RATE-DISTORTION ANALYSIS AND TRAFFIC MODELING OF SCALABLE VIDEO CODERS

A Dissertation

by

MIN DAI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2004

Major Subject: Electrical Engineering

RATE-DISTORTION ANALYSIS AND TRAFFIC MODELING OF SCALABLE VIDEO CODERS

A Dissertation

by

MIN DAI

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Approved as to style and content by:

December 2004

Major Subject: Electrical Engineering

ABSTRACT

Rate-Distortion Analysis and Traffic Modeling

of Scalable Video Coders. (December 2004)

Min Dai, B.S., Shanghai Jiao Tong University;

M.S., Shanghai Jiao Tong University

Co–Chairs of Advisory Committee: Dr. Andrew K. Chan
Dr. Dmitri Loguinov

In this work, we focus on two important goals of the transmission of scalable video over the Internet. The first goal is to provide high quality video to end users and the second one is to properly design networks and predict network performance for video transmission based on the characteristics of existing video traffic. Rate-distortion (R-D) based schemes are often applied to improve and stabilize video quality; however, the lack of R-D modeling of scalable coders limits their applications in scalable streaming.

Thus, in the first part of this work, we analyze R-D curves of scalable video coders and propose a novel operational R-D model. We evaluate and demonstrate the accuracy of our R-D function in various scalable coders, such as Fine Granular Scalable (FGS) and Progressive FGS coders. Furthermore, due to the time-constraint nature of Internet streaming, we propose another operational R-D model, which is accurate yet with low computational cost, and apply it to streaming applications for quality control purposes.

The Internet is a changing environment; however, most quality control approaches only consider constant bit rate (CBR) channels and no specific studies have been conducted for quality control in variable bit rate (VBR) channels. To fill this void, we examine an asymptotically stable congestion control mechanism and combine it with our R-D model to present smooth visual quality to end users under various network conditions.

Our second focus in this work concerns the modeling and analysis of video traffic, which is crucial to protocol design and efficient network utilization for video transmission. Although scalable video traffic is expected to be an important source for the Internet, we find that little work has been done on analyzing or modeling it. In this regard, we develop a frame-level hybrid framework for modeling multi-layer VBR video traffic. In the proposed framework, the base layer is modeled using a combination of wavelet and time-domain methods and the enhancement layer is linearly predicted from the base layer using the cross-layer correlation.

To my parents

ACKNOWLEDGMENTS

My deepest gratitude and respect first go to my advisors Prof. Andrew Chan and Prof. Dmitri Loguinov. This work would never have been done without their support and guidance.

I would like to thank my co-advisor Prof. Chan for giving me the freedom to choose my research topic and for his continuous support to me during all the ups and downs I went through at Texas A&M University. Furthermore, I cannot help feeling lucky to be able to work with my co-advisor Prof. Loguinov. I am amazed and impressed by his intelligence, creativity, and his serious attitude towards research. Had it not been for his insightful advice, encouragement, and generous support, this work could not have been completed.

I would also like to thank Prof. Karen L. Butler-Purry and Prof. Erchin Serpedin for taking their precious time to serve on my committee.

In addition to my committee members, I benefited greatly from working with Mr. Kourosh Soroushian and the research group members at LSI Logic. It was Mr. Soroushian's projects that first attracted me into this field of video communication. Many thanks to him for his encouragement and support during and even after my internship.

In addition, I would like to take this opportunity to express my sincerest appreciation to my friends and fellow students at Texas A&M University. They provided me with constant support and a balanced and fulfilled life at this university. Zigang Yang, Ge Gao, Beng Lu, Jianhong Jiang, Yu Zhang, and Zhongmin Liu have been with me from the very beginning when I first stepped into the Department of Electrical Engineering. Thanks for their strong faith in my research ability and their encouragement when I need some boost of confidence. I would also like to thank

Jun Zheng, Jianping Hua, Peng Xu, and Cheng Peng, for their general help and the fruitful discussions we had on signal processing. I am especially grateful to Jie Rong, for always being there through all the difficult time.

I sincerely thank my colleagues, Seong-Ryong Kang, Yueping Zhang, Xiaoming Wang, Hsin-Tsang Lee, and Derek Leonard, for making my stay at the Internet Research lab an enjoyable experience. In particular, I would like to thank Hsin-Tsang for his generous provision of office snacks and Seong-Ryong for valuable discussions. I owe special thanks to Yuwen He, my friend far away in China, for his constant encouragement and for being very responsive whenever I called for help.

I cannot express enough of my gratitude to my parents and my sister. Their support and love have always been the source of my strength and the reason I have come this far.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

With the explosive growth of the Internet and rapid advances in compression technology, the transmission of video over the Internet has become a predominant part of video applications. In an ideal case, we only need to optimize video quality at a *given* bit rate provided by networks. Unfortunately, the network channel capacity varies over a wide range, depending on network configurations and conditions. Thus, from the video coding perspective, we need a video coder that optimizes the video quality over a given bit rate *range* instead of a *given* bit rate [65]. These video coders are referred to as scalable coders and have attracted much attention in both industry and academia.

A. Problem Statement

Broadly speaking, the mode for video transmission over the Internet can be classified into download mode and streaming mode [110]. As the phrase suggests, the download mode indicates that the entire video file has to be fully downloaded before playback. In contrast, the streaming mode allows users to play video while only partial content has been received and decoded. The former usually results in long and sometimes unacceptable transfer delays, and thus the latter is more preferred. *Internet streaming* particularly refers to the transmission of *stored* video in the streaming mode.

Internet streaming has certain requirements on bandwidth, packet loss, and packet delay. Unlike general data transmissions, video packets must arrive at the receiver before their playout deadlines. In addition, due to its rich content, Internet

---

The journal model is *IEEE/ACM Transactions on Networking*.

streaming often has a minimum bandwidth requirement to achieve acceptable video quality. Furthermore, packet loss can cause severe degradation of video quality and even cause difficulty in reconstructing other frames.

Subject to these constraints, we will say that the best environment for video streaming is a stable and reliable transmission mechanism that can optimize the video quality under various network conditions. Unfortunately, the current best-effort network provides no Quality of Service (QoS) guarantees to network applications, which means that user packets can be arbitrarily dropped, reordered, and duplicated. In addition, unlike conventional data delivery systems using Transmission Control Protocol (TCP) [85], video communications are usually built on top of User Datagram Protocol (UDP) [84], which does not utilize any congestion control or flow control as TCP [85] does.

Besides these QoS requirements, Internet streaming also has to consider heterogeneity problems, such as *network heterogeneity* and *receiver heterogeneity*. The former means that the subnetworks in the Internet having unevenly distributed resources (e.g., bandwidth) and the latter refers to diverse receiver requirements and processing capability [109].

B. Objective and Approach

To address these challenges, extensive research has been conducted to Internet streaming and scalable coding techniques are introduced to this area due to its strong flexibility to varying network conditions and strong error resilience capability. Generally speaking, scalability refers to the capability of decompressing *subsets* of the compressed data stream in order to satisfy certain constraints [103]. In scalable coding, scalability is typically known as providing multiple versions of a video, in terms of different resolutions (quality, spatial, temporal, and frequency) [107].

Among various studies conducted on scalable coders, rate-distortion (R-D) analysis always attracts considerable attention, due to its importance in a compression/communication system. Although R-D analysis comes under the umbrella of source coding, it is also important in video transmission (e.g., optimal bits allocation [107], constant quality control [114]). Despite numerous previous work on R-D modeling, there are few studies done on the R-D analysis of scalable coders, which limits the applicability of R-D based algorithms in scalable video streaming. Thus, we analyze R-D curves of scalable coders and derive an accurate R-D model that is applicable to network applications.

Notice that in order to provide end users high quality video, it is not sufficient to *only* improve video standards. Instead, we also need to study network characteristics and develop control mechanisms to compensate the deficiencies of best-effort networks. Therefore, we analyze congestion control schemes and combine a stable controller with our proposed R-D model to reduce quality fluctuation during streaming.

Aside from video coding techniques, protocol design and network engineering are also critical to efficient and successful video transmissions. Due to the importance of traffic models to the design of a video-friendly network environment, in the later part of this work, we conduct extensive studies of various video traffic and propose a traffic model that can capture the characteristics of original video sequences and accurately predict network performance.

C. Main Contributions

In general, this work makes the following contributions:

- *Propose a new distribution model to describe the statistical properties of the input to scalable coders.* To derive an R-D bound or model, one needs to first characterize the sources, which is usually a difficult task due to the complexity and diversity of sources [82]. Although there are many statistical models for sources of image/non-scalable coders, there is no specific work done to model sources of scalable coders. Compared with existing models, the proposed model is accurate, mathematically tractable, and with low computational complexity.

- *Give a detailed R-D analysis and propose novel R-D models for scalable video coders.* To better understand scalable coders, we examine distortion and bitrate of scalable coders separately, which have not been done in prior studies. Unlike distortion, which only depends on the statistical properties of the signal, bitrate is also related to the correlation structure of the input signal [38]. Thus, we study bitrate based on the specific coding process of scalable coders. Afterwards, two novel operational R-D models are proposed for scalable coders.

- *Design a quality control scheme applicable to both CBR and VBR channels.* There is no lack of quality control methods, but most of them only consider CBR channels and no effective approach provides constant quality to end users in VBR channels. To deal with the varying network environment, we incorporate our R-D model into a *smooth* congestion control mechanism to achieve constant quality during streaming. With this scheme, the server is able to accurately decide the transmitted bits in the enhancement layer according to the available bandwidth and user requirements. The proposed quality control scheme not only outperforms most existing control algorithms in CBR channels, but is also able to provide constant quality during streaming under varying network conditions.

- *Conduct an extensive study with VBR video sequences coded with various standards and propose a traffic model for multi-layer VBR video traffic.* A good traffic model is important to the analysis and characterization of network traffic and network performance. While multi-layer (scalable) video traffic has become an important source of the Internet, most existing approaches are proposed to model single-layer VBR video traffic and less work has been done on the analysis of multi-layer video traffic. Therefore, we propose a model that is able to capture the statistical properties of both single-layer and multi-layer VBR video traffic. In addition, model accuracy studies are conducted under various network conditions.

D. Dissertation Overview

The structure of this dissertation is shown in Fig. 1. As shown in the figure, throughout this document, we provide background knowledge of scalable coders, and then state current problems and describe the proposed approaches in each topic. Chapter II reviews background knowledge that is important to further discussion in this thesis. Chapters III through V, on the other hand, present the author's own contributions to this field.

In Chapter II, we provide a brief overview of video compression standards and some basics of video coding schemes. In addition, we discuss the importance and advantages of scalable coding in video transmission and also describe several popular scalable coders.

In Chapter III, we give a detailed rate-distortion analysis for scalable coders and also shed new light on the investigation of source statistical features. The objectives of this chapter are not only to propose a novel R-D model for scalable video coders,

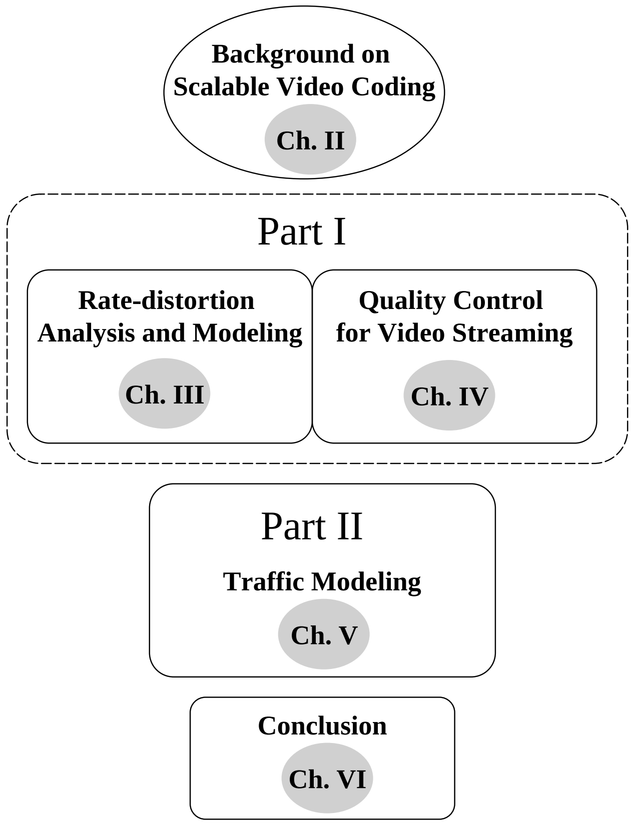

Background on Scalable Video Coding
Ch. II
Part I
Rate-distortion Analysis and Modeling
Ch. III
Quality Control for Video Streaming
Ch. IV
Part II
Traffic Modeling
Ch. V
Conclusion
Ch. VI

Fig. 1. Structure of this proposal.

but also to gain some insight into scalable coding processes.

In Chapter IV, besides providing a short discussion of prior QoS control mechanisms, we present efficient quality control algorithms for Internet streaming in both CBR and VBR channels. Chapter V reviews related work on traffic modeling and proposes a traffic modeling framework, which is able to accurately capture important statistical properties of both single-layer and multi-layer video traffic.

Finally, Chapter VI concludes this work with a summary and some directions for future work.

CHAPTER II

SCALABLE VIDEO CODING

The purpose of this chapter is to provide background knowledge needed for further discussion in this document. In Section A, we review the history of video compression standards and in Section B, we briefly describe the generic building blocks used in recent video compression algorithms. Section C describes the motion compensation algorithms applied in video coders. Finally, in Section D, we discuss several scalable video coding techniques and address their impact on the transmission of video over the Internet.

A. Video Compression Standards

The first international digital video coding standard is H.120 [50], developed by ITU-T (the International Telecommunications Union-Telecommunications) in 1984 and refined in 1988. It includes a conditional replenishment (CR) coder with differential pulse-code modulation (DPCM), scalar quantization, and variable length coding (VLC). The operational bit rate of H.120 is 1544 and 2048 kb/s. Although CR coding can reduce the temporal redundancy in video sequences, it is unable to refine an approximation. In other words, CR coding only allows exact repetition or a complete replacement of each *picture area*. However, it is observed that, in most cases, a refining frame difference approximation is needed to improve compression performance. This concept is called motion-compensated prediction and is first proposed in H.261.

H.261 was first approved by ITU-T in 1990 and revised in 1993 to include a backward-compatible high-resolution graphics transfer mode [51]. H.261 is more popular than H.120 and its target bit rate range is $64 - 2048$ kb/s. H.261 is the *first* standard that develops the basic building blocks that are still used in current video standards. These blocks include motion-compensated prediction, block DCT transform, two-dimensional run-level VLC coding.

In 1991, MPEG-1 was proposed for digital storage media applications (e.g., CD-ROM) and was optimized for noninterlaced video at bitrates from 1.2 Mb/s to 1.5 Mb/s [48]. MPEG-1 gets it acronym from the *Moving Pictures Experts Group* that developed it. MPEG-1 provides better quality than H.261 in high bit rate operations. In terms of technical features, MPEG-1 includes bi-directionally predicted frames (i.e., B-frames) and half-pixel motion prediction.

MPEG-2 was developed as a joint work of both the ISO/IEC and ITU-T organizations and was completed in 1994 [52]. It was designed as a superset of MPEG-1 to support higher bit rates, higher resolutions, scalable coding, and interlaced pictures [52]. Although its original goal is to support interlaced video from conventional television, it is eventually extended to support high-definition television (HDTV) and provides field-based coding and scalability tools. Its primary new technical features include efficient handling of interlaced-scan pictures and hierarchical bit-usage scalability.

H.263 is the first codec specifically designed for *very low bit rate* video [53]. H.263 can code video with the same quality as H.261 but with much less bit rate. The key new technical features of H.263 are variale block-size motion compensation, overlapped-block motion compensation, picture extrapolation motion vectors, three-dimensional VLC coding, and median motion vector prediction.

Unlike MPEG-1/2, H.261/263 are designed for video telephony and only include video coding (no audio coding or systems multiplex). In addition, these standards are primarily intended for conversational applications (i.e., low bit rate and low delay) and thus usually do not support interactivity with stored data [39].

MPEG-4 was designed to address the requirements of a new generation of highly interactive multimedia applications and to provide tools for object-based coding of natural and synthetic audio and video [49]. MPEG-4 includes properties such as object-based coding, synthetic content, and interactivity. The most recent video standard H.264 is capable of providing even higher coding efficiency than MPEG-4. This is a joint work of ITU and MPEG, and it is expected to be a subset of MPEG-4 standard.

In Table I, we list main applications and target bitrate range of these standards in the order of the proposed date.

Table I. A Brief Comparison of Several Video Compression Standards [2].

Standard	Application	Bit Rate
H.261	Video telephony/teleconferencing over ISDN	Multiple of 64 kb/s
MPEG-1	Video on digital storage media (CD-ROM)	1.5 Mb/s
MPEG-2	Digital Television	2-20 Mb/s
H.263	Video telephony over PSTN	$\geq$ 33.6 kb/s
MPEG-4	Object-based coding, synthetic content, interactivity	Variable
H.264	Improved video compression	10's to 100's kb/s

In general, all these video standards are frame-based and block motion-compensated DCT coding. Furthermore, standards only specify bitstream syntax and decoding semantics, which leaves the implementation of encoder and decoder flexible. For example, standards advocate using DCT/IDCT, but do not specify how to implement them. This flexibility enables new encoding and decoding strategies to be employed in a standard-compatible manner.

B. Basics in Video Coding

A video communication system typically includes three parts: compression, transmission, and reconstruction. The encoder compresses raw video into a data stream, the sender retrieves compressed video data from some storage devices and sends data over the network (e.g., the Internet) to the receiver, and the receiver decodes and reconstructs video with the successfully received data.

Recall that a video sequence possesses both spatial correlation and temporal correlation. While the former exists because color value of adjacent pixels in the same video frame usually changes smoothly, the latter happens due to the fact that consecutive frames of a sequence usually show same physical scenes and objects. To reduce the data rate of a video sequence, compression techniques should exploit spatial and temporal correlation.

The current RGB (i.e., red, green, and blue) system is highly correlated and mixes the luminance and chrominance attributes of a light. Since it is often desirable to describe a color in terms of its luminance and chrominance content *separately* for more efficient processing of color signals, a color space conversion is often applied to color signals before compression. In current standards, RGB is often converted into YUV, where Y represents the luminance intensity and (U, V) indicate the chrominance. Since the human visual system (HVS) has lower spatial frequency response and lower sensitivity to (U, V) than to Y, we can sample chrominance with lower frequency and quantize them with larger steps. A popular linear color-space transformation matrix is [2]:

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}. \tag{2.1}$$

1. Compression

Data compression strategies may be classified as either lossless or lossy. While lossless compression can provide a perfect reconstruction of the source, it usually cannot satisfy the high compression requirements of most video applications. Furthermore, HVS can tolerate certain degree of information loss, without interfering with the perception of video sequences. Thus, a lossy compression scheme is often applied in video encoders.

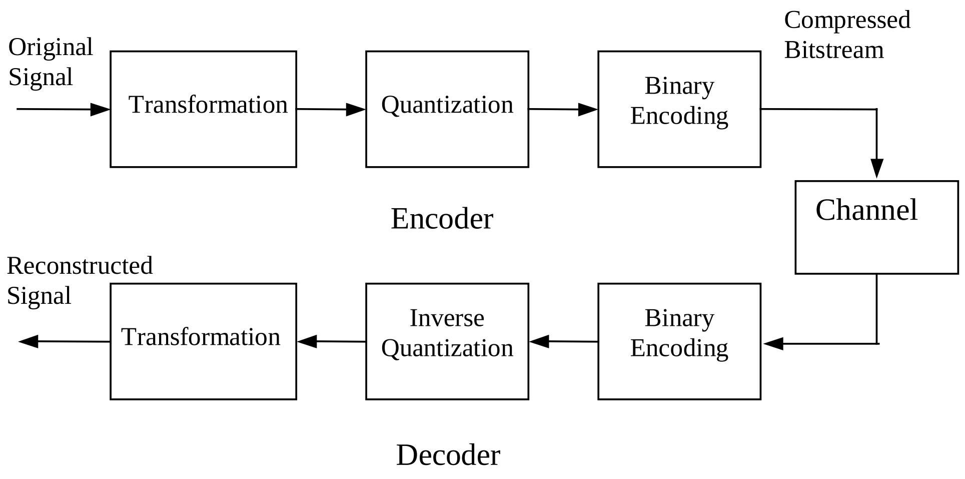


Fig. 2. A generic compression system.

As shown in Fig. 2, a general lossy system includes transformation, quantization/inverse quantization, and binary encoding. Transform coding has been proven to be especially effective for compression of still images and video frames. Aside from reducing spatial correlation between neighboring pixels, transformation can concentrate the energy of these coefficients in certain bands, which makes it possible to further improve compression performance. Another reason for employing transformations in compression algorithms is that they allow the distortion in individual bands to be independently adjusted according to the highly non-uniform frequency response of HVS [103]. Transformations also have advantages for the transmission robustness, in that different degree of protection can be given to different bands of coefficients according to their visual significance.

There are several popular transforms applied in image/video coding schemes, such as Karhunen-Loeve Transform (KLT), Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), and wavelets. KLT can achieve optimal energy compaction, but has high computational cost and requires knowledge of signal covariance. Although wavelet transform provides good energy compaction and better compression performance than DCT for *still* image coding, it does not have a good match for block-matching motion estimation and thus it has not gained acceptance in video coding standards [1]. Due to its low computational complexity and good compaction capability, DCT is widely applied in image and video coding standards. In addition, block-DCT is more suitable than general DCT for video compression, because the former can efficiently cope with both the diversity of image content in video sequences and block-based motion compensation.

2. Quantization and Binary Coding

In a compression system, the transformation and entropy encoding are usually lossless, and the information loss is primarily generated by quantization. Due to the close connection between quantization and coding, we discuss them together in this section.

It is impossible to represent a continuous source with a finite number of bits, and thus quantization is important to produce discrete bit rate representation of visual information. Quantization represents a continuous signal by an approximation chosen

[1]Wavelet transform has been applied in motion JPEG 2000, however, motion JPEG 2000 has different coding process from other video standards, e.g., no motion estimation in JPEG 2000

from a finite set of numbers. The simplest quantization method is scalar quantization, which independently quantizes each sample in a source signal to one of the values in a predesigned reconstruction codebook. Notice that the original source can be either continuous or discrete. The most basic scalar quantizer is a uniform quantizer, which has equal distances between adjacent reconstruction values. To improve the quantization efficiency, minimum mean square error (MMSE) quantizer and optimal scalar quantizers designed using the Lloyd algorithm are introduced.

Rather than quantizing one sample at a time in a scalar quantizer, vector quantization quantizes a group of N samples together, which exploits the underlying redundancy in a correlated input source. In an image, each block of N pixels is considered as a vector to be coded. Given predesigned L patterns, a vector quantizer replaces each block with one of those patterns. The counterpart of uniform quantizers in the vector quantization case is lattice quantizers, in which all the partitioned regions have the same shape and size. Similar to the scalar quantization case, there are optimal quantizers designed with generalized Lloyd algorithm and entropy-constrained optimal quantizers. Despite its efficiency, vector quantization does have a number of drawbacks, such as a large alphabet and a non-trivial algorithm to select optimal symbols from amongst this alphabet. For a comprehensive discussion of vector quantization, the readers are referred to [34].

After obtaining a discrete source (from quantization), binary coding is necessary to represent each possible symbol from a finite alphabet source by a sequence of binary bits, which is often called a *codeword*. The codewords for all possible symbols form a *codebook* or *code*. Notice that a symbol may correspond to one or several quantized values. A useful code should satisfy two properties [107]: (1) it should be uniquely decodable, in other words, there is a one-to-one mapping between the codeword and the symbol; (2) The code should be instantaneously decodable, which requires that no prefix of any codeword is another valid codeword.

The simplest coding strategy is fixed-length coding, which assigns a fixed number of bits to each symbol, e.g., $log_2 L$ bits per symbol for an alphabet of L symbols. In fixed length coding, the code bits corresponding to each symbol are independent. As such, fixed-length coding offers strong error resilience but is relatively inefficient from a compression point of view.

To improve compression performance, variable-length coding (VLC) is introduced into the area of coding. In VLC, the input is sliced into fixed units, while the corresponding output comes in chunks of variable size, e.g., Huffman coding [45]. VLC coding assigns a shorter codeword to a higher probability symbol and achieves lower average bit rate than fixed length coding does. An appropriately designed VLC coder can approach the entropy of the source and thus VLC is also referred to as *entropy coding*.

There are three popular VLC methods: Huffman coding, Lempel-Ziv-Welch-Code (LZW) method, and arithmetic coding. Among them, Huffman coding is the most popular lossless coding approach employed in video coding standards. The idea behind Huffman coding is simply to use shorter bit patterns for symbols with high probability of occurrence and no bit pattern is a prefix of another, which guarantees bit stream uniquely decodable. For instance, suppose that the input alphabet has four characters, with respective occurrence probabilities $P_1 = 0.6, P_2 = 0.3, P_3 = 0.05$, and $P_4 = 0.05$. Then the coded bit patterns are 1, 01, 001, and 000. The average number of bits per symbol is calculated as $\sum l_i P_i = 1.6$, where $l_1 = 1, l_2 = 2, l_3 = l_4 = 3$. Compared with the average number of bits per symbol in fixed-length coding ($\log_2 4 = 2$), Huffman coding has higher compression ratio. However, when Huffman coding applies to individual samples, at least one bit must be assigned for each sample. To further reduce the bit rate, vector Huffman coding is introduced, which gives each group of N samples a codeword. Besides vector Huffman coding, there is another variation of Huffman coding, conditional Huffman coding, which uses different code depending on the symbols taken by previous samples.

A disadvantage of Huffman coding is that each sample or each group of N samples uses at least one bit and thus cannot closely approach the entropy bound unless N is large enough. To overcome this problem, arithmetic coding is proposed to convert a variable number of samples into a variable-length codeword, which allows average coding rate less than one bit per symbol [91]. The idea behind arithmetic coding is to represent a sequence of symbols by an interval in a line ranging from zero to one, with interval length equal to the probability of the symbol sequence. Instead of coding the entire sequence at one time, an arithmetic coder starts from an initial interval determined according to the first symbol and then recursively divides the previous interval after each new symbol joins the sequence. Arithmetic coding is highly susceptible to errors in the bit stream and is more computationally demanding than Huffman coding.

In a word, Huffman coding converts a *fixed* number of symbols into a variable-length codeword, LZW coding converts a *variable* number of symbols into a fixed-length codeword, and arithmetic coding converts a *variable* number of symbols into a variable-length codeword. Furthermore, Huffman coding and arithmetic coding are probability-based methods and both can reach the entropy bound asymptotically. The LZW coding does not require knowledge of source statistics but is less efficient and less commonly used than the other two coding methods.

Since transformations produce many zero symbols and high frequency subband coefficients with zero mean and small variance, *zero-coding* is introduced to exploit statistical dependence between transformation coefficients. Among various zero-coding techniques, run-length coding is commonly applied to video standards. Run-length coding codes the locations of zero symbols via white and black runs, representing the lengths of contiguous non-zero symbols and contiguous zero symbols, respectively [103]. The DC coefficient and absolute value of white and black runs may be coded with other coding techniques (e.g., Huffman). Before quantization, the transformation coefficients are scanned into an one-dimensional signal and thus the scanning order is very important to efficient coding. The zigzag scanning in Fig. 3 is often applied for its good compression performance.

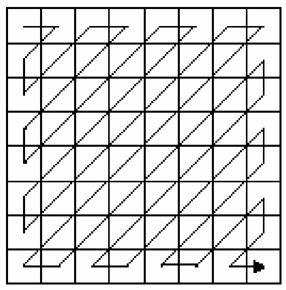


Fig. 3. Zigzag scan order.

C. Motion Compensation

A video sequence is simply a series of pictures taken at closely spaced intervals in time [77]. Except for a scene change, these pictures tend to be quite similar from one to the next, which is considered as temporal redundancy. Thus, video can be represented more efficiently by coding only the changes in video content. Essentially, video compression distinguishes itself from still-image compression with its ability to use temporal redundancy to improve coding efficiency.

The technique that uses information from other pictures in the sequence to predict the current frame is known as inter-frame coding. Frames that are coded based on the previously coded frame are called P-frames (i.e., predictively coded frame)and those that are coded based on both previous and future coded frames are called B-frames (i.e., bi-directionally predicted frames).

When a scene change occurs (and sometimes for other reasons), inter-frame coding does not work and the current frame has to be coded independently of all other frames. These independently coded frames are referred to as I-frames (i.e, intra-coded frame). With I-frames, a video sequence can be divided into many groups of pictures (GOP). As shown in Fig. 4, a GOP is composed of one I-frame and several P/B-frames.

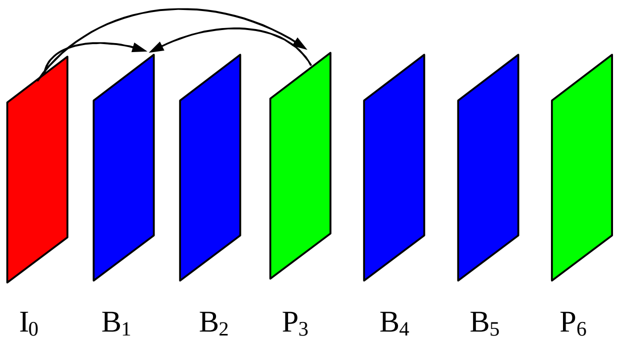


Fig. 4. A typical group of picture (GOP). Arrows represent prediction direction.

As we mentioned earlier, there is a simple method called conditional replenishment (CR), which codes only the changes between frames. In CR coding, an area of current frame is either repeated as that in the previous frame (SKIP mode) or totally re-coded (INTRA mode). However, the current frame is often *slightly* different from the previous one, which does not fit the SKIP mode and is quite inefficient if using the INTRA mode.

An alternative approach proposed for exploiting temporal correlation is *motion-compensated prediction* (MCP). As shown in Fig. 5, an encoder codes the difference between current frame and the prediction from reference frame, which is considered as *motion vector* due to the fact that it is often caused by motion. Using motion vectors and the reference frame, the encoder generates an approximation of the current frame and the residual between the approximation and the original data is coded. In a decoder, motion vector and the residual is decoded and added back to the reference frame to reconstruct the target frame.

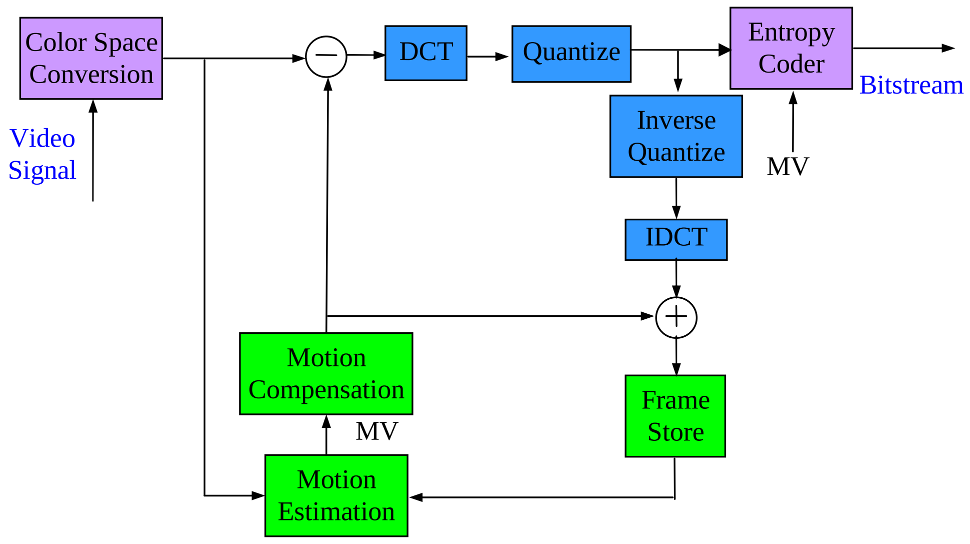


Fig. 5. The structure of a typical encoder.

As for MCP, there is one important step that can not be missing, which is the encoder's search for the best motion vectors, known as *motion estimation* (ME). Ideally, ME partitions video into moving objects and describe their motion. Since identifying objects is generally difficult to implement, a practical approach, the block-matching motion estimation, is often used in encoders.

In the block-matching ME, assuming that all pixels within each block have the same motion, the encoder partitions each frame into non-overlapping $N_1 \times N_2$ blocks (e.g., 16×16) and finds the best matching block in the reference frame for each block, as shown in Fig. 6. The main technical issues related to motion vectors are the precision of the motion vectors, the size of the block, and the criteria used to select the best motion vector value. In general, motion vectors are chosen so that they either maximize correlation or minimize error between a current macroblock and a corresponding one in the reference picture. As correlation calculations are computationally expensive, error measures such as mean square error (MSE) and mean absolute distortion (MAD) are commonly used to choose the best motion vectors.

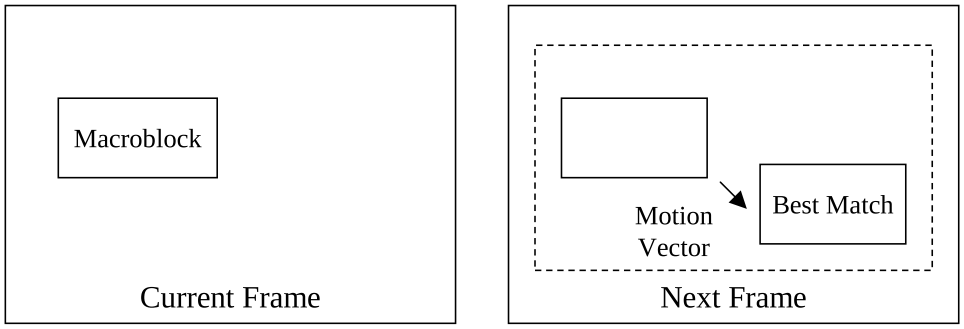


Fig. 6. Best-matching search in motion estimation.

In a straightforward MSE motion estimation, the encoder tests all possible integer values of a motion vector with a range. Given a $\pm L$ range, the complexity of "full-search" ME requires approximately $3(2L+1)^2$ operations per pixel and that of some fast search techniques is proportional to L [77], [102]. From extensive experiments, it is found that $L = 8$ is marginally adequate and $L = 16$ is probably sufficient for most sequences. The smaller the value of L, the higher precision ME can achieve and the higher computational complexity it needs.

Besides the search range, the precision of motion vector is also important to ME. Although video is only known at discrete pixel locations, motion is not limited to integer-pixel offsets. To estimate sub-pixel motion, frames must be spatially interpolated. Therefore, fractional motion vectors are used to represent the sub-pixel motion, e.g., half-pixel ME is used in MPEG-1/2/4. Although sub-pixel ME introduces extra computational complexity, it can capture half-pixel motion and thus improves ME performance. In addition, the average effect resulted from the spatial interpolation in sub-pixel ME diminishes noise in noisy sequences, reduces prediction error, and thus improves compression efficiency.

After obtaining motion vectors, the MCP algorithm predicts the current frame based on reference frame(s) while compensating for the motion. This MCP algorithm estimates a block in the current frame from a corresponding block of the previous frame (P-frame) or together with that of the next frame (B-frame). In B-frame, a block in the current frame is estimated by taking the average of a block from the previous frame and a block from the future frame.

In general, block matching schemes applied in ME and MCP provides good, robust performance for video compression. Both algorithms are not difficult to represent and are periodically applied in the encoding process (one MV per block), which makes its implementation feasible on hardware. However, this scheme only assumes translational motion model and no complex motion is considered.

D. Scalable Video Coding

In an ideal video streaming system, the available network bandwidth is stable, the encoder optimally compresses the raw video at a given bandwidth, and the decoder is able to decode all the received bits. However, the bandwidth is varying in the real network and thus the encoder should optimize the video quality over a given *range* of bitrate instead of *one* specific bitrate. In addition, due to the time-constraint nature of video streaming, the decoder cannot use packets that are later than their playback deadline. However, video packets can be arbitrarily dropped and delayed in current best-effort network.

To deal with these problems, scalable coding is widely applied in video streaming applications. Scalable coding techniques can be classified into coarse granularity (e.g., spatial scalability) and fine granularity (e.g., fine granular scalability (FGS)) [107]. In both coarse and fine granular coding methods, each lower priority layer (e.g., higher-level enhancement layer) is coded with the residual between the original image and the reconstructed image from the higher priority layers (e.g., base layer or lower-level enhancement layer). The major difference between coarse granularity and fine granularity is that the former provides quality improvements only when a *complete* enhancement layer has been received, while the latter continuously improves video quality with every additionally received codeword of the enhancement layer bitstream.

1. Coarse Granular Scalability

The coarse granular scalability includes spatial scalability, temporal scalability, and SNR/quality scalability. There is also a term called frequency scalability, which indicates a form of spatial resolution scalability provided by dropping high frequency DCT coefficients during reconstruction.

a. Spatial Scalability

Spatial scalability was first offered by MPEG-2 for the purposes of compatibility between interlaced and progressively scanned video sequence formats. Spatial scalability represents the same video in varying spatial resolutions. To generate the base layer with a lower spatial resolution, the raw video is spatially down-sampled, DCT-transformed, and quantized. The base layer image is reconstructed, up-sampled, and used as a prediction for the enhancement layer. Afterwards, the residual between the prediction and the original image is DCT-transformed, quantized, and coded into the enhancement layer. Fig. 7 shows an example of transmitting a spatially scalable coded bitstream over the Internet.

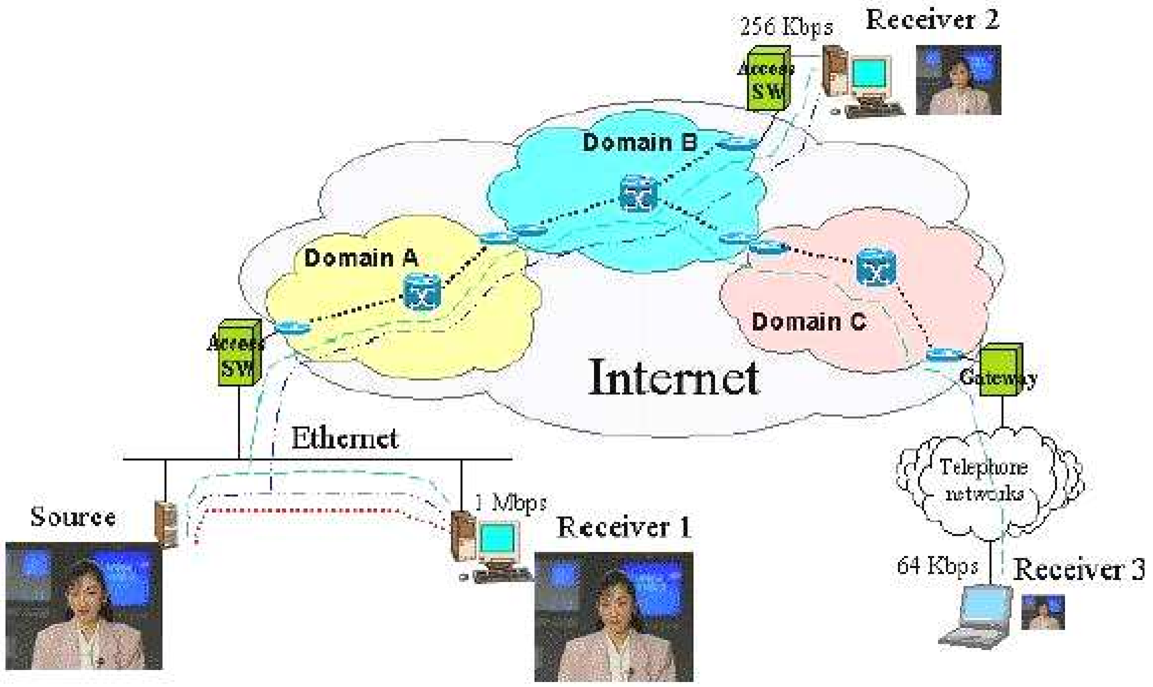


Fig. 7. The transmission of a spatially scalable coded bitstream over the Internet. Source: [109].

b. Temporal Scalability

Temporal scalability represents the same video in various frame rates. The encoder codes the base layer at a lower frame rate and makes use of the temporally up-sampled pictures from a lower layer as a prediction in a higher layer. The simplest way of temporal up-sampling and down-sampling is frame copying and frame skipping.

The coding processes of the spatial and temporal scalability are similar, except that there is spatial up/down-sampling in spatial scalability and temporal up/down-sampling in temporal scalability. As an example, the structure of a two-level spatially/temporally scalable codec is shown in Figure 8.

