Wavelet-Based Adaptive Video Coding for Packet-Switching Networks

by

Ye Gu

S.B., Electrical Engineering
Massachusetts Institute of Technology
Cambridge, Massachusetts
1991

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY
AUGUST, 1994

Signature of Author _____
Department of Electrical Engineering and Computer Science
August 21, 1994

Certified by _____
John T. Wroclawski
Research Scientist, Thesis Supervisor

Accepted by _____
Professor Federic R. Morgenthaler
Chairman, Department Committee on Graduate Students

Wavelet-Based Adaptive Video Coding for Packet-Switching Networks

by

Ye Gu

Submitted to the Department of Electrical Engineering and Computer Science
on August 21, 1994
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Wavelet-based techniques are becoming increasingly widely used for encoding image and video signals. Wavelet-coded video streams are highly compressible and have considerable amount of error resilience built-in. These two features are critical to the performance of a real-time video coding system that communicates over a lossy transmission medium.

A wavelet-based adaptive coding scheme is proposed in this thesis. It is specifically engineered to transport real-time video over packet-switching networks that provide services on a best-effort basis. The goal is to maximize the perceptual quality of the delivered video and to accommodate other flow controlled traffic such as TCP. The proposed adaptive scheme utilizes two key mechanisms — a multiresolution codec and an overload controller. The multiresolution codec is capable of encoding and decoding a video stream at different bit-rates. The particular bit-rate chosen for encoding or decoding a video frame is usually determined by external factors such as available system or network resources. The overload control unit in the sender performs end-to-end flow control based on various feedback information from the network and/or the receiver. It then estimates the output rate of the encoder. Provided that the encoder strictly follows this estimate, network congestion and packet loss rate will be minimized. In turn, this provides opportunities for better error concealment at the receiver.

Thesis Supervisor: John T. Wroclawski
Title: Research Scientist

Acknowledgments

During this period of my graduate study, I am very fortunate to have received help, support and encouragement from numerous individuals. I am deeply indebted to all of them. In particular, I would like to express my gratitude to the following individuals.

To my advisor John T. Wroclawski, thanks for all the insights and guidance. Needless to say, I could not have done this research without the many intellectually stimulating conversions we have had. As usual, your constructive opinions on many different subjects and issues have always made these conversations very interesting and enjoyable.

To David D. Clark, John T. Wroclawski and Karen R. Sollins, thanks for making the Advanced Network Architecture Group a fun and challenging place to do research.

To Christopher J. Lefelhocz, thanks for pressing me to work harder by finishing your thesis way ahead of me. You have given me so much helps and support in the past two years. Your good sense of humor is always a big "plus", especially during those long working hours.

To Janey C. Hoe, my "phantom" officemate for the summer, thanks for being such a good friend. You always there to listen to my complaints and offer advice and consolation. Of course, there is no lack of humorous and good-natured sarcasm either.

To Karen Ho, thanks for being always cheerful and energetic, even at 3 o'clock in the morning. Also, thanks for getting us all hooked with composing those "Top Ten" lists, and making staying-up a more memorable experience.

To Tim Sheperd, thanks for having a "long" answer to every of my random questions. Best of luck in finding your way out of M.I.T. in a reasonable time frame.

To Paul D. Bosco and Charles L. Compton, thanks for all the support, friendship and, of course, caffeine. Ever since the Green Team, we have been a great team. Champagne-wishes to the success of a very "cool" project called the NMIS.

To Michael H. Coen, Stacy Ho, Upendra V. Chaudhari, Jeff H.-K. Kuo, Ravi Soundararajan, and Lalit Jain, thanks. I will never forget those many discussions of life, M.I.T. and life at M.I.T.

To Richard C.-C. Chen and Yongmei Wu, thanks for dragging me away from work from time to time and keeping me "in sync" with real life outside of M.I.T.

Finally, to my parents, thanks for all the love and support. I know that you will always be there for me.

To my father & mother

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Advancing multimedia technologies, coupled with rapidly increasing computer power and storage capacity in personal computers and workstations, are accelerating the integration of multimedia-capable applications into the mainstream desktop computing. Multimedia applications often use digital video to enrich the visual presentation of their information contents. Concurrently, the evolution of current generation of packet-switching networks into Integrated Services Packet Networks (ISPN's) is enabling the delivery of multimedia contents in a unified way. An ISPN is capable of transporting different classes of services simultaneously within the same network infrastructure. In particular, an ISPN is designed to carry real-time traffic across the network in a timely manner. Different grades of services can be offered by an ISPN, with either strictly or loosely guaranteed quality of service (i.e., bandwidth, delay and delay jitter).

Digital video is notorious for its enormous bandwidth requirement. Fortunately, digital video is highly compressible because it contains a large amount of redundant information. Compressed video often exhibits bursty output, especially when a constant level of visual quality is preserved [23, 37]. Such an output characteristic makes compressed video an excellent candidate for transmission over packet-switching networks [23, 27, 28]. This transmission of real-time video via a packet-based transport service is called *packet video*. The

statistical multiplexing nature of packet-switching networks provides opportunities for very high network resource utilization for video traffic aggregation [23].

However, the advantages of packet-switching networks for real-time video delivery can be overweighed by the effects of packet loss, delay and jitter if these issues are not resolved satisfactorily. This thesis will examine the negative impacts of packet loss, delay and delay jitter on the quality of packet video and propose a system that combines a wavelet-based multiresolution video codec and an end-to-end flow control mechanism to improve the performance of a packet video system.

1.1 Packet Video

The most fundamental problem of any packet-switching network is that it cannot guarantee timely and error-free delivery of all packets for a real-time session. Due to the statistical multiplexing of the network, queueing delay at a switch varies with the amount of traffic it needs to route. Consequently, packet-switching networks cannot bound end-to-end delays under general circumstances. [38] proves that guaranteed delay bounds can be achieved in packet-switching networks, under a number of constraints. [10] proposes the ISPN architecture that unifies this type of *guaranteed services* with a new service class called *predicted services* and the traditional *datagram service*. Although delay can be bounded (either strictly or loosely) in this type of networks, error-free packet delivery is not assured under a given delay bound.

