ALGORITHMS FOR SCALABLE AND NETWORK-ADAPTIVE VIDEO CODING AND TRANSMISSION

by

Huisheng Wang

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

December 2007

Dedication

To my family and the unforgettable memory of my beloved father.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Antonio Ortega, for his guidance, inspiration, support and patience throughout the years I have been pursuing my Ph.D. degree at the University of Southern California. Thanks to Prof. Zhen Zhang and Prof. Cyrus Shahabi for serving on my Dissertation committee, and Prof. C.-C. Jay Kuo and Prof. Shrikanth S. Narayanan for serving on my qualifying exam committee. I would also like to thank Prof. Christos Kyriakakis for his support and guidance during my first year at USC. The internship experience at both La Jolla Lab, STMicroelectronics Inc. and HP Labs has been enjoyable and productive. I am grateful to my mentors Dr. George Chen and Dr. Debargha Mukherjee who gave me the opportunity to work with great minds at those labs. I would also like to thank all my group members and friends for their friendship and assistance. Last, but not least, I thank my family for their consistent support through these years. I thank my mother for her unconditional love and support; I thank my daughter for bringing me a completely new life as a mother with so many joyful moments; I especially thank my husband, Yinqing Zhao, for his love, encouragement and many times of technical discussions on my research projects.

This research has been funded in part by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, Cooperative Agreement No. EEC-9529152.

Contents

List Of Tables

List Of Figures

Abstract

Real-time multimedia services over the Internet face some fundamental challenges due to time constraints of those applications and network variations in bandwidth, delay and packet loss rate. Our research addresses the problem of network-adaptive video coding and streaming based on source codecs that provide scalability to match the network environments.

The first part of the thesis focuses on scheduling algorithm design for network-adaptive video streaming. We extend previous work on rate-distortion optimized video streaming to address more general coding techniques that support multiple decoding paths to enhance adaptation flexibility. Prior work had only considered a single decoding path. Examples of multiple decoding paths include cases where there are multiple redundant representations of the media data or where error concealment is used. An example of these codes is our proposed multiple description layered coding (MDLC), which combines the advantages of layered coding and multiple description coding. This work is composed of several main components: (1) a new source model called directed acyclic hyperGraph (DAHG) to estimate the expected end-to-end distortion; (2) rate-distortion based scheduling algorithms to adjust dynamically the system's real-time redundancy to match the channel behavior; and (3) performance analysis on both source redundancy and

transport redundancy. Experimental results show that the proposed streaming framework can provide a very robust and efficient video communication for real-time applications over lossy packet networks.

The second part proposes a framework for adaptive scalable video coding using Wyner-Ziv techniques. The current scalable video coding standards suffer to some degree from a combination of lower coding performance and higher coding complexity, as compared to non-scalable coding. A key issue is how to exploit temporal correlation efficiently in scalable coding. We propose a novel scalable coding approach by introducing distributed source coding in enhancement layer prediction in order to achieve a better coding performance with reasonable encoding complexity. Experimental results show significant improvements in coding efficiency over MPEG-4 FGS, especially for video sequences with high temporal correlation.

Chapter 1

Introduction

1.1 Motivation

Recent technological developments in computing, compression, storage devices, and high-speed networks have made it feasible to provide real-time multimedia services over the Internet. Different from data communications, which are usually not under strict delay constraints, real-time multimedia communication is delay sensitive, i.e., data become useless when arriving late. Real-time delivery of live or pre-encoded (stored) video plays an important role in real-time multimedia. For distribution of live video, such as video conferencing or the live broadcast of an event, encoding and decoding must be accomplished in real-time. In many other applications, such as video on demand, video content is pre-encoded and stored for later viewing. Pre-encoded video has the advantage that it does not have the real-time encoding constraint. Thus, it allows more complicated and efficient encoding techniques. For example, multiple redundant representations can be created in advance for the same video content.

The most popular and widely deployed media communication applications are likely to be those that are accessible over the Internet, through both currently predominant wired and emerging wireless channels. Video transmission on these networks is characterized by variations in channel bandwidth, delay and packet loss rate, which can severely affect the reproduction quality of the video delivered through the network. In addition, the increasing heterogeneity of networks and network access devices makes video streaming more difficult. Thus, real-time multimedia communication over the Internet and wireless networks has posed a number of challenges [23, 27, 76, 88].

To address these challenges, we propose network-adaptive video coding and transmission solutions. Specifically, we aim at providing efficient, robust, scalable and delay-constrained media coding and streaming. The traditional rate control techniques [54, 75] aim to optimize the media quality for a fixed bit rate. This poses a problem when multiple users are trying to access the same media source through different network links and with different computing powers. Even in the case of a single user accessing one media source over a link, when this link suffers from varying channel conditions, relying on a complex rate-control algorithm to make rate adjustments in real time may not be practical (for example, if the changes in rate have to occur in a very short time frame). Thus, it is highly desirable to provide scalability through different video coding methods and transport mechanisms. Scalability refers to the ability of recovering physically meaningful image or video information by decoding only part of the compressed bitstreams. It is very useful in two aspects: (1) providing error resilience to combat potential transmission errors, and (2) enabling dynamic content adaptation to different network and terminal characteristics and user requirements.

Layered coding (LC) [44] and multiple description coding (MDC) [29, 89] have been proposed as two different kinds of "quality adaptation" schemes for video delivery over the current Internet or wireless networks. LC addresses the problem of network bandwidth variation through a sequence of dependent layers, while MDC provides error resilience by introducing redundancy explicitly into its independent descriptions. Most research comparing LC and MDC [13, 43, 53, 63, 65, 72, 101] leads to the conclusion that LC and MDC can each be preferable in different scenarios (e.g., low packet loss rate and long end-to-end delay for LC vs. high loss rate and short delay for MDC), though a few [13, 53] also show that LC with good rate allocation and scheduling techniques may outperform MDC over a broad range of scenarios. This motivates us to create a new multiple description layered coding (MDLC) approach by combining the advantages of LC and MDC to provide a graceful adaptation over a wider range of application and network scenarios (see Chapter 2). The MDLC codec produces multiple redundant representations increasing the flexibility with which a video server can adapt to varying network conditions, without re-encoding the video stream or completely switching between different encoding modes on the fly. But in order to fully exploit the adaptation flexibility of a MDLC codec with redundancy, the system requires an intelligent transport mechanism.

Scalable coding techniques make it easier for media servers to adapt to varying network conditions in real time. To do this, an intelligent transport mechanism is required to select the right packets (layers or descriptions) to send at a given transmission time to maximize the playback quality at the decoder. Some recent work has been focused on rate-distortion optimized scheduling algorithms for scalable video streaming [11, 19–21, 50, 51]. In this case, each packet is not equally important due to different distortion

contributions, playback deadlines, and packet dependencies caused by temporal prediction and layering. Runtime feedback information is exploited to make transport decisions based on current network condition and transmitted packet status (i.e., received or not). However, those works are mainly focused on encoding techniques, such as layered coding, which generate packets that can only be decoded following a *single decoding path* (SDP): a packet can be decoded with distortion reduction d only when all of its dependent data units are received and decodable; otherwise, it contributes 0. In fact, a source codec that supports *multiple decoding paths* (MDP) can greatly enhance adaptation flexibility of a streaming system. Multiple decoding paths can arise when multiple redundant representations of the same video content are created or when error concealment techniques are used. Examples of these codecs include MDLC and multiple independently encoded video streams. Compared to a SDP codec, a MDP codec introduces two additional features that the existing scheduling techniques fail to address: (1) redundancy across data units, and (2) more than one decoding choice is available for a given data unit, each with a different distortion reduction, when different subsets of its dependent data units are received. In our research, we extend the streaming framework in [19, 20] to address a more general problem where multiple decoding paths exist, by taking into account both dependency and redundancy relations among data units. The proposed rate-distortion based scheduling algorithms in Chapter 3 can dynamically adjust the system's real-time redundancy to match the channel behavior, thus achieving better overall expected end-to-end performance.

Unfortunately, all current scalable video coding standards suffer to some degree from a combination of lower coding performance and higher coding complexity, as compared to

non-scalable coding. A key issue is how to exploit temporal correlation efficiently in scalable coding. It is well known that motion prediction increases the difficulty of achieving efficient scalable coding because scalability leads to multiple possible reconstructions of each frame [66]. In this situation either (i) the same predictor is used for all layers, which leads to either drift or coding inefficiency, or (ii) a different predictor is obtained for each reconstructed version and used for the corresponding layer of the current frame, which leads to added complexity. MPEG-2 SNR scalability with a single motion-compensated prediction (MCP) loop and MPEG-4 FGS exemplify the first approach. Some advanced approaches with multiple MCPs are described in [7, 33, 66, 79, 93]. Distributed source coding (DSC) techniques based on network information theory provide a different and interesting viewpoint to tackle these problems. DSC first arose in the context of information theoretical problems, with compression bounds established in the 1970s by Slepian and Wolf [73] for distributed lossless coding and by Wyner and Ziv [94] for lossy coding with decoder side information (SI). But initial results did not address practical design. Recently, DSC has become an area of increasing research interest in the signal processing community for its potential applications [3, 58, 60, 69] in both video coding (e.g., error robustness to channel losses or reduced encoding complexity) and multiterminal communication systems (e.g., sensor networks). In this research, we are particularly interested in the problem of source coding with SI that is only known by decoder. In closed-loop prediction (CLP), in order to prevent drift at the decoder, the encoder needs to generate the same predictor that will be available at the decoder. Instead, a DSC encoder only needs to have access to the *correlation structure* between the current signal and the predictor. Thus there is no need to reproduce the decoded signal at the encoder as long as

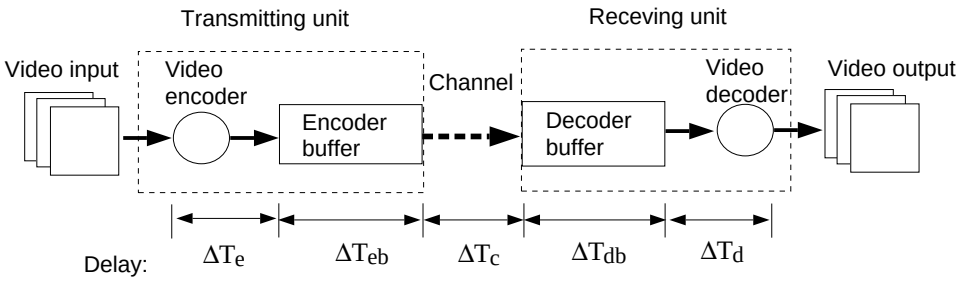


Figure 1.1: Delay components of a communication system. Adapted from [54].

the correlation structure is known, or can be found. Based on DSC principles, we propose a Wyner-Ziv scalable (WZS) coder in Chapter 4 that can achieve higher coding efficiency (up to 3 - 4.5 dB gain over MPEG-4 FGS for video sequences with high temporal correlation), by selectively exploiting the high quality reconstruction of the previous frame in the enhancement layer coding of the current frame. This creates a multi-layer Wyner-Ziv prediction "link", connecting the same bitplane level between successive frames, thus providing improved temporal prediction as compared to MPEG-4 FGS, while keeping complexity reasonable at the encoder.

In the rest of this chapter, we will first provide an overview of a video communication system, and then give brief reviews on the related areas of scalable coding, distributed source coding, and rate-distortion optimized packet scheduling. The contributions of the thesis are summarized at the end of the chapter.

1.2 Overview of a Video Communication System

1.2.1 Delay-Constrained Video Transmission

Figure 1.1 provides an abstraction of a real-time video communication system in terms of delay components. In contrast to data communication or to simple media downloading,

real-time video streaming is often subject to strict delay constraints. The main difference with respect to downloading applications is that media playback starts as data is still being received, so that playback could be interrupted if the decoder ran out of data to decode. As data starts to reach the client, decoding does not start immediately. Instead the client waits for a predetermined startup delay in order to accommodate enough data for decoding. Both encoder and decoder buffers can be used to smooth out the bit rate variation produced by the encoder as well as the channel delay variation, so that the decoded video can be played out at a constant frame rate. Note that when considering pre-encoded media, the encoding delay does not exist, as the video has already been encoded and is ready for transmission. In general, video streaming applications impose some restriction on the initial startup delay. Furthermore, the end-to-end delay requirement imposes a constraint on the encoding rate for each frame, or the number of layers to be transmitted for a scalable bit stream.

1.2.2 Bandwidth Variation and Transmission Impairments

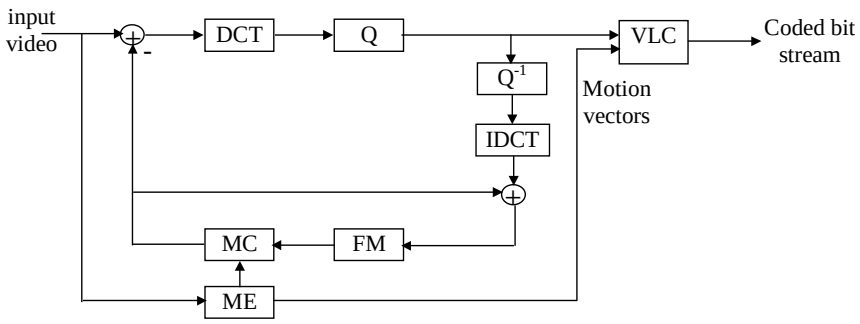
Most currently deployed networks provide no quality of service (QoS) guarantees. Thus it is expected that both network conditions and the available bandwidth of the underlying network may change during a real-time video communication session. Different types of transmission errors may occur, e.g., packet erasure errors or bit errors in IP or wireless based networks. Random bit errors may ultimately lead to erasures in a variable-length coded bit stream, since a single bit error can cause the remaining bits in a data packet to be undecodable. For real-time video, data packets are also treated as lost if they arrive

at the decoder after the playback deadline. Here we assume that when data losses occur, packets are lost.

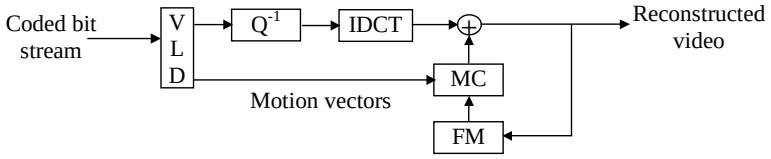
It is well known that compressed video streams are vulnerable to transmission impairments due to the variable-length coding and motion-compensated temporal prediction widely used in current video standards. Thus it is necessary to provide efficient mechanisms to address bandwidth fluctuation and packet losses in real-time video applications in order to provide a graceful quality degradation.

1.2.3 Motion-Compensated Temporal Prediction

Predictive coding is an important technique in image and video coding. The purpose of prediction is to exploit the redundancy between the samples to be coded. Temporal predictive coding using motion-compensated prediction has been widely employed in existing video coding standards, such as the MPEG and ITU-T H.26x families. The block-based hybrid video coding, the core of all the international video coding standards, effectively combines motion-compensated temporal prediction (MCP) and transform coding [88]. Each video frame is divided into a number of blocks with either fixed or variable block sizes. The encoding and decoding process for a block in the typical block-based hybrid video coding system is depicted in Figure 1.2. A block is first predicted from a few previously reconstructed reference frames using block-based motion estimation. The motion vector (MV) indicates the displacement between the current block and the selected reference block. The predicted block is then formed from the reference frame by using motion compensation with the estimated MV. The prediction error block is coded using the DCT transform, quantization and finally variable-length entropy coding (VLC). It



(a) Encoder



(b) Decoder

Figure 1.2: Block diagram for encoder and decoder in a typical block-based hybrid video coding system. ME: motion estimation, MC: motion compensation, FM: frame memory, DCT: discrete cosine transform, IDCT: inverse DCT, VLC: variable-length entropy coding, VLD: variable-length entropy decoding, Q: quantization.

is also noted from Figure 1.2 that the same reconstruction process is required at both the encoder and decoder. The introduction of CLP-based MCP also makes the coded bit stream very sensitive to transmission errors, which may cause error propagation over time. Distributed source coding techniques to be discussed in Section 1.4 lead to an alternative to closed-loop prediction by providing an open-loop prediction framework.

1.3 Scalable Coding

A scalable compressed bitstream typically contains multiple embedded subsets, each of which represents the original video content in a particular amplitude resolution (so called SNR scalability), spatial resolution (spatial scalability), temporal resolution (temporal

scalability) or frequency resolution (frequency scalability, also known in some cases as data partitioning). Scalable coders can have either coarse granularity or fine granularity, and they usually lead to a decrease in compression performance as compared to non-scalable coding. Thus, the design goal in scalable coding is to minimize the reduction in coding efficiency while enabling sufficient scalability to match the network requirements. In this section, we will briefly discuss several SNR scalability techniques in current video coding standards, particularly MPEG-2 SNR scalability and MPEG-4 Fine Granularity Scalability (FGS). A brief review of multiple description coding is also included at the end of this chapter. More detailed descriptions of these techniques can be found in [34, 35, 44, 76, 88].

1.3.1 MPEG-2 SNR Scalability

International video coding standards typically standardize decoders rather than encoders. Figure 1.3 shows the two-layer SNR scalable decoder defined in the MPEG-2 video standard [34]. It can be extended to the multi-layer coding scenario in a straightforward way. The reconstructed DCT coefficients from all the layers are added together before passing through the single inverse DCT (IDCT) and the MCP loop to produce the decoded frame. The enhancement-layer information of previous reference frames is used in the MCP loop for constructing both base and enhancement layers of the current frame. Several encoder configurations have been proposed in the literature [7, 92]. One approach [92] is to apply a single MCP loop at the encoder that also uses the enhancement-layer information. This leads to drift when the enhancement layer is not received by the decoder. A more complicated pyramid encoder [7] uses multiple MCP loops, with one MCP loop per layer

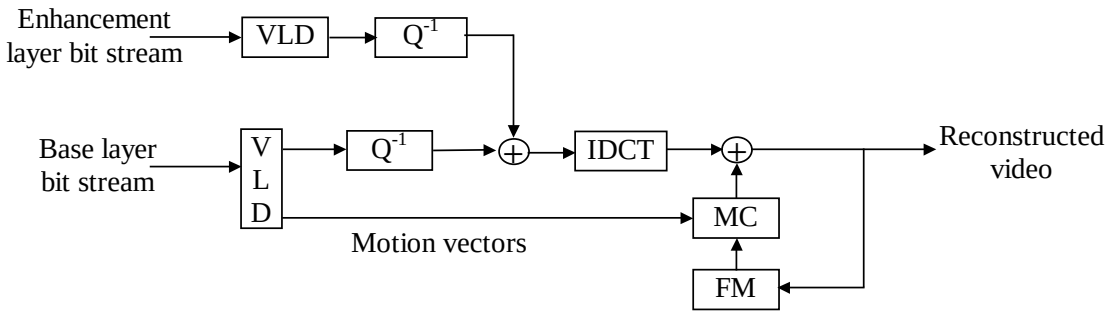


Figure 1.3: MPEG-2 SNR decoder.

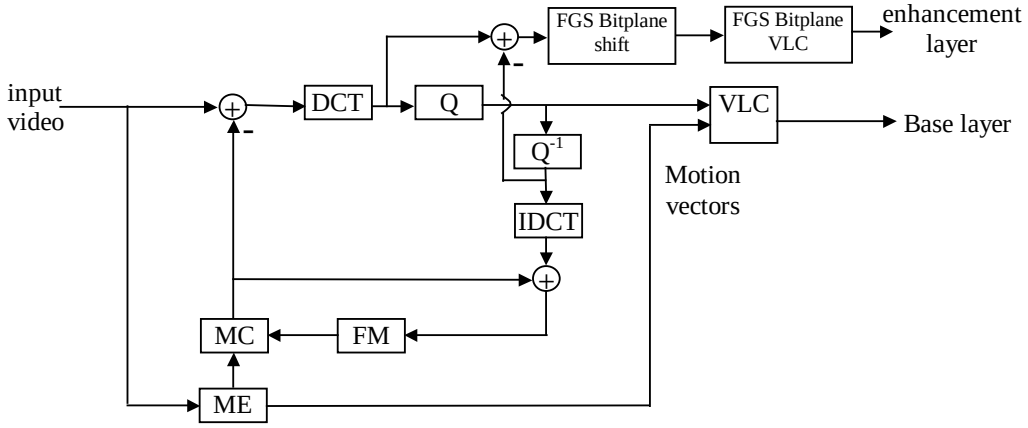


Figure 1.4: Simplified diagram of FGS encoder structure used in MPEG-4 Microsoft reference software.

to control drift in case the higher layers are lost. The frame memory at each layer of the encoder corresponds to the state of the decoder frame memory assuming all its higher layers are not decoded.

1.3.2 Fine Granularity Scalability (FGS) and its Variants

MPEG-2 SNR scalability usually supports only a small number of enhancement layers due to the coding efficiency degradation and extra encoder complexity introduced for each enhancement layer in the pyramid encoder. Thus, the quality can only improve as rate increases in larger discrete steps. A common characteristic for those coarse granularity

scalable coding techniques is that the enhancement layer can contribute to end-to-end distortion reduction only if it is entirely decoded. In other words, it does not provide any partial enhancement. MPEG-4 fine granularity scalability (FGS) [44] provides a way to improve the quality with much smaller incremental steps. Figure 1.4 shows a simplified diagram of the FGS encoder structure used in the MPEG-4 Microsoft reference software. There are three major differences between FGS and MPEG-2 SNR scalability. First, the enhancement-layer bit stream of FGS can be truncated at arbitrary bit positions within each frame to provide partial enhancement proportional to the number of bits decoded. Second, FGS does not use the enhancement information of the previous frames to predict the current frame in motion-compensation loop. Instead, the enhancement layer is represented by coding the residual error based on current base-layer reconstruction. Compared to MPEG-2 SNR scalability, this approach avoids drift but results in lower coding efficiency. Finally, the FGS enhancement layer is coded using bit-plane coding, i.e., the quantization step sizes are in descending powers of two.

FGS provides fine granularity bit-rate scalability, channel adaptation, and elegant error recovery from occasional data losses or errors in enhancement layers [44]. However, it suffers from the disadvantage of low coding efficiency. A number of FGS variants have been proposed to exploit further the temporal correlation in the enhancement layer. Examples include progressive FGS (PFGS) [93], motion-compensation based FGS (MC-FGS) [79], and leaky prediction based FGS (LP-FGS) [33]. These techniques share a common feature in that they employ one or more additional MCP loops for enhancement layers of P and B frames (or B frames only), for which a certain number of FGS bitplanes, M, are included in the enhancement-layer MCP loop, to improve the coding efficiency. In

this case prediction drift will occur within the FGS layers when fewer than M bitplanes are received. M is chosen by considering the trade-off between the coding efficiency and prediction drift. LP-FGS also introduces a second parameter α to adjust the amount of predictive leak to control the construction of the reference frame at the enhancement layer.

Rose and Regunathan [66] also proposed an estimation-theoretic (ET) approach with multiple motion-compensation loops for general SNR scalability. In this approach, the enhancement-layer predictor is optimally estimated by considering all the available information from both base and enhancement layers. It can be easily extended into the FGS framework. However, the underlying CLP prediction requires the encoder to generate all possible decoded versions for each frame, so that each of them can be used to generate a prediction residue. Thus complexity is high at the encoder, especially for multi-layer coding scenarios.

1.3.3 Multiple Description Coding

Layered coding (LC) as both MPEG-2 SNR scalability and MPEG-4 FGS can enhance adaptation of a video delivery system. But it requires strong protection for base layer via either FEC or ARQ schemes. Multiple description coding (MDC) has emerged as another promising approach to combat transmission errors. A multiple description (MD) coder generates several bit streams for the original video source (referred to as descriptions), so that each description alone provides acceptable quality and incremental improvement can be achieved with additional descriptions received. Each description is individually packetized, and transmitted through separate channels or through one physical channel

that is divided into several virtual channels by using appropriate time interleaving techniques. Each description can be decoded independently to provide an acceptable level of quality. For this to be true, all the descriptions must have some basic information about the source, and therefore redundancy is introduced between different descriptions.

A number of MD coders have been proposed, including overlapping quantization [71, 78], correlating linear transforms [87], polyphase transform and subband coding [39], and interleaved spatial-temporal sampling [90]. A comprehensive review of various MD algorithms, particularly for image communications is presented in Goyal's paper [29]. Motion-compensation prediction is a fundamental component in current video coding standards. Since MD coders are designed to "tolerate" channel losses, MD video coders that incorporate motion-compensated prediction must take into account two basic problems: mismatch control and redundancy allocation. Depending on the trade-off between these two problems, Wang *et al.* have categorized the possible predictors for a MD coder into three classes [89]: (1) predictors that introduce no mismatch using either two individual predictors or a single predictor based on information common to both descriptions, (2) a single predictor identical to that used by a single-prediction predictive encoder that minimizes the prediction error while causing mismatch unless both descriptions have been received, and (3) predictors that have parameters to control the trade-off between prediction efficiency and the amount of mismatch. When combined with multiple path transport (MPT) [5, 6, 48, 55], MDC can exploit path diversity to improve error resilience, traffic dispersion and load balancing in the network.

The major difference between LC and MDC is that LC proposes a hierarchical decomposition while MDC decomposes the source signal into non-hierarchical, correlated

descriptions. Thus, MDC does not require special treatment of packets by retransmissions or using FEC to guarantee adequate quality, as LC does for the base layer. It makes MDC particularly useful for those networks that do not support feedback or retransmission, or those applications that only allow very short delay, thus making retransmission unacceptable. However, MDC introduces significant redundancy between the descriptions, which reduces the coding efficiency. This observation motivates us to design an adaptive approach to combine the advantages of MDC and LC so as to provide robustness over a wide range of network scenarios and application requirements. Our proposed approach, multiple description layered coding, will be discussed in Chapter 2.

1.4 Distributed Source Coding

This problem has been studied in the information theory literature back in the 1970s, starting with the work of Slepian and Wolf for distributed lossless coding and that of Wyner and Ziv on "rate-distortion with side information" for the lossy coding. However, the emergence of practical applications has only occurred recently. In this section, we first review the fundamental principles of Slepian-Wolf and Wyner-Ziv coding, and then discuss some practical applications focusing on distributed video compression. A detailed review on this topic was also presented in Girod *et al.*'s recent paper [26]. The theoretical problem of successive refinement of information in the Wyner-Ziv setting will be described in Chapter 4 where we design a Wyner-Ziv scalable codec based on this setting.

Figure 1.5 shows the problem of source coding with side information. If the side information (SI) Y were known at both encoder and decoder, then the problem of compressing

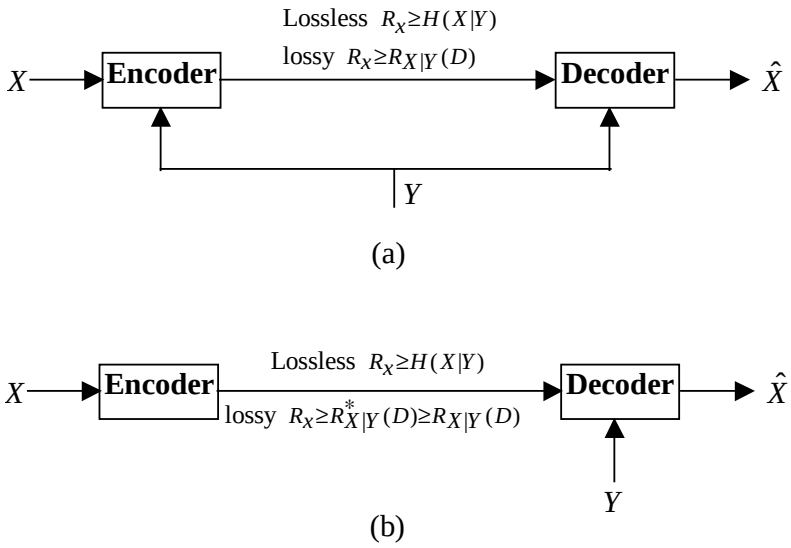


Figure 1.5: source coding with side information, adapted from Figure 1 in [58]. (a) SI Y is available at both the encoder and decoder; (b) SI Y is available at the decoder only.

the source X is well known: the theoretical rate of X is given by $H(X|Y)$ for lossless compression and $R_{X|Y}(D)$ for lossy compression, where $H(X|Y)$ is the conditional entropy of X given Y, and $R_{X|Y}(D)$ is the rate-distortion function if Y is available at both the encoder and decoder. A practical approach in this case is to use differential pulse-code modulation (DPCM). We are now interested in the case where Y is available at the decoder, but not at the encoder. Consider first the problem where X and Y are correlated discrete-alphabet memoryless sources and X is to be compressed losslessly. The Slepian-Wolf Theorem [73] establishes the achievable rate region $R_x \geq H(X|Y)$, which is the same as if Y is also available at the encoder. Later Wyner and Ziv [94] extended this work to the lossy compression cases, for which X and Y can be continuous with infinite alphabets. The Wyner-Ziv rate-distortion function $R^*_{X|Y}(D)$ is then defined as the lower bound of the achievable rate for a given distortion $D = E[d(X|\hat{X})]$. Wyner and Ziv

proved that $R^*_{X|Y}(D) \geq R_{X|Y}(D)$, a possible rate penalty existing when the encoder cannot access to Y. Furthermore, $R^*_{X|Y}(D) = R_{X|Y}(D)$ in the case of Gaussian memoryless sources with quadratic distortion measure. By using a duality argument, Pradhan and Ramchandran recently generalized $R^*_{X|Y}(D) = R_{X|Y}(D)$ to the more general case where source X is the sum of arbitrarily distributed SI Y and independent Gaussian noise [57].

Distributed source coding (DSC) is related to channel coding in that Y can be regarded as a noisy version of X produced by a virtual channel. Instead of performing FEC to protect against the transmission errors introduced by the real channel, we send the parity or syndrome bits to the decoder to correct the "errors" introduced by the virtual dependence channel. The decoder can then perform channel decoding with both information from parity bits and side information Y. The first constructive framework for creating practical codes for DSC was distributed source coding using syndromes (DISCUS), proposed by Pradhan and Ramchandran [58], where a scalar and trellis-based coset construction was presented. Since then, more sophisticated channel codes based on turbo or low-density parity check (LDPC) codes have been applied to DSC by a number of researchers [1, 8, 25, 42, 46]. For Wyner-Ziv coding, Zamir *et al.* [97, 98] proposed a constructive framework based on nested linear/lattice codes. This was further studied by Servetto [70] who considered the design of lattice quantizers and presented a performance analysis at high rates. Xiong *et al.* [95] developed a Wyner-Ziv coding framework consisting of a nested quantizer followed by a Slepian-Wolf coder.

