWGS07] M. Luby, M. Watson, T. Gasiba, and T. Stockhammer. High-quality video distribution using power line communication and aplication layer forward error correction. In *Proceedings IEEE International Symposium on Power Line Communications and Its Applications (ISPLC)*, pages 431–436, 26-28 March 2007.

[LWPW04] G. Liebl, M. Wagner, J. Pandel, and W. Weng. An RTP payload format for erasure-resilient transmission of progressive multimedia streams. Internet Draft, Work in Progress draft-ietf-avt-uxp-06.txt, Internet Engineering Task Force (IETF), February 2004.

[LWSS07] M. Luby, M. Watson, A. Shokrollahi, and T. Stockhammer. Raptor forward error correction scheme for object delivery. Request for Comments (standard) 5053, Internet Engineering Task Force (IETF), October 2007.

[Mas72] J.L. Massey. Variable-length codes and the Fano metric. *IEEE Transactions on Information Theory*, 18(1):196–198, January 1972.

[MBM03] *Multimedia Broadcast/Multicast Service (MBMS); UTRAN/GERAN Requirements*, number TR25.992-140, 2003.

[MBM05] Multimedia multicast and broadcast service (MBMS); protocols and codecs. 3GPP Technical Recommendation TS26.346, 3GPP, September 2005.

[MHKL03] M.S. Malvar, A. Hallapuro, M. Karczewicz, and L. Lerofsky. Low–complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):598–603, July 2003.

[MM00] D. Mukherjee and S. K. Mitra. A vector set partitioning noisy channel image coder with unequal error protection. *IEEE Journal on Selected Areas in Communications*, 18(6):829–840, June 2000.

[MMS02] Multimedia messaging service (MMS); media formats and codecs. 3GPP Technical Recommendation TR26.140, 3GPP, December 2002.

[MO77] R.J. McEliece and J.K. Omura. An improved upper bound on the block coding error exponent for binary-input discrete memoryless channels. *IEEE Transactions on Information Theory*, 23:611–613, September 1977.

[MPG] ISO/IEC JTC 1. *Coding of audio-visual objects, Part 2: Visual*, number ISO/IEC 14496-2(MPEG-4 visual version 1). Apr. 1999; Amendment 1 (version 2), Feb., 2000; Amendment 4 (streaming profile), January, 2001.

[MPG93] ISO/IEC JTC 1. *Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s, Part 2: Video*, number 11172-2(MPEG–1), March 1993.

[MPG94] ITU–T and ISO/IEC JTC 1. *Generic Coding of Moving Pictures and Associated Audio Information, Part 2: Video*, number H.262 – ISO/IEC 13818-2(MPEG-2 Video), November 1994.

[MRL00] A.E. Mohr, E.A. Riskin, and R.E. Ladner. Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction. *IEEE Journal on Selected Areas in Communications*, 18(6):819–828, June 2000.

[MS01] D. Marpe and T. Stockhammer. Test model document changes for data partitioning and NAL support. Technical Report VCEG-L52, ITU-T SG.16, Austin(TX), USA, April 2001.

[MS02a] M. Mecking and T. Stockhammer. Minimizing distortion via multiuser resource allocation. In *Data Compression Conference, 2002. Proceedings. DCC 2002*, page 464, 2-4 April 2002.

[MS02b] M. Mecking and T. Stockhammer. Source–controlled resource allocation. In *Proceedings ITG Conference Source and Channel Coding*, Berlin, Germany, January 2002.

[MSW03] D. Marpe, H. Schwarz, and T. Wiegand. Context–based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, July 2003.

[MTS07] Multimedia subsystem (IMS); multimedia telephony; media handling and interaction. 3GPP Technical Recommendation TS26.114, 3GPP, March 2007.

[MU49] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44:335–341, 1949.

[MVJ97] S. McCanne, M. Vetterli, and V. Jacobson. Low–complexity video coding for receiver–driven layered multicast. *IEEE Journal on Selected Areas in Communications*, 15:983–1001, August 1997.

[NT96] T. Nakai and Y. Tomita. Core experiments on feedback channel operation for H.263+. Doc. LBC96-308, ITU-T SG15, November 1996.

[OBL+04] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi. Video coding with H.264/AVC: Tools, Performance and Complexity. *IEEE Circuits and Systems Magazine*, pages 7–28, April 2004.

[Ohm94] J. Ohm. Three-dimensional subband coding with motion compensation. *IEEE Transactions on Image Processing*, 3:559–571, September 1994.

[OOKF68] Y. Okumura, E. Ohmori, T. Kawano, and K. Fukada. Field strength and its variability in the VHF and UHF land mobile radio service. *Rewiew Electrical Communications Laboratory*, 16(9–10):825–873, 1968.

[OR98] A. Ortega and K. Ramchandran. Rate–distortion methods in image and video compression. *IEEE Signal Processing Magazine*, 15(6):23–50, November 1998.

[OS94] M.T. Orchard and G.J. Sullivan. Overlapped block motion compensation: An estimation–theoretic approach. *IEEE Transactions on Image Processing*, 3(5):693–699, September 1994.