In fact, all packet-switching networks suffer from a certain degree of packet loss. Packet losses can be caused by transmission errors in a link or by link/node failures along a path or (mostly) by congestion in the network. Packet losses may produce error multiplication effects if a feedback error recovery mechanism is used. Therefore, packet loss rate must be minimized. [23] argues that if packet sources can adjust their output rates to respond to the

network dynamics under overloading conditions, network congestion can be greatly attenuated. In turn, this can reduce packet losses.

In the context of conventional data transport services such as `ftp` or `telnet`, retransmission is used to recover from a packet loss. In the context of real-time data communications, such a retransmission technique is not nearly as effective. A real-time packet must arrive before its play-back point (in time) to be useful. If it is retransmitted, it may arrive too late to be used in the play-back, especially when the round-trip delay is large. Furthermore, retransmission can induce positive feedback loops and cause even more packet drops. Luckily, for the majority of packet video applications, retransmission is not necessary if error resilience is built into the coded video streams. A video decoder can apply error concealment techniques to further reduce visual degradation caused by lost packets.

In addition to packet losses, packet-switching networks may produce out-of-order arrival of temporally adjacent packets since packets are routed individually and independently. Unless the network has explicit mechanisms to preserve packet ordering, packets can be delivered out-of-order. For compressed video, each frame is typically sent in multiple packets. However, since these packets are buffered at a receiving station, reordering of packets can be done easily. An Application Level Framing (or ALF) technique [9] can be used to further improve the system performance. The ALF asserts that end-to-end delays can be reduced if out-of-order processing can be done on partially received data in units of Application Data Units, or ADU's. An ADU is an application-specific data unit that can be coded, transmitted, decoded, and manipulated independent of all other ADU's. For video applications, an ADU can be chosen from tiled segments that comprise of each image frame. Generally, the smaller is the size (or granularity) of an ADU, the more effective ALF processing becomes and the more

end-to-end delays are improved. The ALF should be utilized for those interactive packet video applications that are particularly delay sensitive.

In summary, there are several techniques that a coding system can utilize to enhance its performance. A high-performance packet video codec for distributed interactive video applications should provide error resilience, graceful degradation of image quality under heavy load, out-of-order processing, and scalability. Fortunately, all these objectives can be achieved in a unified framework that utilizes both wavelet-based multiresolution coding and adaptive end-to-end flow control.

1.2 Wavelet-Based Adaptive Coding for Packet Video Applications

In the past decade, wavelet-based signal processing has gained increasing attention in both theoretical and applied engineering communities. Wavelet transforms have been viewed as a viable alternative to classical Fourier transforms.

Wavelet-based coding [33, 34, 1] is a member of a special class of coding schemes called *multiresolution coding*. Wavelet-based coding techniques have low computation complexity, and produce both compact representations and good compression results. Wavelet-coded signals form multiresolution hierarchies that are essential to scalable representation of signals, using discrete wavelet transforms. But more importantly, for packet video applications, wavelet-based coding provides robustness against packet losses and other transmission errors, graceful adjustment of video quality under heavy load and out-of-order processing at various granularities.

Both forward and inverse discrete wavelet transforms can be implemented as filtering operations. That is, an input image frame (or a coded one) is convolved with a set of discrete

wavelet filters at each decomposition (or reconstruction) step. Since many wavelet filters have finite and often very narrow support, each filtering operation can be performed over small segment of an input. Any segment of size that is greater than the span of a wavelet filter can be chosen as an ADU for out-of-order processing, constrained only by the amount of overhead introduced to keep track of processed/unprocessed data. This thesis will not pursue out-of-order processing. However, it is pointed out here that out-of-order processing can be used with the proposed adaptive coding system to further enhance the performance.

A wavelet transformed signal consists of a set of recursively structured frequency subbands. Each subband has a different amount of visual information. The higher the frequency, the less visual information a subband contains. A discrete wavelet transform extracts edge information of an image and stores it in high frequency subbands. If data are lost or received erroneously in these subbands, the impact on perceptual image quality is relatively small since the image frame is incorrectly reconstructed only around sharp edges. Therefore, wavelet-coded video is inherently resilient to errors occurred in the high frequency subbands. On the other hand, the lowest frequency subband is extremely sensitive to errors. The lowest frequency subband records the energy information of an image. If this information is lost or corrupted, the result is disastrous — the decoded image will show only the edges in the original. So, error concealment is largely limited to the low frequency subband for wavelet coded images.

Because of the distribution of visual information among subbands, error concealment methods should be designed and applied accordingly. For packet video applications, errors are mainly caused by packet losses. If losses occur in the high frequency subbands, there are a number of ways to visually conceal these errors. Lost packet data may simply be replaced with all 0s. This blurs the corresponding reconstructed image around its edges. Alternatively, one of

the more elaborate data-interpolation techniques may be applied, at the expense of more computation time. Information in a high frequency subband can be interpolated spatially from other subbands within a same image frame, or predicted temporally from the same subband from previous frames. Spatial interpolation is expected to give better results because motion displacement is not involved.

Visual error concealment techniques for the lowest frequency subband are more limited. Spatial interpolation of high frequency subbands is ineffective in recovering much useful information. If there is little motion between the current frame and the previous frame, a simple temporal interpolation can be used — the lowest frequency subband in the current image is replaced by that in the previous image. Motion estimation can be applied to further improve the result of the temporal interpolation.

A wavelet-based decoder, when equipped with visual error concealment mechanisms for all subbands, should be able to tolerate significant amounts of packet losses or other transmission errors. It can reconstruct image frames from partially received information with minimal quality degradation. This thesis will utilize advantages of visual error resilience afforded by discrete wavelet transforms.

Ultimately, packet loss rates should be reduced. As mentioned earlier, most packet losses in a network are caused by congestion. Packet loss rate can be attenuated significantly if an effective congestion control mechanism is incorporated into the overall coding system design. A network adaptive video system can actively assist the network to alleviate congestion by modulating the output bit-rate of the encoder to respond to the current traffic level in the network.

1.3 Design Goals

[23] has recently proposed a general approach to packet video coding. The new scheme is aimed to reduce network congestion and consequently to minimize packet losses. Under this model, the video encoder will interact closely with a network transport agent who informs the encoder about the current traffic level in the network on a regular basis. Both local and end-to-end feedback mechanisms are employed. When a congestion condition rises, the network agent will advise the video encoder to send less packets. This adaptation will lessen the severity of the congestion. When the congestion period is over, the network agent will notify the encoder that more bandwidth has become available. If such rules of interaction are strictly followed by the encoder and exercised by all the packet video sources, better network performance and resource utilization will be achieved.