Recently proposed Wyner-Ziv coding techniques for video applications fall into several categories. First, the emergence of mobile devices, such as wireless video sensors or mobile cameraphones, inspires a new class of video compression applications that require

low-complexity encoding. Motion estimation is the most time-consuming component in traditional MCP-based encoders. The main difference between Wyner-Ziv coding and CLP-based approaches (for example, MCP) is that Wyner-Ziv coding only needs to know the correlation structure between the current signal and the predictor, rather than the exact predictor value. Therefore, it is possible in Wyner-Ziv video codecs to move the motion estimation unit to the decoder in order to reduce the encoder complexity. Puri and Ramchandran [59–61] and Girod *et al.* [2, 3, 26] have proposed several different Wyner-Ziv video coding schemes based on this principle. The second class of applications is to use Wyner-Ziv coding for error resilience. Sehgal *et al.* [67, 69] proposed a Wyner-Ziv coding scheme to prevent error propagation in predictively encoded video. In addition, Wyner-Ziv coding can also be applied in multiple description coding [38] or layered coding [68, 96]. In [96], Xu and Xiong proposed an MPEG-4 FGS-like scheme by treating a standard coded video as a base layer, and building the bit-plane enhancement layers using Wyner-Ziv coding with current base and lower layers as SI. Our Wyner-Ziv scalable approach, to be presented in Chapter 3, can achieve higher coding efficiency by selectively exploiting the high quality reconstruction of the previous frame in the enhancement layer bitplane coding of the current frame.

1.5 Rate-distortion Optimized Packet Scheduling

Achieving real-time video delivery can be very challenging given the limited channel bandwidth as well as packet loss rate and transmission delay variations. It is well recognized that an ideal transport mechanism should adapt to the actual channel conditions, such

as channel bandwidth, delay and packet loss statistics. Chou and Miao [19, 20] provided a rate-distortion optimized framework of packet scheduling over a lossy packet network. They considered streaming as a stochastic control problem, with the goal of determining which packets to be sent (and when and how), in order to maximize the expected end-to-end quality under an average rate constraint. A directed acyclic graph (DAG) model was proposed to capture the dependencies between packets such that it is possible to attach more importance to packets on which multiple other packets depend. The details of this algorithm will be reviewed in Section 3.2. To reduce the scheduling complexity, Miao and Ortega [50, 51] proposed an expected run-time distortion based scheduling (ERDBS) algorithm which simply uses a greedy solution by explicitly considering the effects of data dependencies and delay constraints into a single importance metric. The rate-distortion framework in [19, 20] can be applied in a series of scenarios, including packet scheduling at the sender [19, 20], at the receiver [21], or at the intermediate proxy [11].

End-to-end distortion estimation is a challenging problem in rate-distortion based scheduling algorithms. When the dependency among packets is modelled by a DAG, Chou and Miao [19, 20] computed the distortion contribution of a packet based on the assumption of a single decoding path: a packet can be decoded only when all of its parents are received and decodable. One example of this is layered coding. Cheung and Tan introduced a more general formulation based on the DAG model [15] to include the case where a packet can assume different distortion contributions when different subsets of its dependent packets are available. They considered all possibilities of decoding and delivery scenarios, which leads to significant increases in complexity. In our approach, we propose a new directed acyclic hypergraph (DAHG) on the top of a DAG, by introducing additional

objects to explicitly represent the source redundancy among packets which arises from a given source coding approach (e.g., MDLC) in order to enhance the adaptation flexibility. Compared to [15], our DAHG model provides a more systematic way to represent source codecs that support multiple decoding paths with reasonable complexity. All of the above methods assign distortion on a per packet basis, and calculate the expected end-to-end distortion of a GOP based on the packet relationship represented in the source model. Zhang *et al.* proposed an alternative approach [100] by estimating the expected distortion of a GOP at runtime based on the stored distortion information of this GOP for several selected reference vectors of packet loss probability. Compared to above methods, this approach can only provide an approximation of the expected distortion via a first-order Taylor expansion. The approximation accuracy depends on the number of reference vectors and smoothness of the expected distortion over a range of packet loss probabilities.

1.6 Contributions of This Research

The main contributions of our research include the following:

1. *Network-adaptive video streaming [83, 85, 86].*

    - We develop a novel coding approach, namely, multiple description layered coding (MDLC), which combines hierarchical scalability of LC with the reliability introduced by MDC.
    - We extend the rate-distortion optimized streaming framework proposed in [19, 20] to operate on a general class of coding formats that explicitly support redundancy in their coding structures. Such codecs (e.g., MDLC) produce

multiple redundant representations, which facilitate server adaptation to varying network conditions, without re-encoding the video stream or completely switching between different encoding modes on the fly. We first introduce a new directed acyclic hypergraph (DAHG) to represent the data dependencies and correlation between different video packets, from which the expected end-to-end distortion for a group of packets can be estimated accurately. Based on the DAHG model, we then develop two rate-distortion based packet scheduling algorithms: one extended from the Lagrangian optimization proposed in [19, 20], and another one based on a greedy solution derived from the Taylor analysis of the expected distortion.

- We observe two types of redundancy, namely, source redundancy and transport redundancy, in the proposed streaming framework. We investigate the impacts of both redundancies on error control for a lossy packet network.

2. *Wyner-Ziv Scalable predictive coding [81, 82, 84].*

- We propose a practical video coding framework based on distributed source coding principles, with the goal to achieve efficient and low-complexity scalable coding. Starting from a standard predictive coder as base layer (such as MPEG-4 baseline video coder in our implementation), the proposed Wyner-Ziv scalable coder can achieve higher coding efficiency, by selectively exploiting the high quality reconstruction of the previous frame in the enhancement layer coding of the current frame.

- Correlation estimation at the encoder is a challenging problem in Wyner-Ziv coding. We propose two simple and efficient algorithms, namely, direct estimation and model-based estimation, to explicitly estimate at the encoder the parameters of a model to represent the correlation between the current frame and an optimized side information available only at the decoder. Our estimates closely match the actual correlation between the source and the decoder side information.
- Since the temporal correlation varies in time and space, we propose a block-based adaptive mode selection algorithm for each bit-plane, so that it is possible to switch between different coding modes.

The rest of the thesis is organized as follows. We discuss the MDLC codec in Chapter 2. Then we describe the rate-distortion based scheduling framework for a general class of sources with redundancy in Chapter 3. The Wyner-Ziv scalable coding is presented in Chapter 4. Finally, conclusions and future work are provided in Chapter 5.

Chapter 2

Multiple Description Layered Coding (MDLC)

2.1 Introduction

In this chapter we present an efficient multiple description layered coding (MDLC) system for robust video communication over unreliable channels. Recent technological developments and the rapid growth of Internet and wireless networks make it feasible and more attractive to provide real-time video services over them. However the current best-effort Internet does not offer any QoS guarantees. The congestion, routing delay and fluctuations of network conditions can all result in the packet loss or large delay during the transmission, and thus greatly degrade the received video quality.

A traditional method to provide bandwidth adaptation and error resilience in lossy transmission environments is layered coding (LC) [34, 35, 44], in which a video sequence is coded into a base layer and one or more enhancement layers. The base layer provides a minimum acceptable level of quality, and each additional enhancement layer incrementally improves the quality. Thus, graceful degradation in the face of bandwidth drops or transmission errors can be achieved by decoding only the base layer, while discarding

one or more of the enhancement layers. The enhancement layers are dependent on the base layer, and cannot be decoded if the base layer is not received. Thus LC requires the base layer to be highly protected, which can be achieved via either strong forward error correction (FEC) or automatic repeat request (ARQ) schemes. FEC has the drawback of requiring increased bandwidth, even in cases when errors do not occur, while ARQ may not be a practical alternative if the round-trip time (RTT) is long relative to the end-to-end delay in the application. Some recent work [11, 19–21, 50, 51] has proposed rate-distortion optimized scheduling algorithms for layered video streaming, by attaching different importance to each packet, and thus determining the optimal transmission policy of each packet based on their importance.

Another alternative to reliable communication is multiple description coding (MDC) [29, 39, 71, 78, 87, 89, 90]. With this coding scheme, a video sequence is coded into a number of separate bit streams (referred to as descriptions), so that each description alone provides acceptable quality and incremental improvement can be achieved with additional descriptions. Each description is individually packetized, and transmitted through separate channels or through one physical channel that is divided into several virtual channels by using appropriate time interleaving techniques. Each description can be decoded independently to provide an acceptable level of quality. For this to be true, all the descriptions must have some basic information about the source, and therefore they are likely to be correlated.

There have been some researches on performance comparison between LC and MDC. In [63], Reibman *et al.* first evaluated the performance of LC and MDC for transmission over binary symmetric channels and random erasure channels using FEC codes. Singh *et*

al. compared MDC without retransmission to LC with retransmission through network simulations [72]. In [65], Reibman *et al.* further studied the performance of LC and MDC for transmission over an EGPRS wireless network through either one or two correlated wireless channels. In [43], Lee *et al.* performed a comprehensive comparison between LC and MDC in multi-path environments under different error control scenarios including no error control, ARQ-based and FEC-based error control for both LC and MDC. Zhou *et al.* examined the performance of LC and MDC with or without retransmissions based on rate-distortion lower bounds combined with the effect of excess rate and delay incurred from retransmissions [101]. Since these performance comparisons depend on the actual coder implementations and underlying network environments, observations from these researches are not completely consistent. But most of these studies lead to a common belief that (1) MDC outperforms LC in network scenarios with high error rate, long RTT or stringent real time requirements [43, 63, 72, 101]; and (2) error control techniques such as ARQ or FEC are very useful for both LC and MDC [43, 63, 101]. Some recent work also studied the performance of LC and MDC when an advanced transport mechanism is used. In [53], Nguyen *et al.* showed that with good packet allocations LC outperforms MDC under various network conditions. In [13], Chakareski *et al.* described a large variation in relative performance between LC and MDC depending on the employed transmission scheme. Both works further demonstrate the importance of transport mechanisms optimized for a given source coding technique, complementing traditional transport techniques such as ARQ or FEC.

The above observation motivates us to look for an adaptive approach to combine LC and MDC in order to exploit their individual benefits, so as to provide reliable video

communication over a wider range of network scenarios and application requirements. The main novelty of our work is to demonstrate that it is possible to combine LC with MDC, by adding a standard-compatible enhancement to MPEG-4 version 2 [36]. The new multiple description layered coding (MDLC) approach presented here introduces redundancy in each layer so that the chance of receiving at least one base layer description is greatly enhanced. Furthermore, though LC and MDC each achieve good performance in different limit cases (e.g., long end-to-end delay for LC vs. short delay for MDC), the proposed MDLC system can address intermediate cases as well. As in a LC system with retransmission, the MDLC system can take advantage of a feedback channel that indicates which descriptions have been correctly received. Thus we will show that a low redundancy MDLC system can be implemented with runtime packet scheduling system proposed in Chapter 3 based on *a priori* channel knowledge and runtime feedback information. The goal of our scheduling algorithm is to find a proper on-line packet scheduling policy to maximize the playback quality at the decoder. This chapter is focused on the discussion of the proposed MDLC technique, and the scheduling algorithm for MDLC and its extension to a general class of coding problems with source redundancy will be described in detail in Chapter 3.

Closely related research on combining LC and MDC includes work by Chou *et al.* [22] and by Kondi [41]. In [22], the authors start from an MDC coder that is realized by applying unequal cross-packet FEC to different parts of a scalable bitstream [4], and then convert the descriptions into a base layer and an enhancement layer. In [41], the proposed codec relaxes the hierarchy of LC by producing a base layer and two multiple description

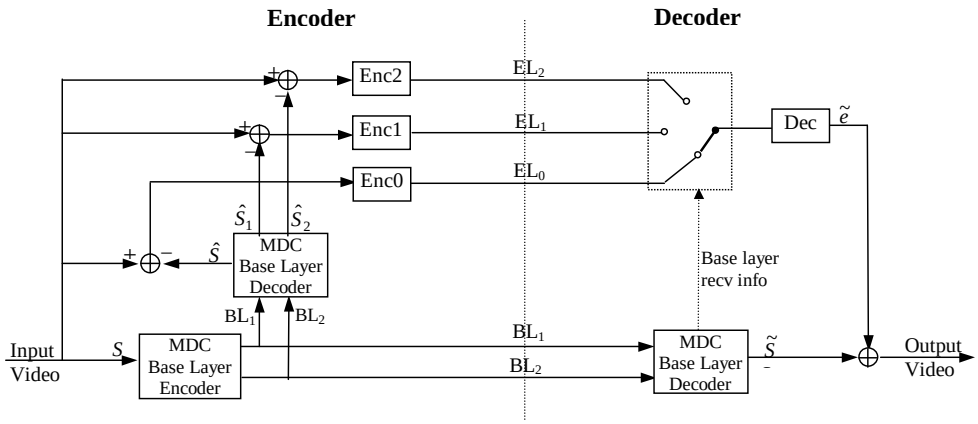


Figure 2.1: Structure of the proposed MDLC codec.

enhancement layers, where the base layer is required for decoding and the two enhancement layers can be decoded independently of each other. Both works optimize source redundancy at the encoding stage. In contrast, our proposed MDLC codec produces multiple redundant representations increasing the flexibility with which a video server can adapt to varying network conditions. An MDLC system can dynamically adjust the run-time redundancy of the compressed bit streams during transmission, for either a live or pre-encoded video, by applying a rate-distortion optimized scheduling algorithm.

The chapter is organized as follows. In Section 2.2 we describe the proposed MDLC approach. Simulation results are presented in Section 2.3. Finally we conclude our work in Section 2.4.

2.2 Proposed MDLC Codecs

2.2.1 General Structure

Our general approach to MDLC video coding uses an MDC encoder to generate two base layer descriptions BL_1 and BL_2, as shown in Figure 2.1. Then the base layer MDC

decoder in the MDLC encoder module mimics the three possible decoding scenarios at the receiver: both descriptions received or either one received. For the case when both descriptions are received, it reproduces the base layer as $\hat{S}$, and the difference between the original video input S and $\hat{S}$ is coded with a standard encoder such as MPEG-4 FGS into an enhancement layer stream EL_0. For the case when only one description is received, the base layer decoder generates a low quality reproduction $\hat{S_1}$ or $\hat{S_2}$, and feeds the difference into two enhancement layer encoders separately to create EL_1 and EL_2.

The key advantage of our MDLC scheme is that it combines the LC hierarchical scalability coding scheme with the reliability introduced by adding redundancy to the base layer by using multiple descriptions. With a well-designed scheduling algorithm, the sender can choose only one base layer description and its corresponding enhancement layer to be sent to the receiver, as in a standard LC system, in situations where the channel losses are low. Or it can send both base layer descriptions and their enhancement layer streams to get the maximum protection when channel conditions worsen. EL_0 is sent instead of either EL_1 or EL_2 to reduce the redundancy when both BL_1 and BL_2 are received, or expected to be received with high probability. The sender can select the packets to be transmitted at any given time during the transmission session based on the feedback information, in such a way as to maximize the playback quality at the decoder.

The proposed decoder system, depicted in Figure 2.1, is composed of two parts: base layer MDC decoder, and enhancement layer switch and decoder. The base layer MDC decoder will generate a reproduction $\tilde{S}$ which will be $\hat{S}$, $\hat{S_1}$ or $\hat{S_2}$ depending on what was received. The enhancement layer switch then selects which EL stream to decode given

what base layer was received. Finally, the decoded base layer and enhancement layer will be combined together to generate the final video output.

Given the general structure in Figure 2.1, a very important part of an MDLC codec design is determining how to construct base layer descriptions using a specific MDC algorithm. In the following discussions, we propose two MDLC codecs using different base layer MDC approaches, one based on DCT duplication and alternation, and the other based on MPEG-4 FGS temporal scalability (FGST) with temporal subsampling.

2.2.2 Codec 1: DCT Duplication and Alternation

We first propose a simple MDLC codec, where the base layer descriptions are formed by simple duplication and alternation of the DCT coefficients. The base layer is obtained by applying a coarse quantizer to the original video in DCT domain. We create our multiple base layer descriptions by repeating important information, such as the motion vectors in inter mode and DC coefficients in intra mode. For the rest of the DCT coefficients, we just alternate them into the two descriptions. For example, if a macroblock is assigned to BL_1, then its neighboring macroblocks are assigned to BL_2. Information from both descriptions is combined before making predictions for the future frames. We first used this codec in [83] to demonstrate the advantage of MDLC over either LC or MDC. While this design is simple, we expect that more complex schemes, such as combining with an optimal rate-distortion splitting [62, 64], would lead to improvements in overall performance.

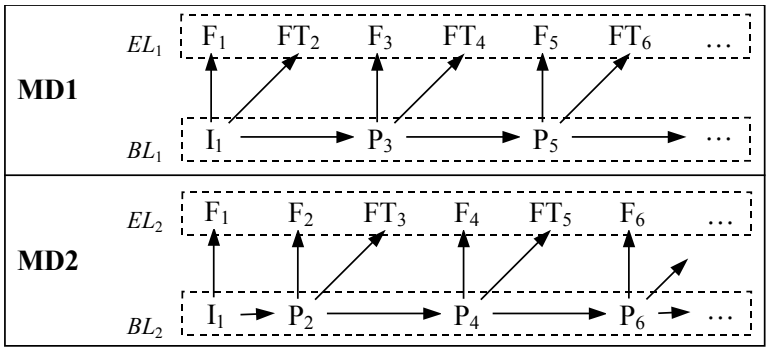


Figure 2.2: MDLC scheme based on MPEG-4 FGST. I: I-frame, P: P-frame, F: the enhancement layer generated by coding the residual between the original frame and its base layer reconstruction, FT: the enhancement layer generated by FGST using forward prediction from the base layer of its previous frame. The subscript of each label indicates the frame number. EL_0 in Figure 2.1 is not shown here as it is simply composed of F_i identical to either EL_1 or EL_2 based on the frame index.

2.2.3 Codec 2: based on MPEG-4 FGST with Temporal Subsampling

In this approach, we use video redundancy coding (VRC) [91] to create an MDC base layer, by partitioning a video sequence into two subsequences each of which mainly contains either odd or even frames. At the base layer, each subsequence is coded independently as an IPP sequence, where only the first frame (I-frame) of each group of pictures (GOP) is shared between both subsequences, as shown in Figure 2.2. Coding efficiency is reduced because the motion-compensated prediction using a past frame farther apart is usually less efficient than using the immediately past frame. If both descriptions are received correctly, each bit stream is decoded independently to produce the even and odd frames that are interleaved for the final base layer reconstruction. However, if only one description is received, the missed description can be estimated by simply copying the closest adjacent frame in the correctly received description or using more complicated motion interpolation techniques by exploiting both past and future frames [5].

In order to construct an MDLC codec, we introduce additional fine granularity bit-rate scalability by generating enhancement layers form the base layer descriptions. For each subsequence, as shown in Figure 2.2, we create enhancement layer descriptions with the MPEG-4 FGS temporal scalability (FGST) approach [44]. Each enhancement layer description codes the difference between the original picture and a reference picture reconstructed from its corresponding base layer description in bit-plane coding of the residual DCT coefficients. The residual DCT coefficients are obtained from different references depending on whether a frame is coded in the base layer description or not. Without loss of generality, consider the P-frame with an odd-index, i, in Figure 2.2. Its enhancement layer EL_1 represents the residue between frame i and its BL_1 reconstruction, denoted by F_i, while its EL_2 is generated using forward prediction from the BL_2 reconstruction of the previous frame $i-1$, denoted FT_i, which contains the enhancement-layer motion vectors as well. At the decoder, depending on what base layer description is received, the enhancement layer can choose to decode either all (e.g., when both base layer descriptions are received) or a subset of the descriptions (when only one base layer description is received). The final enhancement layer quality is the one with the best quality achieved by all decodable descriptions.

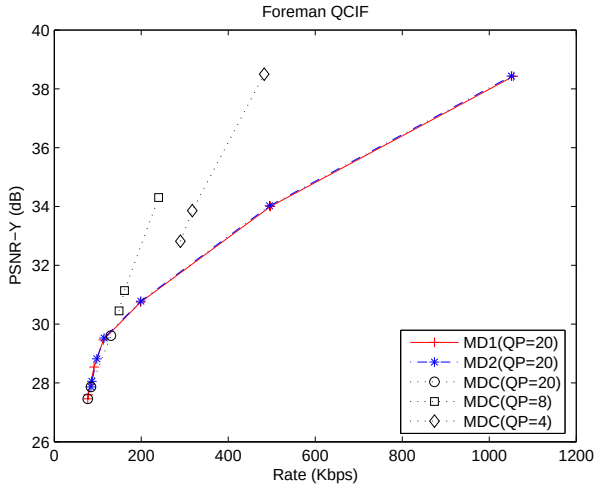
This MDLC approach is used in Chapter 3 as an example to demonstrate the efficiency of our scheduling algorithm for source codecs with redundancy. It is noted that this particular approach has a number of practical advantages in addition to the general features common to MDLC techniques. First, in addition to SNR quality, it provides temporal scalability that leads to a good reconstruction at half the original frame rate even when only one description is received. Second, it can be easily combined with

multiple path transport to improve error resilience. Third, it is straightforward to expand the current approach to more than two descriptions by splitting the frames evenly into multiple independent subsequences and coding each enhancement layer description using the same FGST approach. Last, it has the flexibility to provide unbalanced base layer descriptions by using different quantization steps for each description, which is useful to cover a wide range of bit rates for bandwidth adaptation.

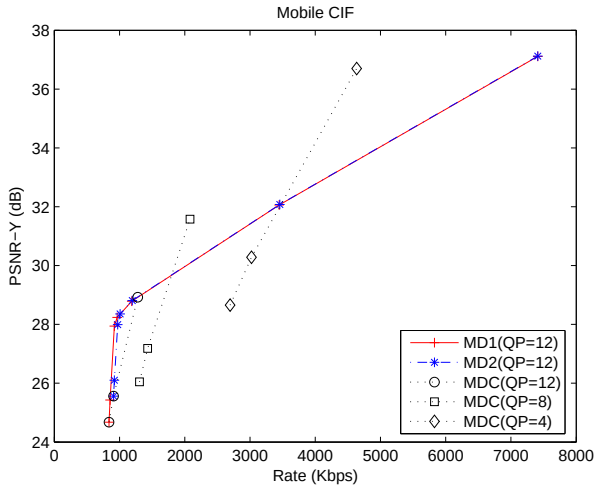
2.3 Experimental Results

We investigate the end-to-end distortion performance of the proposed MDLC technique, by comparing it with both LC and MDC. Here we use the MDLC approach based on MPEG-4 FGST to code video sequences. MPEG-4 FGS is used as the LC implementation, and the MDC system uses the same multiple description generating method as our MDLC for the base layer. The comparisons between MDLC and LC are discussed in detail in Section 3.5, which show that MDLC outperforms LC even when a rate-distortion (R-D) optimized packet scheduling algorithm is used. This is because having a source representation with built-in error resilience, through source redundancy, enables MDLC to provide an end-to-end performance that is less affected by packet losses, particularly in the case of poor channel conditions, such as high packet loss rate and short playback delay.

In this section, we compare the MDLC approach with a set of MDC schemes using different quantization parameters (QP). All of these schemes use an R-D optimized packet scheduling algorithm, based on Lagrangian optimization, described in Chapter 3. The



(a)

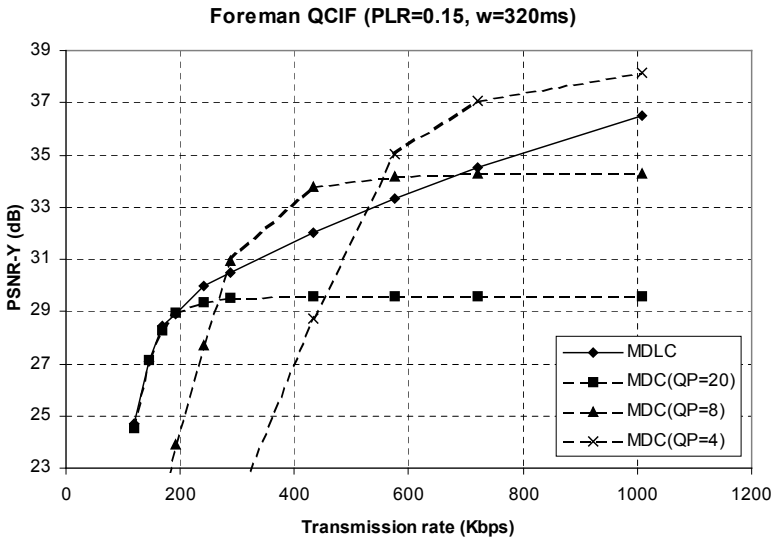


(b)

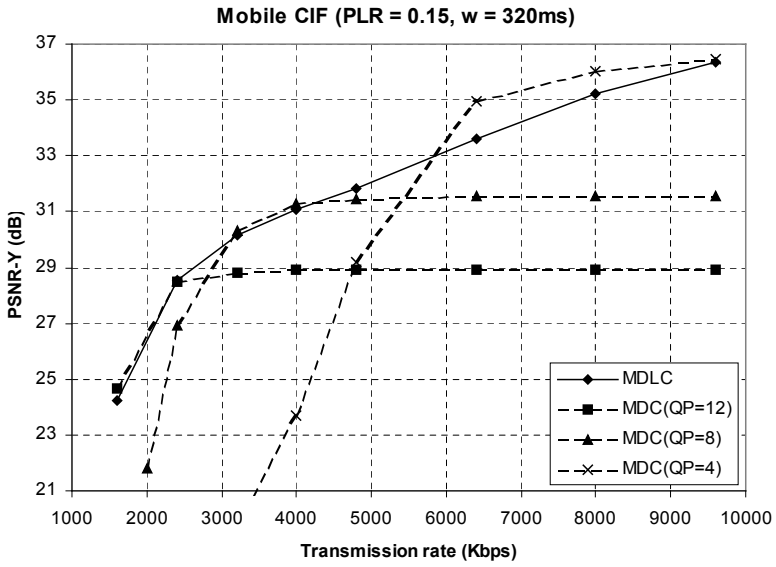
Figure 2.3: Rate-distortion curves of MDC codecs with different quantization parameters, $MD1$ and $MD2$ of MDLC for Foreman and Mobile measured at the encoder without transmission impacts.

video sequences used in the experiments are Foreman (QCIF) and Mobile (CIF). The encoding parameters of MDLC and streaming system setup are described in Section 3.5. In addition to the MDC codec that uses the same QP as that for the MDLC base layer, we employ two other MDC codecs with finer quantization steps to increase the achievable R-D range. Figure 2.3 shows the compression efficiencies of these MDC codecs, $MD1$ and $MD2$ of MDLC for Foreman and Mobile measured at the encoder. The three R-D operating points of each MDC codec are obtained by assuming that either one or both descriptions are received.

Figure 2.4 shows a performance comparison between these schemes when the packet loss rate (PLR) is 15% and the playback delay (denoted by w) is 320 ms. The MDLC approach provides a similar or better performance than its corresponding MDC approach using the same base layer QP over the bandwidth range tested in the simulation. However, the relative performance between MDLC and other MDC approaches (i.e., QP=8 or 4) depends on the available transmission rate. This can be explained as follows, by considering both compression efficiency and rate scalability of a source codec. Since MDLC provides a more graceful degradation of reconstruction quality through layered coding, the performance gain of MDLC over MDC increases at lower transmission rates, when a client may not receive any descriptions for some frames in the MDC case. At medium or high transmission rates, as more and more packets are received at the client, MDLC may perform worse than an MDC approach because of their respective compression efficiencies. For example, in Figure 2.3 (a), the MDC approach with QP=8 can achieve 34.3 dB of luminance PSNR at 240 Kbps in a perfect channel, while MDLC can only achieve less than 32 dB. Correspondingly, in Figure 2.4 (a), we can see that this MDC approach



(a)



(b)

Figure 2.4: Performance comparison between MDLC and a set of MDC schemes with different quantization parameters for Foreman and Mobile.

outperforms MDLC between 300 Kbps and 700 Kbps. As the transmission rate increases further, MDLC may outperform MDC again, as all descriptions of MDC or MDLC can be delivered to the client on time and the reproduction quality at the decoder is bounded by the achievable reconstruction quality at the encoder. In summary, MDLC provides a more graceful bandwidth adaptation in the event of varying transmission bandwidth, compared to the sharp change in end-to-end quality typical of a MDC system. Meanwhile, source redundancy introduced in an MDLC codec improves its error resilience in a lossy packet environment.

2.4 Conclusions

In this chapter we proposed a new approach, multiple description layered coding (MDLC), which combines the hierarchical scalability of LC with the reliability introduced by MDC. The MDLC codec produces multiple redundant representations, enhancing the adaptation flexibility to varying network conditions. Experimental results show that the proposed MDLC system provides more robust and efficient video communication over a wider range of network scenarios and application requirements.

Chapter 3

Rate-Distortion Based Scheduling of Video with Multiple Decoding Path

3.1 Introduction

Internet multimedia applications, such as live video streaming, distance learning and video on demand services, are becoming increasingly popular. Given the best-effort service offered by the current Internet, video transmission is inevitably affected by the network variations in bandwidth, delay and packet loss rate, and thus it is imperative to provide some means to deal with the transmission impairments. Instead of traditional error-resilient encoding techniques that introduce redundancy in the bit stream level, this chapter extends the rate-distortion optimized streaming framework proposed in [19, 20] to operate on a general class of coding formats that explicitly support redundancy in their coding structure by, for example, producing multiple redundant representations of the video content. Note that in the absence of adaptation the redundancy levels may not

match those required by the actual network conditions. Here we propose an on-line rate-distortion based scheduling algorithm that can dynamically adjust the system's real-time redundancy to match the channel behavior so as to achieve better overall quality.

A variety of techniques have been proposed to address error control in the literature, including forward error correction, delay-constrained retransmission [32], intra/inter mode switching [99], reference picture selection [28, 37], dynamic packet dependency control [45], layered coding with unequal error protection [4], soft ARQ for layered streaming media [56], and multiple description coding [29, 89]. An important recent advance to video streaming is the rate-distortion optimized packet scheduling (RaDiO) framework initially proposed by Chou and Miao [19, 20]. This work formalized packet dependencies as a directed acyclic graph (DAG), prioritized packets based on their importance, and scheduled them so as to minimize a Lagrangian cost function combining expected distortion and rate. Some techniques have been proposed to reduce the complexity of the original algorithm. Miao and Ortega [49, 50] simplified the approach by running a greedy algorithm that explicitly combines the effects of data dependencies and delay constraints into a single importance metric. Chou and Sehgal [21] presented simplified methods to approximate the optimized policies. Chakareski *et al.* [10] proposed a family of simplified distortion models to approximate the end-to-end distortion produced by arbitrary packet loss patterns. Recent work by De Vleeschouwer *et al.* [80] improved the performance of greedy scheduling algorithm by delaying some packet scheduling decisions to avoid premature retransmissions. The sender-driven rate-distortion framework in [19] has also been extended into other transmission scenarios, including packet scheduling at the receiver [21], at an intermediate proxy [11], or taking into consideration path diversity [12].