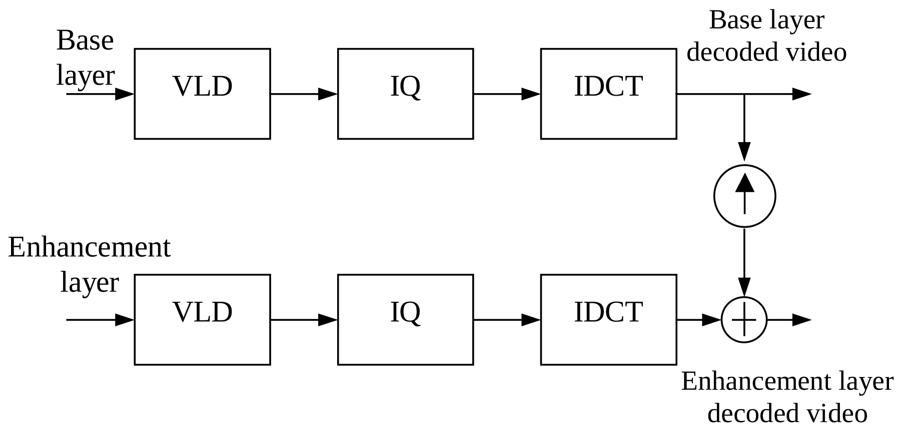


Fig. 8. A two-level spatially/temporally scalable decoder. Source: [107].

c. SNR/Quality Scalability

Quality scalability refers to a mechanism for achieving different quality by successive refinement in the quantization of DCT coefficients. The encoder codes the base layer with a coarse quantizer and an enhancement layer with a finer quantizer. Since different quantization accuracies lead to different PSNRs between the original video and the one reconstructed from different layers, quality scalability is also known as SNR scalability [107].

2. Fine Granular Scalability

Fine granular scalability includes subband/wavelet coding and FGS coding [107]. As addressed in Section 1, subband/wavelet coding has difficulty in block-matching motion estimation and often leads to delay due to its hierarchy structure. Instead, FGS is widely applied in scalable coders and has been accepted in MPEG-4 streaming profile, due to its flexibility and strong error resilience ability [86]. Specifically, FGS coding technique has the following advantages [96]:

- It enables a streaming server to perform minimal real-time processing and rate control;

- It is highly adaptable to unpredictable bandwidth variations, due to receiver heterogeneity (e.g., heterogeneous access-technologies) and network heterogeneity (e.g., congestion events);
- It allows low-complexity decoding and low-memory requirements to provide less-powerful receivers the opportunity to stream and decode Internet video content;
- It supports both multicast and unicast applications;
- FGS-coded bitstreams have strong error-resilient ability.

A limitation with FGS scheme is that it has coding penalty for sequences with high-temporal correlation. To reduce prediction error and thus improve coding efficiency, progressive FGS (PFGS) coding is proposed. The essential difference between them is that FGS only uses the base layer for motion prediction and PFGS also uses part of the enhancement layer as a reference for motion-compensated prediction [110].

CHAPTER III

RATE-DISTORTION ANALYSIS FOR SCALABLE CODERS

In this chapter, we review previous research conducted in the area of rate-distortion (R-D) modeling and derive novel R-D functions for scalable coders. We observe that prior studies are more concerned with theoretical depth instead of practical applicability and usually target at images or non-scalable video coders. Therefore, our objective is not only to impart a conceptual understanding of various theory in obtaining R-D functions, but also to derive R-D functions that are applicable to real video applications.

We start this chapter by giving the motivation of R-D analysis and modeling and some preliminary knowledge in Section A and Section B, respectively. Due to the importance of source properties in R-D modeling, we discuss the source statistical properties of scalable coders and propose a novel model to describe source distribution in Section C. In Section D, we review current R-D functions for image and non-scalable coders.

Section E states current difficulties in modeling distortion of scalable coders and derives a distortion model from different perspectives. By contrast, Section F focuses on the rate analysis for scalable coders. With the proposed distortion model and an existing ρ-domain rate model, we show that distortion D is a function of rate R and its logarithm $\log(R)$ in Section G. However, by considering the time-constraint nature of Internet streaming applications, we propose another operational R-D model, which is accurate but has simpler format than the previous one, in Section H.

A. Motivation

R-D curves are useful in both source coding and Internet video streaming. While it is well-known that R-D based compression approaches can adaptively select quantization steps and maximize video quality under given buffer constraints [20], [65], R-D curves can also be used during streaming rate-control to optimally allocate bits in joint source-channel coding [11], [43], to avoid network congestion [13], [70], and to achieve constant quality [105], [113], [114].

Due to its importance, R-D modeling has attracted great research interest for over 50 years [82]. On the one hand, R-D modeling is undertaken either *empirically* or *analytically*, each of which has its own benefits and drawbacks. An empirical approach obtains R-D curves by interpolating between (R, D) samples of a given encoder [79] and an analytical approach derives R-D models based on rate-distortion or quantization theory with certain assumptions of source statistical properties [22], [38], [54]. While an empirical approach usually results in better estimation of actual R-D curves, it fundamentally lacks theoretical insight into the structure of the coding system. By contrast, an analytical approach is usually not tight enough for practical applications.

Actually, accurate modeling of R-D curves of real encoders is always challenging due to the diversity of source images and the complexity of transmission channels [82]. Therefore, a third type of R-D models, the *operational* approach, is widely used in practice [13], [38], [40]. An operational R-D model expresses the basic structure of R-D curves in a closed-form formula, but then parameterizes the equation according to several parameters sampled from the actual system (e.g., [13], [38], [40]).

On the other hand, R-D models can be classified into two categories according to the theory they apply: models based on Shannon's rate-distortion theory [17] and those derived from high-rate quantization theory [4]. The former assumes that sources are coded using very long (infinite) blocks, while the latter assumes that the encoding rate is arbitrarily high [82]. These two theories are complementary and, as shown in [33], converge to the same lower bound $D \sim e^{-\alpha R}$ when the input block size goes to infinity.

Since block length cannot be infinite in real coding systems, it is commonly recognized that classical rate-distortion theory is often not suitable for accurate modeling of actual R-D curves [82]. Subject to these considerations, Mallat *et al.* [74] propose an R-D function for transform-based low bitrate images based on approximation theory. In this R-D model, distortion D is proportional to the reciprocal of bitrate R.

However, although there are numerous applications of R-D modeling in scalable Internet streaming [105], [109], [113], [114], the majority of current R-D models are built for images or non-scalable video coders [17], [54]. In addition, in scalable streaming applications, the server is often concerned with the bitrate R of the *enhancement* layer where R varies from very low bitrate (e.g., less than 0.5 bit/pixel) to high bitrate (e.g., 4 bits/pixel) depending on streaming conditions [107], [114], which means that the high-bitrate assumption of the quantization theory no longer holds [15], [74]. To overcome this gap in current knowledge of scalable coding R-D systems and provide future video streaming applications with accurate R-D models, this work derives two operational R-D models based on statistical properties of scalable sources and existing bitrate models [40].

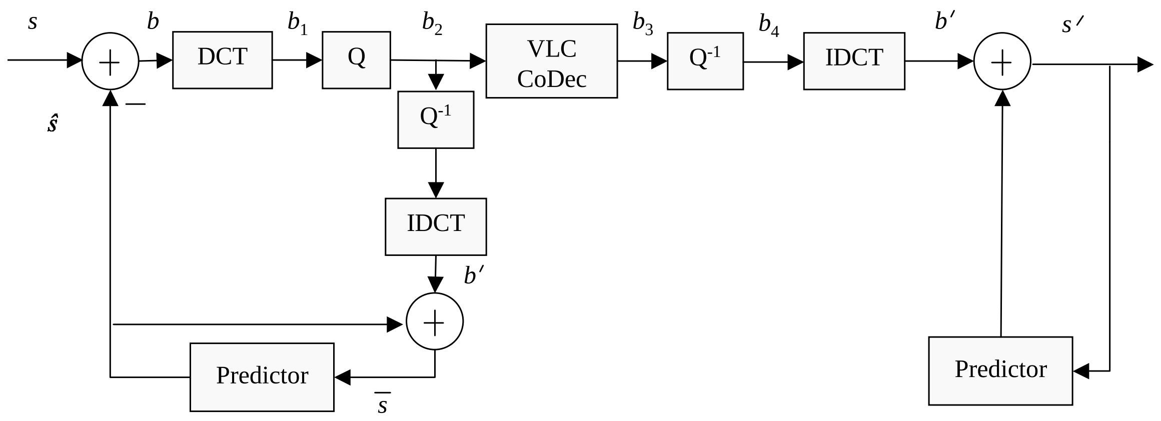


Fig. 9. Basic structure of a MCP coder.

B. Preliminaries

Although developing the appropriate measures for the distortion remains an open issue, the most commonly used measure of distortion is the Mean Squared Error (MSE) between the reconstructed image and the original one. For simplicity and consistency with the literature, video quality measure in this work is quoted in terms of Peak Signal to Noise Ratio (PSNR), derived from MSE as $PSNR = 10\log_{10}\frac{255^2}{MSE}$ for 8 bit source data.

In what follows, we investigate the coding process and the distortion of a typical scalable coder.

1. Brief R-D Analysis for MCP Coders

Motion-compensated prediction (MCP) is the key part in video coders and we start this section with a discussion of the basic structure of motion-compensated predictive coding. As shown in Fig. 9, the original signal is s and the estimated signal by motion compensation is $\hat{s}$. The prediction error $b = s - \hat{s}$ is coded and transmitted through the transmission channel, and the reconstructed signal from the decoder is

s'. Compared with coding the original signal s, coding the prediction error b is more efficient, with the prediction gain:

$$G_p = \frac{\sigma_s^2}{\sigma_b^2}, \tag{3.1}$$

where σ_s^2 and σ_b^2 are the variance of s and b.

We next examine the relationship between coding distortion $D = s - s'$ and other distortions shown in the figure.

Lemma 1 *Assuming that there is no transmission error, coding distortion D is equal to the quantization error between b_1 and b_2.*

Proof: Since the entropy codec (coder and decoder) is lossless and there is no transmission error, we have

$$b_2 = b_3 = b_4. \tag{3.2}$$

Further recall that DCT transformation is ideally lossless and thus

$$b = b_1 \quad \text{and} \quad b' = b_4. \tag{3.3}$$

Therefore, we have $b' = b_4 = b_2$. Furthermore, since the encoder and the decoder use the same reference image, it is obvious that $s = b + \hat{s}$ and $s' = b' + \hat{s}$, as shown in Fig. 9.

Considering all the above discussions and assuming that the distortion between the original signal s and its reconstructed signal s' is calculated in the MSE form as $E(s - s')^2$, we have the following derivation:

$$\begin{aligned} D &= E[(s - s')^2] &&= E[(b + \hat{s} - (b' + \hat{s}))^2] \\ &= E[(b - b')^2] &&= E[(b_1 - b_2)^2]. \end{aligned} \tag{3.4}$$

Therefore, the distortion of a coding system is equal to the quantization error. ■

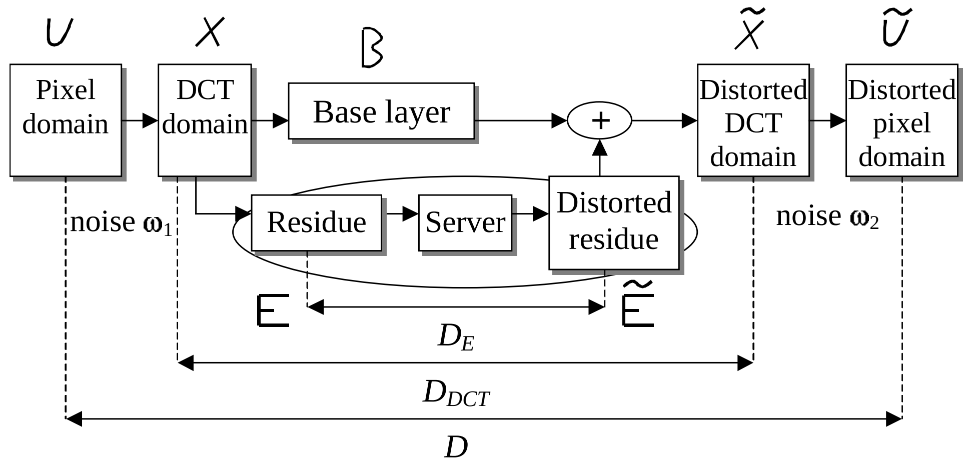


Fig. 10. Different levels of distortion in a typical scalable model.

2. Brief R-D Analysis for Scalable Coders

In a scalable coder, the input to the enhancement layer E is the difference between the original video S and the reconstructed signal from the base layer B', in other terms, $S = E + B'$. Since the reconstructed video of a scalable coder S' is equal to $B' + E'$, the total distortion $D = E[(S-S')^2] = E[(E+B'-(E'+B'))^2] = E[(E-E')^2]$, which means estimating the distortion of the enhancement layer is *sufficient* for analyzing the distortion of a scalable coder. Notice that this result also holds if the encoder computes the residue in the DCT domain. To better understand this scenario, we illustrate the coding process of FGS coders in Fig. 10.

As shown in the figure, signal U in the spatial domain is transformed (with some round-off errors ω_1) into signal X in the DCT domain. Signal X is separated into the base layer B and the enhancement layer E by the encoder (i.e., $B + E = X$). The enhancement layer contains the residual signal, which is necessary to reconstruct the original image from the coded base layer B. During the streaming, the server transmits certain portion of the enhancement layer to the receiver according to user requirements or available bandwidth. Then the residual signal E becomes $\tilde{E}$ and is then added to the based layer at the receiver to produce $\tilde{X} = B + \tilde{E}$ in the DCT domain. Finally, $\tilde{X}$ is converted into the spatial domain (with additional round-off errors ω_2) to become $\tilde{U}$, which is displayed to the user.

In this coding process, there are three levels of distortion: spatial-domain distortion $D = E[(U - \tilde{U})^2]$, DCT-domain distortion $D_{DCT} = E[(X - \tilde{X})^2]$, and the enhancement layer distortion $D_E = E[(E - \tilde{E})^2]$. Notice that D_E is decided by the portion of the enhancement layer the server chooses to transmit during streaming. While spatial-domain distortion D and DCT-domain distortion D_{DCT} are equal, we examine the relationship between D_{DCT} and D_E. Also note that DCT/IDCT round-off noises ω_1 and ω_2 shown in the figure are commonly assumed to be insignificant (which is true except in very high bitrate[1] cases) and are often neglected in R-D modeling. Recall that $\tilde{X} = B + \tilde{E}$, we have

$$D_{DCT} = E[(X - \tilde{X})^2] = E[(X - (B + \tilde{E}))^2] = E[(E - \tilde{E})^2] = D_E. \tag{3.5}$$

As shown in (3.5), all three distortions are equal and we now have a foundation for modeling the enhancement layer distortion D_E as a function of enhancement rate R_E (neither of which requires any information from the base layer). The big advantage of using D_E instead of D is that the statistical properties of DCT residue/coefficients are more mathematically tractable than those of the original signal.

C. Source Analysis and Modeling

A source model that can accurately capture its statistical properties is a pre-request of deriving an effective R-D model. Although there is no lack of source models for image/video coders, a mathematically tractable source model for *scalable* coders is

[1]We use terms bitrate and rate interchangeably throughout this work.

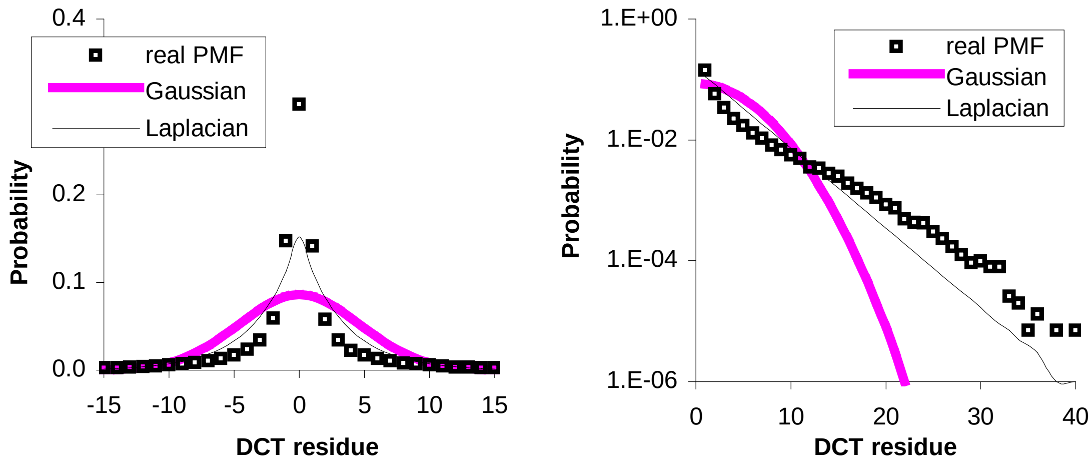
still in demand. As we stated earlier, the input to the enhancement layer in a scalable encoder is the DCT residue between the original image and the reconstructed image in the base layer [78]. As a result, we model the distribution of DCT residue during the journey to obtain an R-D model of scalable coders.

1. Related Work on Source Statistics

It has been a long-standing problem to determine statistical properties of DCT coefficients, due to the fact that DCT transformation has been widely applied to image/video coding methods. Some well-known models for DCT coefficients include the Gaussian [79], the Laplacian [99], and the Generalized Gaussian distribution (GGD) [79]. However, these models are popular more due to their mathematical tractability rather than their performance at describing real video source data. To examine statistical properties of real DCT residue, we conducted an extensive analysis of the probability mass function (PMF) of DCT residue coefficients for different frames and different sequences.

Fig. 11 gives a typical example of the PMF of DCT residue and the estimate of Gaussian and Laplacian models. Fig. 11 (a) demonstrates that the signal is in fact zero-mean; however, neither Gaussian, nor Laplacian distributions fit the center peak. Note that it is important to model the peak of the distribution of embedded visual signals since it often contains a large fraction of the coefficients (in FGS, usually 20% or more). Fig. 11 (b) shows that the logarithmic scale of the positive tails of the actual PMF and different estimates. It is observed that the Gaussian tail decays too quickly and that the Laplacian distribution cannot model both the peak and the tail simultaneously.

Notice that in applications where even higher accuracy is required, the GGD is sometimes used to model source data [79]. Recall that the GGD is given by its



(a) (b)

Fig. 11. (a) The PMF of DCT residue with Gaussian and Laplacian estimation. (b) Logarithmic scale of the PMFs for the positive residue.

density function:

$$f(x) = \frac{\alpha\nu}{2\Gamma(1/\nu)} e^{-|\alpha x|^{v}}, \tag{3.6}$$

where $\Gamma(.)$ denotes the gamma function, ν is the shape parameter,

$$\alpha = \frac{1}{\sigma_x}\sqrt{\frac{\Gamma(3/\nu)}{\Gamma(1/\nu)}}, \tag{3.7}$$

and σ_x is the standard deviation of the source. For $v = 1$, the GGD becomes a Laplacian distribution and for $v = 2$, it becomes a Gaussian distribution.

However, due to its complexity, the GGD does not generally present an analytically appealing alternative to simpler methods. In addition, the statistical properties of DCT residue are different from those of DCT coefficients and thus a direct application of the above models to DCT residue might be inaccurate. In this scenarios, we need a model of DCT residue that is more accurate than Gaussian and Laplacian distributions yet significantly simpler than GGD.

2. Proposed Model for Source Distribution

Yovanof *et al.* [111] point out that a single model is usually insufficient to describe statistical properties of complex sources. Eude *et al.* [25] show that a linear mixture of several distributions offers more degrees of freedom and fits actual samples better. Smoot *et al.* [99] also mention that the mixture model achieves higher accuracy than a single distribution in modeling DCT coefficients.

We can also obtain the same conclusion by examining the tail of the PMF on a log scale in Fig. 11 (b). It is clearly shown that the shape of the actual tail resembles two straight lines (each of which is an exponential function on a log scale). Similar observations hold for other frames and sequences (not shown here). Building upon these observations and on previously suggested methods for non-scalable DCT modeling [25], we notice that a linear *mixture* model of two Laplacian distributions might be a good match.

Motivated by the observation and suggestions from previous work, we propose a mixture Laplacian model, which is defined as follows. Consider that DCT residue is a random variable drawn from two different distributions. Then, the residue is selected from the low-variance Laplacian component with probability p and from the high-variance component with probability $1-p$. Then the density of Xcan be written as:

$$p(x) = p\frac{\lambda_0}{2}e^{-\lambda_0|x|} + (1-p)\frac{\lambda_1}{2}e^{-\lambda_1|x|}, \tag{3.8}$$

where p is the probability to obtain a sample from the low-variance model, and λ_0 and λ_1 are the shape parameters of the corresponding Laplacian distributions. The parameters of (3.8) can be optimally estimated using a variety of methods, including the Expectation-Maximization (EM) algorithm [6] used in this work. We next examine the accuracy of this model in real sequences.

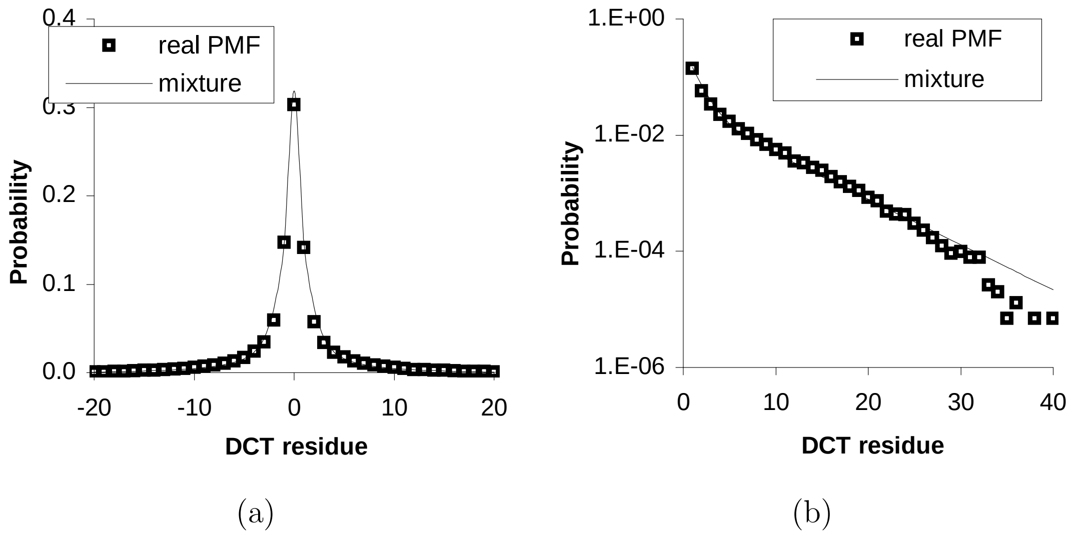


Fig. 12. (a) The real PMF and the mixture Laplacian model. (b) Tails on logarithmic scale of mixture Laplacian and the real PMF.

Fig. 12 demonstrates that (3.8) models the same frame 0 of the CIF Foreman sequence with more accuracy than the traditional Gaussian/Laplacian models. As illustrated in the figure, the mixture Laplacian distribution fits the histogram of the DCT residue much better. The discrepancy at the end of the tail in Fig. 12 (b) does not affect the source model, since only very few of the samples are contained there (0.04% in this example). It should be pointed out that the mixture Laplacian distribution can also describe statistical properties of other signals with sharp peaks and heavy tails , such as base-layer DCT coefficients.

In fact, the fit of the mixture model was even better than that of the GGD in all test sequences. We show this result for Foreman and Carphone using the χ^2 statistic in Table II for 10 and 20 bins utilized in the computation of χ^2. In both cases, the table shows that the mixture model produces much smaller errors χ^2 than any of the other models.

After obtaining an accurate statistical model, we next briefly overview the related

Table II. The Average Values of χ^2 in Test Sequences.

	Bins	Gaussian	Laplacian	GGD	Mixture
Carphone	10	8.2×10^{22}	6.9×10^4	5,756	3,072
Foreman	10	1.3×10^{15}	6.6×10^4	3,437	1,939
Carphone	20	4.6×10^{26}	8.5×10^4	9,160	5,373
Foreman	20	2.5×10^{18}	7.9×10^4	5,735	3,916

work on R-D modeling and analyze the applicability of current R-D models to scalable coders.

D. Related Work on Rate-Distortion Modeling

This subsection includes a theoretical analysis of the R-D function for a generic motion-compensating hybrid coder and an overview of related work in R-D modeling. In subsection 3, we state current problems in R-D modeling. We also evaluate the accuracy of the classical R-D function in video coders and find it no longer applicable to scalable video coders.

1. R-D Functions of MCP Coders

Fig. 13 gives a simplified structure of a generic MCP video coder. Assuming that the input signal $s(x, y; t)$ is stationary, we simulate the transformation between encoder input e and decoder output e' with filter $g(x, y; t)$ plus a temporally uncorrelated noise n.

Since the R-D function itself is known in closed form only for Gaussian sources [82] and due to the central limit theorem, we assume that prediction error e is a stationary, jointly Gaussian, zero-mean signal with PSD function $S_{ee}(\Omega_x, \Omega_y)$. Thus,

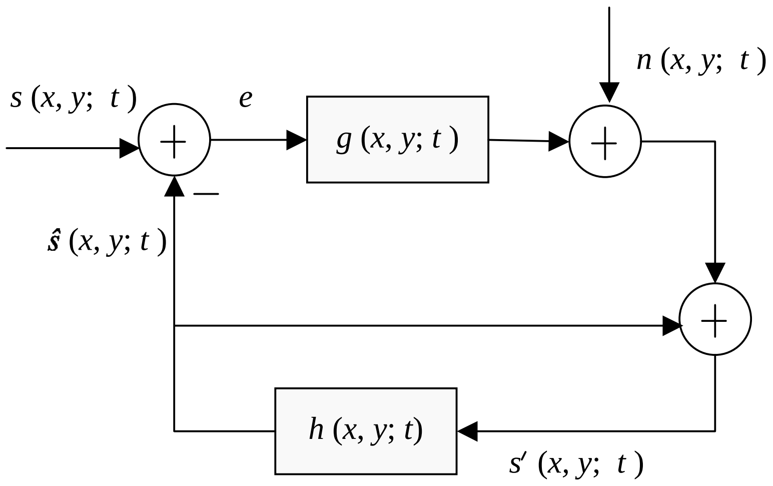


Fig. 13. Generic structure of a coder with linear temporal prediction.

the distortion function is:

$$D(\Theta) = \frac{1}{4\pi^2} \int_{\omega_x}\int_{\omega_y} min[\Theta, S_{ee}(\omega_x, \omega_y)] \mathrm{d}\omega_x \omega_y, \tag{3.9}$$

and the minimum transmission rate that can be achieved is:

$$R(\Theta) = \frac{1}{8\pi^2} \int_{\omega_x}\int_{\omega_y} max \left[0, log_2 \frac{S_{ee}(\omega_x, \omega_y)}{\Theta}\right] \mathrm{d}\omega_x \omega_y, \tag{3.10}$$

where Θ is a parameter that generates the function $R(D)$ by taking on positive real values.

To achieve the lowest transmission rate (3.10), the transfer function $G(\Lambda)$ in the spatial domain is [5] :

$$G(\Lambda) = max \left[0, 1 - \frac{\Theta}{S_{ee}(\Lambda)}\right], \tag{3.11}$$

where $\Lambda = (\omega_x, \omega_y)$, and the noise n has PSD function:

$$S_{nn}(\Lambda) = max \left[0, \Theta \left(1 - \frac{\Theta}{S_{ee}(\Lambda)}\right)\right]. \tag{3.12}$$

Notice that video coders also consider the temporal frequency. We next show the relationship between the PSD function of 2-D image and that of 3-D video. We define $\Omega = (\omega_x, \omega_y, \omega_t)$ as the spatial-temporal frequency vector and $\Lambda = (\omega_x, \omega_y)$ the spatial frequency.

As shown in Fig. 13, the motion-compensation predictor calculates a prediction value $\hat{s}$ at time instant t by a linear combination of reconstructed signal s' at time constant $(t - \Delta t)$ and motion vector $(\hat{d}_x, \hat{d}_y)$. The prediction can be simulated by a filtering process:

$$\hat{s}(x, y, t) = h(x, y, t) * s'(x, y, t). \tag{3.13}$$

From Fig. 13, we can also derive the Fourier transform of e as:

$$E(\Omega) = \frac{[1 - H(\Omega)]S(\Omega) - H(\Omega)N(\Omega)}{1 - H(\Omega) + H(\Omega)G(\Omega)}, \tag{3.14}$$

where $H(\Omega)$ is the Fourier transform of $h(x, y; t)$ and is statistically independent from both $S(\Omega)$ and $N(\Omega)$. Thus, the 3-D PSD function of the prediction error e is [35]:

$$\begin{aligned} S_{ee}(\Omega) = E&\left[\left|\frac{1 - H(\Omega)}{1 - H(\Omega) + H(\Omega)G(\Omega)}\right|^2\right] S_{ss}(\Omega) \\ + E&\left[\left|\frac{H(\Omega)}{1 - H(\Omega) + H(\Omega)G(\Omega)}\right|^2\right] S_{nn}(\Omega), \end{aligned} \tag{3.15}$$

where $E[\cdot]$ is the expected value function, and $S_{ss}(\Omega)$ and $S_{nn}(\Omega)$ denote the 3-D PSD of the original signal s and noise n, respectively. Assume that e is time discrete with a temporal sampling interval $\Delta_t = \frac{2\pi}{\omega_t}$. Then, function $S_{ee}(\Omega)$ is periodic in ω_t and the 2-D PSD function of e:

$$S_{ee}(\Lambda) = \frac{\Delta_t}{2\pi} \int_{\omega_t=0}^{\frac{2\pi}{\Delta_t}} S_{ss}(\Omega) \mathrm{d}\omega_t. \tag{3.16}$$

After combining (3.11), (3.12),(3.15), and (3.16), we obtain the PSD of e as:

$$\begin{aligned}S_{ee}(\Lambda) = &\frac{\Delta_t}{2\pi}\int_0^{\frac{2\pi}{\Delta_t}} E\left[\left|\frac{1-H(\Omega)}{1-H(\Omega)min\left[1,\frac{\Theta}{S_{ee}(\Lambda)}\right]}\right|^2\right] S_{ss}(\Omega)\mathrm{d}\omega_t\\ &+\frac{\Delta_t\Theta}{2\pi}max\left[0,1-\frac{\Theta}{S_{ee}(\Lambda)}\right]\int_0^{\frac{2\pi}{\Delta_t}} E\left[\left|\frac{H(\Omega)}{1-H(\Omega)min\left[1,\frac{\Theta}{S_{ee}(\Lambda)}\right]}\right|^2\right] S_{nn}(\Omega)\mathrm{d}\omega_t.\end{aligned} \tag{3.17}$$

Notice that the PSD of noise n is no larger than Θ (obtained from (3.12)). Therefore, if $S_{ee}(\Lambda)$ is much greater than Θ, we have:

$$\frac{\Theta}{S_{ee}(\Lambda)} \ll 1, \tag{3.18}$$

and thus

$$1 - H(\Omega)min\left[1,\frac{\Theta}{S_{ee}(\Lambda)}\right] \approx 1. \tag{3.19}$$

In this case, (3.17) becomes:

$$\begin{aligned}S_{ee}(\Lambda) = &\frac{\Delta_t}{2\pi}\int_0^{\frac{2\pi}{\Delta_t}} E\left[|1-H(\Omega)|^2\right] S_{ss}(\Omega)\mathrm{d}\omega_t\\ &+\frac{\Delta_t\Theta}{2\pi}\int_0^{\frac{2\pi}{\Delta_t}} E\left[|H(\Omega)|^2\right]\mathrm{d}\omega_t.\end{aligned} \tag{3.20}$$

If $\Theta \geq S_{ee}(\Lambda)$, the prediction error is so small that we can consider $S_{ee}(\Lambda) = S_{ss}(\Lambda)$.

Although (3.20) can give us insight into the R-D modeling process of a generic MCP coder, it is derived under many assumptions such as optimum channel condition (filter and noise) and Gaussian distributed prediction error e, which are hard to be satisfied in practice. Furthermore, we need to insert (3.20) into (3.9) and (3.10) to derive the R-D function, which increases the computational cost of this method even more and makes it almost inapplicable to video applications that have time constraints. To resolve this problem, many simplified R-D functions were proposed in previous work and we will review them in the following subsection.

2. Related Work on R-D Modeling

In rate-distortion theory, there are no explicit R-D models, but only upper and lower bounds for general sources [54]:

$$Q2^{-2R} \leq D(R) \leq \sigma_G^2 2^{-2R}, \tag{3.21}$$

where Q is the entropy power and σ_G^2 is the variance of a Gaussian distributed source.

In contrast, there are two kinds of lower bounds in high-rate quantization theory [37]: the minimum distortion $D_1(N)$ attainable for a constrained number of quantization levels N, and the minimum distortion $D_2(R)$ attainable for a constrained bitrate R. However, in both quantization and rate-distortion theory, D can be expressed as an exponential function of bitrate R [33]:

$$D(R) \sim Ke^{-\alpha R}, \tag{3.22}$$

where parameters K, $\alpha > 0$ are unspecified constants. Model (3.22) is rarely used in practice and many video applications often rely on its refinement [54], [107]:

$$D(R) = \gamma\varepsilon^2\sigma_x^2 2^{-2R}, \tag{3.23}$$

where γ is the correlation coefficient of the source and ε^2 is a source-dependent scaling parameter (1.4 for Gaussian, 1.2 for Laplacian, and 1 for uniform sources).

For uniform quantizers (UQ), the classical model is often decomposed into two separate models with respect to quantizer step Δ: distortion $D(\Delta)$ and rate $R(\Delta)$. Under uniform quantization, both models can be summarized as [38]:

$$D(\Delta) = \frac{\Delta^2}{\beta}, \quad R(\Delta) = \frac{1}{2}\log_2\left(\frac{\varepsilon^2\beta\sigma_x^2}{\Delta^2}\right), \tag{3.24}$$

where β is 12 for small Δ. To account for a wider range of Δ, parameter β typically needs to be empirically adjusted based on samples of the R-D curve or other source parameters [38].

Based on approximation theory, Cohen *et al.* [15] derive an R-D bound for wavelet-based compression schemes. This is the first R-D bound that includes both bitrate R and $\log R$:

$$D(R) \leq CR^{-2\gamma}(\log R)^{2\gamma}, \tag{3.25}$$

where constant C and parameter γ are both positive. Since this bound is specifically developed for wavelet-based coding schemes, Mallat *et al.* [74] extend it to transform-based low bitrate images:

$$D(R) = CR^{1-2\gamma}, \tag{3.26}$$

where $\gamma \approx 1, C > 0$, and the parameters are adjusted with respect to practical coding settings.

For Laplacian sources with density $p(x) = \frac{\lambda}{2}e^{-\lambda|x|}$, the R-D function can be also written in terms of the Mean Absolute Difference (MAD) distortion D_M [104]:

$$R = -\log(\alpha D_M), \tag{3.27}$$

where α is some constant. Using Taylor expansion of (3.27), Chiang *et. al* [13] propose an operational R-D model for Laplacian sources and apply it to the MSE distortion D:

$$R = aD^{-1} + bD^{-2}, \tag{3.28}$$

where parameters a and b are obtained from samples of the empirical R-D curve.

In another recent development, He *et al.* [40] propose a unified *ρ-domain* R-D model, in which the bitrate is estimated by a linear function of the percentage of zero coefficients in each video frame and distortion D is directly computed without any modeling.

Besides the above operational models, there are purely empirical ways to estimate R-D curves, e.g., Lin *et al.* [66] use cubic interpolation of the empirical curve and Zhao *et al.* [113] apply similar methods to FGS-related streaming algorithms.

3. Current Problems

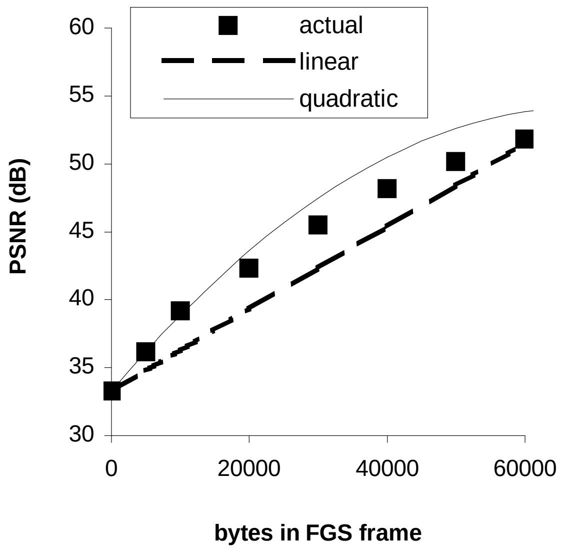
In what follows, we evaluate the accuracy of current R-D models in different frames. Recall that we use PSNR as an objective measurement of video quality and that the traditional R-D framework (3.22) becomes a linear function of rate R in the PSNR domain:

$$PSNR = 10\log_{10}\frac{255^2}{D} = \frac{20R}{\log_2 10} + 10\log_{10}\frac{255}{K}. \tag{3.29}$$

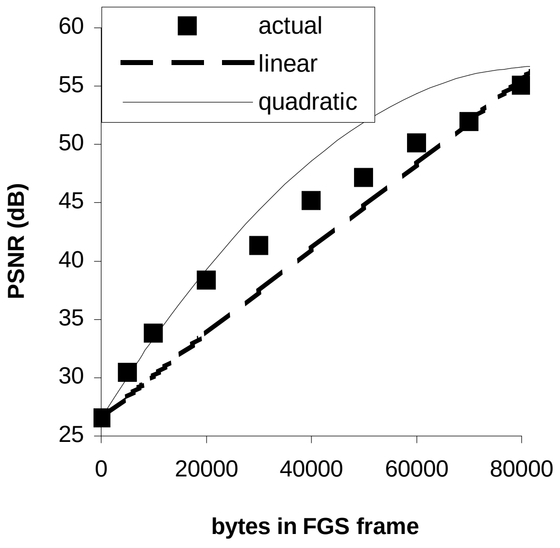
As shown in Fig. 14 for two different frames of CIF Foreman, the actual R-D curve of these frames cannot be modeled by a straight line over the entire range of R. In fact, even a heuristically selected quadratic curve in the figure (used here only for illustration purposes) is incapable of modeling the entire range of the bitrate. Both models exhibit significant discrepancy reaching as high as 5 dB.

In our next example, we evaluate the accuracy of models (3.23) and (3.28), which are extensions and/or improvements of the basic linear model. Fig. 15 shows the R-D curves produced by (3.23) (labeled as "classical" in the figure) and (3.28) (labeled as "Chiang *et al.*"). We use the log-scale of the x-axis in Fig. 15. Notice that (3.28) exhibits bending shape and produces negative values of R for sufficiently large D, which cannot be shown in the figure and the curve simply stops.

From the above figures, we observe that the classical R-D model and its variations do not perform well in scalable coders. The reason is that the classical R-D model $D \sim 2^{-2R}$ is typically obtained under the assumptions of an infinite block length



(a) Frame 39



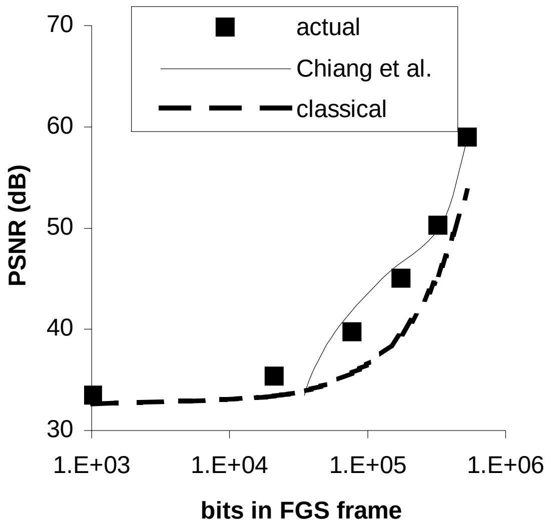
(b) Frame 73

Fig. 14. (a) Frame 39 and (b) frame 73 in FGS-coded CIF Foreman sequence.