This thesis will adopt this network-integrated coding model, but propose, implement and analyze specific algorithms required to construct the adaptive coding system for packet video applications. Specific areas of performance improvement will include:

- Network throughput — Statistical multiplexing of packet-switching networks provides better channel utilization for the aggregation of bursty traffic. An adaptive video coding process offers even more efficient uses of a communication channel by discriminating wavelet-coded data with respect to their visual importance and utilizing available network resources accordingly. At the same time, the adaptive coding reduces packet loss rate. Together, these two factors increase the effective network throughput.
- Network latency — Network delay is mainly contributed by two factors — signal propagation delay through a physical media and queueing delay at a switching node. The first factor is an invariant for any given physical link. The second factor can be

reduced however, by proper choice of queueing disciplines or other mechanisms that can lead to smaller average queue sizes. On the average, an adaptive video coding process produces less data than it would otherwise and lessens network congestion. Therefore, it is reasonable to expect that adaptive coding should improve network latency.

- Video quality under overloading conditions — An effective flow control mechanism is for packet sources to rate-regulate their output under heavy load (based on network feedback information). An adaptive video source should reduce its output rate in response to persistent network congestion conditions. At the same time, it is desirable that the rate-regulation process does not produce rapid oscillation in the perceptual quality of the video during play-back. Since a discrete wavelet transform decomposes images into subbands of different visual significance, each of them can be given a different loss preference. A subband of lesser visual importance can tolerate more losses or distortions. In the wavelet domain, adjustments in perceptual quality can be done in small steps by slowly changing the bit-rate allocated for each subband. Consequently, video quality supported by an adaptive coding system should approach its optimum for any given bandwidth.

To summarize, this thesis will explore and utilize three mechanisms in adaptive coding and control with the objective of optimizing the perceptual quality of packet video streams delivered over the network. These new mechanisms will address multiresolution coding, visual error concealment of lost data and adaptive flow control, respectively, and will be integrated into a single system architecture.

1.4 Criteria for Selecting Algorithms

Because of its timing constraints, real-time video must be coded, transmitted and decoded on a time-scale that is smaller than the interval between two adjacent video frames. This implies that video processing algorithms must have relatively low computational complexities in order to be effective in this context. In signal processing, it is well known that a higher performance system generally requires computationally more intensive algorithms. Given the real-time constraint of a packet video application, it is important that a signal processing algorithm has the lowest computational complexity possible (subject to a certain level of performance). This will allow the algorithm to be implemented on the widest range of general-purpose computer systems. The algorithms used in this thesis have all been selected with this criterion of optimizing the performance-to-complexity ratio.

1.5 Thesis Outline

Chapter two explains the wavelet theory and its applications to video coding. Chapter three details the design and implementation of a scalable video coding system. Chapter four proposes and examines the network adaptation mechanism used by the scalable video codec to optimize the performance of packet video applications. Chapter five gives the simulation results of the proposed flow control mechanism. Chapter six presents the conclusion and discusses future work.

Chapter 2

Wavelet Analysis

An effective packet video coding scheme should either provide or facilitate two functions: reducing the bandwidth requirement for the generated video stream and minimizing the visual degradation in the presence of packet losses. The first function can be achieved by removing redundancy in the video stream and the second by retaining redundancy in the video stream. Because of this conflict, the two coding requirements cannot be achieved simultaneously in the spatial-temporal domain of the original video signal. However, if a transform domain can be found that well matches the response of the human visual system, good compromise between the two can be attained when a signal is coded in that domain and transmitted. A multiresolution representation defines one such transform domain.

Mallat [33, 34] has shown that multiresolution representations are very effective means for analyzing and encoding the information content of a signal. A multiresolution representation uses a hierarchical framework to encode the signal content in a coarse-to-fine resolution pyramid. Information at each resolution is largely "orthogonal" to that at another resolution; therefore, it can be processed or encoded independently of the others (and according to the specific characteristics of the human visual system). Furthermore, Mallat has shown that the *wavelet* transform can generate a compact multiresolution representation of signals.

2.1 On Wavelets

In [39], Resnikoff gives an excellent account of the historical development of the wavelet theory. The basic idea behind wavelet transforms is to represent an arbitrary signal (of finite energy) as a weighted sum of a family of wavelet basis functions (or *wavelets*). This is the same idea as the one behind the Fourier transform, which represents a signal as a weighted sum of sinusoids. However, wavelet transforms offer additional advantages. Wavelet transforms can be performed faster than the Fourier transform because it has an $O(N)$ instead of an $O(N \log N)$ complexity. Also, a wavelet-transformed coefficient is determined by only a small number of samples in the original signal domain. Thus, wavelet transforms can be used to analyze localized features of a signal in details. This property is very important for applications such as edge detection, feature extraction and pattern matching.

A wavelet transform decomposes an input signal into an approximation signal and a set of detailed signals at a coarser resolution. In the same manner, the approximation signal can be further decomposed. This recursive transformation can be repeated as many times as is required by the signal processing or coding application. The method of analyzing a signal at different resolutions is called *multiresolution analysis*. Once decomposed, a signal is completely characterized by the set of transform coefficients; it can be reconstructed as a sum of the wavelet basis functions weighted by these transform coefficients. A wavelet transform produces a compact representation. That is, the transform coefficient set has the same number of data points as the input. Multiresolution decomposition and reconstruction schemes based on wavelet transforms are illustrated in Figure 2.1 and 2.2. There, at each level, a signal is broken into two components — one gives a lower resolution approximation of the original signal and the other contains the difference between the approximation and the original.

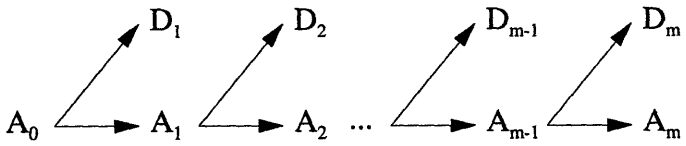

Figure 2.1. Multiresolution decomposition scheme.