[OWS+04] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Rey. Extended RTP profile for RTCP-based feedback (RTP/AVPF). Internet Draft, Work in Progress draft-ietf-avt-rtcp-feedback-08.txt, Internet Engineering Task Force (IETF), January 2004.

[P9196] ITU–T. *Subjective Video Quality Assessment Methods for Multimedia Applications*, number P.910, August 1996.

[Pap91] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw–Hill Book Co., New York, USA, 1991.

[PM93] W.B. Pennebaker and J.L. Mitchell. *JPEG: Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, USA, 1993.

[Pos80a] J. Postel. DoD standard transmission control protocol. Request for Comments (standard) 761, Internet Engineering Task Force (IETF), January 1980.

[Pos80b] J. Postel. User datagram protocol. Request for Comments (standard) 768, Internet Engineering Task Force (IETF), August 1980.

[Pos81] J. Postel. Internet protocol. Request for Comments (standard) 791, Internet Engineering Task Force (IETF), September 1981.

[PR85] J. Postel and J.K. Reynolds. File transfer protocol. Request for Comments (standard) 959, Internet Engineering Task Force (IETF), October 1985.

[PR99] R. Puri and R. Ramchandran. Multiple description source coding through forward error correction. In *Proceedings Asilomar Conference for Signals, Systems, and Computers*, volume 1, pages 342–346, October 1999.

[Pro95] J. G. Proakis. *Digital Communications*. McGraw–Hill Book Co., New York, USA, 1995.

[PS01] I. M. Pao and M. T. Sun. Encoding stored video for streaming applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(2):199–209, February 2001.

[PSC02] Packet switched conversational multimedia applications; default codecs. 3GPP Technical Recommendation TS26.235, 3GPP, December 2002.

[PSS02a] Packet switched conversational multimedia applications; default codecs. 3GPP Technical Recommendation TS26.235, 3GPP, December 2002.

[PSS02b] Transparent end-to-end packet switched streaming service (PSS); RTP usage model. 3GPP Technical Recommendation TR26.937, 3GPP, December 2002.

[RCCR03] J. Ribas-Corbera, P.A. Chou, and S.L. Regunathan. A generalized hypothetical reference decoder for H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):674–687, July 2003.

[RCS99] J. Ribas-Corbera and S.Lei. Rate control in DCT video coding for low-delay communications. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(1):172–185, February 1999.

[Rie97] S. Riedl. *Iterative Decodierung parallel verketteter binärer Faltungscodes*. PhD thesis, Munich University of Technology, Munich, Germany, January 1997.

[Rim94] B. Rimoldi. Successive refinement of information: Characterization of achievable rates. *IEEE Transactions on Information Theory*, 40(1):253–259, January 1994.

[RLM+04] J. Rey, D. Leon, A. Miyazaki, V. Varsa, and R. Hakenberg. RTP retransmission payload format. Internet Draft, Work in Progress draft-ietf-avt-rtp-retransmission-10.txt, Internet Engineering Task Force (IETF), January 2004.

[RM00] D.N. Rowitch and L.B. Milstein. On the performance of hybrid FEC/ARQ systems using rate compatable punctured turbo (RCPT) codes. *IEEE Transactions on Communications*, 48(6):948–959, June 2000.

[RS99] J. Rosenberg and H. Schulzrinne. An RTP payload format for generic forward error correction. Request for Comments (standard) 2733, Internet Engineering Task Force (IETF), December 1999.

[RSL+01] G. Roth, R. Sjöberg, G. Liebl, T. Stockhammer, V. Varsa, and M. Karczewicz. Common test conditions for RTP/IP over 3GPP/3GPP2. Doc. VCEG-N80, ITU-T SG16/Q6 Video Coding Experts Group (VCEG), Santa Barbara(CA), USA, September 2001.

[RVH95] P. Robertson, E. Villebrun, and P. Höher. A comparison of optimal and sub–optimal MAP decoding algorithms operating in the Log domain. In *Proceedings IEEE International Conference on Communications (ICC)*, pages 1009–1013, Seattle(WS), USA, June 1995.

[SAR00] D.G. Sachs, R. Anand, and R. Ramchandran. Wireless image transmission using multiple–description based concatenated codes. In *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, volume 3974, pages 300–311, January 2000.

[Sav66] J.E. Savage. The distribution of sequential decoding computation time. *IEEE Transactions on Information Theory*, 12:145–147, April 1966.

[SB91] G.J. Sullivan and R.L. Baker. Motion compensating for video compression using control grid interpolation. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2713–2716, Toronto, Canada, May 1991.

[SB01] T. Stockhammer and C. Buchner. Progressive texture video streaming for packet lossy networks. In *Proceedings International Packet Video Workshop*, Kyongju, Korea, May 2001.

[SB04] T. Stockhammer and M. Bystrom. H.264/AVC data partitioning for mobile video communication. In *Proceedings International Conference on Image Processing (ICIP)*, volume 1, pages 545–548, 24-27 Oct. 2004.

[SBL00] T. Stockhammer, F. Burkert, and G. Liebl. Modeling and simulation of wireless packet erasure channels. In *Proceedings of the 10th Virginia Tech/MPRG Symposium on Wireless Personal Communications*, pages 203–214, Blacksburg (VA), USA, June 2000.