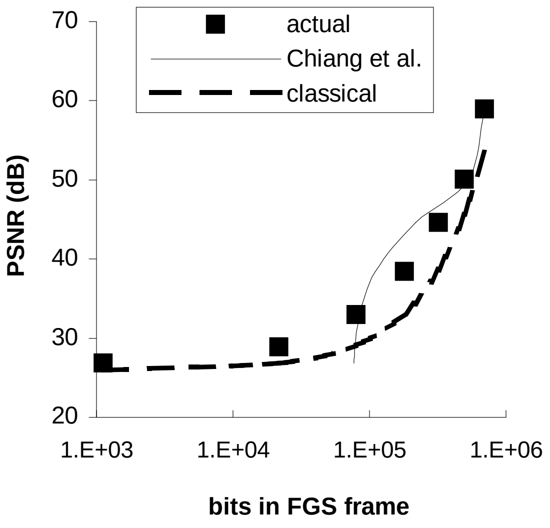
and high-resolution (i.e., small Δ) quantization that allows the PMF of the signal in each Δ-bin to be approximated by a constant [54], [82]. Neither of these two assumptions generally holds in practice, especially in cases of sharply decaying PMF of DCT residue (which is not constant even in small bins) and low-bitrate streaming (which inherently relies on high Δ).

To better understand some of these intricacies, we get back to the R-D functions of a generic MCP coder in (3.9) and (3.10). Assume that the autocorrelation function of $s(x, y)$ follows an isotropic model, which means the correlation between two points only depends on the Euclidean distance between them and this is quite common in an image [81]. Then, the autocorrelation function of s is:

$$r(x, y) = E[s(x_1, y_1)s(x_1 - x, y_1 - y)] = \sigma_s^2 e^{-\alpha\sqrt{x^2+y^2}}, \tag{3.30}$$



(a) Frame 0



(b) Frame 84

Fig. 15. R-D models (3.23), (3.28), and the actual R-D curve for (a) frame 0 and (b) frame 84 in CIF Foreman.

where σ_s^2 is the variance of signal s. The PSD function of s is:

$$S(\omega_x, \omega_y) = \frac{\omega_0}{2\pi} \frac{1}{(\omega_0^2 + \omega_x^2 + \omega_y^2)^{3/2}}, \tag{3.31}$$

where $\omega_0 = \alpha/2\pi$.

Since the PSD function is symmetrical in ω_x and ω_y, we transform (3.31) to polar coordinates f_r and let f_0 equal ω_0. Further notice that there is frequency f_Θ for parameter Θ in (3.9) and (3.10), so that $\Theta = S(f_\Theta)$. Then the R-D function can be written as:

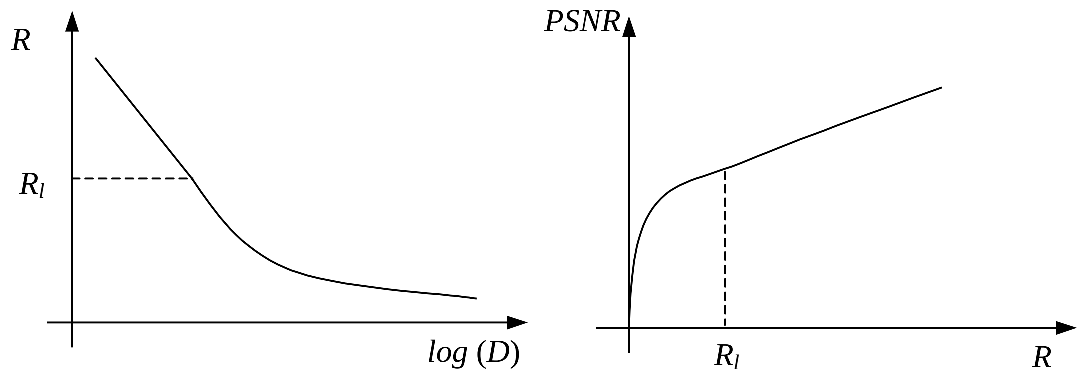
$$R(f_\Theta) = \pi \int_0^{f_\Theta} log \left[\frac{S(f_r)}{\Theta} \right] f_r \mathrm{d}f_r, \tag{3.32}$$

$$D(f_\Theta) = \pi f_\Theta^2 S(f_\Theta) + 2\pi \int_{f_\Theta}^{\infty} f_r S(f_r) \mathrm{d}f_r, \tag{3.33}$$

where f_Θ is the “throw-away” frequency, which means signal s is bandlimited to f_Θ

(recall that image signal is bandlimited).

O'Neal *et al.* [81] give a typical example of R-D curves for bandlimited processes, as shown in Fig. 16. Fig. 16 (a) shows that the $R-\log(D)$ curve is only linear for rate larger than certain rate R_l, which indicates that the PSNR-domain curve in Fig. 16 (b) matches our observation in Fig. 14 and 15.



(a) (b)

Fig. 16. (a) R-D functions for bandlimited process. Source: [81]. (b) The same R-D function in PSNR domain.

E. Distortion Analysis and Modeling

This section is devoted to the discussion of the distortion function. Subsection 1 and 2 model distortion from two different angles, and interestingly, they have very similar final results.

1. Distortion Model Based on Approximation Theory

It is well known that in an ideal orthogonal transform-based coding system, the distortion in the spatial domain is the same as that in the transform domain [54]. Furthermore, recall that the distortion in an ideal transform-based video coder is mostly introduced by quantization errors [54]. Since uniform quantizers are widely applied to video coders due to their asymptotic optimality [36], we show the lower bound on distortion in quantization theory assuming seminorm-based distortion measures (e.g., MSE) and uniform quantizers.

If X, $\hat{X}$ are k-dimensional vectors and the distortion between X and $\hat{X}$ is $d(X, \hat{X}) = ||X - \hat{X}||^r$ (where $||\cdot||$ is a seminorm in k-dimensional Euclidean space and $r \geq 1$), the minimum distortion for uniform quantizers is [37]:

$$D = \frac{k}{k+r}\left(\frac{V_k}{\Delta}\right)^{-r/k}, \tag{3.34}$$

where Δ is the quantization step, $V_k = \frac{2\pi^{k/2}}{k\Gamma(k/2)}$, and Γ is the Gamma function. When $r = 2, k = 1$, we obtain the popular MSE formula for uniform quantizers:

$$D = \frac{\Delta^2}{\beta}, \tag{3.35}$$

where β is 12 if the quantization step is much smaller than the signal variance [38]. However, this assumption is not always valid in real coders and β often becomes an adjustable parameter [38]. In contrast to many previous studies based on rate-distortion and quantization theory [38], [40], in what follows, we investigate the distortion from the perspective of approximation theory. Before we derive the distortion function, we explain the concept of approximation theory.

a. Approximation Theory

"The fundamental problem of approximation theory is to resolve a possibly complicated function, called the *target function*, by simpler, easier to compute functions called the approximates [21]." According to the approach to obtain approximation, there are linear and nonlinear approximation. We explain it in a mathematical way.

Assume that $\{\eta_k\}_{k\in N}$ is an orthonormal basis for a Euclidean space $L_2(R)$. By

n-term linear approximation for some $n \in N$, we approximate a function $f \in L_2(R)$ by truncating the expansion $f = \sum_{k=1}^{\infty} < \eta_k, f > \eta_k$ using only the *first* n terms. However, there might be important information included in the rest terms, which have been thrown away in the linear approximation. In contrast, nonlinear approximation only consider the n *most important* terms, not necessarily the first n terms. The n important terms are chosen according to different principles, e.g., minimizing the norm $\|f - \sum_{k \in \Lambda_n} <\eta_k, f> \eta_k\|$, where $\Lambda_n \subset N$ has cardinality n. For more information on approximation theory, readers are referred to [21].

b. The Derivation of Distortion Function

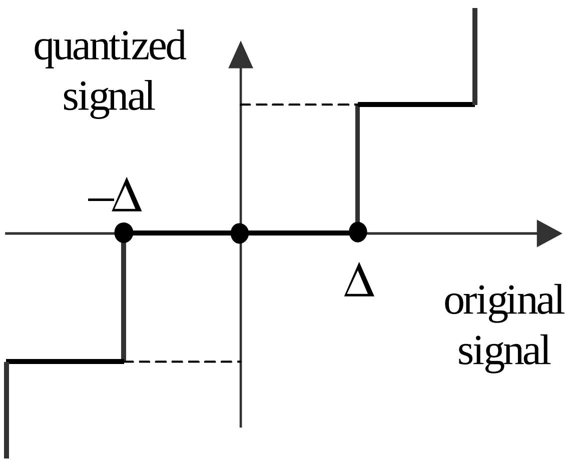


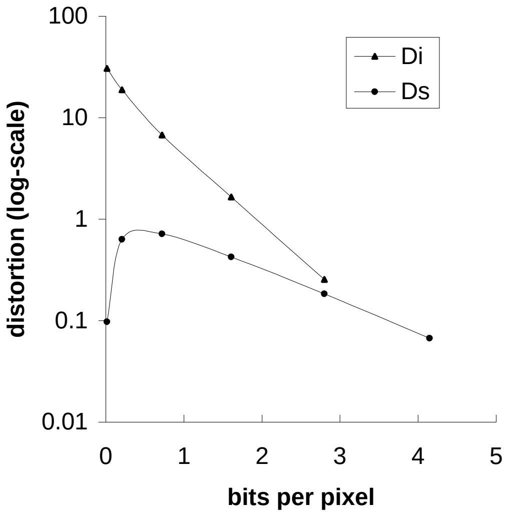
Fig. 17. Uniform quantizer applied in scalable coders.

Assume that signal X is transformed into signal U by an orthogonal transform, which later becomes $\hat{U}$ after quantization. Since a midtread uniform quantizer is commonly used in video coders, coefficients between $(-\Delta, \Delta)$ are set to zero, where Δ is the quantization step, as shown in Fig. 17. We call the coefficients that are larger than Δ *significant*.

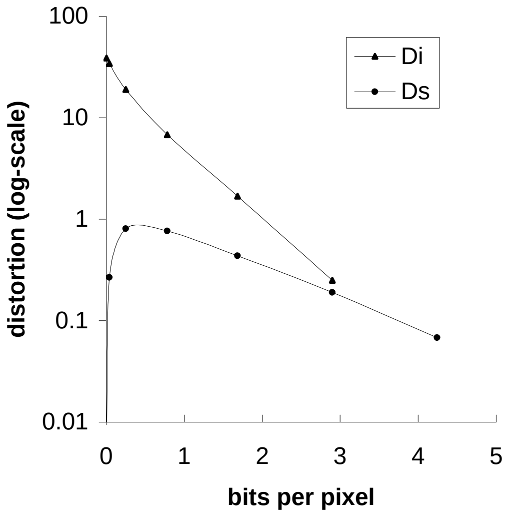
As we stated earlier, distortion D between X and the reconstructed signal $\hat{X}$

equals that between U and $\hat{U}$ [54]. In the transform domain, distortion D consists of two parts: 1) distortion D_i from discarding the insignificant coefficients in $(-\Delta, \Delta)$; and 2) distortion D_s from quantizing the significant coefficients (i.e., those that have larger values than Δ).

In Fig. 18, we compare the value of D_s and D_i to examine their relative importance in distortion D. The curve of D_i stops in the middle, because the theoretical value of D_i equals zero, which cannot be displayed on a log-scale of the figure. As demonstrated by the figure, the value of D_i is much larger than D_s in most cases and D_s is relatively important only if bitrate R is sufficiently large (in this particular case, above 2.5 bits/pixel). For large R, quantization step Δ is very small and thus the value of D_s is also small. Therefore, in many practical situations, distortion D_i, which is considered to be a nonlinear approximation error in approximation theory, plays in fact a critical role in the distortion of scalable coders [14].



(a) Frame 3



(b) Frame 6

Fig. 18. Distortion D_s and D_i in (a) frame 3 and (b) frame 6 in FGS-coded CIF Foreman sequence.

Given the notation and discussion of D_s and D_i, we have the following lemma.

Lemma 2 *Assuming that the total number of transform coefficients U is N and the number of significant coefficients is M, MSE distortion D is:*

$$D = \frac{1}{N} \sum_{|u|<\Delta} |u|^2 + \frac{M}{N} \frac{\Delta^2}{12}, \tag{3.36}$$

where Δ is the quantization step.

Proof: It is easy to understand that distortion D_i is directly the summation of the squares of insignificant coefficients:

$$D_i = \sum_{|u|<\Delta} |u|^2. \tag{3.37}$$

Since the high-resolution quantization hypothesis applies to M significant coefficients [74], their average distortion is $\Delta^2/12$ and thus their total distortion D_s is:

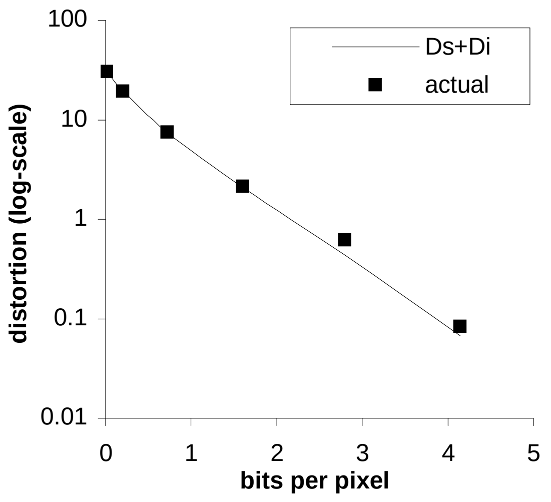
$$D_s = \sum_{|u|\geq\Delta} |u - \hat{u}|^2 = \frac{M\Delta^2}{12}. \tag{3.38}$$

Therefore, the average distortion for each coefficient D is:

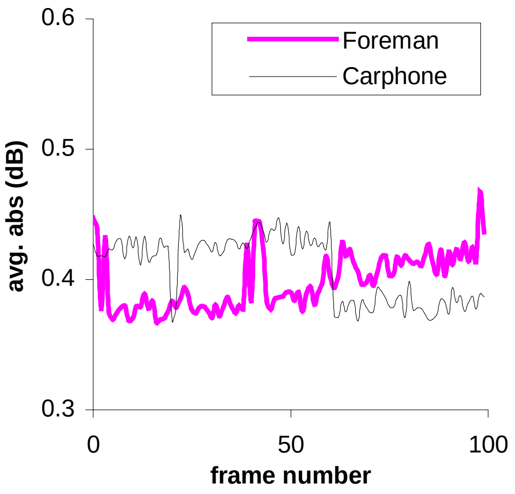
$$D = \frac{D_i + D_s}{N}, \tag{3.39}$$

which, combined with (3.37)-(3.38), leads to the result in (3.36). ■

In Fig. 19, the left side shows an example of actual distortion D and simulation results of model (3.36) for frame 3 in FGS-coded CIF Foreman, and the right side shows the average absolute error between model (3.36) and the actual distortion in FGS-coded CIF Foreman and Carphone sequences. As we observe from the figure, model (3.36) is very accurate and follows the actual distortion well. Furthermore, a less-than-0.5 dB average error is minuscule since a video sequence usually has quality above 30 dB.



(a) actual



(b) Gaussian

Fig. 19. (a) Actual distortion and the estimation of model (3.39) for frame 3 in FGS–coded CIF Foreman. (b) The average absolute error between model (3.36) and the actual distortion in FGS-coded CIF Foreman and CIF Carphone.

1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ... 0, 0	(MSB)
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ... 0, 0	(2nd MSB)
1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, ... 0, 0	(3rd MSB)
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ... 0, 0	(4th MSB or LSB)

Fig. 20. The structure of Bitplane coding.

2. Distortion Modeling Based on Coding Process

In the base layer, the distortion comes from applying a uniform (usually) mid-point quantizer to each DCT coefficient (different quantizers are often applied to different frequencies) [37], [38]. On the other hand, embedded coders such as FGS use *bitplane coding*, in which all coefficients are transmitted bit-by-bit from the most-significant bitplane (MSB) to the least-significant bitplane (LSB). This can be viewed as applying a quantizer step $\Delta = 2^{n-z}$, where n is the total number of bitplanes in the frame and

z is the current bitplane number.[2] For example, assuming that the maximum DCT coefficient is 40, n is 6 and Δ takes the values equal to 32, 16, 8, 4, 2, 1 for bitplanes 1 through 6, respectively. We also give an example of bitplane coding in Fig. 20. Assume the maximum value of the encoded data is 10, then the maximum layer of this block is 4.

Now that we understand bitplane coding and conceptually know that the source data are drawn from two Laplacian distributions, we can proceed to derive the distortion function $D(\Delta)$ for scalable coders in the following lemma.

Lemma 3 *For Laplacian sources with PMF $p(m) = ae^{b|m|}$, $a > 0$ and $b < 0$, the MSE distortion after uniform quantization with step Δ is:*

$$D(\Delta) \approx \frac{2a\xi}{1 - e^{b\Delta}}, \tag{3.40}$$

where ξ is given by:

$$\xi = e^{b(\Delta-1)} \left(\frac{(\Delta-1)^2}{b} - \frac{2(\Delta-1)}{b^2} + \frac{2}{b^3} \right) - \frac{2}{b^3}. \tag{3.41}$$

Proof: Since the PMF $p(m)$ of the source is always symmetric, the distortion after bitplane coding can be written as:

$$D(\Delta) = 2 \sum_{k=0}^{N/\Delta} \sum_{m=k\Delta}^{(k+1)\Delta-1} (m - k\Delta)^2 p(m). \tag{3.42}$$

where N is the maximum value of the quantizer equal to 2^{n-1} (recall than n is the

[2]While traditional quantizers implement mid-point reconstruction, bitplane coding can be viewed as a floor function applied to the result. Further note that MPEG-4 FGS has an option for "quarter-point" reconstruction, in which the decoder adds $\Delta/4$ to the result. For brevity, we omit $\Delta/4$ in all derivations; however, it can be shown that our final result holds for quarter-point quantizers as well.

total number of bitplanes). Replacing m with $k\Delta + i$ in (3.42):

$$\begin{aligned} D(\Delta) &= 2\sum_{k=0}^{N/\Delta}\sum_{i=0}^{\Delta-1}(k\Delta + i - k\Delta)^2 p(k\Delta + i) \\ &= 2\sum_{k=0}^{N/\Delta}\sum_{i=0}^{\Delta-1} i^2 a e^{b(k\Delta+i)} \\ &= 2a\sum_{k=0}^{N/\Delta} e^{bk\Delta}\sum_{i=0}^{\Delta-1} i^2 e^{bi}. \end{aligned} \tag{3.43}$$

The result in (3.43) is a product of two summation terms, each of which can be computed separately. First notice that $\sum i^2 e^{bi}$ is easily estimated using integration:

$$\sum_{i=0}^{\Delta-1} i^2 e^{bi} \approx \int_0^{\Delta-1} x^2 e^{bx} dx. \tag{3.44}$$

Solving (3.44), we have:

$$\int_0^{\Delta-1} x^2 e^{bx} dx = e^{bx}\left(\frac{x^2}{b} - \frac{2x}{b^2} + \frac{2}{b^3}\right)\Bigg|_0^{\Delta-1} = \xi, \tag{3.45}$$

where ξ is given by (3.41). Next consider term $\sum e^{bk\Delta}$ in (3.43) and notice that it is a geometric series with the following expansion:

$$\sum_{k=0}^{N/\Delta} e^{kb\Delta} = \frac{1 - e^{b(N+\Delta)}}{1 - e^{b\Delta}} \approx \frac{1}{1 - e^{b\Delta}}, \tag{3.46}$$

where the last approximation holds since $e^{b(N+\Delta)}$ is negligible and can be omitted for all practical values of N and b. Multiplying (3.46) by $2a$ and ξ, we obtain (3.40). ■

Notice that when $\Delta = 1$, (3.40) produces $D = 0$ and when $\Delta = \infty$, the distortion is reduced to $D = 2/\lambda^2 = \sigma_x^2$, where σ_x^2 is the variance of a Laplacian distribution. A distortion model for a mixture-Laplacian distribution is easily constructed by linearly combining (3.40) with the corresponding probability p and $1 - p$ as shown in (3.8). The result of applying model (3.40) to frame 0 in CIF Foreman is shown in Fig. 21

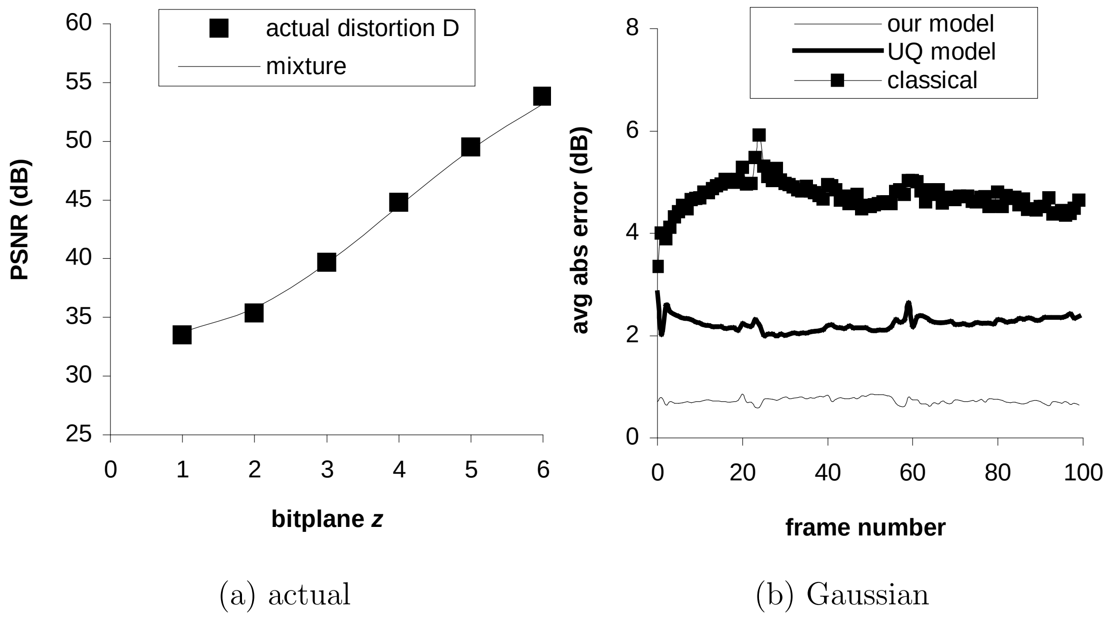


Fig. 21. (a) Spatial-domain distortion D in frame 0 of CIF Foreman and distortion estimated by model (3.40) with mixture-Laplacian parameters derived from the FGS layer. (b) The average absolute error in the CIF Coastguard sequence.

(a).

We extensively analyzed the performance of model (3.40) in other sequences and found that it was very accurate. Fig. 21 (b) compares the performance of (3.40) to that of the classical model (3.23) and UQ model (3.24) in FGS-coded CIF Coastguard. The error in the figure is computed for each frame in the PSNR domain and then averaged over all bitplanes. As the figure shows, (3.40) maintains the average error below 0.8 dB, while the errors in the other two methods average between 2 and 6 dB.

Note, however, that this form of averaging can be misleading since large errors in the last bitplane (where they do not matter due to high signal PSNR) may skew the result obtained from the other bitplanes. Thus, in Table III, we examine the average errors *for each bitplane* over the entire CIF Foreman sequence (similar results hold for Coastguard and Carphone). As the table shows, the PSNR error is quite small for all bitplanes except the last one where approximation (3.44) is most weak and results in the largest discrepancy between the model and the data. It is also worthwhile to note that a 1-dB error in a signal reconstructed at 56 dB is not noticeable, as well as that 0.15-dB errors in 30+ dB signals are relatively minor.

Table III. Estimation Accuracy of (3.40) in CIF Foreman.

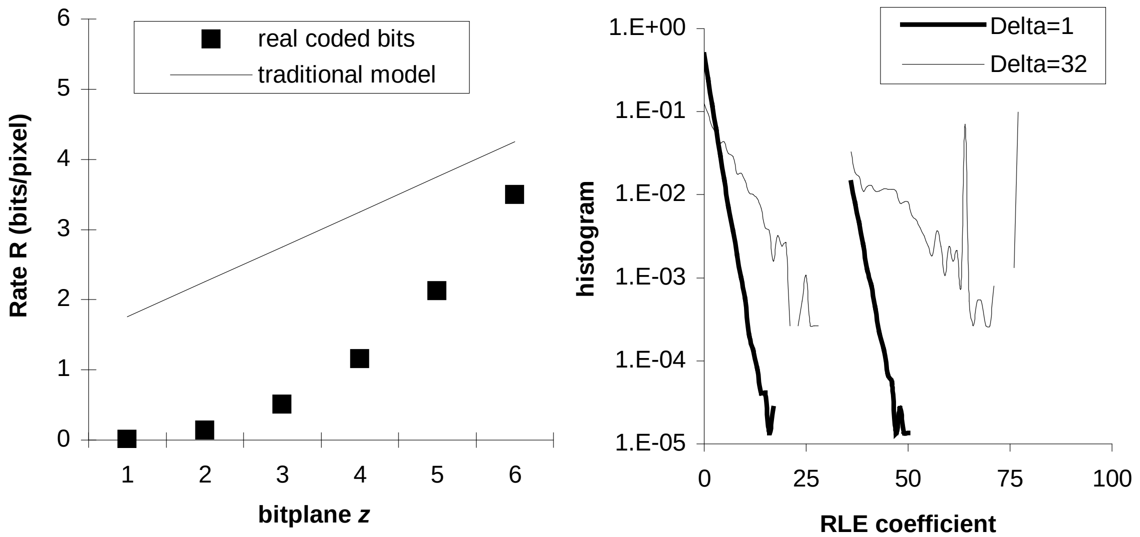
Δ	Average D	Average abs.error	Error in dB
64	81.5 (29.9 dB)	2.987	0.15
32	51.6 (31.2 dB)	1.768	0.15
16	23.1 (34.6 dB)	0.558	0.10
8	7.92 (39.2 dB)	0.239	0.13
4	2.16 (44.6 dB)	0.128	0.24
2	0.62 (49.8 dB)	0.039	0.25
1	0.08 (56.6 dB)	0.043	1.15

Finally note that (3.40) applies to any Laplacian source regardless of reconstruction points and whether the source contains FGS residue or base-layer DCT coefficients. Apart from the distortion analysis, modeling bitrate of scalable coders is another challenging work.

F. Rate Analysis and Modeling

1. Preliminaries

As mentioned, bitplane coding is applied to DCT residue in the enhancement layer to achieve high flexibility during transmission (i.e., the bitstream can be truncated at any codeword). Even though bitplane coding is more efficient than common run-level



(a) actual (b) Gaussian

Fig. 22. (a) Actual FGS bitrate and that of the traditional model (3.24) in frame 0 of CIF Foreman. (b) The distribution of RLE coefficients in frame 84 of CIF Foreman.

coding in the base layer [67], modeling the bitrate of bitplane-coded data is rather difficult since each bitplane has a different correlation model.

Recall that the traditional bitrate model (3.24) can be viewed as a linear function of z in the bitplane domain (i.e., a linear function of $\log \Delta$). While this linear approach may be acceptable for a high-level description of R-D properties of the source, in practice a more accurate model is often needed. Fig. 22 (a) illustrates that the traditional framework (3.24) is accurate only at very high bitrates (i.e., large z). Furthermore, as the figure shows, the straight line of the traditional model does not account for the non-linear shape of the curve in this particular frame. Since this mismatch is predominant in FGS-coded video, we seek an alternative explanation for the shape of the curve.

One possible way of modeling the run-length coding (RLE) structure of bitplane coding is to analyze the distribution of runs within each bitplane. This naturally leads to n distributions per frame, where n is the number of bitplanes. An example of this modeling is shown in Fig. 22 (b), which illustrates the histogram of run-length coefficients in frame 84 of CIF Foreman for two extreme cases of $\Delta = 1$ and $\Delta = 32$ (similar plots are shown in [65]). In Fig. 22 (right), both histograms can be modeled by exponential (geometric) distributions (i.e., straight lines on a log scale) with high accuracy if we ignore the all-zero blocks. This approach is fairly straightforward, but does require modeling many minor details specific to FGS bitplane coding. We thus offer several alternative approaches below.

2. Markov Model

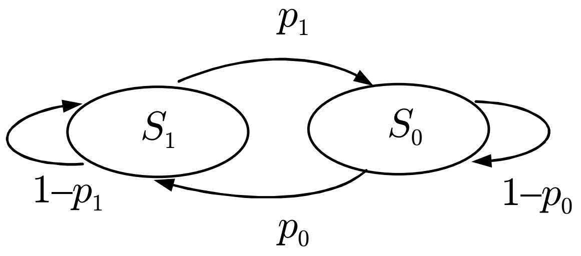


Fig. 23. First-order Markov model for binary sources.

Another way of modeling correlated data in FGS bitplanes is to use Markov chains. Below, we first present a classical RLE Markov model for correlated single-bit data and explain why it is not accurate in fine scalable coders. Then we derive a new Markov model, in which only runs of 0s are coded with RLE, and show that it matches the real data very well.

Assume that we reorganize each bitplane in block order and model the resulting sequence of 0s and 1s *within that bitplane* as a stationary, ergodic, first-order Markov chain $\{X_i\}$. As shown in Fig. 23, the Markov model for binary sources only has two states: S_1 and S_0, which represent 1s and 0s in the binary sequence. Let i represent the current coefficient and $i-1$ the previous one, then transition probabilities p_0 and p_1 in the figure are given by:

$$p_0 = P\{X_i = S_1 | X_{i-1} = S_0\} = P(S_1|S_0), \tag{3.47}$$

$$p_1 = P\{X_i = S_0 | X_{i-1} = S_1\} = P(S_0|S_1). \tag{3.48}$$

Recall that the entropy rate of a two-state Markov process is [80]:

$$H(X) = P(S_1)H(X|S_1) + P(S_0)H(X|S_0), \tag{3.49}$$

where $P(S_j)$ is the probability for the Markov chain to be in state $j, j = 0, 1$:

$$p(S_0) = \frac{p_1}{p_1 + p_0},\ p(S_1) = \frac{p_0}{p_1 + p_0}, \tag{3.50}$$

and $H(X|S_j)$ is the conditional entropy of each state [80]:

$$H(X|S_0) = -p_0 \log_2 p_0 - (1 - p_0)\log_2(1 - p_0), \tag{3.51}$$

$$H(X|S_1) = -p_1 \log_2 p_1 - (1 - p_1)\log_2(1 - p_1). \tag{3.52}$$

The main difficulty in a direct application of the above Markov modeling to FGS data is that (3.49) assumes that runs of 1s are also RLE-coded. However, in FGS bitplane coding, only runs of 0s are coded with RLE, and each occurrence of a 1 produces a special symbol that needs to be separately VLC coded. As a result, model (3.49) is accurate only for the first several bitplanes (which contain very few 1s) and then starts to significantly underestimate the actual bitrate (by as much as 20-30%).

To solve this problem, we next extend the original model (3.49) and derive Markov-based entropy that reflects the coding process of many embedded coders.

Lemma 4 *The bitrate of each bitplane in scalable coding is given by a modified*

Markov model:

$$H(z) = p_0 H(X|S_1) + H(X|S_0), \tag{3.53}$$

where p_0, $H(X|S_0)$, and $H(X|S_1)$ are computed separately for each bitplane z using (3.47), (3.51), and (3.52), respectively.

PROOF: Since there is no specific codeword for 1s in the assumed bitplane coding, we pursue a different approach for normalizing the entropy of each state as compared with the traditional approach in information theory. Instead of modeling the entropy of 1-runs and dividing it by the average length of a 1-run, we count the entropy of state S_1 *as if* it were a part of state S_0 and then divide both entropies by the length of the average zero-run. Thus, the average entropy is given by:

$$H(z) = \frac{H_0 + H(X|S_1)}{r_0}, \tag{3.54}$$

where H_0 is the entropy of S_0 and r_0 is the expected length of a zero-run. Next notice that the probability to encounter a zero-run of length r is given by a geometric distribution $P_0(r) = (1-p_0)^{r-1}p_0$ and that the entropy of the zero state is [80]:

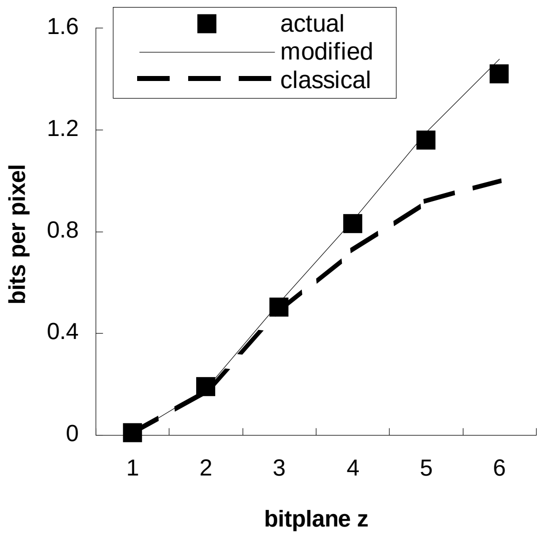
$$\begin{aligned} H_0 &= -\sum_{r=1}^{\infty} P_0(r) \log_2 P_0(r) \\ &= -\left[\log_2 p_0 + \frac{(1-p_0)\log_2(1-p_0)}{p_0}\right]. \end{aligned} \tag{3.55}$$

Finally, it is easy to see that:

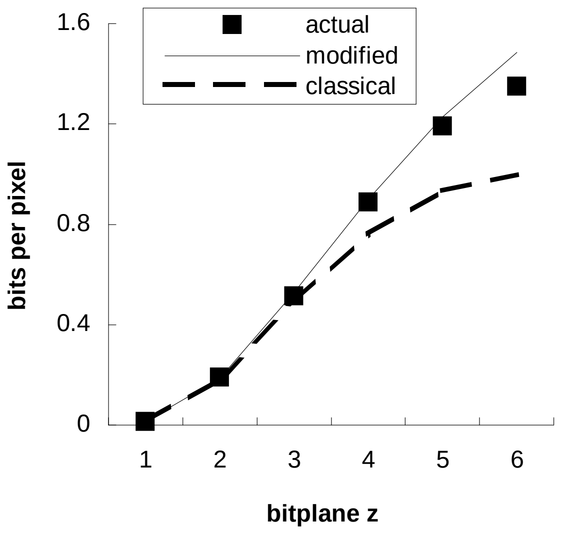
$$r_0 = \sum_{r=1}^{\infty} r P_0(r) = 1/p_0. \tag{3.56}$$

Combining (3.52) and (3.55) in (3.54), we obtain the modified Markov model (3.53). ■

Examples in Fig. 24 show the actual entropy rates for several frames in the CIF Foreman sequence, fitted with classical (3.49) and modified (3.53) Markov models.



(a) Frame 0



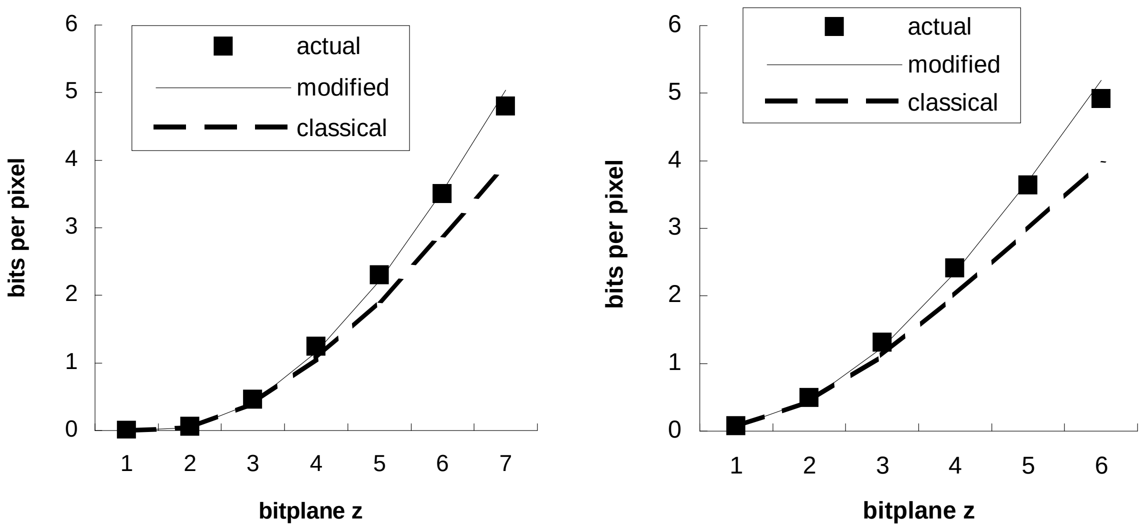
(b) Frame 3

Fig. 24. Entropy estimation of the classical model (3.49) and the modified model (3.53) for (a) frame 0 and(b) frame 3 in CIF Foreman sequence.

As the figure shows, the traditional approach does in fact underestimate the bitrate of the last bitplane by as much as 30%, while the modified model is capable of tracking rate R over the entire range of bitplanes z. Note that these results directly apply only to the rates of individual bitplanes z. Thus, if the server transmits all bitplanes up to and including bitplane z, the cumulative rate $R(z)$ is the summation of the individual bitplane rates:

$$R(z) = \sum_{k=1}^{z} H(k), \tag{3.57}$$

where $z = n - \log_2 \Delta$ and n is the total number of bitplanes in the frame. Fig. 25 and Fig. 26 show the remarkable accuracy of the final result (3.57) in modeling accumulative rate $R(z)$. Although it requires two estimated probabilities (p_0, p_1) per bitplane,this approach is highly accurate. Combined with the distortion model (3.40), the result in (3.57) allows the construction of accurate R-D curves only based on the



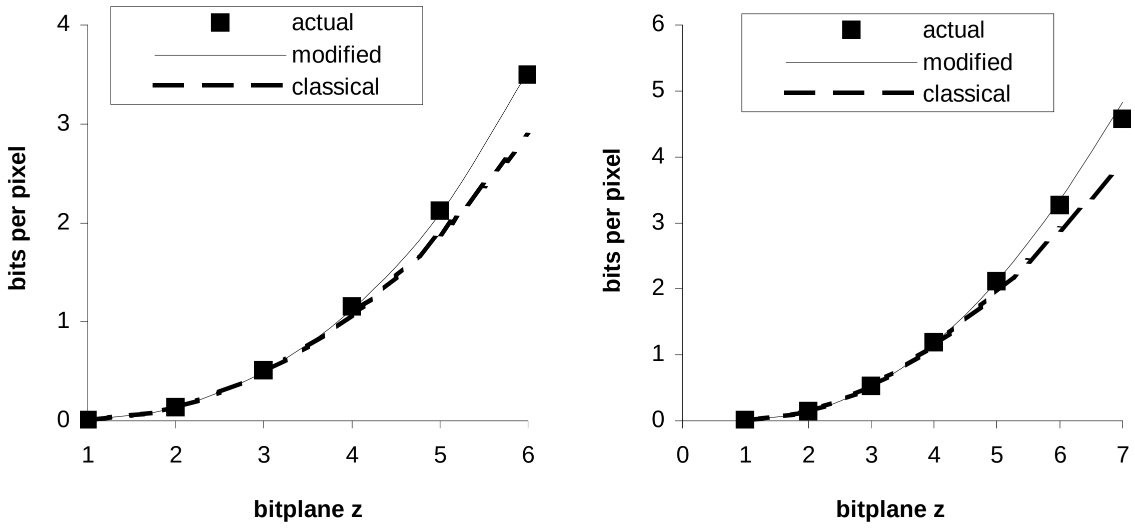
(a) Frame 0 (b) Frame 3

Fig. 25. Bitrate $R(z)$ and its estimation based on (3.57) for (a) frame 0 and (b) frame 3 in CIF Coastguard sequence.

statistical properties of DCT residue.

Although model (3.57) is quite accurate, it requires a non-trivial effort in obtaining transition probabilities for each bitplane and a rather large set of configuration parameters (10-14 parameters per frame) that may be undesirable in real-life streaming situations. Therefore, we next investigate an alternative bitrate model that requires much fewer parameters than the Markov model.

He *et al.* [40] proposed a unified ρ-domain model to estimate the bitrate of image and non-scalable video coders, in which bitrate R is a linear function of the percentage of significant coefficients z in each video frame. Although this model targets at image and non-scalable coders, we extensively examined the relationship between R and z in various video frames and found this linear model still holds for scalable coders. Fig. 27 demonstrates two typical examples of the actual bitrate R and its linear estimation in



(a) Frame 0 (b) Frame 84

Fig. 26. Bitrate $R(z)$ and its estimation based on (3.57) for (a) frame 0 and (b) frame 84 in CIF Foreman sequence.

FGS and PFGS video frames. Using this simple-format rate model, we further derive R-D models in the following section.