Previous work on rate-distortion optimized video scheduling is mainly focused on encoding techniques, such as layered coding [44], which generate packets that can only be decoded following a *single decoding path* (SDP): a packet can be decoded *only* when all the packets it depends on are received and decodable. However, source codecs that explicitly support redundancy to combat transmission errors usually produce *multiple decoding paths* (MDP): there are multiple ways to decode a packet, each with a different distortion reduction depending on which packets, among those it depends on, are received. One example of these codecs is multiple description layered coding (MDLC) proposed in [83], which combines the hierarchical scalability of layered coding (LC) with the reliability of multiple description coding (MDC) so as to provide graceful adaptation over a wider range of application and network scenarios. Other coding examples with MDP include multiple independent encodings or decoding with error concealment.

The scheduling problem becomes more challenging when considering multiple decoding paths. In addition to challenges arising in the framework of [19, 20], such as delay constrained delivery, channel conditions and data dependency, the scheduling algorithm has to take into account the correlation or redundancy between data units, which is needed for end-to-end distortion estimation. The basic RaDiO framework [19, 20] addressed this problem using a simple DAG model to represent data dependency only, and is thus limited to coding scenarios that have a single decoding path. For example, it implicitly excludes the possibility of having multiple descriptions, in which several decoding choices are possible based on which descriptions are received. Cheung and Tan [15] introduced a more general formulation based on the DAG model to include the case where a packet can be decoded in different ways. They considered all possibilities of decoding and

delivery scenarios, which leads to significant increases in complexity. In our approach, we introduce new additional components on top of a DAG, in order to explicitly represent source redundancy among packets. Thus, compared to [15], our approach provides a more systematic way to represent source codecs that support multiple decoding paths with reasonable complexity.

This chapter, extending our prior work [83, 85], focuses on developing a general streaming framework for a class of scenarios that employ redundant source coding structures. In [83], we proposed a heuristic scheduling algorithm for a simplified MDLC codec with only I-frames, and then presented a preliminary version of our general streaming framework in [85]. The present chapter extends our prior work by generalizing the source model to include a general class of source coding approaches such as MDLC with motion prediction, multiple independent encodings and so on, by refining the optimization scheduling algorithms, by introducing an improved MDLC predictive coder, and by evaluating performance under various redundancies for a number of video sequences. Specifically, in this chapter, we first propose a new Directed Acyclic HyperGraph (DAHG) source model to represent both data dependencies and correlation between different video data units. The DAHG model introduces the concepts of multiple decodable states and multiple decoding paths, from which the expected end-to-end distortion D for a group of packets can be estimated accurately, when given a vector of packet loss probabilities, $\boldsymbol{\epsilon}$, for each packet in the group. In addition, a Taylor series expansion of D in terms of $\boldsymbol{\epsilon}$ reveals important properties for different coding scenarios, depending on whether source redundancy exists or not. We then propose two rate-distortion adaptive packet scheduling algorithms, one

based on Lagrangian optimization with the iterative descent approach proposed in [19] and another one based on a greedy solution derived from the Taylor expansion.

It is noted that, in addition to source redundancy explicitly produced at the encoding stage, the proposed streaming framework implicitly introduces a transport redundancy by allowing retransmission of a packet without waiting for either a negative acknowledgement (NAK) from the receiver or a timeout. In this case it is possible for the sender to transmit a packet multiple times so that more than one copy of a given packet may be correctly received at the decoder. We term this resulting rate penalty the *transport redundancy* introduced by the scheduling algorithm. This is different from most traditional ARQ approaches applied in video applications that only retransmit a packet upon the detection of a packet error or loss. This type of redundancy has not been explicitly studied in previous research. In this work, we investigate the impacts of both transport and source redundancy on the error control for a lossy packet network. From our experiments we make the following observations. First, regardless of whether source redundancy exists or not, a well-controlled transport redundancy through the Lagrangian optimized scheduling algorithm can improve the end-to-end performance in a delay-sensitive application. Second, in the absence of transport redundancy, source redundancy helps combat channel errors especially in high packet loss rate or under stringent delay constraints. Finally, these two types of redundancy can complement each other and achieve efficient video streaming even with very poor channel conditions, for example, at very high packet loss rates or relatively long RTT as compared to the end-to-end delay.

In this chapter, we use the MDLC codec proposed in Section 2.2.3 as an example to evaluate the scheduling algorithm performance under different types of redundancies.

Results demonstrate the benefits on error robustness provided by both source and transport redundancies, and show that our proposed system with both redundancies achieves the best end-to-end performance on real-time video communication over a wide range of network scenarios.

Rate-distortion optimized streaming with source redundancy has also been applied to multiple independent encodings [40] and decoding with error concealment [12]. Unlike these techniques which address a particular source coding approach, we formalize the general coding relation between data units in a more structured source model that can represent various source coding approaches including these two examples. Compared to stream switching often used in commercial streaming systems [9, 23], our approach enables finer switching at the packet level rather than the stream level, and further allows more flexible adaptation options than simple switching.

The chapter is organized as follows. In Section 3.2, we briefly review the basic RaDiO framework in [19]. Section 3.3 describes a general DAHG source model that uses the MDLC as an example, the expected end-to-end distortion, and the analysis of its Taylor expansion. Section 3.4 proposes the rate-distortion based scheduling algorithms based on the DAHG model, and describes the concept of transport redundancy. Simulation results are presented in Section 3.5. Conclusions and future work are discussed in Section 3.6.

3.2 Review of Basic RaDiO Framework

In this section we briefly review the rate-distortion optimized streaming framework of [19]. A compressed media stream is packetized into packets or data units. Here, we

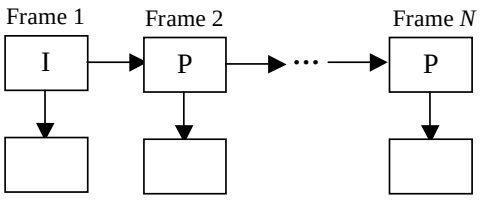


Figure 3.1: A DAG example for a LC system.

simply assume each data unit is put into one packet, and in the following discussion we do not differentiate between a data unit and a packet. The source dependencies between a group of data units are modelled as a directed acyclic graph (DAG), in which each vertex represents a data unit, and each directed edge from data unit i to data unit j indicates the decoding dependence of j on i, i.e., data unit j can only be decoded if i is received and decoded. Figure 3.1 shows a DAG representing a LC system containing a group of I-P frames, with each frame having a base layer and an enhancement layer. Note that this dependency structure corresponds to a typical fine granularity scalability (FGS) codec. Associated with each data unit l in the graph are three constant quantities: its size r_l in bytes, its time deadline t_l, i.e., the time by which it must arrive at the receiver to be useful for decoding, and its distortion value d_l, i.e., the amount by which the distortion of the decoded video will decrease if l is decoded on time at the receiver. The model implicitly assumes that when each data unit becomes decodable the total distortion is reduced by its distortion value.

The streaming system decides whether, when and how to transmit each data unit in a way that maximizes the playback quality at the decoder under the given network conditions and application requirements. This framework assumes that data units are transmitted at discrete intervals of time. At each transmission time, a data unit is

chosen for transmission from those whose deadlines fall within a limited time window. Transmission decisions for such a group of data units at discrete times can be described by a transmission policy $\boldsymbol{\pi}$. For a group of L data units, $\boldsymbol{\pi} = [\pi_1, \cdots, \pi_L]$, in which π_l is a binary vector indicating whether data unit l will be transmitted or not at each of the available transmission opportunities, unless there is an acknowledgement that l has been received. At each transmission time, the algorithm determines which data units to send by optimizing its transmission policy for the current transmission opportunity together with a complete plan for future transmission opportunities that will likely happen. The optimal policy $\boldsymbol{\pi}^*$ is the one that minimizes the expected Lagrangian cost function

$$J(\boldsymbol{\pi}) = D(\boldsymbol{\pi}) + \lambda R(\boldsymbol{\pi}), \tag{3.1}$$

where $D(\boldsymbol{\pi})$ is the expected end-to-end distortion and $R(\boldsymbol{\pi})$ is the expected transmission rate for a given $\boldsymbol{\pi}$. Based on the DAG model, $D(\boldsymbol{\pi})$ is given by

$$D(\boldsymbol{\pi}) = D_0 - \sum_l d_l \prod_{l' \preceq l} (1 - \epsilon(\pi_{l'})) \tag{3.2}$$

where D_0 is the distortion of the media stream if no packets are decoded, $\epsilon(\pi_l)$ is the packet loss probability of data unit l under policy π_l (strictly speaking, the probability that l is lost or does not arrive at the receiver on time), and $\prod_{l' \preceq l}(1 - \epsilon(\pi_{l'}))$ is the probability that l is decodable. $l' \preceq l$ refers to the set of data units that must arrive at the receiver for l to be decoded. The given policy $\boldsymbol{\pi}$ also induces an expected number of transmission times, $\beta(\pi_l)$, for each data unit l, and $R(\boldsymbol{\pi}) = \sum_l r_l \beta(\pi_l)$.

An iterative descent algorithm was proposed in [19] to find $\boldsymbol{\pi}^*$. The algorithm starts with an initial policy, and then proceeds to minimize (3.1) iteratively until $J(\boldsymbol{\pi})$ converges. At each iteration step, (3.1) is minimized with respect to π_l while fixing the transmission policies of other data units, $\pi_{l'}$, $l' \neq l$. The optimization is done for different data units in a round-robin order. To optimize π_l, (3.1) can be rewritten as $J(\pi_l) = \epsilon(\pi_l) + \frac{\lambda r_l}{a_l}\beta(\pi_l)$, where a_l is the partial derivative of (3.2) with respect to $\epsilon(\pi_l)$, indicating the sensitivity (or importance) of receiving data unit l to the overall distortion. $\boldsymbol{\pi}$ is re-optimized at each transmission opportunity to take into account the feedback information and possible changes of the group of data units since the previous transmission opportunity.

3.3 Source Modelling for Redundant Representations

3.3.1 Directed Acyclic Hypergraph (DAHG)

When a video sequence is encoded into multiple redundant representations, source redundancy is introduced between two data units, where each of them can be decoded independently to create different representations of the same source unit. Such examples include BL_1 and BL_2 corresponding to the same frame in MDLC, or data units that contain individual independent encodings of a frame with different quantization parameters. The key problem here is how to represent the redundancy between data units, and furthermore the possible availability of multiple decoding paths due to the redundancy.

To address this class of source coding formats, we introduce a new source model called Directed Acyclic HyperGraph (DAHG) to represent both dependency and redundancy

relationships between different video data units. A DAHG is a generalization of a DAG $G = (V, E)$ where

1. Each vertex $C \in V$, rather than being a simple node, is composed of a set of nodes, each pair of which is connected by an undirected edge. We name this type of vertex a "clique", representing a collection of data units that produce multiple redundant representations of the same source coding unit, such as a frame or a SNR layer of a frame in scalable coding. Each node (or data unit) in a clique represents one encoded version of this source unit, and an undirected edge connecting two nodes in the same clique indicates the redundancy between different encoded versions. A pair of nodes i and j are called *siblings*, and we write $i \sim j$.

2. An edge $(C_1, C_2) \in E$, directed from clique C_1 to clique C_2, is used to represent that decoding of C_2 is directly dependent on C_1. C_1 is said to be a *parent* of C_2, and C_2 is said to be a *child* of C_1. A path is a sequence of vertices such that from each of its vertices there is a directed edge to the next vertex in the sequence. If a path leads from C_1 to C_2, then C_1 is said to be an *ancestor* of C_2, and C_2 is said to be a *descendant* of C_1, written as $C_1 \prec C_2$ or $C_2 \succ C_1$. Each parent of C_2 is certainly an ancestor of C_2. On the other hand, C_1 being an ancestor but not a parent of C_2 indicates an undirected decoding dependence between C_1 and C_2. For example, this would be the case with last P-frame in a GOP (as C_2) depending on the first I-frame (as C_1) through the other intermediate P-frames.

Figure 3.2 shows an example DAHG for the proposed MDLC scheme shown in Figure 2.2. Each frame i contains a base layer clique C_{i1} and an enhancement layer clique C_{i2}.

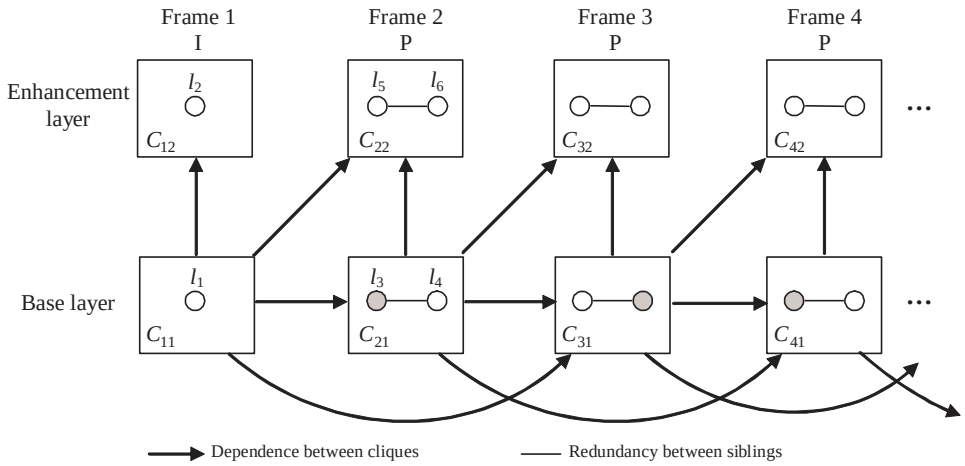


Figure 3.2: The DAHG model of the MDLC scheme shown in Figure 2.2. One of the two base layer nodes (filled with gray color), which has zero data size, is decoded as a copy or motion interpolation from the other description. We label each node sequentially as $l_1, l_2, \cdots$ starting from frame 1. Specifically, in frame 2, l_3, l_4, l_5 and l_6 correspond to BL_1, BL_2, EL_1 and EL_2, respectively.

Since the I-frame of each GOP is coded without redundancy, its base and enhancement layer cliques contain only one node each. Clique C_{i1} of each P frame i consists of two nodes representing BL_1 and BL_2, respectively. One of them is generated by copying (or using motion interpolation on) neighboring frames coded in the other description, such as l_3 of C_{21} in the figure. While this node does not require bits being sent, it does produce a distortion reduction. Clique C_{i2} contains nodes EL_1 and EL_2. Directed edges connecting cliques represent either SNR dependence or temporal dependence. There are two directed edges entering C_{32}, including (C_{31}, C_{32}) for SNR dependence and (C_{21}, C_{32}) for temporal dependence. Thus, C_{31} and C_{21} are parents of C_{32}. C_{11} is an ancestor of C_{32} as a path $(C_{11}, C_{21}, C_{31}, C_{32})$ leads from C_{11} to C_{32}. Figure 3.3 models multiple independent encodings of a video sequence, where the sequence is independently coded twice with different quantization steps using a typical non-scalable codec. Each clique contains two nodes to represent each encoded version. The same graph model can also

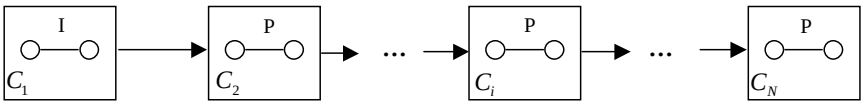


Figure 3.3: Another example of DAHG to represent multiple independent encodings of a video sequence. This model can also be used to represent error concealment.

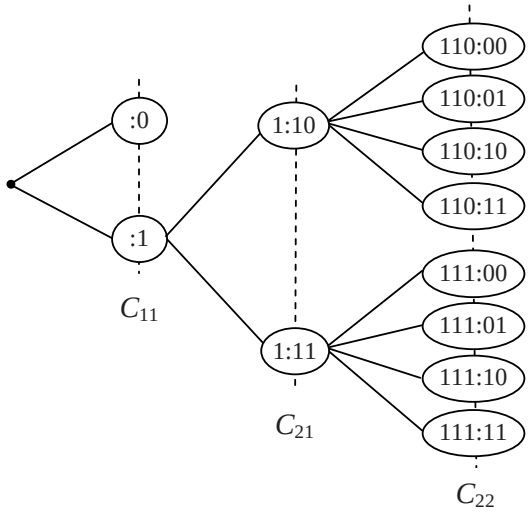
be applied to a simple "copy previous frame" error concealment method, where one of the two nodes in each clique represents a duplicate copy of the previous frame as an approximation of the current frame.

Assume that a clique contains N data units. Since each unit can be either received correctly or not received (due to loss or because it is not transmitted in the first place), there are a total of 2^N possible states for the clique. A *clique state* is represented by a length-N binary string s, with each bit indicating the status of a data unit in the clique. Let b_l denote the corresponding bit location of data unit l in s; the b_lth bit of s is 1 (mathematically, $s[b_l] = 1$) if l arrives at the receiver on time and is 0 otherwise. b_l is set to 1 for those nodes that have a size of zero bits, since they are regarded as being always received[1]. Zero state of a clique is then defined as the state such that no data units are received, and all the other states that have at least one data unit received are called non-zero states. Note that a non-zero clique state does not necessarily mean that this clique is decodable. Decoding of a clique also depends on the states of its ancestor cliques, which will be discussed later. In addition, it is convenient to define $B_C^{(s)} = \{l | l \in C, s[b_l] = 1\}$ and $\bar{B}_C^{(s)} = \{l | l \in C, s[b_l] = 0\}$ to represent two different sets of data units in C based on their state s.

[1]Examples of this type of nodes are given in Figures 3.2 and 3.3. Essentially these nodes are created to separate the direct contribution of a packet to reducing distortion, which requires transmission, from its indirect contribution via interpolation or error concealment, which requires no additional transmission rate once the original packet has been received.

In a directed acyclic graph, a decoding path leading to a vertex can be constructed as an ordered list of its ancestors in the decoding order. In past works that code a video sequence into a single encoded version, such as single description coding, a vertex has only 1/0 states, i.e., either received or not. Thus each vertex node along a decoding path must be received in order for the current node to be decoded, and this forms a single decoding path. In contrast, in the case of source coding with redundancy, a clique can be decoded once all its ancestor cliques are received in a non-zero state. Moreover, each clique in the ordered ancestor list can take multiple non-zero states, with different state combinations resulting in possibly different decoded versions of the current clique. A *decoding path* leading to clique C is then defined as a particular combination of all C's ancestor clique states. Multiple decoding paths become possible as each ancestor may have multiple clique states. In order to use the same mathematical notation, we simply assume there is one virtual decoding path leading to those cliques that do not have parents. Figure 3.4(a) shows the concept of multiple clique states and multiple decoding paths using the MDLC in Figure 3.2 as an example.

In general a complete set of decoding paths leading to C contains all the combinations of C's ancestor clique states. Thus theoretically the number of decoding paths may increase exponentially in the number of ancestor cliques preceding C. However, in practical pre-encoded applications, given a decoder implementation and possible simplification of source modelling, there are only a small number of effective decoding paths. For a given decoder implementation, a clique representing a source unit can only be decoded into a limited number of versions. Many of decoding paths producing the same reconstruction can thus be merged into one decoding path, while some other paths that lead to poor



	C_{11}	C_{21}	C_{22}
Clique state s	$[l_1]$	$[l_3\ l_4]$	$[l_5\ l_6]$
Clique state set S	{[0], [1]}	{[10], [11]}	{[00], [01], [10], [11]}
Decoding path q		$[l_1]$	$[l_1\ l_3\ l_4]$
Decoding path set Q		{[1]}	{[110], [111]}

(a)

Parameters of C_{21}

Decoding path	Distortion vector		Redundancy
	$\mathbf{d}(l_3)$	$\mathbf{d}(l_4)$	$\mathbf{I}_{C_{21}}[11]$
[1]	d_3	d_4	$\min(d_3, d_4)= d_3$

Parameters of C_{22}

Decoding path	Distortion vector		Redundancy
	$\mathbf{d}(l_5)$	$\mathbf{d}(l_6)$	$\mathbf{I}_{C_{22}}[11]$
[110]	$d_5^{(1)}$	0	0
[111]	$d_5^{(2)}$	$d_6^{(2)}$	$\min(d_5^{(2)}, d_6^{(2)})$

d_i : Distortion reduction of data unit l_i when there is only one decoding path
$d_i^{(j)}$: Distortion reduction of data unit l_i in the jth decoding path when there are multiple decoding paths
$\mathbf{I}_C[11]$: Redundancy of clique C at state s=[11]. The other components of the redundancy matrices are zero.
$\min(a,b)$: the minimum between a and b

(b)

Figure 3.4: Description of multiple clique states and multiple decoding paths using cliques C_{11}, C_{21} and C_{22} in Figure 3.2 as an example. (a) Multiple clique states and multiple decoding paths. In frame 2, l_3 is decoded to be a direct copy of the reconstructed frame 1, and l_4 produces a reconstructed frame with better quality than l_3. Each circle in the figure is labelled by a combination of decoding path and clique state in the form "decoding path : clique sate". A decoding path is represented by a concatenation of each ancestor clique state. Nothing before the colon in C_{11} indicates that it has no parents and there is only a virtual decoding path leading to C_{11}. (b) Distortion related parameters assigned to frame 2.

quality solutions are ignored by the decoder. For example, cross-description decoding of EL_2 based on BL_1 or EL_1 based on BL_2 is ignored since the information added by the cross enhancement layer is very small. In addition, even when a decoder supports certain decoding paths, a source model for the purpose of scheduling can also choose to discard some of these paths in order to reduce the computation complexity, at the penalty of some performance loss.

In summary, DAHG is different from DAG mainly in two aspects: (1) multiple decoding paths in DAHG vs. single decoding path in DAG, and (2) multiple decodable clique states in DAHG vs. 0/1 state of the data unit (i.e., it is either decodable or not) in DAG. Estimating expected end-to-end distortion under a DAHG model will be discussed in detail in Section 3.3.3.

3.3.2 Parameters Associated with DAHG

As in [19], each data unit l has a size r_l in bytes and a time deadline t_l by which it must arrive at the receiver to be useful for decoding. However, the distortion reduction of a data unit in a DAHG model can take different values depending on the decoding path in which it is decoded. Let Q_C be the set of decoding paths leading to C. Then we can represent the distortion reduction of data unit l in the clique by a *distortion vector* $\mathbf{d}_l = [d_l^{(1)}, d_l^{(2)}, ..., d_l^{(q)}, ..., d_l^{(|Q_c|)}]$, where $d_l^{(q)}$ is the distortion reduction if l is decoded in the qth decoding path, and $|.|$ denotes the cardinality of the set. Setting $d_l^{(q)}$ to 0 will force the scheduler not to transmit data unit l given the qth decoding path. This can be used to eliminate certain undesirable clique state combinations, and thus reduce the number of effective decoding paths in a DAHG.

Though each of the data units in clique C can produce a certain distortion reduction, the total distortion reduction when more than one data unit is received correctly is usually less than the sum of their respective distortion reductions. Let S_C be the set of all clique states in C. We introduce a *redundancy matrix* $\mathbf{I}_C = [I_C^{(s,q)}]$ of dimension $|S_C| \times |Q_C|$, to represent the redundancy between different data units inside the same clique C. The redundancy of C, when it is in state s and decoded in the qth decoding path, is stored as an entry in row s and column q of the redundancy matrix. $I_C^{(s,q)}$ is defined as

$$I_C^{(s,q)} = \sum_{l \in B_C^{(s)}} d_l^{(q)} - d_C^{(s,q)}, \tag{3.3}$$

where $d_C^{(s,q)}$ is the total distortion reduction of C if it is decoded in state s and the qth decoding path. An important property of this model is that, as the DAG model, the distortion reduction is still additive at the clique level; however, the amount by which the distortion decreases when a node is decoded depends not only on the state of its ancestor cliques but also on whether its siblings in the same clique are decodable. Figure 3.4(b) lists the distortion vectors and redundancy matrices for frame 2 of the MDLC example shown in Figure 3.2.

3.3.3 Expected End-to-End Distortion

Suppose we already have a DAHG model to represent a group of L data units, with each data unit being packetized into one packet. We can now estimate the expected end-to-end distortion of this group of packets (GOPkt) when given a vector of packet loss probability (PLP) providing a loss probability for each packet in the group. Recall that a packet is

considered lost if it is either lost or arrives at the decoder too late to be played. We now define the "*transmission state*" as the PLP vector which accounts for the transmission schedules and the channel conditions. Let ϵ_l be the PLP of packet $l \in \{1, ..., L\}$ and let $\boldsymbol{\epsilon} = [\epsilon_1, ..., \epsilon_L]$ be the real-time transmission state. Computation of expected distortion in a DAHG for a given $\boldsymbol{\epsilon}$ differs from that in [19] by introducing two new concepts, multiple decoding paths and multiple decodable clique states.

To help us write an expression of the expected distortion, we first derive some related probabilities. The probability of occurrence of clique state s is given by

$$p_C^{(s)} = \prod_{l \in B_C^{(s)}} (1 - \epsilon_l) \prod_{l' \in \bar{B}_C^{(s)}} \epsilon_{l'} \tag{3.4}$$

Recall that a decoding path leading to clique C is defined by a particular combination of the clique states of all its ancestors. Thus the probability of occurrence of decoding path q can be written in terms of the probabilities of those clique states as

$$p_C^{(q)} = \prod_{C' \prec C, s_{C'} \in q} p_C^{(s_{C'})} = \prod_{l \in A_C^{(q)}} (1 - \epsilon_l) \prod_{l' \in \bar{A}_C^{(q)}} \epsilon_{l'} \tag{3.5}$$

where $A_C^{(q)} = \bigcup_{C' \prec C, s_{C'} \in q} B_C^{(s_{C'})}$, and $\bar{A}_C^{(q)} = \bigcup_{C' \prec C, s_{C'} \in q} \bar{B}_C^{(s_{C'})}$. We now can write the expected distortion as a function of the transmission state

$$D(\boldsymbol{\epsilon}) = D_0 - \sum_C \sum_{q \in Q_C} p_C^{(q)} [\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)}] \tag{3.6}$$

where D_0 is the distortion of the GOPkt if no packets are decoded, $d_C^{(s,q)} = \sum_{l \in B_C^{(s)}} d_l^{(q)} - I_C^{(s,q)}$ directly derived from (3.3), and $p_C^{(s)}$ and $p_C^{(q)}$ are defined in (3.4) and (3.5), respectively.

Both transmitting and receiving a packet cause a state transition from a state $\boldsymbol{\epsilon_1}$ to another state $\boldsymbol{\epsilon_2}$. The Taylor expansion of D in terms of $\boldsymbol{\epsilon}$ reveals different characteristics of state transitions for different coding scenarios. The distortion reduction when receiving a packet in a multiple-decoding-path scenario depends on more factors than that in a single-decoding-path scenario, because the redundancy between packets plays an important role. Thus, in this case, an optimal scheduling algorithm should be designed to take into account both dependency and redundancy such that the end-to-end distortion is minimized at the decoder.

The Taylor expansion of (3.6) at the current state $\tilde{\boldsymbol{\epsilon}}$ is given by

$$\begin{aligned} D(\boldsymbol{\epsilon}) &= \sum_{k=0}^{\infty} \big[\frac{1}{k!}(\Delta\boldsymbol{\epsilon} \cdot \nabla_{\boldsymbol{\epsilon'}})^k D(\boldsymbol{\epsilon'})\big]_{\boldsymbol{\epsilon'}=\tilde{\boldsymbol{\epsilon}}} \\ &= D(\tilde{\boldsymbol{\epsilon}}) + \sum_i a_i(\epsilon_i - \tilde{\epsilon}_i) + \sum_{i,j} a_{ij}(\epsilon_i - \tilde{\epsilon}_i)(\epsilon_j - \tilde{\epsilon}_j) + \dots \end{aligned} \tag{3.7}$$

where we denote $a_i = \frac{\partial D}{\partial \epsilon_i}$ the first-order partial derivative of D with respect to ϵ_i, $a_{ij} = \frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j}$ the second-order partial derivative, and so on. Note that (3.7) only contains linear terms, since

$$\forall\ 1 \le j \le k,\ \text{if } \exists\ m_j \ge 2,\ \text{then } \frac{\partial^n D}{\partial \epsilon_{i_1}^{m_1}, \cdots, \partial \epsilon_{i_k}^{m_k}} = 0$$

derived directly from (3.6). a_i indicates the importance of packet i in terms of its contribution to the overall distortion reduction given the current transmission state. As receiving a packet will not increase the overall distortion for any coding application, $a_i \geq 0$ for any i. The second or higher-order terms take effect when there is more than one packet whose PLP has changed from a reference state. For example, $\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j}$ shows that a future change of ϵ_j, as packet j is transmitted or its ACK/NAK is received, may affect the importance of transmitting packet i at the current time. To see this, we approximate a_i by its first-order Taylor expansion at $\tilde{\boldsymbol{\epsilon}}$, $a_i(\boldsymbol{\epsilon}) \approx a_i(\tilde{\boldsymbol{\epsilon}}) + \sum_j a_{ij}(\epsilon_j - \tilde{\epsilon}_j)$. Assume that packet j will be transmitted or will arrive at the receiver when the state transits from $\tilde{\boldsymbol{\epsilon}}$ to $\boldsymbol{\epsilon}$, then $\epsilon_j < \tilde{\epsilon}_j$. In this case, a_{ij} will lead to a change in a_i as follows: when $a_{ij} < 0$, a_i increases and vice versa. In other words, the transmission or arrival of packet j may increase or reduce the current importance of packet i depending on the sign of a_{ij}.

Now we compare the difference between single decoding path and multiple decoding paths in terms of the properties of the first-order and second-order partial derivatives of D. First consider the single decoding path case. The expected end-to-end distortion in this case is given in (3.2). We derive its partial derivatives as

$$\frac{\partial D}{\partial \epsilon_i} = \sum_{l \succeq i} d_l \prod_{l' \preceq l, l' \neq i} (1 - \epsilon_{l'}) \tag{3.8}$$

$$\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j} = -\sum_{l \succeq i,j} d_l \prod_{l' \preceq l, l' \neq i,j} (1 - \epsilon_{l'}) \tag{3.9}$$

The right hand side of (3.8) can be written as the sum of two terms f_1 and f_2, where $f_1 = d_i \prod_{l' \prec i}(1 - \epsilon_{l'})$ corresponds to the original distortion of packet i weighted by the probability of receiving all its ancestors, and $f_2 = \sum_{l \succ i} d_l \prod_{l' \preceq l, l' \neq i}(1 - \epsilon_{l'})$ indicates the

importance of packet i to its descendant packets. From (3.9), we can conclude $\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j} \leq 0$ for any i and j, since $\epsilon_{l'} \leq 1$ for any l'.