[SBPP00a] T. Stockhammer, G. Bäse, J. Pandel, and S. Purreiter. Data partitioning test software. Technical Report Q.15-K-17, ITU-T SG.16, Portland(OR), USA, August 2000.

[SBPP00b] T. Stockhammer, G. Bäse, J. Pandel, and S. Purreiter. Network adaptation layer - requirements and concepts. Technical Report Q.15-K-19, ITU-T SG.16, Portland (OR), USA, August 2000.

[SCFJ03] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real–time applications. Request for Comments (standard) 3550, Internet Engineering Task Force (IETF), July 2003.

[Sch01] H. Schulzrinne. IP networks. In M.T. Sun and A.R. Reibman, editors, *Compressed Video over Networks*, pages 81–137. Marcel–Dekker, New York (NY), USA, 2001.

[SFG97] E. Steinbach, N. Färber, and B. Girod. Standard compatible extension of H.263 for robust video transmission in mobile environments. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(6):872–881, December 1997.

[SG88] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36:1445–1453, September 1988.

[SG05] E. Setton and B. Girod. Video streaming with SI and SP frames. In *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, July 2005.

[SGAS+06] T. Stockhammer, T. Gasiba, W. Abdel-Samad, W. Xu, H. Jenkač, and T. Schierl. A weighted layered broadcasting scheme for scalable video transmission with multiple site reception. In *Proceedings of the Mobimedia 2006*, Alghero, Sardinia, Italy, September 2006.

[SGS+07] T. Stockhammer, T. Gasiba, W. Abdel Samad, T. Schierl, H Jenkač, T. Wiegand, and W. Xu. Nested harmonic broadcasting for scalable video over mobile datacast channels. *Wireless Communications and Mobile Computing, Special Issue on Video Communications for 4G Wireless Systems*, 7(2):235–256, February 2007.

[SGWS06] T. Schierl, K. Gänger, T. Wiegand, and T. Stockhammer. SVC-based multisource streaming for robust video transmission in mobile ad hoc networks. *IEEE Wireless Communications Magazine, Special Issue on Multimedia in Wireless/Mobile ad hoc Networks*, October 2006.

[SH05] T. Stockhammer and Miska M. Hannuksela. H.264/AVC video for wireless transmission. *IEEE Wireless Communications*, 12(4):6–13, August 2005.

[Sha48] C.E. Shannon. A Mathematical Theory of Communications. *Bell Systems Technology Journal*, pages 379–423, 623–656, 1948.

[Sha93a] C.E. Shannon. *Collected Papers*. IEEE Press, New York (NY), USA, 1993.

[Sha93b] J.M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *Springer-Verlag*, 41:3445–3462, December 1993.

[SHG+06] T. Schierl, C. Hellge, K. Ganger, T. Stockhammer, and T. Wiegand. Multi source streaming for robust video transmission in mobile ad-hoc networks. In *Proceedings International Conference on Image Processing (ICIP)*, pages 1669–1672, 8-11 Oct. 2006.

[Sho06] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567, June 2006.

[SHW02] T. Stockhammer, M.M. Hannuksela, and S. Wenger. H.26L/JVT coding network abstraction layer and IP-based transport. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages 485–488, 22-25 Sept. 2002.

[SHW03] T. Stockhammer, M.M. Hannuksela, and T. Wiegand. H.264/AVC in wireless environments. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):657–673, July 2003.

[SHX02] V. Stankovic, R. Hamzaoui, and Z. Xiong. Packet loss protection of embedded data with fast local search. In *Proceedings IEEE International Conference on Image Processing*, September 2002.

[SHX04a] V. Stankovic, R. Hamzaoui, and Z. Xiong. Efficient channel code rate selection for forward error correction of packetized multimedia bitstreams varying channels. *IEEE Transactions on Multimedia*, 6(2):240–249, April 2004.

[SHX04b] V. Stankovic, R. Hamzaoui, and Z. Xiong. Real–time error protection of embedded codes for packet erasure and fading channels. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(8):1064–1072, August 2004.

[SJH+07] T. Schierl, S. Johansen, C. Hellge, T. Stockhammer, and T. Wiegand. Distributed rate-distortion optimization for rateless coded scalable video in mobile ad hoc networks. In *Proceedings International Conference on Image Processing (ICIP)*, volume 6, pages VI–497–VI–500, Sept. 16 2007-Oct. 19 2007.

[SJK04] T. Stockhammer, H. Jenkač, and G. Kuhn. Streaming video over variable bit-rate wireless channels. *IEEE Transactions on Multimedia*, 6(2):268–277, April 2004.

[SJW02a] T. Stockhammer, H. Jenkač, and C. Weiss. Error control for wireless progressive video transmission. In *Proceedings International Conference on Image Processing (ICIP)*, volume 1, pages 545–548, 22-25 Sept. 2002.

[SJW02b] T. Stockhammer, H. Jenkač, and C. Weiß. Feedback and error protection strategies for wireless video transmission. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):465–482, July 2002.