G. A Novel Operational R-D Model

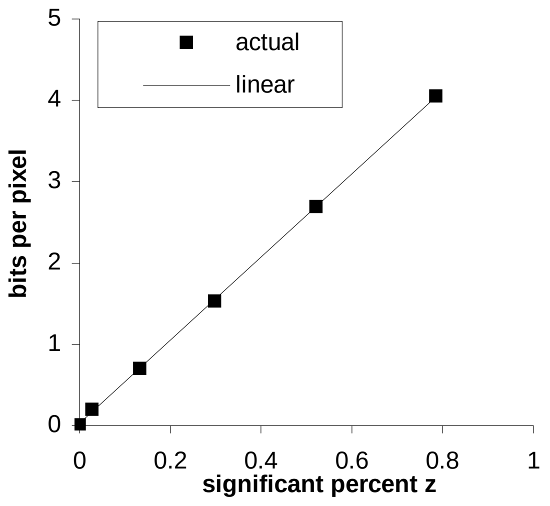
In this section, we derive an operational R-D model using the ρ-domain rate model and the distortion model we just derived. Our main result is as following:

Theorem 1 *The distortion of scalable video coders is given by:*

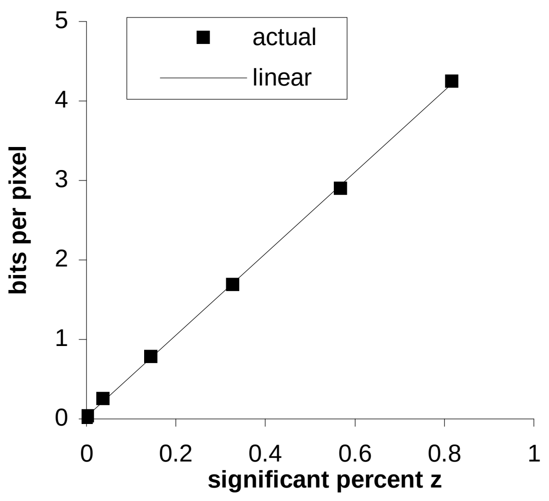
$$D = \sigma_x^2 - \left(a \log^2 R + b \log R + c\right) R, \tag{3.58}$$

for some constants $a - c$.

Proof: Notice that the transform coefficients U of scalable coders often follow a



(a) Foreman



(b) Coastguard

Fig. 27. Bitrate estimation of the linear model $R(z)$ for (a) frame 0 in FGS-coded CIF Foreman and (b) frame 6 in PFGS-coded CIF Coastguard.

mixture Laplacian distribution with density [18]:

$$f(x) = p\frac{\lambda_0}{2}e^{-\lambda_0|x|} + (1-p)\frac{\lambda_1}{2}e^{-\lambda_1|x|}. \tag{3.59}$$

During the discussion, we first use pure Laplacian-distributed sources for simplicity (i.e., $f(x) = \frac{\lambda}{2}e^{-\lambda|x|}$), and then obtain the final version of R-D model.

Since the coefficients inside the zero bin $(-\Delta, \Delta)$ are set to zero after quantization, the average distortion of the insignificant coefficients is:

$$\begin{aligned}\frac{D_i}{N} &= \int_{-\Delta}^{\Delta} x^2 \frac{\lambda}{2}e^{-\lambda|x|}dx \\ &= \frac{2}{\lambda^2} - [(\Delta + \frac{1}{\lambda})^2 + \frac{1}{\lambda^2}]e^{-\lambda\Delta},\end{aligned} \tag{3.60}$$

where N is the total number of coefficients and λ is the shape parameter of the Laplacian distribution. From (3.38), we have the average distortion of the significant coefficients:

$$\frac{D_s}{N} = \frac{M}{N}\frac{\Delta^2}{12}, \tag{3.61}$$

We define $z = M/N$ to be the percentage of significant coefficients. Notice that for Laplacian distributed sources, the percentage of significant coefficients z is:

$$z = 1 - 2\int_0^\Delta \frac{\lambda}{2} e^{-\lambda x} dx = e^{-\lambda\Delta}. \tag{3.62}$$

Thus, distortion D in (3.36) becomes:

$$D = \frac{D_i}{N} + \frac{z\Delta^2}{12} = \frac{2}{\lambda^2} - \zeta e^{-\lambda\Delta} + \frac{e^{-\lambda\Delta}\Delta^2}{12}, \tag{3.63}$$

where $\zeta = (\Delta + \frac{1}{\lambda})^2 + \frac{1}{\lambda^2}$.

Next, recall that He *et al.* [40] demonstrated in numerous simulations that in a variety of image and video coding methods, rate $R(z)$ was proportional to the percentage of non-zero coefficients z in the source data. In other words, bitrate

$$R(z) = \gamma z, \tag{3.64}$$

where γ is some source-dependent constant. Noticing the relationship between Δ and z as presented in (3.62), we express Δ in terms of rate R:

$$\Delta = -\frac{1}{\lambda}\log\frac{R}{\gamma}, \quad 0 < \frac{R}{\gamma} \le e^{-\lambda}. \tag{3.65}$$

Therefore, combining (3.65) with (3.63), distortion D is a function of R and $\log R$:

$$D = \frac{2}{\lambda^2} - \frac{11\tau^2 + 24\tau + 24}{12\lambda^2}\frac{R}{\gamma}, \tag{3.66}$$

where $\tau = \log\gamma - \log R$. We also notice that:

$$D = \begin{cases} \dfrac{2}{\lambda^2} = \sigma_x^2, & R = 0 \\ 0, & R \geq e^{-\lambda}\gamma \end{cases} \tag{3.67}$$

where σ_x^2 is the variance of the source. This observation makes perfect sense since distortion D should not be larger than σ_x^2 [17] and should equal zero when $R = e^{-\lambda}\gamma$ (i.e., the quantization step $\Delta = 1$ and there is no loss of information).

An R-D model for a scalable coder is simply a linear combination of (3.66), with corresponding probability p and distribution parameters λ_0, λ_1 as shown in (3.59). After absorbing the various constants, we have the desired result in (3.58). ■

Proof: Combining result (3.65) with our earlier distortion model (3.40), we have:

$$D(\Delta) \approx \frac{\lambda\xi}{1 - R/\gamma}, \tag{3.68}$$

where ξ is:

$$\xi = \frac{2}{\lambda^3} - \frac{e^{\lambda}R}{\lambda^3\gamma}\left((\log R - \log\gamma + \lambda - 1)^2 + 1\right). \tag{3.69}$$

Expanding (3.69) and combining it with (3.68), we notice that:

$$D = \begin{cases} \dfrac{2}{\lambda^2} = \sigma_x^2, & R = 0 \\ 0, & R \geq e^{-\lambda}\gamma \end{cases} \tag{3.70}$$

where σ_x^2 is the variance of the source. After absorbing the various constants and neglecting small terms, we have the desired result in (3.58). ■

Estimation of γ for a scalable coder is very simple. For example, once the FGS layer is coded, the number of bits $R(z)$ in each bitplane can be easily obtained by scanning the FGS layer for bitplane start codes (whose location can also be saved during encoding). Computing the percentage of zeros ρ_z in each bitplane directly from the DCT residue, the encoder can build the curve $(1 - \rho_z, R(z))$ and estimate its linear slope γ.

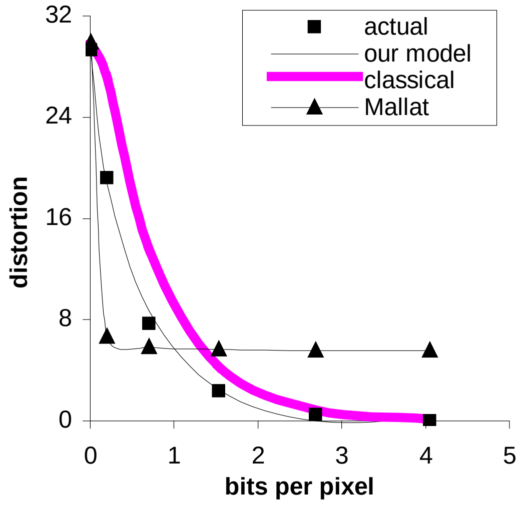
1. Experimental Results

We apply the proposed model (3.58) to various scalable video frames to evaluate its accuracy. Throughout this chapter, we use MPEG-4 FGS and PFGS to code popular CIF sequences such as Foreman, Coastguard, Carphone, and Mobile. The base layer is always coded at 128 kb/s and 10 fps, which is a common evaluation setup for R-D analysis of scalable video streaming [105], [113], [114]. While our discussion mainly involves derivatives of FGS/PFGS, our analytical results are applicable to a wide range of scalable (embedded) coding method and even non-scalable streams of MPEG-4 and H.264.

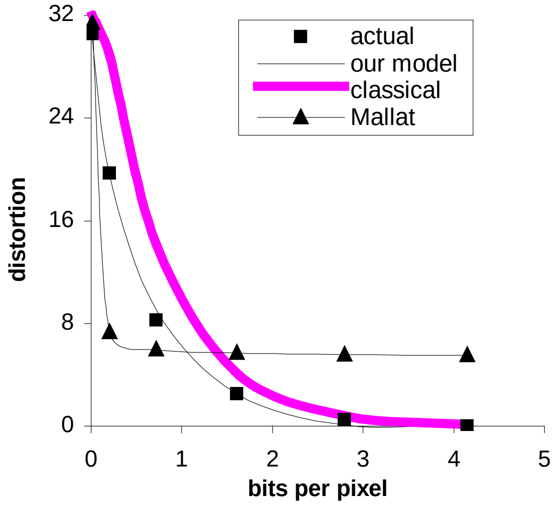
Fig. 28 shows two examples of R-D curves for I (a) and P (b) frames of FGS-coded CIF Foreman. As shown in the figure, the low bitrate model (3.26) tends to under-estimate distortion in general and saturates when bitrate R is large. Fig. 28 also shows that while the classical model (3.23) over-estimates the actual R-D curves, our model (3.58) tracks them with very high precision.

To better understand the estimation accuracy of the proposed model (3.58), we further compare it to models (3.23) and (3.26) in a variety of scalable video sequences. Simulation results in Fig. 29 show that model (3.58) outperforms traditional R-D models and maintains high accuracy in a variety of FGS-coded video frames.

We also compare the performance of the logarithmic model (3.58) to that of other two models in FGS-coded Foreman and Carphone in Fig. 30, and show the same comparison in PFGS-coded Coastguard and Mobile in Fig. 31. As both figures show, model (3.58) keeps the average absolute error quite low compared to that of the other models. Additional experimental results (not shown here due to a lack of space)



(a) I-frame



(b) P-frame

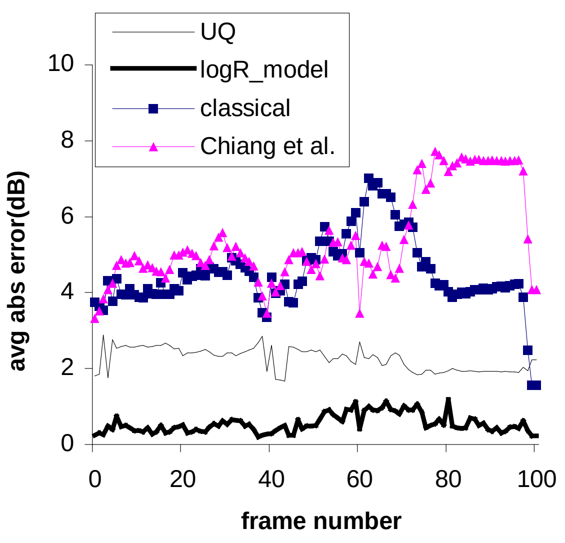
Fig. 28. Actual R-D curves and their estimations for (a) frame 0 and (b) frame 3 in FGS-coded CIF Foreman.

demonstrate that (3.58) significantly outperforms other operational R-D models in a wide variety of scalable sequences.

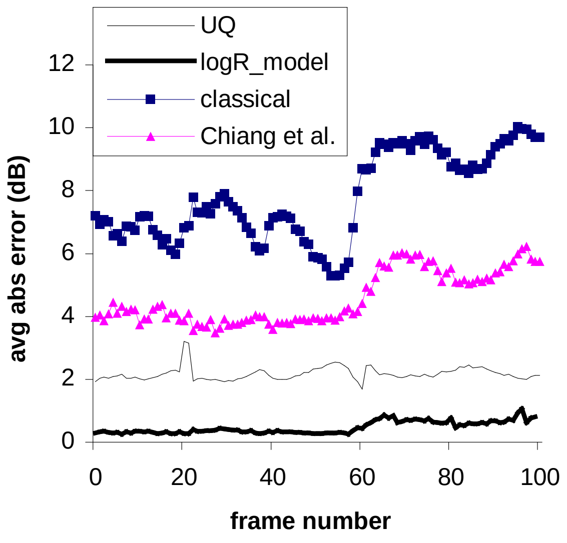
The result in (3.58) provides valuable insight into the coding process and suggests the shape of the resulting R-D curve. Nevertheless, this model is too complicated for time-constrained streaming applications. Thus, we examine an even simpler operational model in the next section and later use it during Internet streaming.

H. Square-Root R-D Model

Next, we derive another R-D model, which will be converted into the PSNR domain for the convenience of quality control during streaming.



(a) Foreman



(b) Carphone

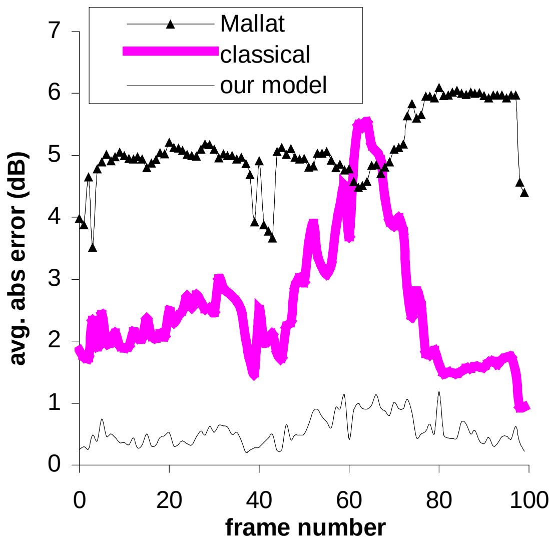
Fig. 29. Comparison between the logarithmic model (3.58) and other models in FGS–coded (a) CIF Foreman and (b) CIF Carphone, in terms of the average absolute error.

1. Simple Quality (PSNR) Model

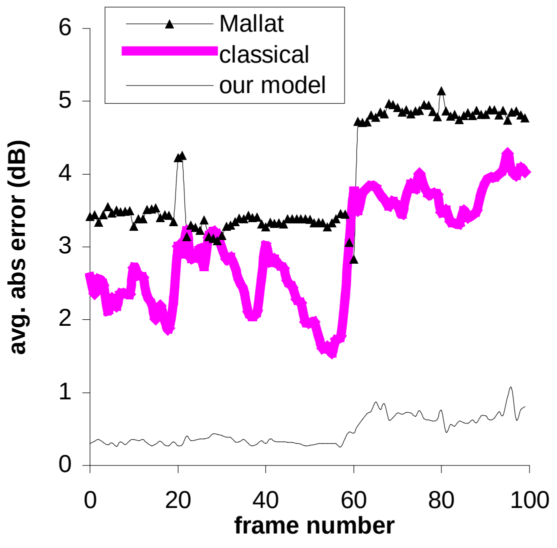
Notice that the previously derived distortion model is too complicated for further analytical manipulation. In the following discussion, we convert D into the PSNR domain and reduce it to a simpler formula through a series of approximations. Taking the logarithm of (3.40), omitting insignificant terms, and grouping constants, we obtain:

$$\log D(\Delta) \approx c_1 + e^{b\Delta} + b\Delta + \log(c_2\Delta^2 + c_3\Delta + c_4). \qquad (3.71)$$

for some constants $c_1, \ldots, c_4$. In the working range of most video coders, Δ is no more than 128 and the number of bitplanes usually does not exceed 7. In this limited range, a number of approximations hold: $\log(x^2 + x + c) \approx a\log^2 x + b\log x + c$ and $x + e^{bx} \approx a\log^2 x + b\log x + c$, for some constants a–c. Then, (3.71) can be further



(a) Foreman



(b) Carphone

Fig. 30. The average absolute errors of the logarithmic model (3.58), classical model (3.23), and model (3.26) in FGS-coded (a) CIF Foreman and (b) CIF Carphone.

simplified to:

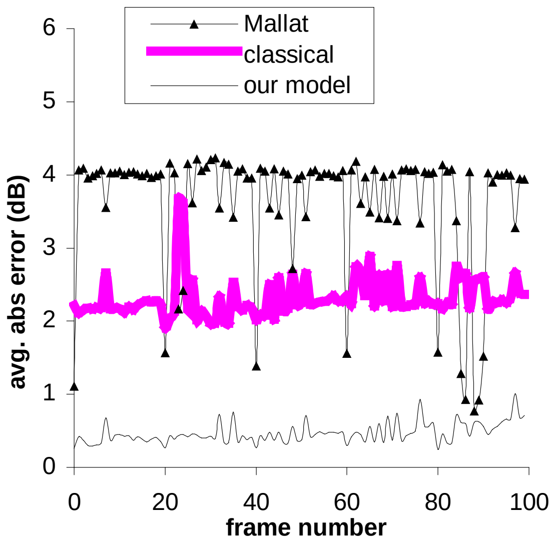
$$\log D(\Delta) \approx e_1 \log^2 \Delta + e_2 \log \Delta + e_3. \tag{3.72}$$

Since $\Delta = 2^{n-z}$, (3.72) shows that PSNR curves of this approximation are quadratic polynomials of the bitplane number z:

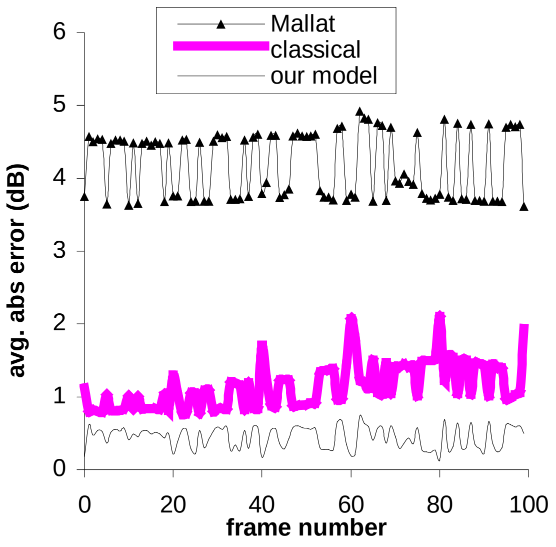
$$PSNR(z) \approx g_1 z^2 + g_2 z + g_3, \tag{3.73}$$

for some constants $g_1, \ldots, g_3$. This expression is very useful since polynomials are easy functions to work with and smoothly generalize the linear model of the traditional framework where g_1 equals zero.

To verify this approximation, we conducted a series of tests by fitting the simplified model (3.73) to the PSNR calculated from the original model (3.40) and found



(a) Coastguard



(b) Mobile

Fig. 31. The average absolute errors of the logarithmic model (3.58), classical model (3.23), and model (3.26) in PFGS-coded (a) CIF Coastguard and (b) CIF Mobile.

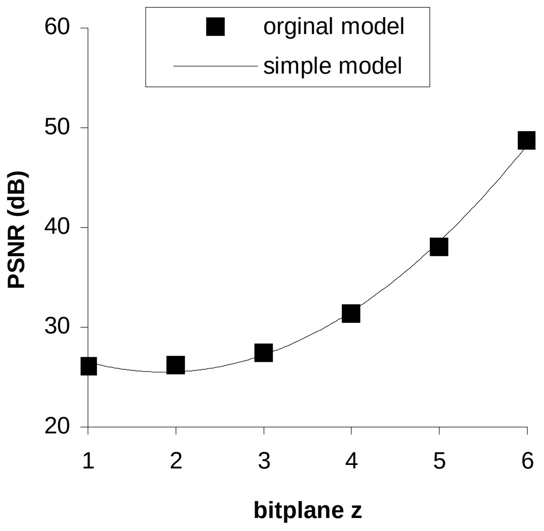
them to be an almost perfect match. The quality of the fit is illustrated on two different Laplacian distributions in Fig. 32. The left side of the figure shows a low-variance (high λ) case and the right side of the figure shows a high-variance (small λ) case; both matched the quadratic model (3.73) with very high accuracy.

2. Simple Bitrate Model

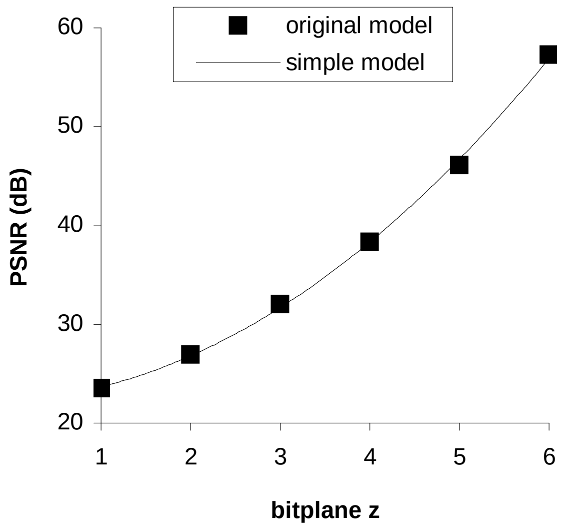
We first need the following supplementary result.

Lemma 5 *Function $R(z)/\gamma$ for $z \in [1, n]$ is monotonically increasing, changes convexity no more than once, and remains in [0,1) for all bitplanes z.*

Proof: Combining (3.64) with (3.65) and keeping in mind that $\Delta = 2^{n-z}$, we



(a) $\lambda = 0.5$



(b) $\lambda = 0.12$

Fig. 32. Comparison between the original Laplacian model (3.40) and the approximation model (3.73) for (a) $\lambda = 0.5$ and (b) $\lambda = 0.12$.

have:

$$\psi(z) = \frac{R(z)}{\gamma} = e^{-\lambda 2^{n-z}} < 1. \tag{3.74}$$

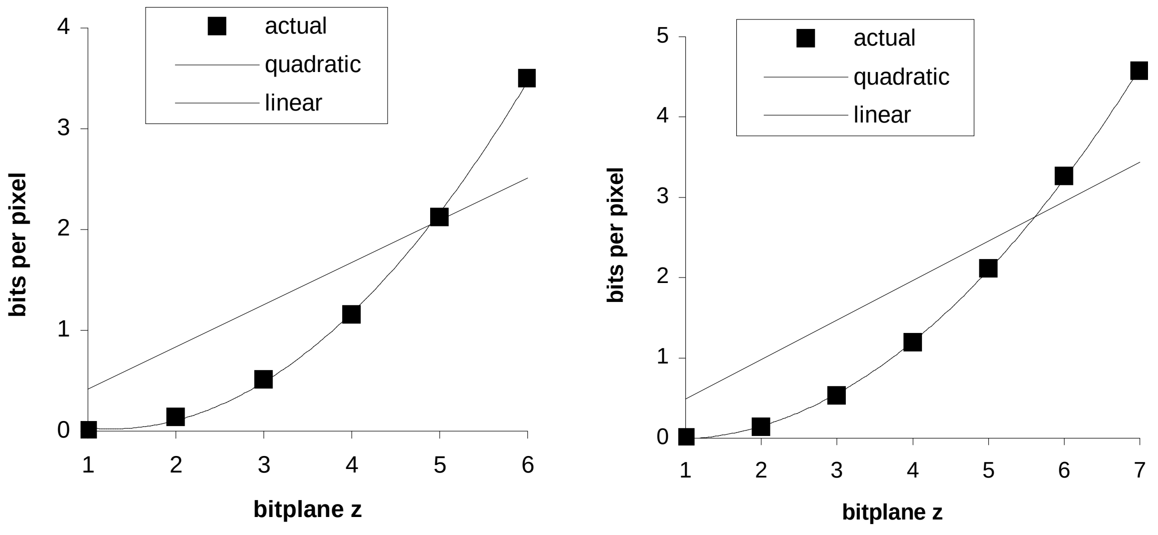
Taking the first two derivatives of (3.74), we have:

$$\psi'(z) = \lambda 2^{n-z} \log(2)\, \psi(z) > 0, \tag{3.75}$$

$$\psi''(z) = \lambda \log(2)\, [-2^{n-z} \log 2\psi(z) + 2^{n-z}\psi'(z)]. \tag{3.76}$$

Analysis of (3.76) shows three important points: (a) for $\lambda \geq 1$, the function ψ remains strictly convex in the entire interval, (b) for $\lambda \leq 2^{1-n}$, the function remains strictly concave, and (c) for the remaining values of λ, there is exactly one point $z = n + \log_2 \lambda$, in which the function changes convexity. ■

Using the theory of coconvex/comonotone approximation [61], an accurate polynomial approximation of $R(z)$ would require a cubic curve to match the possible



(a) Frame 0 (b) Frame 84

Fig. 33. Comparison between quadratic model for $R(z)$ and the traditional linear model in (a) frame 0 and (b) frame 84 of CIF Foreman.

change in convexity of the curve (the rest of the error is small since (3.74) exhibits a good degree of smoothness). However, since working with cubic polynomials is still rather complex (e.g., for realtime rate-control applications), we apply a quadratic approximation to $R(z)$ in the z-domain and reduce (3.74) to:

$$R(z) = a_1 z^2 + a_2 z + a_3, \tag{3.77}$$

where constants $a_1, \ldots, a_3$ can be estimated from empirical data.

To better understand this operational model, we conducted numerous experiments and found that while cubic polynomials were a very good match to $R(z)$, quadratic functions also performed well. Fig. 33 shows one such example for two frames of CIF Foreman, as well as a linear fit derived from model (3.24).

3. SQRT Model

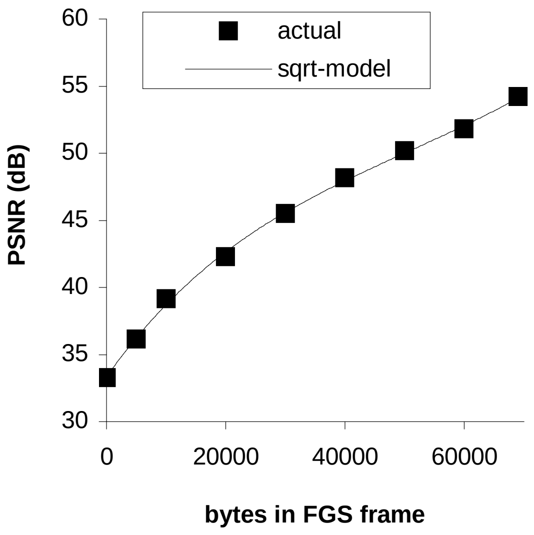
We next combine our proposed bitrate result in (3.77) with the earlier distortion model in (3.73) to obtain a final usable R-D model. After inverting the polynomial in (3.77), inserting $z(R)$ into (3.73), and dropping insignificant terms, we obtain the model that we call *Square Root* (SQRT):

$$PSNR(R) = AR + B\sqrt{R} + C, \tag{3.78}$$

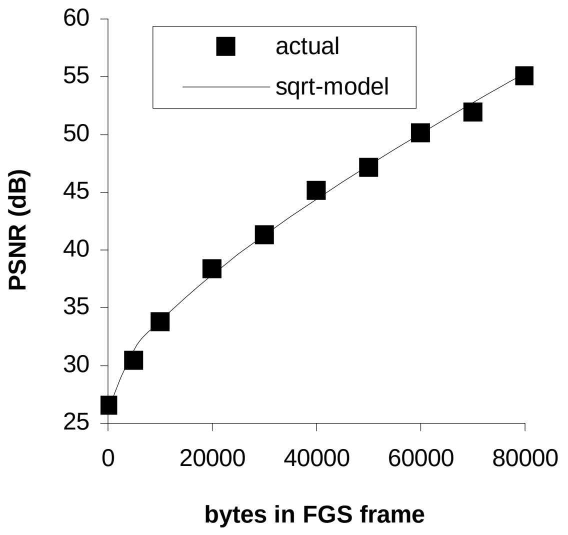
where constants A and B are estimated from at least two (R,D) samples, and $C = 10\log_{10}(255^2/\sigma_x^2)$ for uncorrelated (or weakly correlated) sources such as those in FGS coders. Parameter A and B are strongly negative-correlated (e.g., the 0-lap cross-correlation coefficient between these two parameters is -0.99 in the CIF Foreman sequence).

We next revisit two "difficult" PSNR curves shown earlier in Fig. 14, in which even a quadratic polynomial of R was unable to follow the curve. Fig. 34 shows the new result for the SQRT model (3.78) and demonstrates a much better fit than was possible before.

To better understand the estimation accuracy of the different models discussed so far, we compare the SQRT model (3.78), Chiang's model (3.28), the UQ model (3.24), and classical model (3.23) in various video sequences. Fig. 35 and Fig. 36 show the average absolute error between the actual R-D curve in the PSNR domain and each of the models in several FGS-coded sequences. For example, in the FGS-coded Foreman sequence, the error in SQRT averages 0.25 dB, while it stays as high as 2-8 dB in the other three models. Finally, note that we tested (3.78) in numerous other sequences, as well as at different base-layer bitrates, and found it to significantly outperform traditional models, which often required estimation of the same number



(a) Frame 39



(b) Frame 73

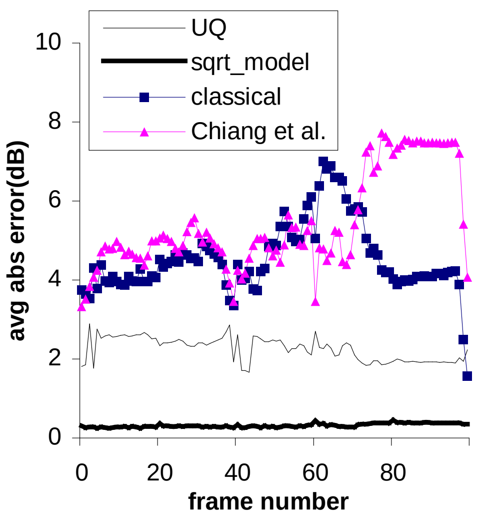
Fig. 34. (a) Frame 39 and (b) frame 73 of CIF Foreman fitted with the SQRT model.

of parameters.

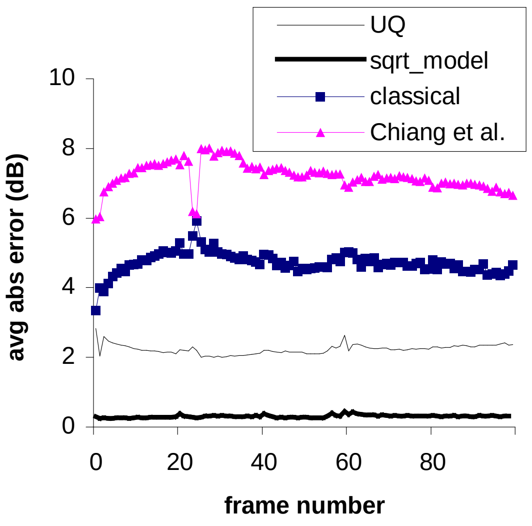
We also examined the accuracy of SQRT in PFGS. Recall that PFGS uses prediction in the enhancement layer to achieve better compression in sequences with high degrees of temporal correlation. Assuming that all predicted bits are transmitted to the client, our derivations and models are applicable to PFGS. Fig. 37 shows that model (3.78) outperforms the traditional R-D model in PFGS-coded sequences. The figure also shows that the UQ model and Chiang's model have large error variation in these sequences, which happens because PFGS not only uses the enhancement layer for prediction but also for reconstruction, which is beyond the range of the UQ model and Chiang's model.

We conclude this section by noting that (3.78) takes the following simple shape in the distortion domain:

$$D = c2^{aR+b\sqrt{R}}, \tag{3.79}$$



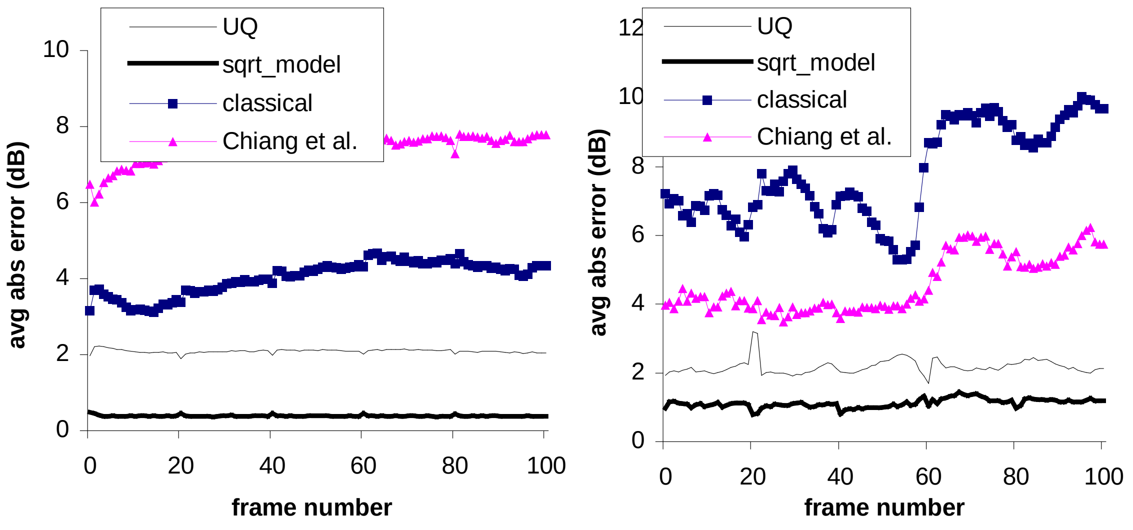
(a) Foreman



(b) Coastguard

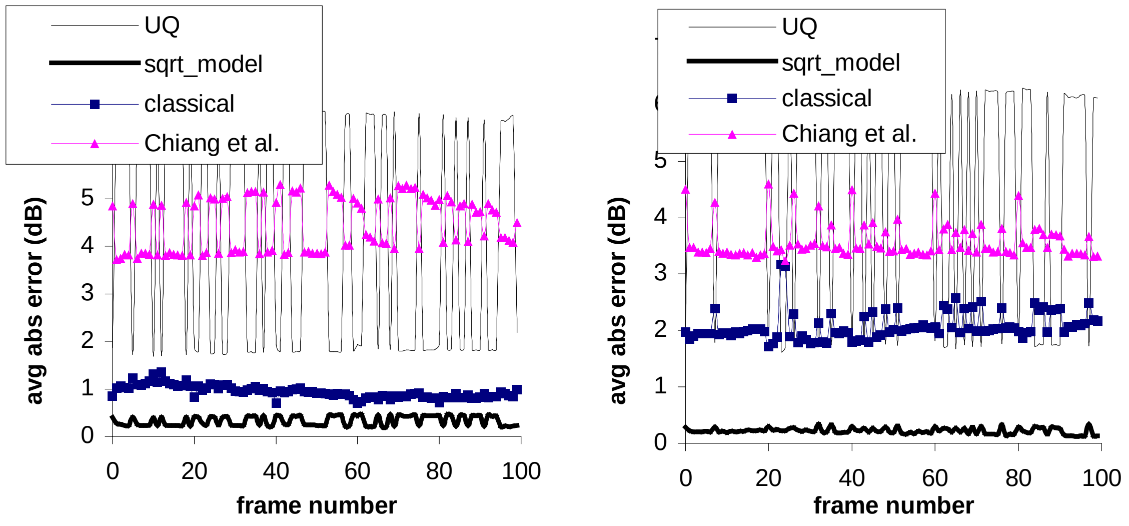
Fig. 35. Comparison between (3.78) and other models in FGS-coded (a) CIF Foreman and (b) CIF Coastguard, in terms of the average absolute error.

where $a < 0$, b are constants and and c is proportional to the source variance. This is a generalization of the traditional R-D function $D = c2^{-2R}$, in which $b = 0$.



(a) Mobile (b) Carphone

Fig. 36. Comparison between (3.78) and other models in FGS-coded (a) CIF Mobile and (b) CIF Carphone, in terms of the average absolute error.



(a) Mobile (b) Coastguard

Fig. 37. Comparison between (3.78) and other models in PFGS-coded (a) CIF Mobile and (b) CIF Coastguard, in terms of the average absolute error.

CHAPTER IV

QUALITY CONTROL FOR VIDEO STREAMING

Compared with the fully-download mode, video streaming has advantages such as short delay before playout and minimum storage requirements in servers. However, video streaming has strict QoS requirements at bandwidth, delay and packet loss, while the current best-effort network does not offer any QoS support. Thus, it is critical to design a scheme that can reliably deliver high-quality video over the Internet. This scheme is often referred to as QoS control, which includes congestion control and error control. Congestion control is developed to reduce packet loss and delay and error control is often employed to overcome the effect of packet loss or delay.

In this chapter, our purpose is to show how an R-D model can be coupled with congestion control to provide high quality video to end users under varying network conditions. After giving a brief survey on existing congestion and error control methods, we analyze a smooth controller and combine it with our proposed R-D model for quality control purposes during Internet streaming.

A. Related Work

1. Congestion Control

Due to the excessive delay in TCP transmission, UDP is usually employed as a replacement for TCP in video streaming and real-time video applications. UDP itself does not have any congestion control mechanism as TCP does and thus the quality of transmitted video heavily relies on network conditions.

Unfortunately, network congestion often causes bursty packet losses and excessive delay, which have devastating effects on video quality. Aside from packet loss and delay, the available bandwidth is often varying in real networks and the sending rate often needs to be adjusted according to it. While sending rate that is much higher than the available bandwidth will cause congestion, sending rate that is lower than it will result in low bandwidth utilization and sub-optimal video quality. As a result of these considerations, it is intuitive to implement a control mechanism on top of UDP to maximize video quality under various network conditions.

To prevent or at least reduce congestions, many congestion control schemes have been proposed and can be grouped into the following categories according to their network characteristics.

a. End-to-End vs. Router-Supported

Many congestion control schemes do not require additional support form the network. These schemes are called *end-to-end* congestion control approaches and can be further separated into *sender-based* and *receiver-based* approaches. In sender-based approaches, while the sender is responsible of using network information and adjusting the sending rate or window size, the receiver only provides feedback.

A receiver-based congestion control is often applied to layered multicast or multi-layer video streams. Under receiver-based control, the receiver regulates the receiving rate by subscribing or unsubscribing from additional layers according to the network situation. Thus, a receiver-based congestion control is often applied to layered multicast or multi-layer video streams.

End-to-end congestion control relies on the collaboration of the end systems; however, the collaboration is not always guaranteed. Unlike end-to-end congestion control, network-centric control needs additional support from networks, which adds burden on networks but greatly ease the design of effective congestion control schemes. It is important to some control schemes, e.g., multicast protocols benefit from addition network functionality, such as feedback aggregation, hierarchical RTT (round trip time) measurements, and group management of receivers.

b. Window-Based vs. Rate-Based

According to the way to transmit workload, congestion control mechanisms are classified into *window-based* and *rate-based.* Similar to TCP, in the window-based control algorithms, the window size decreases one slot when a packet is transmitted, and frees one slot when a packet is received. And the sender is allowed to transmit packets only when a free slot is available. The window size increases, when there is no congestion and decreases when congestion occurs.

A rate-based congestion control scheme dynamically adapts the transmission rate according to the network feedback. It can be further divided into AIMD-based and model-based schemes. In the former approaches, rate-based congestion control protocols mimic TCP's AIMD behavior to achieve TCP fairness, while model-based schemes adjust the sending rate according to a model of TCP traffic. AIMD-based rate schemes have similar results as TCP congestion control and result in a sawtooth-like rate, which is not suitable for continuous media streams. Compared with AIMD-based schemes, model-based congestion control produces much smoother rate by modeling TCP throughput and adapting the sending rate to the average long-term throughput of TCP [83].

In general, a rate-based congestion control scheme offers a smoother rate changes than a window-based one and is more suitable for video transmission over the Internet.

2. Error Control

Although the purpose of congestion control is to reduce packet loss, packet loss is unavoidable in real networks, and unfortunately, compressed video data is very sensitive to transmission errors. In current predictive coding based encoder, bit error or packet loss will cause error propagation within the same frame as well as in the following frames. Under this circumstance, *error control* mechanisms are often employed to overcome the effect of transmission errors.