When multiple decoding paths are possible, we derive its first-order derivative from (3.6) as

$$\frac{\partial D}{\partial \epsilon_i} = f_1 + f_2 + f_3 + f_4 \tag{3.10}$$

with

$$\begin{aligned}
f_1 &= \sum_{q \in Q_C} p_C^{(q)} \Big[\sum_{s \in S_C, i \in B_C^{(s)}} d_C^{(s,q)} \prod_{l \in B_C^{(s)}, l \neq i} (1-\epsilon_l) \prod_{l \in \bar{B}_C^{(s)}} \epsilon_l \Big] \\
f_2 &= -\sum_{q \in Q_C} p_C^{(q)} \Big[\sum_{s \in S_C, i \in \bar{B}_C^{(s)}} d_C^{(s,q)} \prod_{l \in B_C^{(s)}} (1-\epsilon_l) \prod_{l \in \bar{B}_C^{(s)}, l \neq i} \epsilon_l \Big] \\
f_3 &= \sum_{C \succ C_i} \sum_{q \in Q_C, i \in A_C^{(q)}} \Big[\prod_{l \in A_C^{(q)}, l \neq i} (1-\epsilon_l) \prod_{l \in \bar{A}_C^{(q)}} \epsilon_l \Big] \cdot \Big[\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)} \Big] \\
f_4 &= -\sum_{C \succ C_i} \sum_{q \in Q_C, i \in \bar{A}_C^{(q)}} \Big[\prod_{l \in A_C^{(q)}} (1-\epsilon_l) \prod_{l \in \bar{A}_C^{(q)}, l \neq i} \epsilon_l \Big] \cdot \Big[\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)} \Big]
\end{aligned}$$

where C_i represents the clique that contains packet i. f_1 indicates the packet importance due to its own distortion reduction; f_2 represents redundancy when both i and its sibling packets are received; f_3 shows the distortion reduction achieved by the descendant cliques of C_i in the decoding paths that require i to be received; and f_4 represents the impact of receiving i on the descendant cliques of C_i in the remaining decoding paths which do not require i to be received. The signs of these terms indicate whether it is desirable to transmit i or not when different packets have been received at the decoder in the past, as a positive (or negative) term will increase (or decrease) the value of $\frac{\partial D}{\partial \epsilon_i}$.

We now give a concrete example to clearly illustrate how to calculate the expected distortion for the MDLC scheme in Figure 3.2 and the properties of its partial derivatives

in the case of multiple decoding paths. Here we only consider cliques C_{11}, C_{21} and C_{22}. Let ϵ_i denote the packet loss probability of data unit l_i in Figure 3.2. Since l_3 is a direct copy of l_1 without the need to send any bits, $\epsilon_3 = 0$. We use the notation of Figure 3.4 for distortion related parameters.

(1) *Calculation of expected distortion:* The expected distortion D is given by

$$D = D_0 - \Delta D_{C_{11}} - \Delta D_{C_{21}} - \Delta D_{C_{22}} \tag{3.11}$$

where

$$\begin{aligned}
\Delta D_{C_{11}} &= (1-\epsilon_1)d_1 \\
\Delta D_{C_{21}} &= (1-\epsilon_1)\begin{bmatrix} \epsilon_4 & 1-\epsilon_4 \end{bmatrix} \times \begin{bmatrix} d_3 \\ d_4 \end{bmatrix} \\
\Delta D_{C_{22}} &= (1-\epsilon_1)\begin{bmatrix} \epsilon_4 & 1-\epsilon_4 \end{bmatrix} \times \boldsymbol{d}_{C_{22}} \times \boldsymbol{p}_{C_{22}}
\end{aligned}$$

with $\boldsymbol{d}_{C_{22}} = \begin{bmatrix} 0 & d_5^{(1)} & d_5^{(1)} \\ d_6^{(2)} & d_5^{(2)} & \max(d_5^{(2)}, d_6^{(2)}) \end{bmatrix}$, $\boldsymbol{p}_{C_{22}} = \begin{bmatrix} \epsilon_5(1-\epsilon_6) \\ (1-\epsilon_5)\epsilon_6 \\ (1-\epsilon_5)(1-\epsilon_6) \end{bmatrix}$ corresponding to the non-zero clique states of C_{22} in the order [01], [10] and [11]. Each entry of $\boldsymbol{d}_{C_{22}}$ at row q and column s gives the distortion reduction of C_{22} along the qth decoding path in state s. The sth element of $\boldsymbol{p}_{C_{22}}$ is the probability of occurrence of C_{22} at state s.

(2) *First-order partial derivatives:* Take l_4 as an example to show the importance of $\frac{\partial D}{\partial \epsilon}$ to the system's behavior. Written in the same way as (3.10), $\frac{\partial D}{\partial \epsilon_4}$ is derived from (3.11) with $f_1 = (1-\epsilon_1)d_4$, $f_2 = -(1-\epsilon_1)d_3$, $f_3 = (1-\epsilon_1)(\boldsymbol{d}_{C_{22}}(2) \times \boldsymbol{p}_{C_{22}})$, and

$f_4 = -(1-\epsilon_1)(\boldsymbol{d}_{C_{22}}(1) \times \boldsymbol{p}_{C_{22}})$, where $\boldsymbol{d}_{C_{22}}(q)$ $(q = 1, 2)$ is the qth row of $\boldsymbol{d}_{C_{22}}$. From the sign of the above terms, we can see that transmission of l_3 is more favorable when f_1 and f_3 are significant, and less favorable when f_2 and f_4 become dominant.

(3) *Second-order partial derivatives:* Assuming now l_1 and l_4 have been transmitted but without receiving acknowledgements yet, the current state $\tilde{\boldsymbol{\epsilon}} = [\epsilon_1, \epsilon_3, \epsilon_4, \epsilon_5, \epsilon_6] = [\epsilon_1, 0, \epsilon_4, 1, 1]$, $0 \leq \epsilon_1, \epsilon_4 \leq 1$. Consider the following second-order derivatives at $\tilde{\boldsymbol{\epsilon}}$,

- $\frac{\partial^2 D}{\partial \epsilon_4 \partial \epsilon_6} = -(1-\epsilon_1) d_6^{(2)} \leq 0$, since l_6 is dependent on l_4 for decoding;

- $\frac{\partial^2 D}{\partial \epsilon_5 \partial \epsilon_6} = (1-\epsilon_1)(1-\epsilon_4) I_{C_{22}}[11] \geq 0$, since l_5 and l_6 have redundancy with each other.

Though it is complicated to derive a general equation of $\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j}$ from (3.6), we can see from the above example that, in the case of multiple decoding paths, $\frac{\partial^2 D}{\partial \epsilon_i \partial \epsilon_j}$ can be either non-negative or non-positive. In contrast, $\frac{\partial D}{\partial \epsilon_i \partial \epsilon_j} \leq 0$ for single decoding path. This shows that, in the case of single decoding path, the arrival of one packet at the receiver can increase, or at least not reduce the importance of the other packets, in terms of distortion reduction. However, when there are multiple decoding paths, due to the redundancy between packets which affects the high-order terms, the future transmission of packets may decrease the current importance value of a packet that contains redundant information.

3.4 Scheduling Algorithms with DAHG

In this section we study two rate-distortion adaptive packet scheduling algorithms using our proposed DAHG source model, one based on Lagrangian optimization using an iterative descent algorithm [19, 20], and another one based on a greedy solution derived from the Taylor analysis in Section 3.3.3. Finally, we introduce the concept of transport redundancy in terms of the retransmission penalty observed at the client, and discuss its role in both scheduling algorithms.

3.4.1 System Architecture

Figure 3.5 shows an end-to-end video transmission system, in which each video frame is encoded, transmitted and decoded in real-time within some acceptable delay period. The input video is compressed into multiple redundant representations, e.g., using MDLC. For a packet-switched network, these streams are packetized and then fed into the transmission buffer. At each transmission time t, we make a selection decision only among packets whose playback deadlines fall within a time-varying transmission window $[lag(t), lead(t)]$. The time window will advance with t, and thus each packet has a limited number of transmission opportunities. $lag(t)$ is defined such that any packet whose playback deadline is earlier than $lag(t)$ could not arrive at the receiver on time if it were transmitted at t. $lead(t)$ implies the earliest time that a packet is eligible for transmission. [19] has proposed a number of ways to set $lead(t)$ by considering the receiver buffer implementation and the application playback delay. Here we assume the end-to-end delay for each frame will be constant and equal to the initial playback delay w. Thus, $lead(t) = lag(t) + w$.

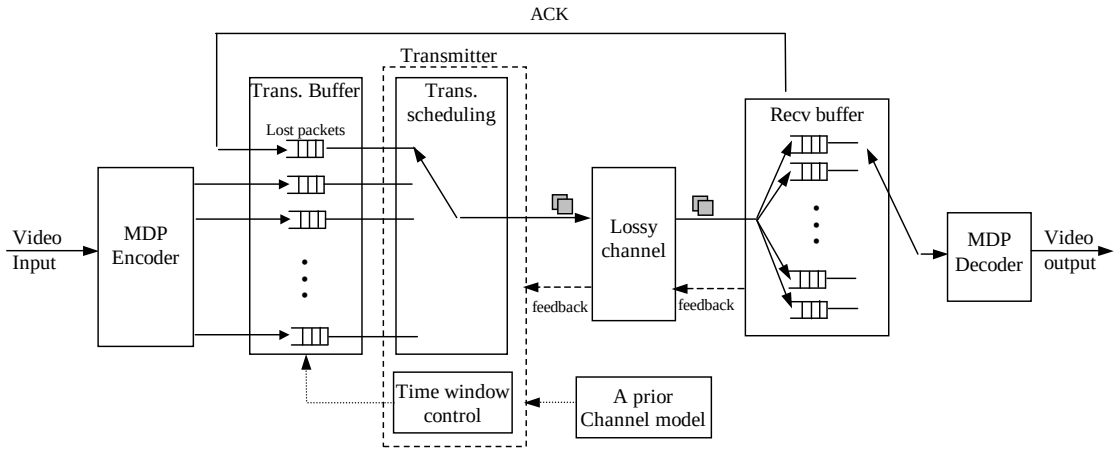


Figure 3.5: Streaming system architecture.

Each transmission at time t is subjected to a constraint of the admissible channel rates during this time interval. The receiver sends an acknowledgement (ACK) back to the sender as soon as it receives a packet. With the feedback information, the sender can estimate the channel conditions such as packet loss rate and round-trip time (RTT).

In our research, we simply model the network as an i.i.d. packet erasure channel with a fixed RTT. That means that a packet sent at t is lost with probability ϵ independent of t. By time $t + \text{RTT}$, the sender will receive an ACK if the packet is received at the decoder; otherwise the packet is considered lost or corrupted. We also assume that the back channel is error-free. Thus, given the transmission policy that there are n transmission times of packet l in the last RTT, the expected PLP of packet l at time t is given by

$$\epsilon_l = \begin{cases} 0 & \text{if the sender has received an ACK of packet } l \text{ by } t, \\ \epsilon^n & \text{otherwise.} \end{cases} \tag{3.12}$$

More complicated network models, such as random network delay and lossy back channel, can be easily combined into our streaming architecture and scheduling algorithms. The

major difference for various network models is how to estimate the expected packet loss probability in (3.12) given a transmission policy. This part has been carefully studied in [19]. In this chapter, we focus on the design of an efficient scheduling algorithm by taking into account both dependency and redundancy between packets.

3.4.2 Optimization Problem Formulation

The goal of scheduling is to minimize the playback distortion for a streaming session, by adapting to the network conditions and application requirements. Though we work with a more general streaming framework that allows multiple decoding paths, we can follow the same problem formulation as originally proposed in [19] for streaming applications with single decoding path. Suppose we wish to transmit a group of L packets whose playback deadlines fall in a limited time window, and the packets are transmitted at discrete time intervals evenly distributed in a time window with a maximum of N transmission opportunities. Let $\pi_l = [v_0, \cdots, v_{N-1}]$ be the transmission policy for packet l along the N transmission opportunities, where $v_i = 1$ indicates "send packet l" and $v_i = 0$ "do not send packet l" at the ith time interval. We are interested in finding an optimal transmission policy $\boldsymbol{\pi}^* = [\pi_1, \cdots, \pi_L]$ for this group of packets such that the expected end-to-end distortion is minimized subject to the data rate constraint, i.e.,

$$\boldsymbol{\pi}^* = \underset{\boldsymbol{\pi}: R(\boldsymbol{\pi}) \leq R_b}{\operatorname{argmin}} \; D(\boldsymbol{\pi}). \tag{3.13}$$

Since the expected PLP ϵ_l for packet l is a function of its transmission policy π_l, the expected end-to-end distortion D also depends on $\boldsymbol{\pi}$. Note that we consider expected

distortion because there is uncertainty about the actual decoded video quality; changes in channel bandwidth, packet loss rate, and so forth will affect the quality of received video.

3.4.3 Lagrangian Optimization Algorithm

As proposed in [19], the constrained optimization problem in (3.13) can be cast as an unconstrained optimization problem using a Lagrange multiplier λ,

$$\boldsymbol{\pi}^* = \underset{\boldsymbol{\pi}}{\operatorname{argmin}}\, D(\boldsymbol{\pi}) + \lambda R(\boldsymbol{\pi}). \tag{3.14}$$

Let $\beta(\pi_l)$ and $\epsilon(\pi_l)$ be the expected number of transmission times and the expected PLP for packet l under π_l, respectively. Then the expected rate of the group of L packets $R(\boldsymbol{\pi}) = \sum_l r_l \beta(\pi_l)$, and the expected distortion $D(\boldsymbol{\pi})$ is given by (3.6) using the DAHG model. Our proposed scheduling algorithm is composed of two components: (1) at each transmission time t, the iterative descent optimization algorithm proposed in [19] is used to update $\boldsymbol{\pi}^*$ for a given λ, by taking into account the source rate-distortion information, current channel condition, transmission history and receiver feedback; (2) a window-based rate-control algorithm is applied regularly (e.g., at each transmission time) to adjust λ such that the average output rate of the scheduler is matched to the channel bandwidth.

First, the iterative descent algorithm in [19] is used to optimize $\boldsymbol{\pi}^*$ for coding applications with multiple decoding paths. For completeness, the Lagrangian optimization algorithm tailored to our DAHG model is summarized in Algorithm 1. The major difference from [19] is that, at each optimization step, we derive the expected distortion from

Algorithm 1 LAGRANGIAN(t, λ, $\boldsymbol{\pi}_{t-1}$)

1: $n = 0$: initialize $\pi_l = \{\pi_{l,t-1}, 0, \ldots, 0\}$ for each packet l, and calculate $\epsilon_l = \epsilon(\pi_l)$, $\beta_l = \beta(\pi_l)$, $D = D_0 - \sum_C \sum_{q \in Q_C} p_C^{(q)} [\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)}]$, $R = \sum_l r_l \beta_l$, $J = D + \lambda R$
2: **repeat**
3: $n = n + 1$
4: select packet l to optimize at step n in a round-robin order
5: $a_l = \frac{\partial D}{\partial \epsilon_l}$ obtained from (3.10)
6: $\pi_l^* = \text{argmin}_{\pi_l} \epsilon(\pi_l) + \frac{\lambda r_l}{a_l} \beta(\pi_l)$
7: $\epsilon_l = \epsilon(\pi_l^*)$, $\beta_l = \beta(\pi_l^*)$, $D = D_0 - \sum_C \sum_{q \in Q_C} p_C^{(q)} [\sum_{s \in S_C} p_C^{(s)} d_C^{(s,q)}]$, $R = \sum_l r_l \beta_l$, $J = D + \lambda R$
8: **until** $\left| J^{(n)} - J^{(n-1)} \right| <$ Threshold
9: **return** $\boldsymbol{\pi}^* = [\pi_1^*, \cdots, \pi_L^*]$

a DAHG model instead of a DAG, as the DAHG can well represent both dependency and redundancy between packets. The input parameter $\boldsymbol{\pi}_{t-1}$ represents the optimal transmission policy determined at previous time $t-1$. Since all the future transmission plans following current time t will be re-optimized, the function LAGRANGIAN is only interested in the segment of $\boldsymbol{\pi}_{t-1}$ that stores the transmission history up to $t-1$. Let $\pi_{l,t-1}$ denote the lth component of this past segment in $\boldsymbol{\pi}_{t-1}$. We first initialize the transmission policy π_l of each packet to be the one with no further transmissions, i.e., setting all the future transmission actions as 0 in Algorithm 1. At each iteration step, the Lagrangian cost J is minimized with respect to π_l of a selected packet l while keeping the policies of all other packets fixed. Upon convergence, π_l of each packet is optimized for its complete window of transmission opportunities. Then the transmitter takes transmission actions at t, and the optimization procedure will be repeated at $t+1$.

Second, in order to approach the channel bandwidth limit, we propose a window-based rate control scheme. That is, at each time, λ is fixed for all packets in the transmission window. The rate budget R_b is increased when a new frame enters into the transmission

Algorithm 2 WINDOW_BASED_RATE_CONTROL(t, R_b)

1: **if** $t = 0$ **then**
2: $\quad R_b = 0$
3: **if** M new frames come in **then**
4: $\quad R_b = R_b + M *$ channel bandwidth $*$ frame interval
5: use bi-section algorithm to find an appropriate λ with rate constraint R_b
6: call LAGRANGIAN(t, λ, π_{t-1})
7: $R_b = R_b - \sum_{l:\pi_l(t)=1} r_l$
8: **return** R_b

window, and decreased when packets are sent out. At each transmission time, we apply the bisection algorithm to find an appropriate λ for R_b. This approach is different from those that fix λ for each group of frames or the whole session in that it can quickly respond to channel bandwidth changes and use the bandwidth in a more efficient way. The rate control algorithm is summarized in Algorithm 2.

3.4.4 Greedy Algorithm

Since only the current transmission action in $\boldsymbol{\pi}$ is used at any given time, instead of determining the complete transmission policy for each packet over all possible transmission opportunities (e.g., as used in the above Lagrangian optimization), we could choose to use a greedy approach by selecting the currently most important packet from the group of L candidate packets. Previous research work [50] has proposed similar solutions for single-decoding-path applications. Here, we derive the greedy algorithm for multiple-decoding-path codecs from the Taylor expansion of the expected distortion. Given the past transmission history of packet i, let $\boldsymbol{\pi}_{i,0}$ be a transmission schedule such that packet i is not transmitted at the current time t and all future time steps, and let $\boldsymbol{\pi}_{i,1}$ be the

same transmission schedule as $\boldsymbol{\pi}_{i,0}$ except that packet i will be transmitted at t. Sending packet i at t induces a state transition from $\boldsymbol{\epsilon}(\boldsymbol{\pi}_{i,0})$ to $\boldsymbol{\epsilon}(\boldsymbol{\pi}_{i,1})$, and thus leads to a distortion reduction by

$$\Delta D_i^{(t)} = D(\boldsymbol{\epsilon}(\boldsymbol{\pi}_{i,0})) - D(\boldsymbol{\epsilon}(\boldsymbol{\pi}_{i,1})) = a_i(\epsilon_{i,0} - \epsilon_{i,1}) = a_i(1-\epsilon)\epsilon_{i,0} \tag{3.15}$$

derived from (3.7), where $\epsilon_{i,0}$ and $\epsilon_{i,1}$ are the PLP of packet i given the schedule $\boldsymbol{\pi}_{i,0}$ or $\boldsymbol{\pi}_{i,1}$, respectively. In fact, $\epsilon_{i,0}$ is the expected PLP of packet i at t given its transmission history as calculated in (3.12), and we simplify the notation to ϵ_i. $\Delta D_i^{(t)}$ indicates the importance of sending packet i at the current time t when no further transmissions are considered. To favor packets with early playback deadlines, we introduce a multiplier ϵ^{m_i} in (3.15), where m_i is designed to approximate the number of possible retransmissions by[2]

$$m_i = (t_i - t)/RTT. \tag{3.16}$$

That is because the future possible transmissions for packet i will decrease the importance of sending it at t. Ignoring the constant term $(1-\epsilon)$, for comparing the importance of sending each packet at t and taking into account the packet size r_i, we have the metric

$$c_i = \epsilon^{m_i}\epsilon_i \frac{a_i}{r_i} \tag{3.17}$$

for each packet and select the one with the largest c_i to send. Note that a_i is calculated at the current state with the assumption that there are no future transmissions of other

[2]Strictly speaking, the number of possible retransmissions can be much larger than the given m_i for a system that allows retransmissions without waiting.

Algorithm 3 GREEDY

1: **for all** $1 \leq i \leq L$ **do**
2: $\quad c_i = \epsilon^{m_i} \epsilon_i \frac{a_i}{r_i}$
3: Find the largest c_i, say j (i.e. $c_j \geq c_i$ for any $i \neq j$)
4: **return** j

packets. Algorithm 3 summarizes the proposed greedy technique. At each transmission time, we choose the most important packets to send by running this algorithm iteratively until the channel rate allocated to this time interval is used up.

A main problem of the greedy algorithm is that it ignores the possibility of future transmissions of other packets. As Section 3.3.3 points out, for applications with multiple decoding paths, the future transmission of a packet may either increase or decrease the importance value of another packet depending on their coding relation. Thus, in an optimal algorithm future transmission probabilities of packets would have increased impact, through the higher-order derivatives of the Taylor expansion, on the decision at the current transmission opportunity. Another problem may arise from possible future retransmissions of the packet itself, for which this algorithm introduces a multiplier on the importance metric to approximate this impact on the current decision. In Section 3.5, we will see that the greedy algorithm experiences a certain performance loss compared to Lagrangian optimization. Other work has studied improved greedy scheduling algorithm to address these problems. Our previous MDLC work in [83] proposed a double time window control to intentionally introduce an extra waiting period for $MD2$ such that it can only be transmitted relatively safely in a future time to avoid unnecessary redundancy with $MD1$. This helps when the acknowledgements generated by early transmissions of $MD1$ are likely to arrive at the sender soon. In order to avoid the penalty introduced

by premature retransmissions, [80] proposed to delay some packet scheduling decisions. However, for a general coding scenario that provides multiple decoding paths, we have not achieved a systematic solution to address the possible future (re)transmissions for the packet itself and its related packets (through either dependency or redundancy). We are now working on a possible solution by taking into account the higher-order partial derivatives described in Section 3.3.3.

3.4.5 Transport Redundancy

Traditional ARQ approaches request retransmission only upon detection of lost or overly delayed packets. Thus the number of retransmissions is very limited for delay constrained real-time video communication. In comparison, our extended streaming framework, together with the one originally proposed in [19] for single decoding path, allows unlimited retransmission of a packet before the playback deadline in the sense that it can retransmit a packet without waiting for a timeout or NAK from the receiver. This approach essentially relieves the delay problem caused by retransmissions. However, it may introduce a rate penalty when both retransmitted and original packets are correctly received at the decoder. We call this "transport redundancy", since the client receives redundant information[3].

One possible variation of the proposed scheduling algorithms is to mimic the traditional ARQ systems by limiting the retransmission of a packet until the last transmission of this packet has not been acknowledged within a predefined timeout. Based on our

[3]If the original packet is not correctly received at the decoder, the duplicated packet contributes to the end-to-end distortion, and thus here we do not count it as a transport-redundant packet.

system assumption with a fixed RTT, the timeout is simply defined to be equal to one RTT. In other systems where the network produces a random delay, the timeout can be set as the mean RTT plus some tolerance (e.g., three times the standard deviation of the RTT, as is frequently used in ARQ systems). This is different from the original scheduling algorithms in that it completely or almost completely avoids the cost penalty due to the transport redundancy. However, if retransmission is controlled appropriately in the case where there is no waiting, the end-to-end performance can be improved without introducing longer delay. We will compare the performance of the scheduling algorithms with or without transport redundancy in Section 3.5.

3.4.6 Complexity Analysis

The complexity of the Lagrangian optimization approach is on the order of $N_\lambda N_i L 2^N$ at each transmission time, where N_i is the number of iterations performed until the algorithm converges for a given λ, and N_λ is the number of iterations for the rate control algorithm to adjust λ to meet the rate limit. The time period to adjust λ could cross multiple transmission times in order to reduce the complexity. L is the number of packets available for transmission in the time window, and N is the number of transmission opportunities of a packet. The complexity of the Greedy approach is $O(L)$ since each packet only needs to be traversed once to choose the most important packet to send at a given time. Note that the limited retransmission variants of the proposed algorithms lead to decreases in complexity as the number of packets to be considered for transmission at each transmission opportunity decreases. Instead of considering L, we consider only those that have not been transmitted or have been transmitted in the distant past (e.g., one

RTT ago) without acknowledgement. In addition, for Lagrangian optimization algorithm, the searching space of an optimal transmission policy for each packet is greatly reduced by the retransmission limitation.

3.5 Experimental Results

In this section, we examine the performance of the proposed streaming framework for video codecs with multiple decoding paths. The video sequences are coded using the proposed MDLC approach based on MPEG-4 FGST. Three standard test sequences are used: Akiyo (QCIF), Foreman (QCIF) and Mobile (CIF). The first 200 frames of each sequence are coded at 30f/s with a constant quantization step size. Each group of pictures (GOP) has 10 frames coded in IPP format. Specifically, at the base layer, 5 frames correspond to $MD1$ and 6 frames to $MD2$ in each GOP. Base layer reconstruction of missing frames is done by simply copying the past frame from the other description. Each base layer packet includes a complete frame. The enhancement layer of a frame in each description is coded bit-plane by bit-plane, with each bit-plane put into one packet. The performance is measured in terms of the average luminance peak signal-to-noise ratio (PSNR) in decibels of the decoded video frames at the receiver as a function of various system parameters, such as available channel bandwidth, packet loss rate (PLR), RTT and application playback delay (denoted by w). In all experiments, the channel RTT is set to 200 ms, and each packet has transmission opportunities every 80 ms. We performed 100 rounds for each experimental scenario and the results shown are the average of these rounds.

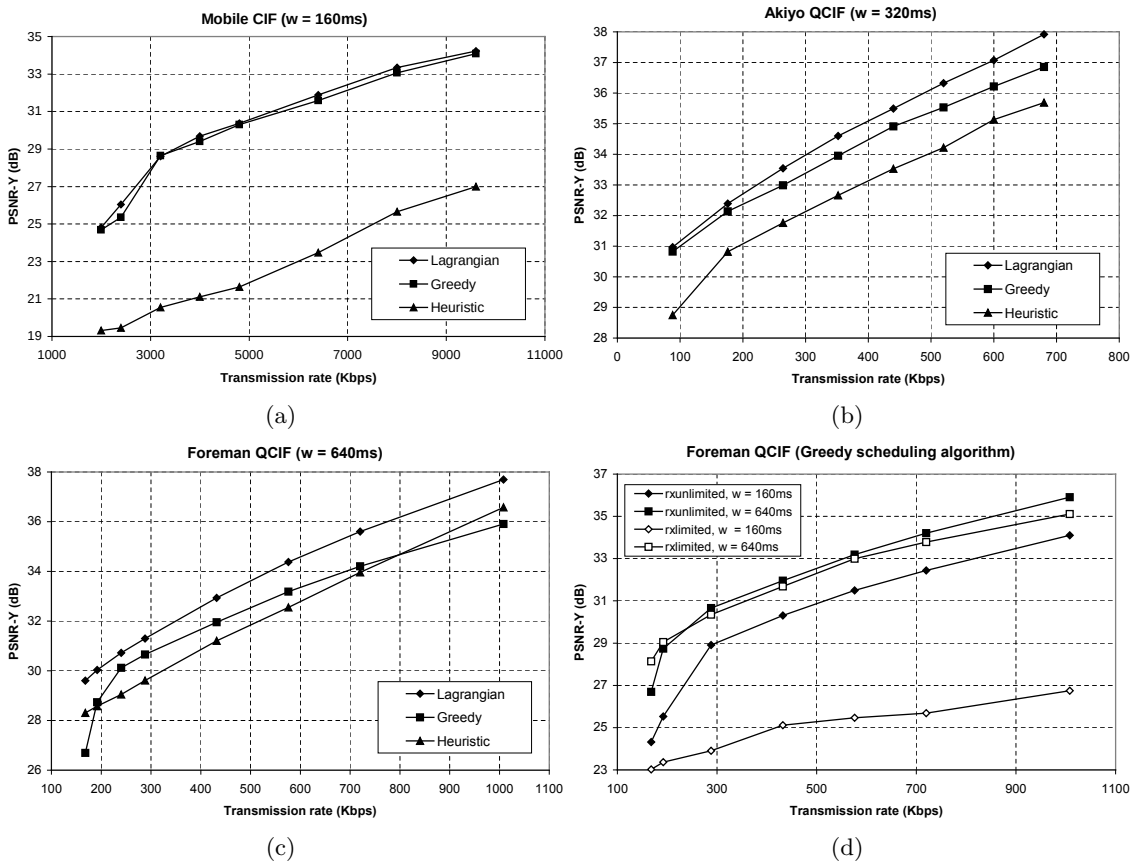


Figure 3.6: Comparison between scheduling algorithms at PLR=0.15 for various playback delays. The base layer quantization parameters for Mobile, Akiyo, and Foreman are set to 12, 20, and 20, respectively.

3.5.1 Comparison between Scheduling Algorithms

In addition to Lagrangian optimization and greedy algorithm, we also include in a comparison with a heuristic scheduling algorithm based on ARQ with prioritized transmission. In this algorithm, when the sender has not received the ACK of a packet after one RTT, it puts the packet back to the transmission queue for retransmission. The scheduler differentiates descriptions and layers by a predefined priority order. We choose the one that is observed in general to achieve better performance than the other orders. That is, $BL1$, $EL1$, $BL2$, $EL2$, in decreasing order of priority, which sends the $MD1$ first and

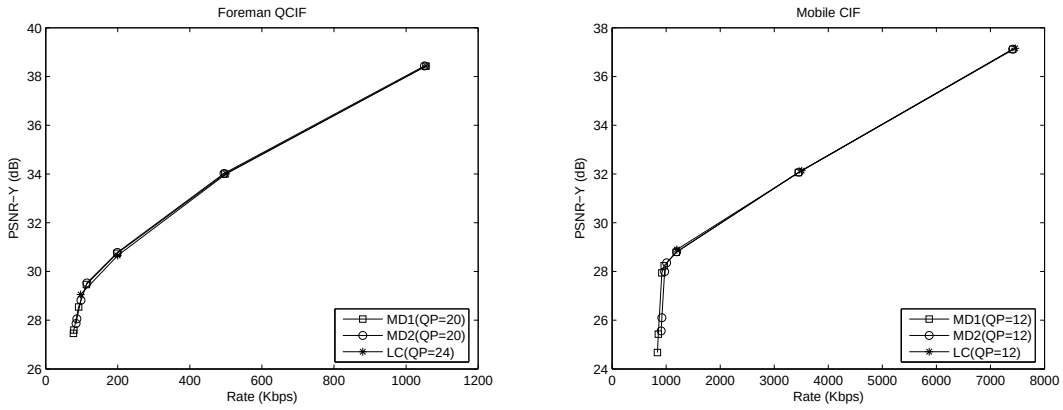
then $MD2$ for increased redundancy if additional rate is available. Among packets with the same priority class, priority is given to those with earlier playback deadlines. Both Lagrangian optimization and the greedy algorithm allow retransmissions without waiting for a timeout.