[SK91] T. Socolofsky and C. Kale. A TCP/IP tutorial. Request for Comments (standard) 1180, Internet Engineering Task Force (IETF), January 1991.

[SK02] T. Stockhammer and D. Kontopodis. Error robust macroblock mode and reference frame restriction. Doc. JVT-B102, Joint Video Team (JVT), Geneva, Switzerland, January 2002.

[SL07] T. Stockhammer and G. Liebl. On practical crosslayer aspects in 3GPP video services. In *Proceeding of ACM Multimedia 2007 - International Workshop on Mobile Video*, Augsburg, Germany, October 2007.

[SLJ+03a] T. Stockhammer, G. Liebl, H. Jenkač, P. Strasser, D. Pfeifer, and J. Hagenauer. WiNe2 - wireless network demonstration platform for ip-based real-time multimedia transmission. In *Proceedings of the 13th International Packet Video Workshop (PVW 2003)*, Nantes, France, April 2003.

[SLJ+03b] T. Stockhammer, G. Liebl, H. Jenkač, P. Strasser, D. Pfeifer, J. Hagenauer, T. Wiegand, and T. Hinz. Real-time demonstration of MPEG-4 based video telephony over wireless systems using WiNe2. In *Proceedings of the 8th International Workshop on Mobile Multimedia Communications*, October 2003.

[SLK+07] T. Stockhammer, G. Liebl, S. Kaiser, H Jenkač, and M. Ruf. "Daumenkino" gets reality. *European Transactions on Telecommunications (ETT)*, 18(8):865–872, May 2007.

[SLW06] T. Stockhammer, G. Liebl, and M. Walter. Optimized H.264/AVC-based bit stream switching for mobile video streaming. *EURASIP Journal on Applied Signal Processing*, 2006(Article ID 91797):19, April 2006.

[SLW08] T. Stockhammer, M. Luby, and M. Watson. Application layer FEC in IPTV services. *IEEE Communications Magazine*, 45(5):94–101, May 2008.

[SM01] T. Stockhammer and D. Marpe. Test model document changes for data partitioning and NAL support. Doc. VCEG-M52, ITU-T SG16/Q6 Video Coding Experts Group (VCEG), Austin (TX), USA, April 2001.

[SM03] T. Stockhammer and M. Mecking. Codierungs- und Übertragungsaspekte für mobile Videoanwendungen. *Information Technology*, 5:285–292, 2003.

[SOW94] S. Shamai (Shitz), L. Ozarow, and A.D. Wyner. Information theoretic considerations for cellular mobile radio. *IEEE Transactions on Vehicular Technology*, 43:359–378, May 1994.

[SOW02] T. Stockhammer, T. Oelbaum, and T. Wiegand. H.26L/JVT coded video transmisson in 3g wireless environments. In *Proceedings of the International Conference on Third Generation Wireless and Beyond (3GWireless)*, San Francisco(CA), USA, May 2002.

[SP96] A. Said and W. A. Pearlman. A new, fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:243–250, June 1996.

[SRC84] J.H. Saltzer, D.P. Reed, and D.D. Clark. End–to–end arguments in system design. *ACM Transactions on Computer Systems*, 2:277–288, November 1984.

[SRL98] H. Schulzrinne, A. Rao, and R. Lanphier. Real–time streaming protocol (RTSP). Request for Comments (standard) 2326, Internet Engineering Task Force (IETF), April 1998.

[SS03] H. Schulzrinne and Casner S. RTP profile for audio and video conferences with minimal control. Request for Comments (standard) 3551, Internet Engineering Task Force (IETF), July 2003.

[SSD98] P. Salama, N.B. Shroff, and E.J. Delp. Error concealment in encoded video. In *Image Recovery Techniques for Image Compression Applications*. Kluwer Academic Press, Dordrecht, The Netherlands, 1998.

[SSW07] T. Schierl, T. Stockhammer, and T. Wiegand. Mobile video transmission using scalable video coding (SVC). *IEEE Transactions on Circuits and Systems for Video Technology*, 17:1204–1217, June 2007.

[SSW+08] T. Stockhammer, A. Shokrollahi, M. Watson, M. Luby, and T. Gasiba. *Application Layer Forward Error Correction for Mobile Multimedia Broadcasting*. CRC Press, Taylor and Francis Group, March 2008.

[SSXM06] E. Seif, T. Stockhammer, W. Xu, and P. Mähönen. Scalability of mobile uplink channels for interactive TV applications. In *Proceedings of the Mobimedia 2006*, Alghero, Sardinia, Italy, September 2006.

[Sto01] T. Stockhammer. UEP – Useful or Useless Error Protection? In *Proceedings Joint Conference on Coding and Communications (JCCC)*, Bad Kleinkirchheim, Austria, March 2001.

[Sto02] T. Stockhammer. Progressive video transmission for packet lossy channels exploiting feedback and unequal erasure protection. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages 169–172, 22-25 Sept. 2002.

[Sto03] T. Stockhammer. Is fine-granular scalable video coding beneficial for wireless video applications? In *International Conference on Proceedings of the Multimedia and Expo (ICME)*, volume 1, pages 193–196, 6-9 July 2003.