Existing error control mechanisms can be classified into four types, namely, forward error correction (FEC), retransmission, error resilience, and error concealment. The first two are in channel coding category and the latter two are in source coding category.

a. Forward Error Correction (FEC)

The basic idea of FEC is to add redundant bits on compressed source bits to enable error detection and correction. In the Internet, redundant packets are transmitted so that the original message can be reconstructed even some packets are lost. For example, if there are K data packets, FEC will add $N-K$ redundant packets and the overhead is N/K. As long as any K of the N packets are correctly received, the original data can be recovered.

The big advantage of FEC is its small transmission delay; however, FEC is ineffective if there are more than $N-K$ consecutive packets lost (bursty error) in the above sample. To avoid this case, FEC is often combined with *interleaving* to spread out the lost packets. The larger interleaving depth, the stronger ability to overcome burst errors, but unfortunately, the larger delay.

In addition, the redundant transmission will add transmission burden and FEC may be poorly matched to channel, since channel loss characteristics are often unknown and time-varying. Since FEC is often ineffective (too little overhead) or inefficient (too much overhead), it is often implemented with *unequal error protection*, which uses stronger channel codes for more important bitstreams.

b. Retransmission

With the assumption that back-channel exists between a pair of receiver and sender, the receiver notifies the sender which packets were received/lost and the sender resends lost packets. This scheme is called *retransmission*. Retransmission efficiently uses bandwidth and easily adapts to changing channel conditions. However, retransmission requires a back-channel, which makes it unapplicable to broadcast, multicast, and unicast without back-channel. Given the back channel, retransmission triples the transmission time and thus this approach is effective only if the one-way trip is short.

Retransmission includes delay-constrained retransmission and priority-based retransmission. The former only retransmits packets that can arrive in time and the latter retransmits important packets before transmitting unimportant packets. In both cases, the sender needs to decide which packet should be transmitted next.

In Table IV, we briefly compare FEC and retransmissions,which are originally designed for reliable data delivery. Unlike them, the next two approaches are usually applicable only to video but not to general data types.

Table IV. Advantage and Disadvantages of FEC and Retransmission.

Categories	PRO	CON
FEC	Low delay, no feedback channel	Overhead, channel information required
Retransmission	High bandwidth utilization	Large latency, back-channel required

c. Error Resilient Coding

Error-resilient coding schemes are developed to mitigate the effect of packet losses or to prevent error propagation from compression perspective. Standardized error-resilient tools include *resynchronization marking*, *data partitioning*, and data recovery

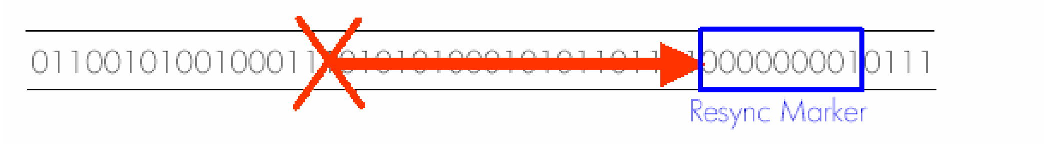


Fig. 38. The resynchronization marker in error resilience. Source: [2].

coding such as reversible variable length coding (RVLC).

Transmission errors most likely happen in two cases: the loss of bitstream synchronization and the error propagation at the decoder. In the first case, the decoder does not know what bits correspond to what parameters, e.g., a single bit error in VLC codeword can lead to significant subsequent loss. To deal with the first kind of error, *resynchronization marking* are often used, in which resync markers are placed periodically in the stream. As shown in Fig. 38, when synchronization loss happens, the corrupted bits are thrown away and the decoder can restart decoding after the resync marker.

Resync markers are designed to be distinct from all codewords, concatenations of codewords, and minor perturbations of concatenated codewords. Resync markers are inserted after *fixed number of blocks* in MPEG-1/2, H.261/3 and after *fixed number of bits* in MPEG-4. The latter way to place resync markers has several advantages over the former one: 1) It simplifies the searching for resync markers; 2) It supports network packetization, which is convenient for network delivery; 3) Since active areas may have more bits in their blocks, the latter scheme will put more resync markers in the corresponding part of bitstream and thus provides better protection to active areas.

Data partitioning is another commonly used method in error-resilience area. From extensive simulations, it is observed that bits closely following resync are more likely to be accurate than those farther away and thus data partitioning is proposed.

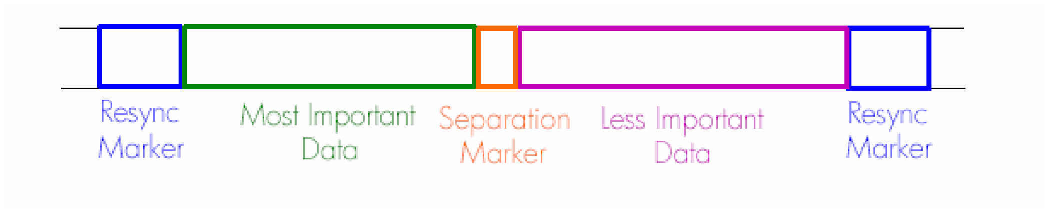


Fig. 39. Data partitioning in error resilience. Source: [2].

As shown in Fig. 39, data partitioning places the most important information (e.g., motion vectors, DC coefficients) immediately after resync markers and less important information (e.g., AC coefficients) later.

Different from resync markers and data partitioning, RVLC is designed from the coding perspective. Conventional VLC codes are decodable only in the forward direction; however, RVLC codes are designed to be also decodable in the backward direction. As shown in Fig. 40, if an error is detected, the decoder jumps to the next resync marker and starts decoding backwards, which enables partial recovery of the data that would be discarded.

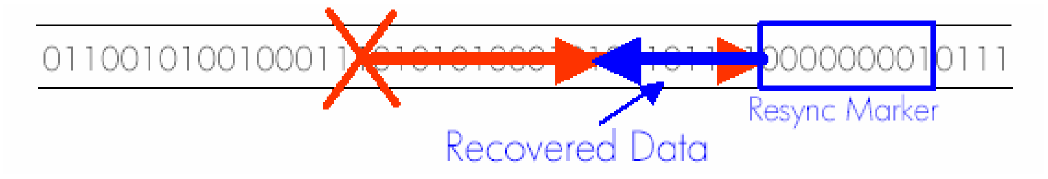


Fig. 40. The RVLC approach in error resilience. Source: [2].

The above standardized error-resilient tools are more suitable to bitwise-error environment such as wireless network and are not the most efficient methods for the packet-based network such as the Internet. For instance, the boundary of a packet already has the function of a resync marker in the VLC coded bitstream. Therefore, *optimal mode selection* and *multiple description coding* (MDC) are proposed recently

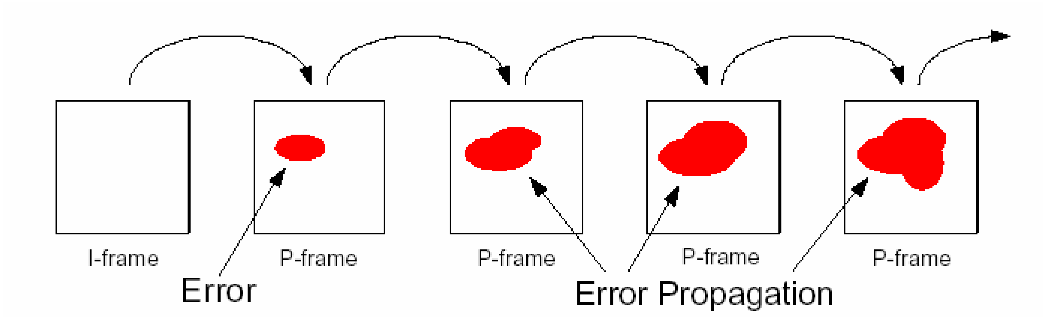


Fig. 41. The error propagation in error resilience. Source: [2].

[106], [108].

As mentioned earlier, error propagation is a major obstacle in predictive coding scheme applied in video coders. As shown in Fig. 41, when the reconstructed reference image at the decoder is different from the reference image at the encoder, incorrect (mismatched) predictions happen and often lead to significant error propagation in the subsequent frames. To limit the effect of error propagation, intra-coding is necessary in video coding; however, too many I-frames will significantly reduce compression efficiency.

As an alternative, the encoder also uses a sufficiently large number of intra-coded macroblocks (MB) in P-frames. There is a trade-off between coding efficiency and error robustness, and thus, how to decide the number and the locations of intra-coded MBs becomes an important issue, which is referred to as *optimal mode selection*. To maximize the video quality under the constraint of available network bandwidth, R-D optimized mode selection methods are often applied, which select the coding mode of MBs according to their R-D curves [108].

Besides optimal mode selection, multiple description coding (MDC) is another way to achieve tradeoff between compression efficiency and error robustness. As Fig. 42 shows, in MDC, a raw video sequence is compressed into multiple streams (de-

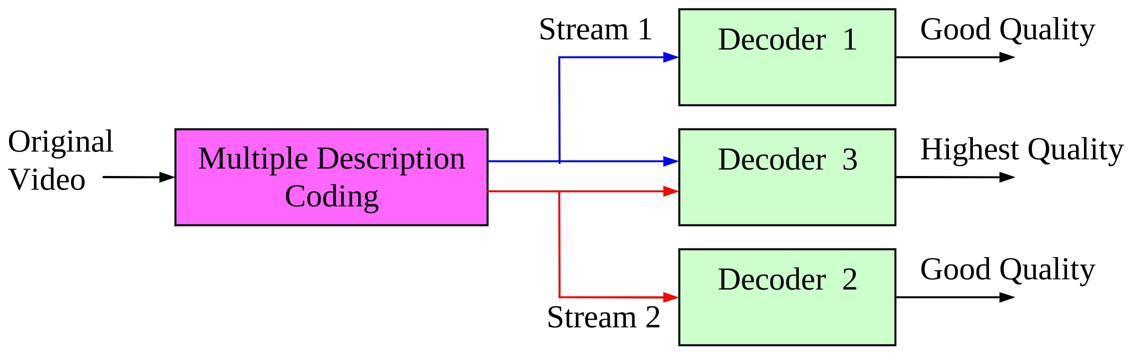


Fig. 42. The structure of multiple description coding. Source: [2].

scriptions), with roughly equal importance. This approach ensures that the decoder can reconstruct an image of acceptable visual quality even if only one description is received and will improve its quality if more descriptions are received. In the case of frame loss or corruption, the multiple description decoder will borrow the corresponding frame from another description, as shown in Fig. 43. Although MDC has strong error-resilient ability, it reduces compression efficiency compared with conventional single description coding.

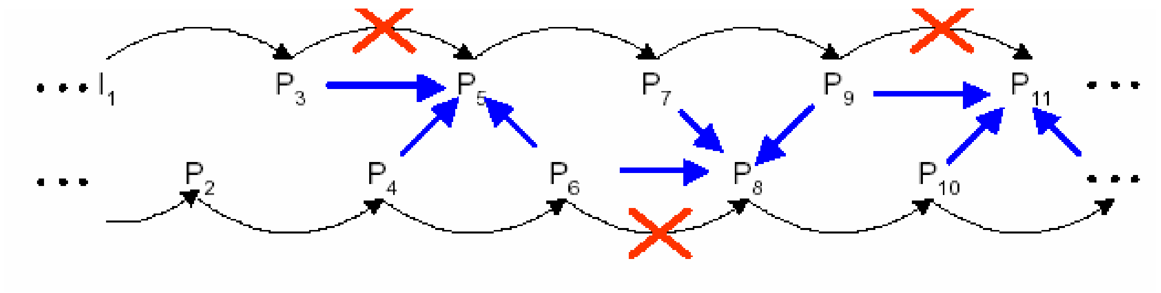


Fig. 43. The error-resilient process in multiple description coding. Source: [2].

d. Error Concealment

Error concealment, unlike error resilient methods, is a *postprocessing* technique executed only by decoders/receivers. Due to significant spatial and temporal correlation in video sequences, the error concealment mechanism performs some forms of spatial/temporal interpolation to estimate the lost information from the correctly received data.

From the spatial interpolation perspective, missing pixels are estimated by smoothly extrapolating surrounding pixels. From the temporal interpolation perspective, the lost MB is reconstructed from the corresponding MB in the previous frame. If there is no motion between the previous frame and the current one, the receiver directly copies the block from the corresponding one at the same spatial location of the previous frame. When loss occurs, usually a row of MBs or an entire frame are lost. In this case, a combination of spatial and temporal interpolation is necessary.

Error concealment offers a viable technique for coping with packet loss and can also be formulated as a signal recovery problem. There are many sophisticated algorithms in this area. Since error concealment is performed at the decoder, new algorithms can be incorporated as standard-compatible enhancements to conventional decoders.

B. Quality Control in Internet Streaming

To supplement the best-effort model of existing networks and to provide a high-quality streaming environment to end users, we study an R-D based quality control framework in this section and also discuss an asymptotically stable congestion controller.

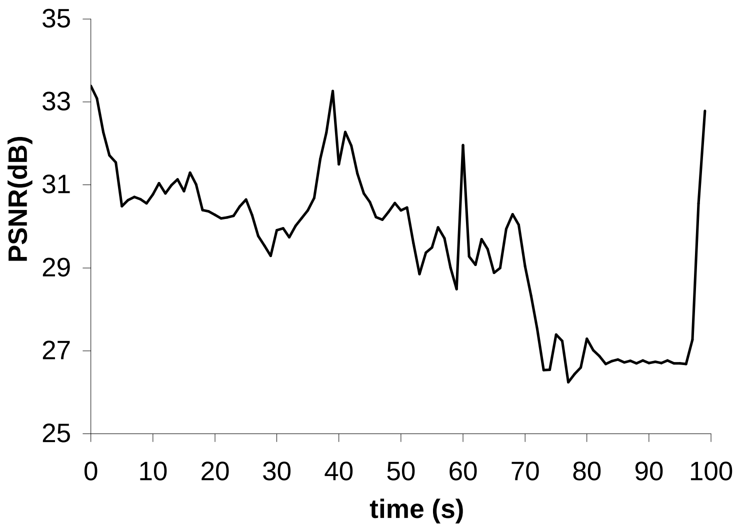


Fig. 44. Base layer quality of the CIF Foreman sequence.

1. Motivation

Although fluctuating visual quality is often unpleasant to end users, it is quite common in streaming applications due to the inherent nature of current video coding schemes and best-effort networks [113], [114]. We show an example in Fig. 44, which indicates a 6-dB drop in quality within just a 10-second fragment in Foreman CIF sequence.

Although scalable coding provides a flexibility for servers to decide transmitted bits during the streaming, how to properly rescale the enhancement layer is a challenging question. On the one hand, a proper rescaling method is critical to match the sending rate to the available bandwidth and user requirements. An R-D model is often applied to decide the transmitting portion of the enhancement layer, in order to make the best trade-off between the amount of transmitted bits and video quality. On the other hand, without a relatively stable network environment, even a proper rescaling method cannot provide high-quality video to end users, which necessitates congestion control in maintaining a stable network environment and avoiding wasting network resources in streaming applications. Therefore, by coupling R-D modeling with congestion control, the server can adjust its sending rate to match the available bandwidth in the network while keeping quality fluctuation as low as possible.

Notice that current congestion control methods built on top of a variety of TCP-friendly schemes cannot asymptotically converge (from a control-theoretic point of view) to a single stationary rate or provide a smooth "virtual" channel to the video application. The asymptotic stability refers to the capability to avoid oscillations in the steady-state and properly respond to external perturbations caused by any change of network condition [112].

After AIMD (Additive Increase, Multiplicative Decrease) has been found to be unacceptable for video streaming due to its large rate fluctuations, recent studies have developed several smooth congestion control control methods for multimedia streaming (e.g., TFRC [30] and binomial algorithms [3]). Unfortunately, these newly-developed methods are not asymptotically stable, nor do they have any stationary points in the operating range of typical applications [112].

Different from the above methods, some researchers model the network from an optimization or game-theoretic point of view [57], [58], [64]. Kelly *et al.* [58] propose a congestion control model from the angle of economic interpretation, where the entire system achieves its optimal performance if each end user maximizes its individual utility. Kelly's control is stable, efficient, and fair under various network conditions and has received significant attention in the theoretical networking community [58], [60], [75]. Thus, we select Kelly's control to achieve quality control purpose during streaming. However, our control scheme is independent of Kelly's control and can be combined with other smooth congestion controllers.

In what follows in this section, we discuss Kelly's control and its modification and then describe R-D based constant-quality control algorithms for both CBR and

VBR channels.

2. Kelly Controls

Before we study Kelly's control, we discuss the rate control function of TCP and classical binary-feedback methods. In general, these methods increase or decrease their rates as following:

$$\frac{dr}{dt} = (1 - sgn(p))F(r) - sgn(p)G(r), \tag{4.1}$$

where $sgn(\cdot)$ is the sign function, $r(t)$ is the rate at time t, $p(t)$ is packet loss, $F(r)$ is the increase function, and $G(r)$ is the decrease function. Under certain conditions on $F(r)$ and $G(r)$, (4.1) oscillates around the equilibrium (equal-share) rate and typically leads to a trade-off between the oscillating range and the feedback of packet loss. Usually, controls that produce small oscillations are susceptible to more packet loss due to their reluctance to back off during congestion.

Notice that the right side of (4.1) does not have roots with certain format of $F(r)$ and $G(r)$, which means that the equation does not have stationary points in some cases. Since binary-feedback methods cannot be asymptotically stable even under *stationary* cross-traffic conditions, we seek alternative methods that are provably stable under both immediate and *delayed* feedbacks. One such alternative is given by Kelly's congestion control framework called *proportional fairness* [58]:

$$\frac{dr}{dt} = r(\alpha U'(r) - \beta \sum_{l \in P} p_l), \tag{4.2}$$

where $U(r) = \log r$ is the utility function of the end user, $\alpha > 0$ and $\beta > 0$ are constants, and p_l is the price that the flow pays for using resource (router) l along the end-to-end path P.

Although Kelly's control has been proven to be stable and efficient, several aspects of the the original framework (4.2) make this controller impractical and a few clarifications are necessary to make it application-friendly. First, the current Internet is best-effort and prices are a meaningless metric for individual routers. The solution to this problem is to use packet loss instead of the price as the feedback from the network. Second, instead of summing up the packet loss experienced by *all* routers of an end-to-end path, it makes more sense to use the *maximum* packet loss among these routers to match the rate of the application to the bandwidth of the *slowest* link in the path:

$$p(t) = \max_{l \in P} p_l. \tag{4.3}$$

Expanding (4.2) using a single feedback $p(t)$ of the most-congested resource or the standard end-to-end feedback, we have a more application-friendly version of the controller:

$$\frac{dr}{dt} = \alpha - \beta p(t) r(t), \tag{4.4}$$

Since the rate adjustment of (4.2) is not continuous, the classic Kelly's control is proved to be globally stable only in the absence of feedback delay. However, feedback delays are very possible to appear in a control loop and are heterogeneous. Therefore, major modifications have to be applied to Kelly's control to assure its asymptotical stability in real networks with a user-friendly format. In light of these considerations, it is natural to add delay to the classic Kelly's control and prove its asymptotically stability in the modified version. Thus, Zhang *et al.* [112] consider several different delays that might encounter in the control scheme and propose a modified version of Kelly's control, *Max-min Kelly Control* (MKC):

$$r_i(t) = r_i(t - D_i) + \alpha - \beta r(t - D_i) p(t - D_{li}^{\leftarrow}), \tag{4.5}$$

where i is the flow number, feedback $p(t)$ is calculated using (4.3), D_i is its round-trip delay, and $D_{li}^{\leftarrow}$ is the backward feedback delay from router l to user i. Note that this version of Kelly's control includes novel max-min changes to the feedback and an extra delay applied to the additive term $r_i(t - D_i)$ in (4.5). Full analysis of this framework is referred to [112], and we only illustrate several important characteristics of this controller.

Lemma 6 *Discrete controller (4.3)-(4.5) is asymptotically stable and fair regardless of round-trip delays D_i, the exact shape of packet loss $p(t)$, or feedback delays $D_{li}^{\leftarrow}$ as long as* $0 < \beta < 2$.

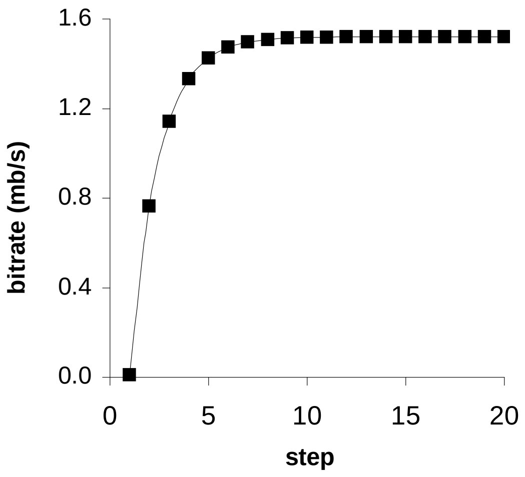
PROOF: See [112]. ■

While (4.3)-(4.5) can operate in the end-to-end context where $p(t)$ is estimated by the receiver, we find that involvement of AQM (Active Queue Management) significantly improves the performance of this controller. In that case, each router counts the total arriving traffic into each queue, divides the result by the fixed duration of the control interval, and inserts feedback $p_l(t)$ into all passing packets:

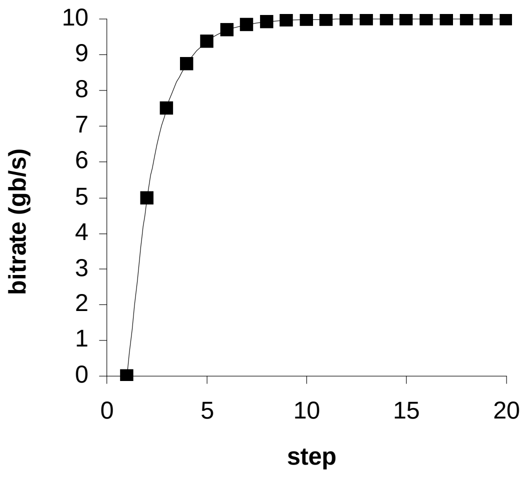
$$p_l(t) = \frac{\sum_{i \in S_l} r_i(t) - C_l}{\sum_{i \in S_l} r_i(t)}, \tag{4.6}$$

where S_l is the set of flows passing through resource l and C_l is the speed of the resource (i.e., its outgoing bandwidth).

To calculate p_l, each router records the total number of bytes placed in the outgoing buffer during the last T time units. At the end of each interval, this counter is divided by T to obtain an estimate of $\sum_{i \in S_l} r_i(t)$, which is then used to calculate p_l using (4.6). The new value of p_l is inserted into each passing packet as long as the corresponding p_{l-1} contained in the packet is *lower* than the value computed by this router. Notice that the router does not need to count the number of flows or estimate their individual rates r_i. This means that the feedback is based on the *aggregate* flow



(a)



(b)

Fig. 45. Exponential convergence of rates for (a) $C = 1.5$ mb/s and (b) $C = 10$ gb/s.

rate $\sum_{i \in S_l} r_i(t)$ rather than on individual flow rates. This in general increases the scalability of these AQM functions inside each router. For additional implementation discussion, see [56].

It is also possible to demonstrate that the convergence rate of Kelly controls is at least exponential, which makes this framework appealing for future very high-speed networks.

Lemma 7 *Under AQM feedback in (4.6), controller (4.3)-(4.5) reaches link utilization exponentially fast.*

PROOF: See [112]. ■

The result of this lemma is illustrated in Fig. 45, in which $\beta = 0.5$ and $\alpha = 10$ kb/s. The figure shows that it takes 8 steps for a single-flow to fill a 1.5 mb/s T1 bottleneck and it takes only 16 steps for the same flow to fill a 10 gb/s link. Note that both flows reach within 5% of C in just 6 steps.

3. Quality Control in CBR Channel

After we obtain a stable and smooth congestion control method, we proceed to present its application to quality control algorithms.

In CBR channels or a channel with predictable bit rate, the challenging question is how to scale the FGS layer to match the available bandwidth R_T (total amount of bits allowed for the entire sequence) while keeping *constant* quality to end users. Notice that only the fine granular scalable streams can be arbitrarily rescaled according to the feedback from the congestion controller and it is relatively hard to achieve constant quality for streams coded with coarse granular scalable coders.

We illustrate the solution to this problem using a simple sequence consisting of two frames, given the target rate R_T and the constant quality (distortion) D_T. As shown in Fig. 46, the server first inverts the result in (3.78) or (3.79) and obtains two $R(D)$ curves (one for each frame). Second, it generates the combined rate curve $R_1(D) + R_2(D)$, which shows the amount of *total* bits required to achieve constant D in both frames. Given R_T, the combined curve needs to be inverted one more time to obtain the value of D_T that provides the required total bitrate R_T. The size of individual frames is given by $R_1(D_T)$ and $R_2(D_T)$ as the final step.

For longer sequences, the server adds the R-D curves of all frames and obtains a combined function $F(D)$, which is constrained by R_T:

$$F(D_T) = \sum_{i=t}^{N} R_i(D_T) = R_T, \tag{4.7}$$

where $R_i(D)$ is the R-D function of frame i, N is the number of frames in the sequence, and t the time at which the server decides to change its rate R_T in response to congestion signals. Partial summation in (4.7) is important since congestion control often changes its rate in the middle of actual streaming and (4.7) needs to be recom-

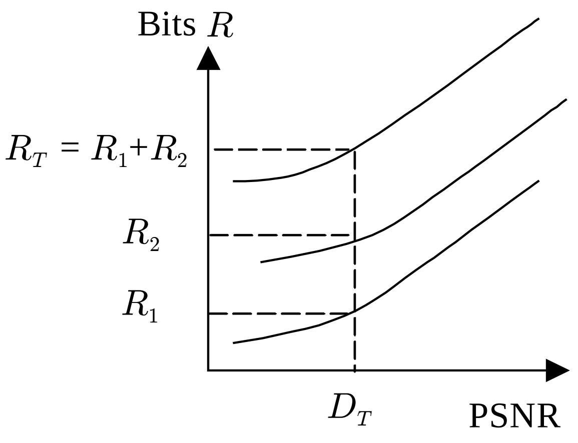


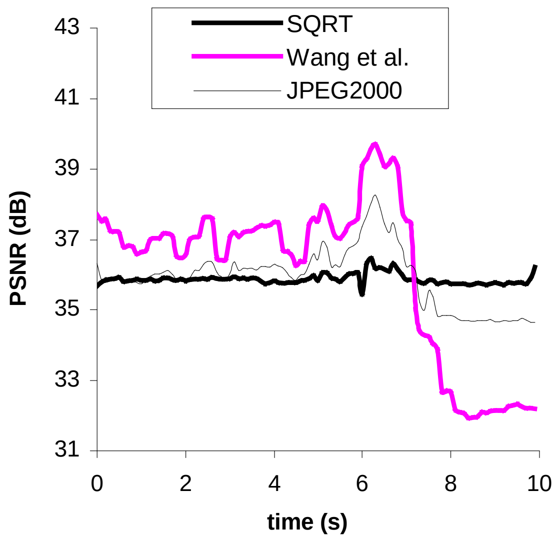
Fig. 46. The R-D curves in a two-frames case.

puted every time such a change is encountered. Finding the root of (4.7) involves inverting $F(D)$ and evaluating

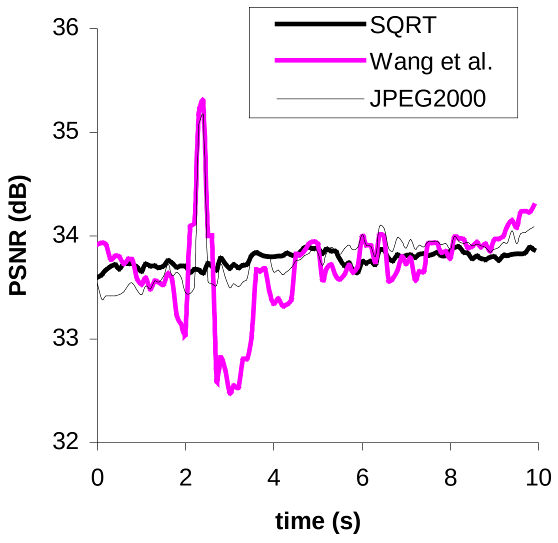
$$D_T = F^{-1}(R_T). \tag{4.8}$$

Once D_T is known, each enhancement layer frame i is scaled to $R_i(D_T)$ and then transmitted to the receiver. Even if there is probably no closed-form solution for F^{-1}, each R-D curve can be generated with high accuracy using only a 3-point interpolation and thus the resulting function $F(D)$ can be computed (and then inverted) very efficiently.

In Fig. 47, we illustrate the simulation results of this R-D based quality control algorithm assuming that the channel capacity is fixed (variable channel rates are studied in the next subsection). The figure shows simulation results using Foreman CIF with 768 kb/s available in the network for the enhancement layer in comparison with two other rate-control methods – those proposed in the JPEG2000 [55] image coding standard and in Wang *et al.* [105]. Experimental results show that the proposed R-D framework can be successfully used to both dramatically reduce undesirable quality fluctuation during streaming and to relieve the server from expensive interpolation.



(a)



(b)

Fig. 47. Comparison in CBR streaming between our R-D model, the method from [105], and rate control in JPEG2000 [55] in (a) CIF Foreman and (b) CIF Coastguard.

The variance in PSNR between adjacent frames in the SQRT curve is only 0.04 dB in Fig. 47 (a) and 0.004 dB in Fig. 47 (b).

During this study, we find that most constant quality control approaches stop at the CBR case [105], [113], [114], which makes the previous work almost unapplicable to real networks. Hence, we feel that an important research direction in the constant quality of video streaming is to develop an algorithm on top of a proper congestion controller, e.g., a Max-min Kelly's controller.

4. Quality Control in VBR Networks

The combination of an R-D model with the Max-min Kelly's controller is quite straightforward. The sending rate is the smaller rate between the controller's decision (4.5) and the result of the R-D curve (4.8). Unlike the CBR case, the target rate R_T is not known a-priori but is rather supplied by real-time congestion control and keeps varying during streaming.

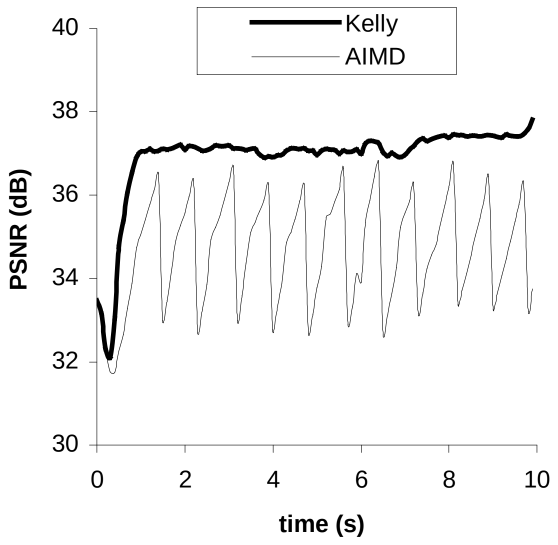
In what follows, we show simulation results to better understand this scenario. We obtained the traces of $r(t)$ from ns2 simulations and then applied them to the video scaling algorithm offline.

To set a baseline example, in Fig. 48 (a), we compare the AIMD (1, 0.5) control with the modified framework (4.5) using PSNR quality curves. In this simulation, a single flow is run over a bottleneck resource of capacity $C = 1$ mb/s (the round-trip delay is 100 ms). As the figure shows, both controls at first follow the PSNR of the base layer, since there is no enough discovered bandwidth to send any FGS data. Once this stage is passed, both controls achieve high PSNR; however, the difference is that AIMD backs off by half upon every packet loss, while Kelly controls eventually stabilize at a fixed rate. Rate fluctuation in AIMD results in periodic jumps (sometimes as high as 4 dB) throughout the entire sequence.

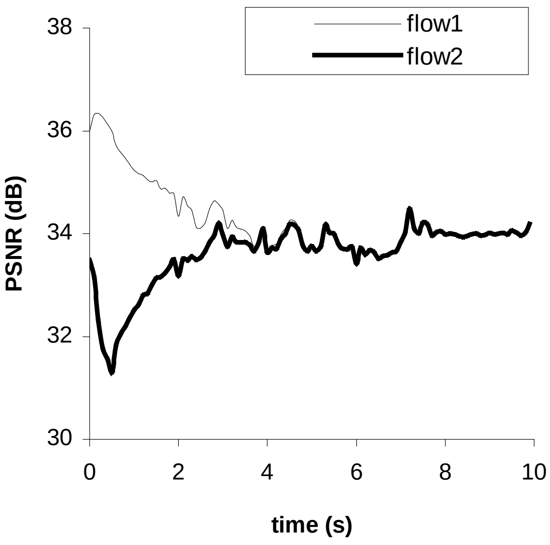
Fig. 48 (b) shows another scenario where two Kelly flows are sharing the same bottleneck link C under identical 100-ms round-trip delays. Flow$_1$ in the figure is started with $r_1(0) = C$ and flow$_2$ is started with its base-layer bandwidth. As seen in the figure, the two flows converge to a fair allocation at approximately $t = 3$ seconds and then follow the same flat quality curve.

The next issue to examine is whether different round-trip delays D have any effect on fairness. Fig. 49 (a) shows a scenario in which two flows with different RTTs start in the same unfair states as before. The corresponding delays are 400 and 100 ms; however, this has little effect on the resulting fairness as both flows stabilize at 34.5 dB around $t = 7$ seconds.

We also examine the effect of *random* feedback delays on our quality-control framework, in which the round-trip delay is uniformly distributed between 100 and



(a)

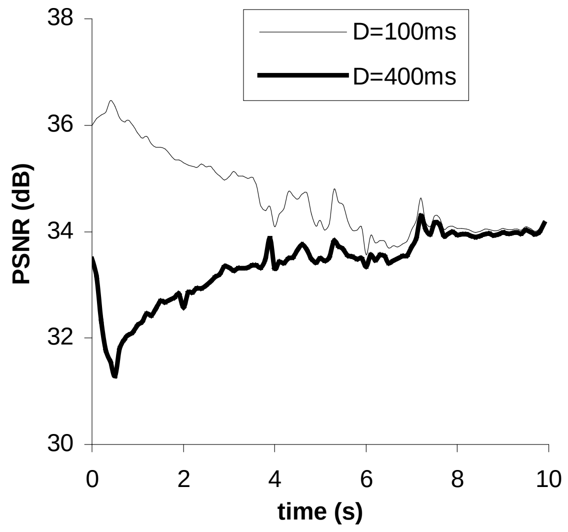


(b)

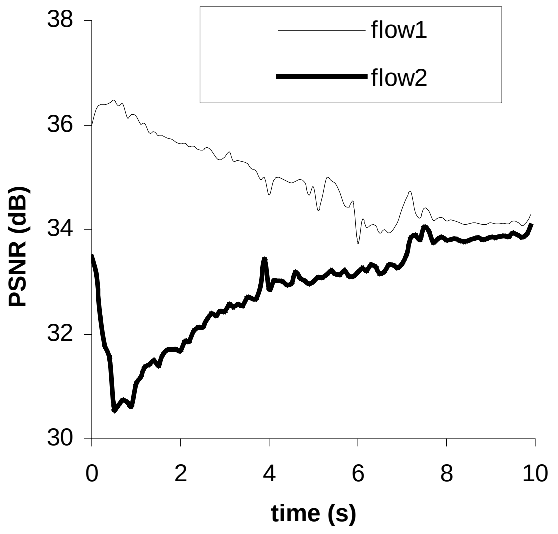
Fig. 48. (a) Comparison of AIMD and Kelly controls over a 1 mb/s bottleneck link. (b) Kelly controls with two flows starting in unfair states.

400 ms and the initial states are as before. Fig. 49 (b) shows that although the convergence is somewhat slower than in the previous examples ($t = 8$ seconds), both flows manage to provide a stable quality after the convergence. This confirms our earlier result regarding stability of (4.3)-(4.5) under arbitrary delays.

Finally, we examine the case of $n = 10$ flows over a bottleneck $C = 10$ mb/s. In this case, one flow initially occupies the whole bandwidth and then 9 other flows enter the path. All delays are random numbers between 100 and 400 ms, as shown in Fig. 50(a). Fig. 50(b) shows the trajectory of one (randomly selected) flow. As the figure shows, at first only the base layer is transmitted, but starting at $t = 2$ seconds, the FGS layer "kicks in" and the flow smoothly converges to 37 dB without any oscillations. The time to stabilize at 37 dB is approximately 9.5 seconds, which appears to be reasonable under many streaming conditions.



(a)

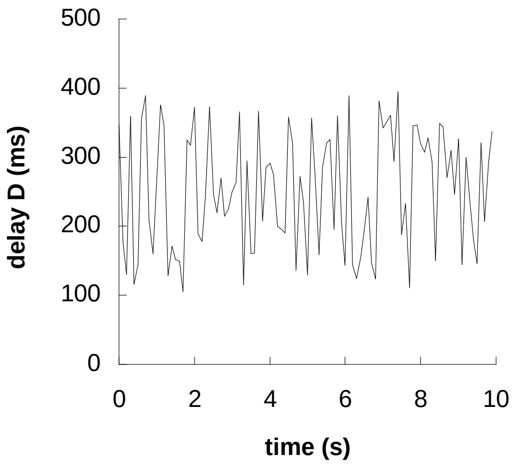


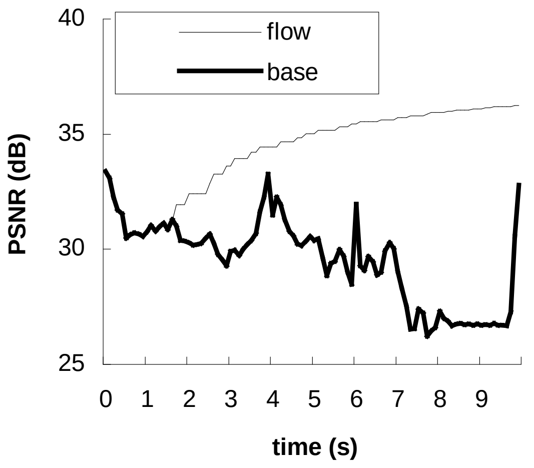
(b)

Fig. 49. PSNR comparison of (a) two flows with different (but fixed) round-trip delays D and (b) two flows with random round-trip delays.

In summary, Kelly controls converge to equilibrium without oscillation and then stay there as long as the number of flows at the bottleneck remains fixed. When new flows join or leave, the transition between fair (equilibrium) points is monotonic in most situations. This provides a nice foundation for video-on-demand and other entertainment-oriented video services where each flow is long-lived and can take full advantage of this smooth congestion control framework.

One limitation of this approach is that we assume the transmitted packets are protected and do not take into account the effect of lost packets during the simulation in this section. This is reasonable since in Kelly controls, the amount of packet loss p^* in the steady state is fixed and known to the end flow once it reaches the equilibrium [112].





(a) (b)

Fig. 50. (a) Random delay D for the flow. (b) A single-flow PSNR when $n = 10$ flows share a 10 mb/s bottleneck link.

5. Related Error Control Mechanism

One special characteristic of scalable video streams is that they often carry information of different importances. In all layered video coding schemes, the higher sections of the enhancement layer cannot be decoded until the base layer and the lower sections are received and decoded. However, the current best-effort Internet transmits all packets with *equal* importance, which conflicts with the heterogeneous nature of video packets. In the worst case, the bottleneck link may transmit a large number of packets that are useless and eventually get dropped by the decoder.

To resolve this difficulty, significant research has been done to supplement the best-effort Internet. While one direction of the related work offers QoS guarantees to end flows in the form of DiffServ [7], [10] or IntServ [8], others employ Active Queue Management (AQM) that performs special operations in the router to achieve better performance for end flows [19], [26]. These schemes either focus on providing fairness to competing flows [98], or attempt to avoid congestion by randomly dropping/marking packets with a probability proportional to the level of congestion [28], [29].

Nevertheless, none of these methods provide a scalable, low-overhead, and low-delay platform for streaming applications, and thus Kang *et al.* [56] propose a *Partitioned Enhancement Layer Streaming* (PELS) framework to provide optimal video quality in best-effort networks. In this framework, the base layer is marked as *green* and the enhancement layer is partitioned into *yellow* and *red* packets. The green packets have the highest priority, then the yellow ones, and the red packets have the lowest priority. The lower priority a packet has, the higher risk that it will be dropped. The base layer is the most important because it is the prerequisite to decode the enhancement layer, and the higher portion of the enhancement layer can not be encoded until both the lower portion and the base layer are encoded. During streaming, the server probes for the available bandwidth and adjusts the sending portions of packets of different priorities.