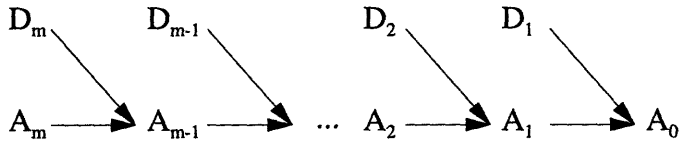

Figure 2.2. Multiresolution reconstruction scheme.

2.2 Discrete Wavelet Transforms

In the following sections, wavelets will be examined from the frequency domain. In particular, discrete wavelet transforms will be explained in terms of filtering operations. This frequency domain interpretation will, hopefully, give readers a more intuitive way of thinking about wavelet transforms.

2.2.1 DWT and the Fast Wavelet Transform

Discrete wavelet transforms are completely characterized by the four sets of coefficients (h_k), (g_k), $(\tilde{h}_k)$, and $(\tilde{g}_k)$ (as defined by equations A.4, A.6, A.9, and A.10 of Appendix A respectively). For orthogonal wavelet transforms with exact reconstruction, the coefficient sequences are related by

$$g_k = (-1)^k \overline{h_{1-k}}, \quad \tilde{h}_k = h(-k), \quad \text{and} \quad \tilde{g}_k = g(-k).$$

For biorthogonal wavelet transforms with exact reconstruction, these sequences are related by

$$g_k = (-1)^k \overline{\tilde{h}_{1-k}} \quad \text{and} \quad \tilde{g}_k = (-1)^k \overline{h_{1-k}}.$$

Let H, G, $\tilde{H}$, and $\tilde{G}$ be discrete filters with impulse responses given by the coefficient sequences (h_k), (g_k), $(\tilde{h}_k)$, and $(\tilde{g}_k)$; that is, their frequency responses are given by

$$H(\omega)=\sum_k h_k e^{-jk\omega}, \quad G(\omega)=\sum_k g_k e^{-jk\omega},$$
$$\tilde{H}(\omega)=\sum_k \tilde{h}_k e^{-jk\omega}, \quad \tilde{G}(\omega)=\sum_k \tilde{g}_k e^{-jk\omega}.$$

Mallat [33, 34] shows that the discrete wavelet decomposition and reconstruction of an arbitrary signal f with finite energy can be implemented by a scheme in Figure 2.3. This filter-bank structure is known as the *quadrature mirror filters* (QMF's) has been studied extensively [46, 16, 50, 51]. The wavelet decomposition consists of filtering of a signal f by a low-pass filter $\tilde{H}$ and a high-pass filter $\tilde{G}$, and down-sampling the resulting frequency subband signals by a factor of 2. This gives two sets of wavelet transform coefficients. Conversely, the wavelet reconstruction consists of up-sampling the wavelet coefficients by 2 and interpolative filtering (by H and G) and adding the results. Because of the down-sampling operation in the decomposition, the band-pass filters $\tilde{H}$ and $\tilde{G}$ should ideally be the "square box" filters to avoid frequency aliasing. However, it is well known that "square box" filters cannot be realized with finite length filters. Fortunately, Croiser, Esteban and Galand [17] have found a class of anti-aliasing filters, called the QMF's. These filter structures allow signal to be reconstructed from frequency subbands without aliasing. Note that the band-pass filters in Figure 2.3 are QMF's even though they are drawn as the "square box" filters.

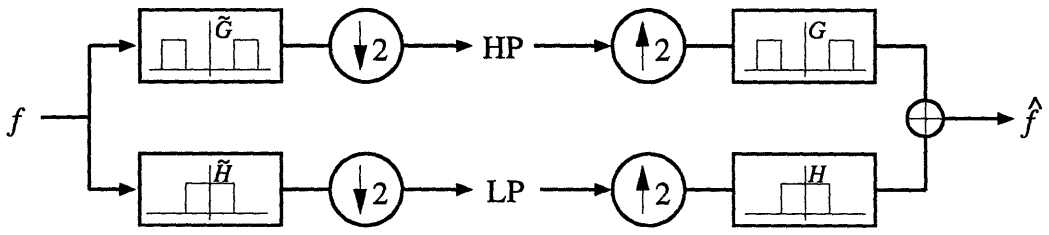

Figure 2.3. Fast wavelet transform scheme.

The frequency domain effect of the discrete wavelet transform, when applied recursively, is illustrated in Figure 2.4. First, the wavelet transform is applied to a discrete input signal and splits its frequencies into two frequency subbands (one for the frequency range $[-\pi/2, \pi/2]$ and the other for the $[-\pi, -\pi/2] \cup [\pi/2, \pi]$)[2.1] . This transformation operation generates two new signals — a low-pass filtered signal containing a coarse resolution approximation of the original and a detailed signal containing the difference. Together, the two signals fully describe the original. That is, the original input can be perfectly recovered from these two signals. To form a hierarchical multiresolution representation, the low-pass filtered signal (the gray region in Figure 2.4) is further split into another two signals. This decomposition process can be applied recursively until the desired number of detailed signal levels is obtained.

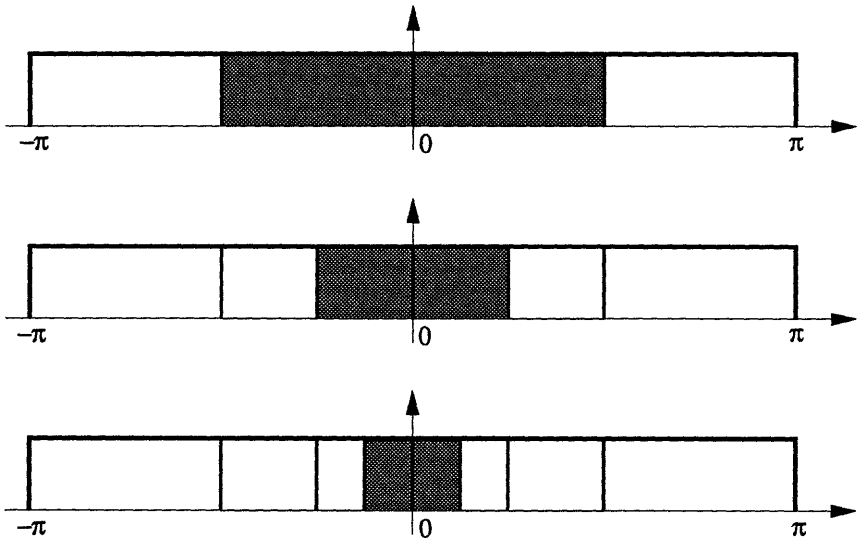

Figure 2.4. Frequency segmentation resulted from the discrete wavelet transform. Each successive picture shows the effect of applying the transform recursively to the low-pass filtered signal (the gray region) of the previous picture.