[Sto06] T. Stockhammer. Robust system and cross-layer design for H.264/AVC-based wireless video applications. *EURASIP Journal on Applied Signal Processing*, 2006(Article ID 89371):15, March 2006.

[SW98] G.J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, November 1998.

[SW01] T. Stockhammer and C. Weiss. Channel and complexity scalable image transmission. In *Proceedings International Conference on Image Processing (ICIP)*, volume 1, pages 102–105, 7-10 Oct. 2001.

[SW02] T. Stockhammer and S. Wenger. Standard compliant enhancements of jvt coded video over fixed and wireless ip. In *Proceedings of 2002 International Tyrrhenian Workshop on Digital Communications (IWDC 2002)*, Capri, Italy, September 2002.

[SW05] G. Sullivan and T. Wiegand. Video compression – from concepts to the H.264/AVC standard. *Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery*, 93(1):18–31, January 2005.

[SWH00] T. Stockhammer, C. Weiss, and J. Hagenauer. Regressive channel coding with sequential decoding for embedded source coders. In *Information Theory, 2000. Proceedings. IEEE International Symposium on*, page 228, 25-30 June 2000.

[SWH02] T. Stockhammer, S. Wenger, and M.M. Hannuksela. H.26L/JVT coding network abstraction layer and IP-based transport. In *Proceedings IEEE International Conference on Image Processing*, Rochester (NY), USA, September 2002.

[SWK02] T. Stockhammer, T. Wiegand, and D. Kontopodis. Rate-distortion optimization for JVT/H.26L coding in packet loss environment. In *Proceedings International Packet Video Workshop*, Pittsburgh(PY), USA, April 2002.

[SWL05] T. Stockhammer, M. Walter, and G. Liebl. Optimized H.264-based bitstream switching for wireless video streaming. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1396–1399, 6-8 July 2005.

[SWL08] T. Stockhammer, M. Watson, and M. G. Luby. DVB application layer fec in IPTV services. In *Proceeding of the SMPTE & VSF 2008 Joint Conference*, Houston (TX), USA, February 2008.

[SWOO03] T. Stockhammer, T. Wiegand, T. Oelbaum, and F. Obermeier. Video coding and transport layer techniques for H.264/AVC-based transmission over packet-lossy networks. In *Proceedings International Conference on Image Processing (ICIP)*, volume 3, pages 481–484, 14-17 Sept. 2003.

[SWS01] G. Sullivan, T. Wiegand, and T. Stockhammer. Using the draft H.26L video coding standard for mobile applications. In *Proceedings International Conference on Image Processing (ICIP)*, Thessaloniki, Greece, October 2001.

[SWW02] T. Stockhammer, T. Wiegand, and S. Wenger. Optimized transmission of H.26L/JVT coded video over packet-lossy networks. In *Proceedings International Conference on Image Processing (ICIP)*, volume 2, pages 173–176, 22-25 Sept. 2002.

[SZ98] P.G. Sherwood and K. Zeger. Error protection for progressive image transmission over memoryless and fading channels. *IEEE Transactions on Communications*, 46(12):1555–1559, December 1998.

[SZ01] T. Stockhammer and K. Zeger. Distortion bounds and channel code rates for progressive quantization. In *Proceedings IEEE International Symposium on Information Theory (ISIT).*, page 263, 24-29 June 2001.

[SZ04] T. Stockhammer and K. Zeger. Distortion bounds on progressive quantization and unequal error protection. Technical report, October 2004. Internal Technical Report at University of California, San Diego (UCSD).

[SZ07] T. Stockhammer and W. Zia. *Error Resilient Coding and Decoding Strategies for Video Communications*. Elsevier Academic Press, march 2007. In: Multimedia Over IP and Wireless Networks: Compression, Networking, and Systems,.

[Tau03] D. Taubman. Successive refinement of video: Fundamental issues, past efforts and new directions. In *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, pages 649–663, Lugano, Switzerland, July 2003.

[TH96] T. Turletti and C. Huitema. RTP payload format for H.261 video streams. Request for Comments (standard) 2032, Internet Engineering Task Force (IETF), October 1996.

[THG05] D. Tian, M.M. Hannuksela, and M. Gabbouj. Sub-sequence video coding for improved temporal scalability. In *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, Kobe, Japan, May 2005.

[TKI97] Y. Tomita, T. Kimura, and T. Ichikawa. Error resilient modified inter–frame coding system for limited reference picture memories. In *Proceedings Picture Coding Symposium*, Berlin, Germany, September 1997.

[TR03a] E. Tuncel and K. Rose. Additive successive refinement. *IEEE Transactions on Information Theory*, 49(8):1983–1991, August 2003.

[TR03b] E. Tuncel and K. Rose. Computation and analysis of the n-layer scalable rate-distortion function. *IEEE Transactions on Information Theory*, 49(5):1218–1230, May 2003.

[TR03c] E. Tuncel and K. Rose. Error exponents in scalable source coding. *IEEE Transactions on Information Theory*, 49(1):289–296, January 2003.