Figures 3.6(a)-(c) show the performance comparison between these systems when PLR = 0.15. First, the Lagrangian method provides substantial gains over the heuristic approach for the whole range of bandwidths under consideration at various playback delays. The performance gain is in the range of 1-7 dB and decreases as the playback delay increases. The heuristic approach prioritizes different descriptions and layers in a predefined order without exploiting rate-distortion information of source packets. Thus, the predefined order may lead to a mismatch between the added redundancy and that required by the system conditions. For example, it does not introduce enough redundancy in the case of short delay as shown in Figure 3.6(a), in that $MD2$ is not transmitted until all base and enhancement layers of $MD1$ have been sent. Furthermore, the number of retransmissions is restricted to be low due to the delay requirement. Therefore, the transmission of less significant enhancement layers of $MD1$ is more likely a waste of bandwidth due to the loss of its more significant layers. Second, the Lagrangian method outperforms the greedy algorithm by up to 3 dB, and both algorithms achieve similar performance in the short playback delay case of Figure 3.6(a). The greedy algorithm tends to introduce more redundancy in the system since it makes scheduling decisions without considering possible future packet transmissions. In some sense, the greedy algorithm is not able to exploit a longer playback delay in a cost-efficient way. Finally, the greedy algorithm performs better than the heuristic approach in most cases, since it exploits

the knowledge of distortion impact of a packet loss on the reconstructed video quality. However, in the case of long playback delay, the greedy algorithm performs poorly in some transmission rates for the reasons we just explained.

3.5.2 Redundancy's Role in Adaptive Streaming

We now describe a detailed performance analysis when the two types of redundancy, namely source and transport redundancy, are used in the streaming system. We use the Lagrangian optimization algorithm as a default scheduling algorithm unless otherwise explicitly mentioned. For all the experiments, we use LC as a representative single-decoding-path (SDP) codec, and MDLC as an example of multiple-decoding-path (MDP) codecs. In order to emphasize the differences in the end-to-end reconstructed quality due to the transmission impacts and the adaptation flexibility introduced by source redundancy, we adjust the base layer quantization parameters (QP) so MDLC and LC perform similarly in terms of coding efficiency. Figure 3.7 shows the base layer QP and rate-distortion curves of LC, $MD1$ and $MD2$ of MDLC for Mobile and Foreman measured at the encoder without any transmission impact. Three scenarios are considered in the experiments. In the first one, we compare the performance with or without transport redundancy for each type of codec. In the second scenario, when transport redundancy is not available, we compare the performance of SDP and MDP codecs, i.e., LC and MDLC. Finally, the third scenario under consideration represents a combination of the above two kinds of redundancy. Specifically, in this scenario, we examine streaming performance when both source redundancy and transport redundancy are introduced in the streaming system.



(a) (b)

Figure 3.7: Rate-distortion curves of LC, $MD1$ and $MD2$ of MDLC for Foreman and Mobile measured at the encoder without transmission impacts.

3.5.2.1 Transport Redundancy

We first examine in Figure 3.8 the performance of streaming Foreman, as a function of the available transmission rate and playback delay when LC and MDLC are used, respectively. Here, as discussed in Section 3.4.5, unlimited retransmission corresponds to the scheduling algorithms that allow retransmissions without waiting, while limited retransmission indicates the case where retransmissions have to wait till a timeout period. First, it can be seen that, for both source codecs, unlimited retransmission outperforms limited retransmission with a significant margin over the entire range of transmission rate. This is due to the fact that for the former approach the chance of multiple retransmissions is greatly increased without incurring an unacceptable delay, and therefore the additional bandwidth can be efficiently used to retransmit the most important packets so as to improve the end quality at the receiver. Since the set of possible choices for $\boldsymbol{\pi}$ in (3.14) for limited retransmission is a subset of the corresponding set of unlimited retransmission, the Lagrangian optimization should ideally always achieve better performance

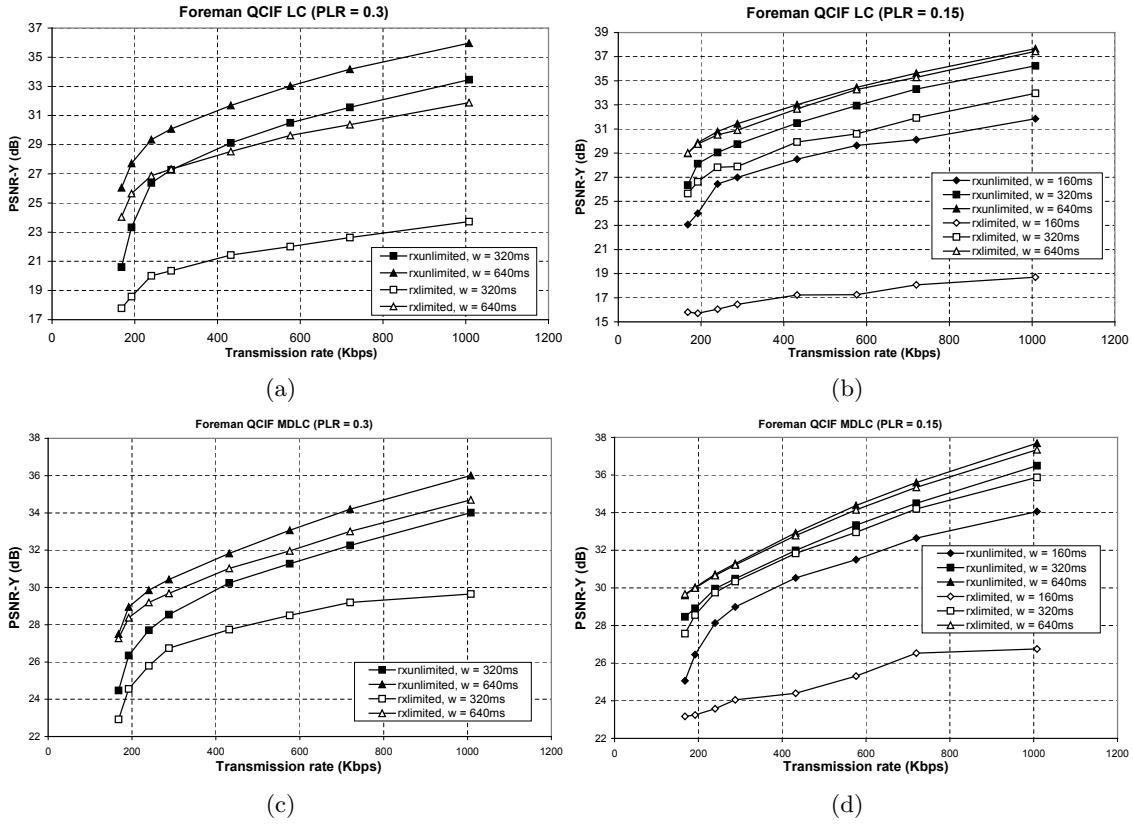


Figure 3.8: The impact of transport redundancy on streaming performance when using Lagrangian optimization algorithm.

when removing the retransmission restriction. Transport redundancy is well adjusted by Lagrangian optimization such that retransmissions are not wasted by exploiting the statistical knowledge of the channel and past transmission history. For example, it is observed that, if the playback delay is long enough, the scheduler chooses to wait for one RTT before initiating a new retransmission, so that the retransmission only occurs if the sender does not receive the ACK. However, for an algorithm that does not take into account future packet transmissions, transport redundancy introduced by unlimited retransmissions may deteriorate the algorithm performance as shown in Figure 3.6(d) with bandwidth below 200 Kbps when the greedy algorithm is used. Second, the performance gain of LC through transport redundancy tends to be more significant than that of MDLC in the same system setting. As seen in Figures 3.8 (a) and (c) at $w = 320$ ms, the gain reaches up to 9 dB for LC and 4 dB for MDLC at high transmission rates. This is because source redundancy in MDLC provides a benefit similar to that of transport redundancy in terms of improving error robustness in a lossy packet network. Finally we observe that the performance difference between unlimited and limited retransmission is larger under poor channel conditions, such as high PLR and short playback delay. Thus limited retransmission may be appropriate as a lower complexity scheduling technique in the case of low PLR and long playback delay.

3.5.2.2 Source Redundancy without Transport Redundancy

We then compare in Figure 3.9 the performance of LC and MDLC in the absence of transport redundancy, i.e., when Lagrangian algorithm is used with limited retransmissions. MDLC provides a significant gain over LC in the case of short playback delay and high

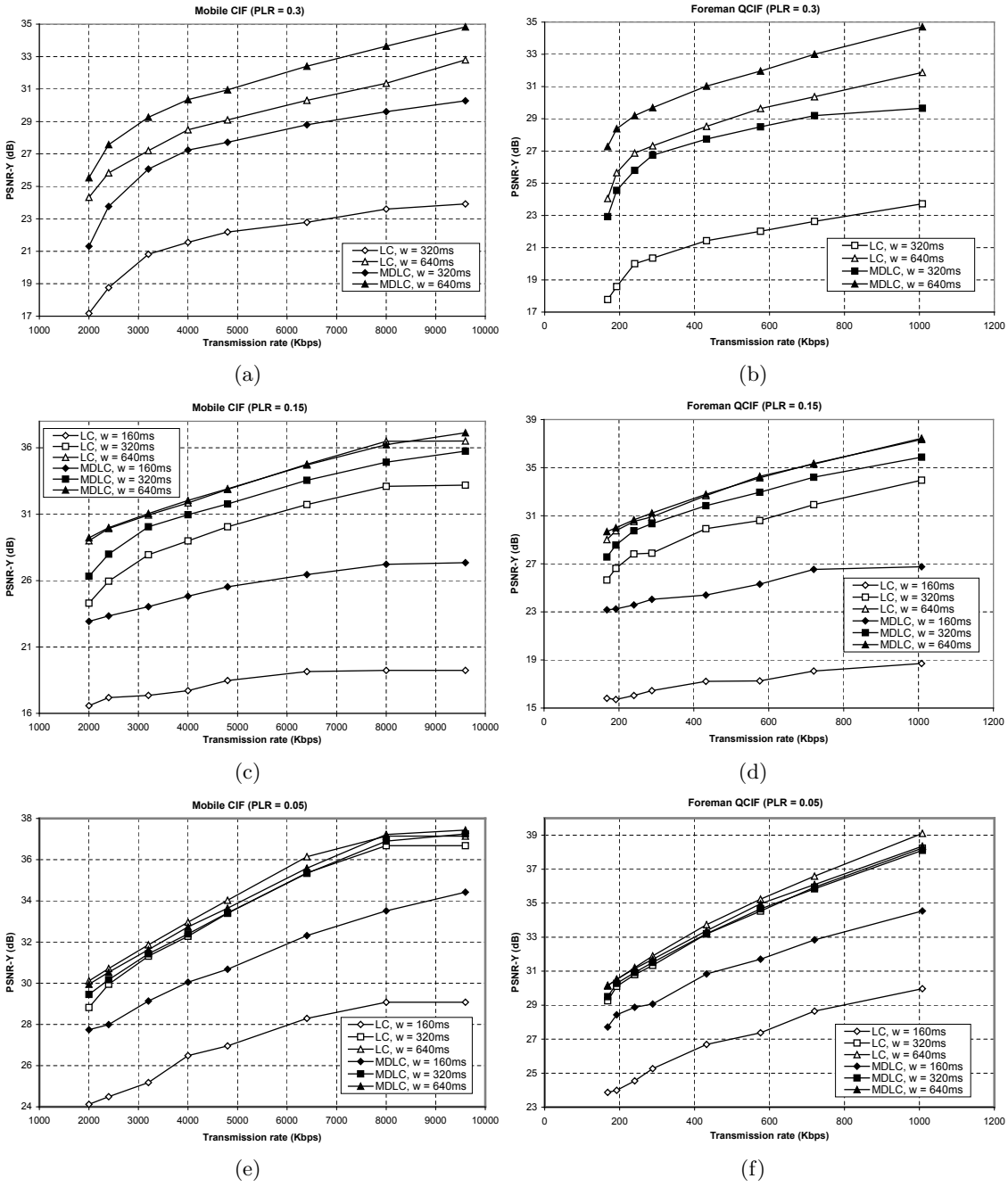
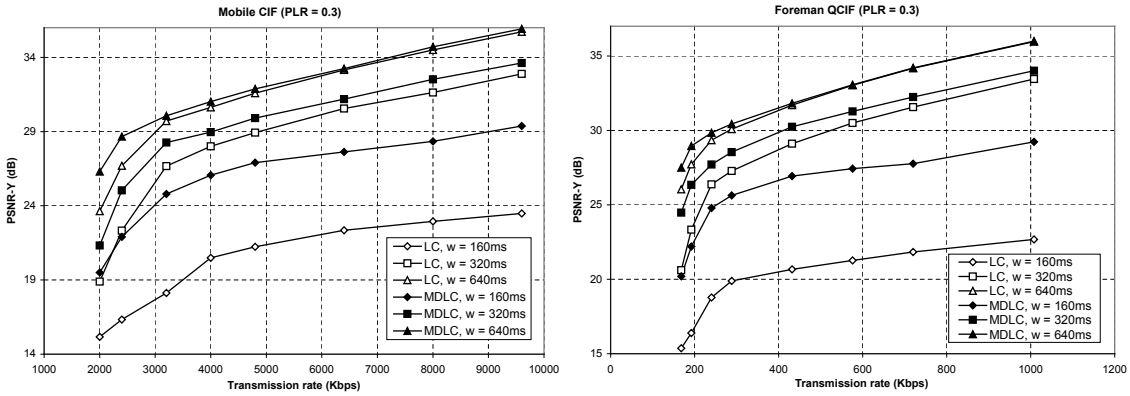


Figure 3.9: Comparing LC and MDLC with limited retransmissions. The performance at $w = 160$ ms and PLR $= 0.3$ for both sequences is not included in the figure as the low PSNR achieved is out of acceptable range.

PLR, where source redundancy introduced by multiple descriptions greatly improves error robustness. The performance gain achieves up to 8 dB for both Mobile and Foreman when $w = 160$ ms at PLR = 0.15. As w increases, the performance of LC improves as the number of possible retransmissions increases, and finally is close to that of MDLC at $w = 640$ ms. However, at very high PLR (e.g., PLR = 0.3), MDLC outperforms LC again with a gain of 1-3 dB when $w = 640$ ms. In very few cases, there is a penalty of up to 0.6 dB for MDLC over LC. This maybe be due to the possible local minimum involved in the Lagrangian optimization of MDLC. To summarize the results, source redundancy provides significant benefits on robust video communication especially in the case of high PLR and short playback delay, which is known to be a very difficult environment for video communication. When system conditions become favorable, source redundancy may not be necessary considering the additional complexity it introduces.

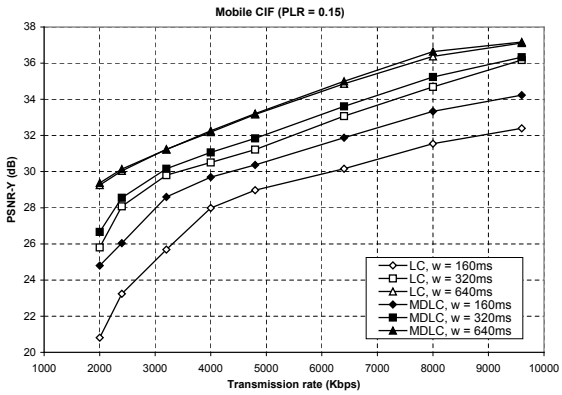
3.5.2.3 Source Redundancy with Transport Redundancy

The final comparison is between LC and MDLC when transport redundancy is applied, i.e., when using the Lagrangian optimization algorithm with unlimited retransmissions. Figure 3.10 shows the performance of streaming Mobile and Foreman under the same system settings as Figure 3.9. Note that in this case the performance difference between MDLC and LC is not as large as in the previous case. This is expected as it is no longer necessary to wait for a timeout and thus packet retransmission becomes possible even in a delay-sensitive application, where end-to-end delay is of the order of the RTT. The scheduling algorithm essentially provides unequal levels of transport redundancy to different packets based on their estimated importance, and thus overcomes the sensitivity

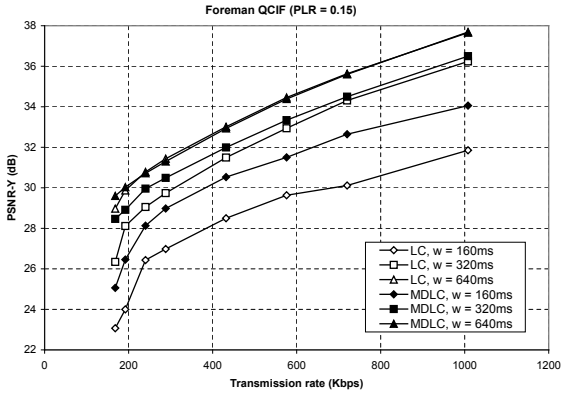


(a)

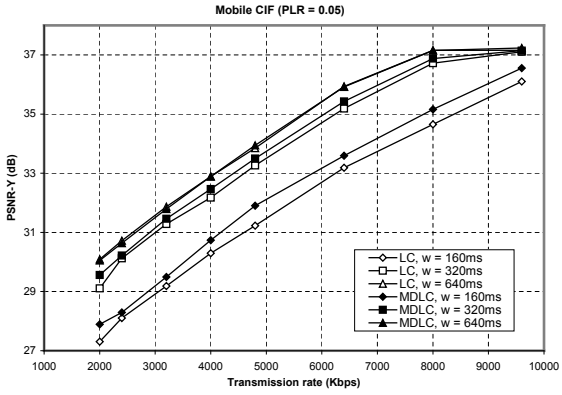
(b)



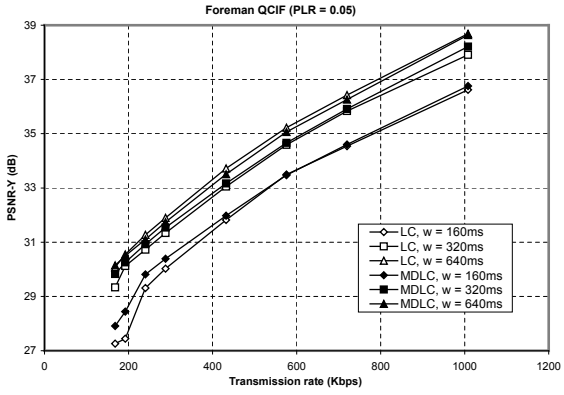
(c)



(d)



(e)



(f)

Figure 3.10: Comparing LC and MDLC with unlimited retransmissions.

of a LC system to transmission losses. But we can still observe a performance gain of up to 6 dB at $w = 160$ ms and PLR $= 0.3$. Similar to the second experiment, the performance gain varies as a function of channel conditions, and is larger for high PLR and low playback delay. Thus we can conclude that transport redundancy and source redundancy can both improve the end-to-end performance by enhancing the error robustness. Furthermore, the best performance is achieved when both types of redundancies are applied in the streaming system. Such a system can achieve efficient video streaming even under very poor channel conditions, such as for very high PLR or relatively long RTT compared to the playback delay.

3.6 Conclusions

In this chapter we have extended recent work on rate-distortion based video scheduling to the general case where multiple decoding paths are possible. We proposed a new source model called Directed Acyclic Hypergraph (DAHG) to describe the decoding dependence and redundancy between different data units. Based on this model, we have proposed two rate-distortion based scheduling algorithms, i.e., the Lagrangian optimization and greedy algorithm. Experimental results demonstrate the performance improvement by exploiting coding relation and rate-distortion information of data units in the scheduling algorithms. The results show that our proposed system with both source and transport redundancy can provide very robust and efficient real-time video communication over lossy packet networks.

Chapter 4

A Framework for Adaptive Scalable Video Coding Using Wyner-Ziv Techniques

4.1 Introduction

Scalable coding is well-suited for video streaming and broadcast applications as it facilitates adapting to variations in network behavior, channel error characteristics and computation power availability at the receiving terminal. Predictive coding, in which motion compensated predictors are generated based on previously reconstructed frames, is an important technique to remove temporal redundancy among successive frames. It is well known that predictive techniques increase the difficulty of achieving efficient scalable coding because scalability leads to multiple possible reconstructions of each frame [66]. In this situation either (i) the same predictor is used for all layers, which leads to either drift or coding inefficiency, or (ii) a different predictor is obtained for each reconstructed version and used for the corresponding layer of the current frame, which leads to added complexity. MPEG-2 SNR scalability with a single motion-compensated prediction loop and MPEG-4 FGS exemplify the first approach. MPEG-2 SNR scalability

uses the enhancement-layer (EL) information in the prediction loop for both base and enhancement layers, which leads to drift if the EL is not received. MPEG-4 FGS provides flexibility in bandwidth adaptation and error recovery because the enhancement layers are coded in "intra" mode, which results in low coding efficiency especially for sequences that exhibit high temporal correlation.

Rose and Regunathan [66] proposed a multiple motion-compensated prediction loop approach for general SNR scalability, in which each EL predictor is optimally estimated by considering all the available information from both base and enhancement layers. Several alternative multi-layer techniques have also been proposed to exploit the temporal correlation in the EL inside the FGS framework [33, 79, 93]. They employ one or more additional motion-compensated prediction loops to code the EL, for which a certain number of FGS bit-planes are included in the EL prediction loop to improve the coding efficiency. Traditional closed-loop prediction (CLP) techniques have the disadvantage of requiring the encoder to generate all possible decoded versions for each frame, so that each of them can be used to generate a prediction residue. Thus, the complexity is high at the encoder, especially for multi-layer coding scenarios. In addition, in order to avoid drift, the exact same predictor has to be used at both the encoder and decoder.

Distributed source coding techniques based on network information theory provide a different and interesting viewpoint to tackle these problems. Several video codecs using side information (SI) at the decoder [3, 26, 60, 61, 67, 69] have been recently proposed within the Wyner-Ziv framework [94]. These can be thought of as an intermediate step between "closing the prediction loop" and coding each frame independently. In closed-loop prediction in order for the encoder to generate a residue it needs to generate the

same predictor that will be available at the decoder. Instead, a Wyner-Ziv encoder only requires the *correlation structure* between the current signal and the predictor. Thus there is no need to generate the decoded signal at the encoder as long as the correlation structure is known, or can be found.

Some recent work [68, 74, 77, 96] has addressed the problem of scalable coding in the distributed source coding setting. Steinberg and Merhav [74] formulated the theoretical problem of successive refinement of information in the Wyner-Ziv setting, which serves as the theoretical background of our work. In our work, we target the application of these principles to actual video coding systems. The two most related recent algorithms are in the works by Xu and Xiong [96] and Seghal *et al.* [68]. There are a number of important differences between our approach and those techniques. In [96], the authors presented a scheme similar to MPEG-4 FGS by building the bit-plane ELs using Wyner-Ziv coding (WZC) with the current base and more significant ELs as SI, ignoring the EL information of the previous frames. In contrast, our approach explores the remaining temporal correlation between the successive frames in the EL using WZC to achieve improved performance over MPEG-4 FGS. In [68], multiple redundant Wyner-Ziv encodings are generated for each frame at different fidelities. An appropriate encoded version is selected for streaming, based on the encoder's knowledge of the predictor available at the decoder. This scheme requires a feedback channel and additional delay and thus it is not well-suited for broadcast or low-delay applications. In short, one method [96] ignores temporal redundancy in the design, while the other [68] creates separate and redundant enhancement layers, rather than a single embedded enhancement layer. In addition to these approaches for SNR scalability, Tagliasacchi *et al.* [77] have proposed a spatial

and temporal scalable codec using distributed source coding. They use the standards-conformant H.264/AVC to encode the base layer, and a syndrome-based approach similar to [61] to encode the spatial and temporal enhancement layers. Motion vectors from the base layer are used as coarse motion information so that the enhancement layers can obtain a better estimate of the temporal correlation. In contrast, our work focuses on SNR scalability.

We propose, extending our previous work [81, 84], an efficient solution to the problem of scalable predictive coding by recasting it as a Wyner-Ziv problem. Our proposed technique achieves scalability without feedback and exploits both spatial and temporal redundancy in the video signal. In [84] we introduced the basic concept on a first-order DPCM source model, and then presented a preliminary version of our approach in video applications in [81]. Our approach, Wyner-Ziv scalable coding (WZS), aims at applying in the context of Wyner-Ziv the CLP-based estimation-theoretic (ET) technique in [66]. Thus, in order to reduce the complexity, we do not explicitly construct multiple motion-compensation loops at the encoder, while, at the decoder, SI is constructed to combine spatial and temporal information in a manner that seeks to approximate the principles proposed in [66]. In particular, starting from a standard CLP base-layer (BL) video coder (such as MPEG-4 in our implementation), we create a multi-layer Wyner-Ziv prediction "link", connecting the same bit-plane level between successive frames. The decoder generates the enhancement-layer SI with either the estimation theoretic approach proposed in [66] or our proposed simplified switching algorithm to take into account all the available information to the EL. In order to design channel codes with appropriate rates,

the encoder estimates the correlation between the current frame and its enhancement-layer SI available at the decoder. By exploiting the EL information from the previous frames, our approach can achieve significant gains in EL compression, as compared to MPEG-4 FGS, while keeping complexity reasonably low at the encoder.

A significant contribution of our work is to develop a framework for integrating WZC into a standard video codec to achieve efficient and low-complexity scalable coding. Our proposed framework is backward compatible with a standard base-layer video codec. Another main contribution of this work is to propose two simple and efficient algorithms to explicitly estimate at the encoder the parameters of a model to describe the correlation between the current frame and an optimized SI available only at the decoder. Our estimates closely match the actual correlation between the source and the decoder SI. The first algorithm is based on constructing an estimate of the reconstructed frame and directly measuring the required correlations from it. The second algorithm is based on an analytical model of the correlation structure, whose parameters the encoder can estimate.

The chapter is organized as follows. In Section 4.2, we briefly review the theoretical background of successive refinement for the Wyner-Ziv problem. We then describe our proposed practical WZS framework and the correlation estimation algorithms in Sections 4.3 and 4.4, respectively. Section 4.5 describes the codec structure and implementation details. Simulation results are presented in Section 4.6, showing substantial improvement in video quality for sequences with high temporal correlation. Finally, conclusions and future work are provided in Section 4.7.

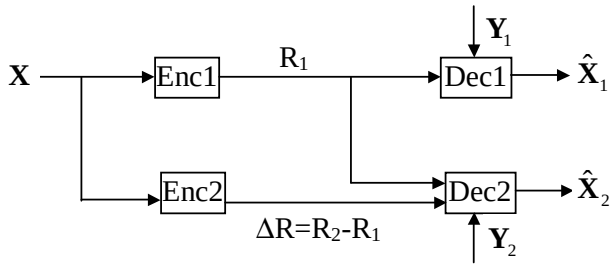


Figure 4.1: Two-stage successive refinement with different side information Y_1 and Y_2 at the decoders, where Y_2 has better quality than Y_1, i.e. $X \to Y_2 \to Y_1$.

4.2 Successive Refinement for the Wyner-Ziv Problem

Steinberg and Merhav [74] formulated the theoretical problem of successive refinement of information, originally proposed by Equitz and Cover [24], in a Wyner-Ziv setting (see Fig. 4.1). A source X is to be encoded in two stages: at the coarse stage, using rate R_1, the decoder produces an approximation, $\hat{X}_1$ with distortion D_1 based on SI Y_1. At the refinement stage, the encoder sends an additional ΔR refinement bits so that the decoder can produce a more accurate reconstruction, $\hat{X}_2$, with a lower distortion D_2 based on SI Y_2. Y_2 is assumed to provide a better approximation to X than Y_1 and to form a Markov chain $X \to Y_2 \to Y_1$. Let $R^*_{X|Y}(D)$ be the Wyner-Ziv rate-distortion function for coding X with SI Y. A source X is successively refinable if [74]:

$$R_1 = R^*_{X|Y_1}(D_1), \text{ and } R_1 + \Delta R = R^*_{X|Y_2}(D_2). \tag{4.1}$$

Successive refinement is possible under a certain set of conditions. One of the conditions, as proved in [74], requires that the two SIs, Y_1 and Y_2, be equivalent at the distortion level D_1 in the coarse stage. To illustrate the concept of "equivalence", we first consider the classical Wyner-Ziv problem (i.e., without successive refinement) as follows. Let Y

be the SI available at the decoder only, for which a joint distribution with source X is known by the encoder. Wyner and Ziv [94] have shown that

$$R^*_{X|Y} = \min_U [I(X;U|Y)] \tag{4.2}$$

where U is an auxiliary random variable, and the minimization of mutual information between X and U given Y is over all possible U such that $U \rightarrow X \rightarrow Y$ forms a Markov chain and $E[d(X, f(U,Y))] \leq D$. For the successive refinement problem, Y_2 is said to be equivalent to Y_1 at D_1, if there exists a random variable U achieving (4.2) at D_1 and satisfying $I(U;Y_2|Y_1) = 0$ as well. In words, when Y_1 is given, Y_2 does not provide any more information about U.

It is important to note that this equivalence is unlikely to arise in scalable video coding. As an example, assume that Y_1 and Y_2 correspond to the BL and EL reconstruction of the previous frame, respectively. Then, the residual energy when the current frame is predicted based on Y_2 will in general be lower than if Y_1 is used. Thus, in general, this equivalence condition will not be met in the problem we consider and we should expect to observe a performance penalty with respect to a non-scalable system. Note that one special case where equivalence holds is that where identical SIs are used at all layers, i.e., $Y_1 = Y_2$. For this case and for a Gaussian source with quadratic distortion measure the successive refinement property holds [74]. Some practical coding techniques have been developed based on this equal SI property, e.g., in the work of Xu and Xiong [96], where the BL of the current frame is regarded as the only SI at the decoder at both the coarse and refinement stages. However, as will be shown, constraining the decoder to use the

same SI at all layers leads to suboptimal performance. In our work, the decoder will use the EL reconstruction of the previous frame as SI, outperforming an approach similar to that proposed in [96].