2.1 For the sake of a simpler explanation, "square box" filters are used here for illustration. In the actual implementation, these are QMFs.

As mentioned earlier, in the actual implementation of the discrete wavelet transform, QMF's are used instead of the "square box" filters. QMF's are anti-aliasing filters. Two wavelets are used in this thesis — the D_4 orthogonal wavelet and the 7-9 tap biorthogonal wavelet (see Appendix A for their definitions). The frequency responses of the QMF's associated with these wavelets are shown in Figure 2.5 and 2.6 respectively.

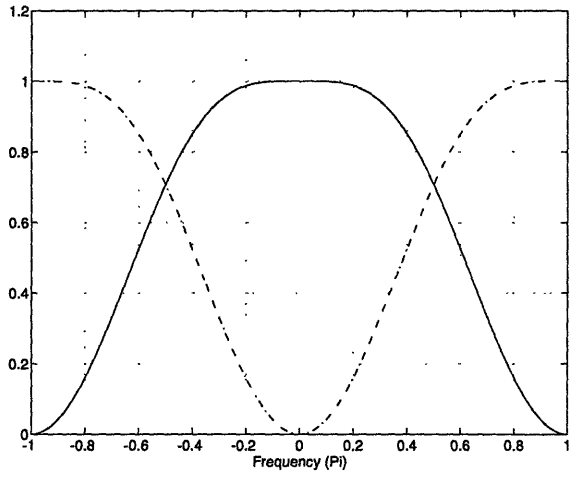

Figure 2.5-a. D_4 analysis filters.

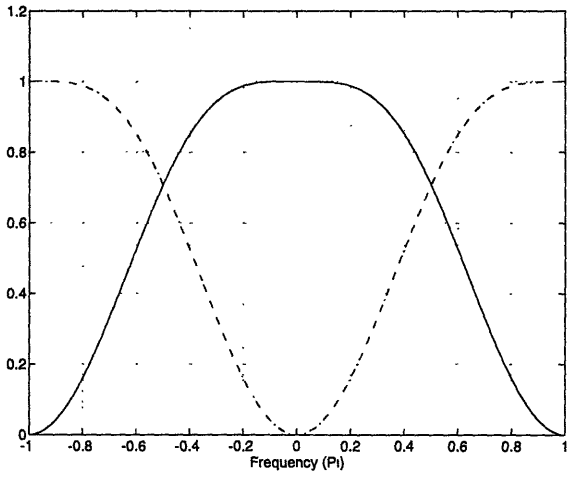

Figure 2.5-b. D_4 synthesis filters.

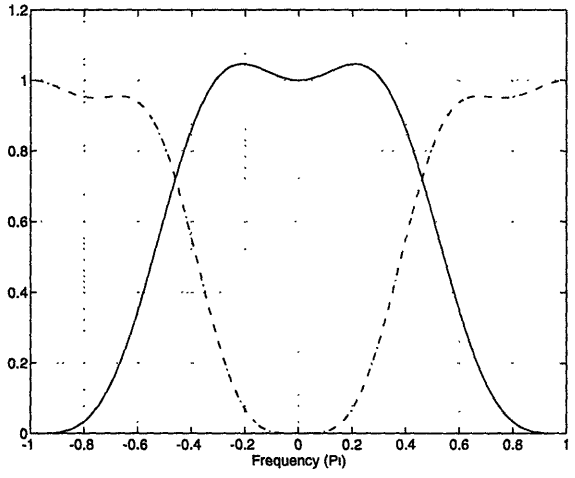

Figure 2.6-a. 7-9 tap analysis filters.

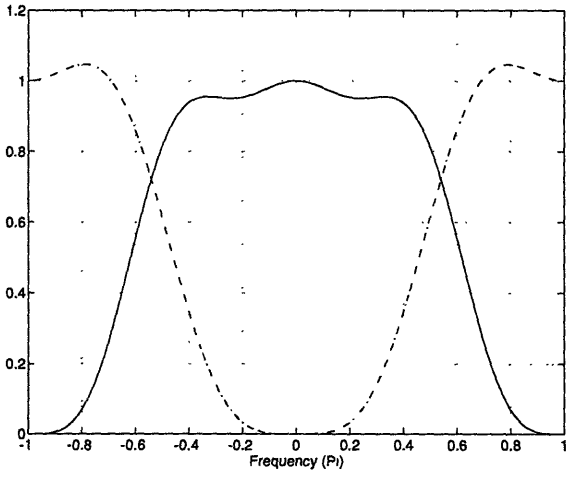

Figure 2.6-b. 7-9 tap synthesis filters.

2.3 Multi-Dimensional DWT

There are several ways of applying wavelet transforms to multi-dimensional signals [33, 12, 29, 47]. The most straightforward method is to carry out the one-dimensional transform for each of the dimensions separately. This approach preserves the $O(n)$ computational complexity.

Mallat [33] has shown that separable two-dimensional wavelets can be used to generate a multiresolution representation of images. Separable two-dimensional DWT's can be efficiently implemented using filter bank structures in Figure 2.7-a and 2.7-b. A separable two-dimensional DWT decomposes an input signal into four subband signals in each transform. With QMF's, these subband signals can be recombined to produce the original signal without aliasing. For image data, the low-pass filtered subband gives a coarse resolution approximation of the original picture and the other three contain spatially oriented information of the picture — namely, the horizontal, vertical and diagonal edges. In other words, the two-stage cascaded one-dimensional filters (in Figure 2.7-a) are performing spatially oriented filtering in the two-dimensional frequency space, as illustrated in Figure 2.8. The frequency response of the separable two-dimensional D_4 wavelet is shown in Figure 2.9 and that of the 7-9 tap wavelet in Figure 2.10.