[TZ94] D. Taubman and A. Zakhor. Multirate 3–D subband coding of video. *IEEE Transactions on Image Processing*, 3:572–584, September 1994.

[vdSR02] M. van der Schaar and H. Radha. Adaptive motion–compensated fine–granular–scalability (AMC-FGS) for wireless video. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):360–371, June 2002.

[VHW01] V. Varsa, M.M. Hannuksela, and Y.-K. Wang. Non-normative error concealment algorithms. Doc. VCEG-N62, ITU-T SG16/Q6 Video Coding Experts Group (VCEG), Santa Barbara(CA), USA, September 2001.

[Vit67] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, April 1967.

[VK01] V. Varsa and M. Karczewicz. Long window rate control for video streaming. In *Proceedings International Packet Video Workshop*, Kyongju, Korea, May 2001.

[VO79] A.J. Viterbi and J.K. Omura. *Principles of Digital Communications*. McGraw–Hill Book Co., New York, USA, 1979.

[VS07] I. Varga and T. Stockhammer. Enrichment of speech calls by live video. In *Proceedings of the Sarnoff Symposium*, Princeton (NJ), USA, April 2007.

[Wad89] W. Wada. Selective recovery of video packet losses using error concealment. *IEEE Journal on Selected Areas in Communications*, 7:807–814, June 1989.

[Wei04] C. Weiß. *Iterative Decoding of Tailbiting Codes*. PhD thesis, Munich University of Technology, Munich, Germany, January 2004.

[Wen99] S. Wenger. Error patterns for internet experiments. Doc. Q15-I-16r1, ITU-T Q.15/SG16, Redbank(NJ), USA, October 1999.

[Wen01] S. Wenger. Common conditions for wireline, low delay IP/UDP/RTP packet loss resilient testing. Doc. VCEG-N79, ITU-T SG16/Q6 Video Coding Experts Group (VCEG), Santa Barbara(CA), USA, September 2001.

[Wen03] S. Wenger. H.264/AVC over IP. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):645–656, July 2003.

[WFSG00] T. Wiegand, N. Färber, K. Stuhlmüller, and B. Girod. Error–resilient video transmission using long-term memory motion–compensated prediction. *IEEE Journal on Selected Areas in Communications*, 18(6):1050–1062, June 2000.

[WG01] T. Wiegand and B. Girod. Lagrangian multiplier selection in hybrid video coder control. In *Proceedings IEEE International Conference on Image Processing*, Thessaloniki, Greece, October 2001.

[WHS01] S. Wenger, M.M. Hannuksela, and T. Stockhammer. Identied H.26L applications. Doc. VCEG-L34d1, ITU-T SG16/Q6 Video Coding Experts Group (VCEG), Eibsee, Germany, January 2001.

[WHS+05] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer. RTP payload format for H.264 video. Request for Comments (standard) 3984, Internet Engineering Task Force (IETF), February 2005.

[WHV+02] Y.-K. Wang, M.M. Hannuksela, V. Varsa, A. Hourunranta, and M. Gabbouj. The error concealment feature in the H.26L test model. In *Proceedings IEEE International Conference on Image Processing*, volume 2, pages 729–732, Rochester(NY), USA, September 2002.

[WLM+96] T. Wiegand, M. Lightstone, D. Mukherjee, T.G. Campbell, and S.K. Mitra. Rate–distortion optimized mode selection for very low bit rate video coding and the emerging H.263 standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:182–190, April 1996.

[WLZ01] F Wu, S. Li, and Y.Q. Zhang. A framework for efficient progressive fine granularity scalable video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):332–344, March 2001.

[WM03] T. Wedi and H.G. Musmann. Motion- and aliasing-compensated prediction for hybrid video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):577–586, July 2003.

[WR61] J.M. Wozencraft and B. Riefen. *Sequential Decoding*. MIT Press, Cambridge (MA), USA, 1961.

[WRH96] C. Weiß, S. Riedel, and J. Hagenauer. Sequential decoding using a–priori information. *Electronic Letters*, pages 1190–1191, June 1996.

[WS01] S. Wenger and T. Stockhammer. H.26L over IP and H.324 framework. Technical Report VCEG-N52, ITU-T SG.16, Santa Barbara(CA) USA, September 2001.

[WS02] S. Wenger and T. Stockhammer. Overview of NAL concept and VCL/NAL interface. Doc. JVT-B028, Joint Video Team (JVT), Geneva, Switzerland, January 2002.

[WSBL03] T. Wiegand, G.J. Sullivan, G. Bjøntegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003.

[WSDH01] C. Weiß, T. Stockhammer, A. Donner, and J. Hagenauer. Adaptive channel coding for mobile channels - or why wasting bandwidth for error detection? In *Proceedings of the International Conference on Information, Communications, and Signal Processing*, Singapore, Singapore, October 2001.

[WSH01] C. Weiss, T. Stockhammer, and J. Hagenauer. The far end error decoder with application to image transmission. In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 2, pages 1405–1409, 25-29 Nov. 2001.

[WSJ+03] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G.J. Sullivan. Rate–constrained coder control and comparison of video coding standards. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):688–703, July 2003.