4.3 Proposed Prediction Framework

In this section, we propose a practical framework to achieve Wyner-Ziv scalability for video coding. Let video be encoded so that each frame i is represented by a base layer BL_i, and multiple enhancement layers EL_{i1}, EL_{i2}, ..., EL_{iL}, as shown in Fig. 4.2. We assume that in order to decode EL_{ij} and achieve the quality provided by the j-th EL, the decoder will need to have access to: (1) the previous frame decoded up to the j-th EL, $EL_{i-1,k}$, $k \leq j$, and (2) all information for the higher significance layers of the current frame, EL_{ik}, $k < j$, including reconstruction, prediction mode, BL motion vector for each Inter-mode macroblock, and the compressed residual. For simplicity, the BL motion vectors are reused by all EL bit-planes.

With the structure shown in Fig. 4.2, a scalable coder based on WZC techniques would need to combine multiple SIs at the decoder. More specifically, when decoding the information corresponding to $EL_{i,k}$, the decoder can use as SI decoded data corresponding to $EL_{i-1,k}$ and $EL_{i,k-1}$. In order to understand how several different SIs can be used together we first review a well-known technique for combining multiple predictors in the context of closed-loop coding (Section 4.3.1 below). We then introduce an approach to formulate our problem as one of source coding with side information at the decoder (Section 4.3.2).

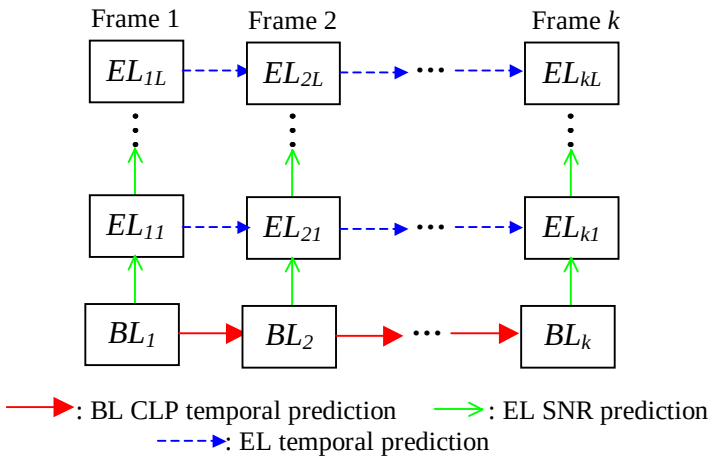


Figure 4.2: Proposed multi-layer prediction problem. BL_i: the base layer of the ith frame. EL_{ij}: the jth EL of the ith frame, where the most significant EL bit-plane is denoted by $j = 1$.

4.3.1 Brief Review of ET Approach

In this section we briefly review the ET approach proposed in [66]. The temporal evolution of DCT coefficients can be usually modelled by a first-order Markov process

$$x_k = \rho x_{k-1} + z_k, \; x_{k-1} \perp z_k \tag{4.3}$$

where x_k is a DCT coefficient in the current frame and x_{k-1} is the corresponding DCT coefficient in the previous frame after motion compensation. Let $\hat{x}_k^b$ and $\hat{x}_k^e$ be the base and enhancement layer reconstruction of x_k, respectively. After the BL is generated we know that $x_k \in (a, b)$, where (a, b) is the quantization interval generated by the BL. In

addition, assume that the EL encoder and decoder have access to the EL reconstructed DCT coefficient $\hat{x}^e_{k-1}$ of the previous frame. Then the optimal EL predictor is given by

$$\begin{aligned} \tilde{x}^e_k &= E[x_k|\hat{x}^e_{k-1},\, x_k \in (a,b)] \\ &\approx \rho\hat{x}^e_{k-1} + E[z_k|z_k \in (a-\rho\hat{x}^e_{k-1}, b-\rho\hat{x}^e_{k-1})]. \end{aligned} \tag{4.4}$$

The EL encoder then quantizes the residual

$$r^e_k = x_k - \tilde{x}^e_k. \tag{4.5}$$

Let (c,d) be the quantization interval associated with r^e_k, i.e., $r^e_k \in (c,d)$, and let $e = \max(a, c+\tilde{x}^e_k)$ and $f = \min(b, d+\tilde{x}^e_k)$. The optimal EL reconstruction is given by

$$\hat{x}^e_k = E[x_k|\hat{x}^e_{k-1},\, x_k \in (e,f)]. \tag{4.6}$$

The EL predictor in (4.4) can be simplified in the following two cases: (1) $\tilde{x}^e_k \approx \hat{x}^b_k$ if correlation is low, $\rho \approx 0$, or the total rate is approximately the same as the BL rate, i.e., $\hat{x}^e_{k-1} \approx \hat{x}^b_{k-1}$; (2) $\tilde{x}^e_k \approx \hat{x}^e_{k-1}$ for cases where temporal correlation is higher or such that the quality of the BL is much lower than that of EL.

Note that in addition to optimal prediction and reconstruction, the ET method can lead to further performance gains if efficient context-based entropy coding strategies are used. For example, the two cases $\tilde{x}^e_k \approx \hat{x}^b_k$ and $\tilde{x}^e_k \approx \hat{x}^e_{k-1}$ could have different statistical properties. In general, with the predictor of (4.4), since the statistics of z_k tend to be different depending on the interval $(a-\rho\hat{x}^e_{k-1}, b-\rho\hat{x}^e_{k-1})$, the encoder could use different

entropy coding on different intervals [66]. Thus, a major goal in this chapter is to design a system that can achieve some of the potential coding gains of conditional coding *in the context of a WZC technique*. To do so we will design a switching rule at the encoder that will lead to different coding for different types of source blocks.

4.3.2 Formulation as a Distributed Source Coding Problem

The main disadvantage of the ET approach for multi-layer coding resides in its complexity, since multiple motion-compensated prediction loops are necessary for EL predictive coding. For example, in order to encode EL_{21} in Fig. 4.2, the exact reproduction of EL_{11} must be available at the encoder. If the encoder complexity is limited, it may not be practical to generate all possible reconstructions of the reference frame at the encoder. In particular, in our work we assume that the encoder can generate *only* the reconstructed BL, and does not generate any EL reconstruction, i.e., none of the EL_{ij} in Fig. 4.2 are available at the encoder. Under this constraint we seek efficient ways to exploit the temporal correlation between ELs of consecutive frames. In this chapter, we propose to cast the EL prediction as a Wyner-Ziv problem, using Wyner-Ziv coding to replace the closed loop between the respective ELs of neighboring frames.

We first focus on the case of two-layer coders, which can be easily extended to multi-layer coding scenarios. The basic difference at the encoder between CLP techniques, such as ET, and our problem formulation is illustrated in Fig. 4.3. A CLP technique would compute an EL predictor

$$\tilde{x}_k^e = f(\hat{x}_{k-1}^e, \hat{x}_k^b) \tag{4.7}$$

where $f(\cdot)$ is a general prediction function (in the ET case $f(\cdot)$ would be defined as in (4.4)). Then, the EL encoder would quantize the residual r_k^e in (4.5) and send it to the decoder.

Instead, in our formulation, we assume that the encoder can only access $\hat{x}_k^b$, while the decoder has access to both $\hat{x}_k^b$ and $\hat{x}_{k-1}^e$. Therefore, the encoder cannot generate the same predictor $\tilde{x}_k^e$ as (4.7) and cannot explicitly generate r_k^e. Note, however, that $\hat{x}_k^b$, one of the components in (4.7), is in fact available at the encoder, and would exhibit some correlation with x_k. This suggests making use of $\hat{x}_k^b$ at the encoder. First, we can rewrite r_k^e as

$$r_k^e = x_k - \tilde{x}_k^e = (x_k - \hat{x}_k^b) - (\tilde{x}_k^e - \hat{x}_k^b) \tag{4.8}$$

and then to make explicit how this can be cast as a Wyner-Ziv coding problem, let $u_k = x_k - \hat{x}_k^b$ and $v_k = \tilde{x}_k^e - \hat{x}_k^b$. With this notation u_k plays the role of the input signal and v_k plays the role of SI available at the decoder only. We can view v_k as the output of a hypothetical communication channel with input u_k corrupted by correlation noise. Therefore, once the correlation between u_k and v_k has been estimated, the encoder can select an appropriate channel code and send the relevant coset information such that the decoder can obtain the correct u_k with SI v_k. Section 4.4 will present techniques to efficiently estimate the correlation parameters at the encoder.

In order to provide a representation with multiple layers coding, we generate the residue u_k for a frame and represent this information as a series of bit-planes. Each bit-plane contains the bits at a given significance level obtained from the absolute values of all DCT coefficients in the residue frame (the difference between the base layer reconstruction

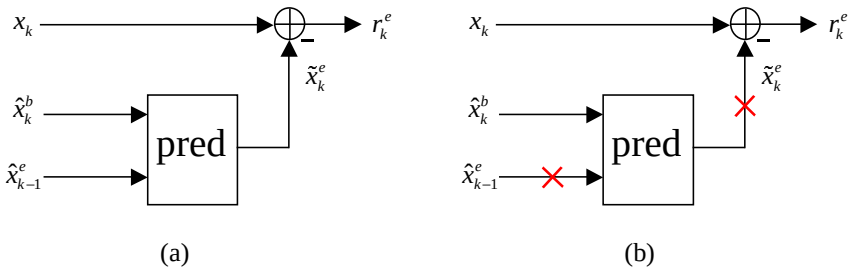


Figure 4.3: Basic difference at the encoder between the CLP techniques such as ET and our proposed problem: (a) CLP techniques, (b) our problem setting.

and the original frame). The sign bit of each DCT coefficient is coded once in the bit-plane where that coefficient becomes significant (similar to what is done in standard bit-plane based wavelet image coders). Note that this would be the same information transmitted by an MPEG-4 FGS technique. However, differently from the intra bit-plane coding in MPEG-4 FGS, we create a multi-layer Wyner-Ziv prediction link, connecting a given bit-plane level in successive frames. In this way we can exploit the temporal correlation between corresponding bit-planes of u_k and v_k, without reconstructing v_k explicitly at the encoder.

4.4 Proposed Correlation Estimation

Wyner-Ziv techniques are often advocated because of their reduced encoding complexity. It is important to note, however, that their compression performance depends greatly on the accuracy of the correlation parameters estimated at the encoder. This correlation estimation can come at the expense of increased encoder complexity, thus potentially eliminating the complexity advantages of WZC techniques. In this section, we propose estimation techniques to achieve a good tradeoff between complexity and coding performance.

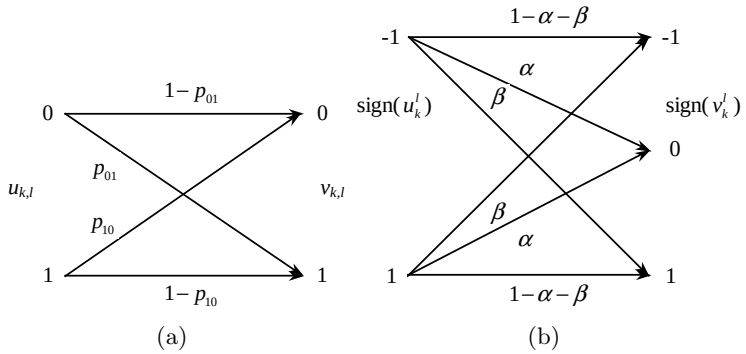


Figure 4.4: Discrete memoryless channel model for coding u_k: (a) binary channel for bit-planes corresponding to absolute values of frequency coefficients (i.e., $u_{k,l}$ at bit-plane l), (b) discrete memoryless channel with binary inputs ("-1" if $u_k^l < 0$ and "1" if $u_k^l > 0$) and three outputs ("-1" if $v_k^l < 0$, "1" if $v_k^l > 0$ and "0" if $v_k^l = 0$) for sign bits,

4.4.1 Problem Formulation

Our goal is to estimate the correlation statistics (e.g., the matrix of transition probabilities in a discrete memoryless channel) between bit-planes of same significance in u_k and v_k. To do so, we face two main difficulties. First, and most obvious, $\hat{x}_{k-1}^e$, and therefore v_k, are not generated at the encoder as shown in Fig. 4.3. Second, v_k is generated at the decoder by using the predictor $\tilde{x}_k^e$ from (4.7), which combines $\hat{x}_{k-1}^e$ and $\hat{x}_k^b$. In Section 4.4.2 we will discuss the effect of these combined predictors on the estimation problem, with a focus on our proposed mode-switching algorithm.

In what follows the most significant bit-plane is given the index "1", the next most significant bit-plane index "2", and so on. $u_{k,l}$ denotes the l-th bit-plane of absolute values of u_k, while u_k^l indicates the reconstruction of u_k (including the sign information) truncated to its l most significant bit-planes. The same notation will be used for other signals represented in terms of their bit-planes, such as v_k.

In this work, we assume the channel between source u_k and decoder SI v_k to be modeled as shown in Fig. 4.4. With a binary source $u_{k,l}$, the corresponding bit-plane of v_k, $v_{k,l}$, is assumed to be generated by passing this binary source through a binary channel. In addition to the positive (symbol "1") and negative (symbol "-1") sign outputs, an additional output symbol "0" is introduced in the sign bit channel to represent the case when SI $v_k = 0$.

We propose two different methods to estimate crossover probabilities, namely, (1) a direct estimation (Section 4.4.3), which generates estimates of the bit-planes first, then directly measures crossover probabilities for these estimated bit-planes, and (2) a model-based estimation (Section 4.4.4), where a suitable model for the residue signal $(u_k - v_k)$ is obtained and used to estimate the crossover probabilities in the bit-planes. These two methods will be evaluated in terms of their computational requirements, as well as their estimation accuracy.

4.4.2 Mode-Switching Prediction Algorithm

As discussed in Section 4.3, the decoder has access to two SIs, $\hat{x}^e_{k-1}$ and $\hat{x}^b_k$. Consider first the prediction function in (4.7) when both SIs are known. In the ET case, $f(\cdot)$ is defined as an optimal prediction as in (4.4) based on a given statistical model of z_k. Alternatively, the optimal predictor $\tilde{x}^e_k$ can be simplified to either $\hat{x}^e_{k-1}$ or $\hat{x}^b_k$ for a two-layer coder, depending on whether the temporal correlation is strong (choose $\hat{x}^e_{k-1}$) or not (choose $\hat{x}^b_k$).

Here we choose the switching approach due to its lower complexity, as compared to the optimal prediction, and also because it is amenable to an efficient use of "conditional"

entropy coding. Thus, a different channel code could be used to code u_k when $\tilde{x}_k^e \approx \hat{x}_k^b$ and when $\tilde{x}_k^e \approx \hat{x}_{k-1}^e$. In fact, if $\tilde{x}_k^e = \hat{x}_k^b$, then $v_k = 0$, and we can code u_k directly via entropy coding, rather than using channel coding. If $\tilde{x}_k^e = \hat{x}_{k-1}^e$, we apply WZC to u_k with the estimated correlation between u_k and v_k.

For a multi-layer coder, the temporal correlation usually varies from bit-plane to bit-plane, and thus the correlation should be estimated at each bit-plane level. Therefore, the switching rules we just described should be applied before each bit-plane is transmitted. We allow a different prediction mode to be selected on a macroblock (MB) by macroblock basis (allowing adaptation of the prediction mode for smaller units, such as blocks or DCT coefficients may be impractical). At bit-plane l, the source u_k has two SIs available at the decoder: u_k^{l-1} (the reconstruction from its more significant bit-planes), and $\hat{x}_{k-1}^e$ (the EL reconstruction from the previous frame). The correlation between u_k and each SI is estimated as the absolute sum of their difference. When both SIs are known, the following parameters are defined for each MB,

$$\begin{aligned} E_{intra} &= \textstyle\sum_{MB_i} |u_k - u_k^{l-1}| \\ E_{inter} &= \textstyle\sum_{MB_i} |u_k - (\hat{x}_{k-1}^e - \hat{x}_k^b)| = \sum_{MB_i} |x_k - \hat{x}_{k-1}^e|, \end{aligned} \tag{4.9}$$

where only the luminance component is used in the computation. Thus, we can make the mode decision as follows: WZS-MB (coding of MB via WZS) mode is chosen if

$$E_{inter} < E_{intra}. \tag{4.10}$$

Otherwise, we code u_k directly via bit-plane by bit-plane refinement (FGS-MB), since it is more efficient to exploit spatial correlation through bit-plane coding.

In general mode-switching decisions can be made at either encoder or decoder. Making a mode decision at the decoder means deciding which SI should be used to decode WZC data sent by the encoder. The advantage of this approach is that all relevant SI is available. A disadvantage in this case is that the encoder has to estimate the correlation between u_k and v_k without exact knowledge of the mode decisions that will be made at the decoder. Thus, because it does not know which MBs will be decoded using each type of SI, the encoder has to encode all information under the assumption of a single "aggregate" correlation model for all blocks. This prevents the full use of conditional coding techniques discussed earlier.

Alternatively, making mode decisions at the encoder provides more flexibility as different coding techniques can be applied to each block. The main drawback of this approach is that the SI $\hat{x}^e_{k-1}$ is not available at the encoder, which makes the mode decision difficult and possibly suboptimal. In this chapter, we select to make mode decisions at the encoder, with mode switching decisions based on the estimated levels of temporal correlation. Thus E_{inter} cannot be computed exactly at the encoder as defined in (4.9), since $\hat{x}^e_{k-1}$ is unknown; this will be further discussed once specific methods to approximate E_{inter} at the encoder have been introduced.

4.4.3 Direct Estimation

For the l-th bit-plane, $1 \leq l \leq L$, where L is the least significant bit-plane level to be encoded, we need to estimate the correlation between $u_{k,l}$ and v_k given all $u_{k,j}$ $(1 \leq$

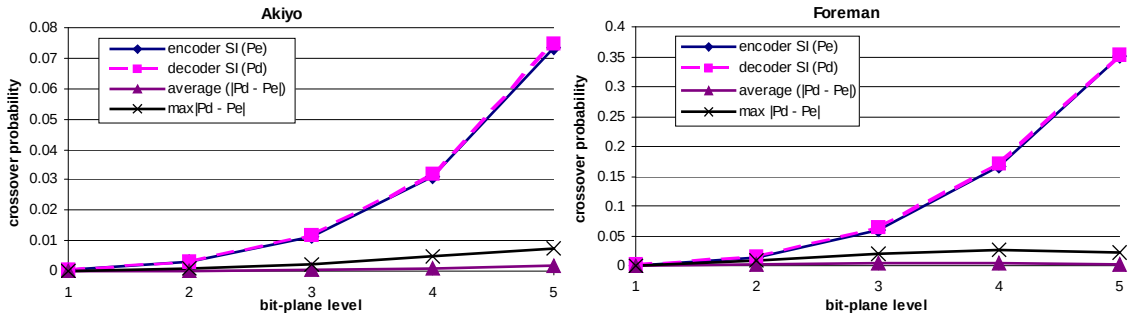
$j < l$) which have been sent to the decoder. While, in general, for decoding u_k all the information received by the decoder can be used, here, we estimate the correlation under the assumption that to decode bit-plane l, we use only the l most significant bit-planes of the previous frame. The SI for bitplane l in this particular case is denoted by $\check{v}_k(l)$, which is unknown at the encoder.

We compute $\bar{v}_k(l)$ at the encoder to approximate $\check{v}_k(l)$, $1 \leq l \leq L$. Ideally we would like the following requirements to be satisfied: (1) The statistical correlation between each bit-plane $u_{k,l}$ and $\check{v}_k(l)$, given all $u_{k,j}$ $(1 \leq j < l)$ can be well approximated by the corresponding correlation between $u_{k,l}$ and $\bar{v}_k(l)$; and (2) $\bar{v}_k(l)$ can be obtained at the encoder in a simple way without much increased computational complexity. This can be achieved by processing the original reference frame x_{k-1} at the encoder. We first calculate the residual

$$s_k = x_{k-1} - \hat{x}_k^b \tag{4.11}$$

at the encoder, and then generate bit-planes s_k^l, in the same way as the u_k^l are generated. Let $\bar{v}_k(l) = s_k^l$ for $1 \leq l \leq L$. While $\bar{v}_k(l)$ and $\check{v}_k(l)$ are not equal, the correlation between $\bar{v}_k(l)$ and $u_{k,l}$ provides a good approximation to the correlation between $\check{v}_k(l)$ and $u_{k,l}$, as is seen in Fig. 4.5, which shows the probability that $u_k^l \neq s_k^l$ (i.e. the values of u_k and s_k do not fall into the same quantization bin), as well as the corresponding crossover probability between u_k and decoder SI $\check{v}_k(l)$. The crossover probability here is an indication of the correlation level.

SI s_k^l can be used by the encoder to estimate the level of temporal correlation, which is again used to perform mode switching and determine the encoding rate of the channel



(a) (b)

Figure 4.5: Measurement of approximation accuracy for *Akiyo* and *Foreman* sequences. The crossover probability is defined as the probability that the values of the source u_k and side information do not fall into the same quantization bin. The average and maximum absolute differences over all frames between the two crossover probabilities are also shown.

codes applied to MBs in WZS-MB mode. Replacing the term $(\hat{x}^e_{k-1} - \hat{x}^b_k)$ in (4.9) by s^l_k, E_{inter} is redefined as

$$E_{inter} = \sum_{MB_i} |u_k - s^l_k|. \tag{4.12}$$

Clearly, the larger E_{intra}, the more bits will be required to refine the bit-plane in FGS-MB mode. Similarly E_{inter} gives an indication of the correlation present in the i-th MB between u^l_k and s^l_k, which are approximations of u_k and v_k at the l-th bit-plane, respectively. To code MBs in WZS-MB mode, we can further approximate the ET optimal predictor in (4.4) by taking into account both SIs, u^{l-1}_k and s^l_k, as follows: If s_k is within the quantization bin specified by u^{l-1}_k, the EL predictor is set to s^l_k; however, if s_k is outside that quantization bin, the EL predictor is constructed by first clipping s_k to the closest value within the bin and then truncating this new value to its l most significant bit-planes. For simplicity, we still denote the improved EL predictor of the lth bit-plane as s^l_k in the following discussion.

Table 4.1: Channel parameters and the *a priori* probabilities for the 3rd bit-plane of frame 3 of *Akiyo* CIF sequence when BL quantization parameter is 20 (with the same symbol notation as Fig. 4.4).

$\Pr(u_{k,l}=1)$	p_{01}	p_{10}	$\Pr(\text{sign}(u_k^l)=1)$	α	β
0.13	0.019	0.14	0.49	0.13	0.001

At bit-plane l, the rate of the channel code used to code $u_{k,l}$ (or the sign bits that correspond to that bit-plane) for MBs in WZS-MB mode is determined by the encoder based on the estimated conditional entropy $H(u_{k,l}|s_{k,l})$ (or $H(\text{sign}(u_k^l)|\text{sign}(s_k^l))$). For discrete random variables X and Y, $H(X|Y)$ can be written as

$$H(X|Y) = \sum_{y_i} \Pr(Y = y_i) H(X|Y = y_i), \tag{4.13}$$

where both $\Pr(Y = y_i)$ and $H(X|Y = y_i)$ can be easily calculated once the *a priori* probability of X and the transition probability matrix are known. The crossover probability, for example p_{01} in Fig. 4.4 (a), is derived by counting the number of coefficients such that $u_{k,l} = 0$ and $u_{k,l} \neq s_{k,l}$. Table 4.1 shows an example of those parameters for both $u_{k,l}$ and the sign bits. Note that the crossover probabilities between $u_{k,l}$ and $s_{k,l}$ are very different for source symbols 0 and 1, and therefore an asymmetric binary channel model will be needed to code $u_{k,l}$ as shown in Fig. 4.4 (a). However, the sign bit has almost the same transitional probabilities whenever the input is -1 or 1, and is thus modelled as a symmetric discrete memoryless channel in Fig. 4.4 (b).

In terms of complexity, note that there are two major steps in this estimation method: i) bit-plane extraction from s_k and ii) conditional entropy calculation (including the counting to estimate the crossover probabilities). Bit-planes need to be extracted only once per

frame and this is done with a simple shifting operation on the original frame. Conditional entropy will be calculated for each bit-plane based on the crossover probabilities estimated by simple counting. In Section 4.5 we will compare the complexity of the proposed WZS approach and the ET approach.

4.4.4 Model-based Estimation

In this section we introduce a model-based method for correlation estimation that has lower computational complexity, at the expense of a small penalty in coding efficiency. The basic idea is to estimate first the probability density functions (pdf) of the DCT residuals ($u_k, v_k, z_k = v_k - u_k$), and then use the estimated pdf to derive the crossover probabilities for each bit-plane.

Assume that u_k, v_k, z_k are independent realizations of the random variables U, V, and Z, respectively. Furthermore, assume that $V = U + Z$, with U and Z independent. We start by estimating of the pdf's $f_U(u)$ and $f_Z(z)$. This can be done by choosing appropriate models for the data samples, and estimating the model parameters using one of the standard parameter estimation techniques, e.g., maximum likelihood estimation, expectation-maximization (EM), etc. Note that since the v_k are not available in our encoder, we use s_k to approximate v_k in the model parameter estimation.

Once we have estimated $f_U(u)$ and $f_Z(z)$ we can derive the crossover probabilities at each bit-plane as follows. Recall that we consider there is no crossover when u_k, v_k fall into the same quantization bin. This corresponds to the event denoted by the shaded

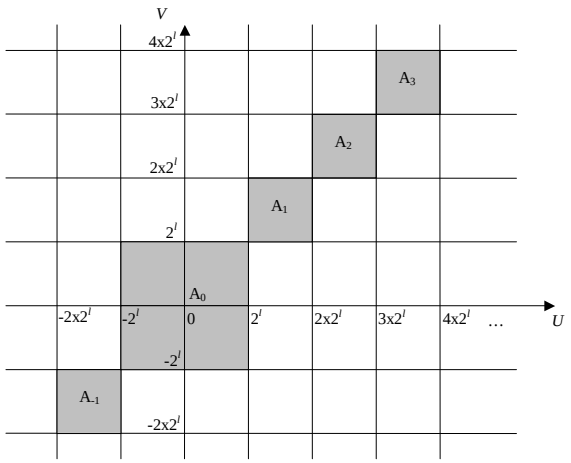


Figure 4.6: Crossover probability estimation. The shaded square regions A_i correspond to the event that crossover does not occur at bit-plane l.

square regions in Fig. 4.6. Hence we can find the estimate of the crossover probability at bit-plane l (denoted as $\hat{p}(l)$) by

$$\hat{p}(l) = 1 - I(l), \tag{4.14}$$

where $I(l)$ is given by

$$I(l) = \sum_i \int\int_{A_i} f_{UV}(u,v)dudv = \sum_i \int\int_{A_i} f_U(u) f_{V|U}(v|u)dudv. \tag{4.15}$$

$I(l)$ is simply the probability that U, V fall into the the same quantization bin. The conditional pdf $f_{V|U}(v|u)$ can be obtained as

$$f_{V|U}(v|u) = f_Z(v-u) \tag{4.16}$$

and the integral in (4.15) can be readily evaluated for a variety of densities. In practice we only need to sum over a few regions, A_i, where the integrals are non-zero.

We found that U and Z can be well-modeled by mixtures of two zero-mean Laplacians with different variances. We use the EM algorithm to obtain the maximum-likelihood estimation of the model parameters, and use (4.15) and (4.16) to compute the estimates of the crossover probabilities.

The main advantage of this model-based estimation approach as compared with the direct estimation is that it incurs less complexity and requires less frame data to be measured. In our experiment the EM was operating on only 25 % of the frame samples. Moreover, since the model parameters do not vary very much between consecutive frames (Fig. 4.7) it is viable to use the previous estimates to initialize the current estimation and this can usually lead to convergence within a few iterations. Once we have found the model parameters, computing the crossover probability of each bit-plane from the model parameters requires only negligible complexity since this can be done using closed-form expressions obtained from the integrals in (4.15). However, the approach suffers some loss in compression efficiency due to the inaccuracy in the estimation. We can assess the compression efficiency by evaluating the entropy function on the estimates of the crossover probabilities (which gives the theoretical limit in compressing the bit-planes given the estimates [46]), and compare to that of the direct estimation. Experiments using video frames from the *Akiyo* sequence show that with base layer quantization parameter (QP) set to 31 and 20, the percentage differences in entropy are about 2.5% and 4.7%, respectively. However, the percentage difference is 21.3% when the base layer QP is set to 8. This large deviation is due to the fact that with QP equal to 8, the base layer is of very high quality, so that the distribution of U has a higher probability of zero, which is not

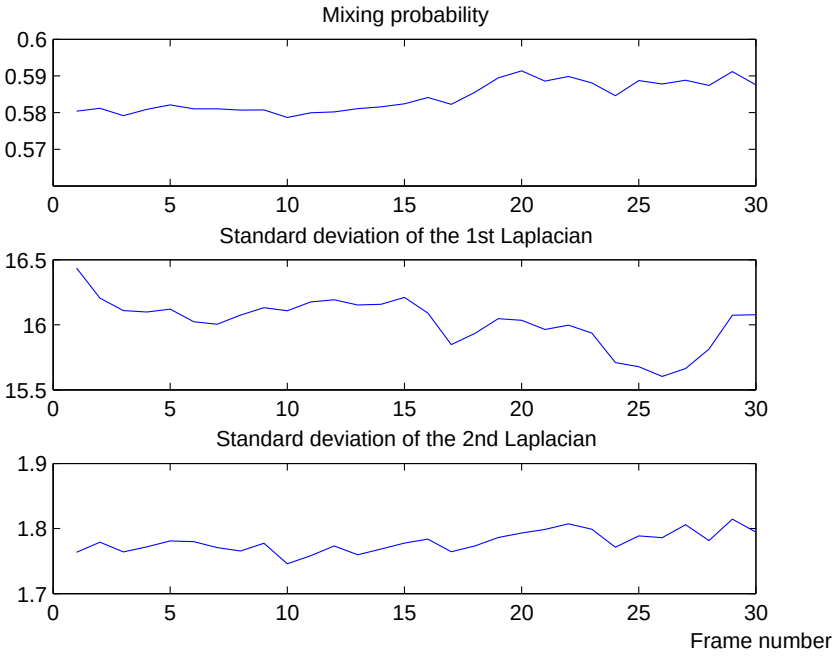
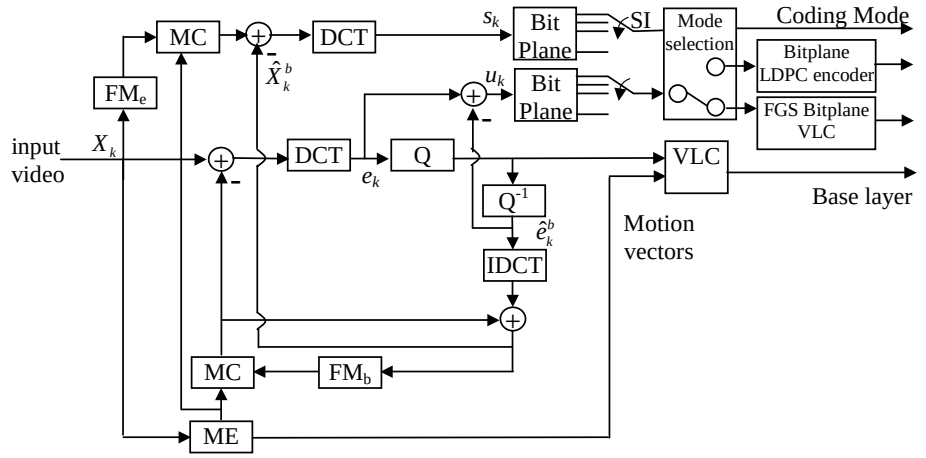


Figure 4.7: Model parameters of u_k estimated by EM using the video frames from *Akiyo*.