The x- and y-axis of the plots in Figure 2.9 and 2.10 represent the spatial frequency values (in the horizontal and vertical directions) of the two-dimensional wavelets. The z-axis gives the magnitudes of the discrete filters associated with these wavelets. The filtering operations of the two-dimensional discrete wavelet transforms are clearly identifiable from these plots. For example, to extract the base-band of a image in the wavelet domain, the original image is filtered with a two-dimensional low-pass filter (e.g., the one in Figure 2.9-a or 2.10-a). After the filtering (or wavelet transform), the resulting frequency subband contains mostly the low frequency information of the original image. Likewise, the horizontal, vertical

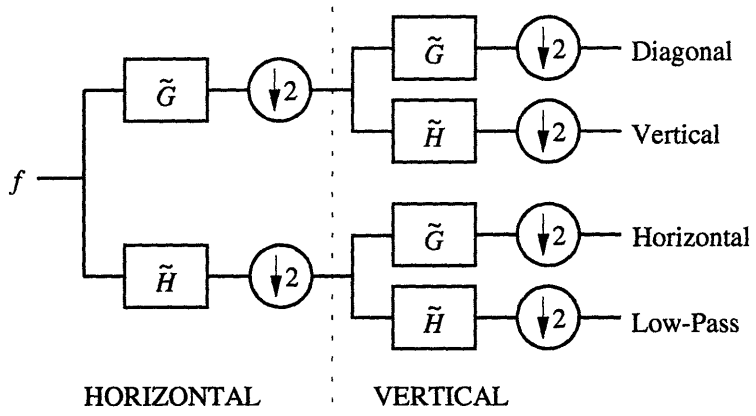

Figure 2.7-a. Two-dimensional orthogonal/biorthogonal wavelet decomposition.

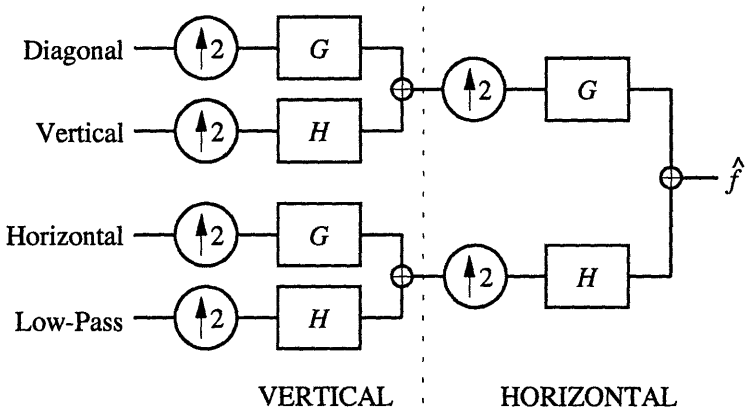

Figure 2.7-b. Two-dimensional orthogonal/biorthogonal wavelet reconstruction.

and diagonal edge bands can be computed by filtering the original image by an appropriate high-pass filter (e.g., the one in Figure 2.9-b, 2.9-c, 2.9-d, 2.10-b, 2.10-c, or 2.10-d).

Digital video signals are three-dimensional signals. Wavelet transforms of digital video can be carried out in at least two different ways, using separable wavelets. The first method treats a video signal as a sequence of two-dimensional images. Each video frame is transformed independently by a separable two-dimensional wavelet as described above. The second method analyzes a video signal using separable three-dimensional wavelets. The

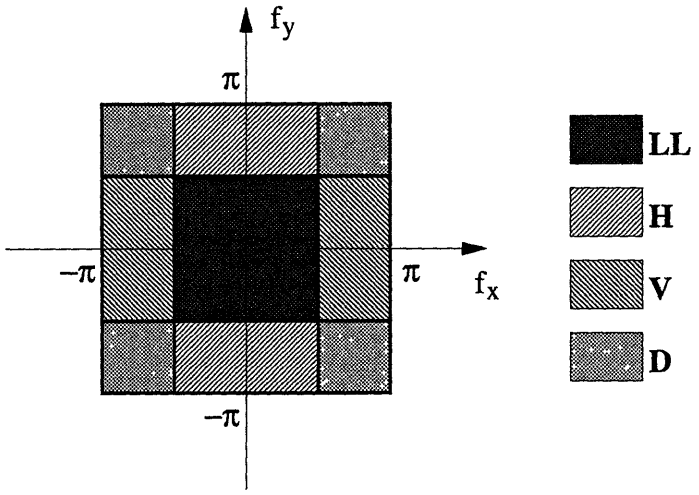

Figure 2.8. Frequency segmentation for a 2-dimensional wavelet transform.

second method is well suited for video compression applications because it can exploit and remove both spatial and temporal redundancy in a video sequence to give very high compression ratios. On the other hand, it introduces inter-frame dependency (i.e., dependency between temporally adjacent frames) in the coded video stream. This implies that the cost of any error will be high because visual degradation due to an error can persist over many frames. Furthermore, error handling for this style of coding is generally complex and difficult since error concealment may have to be applied to several frames for each error occurred. These two factors make separable three-dimensional wavelet transforms less desirable for applications that exchange video information over lossy communications systems. In contrast, the first method limits errors to a single video frame and can provide some amount of redundancy needed for error concealment. This method is chosen for the scalable packet video coding system that is proposed and implemented for this thesis, as described in the following chapter.

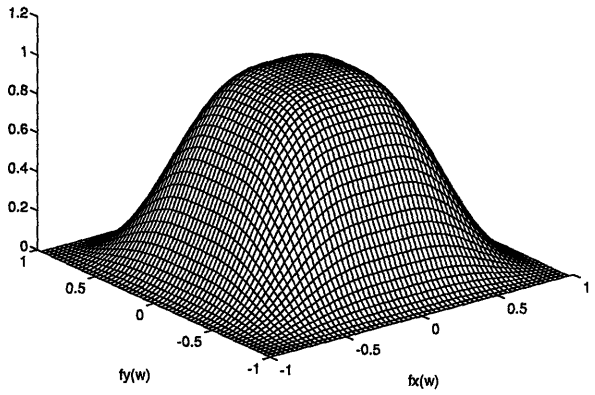

Figure 2.9-a. D_4 base-band filter.