At the first glance, this framework looks like a combination of a congestion controller and a three-color marker (TCM) that gives packets different priorities. The most significant difference between the PELS framework and previous work is that it has closed-form expressions for the selection of red packets in the enhancement layer and the penalty inflicted on scalable traffic flows under uniform packet loss. In addition, this framework makes full usage of available bandwidth and guarantees the usability of received packets at the decoder. Readers are referred to [56] for detailed discussion.

CHAPTER V

TRAFFIC MODELING

Video traffic modeling plays an important role in the characterization and analysis of network traffic. Besides providing an insight into the coding process and structure of video traffic, traffic models can later be used for many practical purposes including allocation of network resources, design of efficient streaming networks, and delivery of certain Quality of Service (QoS) guarantees to end users.

To achieve the above goals, a traffic model should capture the important characteristics of video sequences, which often refers to the distribution and the autocorrelation function (ACF) of frame sizes. Several models have been proposed for the frame-size distribution, including the lognormal [59], Gamma [95], and various hybrid distributions (e.g., Gamma/Pareto [68] or Gamma/lognormal [92]). Compared to modeling the frame-size distribution, capturing the ACF structure of VBR video traffic is more challenging due to the fact that VBR video exhibits both LRD and SRD properties [32], [72]. The coexistence of SRD and LRD indicates that the ACF structure of video traffic is similar to that of SRD processes at small time lags and to that of LRD processes at large time lags [32]. Thus, using either a long-range dependent or a short-range dependent model *alone* does not provide satisfactory results.

Plenty of work has addressed the challenge of accurately capturing the ACF structure, but only a few of them have managed to model the complicated LRD/SRD ACF structure of real video traffic (e.g., [68], [72]). Furthermore, the correlation that most models try to capture is the *inter-GOP* correlation, which is well characterized by the ACF of the I-frames. However, another dimension of video traffic, the *intra-*

GOP correlation[1], is rarely addressed in related work, even though it is an important characteristic useful in computing precise bounds on network packet loss [63].

On the other hand, although many studies have been conducted in this area, most existing traffic models only apply to single-layer VBR video and often overlook the multi-layer aspects of common streaming video traffic in the current Internet [9], [115]. In addition, research on traffic modeling is falling behind the rapid advances in video techniques, e.g., there is no traffic model for sequences coded with the most recent coding technique H.26L.

Therefore, the goal of our work is to better understand the statistical properties of various video sequences and to develop a model that can generate synthetic traffic with the properties close to those of original single/multi-layer MPEG-4 and H.26L video sequences. Notice that video sequence could be constant-bit-rate (CBR) encoded or variable-bit-rate (VBR) encoded. Although the CBR encoding has almost constant output bit rate of the encoder, its video quality has severe fluctuation. In contrast, VBR-coded streams have less quality variation and thus are more common in multimedia applications.

In the remainder of this chapter, we briefly overview the related work on traffic modeling in Section A. In Section B, we provide the background on wavelet analysis and show how to generate synthetic I-frame sizes in the wavelet domain. In Section C, we discuss the intra-GOP correlation in various sequences and present a linear model for P and B-frame sizes. Section D analyzes the cross-correlation between the base layer and the enhancement layer, and explains how to generate a synthetic enhancement layer based on the cross-correlation. In Section E, we evaluate the accuracy of our model using both single-layer and multi-layer video traffic.

[1]The correlation between P/B-frames and the I-frame in the same GOP.

The specifics of the four sequences discussed in this chapter are as following: a single layer MPEG-4 *Star Wars IV* [27] (25 frames/s), a single layer H.26L *Starship Troopers* [87] (25 frames/s), a two-layer spatially-scalable *The Silence of the Lambs* [87] (30 frames/s), and a two-layer FGS-coded *Star Wars IV* [87] (30 frames/s). All four sequences have GOP structure *IBBPBBPBBPBB*.

A. Related Work on VBR Traffic Modeling

In this section, we provide a brief overview of related work on single-layer and multi-layer models.

1. Single Layer Video Traffic

Numerous studies have been conducted in modeling VBR video traffic. According to the dominant stochastic method applied in each model, we group them into five categories: autoregressive (AR) models [31], [59], [42], [68], Markov-modulated models [62], [95], self-similar (fractal) models [32], [44], wavelet-based methods [72], [90], and other approaches [76].

a. Autoregressive (AR) Models

AR models are considered as a classical approach in the area of traffic modeling. An AR process of order k is expressed as:

$$x(n) = a_0 + \sum_{i=1}^{k} a_i(x)(n-i) + e(n), \qquad n = k+1, \cdots, N \tag{5.1}$$

where a_0 is a constant, $\{a_i, 1 \leq i \leq p\}$ are AR coefficients, and $\{e(n)\}$ is an uncorrelated process with zero mean and variance σ^2. An AR process of order p is denoted by $AR(p)$. Approaches that are used to estimate the coefficients for an AR process include Yule-Walker estimation, Levinson-Durbin algorithm, maximum-likelihood estimation, and least-square estimation.

After the first AR model was applied to video traffic in 1988 [73], AR processes and their variations remain highly popular in this area of research [68]. Although AR(1) model is simple, its performance is not satisfactory in many cases and many of its variations have been proposed. For example, Corte *et al.* [16] use a linear combination of two AR(1) processes to model the ACF of the original video traffic, in which one AR(1) model is used for modeling small lags and the other one for large lags.

Since using a single AR process is generally preferred, Krunz *et al.* [59] model the deviation of I-frame sizes from their mean in each scene using an AR(2) process. Building upon Krunz' work [59], Liu *et al.* [68] propose a *nested* AR(2) model, which uses a second AR(2) process to model the mean frame-size of each scene.

In [41], Heyman *et al.* the number of ATM cells per frame is modeled based on a Markov-chain, whose transition probabilities are estimated by a *discrete* AR(1) model. This framework is suitable for video conference sequences with no significant scene changes and moderate motion. In addition, parameter estimation and other calculations are non-trivial burden in this model.

To reduce the computational cost, Heyman [42] propose a gamma-beta autoregressive (GBAR) model, which is an AR model with with Gamma-distributed marginal statistics and a geometric autocorrelation. Compared with model in [41], the parameters of this model are easy to estimate. However, it only intends to model video teleconferencing and does not consider the group-of-picture (GOP) cyclic structure of video traffic. Since GOP structure is typical in recent video standards, Frey *et al.* [31] extend the GBAR model in [42] to the GOP-GBAR model.

b. Markov-modulated Models

Markov-modulated models employ Markov chains to create other processes (e.g., the Bernoulli process [62]). Rose [93] uses nested Markov chains to model GOP sizes. Since synthetic data is generated at the GOP level, this model actually coarsens the time scale and thus is not suitable for high-speed networks. Chen *et al.* [12] use a doubly Markov modulated punctured AR model, in which a nested Markov process describes the transition between the different states and an AR process describes the frame size at each state. The computation complexity of this method is quite high due to the combination of a doubly Markov model and an AR process. Sarkar *et al.* [95] propose two Markov-modulated Gamma-based algorithms. At each state of the Markov chain, the sizes of I, P, and B-frames are generated as Gamma-distributed random variables with different sets of parameters. Although Markov-modulated models can capture the LRD of video traffic, it is difficult to accurately define and segment video sources into the different states in the time domain due to the dynamic nature of video traffic [72].

c. Models Based on Self-similar Process

A simple explanation of self-similar process is that the samples for that process look "roughly" the same on any time scale. Hurst discovered self-similarity in an investigation of the amount of storage required in the Great Lakes of the Nile river basin [46]. Fractals are a particularly interesting class of self-similar objects. Self-similar objects with parameters N and s are described by a power law such as

$$N = s^H, \tag{5.2}$$

where H is called Hurst parameter and is expressed as:

$$H = \frac{log(N)}{log(n)}. \tag{5.3}$$

Once H is estimated, a process such as fractional ARIMA (Autoregressive Integrated Moving Average) or ffGN (fast fractional Gaussian noise) is used to create a background sequence, which will be used to generate the foreground sequence using the desired empirical marginal bitrate distribution.

Garrett *et al.* [32] propose a fractional ARIMA model to replicate the LRD properties of compressed sequences, but do not provide an explicit model for the SRD structure of video traffic. Using the results of [32], Huang *et al.* [44] present a self-similar fractal traffic model; however, this model does not capture the multi-timescale variations in video traffic [59].

d. Other Models

The above problem can be overcome using the Transform-Expand-Sample (TES) method [76]. This method generates a background process $\{U_n\}$ and uses $\{U_n\}$ to generate foreground process $\{X_n\}$ by a transformation. The process $\{U_n\}$ defines a random walk on the unit circle based on an operator that is defined as $< x >= x - [x]$. Specifically, process $\{U_n\}$ includes $\{U_n^+\}$ and $\{U_n^-\}$, which are defined as,

$$U_n^+ = \begin{cases} U_0, & n = 0 \\ < U_{n-1}^+ + V_n >, & n > 0 \end{cases} \quad U_n^- = \begin{cases} U_n^+, & \text{n even} \\ 1 - U_n^+, & \text{n odd} \end{cases}, \tag{5.4}$$

where U_0 is uniformly distributed on the interval $[0, 1)$ and $\{V_n\}$, called the innovation sequence, is determined by:

$$V_n = L + (R - L)Z_n, \tag{5.5}$$

where $-0.5 \leq L < R < 0.5$ and Z_n is *i.i.d.* uniformly distributed random variable on interval $[0, 1)$.

Although this method is accurate in matching the ACF at both small and large lags, it has high computational complexity and often must be used in special software (e.g., *TEStool*) that generates synthetic sequences.

Different from the above time-domain methods, several wavelet models [71], [72], [90] recently emerged due to their ability to accurately capture both LRD and SRD properties of video traffic [72]. It has been proven that wavelets can capture the LRD and are used to estimate the Hurst parameter in a fractional Brownian motion (fBm) processes, which is often employed in traffic modeling [1], [24]. However, wavelets are also able to capture the short-term correlation [72]. We give more explanation of wavelets in the following section.

2. Scalable Video Traffic

All models discussed above focus on single-layer video traffic and only a handful of studies analyze multi-layer sequences. For example, Chandra *et al.* [9] use a finite-state Markov chain to model one- and two-layer video traffic of all activity levels. They assume that only one I-frame exists in the whole video sequence and the I-frame size is simply an *i.i.d.* Gaussian random variable. The model clusters P-frame sizes into K states according to the correlation between successive P-frame sizes and uses a first-order AR process to model the frame size in each state. The goal of [9] is to model one or two-layer video traffic with a CBR base layer, while many multi-layer video sequences have *more* than two layers and the base-layer is VBR.

Similarly to the work in [9], Zhao *et al.* [115] build a K-state Markov chain based on frame-size clusters. The clustering feature in [115] is the cross-correlation between the frame size of the base layer and that of the enhancement layer at the same frame index. In each state of the Markov chain, the base and the enhancement-layer frame sizes follow a multivariate normal distribution. However, the computational cost of the hierarchical clustering approach applied in [115] limits its application to short video sequences. Furthermore, in both [9] and [115], there is no general method for choosing the optimal number of states and the parameters are often chosen empirically.

Next, we will address the modeling of I-frame sizes and show a novel method for estimating the coefficients of the wavelet transform.

B. Modeling I-Frame Sizes in Single-Layer Traffic

In this section, we generate the synthetic I-frame sizes using the estimated wavelet coefficients, which preserve the LRD and SRD properties of the original traffic. There are two contributions to our framework discussed below: (1) we show a novel method for estimating the coefficients of the wavelet transform, which is both efficient and accurate; and (2) we model the intra-GOP correlation and propose a simple model that accurately generates synthetic P-frame sizes, which is in contrast to much of the previous work that relied on *i.i.d.* random variables to model the sizes of P/B-frame sizes in each GOP [59], [44], [68], [95].

1. Wavelet Models and Preliminaries

Wavelet analysis is typically based on the decomposition of a signal using an orthonormal family of basis functions, which includes a high-pass *wavelet* function and a low-pass *scaling* filter. The former generates the *detailed* coefficients, while the latter produces the *approximation* coefficients of the original signal. The wavelet transform strongly reduces the temporal correlation in the input signal, which means that signals with LRD properties produce short-range dependent wavelet coefficients [72].

In order to understand the structure of the wavelet transform, we next examine the relationship between the original signal and the detailed and approximation coefficients. We use the Haar wavelet transform as a typical example since it is often chosen for its simplicity and good performance [72], [90].

In the following discussion, we define $\{A_j\}$ to be the random process modeling approximation coefficients A_j^k and $\{D_j\}$ to be the process modeling detailed coefficients D_j^k at the wavelet decomposition level j, where k is the spatial location of A_j^k and D_j^k. We also assume that $j = J$ is the coarsest scale and $j = 0$ is the original signal. Recall that Recall that the Haar scaling and wavelet functions are, respectively:

$$\varphi(t) = \begin{cases} 1 & 0 \le t < 1 \\ 0 & \text{otherwise} \end{cases}, \quad \psi(t) = \begin{cases} 1 & 0 \le t < 1/2 \\ -1 & 1/2 \le t < 1 \\ 0 & \text{otherwise} \end{cases}. \tag{5.6}$$

Thus, the approximation coefficients A_j^k in Haar wavelets are obtained via [90]:

$$A_j^k = 2^{-1/2}(A_{j-1}^{2k} + A_{j-1}^{2k+1}). \tag{5.7}$$

In Fig. 51 (a), we show the autocorrelation of processes $\{A_3\}$ and $\{D_3\}$ computed based on the I-frame sizes in single-layer *Star Wars IV* using Haar wavelets (labeled as "ACF detailed" and "ACF approx", respectively). As shown in the figure, the ACF of $\{D_3\}$, which is a typical example of detailed coefficients, is almost zero at non-zero lags, which means that it is an *i.i.d.* (uncorrelated) noise. This explains why previous literature commonly models detailed coefficients as zero-mean *i.i.d.* Gaussian variables [72]. Fig. 51 (a) also shows that the approximation coefficients have a slower decaying ACF compared to that of the detailed coefficients, which implies that they *cannot* be modeled as *i.i.d.* random variables.

Recalling that I-frame sizes $\{A_0\}$ follow a Gamma distribution [92], we next examine the relationship between $\{A_0\}$ and the approximation coefficients $\{A_j, j > 0\}$ in various sequences with the help of the following lemma. Notice that $\{A_j\}$ is a random process $A_j = (A_j^1, A_j^2, \cdots, A_j^k, \cdots)$ and A_j^k is a random variable.

Lemma 8 *Given that the I-frame sizes follow a Gamma distribution, the approximation coefficients $A_j^k, j \geq 1$ is a linear combination of several Gamma distributions.*

Proof: For brevity, we only derive the distribution of A_1^k and note that the derivations for $A_j^k, j \geq 2$ are very similar. According to (5.7), each value of A_1^k is a linear summation of the sizes of two neighboring I-frames, which we denote by X_1^k and X_2^k, respectively. Notice that X_1^k and X_2^k are two correlated Gamma distributed random variables. Then,

$$A_1^k = 2^{-1/2}(X_1^k + X_2^k), \tag{5.8}$$

where $X_i^k \sim \text{Gamma}(\alpha_i, \lambda_i), i = 1, 2$. We can rewrite X_i^k in the form of the standard Gamma distribution:

$$X_1^k = \lambda_1 Y_1, \tag{5.9}$$

$$X_2^k = \lambda_2 Y_2, \tag{5.10}$$

where $Y_i \sim \text{Gamma}(\alpha_i, 1)$ are two standard Gamma random variables.

To catch the correlation between X_1^k and X_2^k , we further decompose Y_1 and Y_2 into a sum of two *independent* standard Gamma random variables using the decomposition properties of standard Gamma distributions [31]:

$$Y_1 = Y_{11} + Y_{12}, \tag{5.11}$$

$$Y_2 = Y_{12} + Y_{22}, \tag{5.12}$$

where Y_{11}, Y_{12}, and Y_{22} are independent of each other and follow the standard Gamma distribution with parameters α_{11}, α_{12}, and α_{22}, respectively. Then the correlation between X_1^k and X_2^k becomes:

$$\mathrm{cov}(X_1^k, X_2^k) = \lambda_1\lambda_2 \mathrm{var}(Y_{12}) = \lambda_1\lambda_2\alpha_{22}. \tag{5.13}$$

Combining (5.8) and (5.13), re-write A_1^k as:

$$A_1^k = 2^{-1/2}\left(\lambda_1 Y_{11} + (\lambda_1 + \lambda_2)Y_{12} + \lambda_2 Y_{22}\right). \tag{5.14}$$

As observed from (5.14), A_1^k is a linear combination of independent standard Gamma distributions, which leads to the statement of the lemma. ■

We illustrate the distribution of the approximation coefficients $\{A_3\}$ and that of $\{A_0\}$ (original I-frame sizes) of single-layer *Star Wars IV* in Fig. 51 (b). The figure shows that the two distributions have a similar shape, but with different parameters. In the next section, we use this information to efficiently estimate the approximation coefficients.

2. Generating Synthetic I-Frame Sizes

Since the wavelet transform has a great advantage over the time-domain methods in capturing the LRD and SRD properties of video [72], [90], we model the I-frame sizes in the wavelet domain and thus need to estimate both detailed and approximation coefficients, which we already defined as $\{D_j\}$ and $\{A_j\}$, respectively.

Even though previous wavelet-based traffic modeling methods often model $\{D_j\}$ as zero-mean *i.i.d.* Gaussian variables [72], there is insufficient evidence as to the distribution of the actual $\{D_j\}$ found in GOP-based video traffic. To provide some insight into the structure of detailed coefficients, we compare the histogram of the *actual* coefficients $\{D_1\}$ in *Star Wars IV* with those generated by several alternative

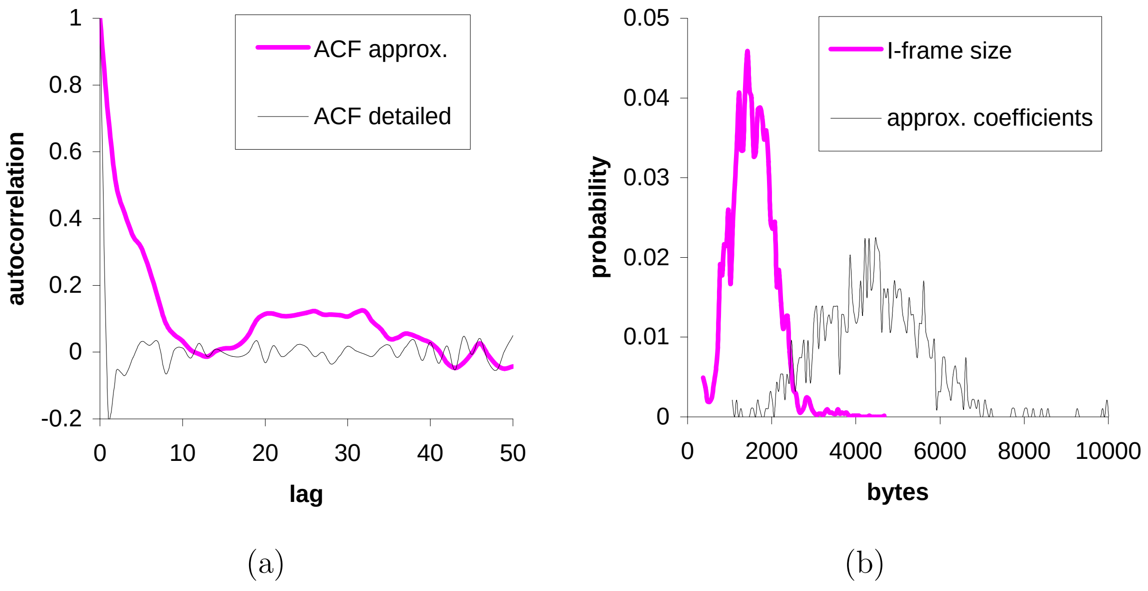


Fig. 51. (a) The ACF structure of coefficients $\{A_3\}$ and $\{D_3\}$ in single-layer *Star Wars IV*. (b) The histogram of I-frame sizes and that of approximation coefficients $\{A_3\}$.

models in Fig. 52 (note that the y-axis is scaled logarithmically). Fig. 52 (a) displays the histogram of the actual $\{D_1\}$, part (b) shows that the Gaussian fit matches neither the shape, nor the range of the actual distribution, and part (c) demonstrates that the Generalized Gaussian Distribution (GGD) produces an overly sharp peak at zero (the number of zeros in GGD is almost three times larger than that in the actual $\{D_1\}$) and also does not model the range of the real $\{D_1\}$.

Additional simulations (not shown for brevity) demonstrate that a single Laplacian distribution is not able to describe the fast decay and large data range of the actual histogram; however, a *mixture*-Laplacian distribution follows the real data very well:

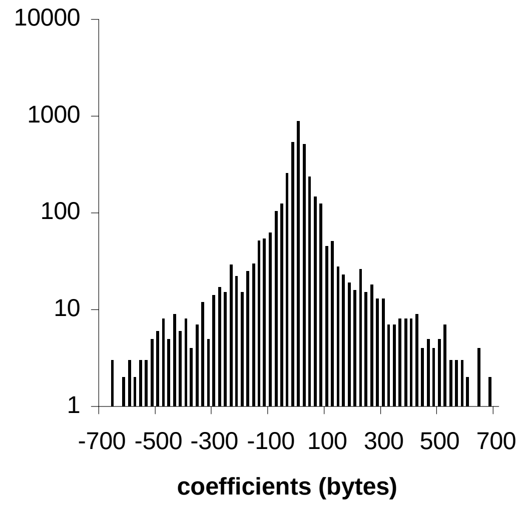
$$f(x) = p\frac{\lambda_0}{2}e^{-\lambda_0|x|} + (1-p)\frac{\lambda_1}{2}e^{-\lambda_1|x|}, \tag{5.15}$$

where $f(x)$ is the PDF of the mixture-Laplacian model, p is the probability to obtain a sample from a low-variance Laplacian component, and λ_0 and λ_1 are the shape parameters of the corresponding low- and high-variance Laplacian distributions. Fig. 52 (d) shows that the histogram of the mixture-Laplacian synthetic coefficients $\{D_1\}$ is much closer to the actual one than the other discussed distributions.

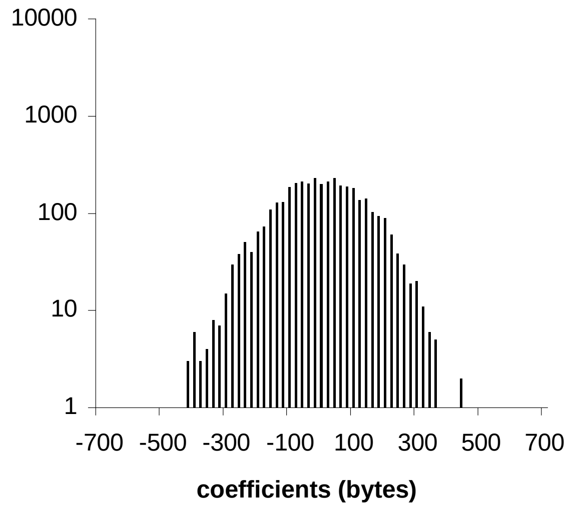
We next discuss approximation coefficients $\{A_j\}$. Recall that current methods generate the coarsest approximation coefficients (i.e., $\{A_J\}$) either as independent Gaussian [72] or Beta random variables [90]. However, as mentioned in Section 1, the approximation coefficients are non-negligibly correlated and are not *i.i.d.* To preserve the correlation of approximation coefficients and achieve the expected distribution in the synthetic coefficients, we assume that the coarsest approximation coefficients $\{A_J\}$ are *dependent* random variables with marginal Gamma distributions. We first generate N dependent Gaussian variables x_i using a $k \times k$ correlation matrix, where N is the length of $\{A_J\}$ and the correlation matrix is obtained from the actual coefficients $\{A_J\}$. The number of preserved correlation lags k is chosen to be a reasonable value (e.g., the average scene length[2]). By applying the Gaussian CDF $F_G(x)$ directly to x_i, we convert them into a uniformly distributed set of variables $F_G(x_i)$. It is well known that if F is a continuous distribution with inverse F^{-1} and u is a uniform random number, then $F^{-1}(u)$ has the distribution F. Based on this insight, we pass the result from the last step through the inverse Gamma CDF to generate (still dependent) Gamma random variables [23].

Using the estimated approximation and detailed coefficients, we perform the inverse wavelet transform to generate synthetic I-frame sizes. Fig. 53 (a) shows the ACF of the actual I-frame sizes and that of the synthetic traffic in long range. Fig. 53 (b) shows the correlation of the synthetic traffic from the GOP-GBAR model [31]

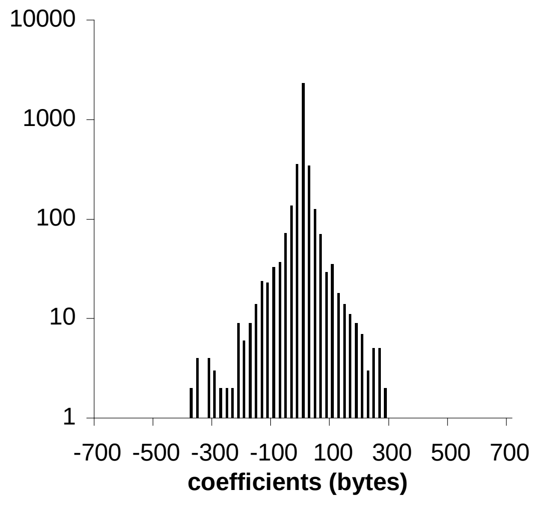
[2]This is a reasonable choice because there is much less correlation among I-frames of different scenes than among I-frames of the same scene.



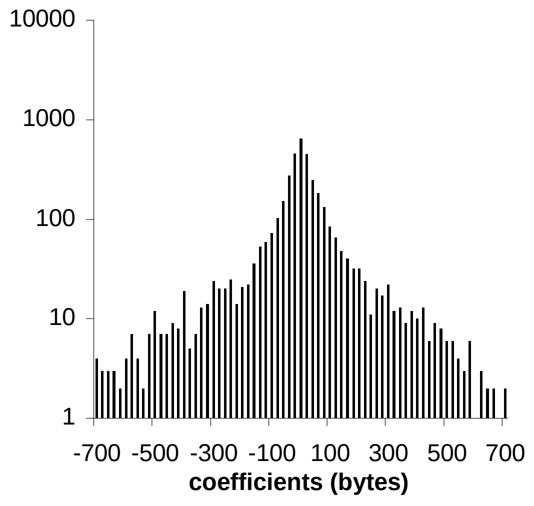
(a) actual



(b) Gaussian

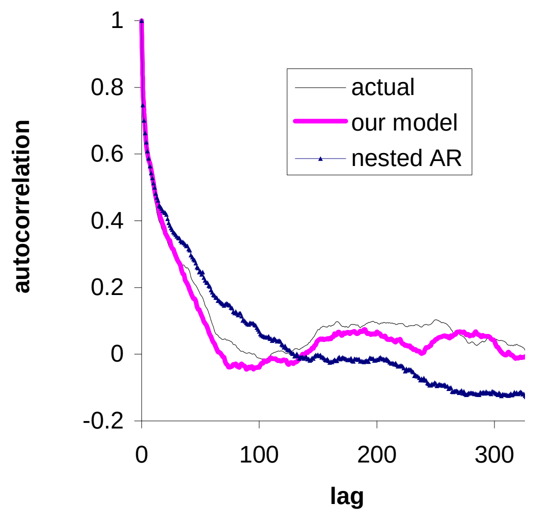


(c) GGD

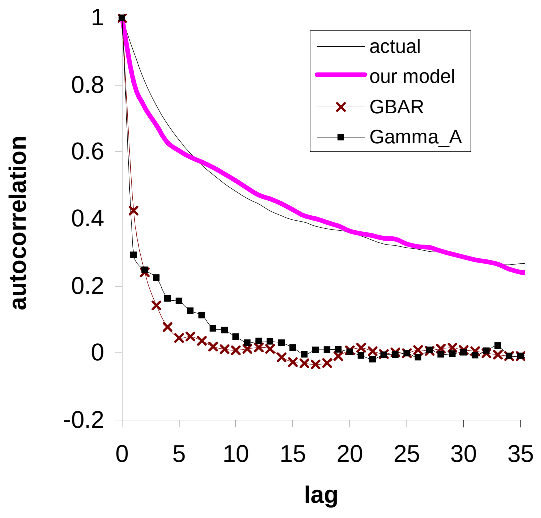


(d) mixture-Laplacian

Fig. 52. Histograms of (a) the actual detailed coefficients; (b) the Gaussian model; (c) the GGD model; and (d) the mixture-Laplacian model.



(a) LRD



(b) SRD

Fig. 53. The ACF of the actual I-frame sizes and that of the synthetic traffic in (a) long range and (b) short range.

and Gamma_A model [95] in short range. As observed in both figures, our synthetic I-frame sizes capture both the LRD and SRD properties of the original traffic better than the previous models.

C. Modeling P/B-Frame Sizes in Single-layer Traffic

We next model P-frame sizes in the time domain based on intra-GOP correlation. The framework in this section has two contributions: (1) give a detailed analysis of intra-GOP correlation for various video sequences, and (2) model intra-GOP correlation and propose a simple model that accurately generates synthetic P/B-frame sizes based on intra-GOP correlation, which is in contrast to much of the previous work that relied on *i.i.d.* random variables to model the P/B-frame sizes in each GOP [59], [44], [68], [95].

Before further discussion, we define I, P and B-*frame size sequences* as follows.

Assuming that $n \geq 1$ represents the GOP number, we define $\phi^I(n)$ to be the I-frame size of the n-th GOP, $\phi_i^P(n)$ to be the size of the i-th P-frame in GOP n, and $\phi_i^B(n)$ to be the size of the i-th B-frame in GOP n. For example, $\phi_3^P(10)$ represents the size of the third P-frame in the 10-th GOP.

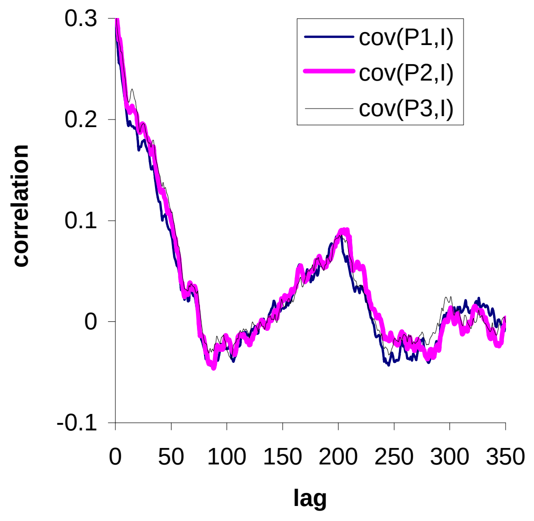
1. Intra-GOP Correlation

Lombardo *et al.* [62] noticed that there is a strong correlation[3] between the P/B-frame sizes and the I-frame size belonging to the same GOP, which is also called intra-GOP correlation. Motivated by their results, we investigate various video sequences coded at different quantization steps. Our analysis includes two parts: (a) given the same quantization step Q, the correlation between $\{\phi^I(n)\}$ and $\{\phi_i^P(n)\}$ for different i in a specific video sequence; and (b) given same i, the correlation between $\{\phi^I(n)\}$ and $\{\phi_i^P(n)\}$ or $\{\phi_i^B(n)\}$ for sequences coded at different Q.

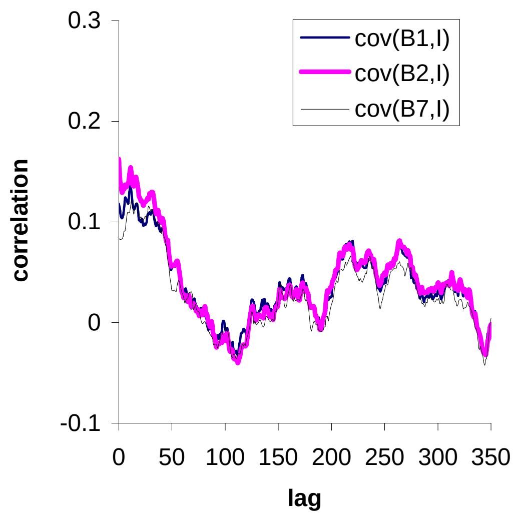
For the first part of our analysis, we display the correlation between $\{\phi^I(n)\}$ and $\{\phi_i^P(n)\}$ and that between $\{\phi^I(n)\}$ and $\{\phi_i^B(n)\}$ in single-layer *Star Wars IV* for $i = 1, 2, 3$ in Fig. 54. As shown in the figure, the correlation is almost identical for different i, which is rather convenient for our modeling purposes.

For the second part of our analysis, we examine various video sequences coded at different quantization steps to understand the relationship between intra-GOP correlation and quantization steps. We show the correlation between $\{\phi^I(n)\}$ and $\{\phi_1^P(n)\}$ and that between $\{\phi^I(n)\}$ and $\{\phi_1^B(n)\}$ in five MPEG-4 coded video sequences in Fig. 55. These five MPEG-4 sequences shown are [27]: *Star Wars IV*, *Jurassic Park I*, *The Silence of the Lambs*, *Star Trek - First Contact*, and *Starship Troopers*. All sequences are in QCIF format, coded at 25 frames/s with GOP

[3]In traffic modeling literature, the normalized auto-covariance function is often used instead of the autocorrelation function [68].



(a)



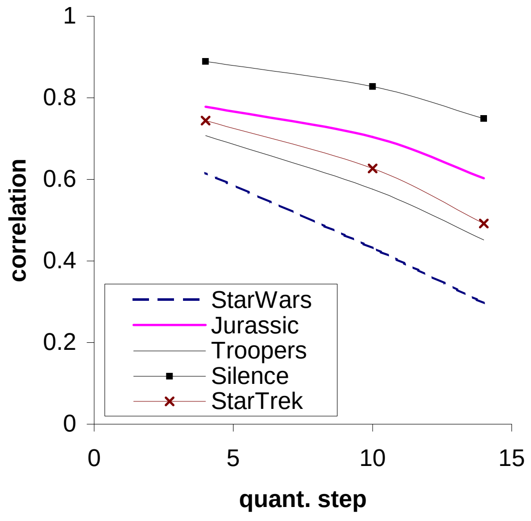
(b)

Fig. 54. (a) The correlation between $\{\phi_i^P(n)\}$ and $\{\phi^I(n)\}$ in *Star Wars IV*, for $i = 1, 2, 3$. (b) The correlation between $\{\phi_i^B(n)\}$ and $\{\phi^I(n)\}$ in *Star Wars IV*, for $i = 1, 2, 7$.

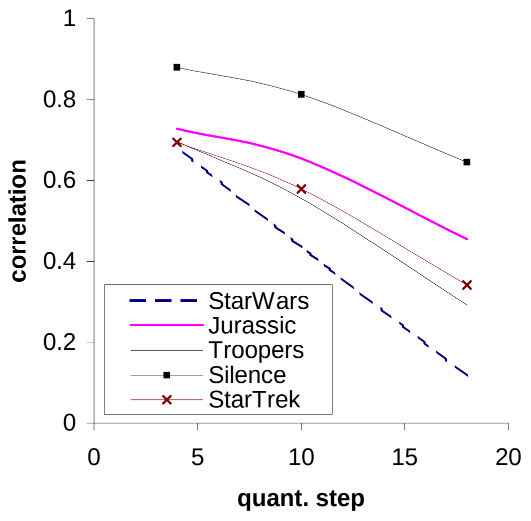
structure *IBBPBBPBBPBB*.

We also show the same correlation in H.26L coded *Starship Troopers* [87] and in the base layer of the spatially scalable *The Silence of the Lambs* in Fig. 56 (a) and (b), respectively. As observed from Fig. 55 and Fig. 56, the intra-GOP correlation decreases while the quantization step increases. This result can be very useful for modeling sequences coded from the same video but at different quantization steps Q.

To better model P and B-frame sizes, we also investigate the relationship between P/B-frame sizes and the size of I-frame belong to the same GOP. Lombardo *et al.* [62] modeled the sizes of MPEG-1 coded P/B-frames as Gamma distributed random variables, with mean and variance estimated by a linear function of $\{\phi^I(n)\}$. However, we find that this linear estimation does not hold for general video traffic. As shown in Fig. 57, the means of P and B-frames are *not* linear functions of I-frame sizes in



(a)



(b)

Fig. 55. (a) The correlation between $\{\phi^I(n)\}$ and $\{\phi_1^P(n)\}$ in MPEG-4 sequences coded at $Q = 4, 10, 14$. (b) The correlation between $\{\phi^I(n)\}$ and $\{\phi_1^B(n)\}$ in MPEG-4 sequences coded at $Q = 4, 10, 18$.

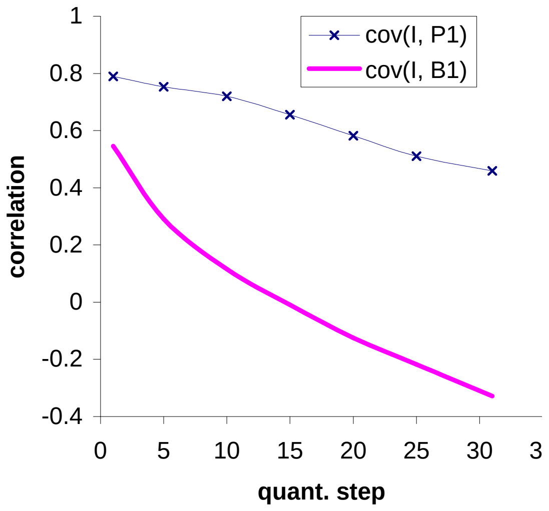
MPEG-4 coded *Star Wars IV* and *The Silence of the Lambs*. Therefore, in the next section, we propose an alternative model for generating P and B-frame sizes, which captures the intra-GOP correlation in general GOP-based VBR video.

2. Modeling P and B-Frame Sizes

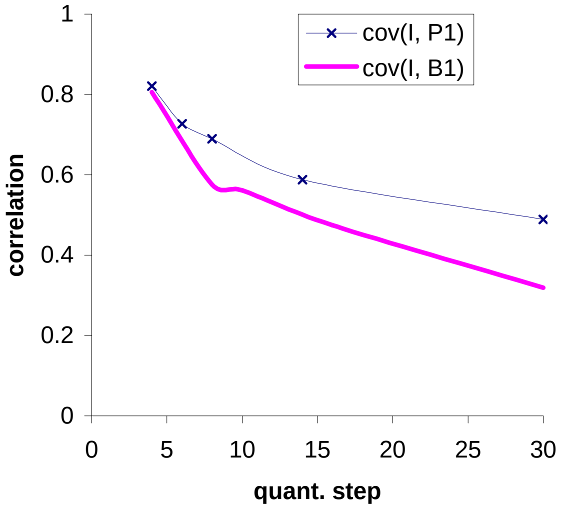
The above discussion shows that there is a similar correlation between $\{\phi_i^P(n)\}$ and $\{\phi^I(n)\}$ with respect to different i. Motivated by this observation, we propose a linear model to estimate the size of the i-th P-frame in the n-th GOP:

$$\phi_i^P(n) = a\tilde{\phi}^I(n) + \tilde{v}(n), \tag{5.16}$$

where $\tilde{\phi}^I(n) = \phi^I(n) - E[\phi^I(n)]$ and $\tilde{v}(n)$ is a synthetic process (whose properties we study below) that is independent of $\tilde{\phi}^I(n)$.



(a)



(b)

Fig. 56. The correlation between $\{\phi^I(n)\}$ and $\{\phi_1^P(n)\}$ and that between $\{\phi^I(n)\}$ and $\{\phi_1^B(n)\}$ in (a) H.26L *Starship Troopers* and (b) the base layer of the spatially scalable *The Silence of the Lambs* coded at different Q.

Lemma 9 *To capture the intra-GOP correlation, the value of coefficient a in (5.16) must be equal to:*

$$a = \frac{r(0)\sigma_P}{\sigma_I}, \tag{5.17}$$

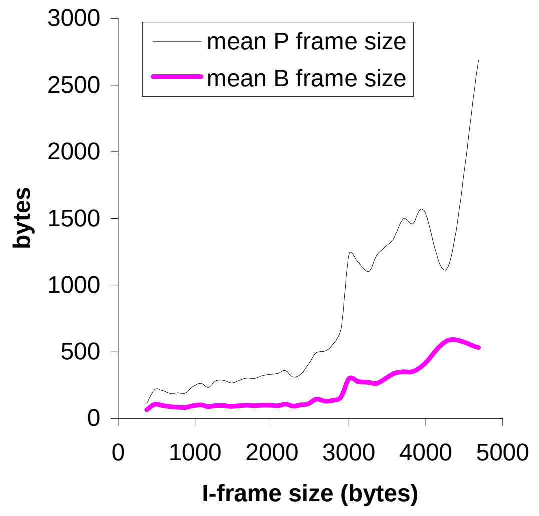
where σ_P is the standard deviation of $\{\phi_i^P(n)\}$, σ_I is the standard deviation of $\{\phi^I(n)\}$, and $r(0)$ is their normalized correlation coefficient at lag zero.