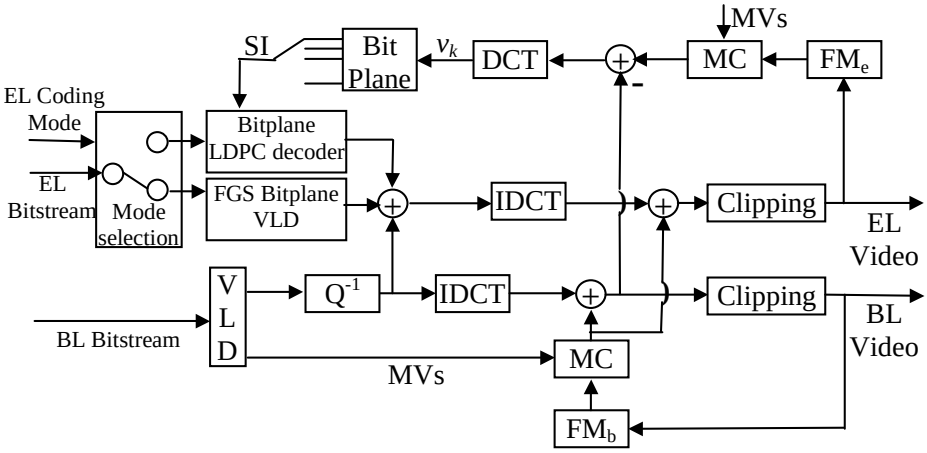
well captured by our model. Note, however, that such high quality base layer scenarios are in general of limited practical interest.

4.5 Codec Architecture and Implementation Details

Fig. 4.8 depicts the WZS encoding and decoding diagrams implemented based on the MPEG-4 FGS codec. Let X_k, $\hat{X}_k^b$ and $\hat{X}_k^e$ be the current frame, its BL and EL reconstructed frames, respectively.



(a)



(b)

Figure 4.8: Diagram of WZS encoder and decoder. (a) WZS encoder, (b) WZS decoder. FM: frame memory, ME: motion estimation, MC: motion compensation, SI: side information, BL: base layer, EL: enhancement layer, VLC: variable-length encoding, VLD: variable-length decoding.

4.5.1 Encoding Algorithm

At the base layer, the prediction residual e_k in DCT domain, as shown in Fig. 4.8 (a), is given by

$$e_k = T(X_k - MC_k[\hat{X}^b_{k-1}]), \tag{4.17}$$

where $T(.)$ is the DCT transform, and $MC_k[.]$ is the motion-compensated prediction of the kth frame given $\hat{X}^b_{k-1}$. The reconstruction of e_k after base layer quantization and dequantization is denoted by $\hat{e}^b_k$.

Then, at the enhancement layer, as in Section 4.3.2, we define

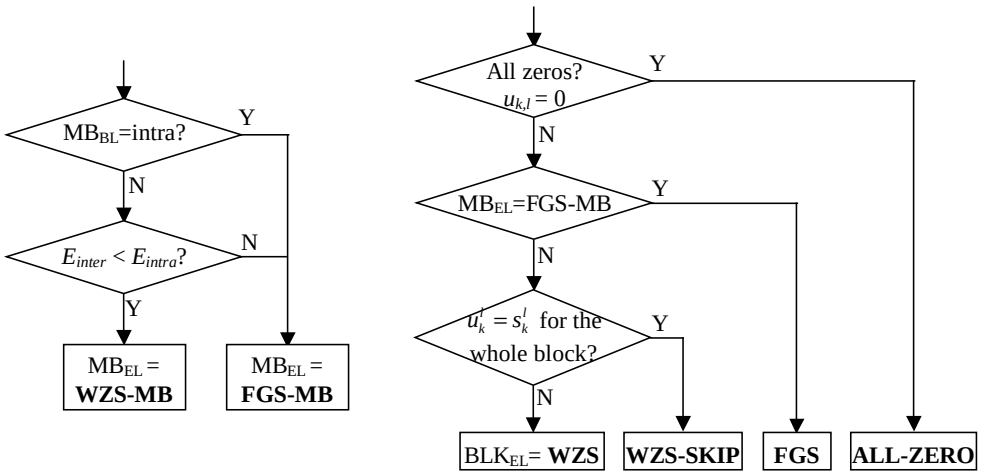
$$u_k = e_k - \hat{e}^b_k = T(X_k - MC_k[\hat{X}^b_{k-1}]) - \hat{e}^b_k. \tag{4.18}$$

The encoder SI s_k is constructed in a similar way as (4.11), while taking into account the motion compensation and DCT transform as

$$s_k = T(MC_k[X_{k-1}] - \hat{X}^b_k). \tag{4.19}$$

Both u_k and s_k are converted into bit-planes.

Based on the switching rule given in Section 4.4.2, we define our mode selection algorithm as shown in Fig. 4.9. At each bit-plane, we first decide the coding mode on the MB-basis as in Fig. 4.9 (a), and then in each MB we will decide the corresponding modes at the DCT block level to include the two special cases ALL-ZERO and WZS-SKIP (see Fig. 4.9 (b)). In either ALL-ZERO or WZS-SKIP mode, no additional information is sent to refine the block. The ALL-ZERO mode already exists in the current MPEG-4 FGS



(a) MB-based (b) Block-based

Figure 4.9: The block diagram of mode selection algorithm.

syntax. For a block coded in WZS-SKIP, the decoder just copies the corresponding block of the reference frame[1]. All the blocks in FGS mode are coded directly using MPEG-4 FGS bit-plane coding.

For blocks in WZS mode, we apply channel codes to exploit the temporal correlation between neighboring frames. Here, we choose low-density parity check (LDPC) codes [46, 47] for their low probability of undetectable decoding errors and near-capacity coding performance. A (n, k) LDPC code is defined by its parity-check matrix H with size $n \times (n-k)$. Given H, to encode an arbitrary binary input sequence c with length n, we multiply c with H and output the corresponding syndrome z with length $(n-k)$ [46]. In a practical implementation, this involves only a few binary additions due to the low-density property of LDPC codes. At bit-plane l, we first code the binary number $u_{k,l}$ for all coefficients in the WZS blocks, using LDPC codes to generate syndrome bits

[1]The WZS-SKIP mode may introduce some small errors due to the difference between the SI at the encoder and decoder.

at a rate determined by the conditional entropy in (4.13). We leave a margin of about 0.1 bits above the Slepian-Wolf limit (i.e., the conditional entropy) to ensure that the decoding error is negligible. Then, for those coefficients that become significant in the current bit-plane (i.e., coefficients that were 0 in all the more significant bit-planes and become 1 in the current bit-plane) , their sign bits are coded in a similar way using the sign bits of the corresponding s_k as SI.

The adaptivity of our scalable coder comes at the cost of an extra coding overhead. It includes: (1) the prediction modes for MBs and DCT blocks, (2) the *a priori* probability for $u_{k,l}$ (based on our experiments we assume a uniform distribution for sign bits) and channel parameters, and (3) encoding rate $(1 - k/n)$. A 1-bit syntax element is used to indicate the prediction mode for each MB at each bit-plane. The MPEG-4 FGS defines a most significant bit-plane level for each frame, which is found by first computing the residue with respect to the corresponding base layer for the frame and then determining what is the minimum number of bits needed to represent the largest DCT coefficient in the residue. Clearly, this most significant bit-plane level varies from frame to frame. Note that representation of many DCT blocks in a given frame is likely to require fewer bit-planes than the maximum number of bit-planes for the frame. Thus, for these blocks, the first few most significant bit-planes to be coded are likely to be all zero (for these blocks the residual energy after interpolation using the base layer is low, so that most DCT coefficients will be relatively small). To take advantage of this the MB prediction mode for a given bit-plane is not sent if all its six DCT blocks are ALL-ZERO. Note also that the number of bits needed to represent the MB mode is negligible for the

Table 4.2: Coding overhead for *News* sequence.

Bit-plane	1	2	3	4
Overhead percentage (%)	19.8	9.6	7.5	4.6

least significant bit-planes, as compared to the number of bits needed to code the bit-planes. It is also worth pointing out that this mode selection overhead is required as well for a closed-loop coder that attempts to exploit temporal correlation through the mode-switching algorithm. For a MB in WZS-MB mode, the block mode (either WZS or WZS-SKIP) is signaled by an additional 1-bit syntax. This overhead depends on the number of MBs in WZS-MB mode, and a good entropy coding can be applied to reduce the overhead, since we have observed in our experiments that the two different modes have biased probabilities (see Fig. 4.11). The encoding rate of syndrome codes varies from 1/64 to 63/64 in incremental steps of size 1/64, and thus 6 bits are used to code the selected encoding rate. We use a fixed-point 10 bit representation for the different kinds of probabilities to be sent to the decoder. An example of the total overhead percentage at each bit-plane, which is calculated as the ratio between the number of overhead bits and the number of total bits to code this bit-plane, is given in Table 4.2 for *News* sequence.

4.5.2 Decoding Algorithm

Decoding of the EL bit-planes of X_k proceeds by using the EL reconstruction of the previous frame $\hat{X}^e_{k-1}$ to form the SI for each bit-plane. The syndrome bits received are used to decode the blocks in WZS mode. The procedure is the same as at the encoder,

except that the original frame X_{k-1} is now replaced by the high quality reconstruction $\hat{X}^e_{k-1}$ to generate SI

$$v_k = T(MC_k[\hat{X}^e_{k-1}] - \hat{X}^b_k). \tag{4.20}$$

The corresponding SI at each bit-plane is formed by converting v_k into bit-planes. The decoder performs sequential decoding since decoding a particular bit-plane can only be done after more significant bit-planes have been decoded.

We modified the conventional LDPC software [47, 52] for the Slepian-Wolf approach by taking the syndrome information into account during the decoding process based on probability propagation. We follow a method similar to that described in [46, 98] to force the search of the most probable codeword in a specified coset determined by the syndrome bits. One main difference is that the *a priori* probability of the source bits $u_{k,l}$ ($p_0 = \Pr(u_{k,l} = 0)$ and $p_1 = 1 - p_0$) is also considered in the decoding process. The likelihood ratio for each variable node at bit-plane l is given by

$$\begin{aligned} LLR &= \log \frac{\Pr(u_{k,l}=1|v_{k,l})}{\Pr(u_{k,l}=0|v_{k,l})} \\ &= \begin{cases} \log \frac{p_{10}}{1-p_{01}} + \log \frac{p_1}{p_0}, & \text{if } v_{k,l} = 0, \\ \log \frac{1-p_{10}}{p_{01}} + \log \frac{p_1}{p_0}, & \text{if } v_{k,l} = 1. \end{cases} \end{aligned} \tag{4.21}$$

where p_{ij} is the crossover probability defined in Fig.4.4 (a). The syndrome information is considered in the same way as in [46] when calculating the likelihood ratio at the check node.

4.5.3 Complexity Analysis

In our approach, the base layer structure is the same as in an MPEG-4 FGS system. An additional set of frame memory, motion-compensation (MC) and DCT modules is introduced for the EL coding at both the encoder and decoder. The MC and DCT operations are only done once per frame even for multi-layer coding. In comparison, the ET approach requires multiple motion-compensation prediction loops, each of which needs a separate set of frame memory, MC and DCT modules, as well as additional dequantization and IDCT modules to obtain each EL reconstruction. More importantly, for each EL, the ET approach needs to repeat all the operations such as reconstruction and prediction. Though our proposed approach requires correlation estimation at the encoder as discussed in Section 4.4, the additional complexity involved is very limited, including simple shifting, comparison and $+/-$ operations. Therefore, the proposed approach can be implemented in a lower complexity even for multiple layers.

It should be noted that the complexity associated with reconstructing the enhancement layers can be a significant portion of the overall encoding complexity in a closed-loop scalable encoder. While it is true that *full search* motion estimation (ME) (in base layer) may require a large amount of computational power, practical encoders will employ some form of fast ME, and the complexity of ME module can be substantially reduced. For example, [14] reports that ME (full-pel and sub-pel) takes only around 50% of the overall complexity in a practical non-scalable video encoder employing fast ME. As a result, the complexity of closing the loop (motion compensation, forward and inverse transforms, quantization and inverse quantization) becomes a significant fraction of overall codec

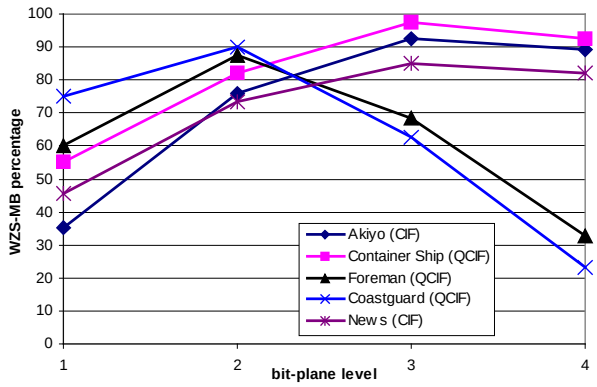
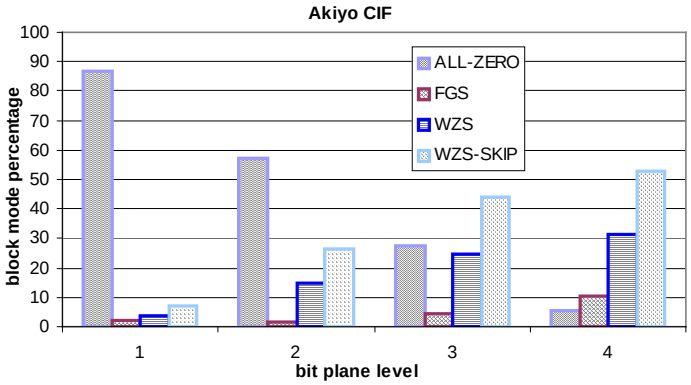


Figure 4.10: WZS-MB percentage for sequences in CIF and QCIF formats (BL quantization parameter=20, frame rate=30Hz).

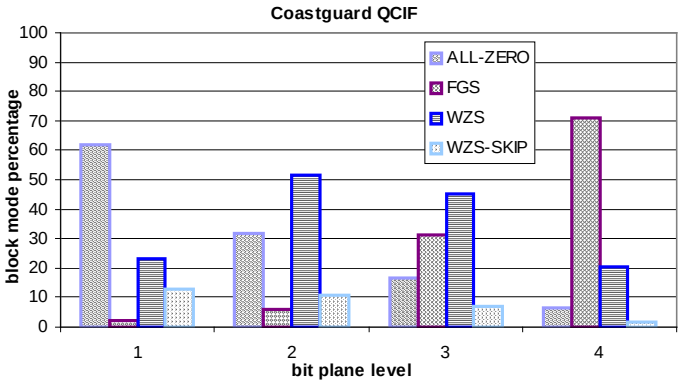
complexity. Moreover, we need to perform these operations in every enhancement layer in a closed-loop scalable system (while usually we perform ME only in base layer). In addition to computational complexity reduction, our system does not need to allocate the frame buffers to store the reconstructions in each enhancement layer. This can lead to considerable savings in memory usage, which may be important for embedded applications.

4.6 Experimental Results

Several experiments have been conducted to test the performance of the proposed WZS approach. We implemented a WZS video codec based on the MPEG-4 FGS reference software. In the experiments, we used the direct correlation estimation method, as it can lead to better compression efficiency as compared the model-based approach.



(a)



(b)

Figure 4.11: Percentages of different block modes for *Akiyo* and *Coastguard* sequences (BL quantization parameter=20, frame rate=30Hz).

4.6.1 Prediction Mode Analysis

In this section we analyze the block prediction modes at each bit-plane for various video sequences. Fig. 4.10 shows that the percentage of MBs in WZS-MB mode exceeds 50% for most video sequences (in some cases surpassing 90%, as in bit-plane 3 for *Akiyo* and *Container Ship*). Therefore there is potentially a large coding gain over MPEG-4 FGS with our proposed approach. The percentage of MBs in WZS-MB is on average higher for low-motion sequences (such as *Akiyo*) than for high-motion sequences (such as *Coastguard*), especially for lower significance bit-planes. Moreover, this percentage varies from bit-plane to bit-plane. For the most significant bit-planes, the FGS-MB mode tends to be dominant for some sequences (such as *Akiyo* and *News*), due to the low quality of the EL reconstruction of the previous frame. When the reconstruction quality improves, as more bit-planes are decoded, the temporal correlation is higher and the WZS-MB mode becomes dominant, for example for bit-planes 2 and 3 in Fig. 4.10. However, the WZS-MB percentage starts to drop for even lower significance bit-planes. This is because the temporal correlation decreases for these bit-planes which tend to be increasingly "noise-like".

The DCT block mode distribution in Fig. 4.11 illustrates how the motion characteristics of the source sequence affect the relative frequency of occurrence of each MB type. The *Akiyo* sequence has a much larger WZS-SKIP percentage, and a larger percentage of WZ coded blocks, than *Coastguard*; thus *Akiyo* sees more significant reductions in coding rate when WZS is introduced. In contrast, for *Coastguard* the percentage of blocks in WZS mode is less than that in FGS mode starting at bit-plane 4, thus showing that

as motion in the video sequence increases the potential benefits of exploiting temporal correlation in the manner proposed in this chapter decreases. Note that neither Fig. 4.10 nor Fig. 4.11 include the least two significant bit-planes since the PSNR range for these bit-planes are not of practical interest.

4.6.2 Rate-distortion Performance

4.6.2.1 Coding efficiency of WZS

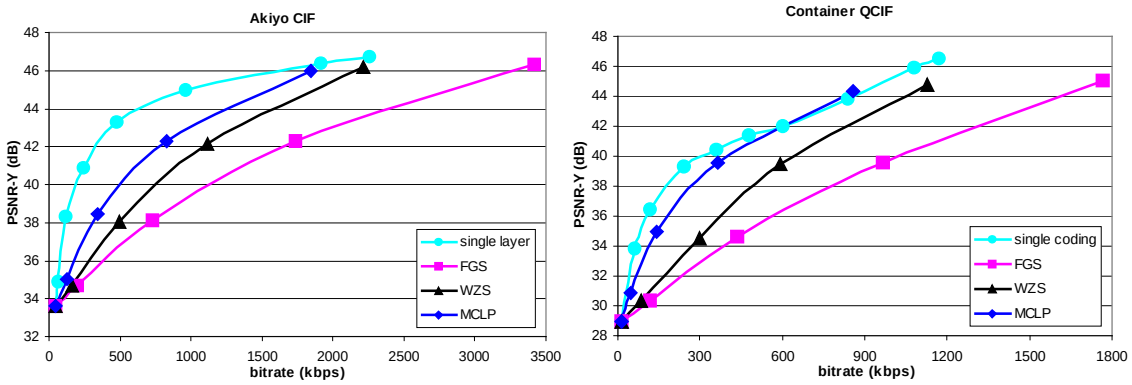
In this section we evaluate the coding efficiency of the proposed WZS approach. Simulation results are given for a series of test sequences in CIF (352×288) and QCIF (176×144) resolutions with frame rate 30Hz. *Akiyo* and *Container Ship* sequences have limited motion and low spatial detail, while the *Coastguard* and *Foreman* sequences have higher motion and more spatial detail. *News* sequence is similar to *Akiyo*, but with more background motion.

In addition to the MPEG-4 FGS and nonscalable (single layer) coding, we also compare our proposed approach with a multi-layer closed-loop (MCLP) system that exploits EL temporal correlation through multiple motion-compensation loops at the encoder. The same MPEG-4 baseline video coder is used for all the experimental systems (note that the proposed WZS framework does not inherently require the use of a specific BL video coder). The first video frame is intra-coded and all the subsequent frames are coded as P-frame (*i.e.*, IPPP...). The BL quantization parameter (QP) is set to 20. Prior to reporting the simulation results, we give a brief description of our proposed system together with the MCLP system.

Proposed WZS system. The DCT blocks are coded in four different modes as described in Section 4.6.1. An LDPC code is used to code those blocks in WZS mode at each bit-plane to exploit the EL correlation between adjacent frames. The encoding rate is determined by the correlation estimated at the encoder without constructing multiple motion-compensation loops. To limit the error introduced by WZS-SKIP mode due to the small difference between the encoder and decoder SI, we disable WZS-SKIP mode once every 10 frames in our implementation.

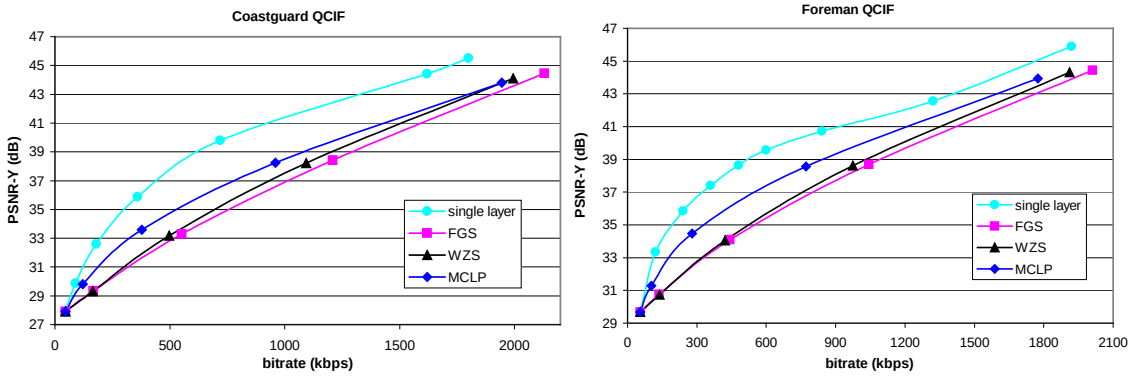
Multiple closed-loop (MCLP) system. This system is an approximation to the ET approach discussed in Section 4.3.1 through the mode-switching algorithm. We describe the coding procedure for each enhancement layer as follows. To code an EL which corresponds to the same quality achieved by bit-plane l in MPEG-4 FGS the encoder goes through the following steps. (i) Generate the EL reconstruction of the previous frame up to this bit-plane level, which we denote $\hat{x}_{k-1}^{l}$. (ii) Follow a switching rule similar to that proposed for the WZS system to determine the prediction mode of each MB, i.e., inter mode is chosen if $E_{inter} < E_{intra}$, and the FGS mode is chosen otherwise. Since the EL reconstruction is known at the encoder, it can calculate E_{inter} directly using the expression of (4.9). (iii) Calculate the EL residual r_k^e following (4.5) by using $\hat{x}_{k-1}^{l}$ as the predictor for inter mode, and the reconstruction of the current frame with more significant ELs $\hat{x}_k^{l-1}$ as the predictor for FGS mode. (iv) Convert r_k^e to bit-planes, and code those bit-planes that are at least as significant as bit-plane l (i.e., quantize to the lth bit-plane) to generate the compressed bitstream.

Figs. 4.12-4.14 provide a comparison between the proposed WZS, nonscalable coder, MPEG-4 FGS and the MCLP coder. The PSNR gain obtained by the proposed WZS



(a) (b)

Figure 4.12: Comparison between WZS, nonscalable coding, MPEG-4 FGS and MCLP for *Akiyo* and *Container Ship* sequences.



(a) (b)

Figure 4.13: Comparison between WZS, nonscalable coding, MPEG-4 FGS and MCLP for *Coastguard* and *Foreman* sequences.

approach over MPEG-4 FGS depends greatly on the temporal correlation degree of the video sequence. For sequences with higher temporal correlation, such as *Akiyo* and *Container Ship*, the PSNR gain of WZS is greater than that for lower temporal correlation sequences, such as *Foreman*, e.g., 3-4.5 dB PSNR gain for the former, as compared to 0.5-1 dB gain for the latter.

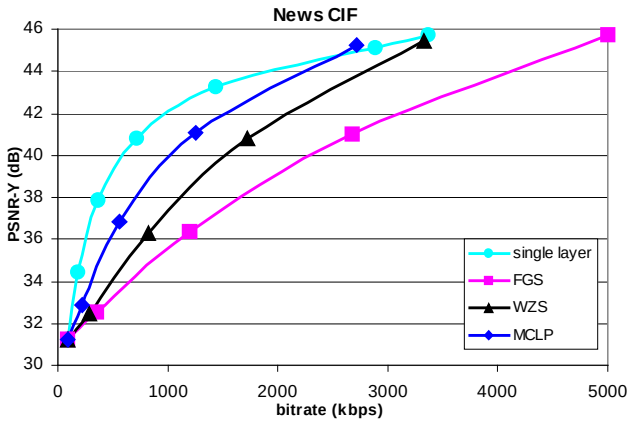
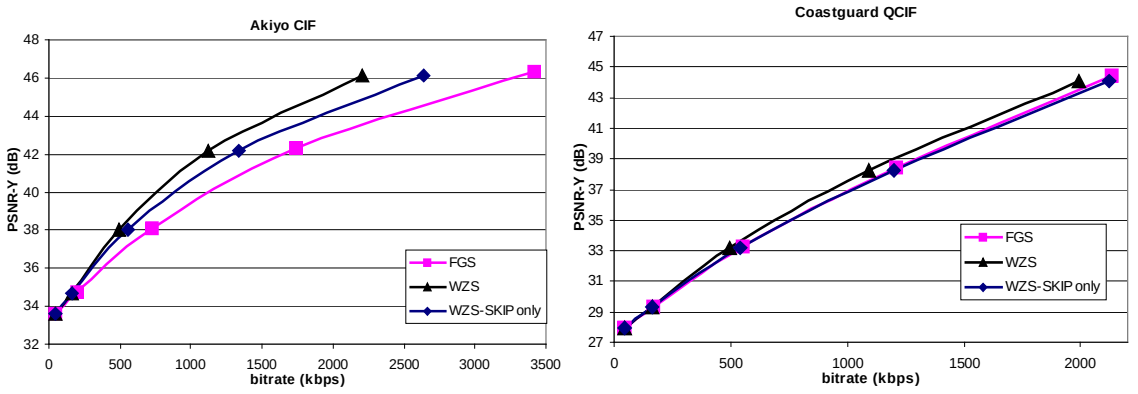


Figure 4.14: Comparison between WZS, nonscalable coding, MPEG-4 FGS and MCLP for *News* sequence.

To demonstrate the efficiency of Wyner-Ziv coding for WZS blocks, we compare the proposed coder to a simplified version that uses only the ALL-ZERO, FGS, and WZS-SKIP modes (which we call the "WZS-SKIP only" coder). The "WZS-SKIP only" coder codes the WZS blocks in FGS instead. Fig. 4.15 shows that, for both *Akiyo* and *Coast-guard* sequences, there is a significant improvement by adding the WZS mode. Note that the PSNR values for a given bit-plane level are exactly the same for the two coders. The only difference is the number of bits used to code those blocks that are coded in WZS mode. Thus the coding gain of Wyner-Ziv coding (exploiting temporal correlation) over the original bit-plane coding (that does not exploit temporal correlation) can be quantified as a percentage reduction in rate We present this in two different ways as shown in Tables 4.3 and 4.4 [2]. Table 4.3 provides the rate savings for only those blocks in WZS mode. It can be seen that *Akiyo* achieves larger coding gain than *Coastguard* due to higher temporal correlation. Table 4.4 provides the overall rate savings (i.e., based on

[2]It is usually required for a LDPC coder to have a large code length to achieve good performance. If the number of WZS blocks is not enough for the required code length we force all blocks in the bit-plane to be coded in FGS mode instead. This happens, for example, for the most significant bit-plane of most sequences. Thus, only the results for bit-planes 2-4 are shown in these tables.



(a) (b)

Figure 4.15: Comparison between WZS and "WZS-SKIP only" for *Akiyo* and *Coastguard* sequences.

Table 4.3: Rate savings due to WZS for WZS blocks only (percentage reduction in rate with respect to using FGS instead for those blocks).

Bit-plane level	2	3	4
Akiyo	24.66	31.20	26.71
Coastguard	19.98	22.91	19.19

Table 4.4: Overall rate savings due to WZS (percentage reduction in overall rate as compared to FGS).

Bit-plane level	2	3	4
Akiyo	10.98	16.61	16.11
Coastguard	8.38	8.68	6.08

the total rate needed to code the sequence). These rate savings reflect not only the efficiency of coding each WZS block by Wyner-Ziv coding but also the percentage of blocks that are coded in WZS mode.

As seen from Figures 4.12-4.14, there is still a performance gap between WZS and MCLP. We compare the main features of these two approaches that affect the coding performance in Table 4.5. It should be clarified that occasionally, at very high rate for low-motion sequences, the MCLP approach can achieve similar (or even better) coding performance than the nonscalable coder. That is because bit-plane coding is more efficient

Table 4.5: Comparisons between MCLP and WZS

Multiple closed-loop approach	*Wyner-Ziv scalability approach*
1. Exploits temporal correlation through closed-loop predictive coding 2. Efficient bit-plane coding (run, end-of-plane) of the EL residual in (4.5) to exploit the correlation between consecutive zeros in the same bit-plane 3. The residual error between the source and EL reference from the previous frame may increase the dynamic range of the difference and thus cause fluctuations in the magnitude of residue coefficients (as the number of refinement iterations grows, the magnitude of residues in some coefficients can actually increase, even if the overall residual energy decreases). 4. Each EL has to code its own sign map, and therefore for some coefficients the sign bits are coded more than once.	1. Exploits temporal correlation through Wyner-Ziv coding 2. Channel coding techniques designed for memoryless channels cannot exploit correlation between source symbols 3. The source information to be coded is exactly the same as the EL bit-planes in MPEG-4 FGS, and therefore there are no fluctuations in magnitude and no additional sign bits are needed. 4. An extra encoding rate margin is added to compensate for the small mismatch between encoder and decoder SI as well as for the practical channel coders which cannot achieve the Slepian-Wolf limit exactly.

than nonscalable entropy coding when compressing the high-frequency DCT coefficients. We believe that the performance gap between WZS and MCLP is mainly due to the relative inefficiency of the current channel coding techniques as compared to bit-plane coding. We expect that large rate savings with respect to our present WZS implementation can be achieved if better channel codes are used, that can perform closer to the Slepian-Wolf limit, or more advanced channel coding techniques are designed for more complex channels, which can take advantage of the existence of correlation among channel errors.