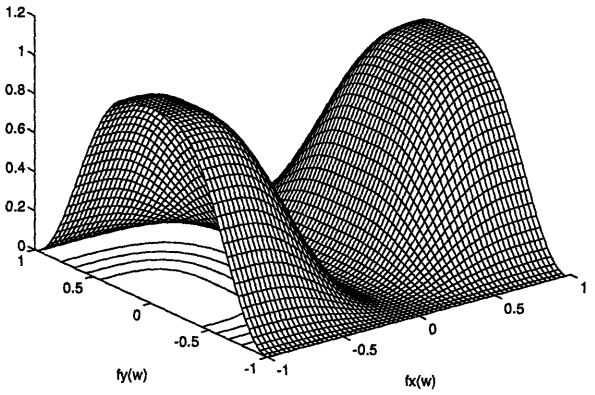

Figure 2.9-b. D_4 vertical subband filter.

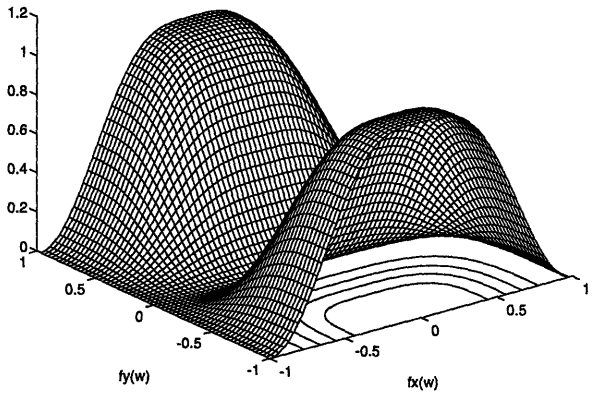

Figure 2.9-c. D_4 horizontal subband filter.

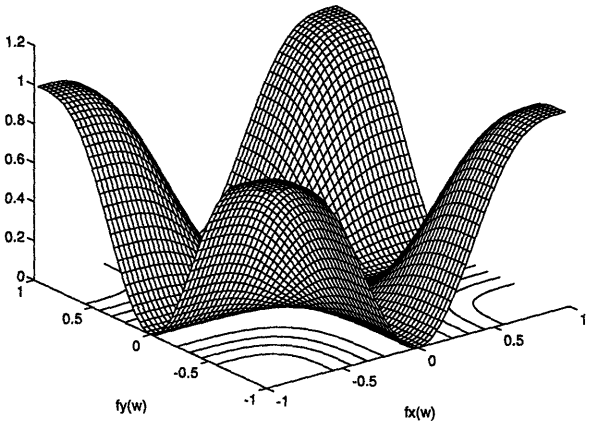

Figure 2.9-d. D_4 diagonal subband filter.

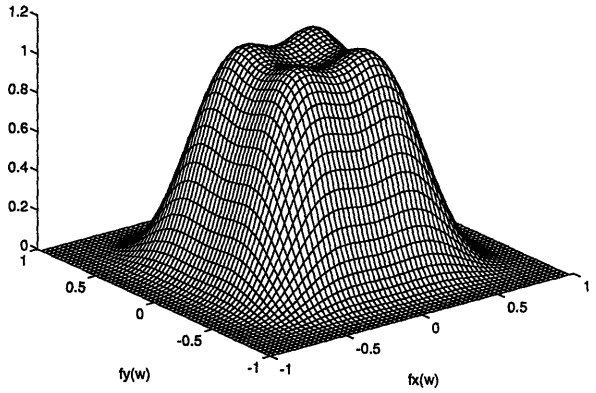

Figure 2.10-a. 7-9 tap base-band filter.

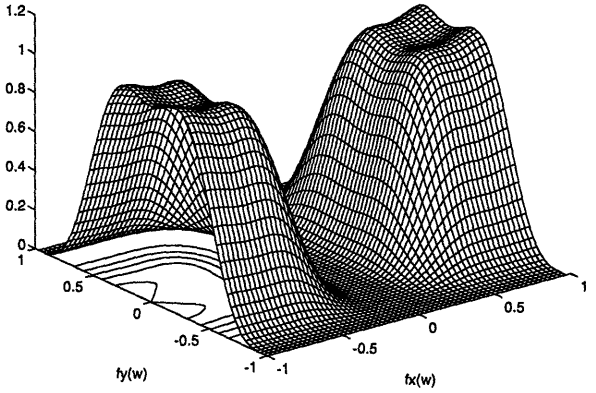

Figure 2.10-b. 7-9 tap vertical subband filter.

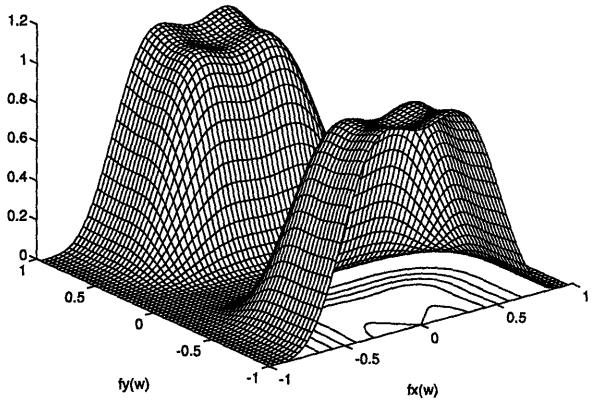

Figure 2.10-c. 7-9 tap horizontal subband filter.

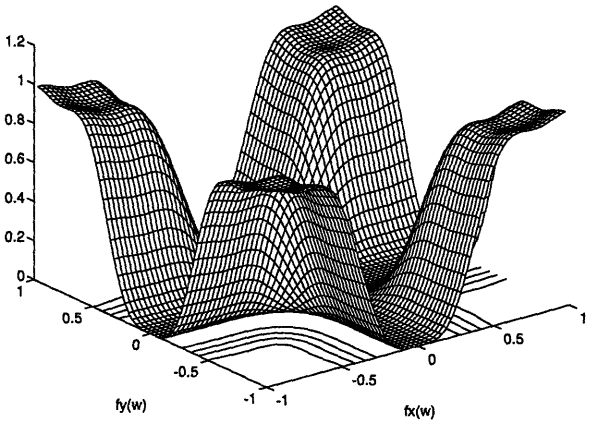

Figure 2.10-d. 7-9 tap diagonal subband filter.

Chapter 3

Scalable Coding for Packet Video

A well-designed packet video system will have at least three system components — a video encoder, a video decoder and a pair of network transport agents who are responsible for doing flow control. The video encoder proposed and implemented in this thesis is *scalable* in terms of the coded video quality and bandwidth usage. The focus of this chapter is on the design and implementation of a scalable video codec system. The networking aspects of the adaptive packet video system will be addressed in Chapter 4.