[WWZ80] J.K. Wolf, A.D. Wyner, and J. Ziv. Source coding for multiple descriptions. *Bell System Technical Journal*, 59:1417–1426, October 1980.

[WZ98] Y. Wang and Q. Zhu. Error control and concealment for video communication: A review. *Proceeding of the IEEE*, 86:974–997, May 1998.

[WZG99] T. Wiegand, X. Zhang, and B. Girod. Long–term memory motion–compensated prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(1):70–84, February 1999.

[XGS+05] W. Xu, T. Gasiba, T. Stockhammer, H. Jenkač, and G. Liebl. Iterative decoding for GERAN MBMS. In *Proceedings IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, volume 3, pages 1724–1728Vol.3, 11-14 Sept. 2005.

[XZ05] B. Xie and W. Zeng. On the rate-distortion performance of dynamic bitstream switching mechanisms. In *Proceedings IEEE International Conference on Multimedia and Expo*, August 2005.

[YASSZ07] A. Younus, W. Abdel-Samad, T. Stockhammer, and W. Zia. Comparison of mobility scenarios for delivery of dynamic interactive multimedia scenes in mobile broadcast environments. In *Proceedings of the 2007 International Conference on Wireless Networks*, Las Vegas (NV), USA, June 2007.

[YR03] H. Yang and K. Rose. Source–channel prediction in error–resilient video coding. In *Proceedings IEEE International Conference on Multimedia and Expo*, Baltimore, MD, USA, July 2003.

[YSS+07] A. Younus, W.A. Samad, T. Stockhammer, W. Zia, and W. Xu. Dynamic interactive multimedia scenes in mobile broadcast environments. In *Proceedings IEEE International Conference on Communications (ICC)*, pages 1833–1838, 24-28 June 2007.

[Zad82] P. Zador. Asymptotic quantization error of continuous signals and the quantization dimension. *IEEE Transactions on Information Theory*, 28(2):139–149, March 1982. previously an unpublished Bell Laboratories memorandum, "Topics in Asymptotic Quantization of Random Variables," 1966.

[ZAX+07] W. Zia, T. Afzal, W. Xu, G. Liebl, and T. Stockhammer. Interactive error control for mobile video telephony. In *Proceedings IEEE International Conference on Communications (ICC)*, pages 1797–1802, 24-28 June 2007.

[ZDS07] W. Zia, K. Diepold, and T. Stockhammer. Complexity constrained robust video transmission for hand-held devices. In *Proceedings International Conference on Image Processing (ICIP)*, volume 4, pages 261–264, September 2007.

[Zhu97] C. Zhu. RTP payload format for H.263 video streams. Request for Comments (standard) 2190, Internet Engineering Task Force (IETF), September 1997.

[Zim80] H. Zimmermann. OSI reference model – the ISO model of architecture for open system interconnection. *IEEE Transactions on Communications*, 28:425–432, 1980.

[ZK99] Q.F. Zhu and L Kerofsky. Joint source coding, transport processing, and error concealment for H.323-based packet video. In *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, volume 3653, pages 52–62, San Jose(CA), USA, January 1999.

[ZM94] K. Zeger and V. Manzella. Asymptotic bounds on optimal noisy channel quantization via random coding. *IEEE Transactions on Information Theory*, 40(6):1926–1938, November 1994.

[ZRR00] R. Zhang, S.L. Regunthan, and K. Rose. Video coding with optimal inter/intra-mode switching for packet loss resilience. *IEEE Journal on Selected Areas in Communications*, 18(6):966–976, June 2000.

Supervised Diploma and Master Theses

[Afz05] J. Afzal. System design and advanced receiver techniques for mbms streaming services. Master's thesis, Munich University of Technology (TUM), Munich, Germany, October 2005.

[Aym01] M. Aymeric. Effiziente Codierung und Suche von bewegungsbasierten Videodeskriptoren in MPEG-7. Diploma thesis, ENSIEG, Grenoble, France, August 2001.

[Blo03] F. Bloemer. Scheduling for streaming users in high speed downlink packet accesss (HSDPA). Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 2003.

[Bru04] P. Brun. Subjektive Beurteilung des H.264/AVC-Coders für niederratige Bildsequenzen. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, March 2004.

[Buc00] C. Buchner. Progressive video coding for error–prone channels. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, October 2000.

[Ded98] F. Dedek. Fehlerrobuste H.263-Videocodierung. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, May 1998.

[Den98] W. Denk. Sequentielle Kanaldecodierung von Huffman-codierter Bildern. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, April 1998.

[Don99] A. Donner. FEED - Angepasster Fehlerschutz für hochkomprimierte Quellen. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, May 1999.

[Eff00] T. Effern. Combined sequential source and channel decoding. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, February 2000.

[Fin03] M. Findeli. MPEG-4 based video streaming over UMTS. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, 02 2003.

[Gar99] C. Gardumi. MPEG4-Videoübertragung über GSM. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, May 1999.

[Glo00] C. Glomb. GSM-GPRS: Modellierung und evaluierung des paketdienstes in verschiedenen szenarien. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, January 2000.