Table 4.6: The base layer PSNR (dB) for different QP.

Base layer QP	31	20	8
Akiyo	32.03	33.61	38.05
Container Ship	26.97	28.93	34.11
News	29.17	31.23	36.09

4.6.2.2 Rate-distortion performance vs. base layer quality

It is interesting to consider the effects of the base-layer quality on the EL performance of the WZS approach. We use *Akiyo*, *Container Ship* and *News* sequences in the experiment. Table 4.6 shows the base layer PSNR (for luminance component only) for several sequences under different base layer quantization parameter (QP) values. The PSNR gains obtained by the proposed WZS approach over MPEG-4 FGS are plotted in Fig. 4.16. The coding gain achieved by WZS decreases if a higher quality base layer is used, as seen from Fig. 4.16 when the base layer QP decreases to 8. That is because the temporal correlation between the successive frames is already well exploited by a high-quality base layer. This observation is in agreement with the analysis in Section 4.3.1.

4.6.2.3 Comparisons with Progressive Fine Granularity Scalable (PFGS) coding

The PFGS scheme proposed by Wu et al. [93] improves the coding efficiency of FGS, by employing an additional motion compensation loop to code the EL, for which several FGS bit-planes are included in the loop to exploit EL temporal correlation. Fig. 4.17 compares the coding performance between WZS and PFGS for *Foreman* sequence. WZS performs

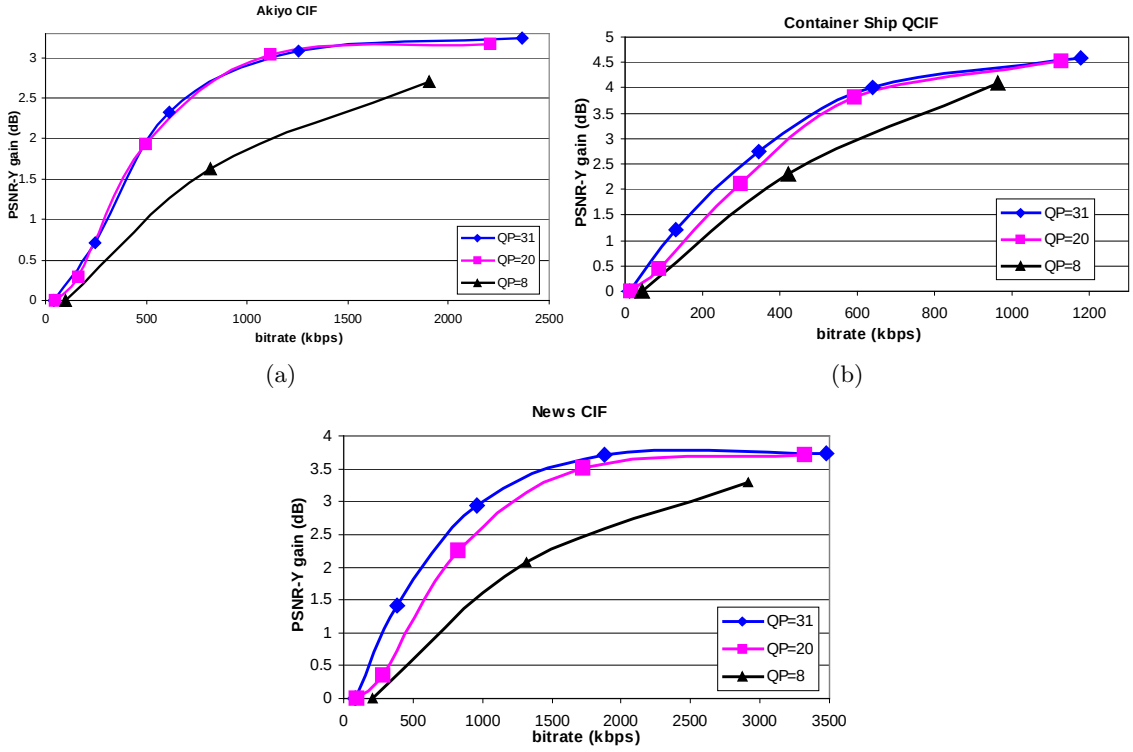


Figure 4.16: The PSNR gain obtained by WZS over MPEG-4 FGS for different base layer qualities.

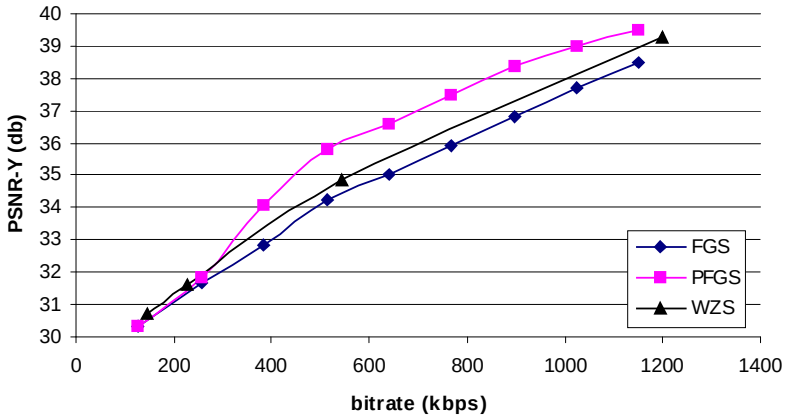


Figure 4.17: Compare WZS with MPEG-4 PFGS for *Foreman* CIF sequence (Base layer QP=19, frame rate=10Hz). The PFGS results are provided by Wu et al. from [31].

worse than PFGS. In addition to the limitation of current techniques for Wyner-Ziv coding, the performance gap may come from the difference of the prediction link structure between these two approaches. WZS creates a multi-layer Wyner-Ziv prediction link to connect the same bit-plane level in successive frames. However, in PFGS, usually at least two or three FGS bit-planes are used in the EL prediction for all the bit-planes. Thus, this structure is beneficial to the most significant bit-planes (for example, the 1st or 2nd bit-plane) as they have higher-quality reference than what they would in WZS.

On the other hand, our proposed WZS techniques can be easily combined with a PFGS coder such that the more significant bit-planes can be encoded in a closed-loop manner by PFGS techniques, while the least significant bit-planes are predicted through Wyner-Ziv links to exploit the remaining temporal correlation. Fig. 4.10 shows that for some sequences (especially those with low-motion) the temporal correlation for some lower significance bit-planes (e.g., bit-plane 4) is still high, so that WZS-MB mode is chosen for a considerable percentage of MBs. Thus, we expect that further gain would be achieved with our techniques over what is achievable with PFGS.

4.7 Conclusions

We have presented a new practical Wyner-Ziv scalable coding structure to achieve high coding efficiency. By using principles from distributed source coding, the proposed coder is able to exploit the enhancement-layer correlation between adjacent frames without explicitly constructing multiple motion-compensation loops, and thus reduce the encoder complexity. In addition, it has the advantage of backward compatibility with standard video codecs by using a standard CLP video coder as base layer. Two efficient methods are proposed for correlation estimation based on different trade-off between the complexity and accuracy at the encoder even when the exact reconstruction value of the previous frame is unknown. Simulation results show much better performance over MPEG-4 FGS for sequences with high temporal correlation and limited improvement for high-motion sequences. Though we implemented the proposed Wyner-Ziv scalable framework in the MPEG-4 FGS software as bit-planes, it can be integrated with other SNR scalable coding techniques.

Further work is needed within the proposed framework to improve coding efficiency and provide flexible bandwidth adaptation and robustness. In particular, the selection of efficient channel coding techniques well suited for distributed source coding deserves additional investigation. Another possible reason for the gap between our proposed coder and a nonscalable coder is due to less accurate motion compensation prediction in the enhancement layer when sharing motion vectors with the base layer. This can be improved by exploring the flexibility at the decoder, an important benefit of Wyner-Ziv

coding, to refine the enhancement layer motion vectors by taking into account the received enhancement layer information from the previous frame. In terms of bandwidth adaptation, the current coder cannot fully achieve fine granularity scalability, given that the LDPC coder can only decode the whole block at the bit-plane boundary. There is recent interest on punctured LDPC codes [30], and the possibility of using this code for bandwidth adaptation is under investigation. In addition, it is also interesting to evaluate the error resilience performance of the proposed coder. In principle, the Wyner-Ziv coding has more tolerance on noise introduced to the side information.

Chapter 5

Conclusions and Future Work

In this thesis we have addressed the problem of network-adaptive video coding and streaming, particularly for those source coding algorithms that provide scalability to match changes in the network environment.

First, we extend the rate-distortion optimized streaming (RaDiO) framework proposed in [19, 20] to address a more general coding scenario, where multiple decoding paths are possible. The source encoder produces redundant encoded data to provide flexibility for the scheduling algorithm to choose the right subset of data units to send, based on the current network conditions and previous transmission history. This approach allows the system redundancy to be optimized dynamically at the streaming stage rather than those pre-optimized at the encoding stage. Therefore, the proposed streaming framework is more appropriate for adaptation in a wide range of network environments. In terms of source coding, we propose a new coding algorithm that supports multiple decoding paths, namely, multiple description layered coding (MDLC), which combines the advantages of layered coding (LC) and multiple description coding (MDC). Experimental results show that, when applied together with the extended RaDiO framework, the proposed MDLC

system provides more robust and efficient video communication over a wider range of network scenarios and application requirements.

Second, we propose a novel scalable coding approach by introducing distributed source coding (DSC) in the enhancement-layer prediction, to achieve a better coding performance with reasonable encoding complexity. In this approach the decoder complexity increases as compared to existing approaches. Specifically, in order to reduce the encoding complexity, we do not explicitly construct multiple motion-compensation loops at the encoder, while, at the decoder, side information (SI) is constructed to combine spatial and temporal information in a manner that seeks to approximate the ET approach in [66]. Experimental results show improvements in coding efficiency of 3-4.5 dB over MPEG-4 FGS for video sequences with high temporal correlation.

It is worth noting that some of the novel concepts proposed in this research are not limited to scalable coding. For example, the rate-distortion optimized streaming framework can support the case of multiple independently encoded non-scalable bit streams. The DSC principle has also shown great potential to provide important functionalities which are difficult to achieve using traditional techniques. One example is to provide flexible playback at the decoder, by for example supporting both backward and forward frame-by-frame playback [17]. Despite the fact that this research has achieved interesting results and proposed a number of novel algorithms, there are still relevant topics or ideas that can be addressed in future research. We describe several possible research directions as follows:

- *Source Optimization for the Extended RaDiO Framework.* Our current work has proposed rate-distortion optimized scheduling algorithms for a given source codec

with redundancy. Although optimization of source redundancy (e.g., in a MDC codec) has been studied by many researchers when simple scheduling algorithms are used, the appropriate redundancy allocation for such a source codec (e.g., MDC or MDLC) is still an open issue when the coder is to be used along with an intelligent scheduling algorithm.

- *Packet Scheduling Algorithms for Peer-to-Peer (P2P) Systems.* P2P video streaming is highly difficult due to the increased uncertainty about the number of available peers and bandwidth throughput from each sender. The scheduling algorithm should decide not only which packets to send and when to send them, but also from which peer to send them. The optimization should consider system-level issues, such as the peer and path stability, and the congestion level of the network. Furthermore, it should also provide efficient coordination among peers to tolerate a certain peer's failure to provide service.

- *Study of Trade-Off between Entropy Coding and DSC.* DSC replaces the traditional entropy coding by a representation (e.g., syndrome codes) based on a channel code. Thus, for a particular source it is hard to exploit the biased symbol probabilities and the correlation inside the symbol sequence. Future work should start from a detailed study of the tradeoffs between entropy coding and DSC-based coding under different scenarios, and consider techniques to combine entropy coding and DSC-based codes.

- *Source-Adaptive Correlation Estimation for DSC.* Correlation estimation is a key issue to impact the coding performance. Current approaches [16, 18, 26] tend to

oversimplify the correlation structure and then apply an advanced channel code (e.g., LDPC code or turbo code) to hope for good coding performance. This may be applicable for simple source-SI characteristics, for example, the additive Gaussian white noise channel (AWGN). But video signals are so complex that a unified correlation structure may fail to capture correlation information that is potentially useful for coding efficiency. Thus, it may be interesting to consider an alternative approach by developing a more complicated correlation structure to increase the adaptivity to the source, following a similar idea to that applied in H.264, with the introduction of more prediction modes. In this case a simple channel code (of short length due to a possible small-unit based correlation estimation) with good correlation structure may achieve better coding performance. Here, decoding errors are possible due to the short-length weak channel code and the algorithm should provide a way to limit the error propagation.

Bibliography

[1] A. Aaron and B. Girod. Compression with side information using turbo codes. In *Proc. Data Compression Conference*, pages 252–261, Apr. 2002.

[2] A. Aaron, E. Setton, and B. Girod. Towards practical Wyner-Ziv coding of video. In *Proc. Int'l Conf. Image Processing*, Sept. 2003.

[3] A. Aaron, R. Zhang, and B. Girod. Wyner-Ziv coding of motion video. In *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 2002.

[4] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Trans. Information Theory*, 42(6):1737–1744, Nov. 1996.

[5] J. Apostolopoulos. Reliable video communication over lossy packet networks using multiple state encoding and path diversity. In *Proc. Visual Communications and Image Processing*, pages 392–409, Jan. 2001.

[6] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee. On multiple description streaming with content delivery networks. In *Proc. Conf. Computer Communications (INFOCOM)*, pages 1736–1745, June 2002.

[7] J. Arnold, M. Frater, and Y. Wang. Efficient drift-free signal-to-noise ratio scalability. *IEEE Trans. Circuits and Systems for Video Technology*, 10(1):70–82, Feb. 2000.

[8] J. Bajcsy and P. Mitran. Coding for the Slepian-Wolf problem with turbo codes. In *Proc. Global Telecommunications Conf. (GLOBECOM)*, pages 1400–1404, Nov. 2001.

[9] B. Birney. Intelligent streaming. http://www.microsoft.com/windows/ windows-media/howto/articles/intstreaming.aspx, May 2003.

[10] J. Chakareski, J. Apostolopoulos, S. Wee, W. Tan, and B. Girod. Rate-distortion hint tracks for adaptive video streaming. *IEEE Trans. Circuits and Systems for Video Technology*, 15(10):1257–1269, Oct. 2005.

[11] J. Chakareski, P. Chou, and B. Girod. Rate-distortion optimized streaming from the edge of the network. In *Proc. Workshop on Multimedia Signal Processing*, Dec. 2002.

[12] J. Chakareski and B. Girod. Rate-distortion optimized packet scheduling and routing for media streaming with path diversity. In *Proc. Data Compression Conference*, Mar. 2003.

[13] J. Chakareski, S. Han, and B. Girod. Layered coding vs. multiple descriptions for video streaming over multiple paths. *ACM/Springer Multimedia Systems Journal*, 10(4):275–285, Apr. 2005.

[14] H.-Y. Cheong, A. M. Tourapis, and P. Topiwala. Fast motion estimation within the JVT codec. Technical Report JVT-E023, JVT meeting, Geneva, Oct. 2002. http://ftp3.itu.ch/av-arch/jvt-site/2002_10_Geneva/JVT-E023.doc.

[15] G. Cheung and W. Tan. Directed acyclic graph based source modeling for data unit selection of streaming media over QoS networks. In *Proc. Int'l Conf. Multimedia and Exhibition*, volume 1, Aug. 2002.

[16] N.-M. Cheung, H. Wang, and A. Ortega. Correlation estimation for distributed source coding under information exchange constraints. In *Proc. Int'l Conf. Image Processing*, Sept. 2005.

[17] N.-M. Cheung, H. Wang, and A. Ortega. Video compression with flexible playback order based on distributed source coding. In *Proc. Visual Communications and Image Processing*, Jan. 2006.

[18] N.-M. Cheung, H. Wang, and A. Ortega. Sampling-based correlation estimation for distributed image and video coding under rate and complexity constraints. *Submitted to IEEE Trans. Image Processing*, Aug. 2007.

[19] P. Chou and Z. Miao. Rate-distortion optimized sender-driven streaming over best-effort networks. In *Proc. Workshop on Multimedia Signal Processing*, volume 1, pages 587–592, Oct. 2001.

[20] P. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. *IEEE Trans. Multimedia*, 8(2):390–404, Apr. 2006.

[21] P. Chou and A. Sehgal. Rate-distortion optimized receiver-driven streaming over best-effort networks. In *Proc. Int'l Packet Video Workshop*, volume 1, Apr. 2002.

[22] P. Chou, H. Wang, and V. Padmanabhan. Layered multiple description coding. In *Proc. Int'l Packet Video Workshop*, volume 1, Apr. 2003.

[23] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik. Video coding for streaming media delivery on the internet. *IEEE Trans. Circuits and Systems for Video Technology*, 11(3):269–281, Mar. 2001.

[24] W. Equitz and T. Cover. Successive refinement of information. *IEEE Trans. Information Theory*, 37(2):269–275, Mar. 1991.

[25] J. Garcia-Frias. Compression of correlated binary sources using turbo codes. *IEEE Communications Letters*, 5(10):417–419, Oct. 2001.

[26] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero. Distributed video coding. *Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery*, 93(1):71–83, Jan. 2005.

[27] B. Girod, J. Chakareski, M. Kalman, Y. Liang, E. Setton, and R. Zhang. Advances in network-adaptive video streaming. In *Proc. 2002 Tyrrhenian International Workshop on Digital Communications (IWDC 2002)*, Sept. 2002.

[28] B. Girod and N. Farber. Feedback-based error control for mobile video transmission. *Proceedings of the IEEE*, 87(10):1707–1723, Oct. 1999.

[29] V. Goyal. Multiple description coding: compression meets the network. *IEEE Signal Processing Magazine*, 18(5):74–93, Sept. 2001.

[30] J. Ha, J. Kim, and S. McLaughlin. Rate-compatible puncturing of low-density parity-check codes. *IEEE Trans. Information Theory*, 50(11):2824–2836, Nov. 2004.

[31] Y. He, F. Wu, S. Li, Y. Zhong, and S. Yang. H.26L-based fine granularity scalable video coding. In *Proc. Int'l Symp. Circuits and Systems*, May 2002.

[32] C.-Y. Hsu, A. Ortega, and M. Khansari. Rate control for robust video transmission over burst-error wireless channels. *IEEE J. Selected Areas in Communications*, 17(5):1–18, May 1999.

[33] H.-C. Huang, C.-N. Wang, and T. Chiang. A robust fine granularity scalability using trellis-based predictive leak. *IEEE Trans. Circuits and Systems for Video Technology*, 12(6):372–385, June 2002.

[34] ISO/IEC 13818-2. Generic coding of moving pictures and associated audio, part-2 video. Nov. 1994.

[35] ISO/IEC 14496-2/FPDAM4. Coding of audio-visual objects, part-2 visual, amendment 4: streaming video profile. July 2000.

[36] ISO/IEC JTC1/SC29/WG11/N2202. MPEG-4 version 2 visual working draft rev. 3.0. Mar. 1998.

[37] ITU-T Recommendation H.263 version 2 (H.263+). Video coding for low bitrate communication. Feb. 1998.

[38] A. Jagmohan and N. Ahuja. Wyner-Ziv encoded predictive multiple descriptions. In *Proc. Data Compression Conference*, pages 213–222, Mar. 2003.

[39] W. Jiang and A. Ortega. Multiple description coding via polyphase transform and selective quantization. In *Proc. Visual Communications and Image Processing*, Jan. 1999.

[40] M. Kalman and B. Girod. Rate-distortion optimized streaming of video with multiple independent encodings. In *Proc. Int'l Conf. Image Processing*, Oct. 2004.

[41] L. Kondi. A rate-distortion optimal hybrid scalable/multiple-description video codec. *IEEE Trans. Circuits and Systems for Video Technology*, 15(7):921–927, July 2005.

[42] C.-F. Lan, A. Liveris, K. Narayanan, Z. Xiong, and C. Georghiades. Slepian-Wolf coding of multiple m-ary sources using LDPC codes. In *Proc. Data Compression Conference*, page 549, Mar. 2004.

[43] Y.-C. Lee, J. Kim, Y. Altunbasak, and R. Mersereau. Performance comparisons of layered and multiple description coded video streaming over error-prone networks. In *Proc. Int'l Conf. Communications*, pages 35–39, May 2003.

[44] W. Li. Overview of fine granularity scalalability in MPEG-4 video standard. *IEEE Trans. Circuits and Systems for Video Technology*, 11(3):301–317, Mar. 2001.

[45] Y. J. Liang and B. Girod. Prescient r-d optimized packet dependency management for low-latency video streaming. In *Proc. Int'l Conf. Image Processing*, Sept. 2003.

[46] A. Liveris, Z. Xiong, and C. Georghiades. Compression of binary sources with side information at the decoder using LDPC codes. In *IEEE Communications Letters*, Oct. 2002.

[47] D. MacKay and R. Neal. Near Shannon limit performance of low density parity check codes. *Electronics Letters*, 32:1645–1646, Aug. 1996. Reprinted with printing errors corrected in vol. 33, pp. 457-458.

[48] S. Mao, S. Lin, S. Panwar, Y. Wang, and E. Celebi. Video transport over ad hoc networks: multistream coding with multipath transport. *IEEE J. Selected Areas in Communications*, 21(10):1721–1737, Dec. 2003.

[49] Z. Miao and A. Ortega. Optimal scheduling for streaming of scalable media. In *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 2000.

[50] Z. Miao and A. Ortega. Expected run-time distortion based scheduling for delivery of scalable media. In *Proc. Int'l Packet Video Workshop*, volume 1, Apr. 2002.

[51] Z. Miao and A. Ortega. Fast adaptive media scheduling based on expected run-time distortion. In *Proc. Asilomar Conf. Signals, Systems, and Computers*, volume 1, Nov. 2002.

[52] R. Neal. LDPC software [Online].
Available: http://www.cs.toronto.edu/∼radford/ldpc.software.html.

[53] V. Nguyen, E. Chang, and W. Ooi. Layered coding with good allocation outperforms multiple description coding over multiple paths. In *Proc. Int'l Conf. Multimedia and Exhibition*, pages 1067–1070, June 2004.

[54] A. Ortega and K. Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*, 15(6):23–50, Nov. 1998.

[55] V. Padmanabhan, H. Wang, and P. Chou. Resilient peer-to-peer streaming. In *Proc. Int'l Conf. Network Protocols*, Nov. 2003.

[56] M. Podolsky, S. McCanne, and M. Vetterli. Soft ARQ for layered streaming media. *The Journal of VLSI Signal Processing*, 27(1-2):81–97, Feb. 2001.

[57] S. Pradhan, J. Chou, and K. Ramchandran. Duality between source coding and channel coding and its extension to the side information case. *IEEE Trans. Information Theory*, 49(5):1181–1203, May 2003.

[58] S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (DISCUS): design and construction. *IEEE Trans. Information Theory*, 49(3):626–643, Mar. 2003.

[59] R. Puri, A. Majumdar, P. Ishwar, and K. Ramchandran. Distributed video coding in wireless sensor networks. *IEEE Signal Processing Magazine*, 23(4):94–106, July 2006.

[60] R. Puri and K. Ramchandran. PRISM: a new robust video coding architecture based on distributed compression principles. In *Proc. Allerton Conf. Communications, Control, and Computing*, Oct. 2002.

[61] R. Puri and K. Ramchandran. PRISM: a video coding architecture based on distributed compression principles. Technical report, UC Berkeley/ERL Technical Report, Mar. 2003.

[62] A. Reibman. Optimizing multiple description video coders in a packet loss environment. In *Proc. Int'l Packet Video Workshop*, Apr. 2002.

[63] A. Reibman, H. Jafarkhani, M. Orchard, and Y. Wang. Performance of multiple description coders on a real channel. In *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing*, pages 2415 –2418, Mar. 1999.

[64] A. Reibman, H. Jafarkhani, Y. Wang, and M. Orchard. Multiple description video using rate-distortion splitting. In *Proc. Int'l Conf. Image Processing*, pages 978 –981, Oct. 2001.

[65] A. Reibman, Y. Wang, X. Qiu, Z. Jiang, and K. Chawla. Transmission of multiple description and layered video over an EGPRS wireless network. In *Proc. Int'l Conf. Image Processing*, pages 136–139, Sept. 2000.

[66] K. Rose and S. Regunathan. Toward optimality in scalable predictive coding. *IEEE Transactions on Image Processing*, 10:965–976, July 2001.

[67] A. Sehgal, A. Jagmohan, and N. Ahuja. A casual state-free video encoding paradigm. In *Proc. Int'l Conf. Image Processing*, Sept. 2003.

[68] A. Sehgal, A. Jagmohan, and N. Ahuja. Scalable video coding using Wyner-Ziv codes. In *Proc. Picture Coding Symposium*, Dec. 2004.

[69] A. Sehgal, A. Jagmohan, and N. Ahuja. Wyner-Ziv coding of video: an error-resilient compression framework. *IEEE Trans. Multimedia*, 6(2):249–258, Apr. 2004.

[70] S. Servetto. Lattice quantization with side information. In *Proc. Data Compression Conference*, pages 510–519, Mar. 2000.

[71] S. Servetto, V. Vaishampayan, and N. Sloane. Multiple description lattice vector quantization. In *Proc. Data Compression Conference*, pages 13–22, Mar. 1999.

[72] R. Singh, A. Ortega, L. Perret, and W. Jiang. Comparison of multiple description coding and layered coding based on network simulations. In *Proc. Visual Communications and Image Processing*, pages 929–939, jan 2000.

[73] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *IEEE Trans. Information Theory*, 19(4):471–480, July 1973.

[74] Y. Steinberg and N. Merhav. On successive refinement for the Wyner-Ziv problem. *IEEE Trans. Information Theory*, 50(8):1636 – 1654, Aug. 2004.

[75] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, Nov. 1998.

[76] M.-T. Sun and A. Reibman. *Compressed Video over Networks*. Marcel Dekker, New York, 2001.

[77] M. Tagliasacchi, A. Majumdar, and K. Ramchandran. A distributed-source-coding based robust spatio-temporal scalable video codec. In *Proc. Picture Coding Symposium*, Dec. 2004.

[78] V. Vaishampayan. Design of multiple description scalar quantizers. *IEEE Trans. Information Theory*, 39(3):821–834, May 1993.

[79] M. van der Schaar and H. Radha. Adaptive motion-compensation fine-granular-scalability (AMC-FGS) for wireless video. *IEEE Trans. Circuits and Systems for Video Technology*, 12(6):360–371, June 2002.

[80] C. D. Vleeschouwer, J. Chakareski, and P. Frossard. The virtue of patience in low-complexity scheduling of packetized media with feedback. *IEEE Trans. Multimedia*, 9(2):348–365, Feb. 2007.

[81] H. Wang, N.-M. Cheung, and A. Ortega. WZS: Wyner-Ziv scalable predictive video coding. In *Proc. Picture Coding Symposium*, Dec. 2004.

[82] H. Wang, N.-M. Cheung, and A. Ortega. A framework for adaptive scalable video coding using Wyner-Ziv techniques. *EURASIP Journal on Applied Signal Processing*, 2006.

[83] H. Wang and A. Ortega. Robust video communication by combining scalability and multiple description coding techniques. In *Proc. Symp. Electronic Imaging*, volume 1, Jan. 2003.

[84] H. Wang and A. Ortega. Scalable predictive coding by nested quantization with layered side information. In *Proc. Int'l Conf. Image Processing*, Oct. 2004.

[85] H. Wang and A. Ortega. Rate-distortion based scheduling of video with multiple decoding paths. In *Proc. Symp. Electronic Imaging*, Jan. 2005.

[86] H. Wang and A. Ortega. Rate-distortion adaptive streaming of video with multiple decoding paths. *Submitted to IEEE Trans. Image Processing*, Sept. 2007.

[87] Y. Wang, M. Orchard, V. Vaishampayan, and A. Reibman. Multiple description coding using pairwise correlating transforms. *IEEE Trans. Image Processing*, 10(3):351–366, Mar. 2001.

[88] Y. Wang, J. Ostermann, and Y. Zhang. *Video Processing and Communication*. Prentice Hall, New Jersy, 2002.

[89] Y. Wang, A. Reibman, and S. Lin. Multiple description coding for video delivery. *Proceedings of the IEEE*, 93(1):57–70, Jan. 2005.

[90] S. Wenger. Video redundancy coding in H.263+. In *Workshop on Audio-Visual Services for Packet Networks (AVSPN 1997)*, Sept. 1997.

[91] S. Wenger, G. Knorr, J. Ott, and F. Kossentini. Error resilience support in h.263+. *IEEE Trans. Circuits and Systems for Video Technology*, 8(7):867–877, Nov. 1998.

[92] D. Wilson and M. Ghanbari. Optimisation of two-layer SNR scalability for MPEG-2 video. In *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing*, Apr. 1997.

[93] F. Wu, S. Li, and Y.-Q. Zhang. A framework for efficient progressive fine granularity scalable video coding. *IEEE Trans. Circuits and Systems for Video Technology*, 11(3):332–344, Mar. 2001.

[94] A. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Information Theory*, 22:1–10, Jan. 1976.

[95] Z. Xiong, A. Liveris, S. Cheng, and Z. Liu. Nested quantization and Slepian-Wolf coding: A Wyner-Ziv coding paradigm for i.i.d. sources. In *Proc. IEEE Workshop on Statistical Signal Processing (SSP)*, Sept. 2003.

[96] Q. Xu and Z. Xiong. Layered Wyner-Ziv video coding. In *Proc. Visual Communications and Image Processing*, Jan. 2004.

[97] R. Zamir and S. Shamai. Nested linear/lattice codes for Wyner-Ziv encoding. In *Proc. Information Theory Workshop*, pages 92–93, June 1998.

[98] R. Zamir, S. Shamai, and U. Erez. Nested linear/lattice codes for structured multiterminal binning. *IEEE Trans. Information Theory*, 48:1250–1276, June 2002.

[99] R. Zhang, S. Regunathan, and K. Rose. Video coding with optimal inter/intra-mode switching for packet loss resilience. *IEEE J. Selected Areas in Communications*, 18(6):966–976, June 2000.

[100] R. Zhang, S. Regunathan, and K. Rose. Optimized video streaming over lossy networks with real-time estimation of end-to-end distortion. In *Proc. Int'l Conf. Multimedia and Exhibition*, volume 1, Aug. 2002.

[101] Y. Zhou and W.-Y. Chan. Performance comparison of layered coding and multiple description coding in packet networks. In *Proc. Global Telecommunications Conference*, Nov. 2005.