Proof: Without loss of generality, we assume that both $\tilde{\phi}^I(n)$ and $\phi_i^P(n)$ are wide-sense stationary processes. Thus, $E[\phi_i^P(n)]$ is constant and:

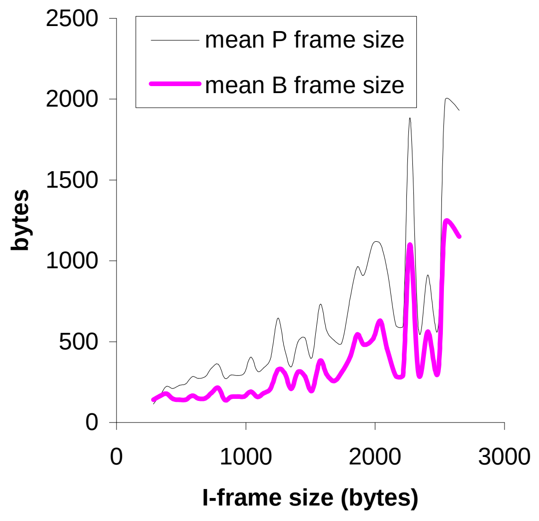
$$E[\tilde{\phi}^I(n-k)] = E[\tilde{\phi}^I(n)] = 0. \tag{5.18}$$

Denote by $C(k)$ the covariance between $\phi_i^P(n)$ and $\tilde{\phi}^I(n)$ at lag k:

$$C(k) = E[(\phi_i^P(n) - E[\phi_i^P])(\tilde{\phi}^I(n-k) - E[\tilde{\phi}^I])]. \tag{5.19}$$



(a) Star Wars IV



(b) The Silence of the Lambs

Fig. 57. The mean sizes of P and B-frames of each GOP given the size of the corresponding I-frame in (a) the single-layer *Star Wars IV* and (b) the base layer of the spatially scalable *The Silence of the Lambs*.

Recall that $v(n)$ and $\tilde{\phi}^I(n)$ are independent of each other and thus $E[v(n)\cdot\tilde{\phi}^I(n)] = E[v(n)]\cdot E[\tilde{\phi}^I(n)] = 0$. Then $C(k)$ becomes:

$$\begin{aligned} C(k) &= E[(a\tilde{\phi}^I(n) + v(n) - E[\phi_i^P])\tilde{\phi}^I(n-k)] \\ &= aE[\tilde{\phi}^I(n)\tilde{\phi}^I(n-k)] \end{aligned} \tag{5.20}$$

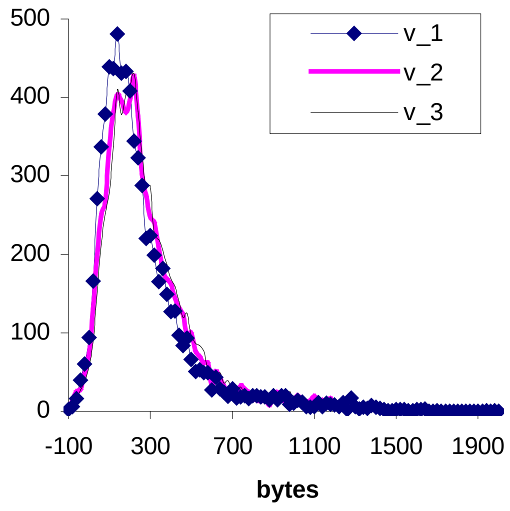
Next, observe that the normalized correlation coefficient r at lag zero is:

$$r(0) = \frac{C(0)}{\sigma_P\sigma_{\tilde{I}}} = \frac{aE[\tilde{\phi}^I(n)^2]}{\sigma_P\sigma_{\tilde{I}}}, \tag{5.21}$$

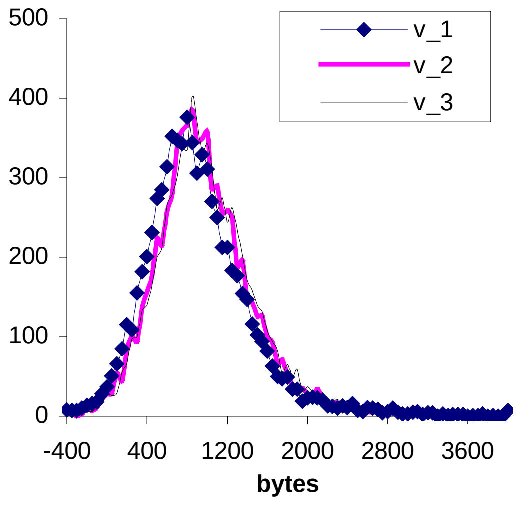
where $\sigma_{\tilde{I}}$ is the standard deviation of $\tilde{\phi}^I(n)$. Recalling that $E[\tilde{\phi}^I(n)] = 0$, we have $E[\tilde{\phi}^I(n)^2] = \sigma_{\tilde{I}}^2 = \sigma_I^2$ and:

$$\frac{a\cdot\sigma_I}{\sigma_P} = r(0), \tag{5.22}$$

which leads to (5.17). ■



(a) Star Wars IV



(b) Jurassic Park I

Fig. 58. Histograms of $\{v(n)\}$ for $\{\phi_i^P(n)\}$ with $i = 1, 2, 3$ in (a) *Star Wars IV* and (b) *Jurassic Park I*. Both sequences are coded at $Q = 14$.

To understand how to generate $\{\tilde{v}(n)\}$, we next examine the *actual* residual process $v(n) = \phi_i^P(n) - a\tilde{\phi}^I(n)$ for each i. We show the histograms of $\{v(n)\}$ for P-frame sequences $i = 1, 2, 3$ in the single-layer *Star Wars IV* and *Jurassic Park I* in Fig. 58. The figures shows that the residual process $\{v(n)\}$ does not change much as a function of i.

In Fig. 59 (a), we show the histograms of $\{v(n)\}$ for sequences coded at different Q. The figure shows that the histogram becomes more Gaussian-like when Q increases. Due to the diversity of the histogram of $\{v(n)\}$, we use a generalized Gamma distribution $Gamma(\gamma, \alpha, \beta)$ to estimate $\{v(n)\}$. Fig. 59 (b) shows that the smaller the quantization step Q, the larger the value of parameter a in (5.17), which is helpful for further modeling sequences coded from the same video content but at different quantization steps.

From Fig. 55 (b), we observe that the correlation between $\{\phi_i^B(n)\}$ and $\{\phi^I(n)\}$ could be as small as 0.1 (e.g., in *Star Wars IV* coded at $Q = 18$) or as large as 0.9 (e.g., in *The Silence of the Lambs* coded at $Q = 4$). Thus, we can generate the synthetic B-frame traffic simply by an *i.i.d.* lognormal random number generator when the correlation between $\{\phi_i^B(n)\}$ and $\{\phi^I(n)\}$ is small, or by a linear model similar to (5.16) when the correlation is large. The linear model has the following form:

$$\phi_i^B(n) = a\tilde{\phi}^I(n) + \tilde{v}_B(n), \tag{5.23}$$

where $a = r(0)\sigma_B/\sigma_I$, $r(0)$ is the lag-0 correlation between $\{\phi^I(n)\}$ and $\{\phi_i^B(n)\}$, σ_B and σ_I are the standard deviation of $\{\phi_i^B(n)\}$ and $\{\phi^I(n)\}$, respectively. Process $\tilde{v}_B(n)$ is independent of $\tilde{\phi}^I(n)$.

We illustrate the difference between our model and a typical *i.i.d.* method of prior work (e.g., [68], [95]) in Fig. 60. The figure shows that our model indeed preserves the intra-GOP correlation of the original traffic, while the previous methods produce white (uncorrelated) noise. Statistical parameters $(r(0), \sigma_P, \sigma_I, \gamma, \alpha, \beta)$ needed for this model are easily estimated from the original sequences.

D. Modeling the Enhancement Layer

In this section, we provide brief background knowledge of multi-layer video, investigate methods to capture cross-layer dependency, and model the enhancement-layer traffic.

Due to its flexibility and high bandwidth utilization, layered video coding is common in video applications. Layered coding is often referred to as “scalable coding,” which can be further classified as coarse-granular (e.g., spatial scalability) or fine-granular (e.g., fine granular scalability (FGS)) [107]. The major difference between

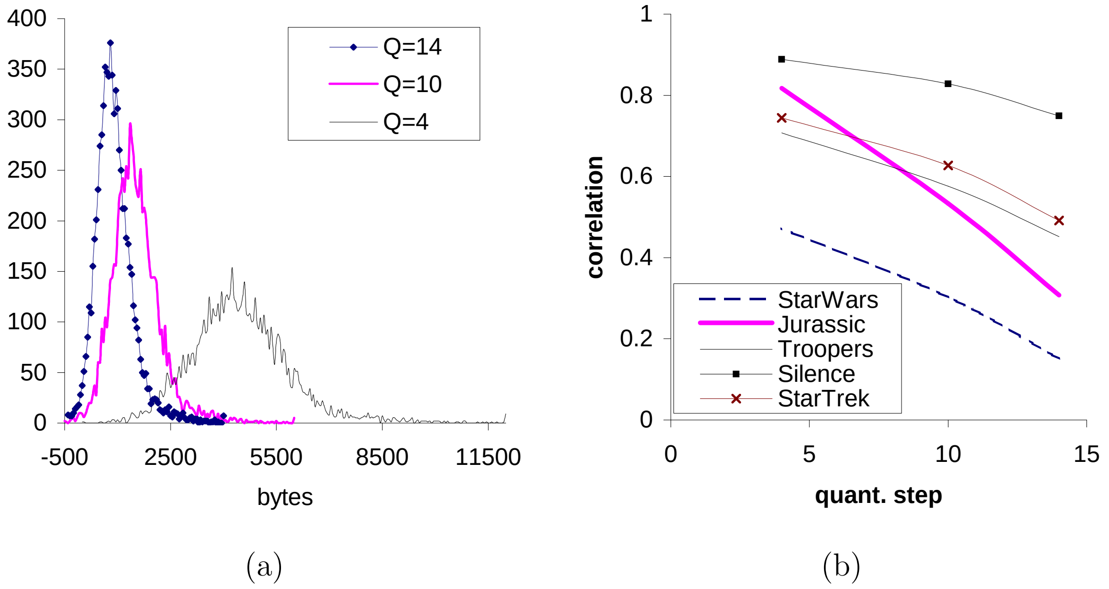
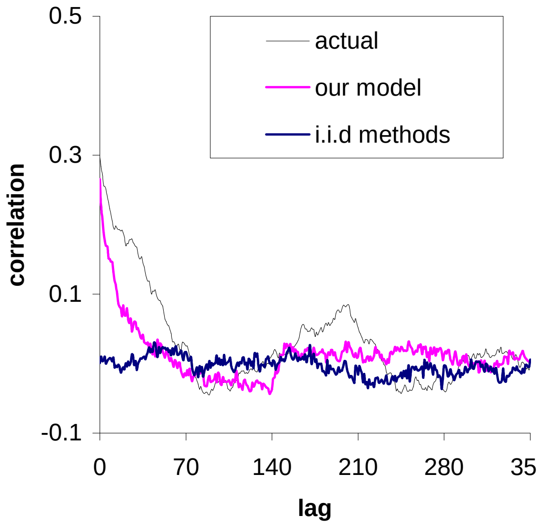


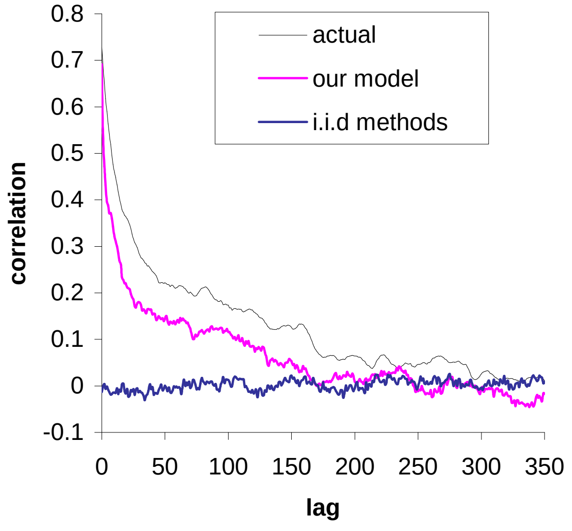
Fig. 59. (a) Histograms of $\{v(n)\}$ for $\{\phi_1^P(n)\}$ in *Jurassic Park I* coded at $Q = 4, 10, 14$. (b) Linear parameter a for modeling $\{\phi_i^P(n)\}$ in various sequences coded at different Q.

coarse granularity and fine granularity is that the former provides quality improvements only when a *complete* enhancement layer has been received, while the latter continuously improves video quality with every additionally received codeword of the enhancement layer bitstream.

In both coarse granular and fine granular coding methods, an enhancement layer is coded with the residual between the original image and the reconstructed image from the base layer. Therefore, the enhancement layer has a strong dependency on the base layer. Zhao *et al.* [115] also indicate that there exists a cross-correlation between the base layer and the enhancement layer; however, this correlation has not been fully addressed in previous studies. In the next subsection, we investigate the cross-correlation between the enhancement layer and the base layer using spatially scalable *The Silence of the Lambs* sequence and an FGS-coded *Star Wars IV* sequence as



(a)



(b)

Fig. 60. (a) The correlation between $\{\phi_1^P(n)\}$ and $\{\phi^I(n)\}$ in *Star Wars IV*. (b) The correlation between $\{\phi_1^B(n)\}$ and $\{\phi^I(n)\}$ in *Jurassic Park I*.

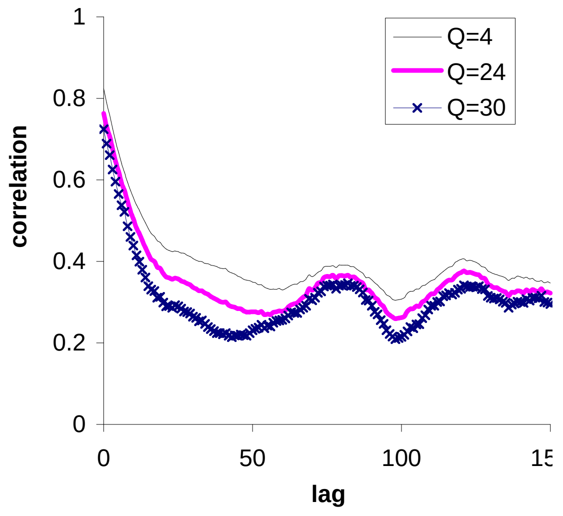
examples. We only show the analysis of two-layer sequences for brevity and similar results hold for video streams with more than two layers.

1. Analysis of the Enhancement Layer

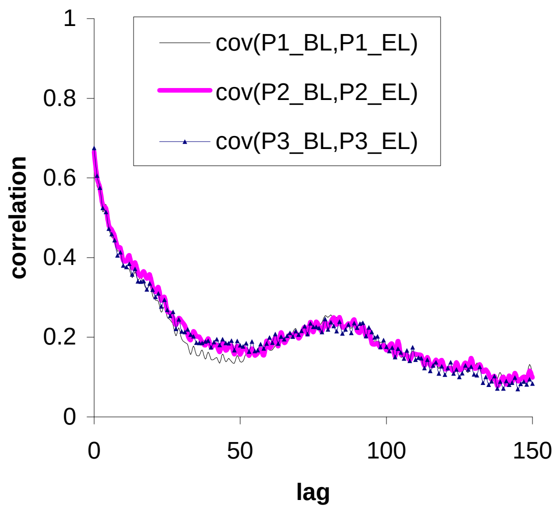
Notice that We do not consider temporal scalable coded sequences, in which the base layer and the enhancement layer are approximately equivalent to extracting I/P-frames and B-frames out of a single-layer sequence, respectively [87].

For discussion convenience, we define the enhancement layer frame sizes as follows. Similar to the definition in the base layer, we define $\varepsilon^I(n)$ to be the I-frame size of the n-th GOP, $\varepsilon_i^P(n)$ to be the size of the i-th P-frame in GOP n, and $\varepsilon_i^B(n)$ to be the size of the i-th B-frame in GOP n.

Since each frame in the enhancement layer is predicted from the corresponding



(a)

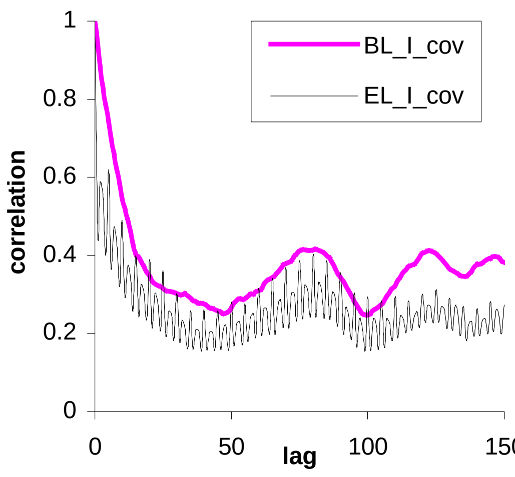


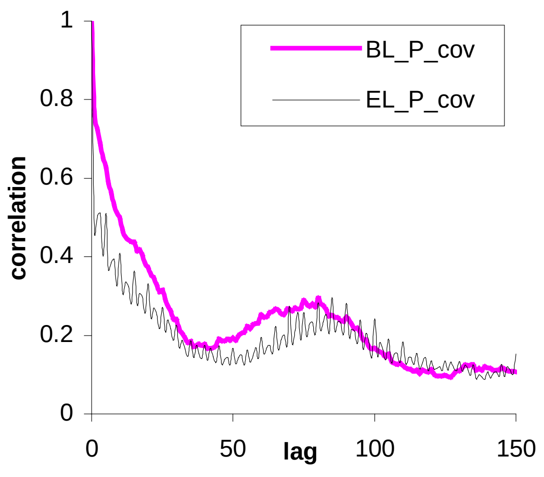
(b)

Fig. 61. (a) The correlation between $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ in *The Silence of the Lambs* coded at $Q = 4, 24, 30$. (b) The correlation between $\{\varepsilon_i^P(n)\}$ and $\{\phi_i^P(n)\}$ in *The Silence of the Lambs* coded at $Q = 30$, for $i = 1, 2, 3$.

frame in the base layer, we examine the cross-correlation between the enhancement layer frame sizes and the corresponding base layer frame sizes in various sequences. In Fig. 61 (a), we display the correlation between $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ in *The Silence of the Lambs* coded at different Q. As observed from the figure, the correlation between $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ is stronger when the quantization step Q is smaller. However, the difference among these cross-correlation curves is not as obvious as that in intra-GOP correlation. We also observe that the cross-correlation is still strong even at large lags, which indicates that $\{\varepsilon^I(n)\}$ exhibits LRD properties and we should preserve these properties in the synthetic enhancement layer I-frame sizes.

In Fig. 61 (b), we show the cross-correlation between processes $\{\varepsilon_i^P(n)\}$ and $\{\phi_i^P(n)\}$ for $i = 1, 2, 3$. The figure demonstrates that the correlation between the enhancement layer and the base layer is quite strong, and the correlation structures



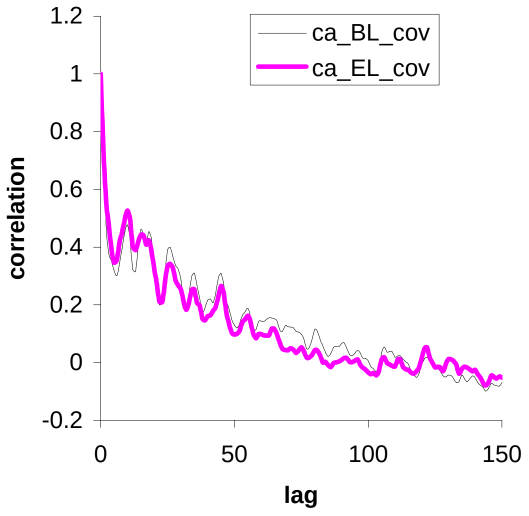


(a) (b)

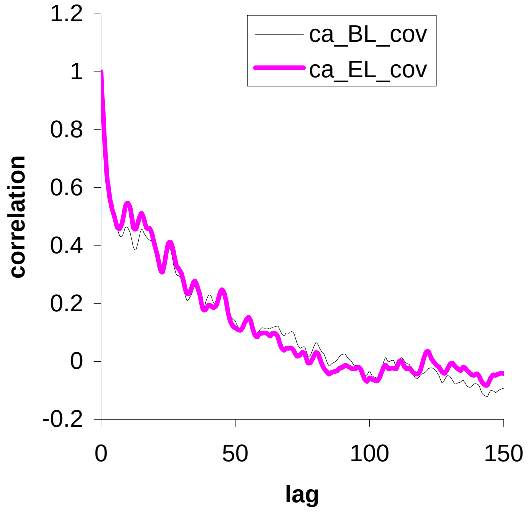
Fig. 62. (a) The ACF of $\{\varepsilon^I(n)\}$ and that of $\{\phi^I(n)\}$ in *Star Wars IV*. (b) The ACF of $\{\varepsilon^P_1(n)\}$ and that of $\{\phi^P_1(n)\}$ in *The Silence of the Lambs*.

between each $\{\varepsilon^P_i(n)\}$ and $\{\phi^P_i(n)\}$ are very similar to each other. To avoid repetitive description, we do not show the correlation between $\{\varepsilon^B_i(n)\}$ and $\{\phi^B_i(n)\}$, which is similar to that between $\{\varepsilon^P_i(n)\}$ and $\{\phi^P_i(n)\}$.

Aside from cross-correlation, we also examine the autocorrelation of each frame sequence in the enhancement layer and that of the corresponding sequence in the base layer. We show the ACF of $\{\varepsilon^I(n)\}$ and that of $\{\phi^I(n)\}$ (labeled as "EL_I_cov" and "BL_I_cov", respectively) in Fig. 62 (a); and display the ACF of $\{\varepsilon^P_1(n)\}$ and that of $\{\phi^P_1(n)\}$ in Fig. 62 (b). The figure shows that although the ACF structure of $\{\varepsilon^I(n)\}$ has some oscillation, its trend closely follows that of $\{\phi^I(n)\}$. One also observes from the figures that the ACF structures of processes $\{\varepsilon^P_i(n)\}$ and $\{\phi^P_i(n)\}$ are similar to each other.



(a) $Q = 30$



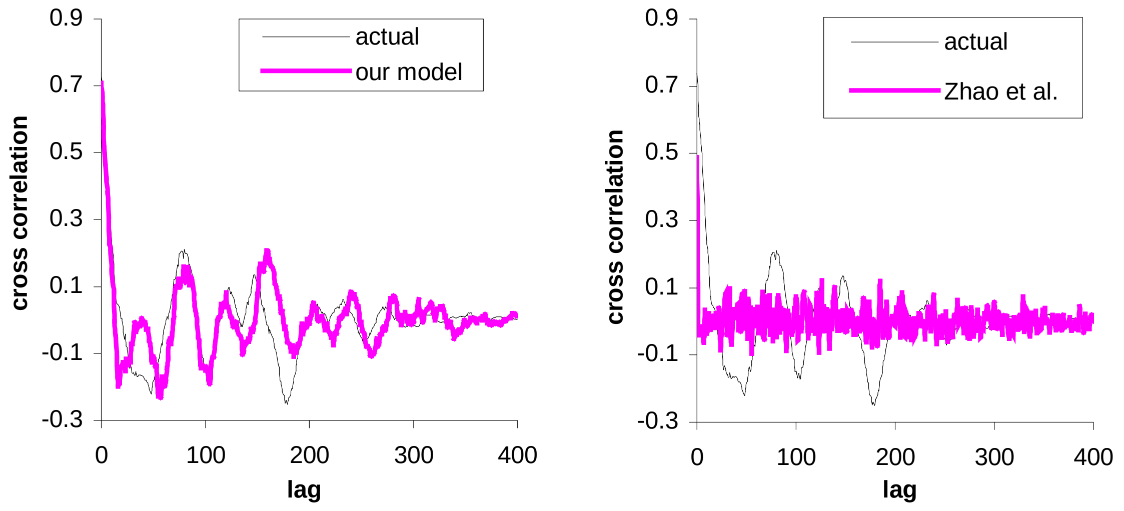
(b) $Q = 4$

Fig. 63. The ACF of $\{A_3(\varepsilon)\}$ and $\{A_3(\phi)\}$ in *The Silence of the Lambs* coded at (a) $Q = 30$ and (b) $Q = 4$.

2. Modeling I-Frame Sizes

Although cross-layer correlation is obvious in multi-layer traffic, previous work neither considered it during modeling [9], nor explicitly addressed the issue of its modeling [115]. In this section, we first describe how we model the enhancement layer I-frame sizes and then evaluate the performance of our model in capturing the cross-layer correlation.

Recalling that $\{\varepsilon^I(n)\}$ also possesses both SRD and LRD properties, we model it in the wavelet domain as we modeled $\{\phi^I(n)\}$. We define $\{A_j(\varepsilon)\}$ and $\{A_j(\phi)\}$ to be the approximation coefficients of $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ at the wavelet decomposition level j, respectively. To better understand the relationship between $\{A_j(\varepsilon)\}$ and $\{A_j(\phi)\}$, we show the ACF of $\{A_3(\varepsilon)\}$ and $\{A_3(\phi)\}$ using Haar wavelets (labeled as "ca_EL_cov" and "ca_BL_cov", respectively) in Fig. 63.



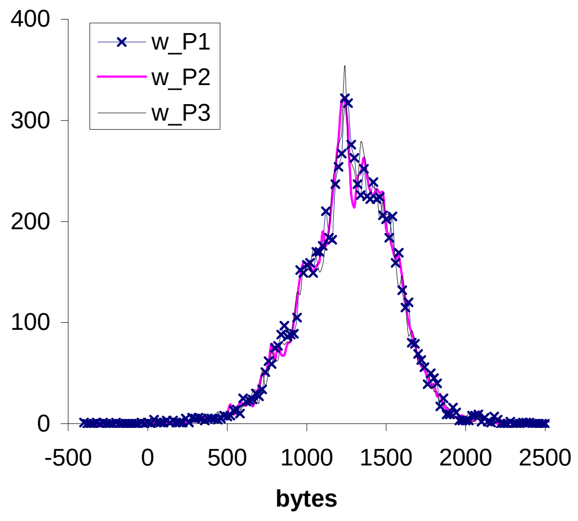
(a) our model (b) model [115]

Fig. 64. The cross-correlation between $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ in *The Silence of the Lambs* and that in the synthetic traffic generated from (a) our model and (b) model [115].

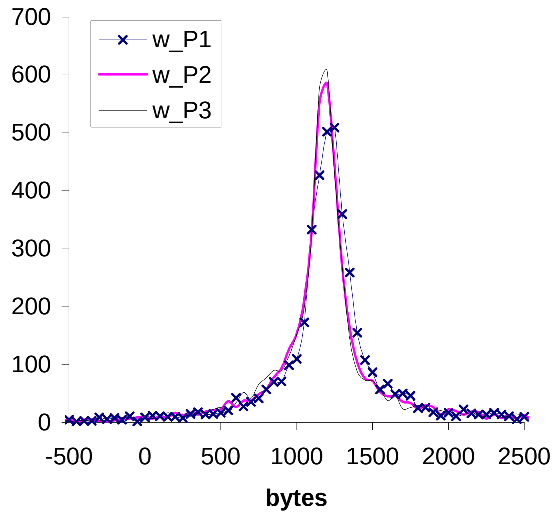
As shown in Fig. 63, $\{A_j(\varepsilon)\}$ and $\{A_j(\phi)\}$ exhibit similar ACF structure. Thus, we generate $\{A_J(\varepsilon)\}$ by borrowing the ACF structure of $\{A_J(\phi)\}$, which is known from our base-layer model. Using the ACF of $\{A_J(\phi)\}$ in modeling $\{\varepsilon^I(n)\}$ not only saves computational cost, but also preserves the cross-layer correlation. In Fig. 64, we compare the actual cross-correlation between $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ to that between the synthetic $\{\varepsilon^I(n)\}$ and $\{\phi^I(n)\}$ generated from our model and Zhao's model [115]. The figure shows that our model significantly outperforms Zhao's model in preserving the cross-layer correlation.

3. Modeling P and B-Frame Sizes

Recall that the cross-correlation between $\{\varepsilon_i^P(n)\}$ and $\{\phi_i^P(n)\}$ and that between $\{\varepsilon_i^B(n)\}$ and $\{\phi_i^B(n)\}$ are also strong, as shown in Fig. 61. We use the linear model



(a) Star Wars IV



(b) The Silence of the Lambs

Fig. 65. Histograms of $\{w_1(n)\}$ in (a) *Star Wars IV* and (b) *The Silence of the Lambs* ($Q = 24$), with $i = 1, 2, 3$.

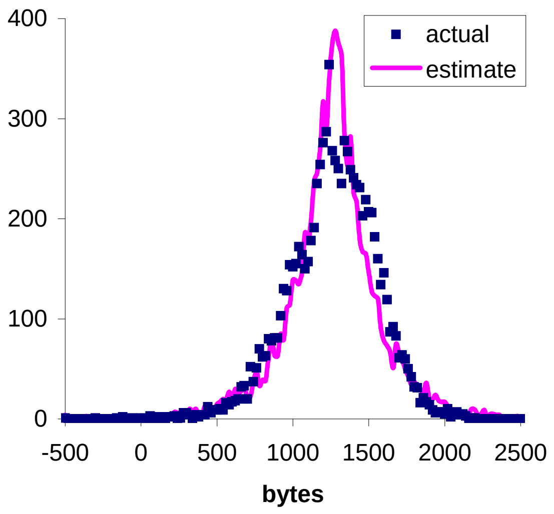
from Section 2 to estimate the sizes of the i-th P and B-frames in the n-th GOP:

$$\varepsilon_i^P(n) = a\phi_i^P(n) + \tilde{w}_1(n), \tag{5.24}$$

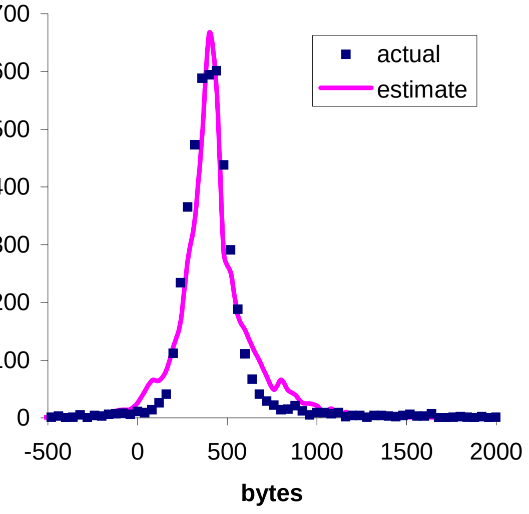
$$\varepsilon_i^B(n) = a\phi_i^B(n) + \tilde{w}_2(n), \tag{5.25}$$

where $a = r(0)\sigma_\varepsilon/\sigma_\phi$, $r(0)$ is the lag-0 cross-correlation coefficient, σ_ε is the standard deviation of the enhancement-layer sequence, and σ_ϕ is the standard deviation of the corresponding base-layer sequence. Processes $\{\tilde{w}_1(n)\}, \{\tilde{w}_2(n)\}$ are independent of $\{\phi_i^P(n)\}$ and $\{\phi_i^B(n)\}$. We examine $\{w_1(n)\}$ and $\{w_2(n)\}$ and find they exhibit similar properties. We show two examples of $\{w_1(n)\}$ in Fig. 65.

As observed from Fig. 65, the histogram of $\{w_1(n)\}$ is asymmetric and decays fast on both sides. Therefore, we use two exponential distributions to estimate its PDF. We first left-shift $\{w_1(n)\}$ by an offset δ to make the mode (i.e., the peak) appear at zero. We then model the right side using one exponential distribution $exp(\lambda_1)$



(a) Star Wars IV



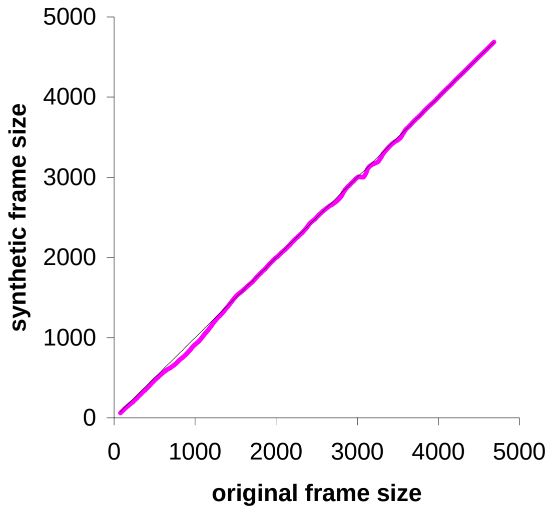
(b) The Silence of the Lambs

Fig. 66. Histograms of $\{w_1(n)\}$ and $\{\tilde{w}_1(n)\}$ for $\{\varepsilon_1^P(n)\}$ in (a) *Star Wars IV* and (b) *The Silence of the Lambs* ($Q = 30$).

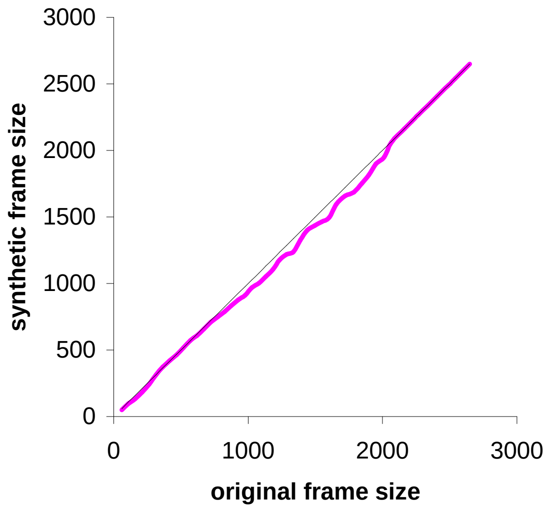
and the absolute value of the left side using another exponential distribution $exp(\lambda_2)$. Afterwards, we generate synthetic data $\{\tilde{w}_1(n)\}$ based on these two exponential distributions and right-shift the result by δ. As shown in Fig. 66, the histograms of $\{\tilde{w}_1(n)\}$ are close to those of the actual data in both *Star Wars IV* and *The Silence of the Lambs*. We generate $\{\tilde{w}_2(n)\}$ in the same way and find its histogram is also close to that of $\{w_2(n)\}$.

E. Model Accuracy Evaluation

As we stated earlier, a good traffic model should capture the statistical properties of the original traffic and be able to accurately predict network performance. There are three popular studies to verify the accuracy of a video traffic model [95]: quantile-quantile (QQ) plots, the variance of traffic during various time intervals, and buffer



(a) Star Wars IV



(b) The Silence of the Lambs

Fig. 67. QQ plots for the synthetic (a) single-layer *Star Wars IV* traffic and (b) *The Silence of the Lambs* base-layer traffic.

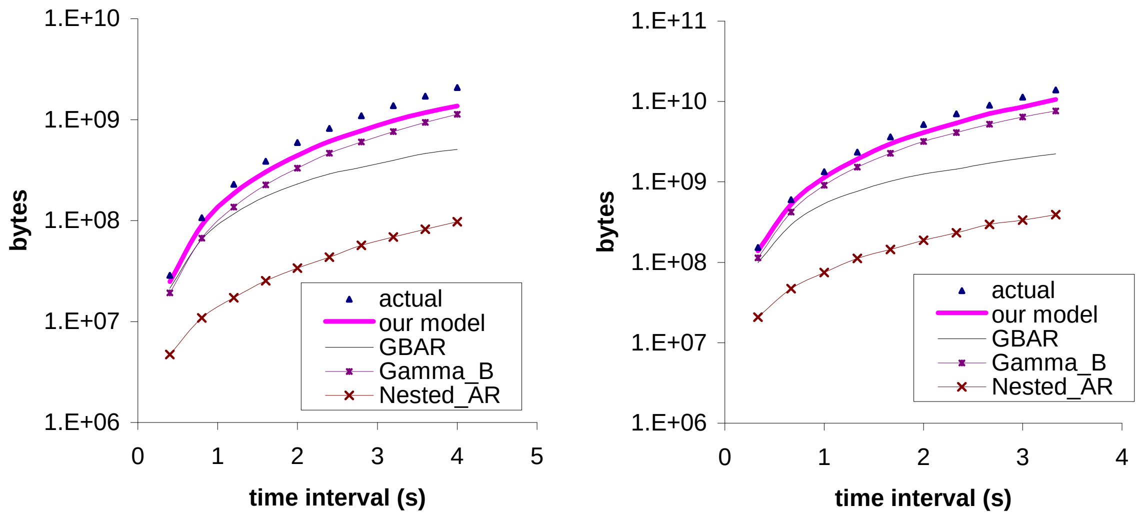
overflow loss evaluation. While the first two measures visually evaluate how well the distribution of the synthetic traffic and that of the original one matches, the overflow loss simulation examines the effectiveness of a traffic model to capture the temporal burstiness of original traffic.

The QQ plot is a graphical technique to verify the distribution similarity between two test data sets. If the two data sets have the same distribution, the points should fall along the 45 degree reference line. The greater the departure from this reference line, the greater the difference between the two test data sets.

Different from QQ plot, the variance of traffic during various time intervals shows whether the second-order moment of the synthetic traffic fits that of the original one. This second-order descriptor is used to capture burstiness properties of arrival processes [9]. This measure operates as follows. Assume that the length of a video sequence is l and there are m frames at a given time interval. We segment the one-dimensional data into a $m \times n$ matrix, where $n = l/m$. After summarizing all the data in each column, we obtain a sequence of length n and then calculate its variance. Thus, we can obtain a set of variances given a set of time intervals.

Besides the distribution, we also examine how well our approach preserves the temporal information of the original traffic. A common test for this is to pass the synthetic traffic through a generic router buffer with capacity c and drain rate d [95]. The drain rate is the number of bytes drained per second and is simulated as different multiples of the average traffic rate $\bar{r}$.

In the following two sections, we evaluate the accuracy of our model in both single-layer and multi-layer traffic using the above three measures. We should note that simulations with additional video sequences have demonstrated results similar to those shown throughout this section.



(a) Star Wars IV (b) The Silence of the Lambs

Fig. 68. Comparison of variance between synthetic and original traffic in (a) single-layer *Star Wars IV* and (b) *The Silence of the Lambs* base layer.

1. Single-layer and the Base Layer Traffic

We first show QQ plots of the synthetic single-layer *Star Wars IV* and the synthetic base layer of *The Silence of the Lambs* that are generated by our model in Fig. 67 (a) and (b), respectively. As shown in the figure, the generated frame sizes and the original traffic are almost identical.

In Fig. 68, we give a comparison between variance of the original traffic and that of the synthetic traffic generated from differen models at various time intervals. The figure shows that the second-order moment of our synthetic traffic is in a good agreement with that of the original one.

We also compare the accuracy of several models using a leaky-bucket simulation. To understand the performance differences between various models, we define the relative error e as the difference between the *actual* packet loss p observed in the buffer fed with the original traffic and that observed using the synthetic traffic generated by each of the models:

$$e = \frac{|p - p_{model}|}{p}. \tag{5.26}$$

In Table V, we illustrate the values of e for various buffer capacities and drain rates d. As shown in the table, the synthetic traffic generated by our model provides a very accurate estimate of the actual data loss probability p and significantly outperforms the other methods. In addition, our synthetic traffic is approximately 30% more accurate than the *i.i.d.* models of prior work in estimating the loss ratio of P-frames.

In Fig. 69, we show the relative error e of synthetic traffic generated from different models in H.26L *Starship Troopers* coded at $Q = 1, 31$, given $d = \bar{r}$. Since GOP-GBAR model [31] is specifically developed for MPEG traffic, we do not apply it to H.26L sequences. The figure shows that our model outperforms the other three models

Table V. Relative Data Loss Error e in *Star Wars IV*.