[Gri99] O. Grimm. Kombinierte Quellen- und Kanalcodierung für MPEG4-Video über UMTS. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, January 1999.

[Haa04] H. Haas. Causal transmission of correlated sources over packet lossy channels. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, January 2004.

[Hof01] P. Hofmann. Statistical resource allocation for video transmission over the multiuser system UMTS. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, July 2001.

[Jak97] T. Jakob. Entwicklung von Methoden zur Fehlernachbearbeitung im MPEG4-Videodecoder. Master's thesis, Munich University of Technology (TUM), Munich, Germany, August 1997.

[Jen01] H. Jenkač. Feedback and error protection strategies for wireless video transmission. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, October 2001.

[Kam04] H. Kamoun. Link state communication for video streaming in ad hoc networks. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, November 2004.

[Kar03] M. Karl. Scalability and data partitioning in H.264-based conversational wireless video applications. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 2003.

[Kon01] D. Kontopodis. Error resilient video coding for packet switched networks. Master's thesis, Munich University of Technology (TUM), Munich, Germany, October 2001.

[Lie99] G. Liebl. Modeling, theoretical analysis and coding for wireless packet erasure channels. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 1999.

[Lie01] D. Liebermann. Dynamische Steuerung eines Paketverlustdemonstrators mittels extern gemessener Kanalstatistiken. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, September 2001.

[Moz02] A. Agustin Diaz-Pines Lopez De Los Mozos. Enhanced reference picture selection and data partitioning for scalability in H.26L. Diploma thesis, Universidad Politecnica de Madrid, Madrid, Spain, October 2002.

[Muf00] V. Mufisev. Erasure-resilient video-streaming over packet radio networks. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, November 2000.

[Ned00] T. Neder. Sequentielle Decodieralgorithmen in verketteten Codierschemata. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, February 2000.

[Obe02] F. Obermeier. H.264 video in IP-based conversational applications. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, November 2002.

[Oel01] T. Oelbaum. H.26L for mobile applications. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 2001.

[Ohl98] T. Ohl. Kombinierte Video- und Kanalcodierung für Mobilkanäle mit Paketverlusten. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, July 1998.

[Per99] C. Perianu. Angepaßte Kanalcodierung für sequentielle Decodierung im GSM-System. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, May 1999.

[Pfe03] D. Pfeifer. Resource-optimized video streaming over mobile systems. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, April 2003.

[Pie99] E. Pielmeier. Vergleich von optimalen und suboptimalen CDMA-Empfängern. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, February 1999.

[Pur01] S. Purreiter. Verbesserte Videocodierung für UMTS. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, July 2001.

[Red03] A.A. Redweik. Interactive WWW-based presentation and assessment of wireless video transmission. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, May 2003.

[Rei03] F. Reif. IP-basierte AMR-Sprachübertragung für GSM-paketkanäle. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 2003.

[Sch00] T. Schmid. Sequentielle Decodieralgorithmen bei Multilevel-Codierung. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, January 2000.

[Sch04] P. Scharl. Fountain-Codes für Multimedia-Broadcast im Mobilfunk. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, October 2004.

[Sol00] G. Soldannski. Joint iterative phase estimation and decoding. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, March 2000.

[Str01] M. Strobl. Video-Streaming für Kanäle mit variabler Bitrate. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, June 2001.

[Str02] P. Strasser. Efficient multiplexing of IP-streams at the GSM-GPRS air-interface. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, March 2002.

[Tse03] A. Tsetsos. Generischer Vorwärtsfehlerschutz für IP-basierte Multimediaübertragung. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, July 2003.

[Vla97] R. Vlaicu. Fehlerrobuste MPEG-4 Videoübertragung über DAB. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 1997.

[Wal04] M. Walter. Advanced bitstream switching for wireless video streaming. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, December 2004.

[Xu02] T. Xu. Fast channel-optimized mode selection for H.26L video transmission. Master's thesis, Munich University of Technology (TUM), Munich, Germany, August 2002.

[You06] A. Younus. Dynamic and interactive multimedia scenes in mobile broadcast environments. Master's thesis, Munich University of Technology (TUM), Munich, Germany, September 2006.

[Yu01] D. Yu. Channel coding toolbox for multimedia transmission. Master's thesis, Munich University of Technology (TUM), Munich, Germany, October 2001.

[Zha03] Q. Zhao. Media-streaming strategies over wireless channels. Master's thesis, Munich University of Technology (TUM), Munich, Germany, October 2003.

[Zha04a] P. Zhang. Error concealment for mobile video communication. Diploma thesis, Munich University of Technology (TUM), Munich, Germany, August 2004.

[Zha04b] Q. Zhang. Application layer coding for mobile multimedia broadcast. Master's thesis, Munich University of Technology (TUM), Munich, Germany, August 2004.

[Zhu05a] Y. Zhu. Channel coding options for multicarrier GSM. Master's thesis, Munich University of Technology (TUM), Munich, Germany, September 2005.

[Zhu05b] Y. Zhu. Scalable video coding for multimedia broadcast services. Master's thesis, Munich University of Technology (TUM), Munich, Germany, September 2005.