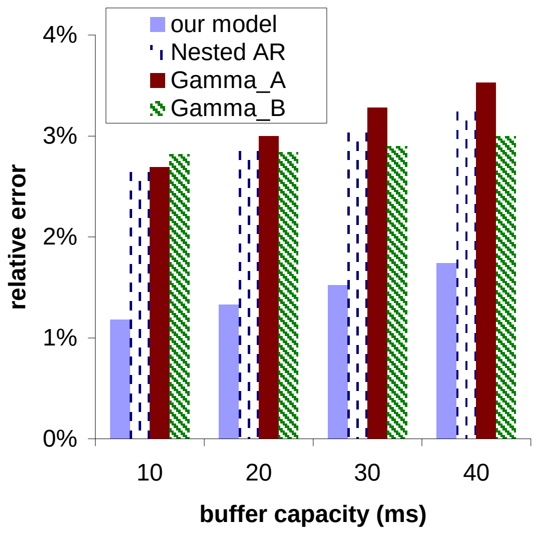
Buffer capacity	Traffic type	Drain rate		
		$2\bar{r}$	$4\bar{r}$	$5\bar{r}$
10ms	Our Model	1.80%	0.93%	0.50%
	GOP-GBAR [31]	2.44%	2.51%	4.01%
	Nested AR [68]	4.02%	2.05%	5.63%
	Gamma_A [95]	5.54%	1.04%	0.99%
	Gamma_B [95]	5.76%	1.81%	1.15%
20ms	Our Model	0.93%	0.61%	1.13%
	GOP-GBAR [31]	3.84%	2.16%	3.77%
	Nested AR [68]	5.81%	2.77%	8.46%
	Gamma_A [95]	5.20%	0.61%	2.57%
	Gamma_B [95]	4.89%	1.93%	2.05%
30ms	Our Model	0.25%	0.33%	0.95%
	GOP-GBAR [31]	4.94%	3.33%	5.68%
	Nested AR [68]	6.94%	4.14%	9.92%
	Gamma_A [95]	4.88%	1.10%	4.48%
	Gamma_B [95]	4.67%	2.17%	4.03%

in *Starship Troopers* coded at small Q and performs as good as model Gamma_A [95] in the large Q case (the relative error e of both models is less than 1% in Fig. 69 (b)).

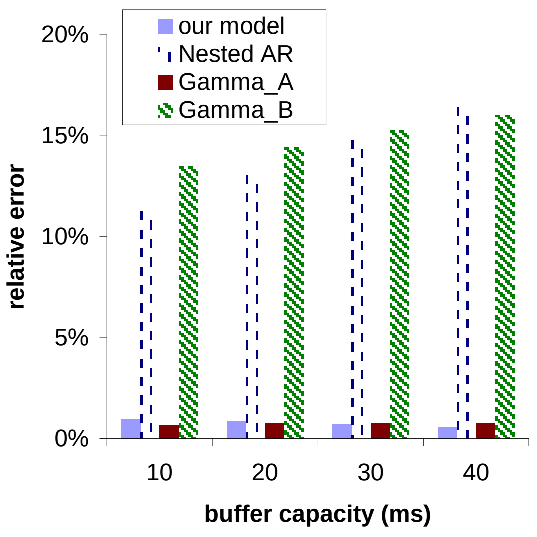
2. The Enhancement Layer Traffic

We evaluate the accuracy of the synthetic enhancement layer by using QQ plots and show two examples in Fig. 70, which displays two QQ plots for the synthetic *The Silence of the Lambs* and *Star Wars IV* enhancement-layer traffic. The figure shows that the synthetic frame sizes in both sequences have the same distribution as those in the original traffic.

We also compare the variance of the original traffic and that of the synthetic traffic in Fig. 71. Due to the computational complexity of model [115] in calculating long sequences, we only take the first 5000 frames of *Star Wars IV* and *The Silence of the Lambs.* As observed from the figure, our model well preserves the second-order



(a) $Q = 1$

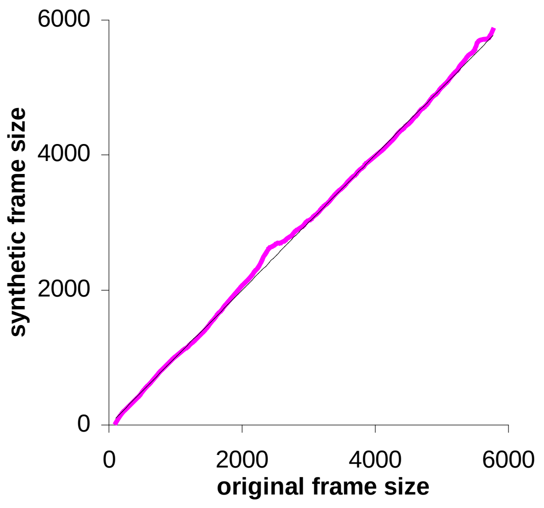


(b) $Q = 31$

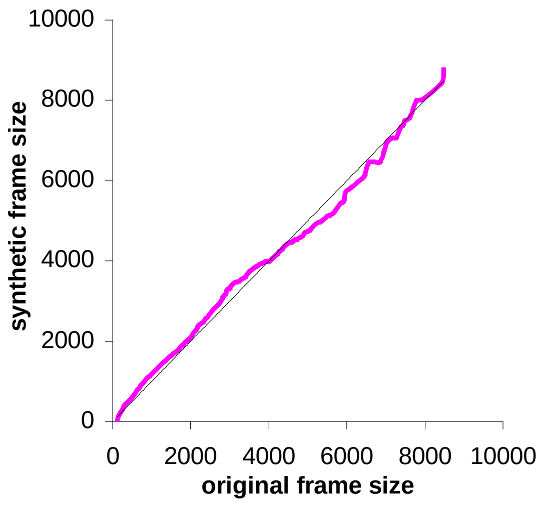
Fig. 69. Given $d = \bar{r}$, the error e of various synthetic traffic in H.26L *Starship Troopers* coded at (a) $Q = 1$ and (b) $Q = 31$.

moment of the original traffic.

We next examine the data loss ratio predicted by our synthetic traffic passed through a generic buffer as shown in the previous section. Recall that the model in [9] is only applicable to sequences with a CBR base layer and the one in [115] is suitable only for short sequences. Therefore, we are not able to show results using leaky-bucket simulations for these multi-layer models given the nature of our sample sequences. In Fig. 72 and Fig. 73, we show the overflow data loss ratio of the enhancement layers in both *The Silence of the Lambs* (54,000 frames) and *Star Wars IV* (108,000 frames) with different drain rates d for buffer capacity $c = 10$ ms and $c = 30$ ms, respectively. The x-axis in the figure represents the ratio of the drain rates to the average traffic rate $\bar{r}$. The figure shows that the synthetic enhancement layer preserves the temporal information of the original traffic very well.

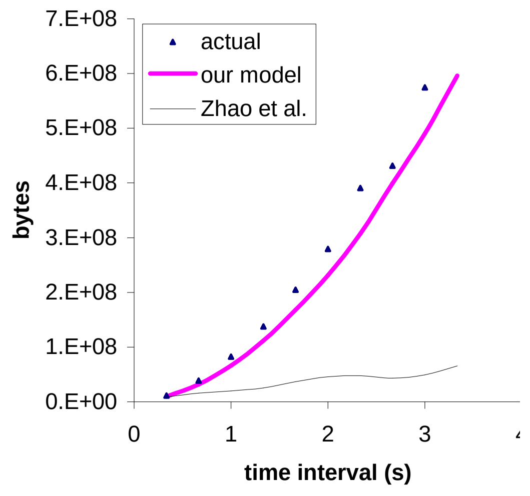


(a) Star Wars IV

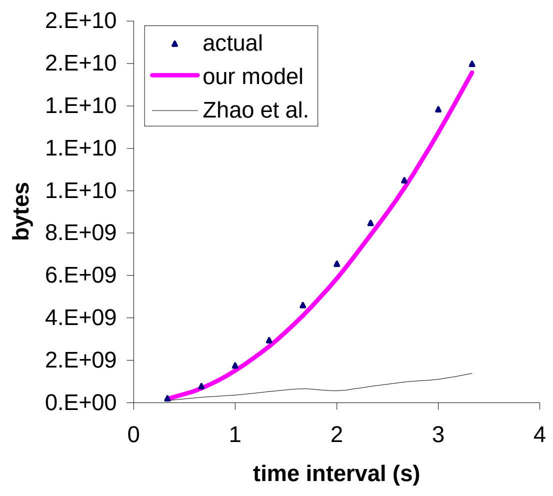


(b) The Silence of the Lambs

Fig. 70. QQ plots for the synthetic enhancement-layer traffic: (a) *Star Wars IV* and (b) *The Silence of the Lambs*.

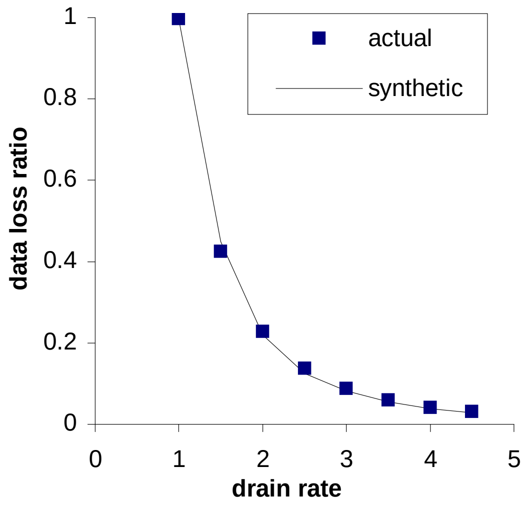


(a) Star Wars IV

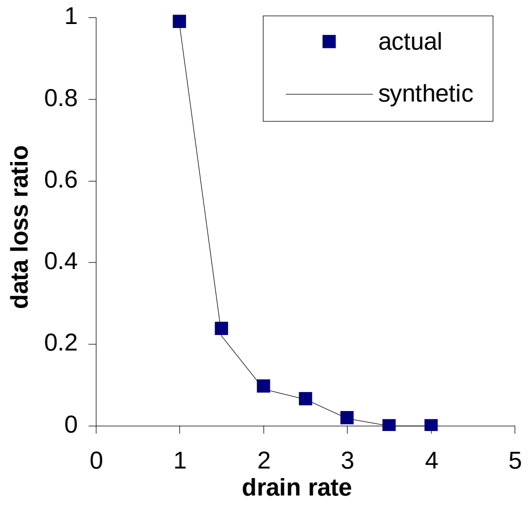


(b) The Silence of the Lambs

Fig. 71. Comparison of variance between the synthetic and original enhancement layer traffic in (a) *Star Wars IV* and (b) *The Silence of the Lambs*.

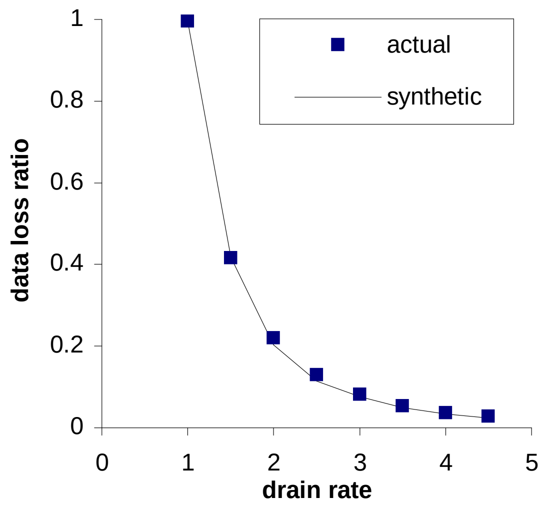


(a) The Silence of the Lambs

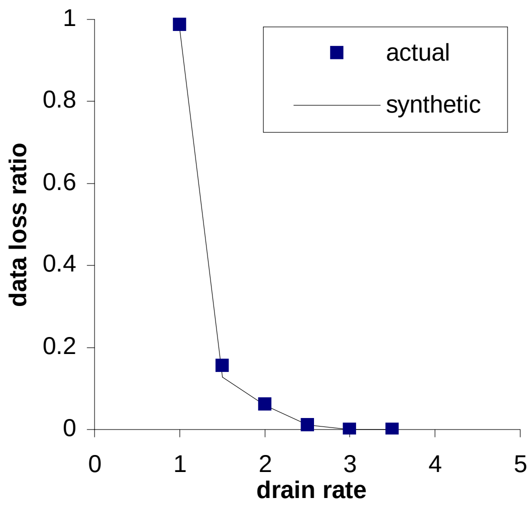


(b) Star Wars IV

Fig. 72. Overflow data loss ratio of the original and synthetic enhancement layer traffic for $c = 10$ ms for (a) *The Silence of the Lambs* and (b) *Star Wars IV*.



(a) The Silence of the Lambs



(b) Star Wars IV

Fig. 73. Overflow data loss ratio of the original and synthetic enhancement layer traffic for $c = 30$ ms for (a) *The Silence of the Lambs* and (b) *Star Wars IV*.

CHAPTER VI

CONCLUSION AND FUTURE WORK

The ideas presented in this document have been expressed in terms of an R-D modeling framework and a traffic model for scalable video coders, with the final goal of providing high quality video to end users. In this chapter, we summarize the major work we did and indicate some future directions for extension of the work.

A. Conclusion

Rate-distortion analysis has attracted great research interest after Shannon's work was published [97]. The focus of previous work has been to a large extent the derivation into some *ideal* bounds, which give us insight of achievable and non-achievable regions but are not directly applicable in practice. In stead, one goal in this work is to provide a practically useful R-D function for scalable coders.

In Chapter III, we first modeled the statistical properties of the input to scalable coders and then presented a detailed analysis of rate and distortion for scalable coders. We also reviewed the performance bound for a generic hybrid coder using motion-compensated prediction. Based on the understanding of scalable coding processes and approximation theory, we derived a distortion model and an operational R-D model. Although this R-D model is accurate, its complex format limits its usage in video streaming applications.

Therefore, we proposed another operational R-D model for streaming applications. We expressed it in the PSNR domain for the convenience of quality control. Interestingly, we found that in the PSNR domain, both our R-D model and the theoretical upper bound in [81] have a similar concave shape in the working range of scalable coders, which also matches the trend of actual R-PSNR curves.

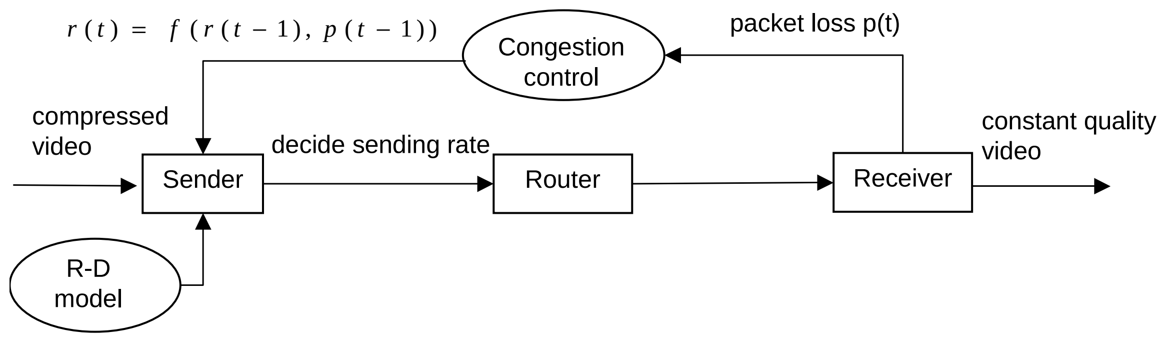


Fig. 74. R-D based quality control.

In view of the inherent lack of stable quality associated with the base layer, we provided a quality control algorithm to provide constant quality video to end users in both CBR and VBR channels. In CBR channel, the algorithm proposed in Chapter IV performs better than most existing constant quality algorithms, in regard to both computational cost and performance. Furthermore, we studied modified Kelly control and showed that it can provide a stable environment for video transmission. Thus, we coupled our R-D model with this controller to achieve constant quality even under varying network conditions. The whole work in Chapter III and IV can be depicted in Fig. 74.

In Chapter V, we presented a framework for modeling H.26L and MPEG-4 multi-layer full-length VBR video traffic. This work precisely captured the inter- and intra-GOP correlation in compressed VBR sequences, by incorporating wavelet-domain analysis into time-domain modeling. Whereas many previous traffic models are developed at slice-level or even block-level [95], our framework uses frame-size level, which allows us to examine the loss ratio for each type of frames and apply other methods to improve the video quality at the receiver. We also proposed novel methods to model cross-layer correlation in multi-layer sequences and successfully described the inter-layer correlation.

B. Future Work

In future work, we are interested in designing peer-to-peer streaming systems, where scalable video coders will play an important role and our traffic model will be helpful in its design.

A peer-to-peer streaming system differs from a general peer-to-peer system in three aspects: (1) Peer-to-peer video streaming uses streaming mode and has high user requirements on video quality; (2) In a peer-to-peer video streaming system, a requesting peer can also play the role of a supplying peer as long as a certain amount of media data has been stored; (3) A requesting peer in a peer-to-peer streaming system can receive video data from multiple supplying peers simultaneously, while a requesting peer in a general peer-to-peer system usually only has one supplying peer at one time instant.

There are two challenges in designing a peer-to-peer streaming system. One is to cooperate multiple supplying peers with high bandwidth utilization, and the other is to ensure a continuous playback with graceful quality adaptation. To address these two issues, we plan to design a scalable peer-to-peer video streaming system. Although a fine granularly scalable coded bitstream is preferred, general layered coded bitstreams are also applicable.

In the proposed scheme, we will abide by a *differentiated admission policy*, which means that if a supplying peer has enough resource to provide service to several requesting peers, we admit the requesting peer with the highest outgoing bandwidth. Intuitively, this policy has two benefits: (1) It will quickly increase the system capacity. If a requesting peer with the highest outgoing bandwidth has been admitted, sometime later it will become another supplying peer and is able to contribute more to the system than those peers with less outgoing bandwidth; (2) It will encourage the requesting peers to offer more outgoing bandwidth.

In what follows, we discuss how to cooperate supplying peers in this scheme.

1. Supplying Peers Cooperation System

Assume that for each requesting peer P_r, there is a supplying peer set P_s, which includes M supplying peers $P_s^1, P_s^2, \ldots, P_s^M$ at time t and these supplying peers are selected via existing peer-to-peer lookup mechanisms (e.g., [101]). We also define the incoming bandwidth of P_r is I_r and the outgoing bandwidth of P_r is O_r.

It is obvious that if a supplying peer P_s^i has the higher layers of the data stream, it must also have the lower layers. Since the base layer bandwidth is guaranteed, we know that the outgoing bandwidth O_r is always larger than or equal to the base layer bandwidth W_b. We describe the cooperation scheme as follows:

- To maximize the outgoing bandwidth of supplying peers, we select the first supplying peer as the lower layer supplying peer. Each packet is labeled with a layer number and a packet number.

- After transmitting the base layer (which is CBR coded in FGS coders), the incoming bandwidth of requesting peer P_r is updated to $I_r - W_b$. Although supplying peer P_s^M has the highest outgoing bandwidth, its sending rate might be slow due to various reasons (e.g., requests from other peers). If the enhancement layer can be finely divided, the requesting peer will be able to allocate different portion of the enhancement layer to different supplying peers to achieve fast transmission and better video quality.

- If a supplying peer P_s^i fails, the buffer at the requesting peer side will allow a quick supplying-peer switch without quick quality degradation. If no other supplying peers can take over the data that P_s^i used to transmit, the sending portion of other supplying peers will be adjusted and the video quality at the receiver might be degraded.

In addition, a quality control scheme is often in demand for continuous playback.

2. Scalable Rate Control System

Since the current best-effort Internet does not provide any QoS guarantees to video applications, end users often suffer from quality fluctuations and playout starvation (i.e., receiver-buffer underflow). While the former mainly results from varying bandwidth, the latter happens when the receiver buffer is empty and the playout rate is faster than the incoming frame rate. Many studies have been conducted to provide good video quality to end users. Steinbach *et al.* [100] propose a client-controlled method to flexibly scale the playout rate to prevent playout starvation. However, end users often prefer constant playout rate.

Thus, as an alternative, adaptive rate control mechanisms are proposed to adjust the sending rate according to the available bandwidth and the feedback from receiver buffers [69], [88], [94]. The fundamental idea of these mechanisms is to dynamically allocate bandwidth. When the total bandwidth of all available supplying peers is insufficient to support the requested bitstream from a requesting peer P_r, P_r can either request more frames covering fewer number of layers or fewer frames covering more layers. The switch threshold T_H is decided by buffer condition, playout rate, and available incoming bandwidth I_r.

REFERENCES

[1] P. Abry and V. Darryl, “Wavelet analysis of long-range-dependent traffic,” *IEEE Trans. Inform. Theory*, vol. 44, Jan. 1998.

[2] J. G. Apostolopoulos and S. J. Wee, “Video compression standards”, available at http://web.mit.edu/afs, Apr. 2002.

[3] D. Bansal and H. Balakrishnan, “Binomial congestion control algorithms,” in *Proc. IEEE INFOCOM*, Anchorage, Alaska, Apr. 2001, pp. 631–640.

[4] W. R. Bennett, “Spectra of quantized signals,” *Bell Sys. Tech. Journal*, vol. 27, pp. 446–472, July 1948.

[5] T. Berger, *Rate Distortion Theory*, Englewood Cliffs, NJ: PrenticeHall, 1971.

[6] J. A. Bilmes, “A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden Markov models,” International Computer Science Institute, Berkeley, California, Apr. 1998.

[7] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An architecture for differentiated services,” *IETF RFC 2475*, 1998.

[8] R. Braden, D. Clark, and S. Shenker, “Integrated services in the internet architecture: An overview,” *IETF RFC 1633*, 1994.

[9] K. Chandra and A. R. Reibman, “Modeling one- and two-layer variable bit rate video,” *IEEE/ACM Trans. on Networking*, vol. 7, pp. 398–413, June 1999.

[10] D. Clark and W. Fang, “Explicit allocation of best effort packet delivery service,” *IEEE/ACM Trans. on Networking*, vol. 6, pp. 362–373, Aug. 1998.

[11] J.-J. Chen and D.W. Lin, “Optimal bit allocation for coding of video signals over ATM networks,” *IEEE. J. on Sel. Areas in Comm.*, vol.15, pp. 1002–1015, Aug. 1997.

[12] T. P.-C. Chen and T. Chen, “Markov modulated punctured auto-regressive processes for video traffic and wireless channel modeling,” in *Packet Video*, Apr. 2002.

[13] T. Chiang and Y. Q. Zhang, “A new rate control scheme using quadratic distortion model,” *IEEE Trans. on CSVT*, vol. 7, pp. 246–250, Feb. 1997.

[14] A. Cohen and J.-P. D’ales, “Nonlinear approximation of random functions,” *SIAM Journal on Appl. Math*, vol.57, pp. 518–540, Apr. 1997.

[15] A. Cohen, I. Daubechies, O. G. Guleryuz, and M.T. Orchard, “On the importance of combining wavelet-based nonlinear approximation with coding strategies,” *IEEE Trans. on Information Theory*, vol. 48, pp. 1895 - 1921, July 2002.

[16] A. L. Corte, A. Lombardo, S. Palazzo, and S. Zinna, “Modeling activity in VBR video sources,” *Signal Processing: Image Communication*, vol. 3, pp. 167–178, June 1991.

[17] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York: John Wiley, 1991.

[18] M. Dai, D. Loguinov, and H. Radha, “Statistical analysis and distortion modeling of MPEG-4 FGS,” in *Proc. IEEE ICIP*, Barcelona, Spain, Sept. 2003, pp. 301–304.

[19] A. Demers, S. Keshav, and S. Shenker, “Analysis and simulation of a fair queuing algorithm,” in *ACM SIGCOMM*, vol.1, pp.3–26, 1990.

[20] R. A. Devore, B. Jawerth, and B. J. Lucier, “Image compression through wavelet transform coding,” *IEEE Trans. on Information Theory*, vol. 38, pp. 719 - 746, Mar. 1992.

[21] R. A. Devore, “Nonlinear approximation,” in *Acta nnumerica*, New York: Cambridge Univ. Press, Cambridge, 1998.

[22] W. Ding and B. Liu, “Rate control of MPEG video coding and recording by rate-quantization modeling,” *IEEE Trans. on CSVT*, vol.6, pp. 12–20, Feb. 1996.

[23] P. Embrechts, F. Lindskog, and A. McNeil, “Correlation and dependence in risk management: Properties and pitfalls,” available at http://www.ccfz.ch, Aug. 1999.

[24] A. Erramilli, O. Narayan, and W. Willinger, “Experimental queueing analysis with long-range dependent packet traffic,” *IEEE/ACM Trans. Networking*, vol. 4, pp. 209–223, Apr. 1996.

[25] T. Eude, R. Grisel, H. Cherifi, and R. Debrie, “On the distribution of the DCT coefficients,” in *Proc. IEEE Conf. Acoustics, Speech, Signal Processing*, vol. 5, Apr. 1994, pp. 365–368.

[26] V. Firoiu, and M. Borden, “A study of active queue management for congestion control,” in *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 1435–1444.

[27] F. H. P. Fitzek and M. Reisslein, “MPEG-4 and H.263 video traces for network performance evaluation (extended version),” available at http://www-tkn.ee.tu-berlin.de, Oct. 2000.

[28] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. on Networking*, vol. 1, pp. 397–413, Aug. 1993.

[29] S. Floyd, "TCP and explicit congestion notification," *ACM Computer Communication Review*, vol. 24, pp. 8–23, Oct. 1994.

[30] S. Floyd, M. Handley, and J. Padhye, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Stockholm, Sweden, Sept. 2000, pp. 43–56.

[31] M. Frey and S. Nguyen-Quang, "A Gamma-based framework for modeling variable-rate MPEG video sources: The GOP GBAR model," *IEEE/ACM Trans. on Networking*, vol. 8, pp. 710–719, Dec. 2000.

[32] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," in *Proc. ACM SIGCOMM*, London, UK, Aug. 1994, pp. 269–280.

[33] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. on Information Theory*, vol. 25, pp. 373–380, July 1979.

[34] A. Gersho and R. Gray, *Vector Quantization and Signal Compression*, Boston, MA: Kluwer Academic Publishers, 1992.

[35] B. Girod, "The efficiency of motion-compensating prediction for hybrid coding of video sequences," *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, pp. 1140–1154, Aug. 1987.

[36] H. Gish and J. N. Pierce, "Asymptotically efficient quantizing," *IEEE Trans. on Information Theory*, vol. IT-14, pp. 676–683, Sept. 1968.

[37] R. M. Gray, *Source Coding Theory*, Boston, MA: Kluwer Academic Publishers, 1990.

[38] H.-M. Hang and J.-J. Chen, "Source model for transform video coder and its application —Part I: fundamental theory," *IEEE Trans. on CSVT*, vol. 7, pp. 287–298, Apr. 1997.

[39] B. G. Haskell, A. Puri, A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, Boston, MA: Kluwer Academic Publishers, 2002.

[40] Z. He and S. K. Mitra, "A unified rate-distortion analysis framework for transform coding," *IEEE Trans. on CSVT*, vol. 11, pp. 1221–1236, Dec. 2001.

[41] D. P. Heyman, A. Tabatabai, T. V. Lakshman, "Statistical analysis and simulation study of video teleconference traffic in ATM networks," *IEEE Trans. on CSVT*, vol. 2, pp. 49–59, Mar. 1992.

[42] D. P. Heyman, "The GBAR source model for VBR video conferences," *IEEE/ACM Trans. on Networking*, vol. 5, pp. 554–560, Aug. 1997.

[43] C.-Y. Hsu, A. Ortega, and A. Reibman, "Joint selection of source and channel rate for VBR video transmission under ATM policing constraints," *IEEE Journal on Selected Areas in Communication*, vol. 15, pp. 1016–1028, Aug. 1997.

[44] C. Huang, M. Devetsikiotis, I. Lambadaris, and A. R. Kaye, "Modeling and simulation of self-similar variable bit rate compressed video: A unified approach," in *Proc. ACM SIGCOMM*, Cambridge, MA, Aug. 1995, pp. 114–125.

[45] D. Huffman, "A method for the construction of minimal redundancy codes," in *Proc. IRE*, Sept. 1952, pp. 1098–1101.

[46] H. E. Hurst, "Long-term storage capacity of reservoirs," *Trans. on American Society of Civil Engineers*, vol. 116, pp. 770-799, 1951.

[47] T. Y. Hwang and P. H. Huang, "On new moment estimation of parameters of the Gamma distribution using its characterization," *Annals of the Institute of Statistical Mathematics*, vol. 54, Issue 4, 2002.

[48] ISO/IEC JTC1, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5Mb/s–Part2: video," ISO/IEC 11172-2 (MPEG-21), 1993.

[49] ISO/IEC JTC1, "Information technology-coding of audio-visual objects- Part2: video," ISO/IEC 14496-2 (MPEG-4), 1999.

[50] ITU-T, "Codec for videoconferencing using primary digital group transmission," ITU-T Recommendation H.120; version 1, 1984; version 2, 1988.

[51] ITU-T, "Video codec for audiovisual services at $p \times 64$ kbits/s," ITU-T Recommendation H.120; version 1, 1990; version 2, 1993.

[52] ITU-T and ISO/IEC JTC1, "Generic coding of moving pictures and associated audio informaiton–Part2: video," ISO/IEC 13818-2 (MPEG-2), 1994.

[53] ITU-T, "Video coding for low bitrate communication," ITU-T Recommendation H.263; version 1, 1995; version 2, 1998.

[54] N. Jayant and P.Noll, *Digital Coding of Waveforms*, Englewood Cliffs, NJ: Prentice Hall, 1984.

[55] JPEG, "JPEG2000 part I final committee draft version 1.0," ISO/IEC JTCI/SC29 WGI, Mar. 2000.

[56] S.-R. Kang, Y. Zhang, M. Dai, and D. Loguinov, "Multi-layer active queue management and congestion control for scalable video streaming," in *Proc. IEEE ICDCS*, Tokyo, Japan, Mar. 2004, pp. 768–777.

[57] K. Kar, S. Sarkar, and L. Tassiulas, "Simple rate control algorithm for max total user utility," in *Proc. IEEE INFOCOM*, Anchorage, Alaska, Apr. 2001, pp. 133–141.

[58] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[59] M. Krunz and S. K. Tripathi, "On the characterization of VBR MPEG streams," in *Proc. of ACM SIGMETRICS*, Seattle, WA, June 1997, pp. 192–202.

[60] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," in *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 1323–1332.

[61] D. Leviatan and I. A. Shevchuk, "Coconvex approximation," *Journal of Approx. Theory*, vol. 118, pp. 20–65, 2002.

[62] A. Lombardo, G. Morabito, and G. Schembra, "An accurate and treatable Markov model of MPEG-Video traffic," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 1998, pp. 217–224.

[63] A. Lombardo, G. Morabito, S. Palazzo, and G. Schembra, "A Markov-based algorithm for the generation of MPEG sequences matching intra- and inter-GOP correlation," *European Trans. on Telecommunications*, vol. 12, pp. 127–142, Mar./Apr. 2001.

[64] S. H. Low and D. E. Lapsley, “Optimization flow control I: Basic algorithm and convergence,” *IEEE/ACM Trans. on Networking*, vol. 7, pp. 861–874, Dec. 1999.

[65] W. Li, “Overview of fine granularity scalability in MPEG-4 video standard,” *IEEE Trans. on CSVT*, pp. 301–317, Mar. 2001.

[66] J. Lin and A. Ortega, “Bit-rate control using piecewise approximation rate-distortion characteristics,” *IEEE Trans. on CSVT*, vol. 8, pp. 446–459, Aug. 1998.

[67] F. Ling, W. Li, and H. Sun, “Bitplane coding of DCT coefficients for image and video compression,” *Proc. SPIE Visual Communications and Image Processing*, San Jose, CA, Jan. 1999, pp. 500–508.

[68] D. Liu, E. I. Sára, and W. Sun, “Nested auto-regressive processes for MPEG-encoded video traffic modeling,” *IEEE Trans. on CSVT*, vol. 11, pp. 169–183, Feb. 2001.

[69] T. Liu, H.-J. Zhang, W. Qi, and F. Qi, “A systematic rate controller for MPEG-4 FGS video streaming,” *Multimedia Systems*, vol. 8, Dec. 2002.

[70] D. Loguinov and H. Radha, “Increase-decrease congestion control for real-time streaming: Scalability,” in *Proc. IEEE INFOCOM*, New York, June 2002, pp. 525–534.

[71] S. Ma and C. Ji, “Modeling video traffic using wavelets,” *IEEE Communication Letters*, vol. 2, no. 4, pp. 100–103, Apr. 1998.

[72] S. Ma and C. Ji, “Modeling heterogeneous network traffic in wavelet domain,” *IEEE/ACM Trans. on Networking*, vol. 9, pp. 634–649, Oct. 2001.

[73] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Trans. on Comm.*, vol. 36, pp. 834–844, July 1988.

[74] S. Mallat and F. Falzon, "Analysis of low bit rate image transform coding," *IEEE Trans. on Signal Processing,* vol.46, pp. 1027–1042, Apr. 1998.

[75] L. Massoulié, "Stability of distributed congestion control with heterogeneous feedback delays," *IEEE Trans. on Automatic Control*, vol. 47, pp. 895–902, June 2002.

[76] B. Melamed and D. E. Pendarakis, "Modeling full-length VBR video using Markov-renewal-modulated TES models," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 600–611, June 1998.

[77] J. L. Mitchell, *MPEG Video: Compression Standard*, Boston, MA: Kluwer Academic Publishers, 2002.

[78] MPEG, "Coding of moving pictures and audio," ISO/IEC JTC1/SC29/WG11 N3908, Jan. 2001.

[79] F. Muller, "Distribution shape of two-dimensional DCT coefficients of natural images," *Electronics Letters*, vol. 29, Oct. 1993.

[80] A. N. Netravali and B. G. Haskell, *Digital Pictures Presentation, Compression, and Standards.* New York: Plenum, 1988.

[81] J. O'Neal, T. Natarajan, "Coding isotropic images," *IEEE Trans. on Information Theory*, vol. 23, pp. 697–707, Nov 1977.

[82] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 23–50, Nov. 1998.

[83] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP reno performance: A simple model and its empirical validation," *IEEE/ACM Trans. on Networking*, vol. 8, pp. 133–145, Apr. 2000.

[84] J. Postel, "User datagram protocol," *RFC 768*, IETF standard, Aug. 1980.

[85] J. Postel, "Transmission control protocol C DARPA Internet program protocol specification," *RFC 793*, IETF standard, Sept. 1981.

[86] H. Radha, M. V. Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. on Multimedia*, vol. 3, pp. 53–68, Mar. 2001.

[87] M. Reisslein, J. Lassetter, S. Ratnam, O. Lotfallah, F. H. P. Fitzek, and S. Panchanathan, "Video traces for network performance evaluation," available at http://trace.eas.asu.edu, 2004.

[88] R. Rejaie, M. Handley, "Quality adaptation for congestion controlled video playback over the Internet," in *Proc. of ACM SIGCOMM*, Cambridge, MA, Sep. 1999, pp. 189–200.

[89] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for real-time streams in the Internet," in *Proc. IEEE INFOCOM*, New York, USA, Mar. 1999, pp. 1337–1345.

[90] V. J. Ribeiro, R. H. Riedi, M. S. Crouse, and R. G. Baraniuk, "Multiscale queuing analysis of long-range-dependent network traffic," in *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 1026–1035.

[91] J. Rissanen and G. Langdon, "Arithmetic coding," *IBM Journal of Research and Development*, vol. 23, pp. 149–162, Mar. 1979.

[92] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," in *Proc. of the 20th Annual Conference on Local Computer Networks*, Minneapolis, MN, Oct. 1995, pp. 397–406.

[93] O. Rose, "Simple and efficient models for variable bit rate MPEG video traffic," in *Performance Evaluation*, vol. 30, pp. 69–85, 1997.

[94] D. Saparilla and K. Ross, "Optimal streaming of layered video," in *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000, pp. 737–746.

[95] U. K. Sarkar, S. Ramakrishnan, and D. Sarkar, "Modeling full-length video using Markov-modulated Gamma-based framework," *IEEE/ACM Trans. on Networking*, vol. 11, pp. 638–649, Aug. 2003.

[96] M. van der Schaar, "System and network-constrained video compression," Ph.D. dissertation, Eindhoven University of Technology and Delft University of Technology, Netherlands, 2001.

[97] C. E. Shannon, "A mathematica theory of communication," *Bell Syst. Tech. Journal*, vol. 27, pp. 379–423, 1948.

[98] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," *IEEE/ACM Trans. on Networking*, vol. 4, pp. 375–385, June 1996.

[99] S. R. Smoot and L. A. Rowe, "Study of DCT coefficient distributions," in *Proc. SPIE Symposium on Electr. Imaging*, San Jose, CA, vol. 2657, Jan. 1996.

[100] E. Steinbach, N. Farber, and B. Girod, "Adaptive playout for low latency video streaming," in *Proc. IEEE ICIP*, Thessaloniki, Greece, Oct. 2001, pp. 962–965.

[101] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. "Chord: A scalable peer-to-peer lookup service for Internet applications," in *Proc. ACM SIGCOMM*, San Diego, CA, Aug. 2001, pp. 149–160.

[102] G. J. Sullivan and T. Wiegand, "Rate-Distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 74–90, Nov. 1998.

[103] D. S. Taubman, "Directionality and scalability in image and video compression," Ph.D. dissertation, University of California At Berkeley, Berkeley, CA, 1994.

[104] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, New York: McGraw-Hill, 1979.

[105] Q. Wang, Z. Xiong, F. Wu, and S. Li, "Optimal rate allocation for progressive fine granularity scalable video coding," *IEEE Signal Processing Letters*, vol. 9, pp. 33–39, Feb. 2002.

[106] Y. Wang, M. T. Orchard, and A. R. Reibman, "Multiple description image coding for noisy channels by pairing transform coefficients," in *Proc. IEEE Workshop on Multimedia Signal Processing*, Princeton, NJ, June 1997, pp. 419–424.

[107] Y. Wang, J. Ostermann, and Y.-Q. Zhang, *Video Processing and Communications*, NJ: Prentice Hall, 2001.

[108] D. Wu, Y. T. Hou, B. Li, W. Zhu, Y.-Q. Zhang, and H. J. Chao, "An end-to-end approach for optimal mode selection in Internet video communication: Theory and application," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1–20, June 2000.

[109] D. Wu, Y. T. Hou, W. Zhu, H.-J. Lee, T. Chiang, Y.-Q. Zhang, and H. J. Chao, "On end-to-end architecture for transporting MPEG-4 video over the Internet," *IEEE Trans. on CSVT*, vol. 10, pp. 923–941, Sept. 2000.

[110] D. Wu, Y.T. Hou, W. Zhu, Y.-Q. Zhang, J.M. Peha, "Streaming video over the Internet: approaches and directions," *IEEE Trans. on CSVT*, vol. 11, pp. 282–300, Mar. 2001.

[111] G. S. Yovanof and S. Liu, "Statistical analysis of the DCT coefficients and their quantization error," in *Conf. Rec. 30th Asilomar Conf. Signals, Systems, Computers*, Pacific Grove, CA, Nov. 1996, pp. 601–605.

[112] Y. Zhang, S-R. Kang, and D. Loguinov, "Delayed stability and performance of distributed congestion control," in *Proc. ACM SIGCOMM*, Portland, OR, Aug. 2004, pp. 307–318.

[113] L. Zhao, J. W. Kim, and C.-C. Kuo, "MPEG-4 FGS video streaming with constant-quality rate control and differentiated forwarding," in *Proc. SPIE Visual Communications and Image Processing*, San Jose, CA, Jan. 2002, pp. 230–241.

[114] X. J. Zhao, Y. W. He, S. Q. Yang, and Y. Z. Zhong, "Rate allocation of equal image quality for MPEG-4 FGS video streaming," in *Packet Video*, Pittsburgh, PA, Apr. 2002.

[115] J.-A. Zhao, B. Li, and I. Ahmad, "Traffic model for layered video: An approach on Markovian arrival process," in *Packet Video*, Nantes, France, Apr. 2003.

VITA

Min Dai received her B.S. and M.S. degree in precise instruments from Shanghai Jiao Tong University, China, in 1996 and 1998, respectively. She has been pursuing her Ph.D. degree in electrical engineering at Texas A&M University since 1999.

She was a research intern with LSI Logic Company, San Jose, CA, from January 2002 to August 2002. Afterwards, she joined the Internet Research Lab, Department of Computer Science, Texas A&M University.

Her research interests include scalable video streaming, video traffic modeling, and image denoising. She may be contacted at:

Min Dai C/O Shanren Dai

11 Shucheng Road, the 8th Floor

Hefei, Anhui, 230001

P. R. China