3.1 Design Objectives

To facilitate network adaptation, a packet video encoder should be able to code a video stream at different bit-rates in response to the current traffic level in the network. If the network is approaching a congested state, ideally all the non-critical video sources[3.1] that are contributing to the congestion should reduce their output bit-rates. When the congestion period is over, these video sources can increase their output back to the original levels. The output rates of the encoders should be adjusted gradually to avoid abrupt changes or large swings in the perceptual quality of a coded video stream. At the same time, they should be changed in such a way to

---

3.1 Here, *non-critical video sources* refer to video sources that can tolerate temporary quality degradation. An example of a non-critical video source is a video-conferencing stream.

maintain stability of the traffic flows. A scalable coding system can achieve this fine-granular control of video output rates.

3.2 Video Encoder

The scalable video encoder proposed in this thesis is an intra-frame coder with two main stages (Figure 3.1). The first stage performs the 2-dimensional discrete wavelet transform, using the 7-9 tap wavelet. Each video frame is transformed into four spatially oriented subbands (Figure 2.9-a); the low-pass filtered frequency subband (or the base-band) is decomposed two more times by applying the wavelet transform recursively to the base-band signal. This generates a total of ten subbands. These subbands are individually quantized in the second stage. The base-band and the lowest horizontal and vertical subbands are scalar quantized and the other seven subbands are vector quantized. Scalability in coded video is achieved by encoding an input video source at multiple resolutions as enabled by the wavelet transform and at multiple bit-rates as supported by the multiresolution vector quantization.

3.2.1 Wavelet Transform

The successive wavelet decomposition process is illustrated in Figure 3.2. The purpose of the wavelet transform is to map each video frame into a domain that is close to the one in which the

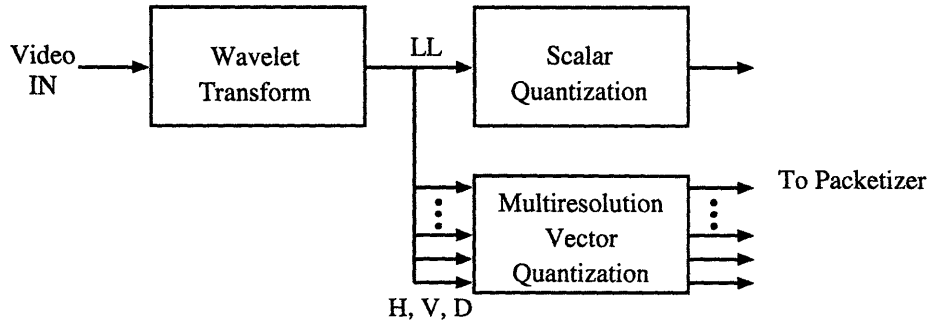

Figure 3.1. Packet video encoder.

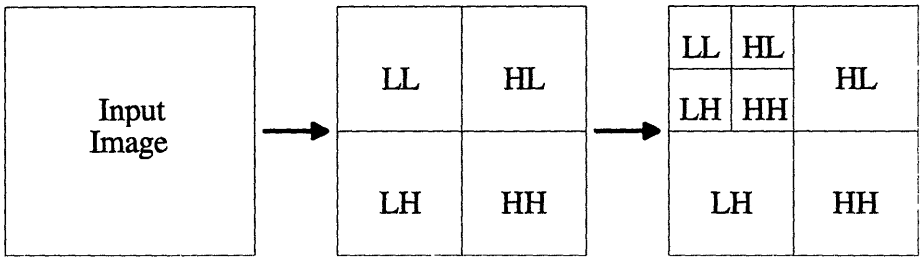

Figure 3.2. Wavelet decomposition of a video frame.

human visual system operates [34]. This mapping enables video frames to be processed or coded in ways that are best suited for human viewers. For the encoder designed in this thesis, each video frame is decorrelated into ten subbands — a base-band containing a very low resolution approximation of the original and nine subbands containing the edge information at three different resolutions.

The base-band signal is very important because it holds most of the intensity information of an image. The horizontal, vertical and diagonal subbands add edge details to the otherwise blurred base-band image when it is scaled to the next (higher) resolution level.

Together, the subbands generated by the recursive wavelet transforms form a hierarchical multiresolution representation of each input video frame. This implies that each video frame can be coded at very different bit-rates to give different play-back qualities by discarding certain frequency subbands. For example, a simplistic way of reducing bit-rate is to encode only the base-band at a certain resolution level. The effect of this method is illustrated in Figure 3.3. Here, a sample image is represented by its base-band transform coefficients at resolution levels 1, 2 and 3 and later restored to its original size. It is evident from the figure that the quality of the coded images differ significantly from one resolution level to the next. At a lesser extreme, only one subband is removed at a time. The resulting changes in picture quality are less drastic than those in the previous case. Nevertheless, the changes can still be quite noticeable. If instead, each subband is encoded at multiple resolutions, much subtle

adjustments (of output bit-rate and therefore of image quality) can be made. Such fine granular changes can be achieved by quantizing the subbands using multiresolution vector quantizers.

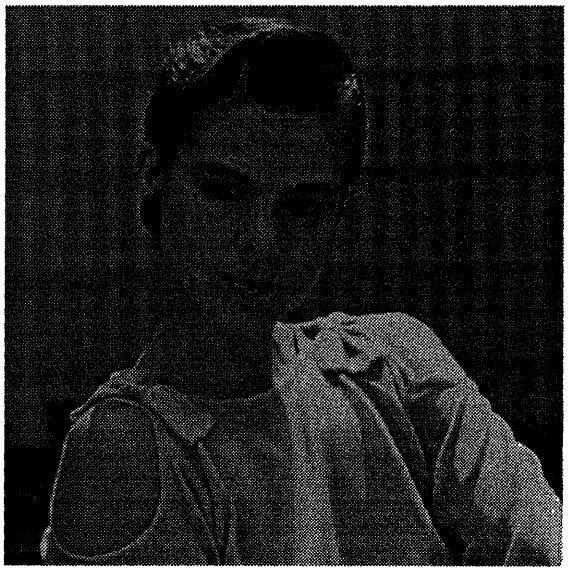

Figure 3.3-a. Image reconstructed from all subbands.

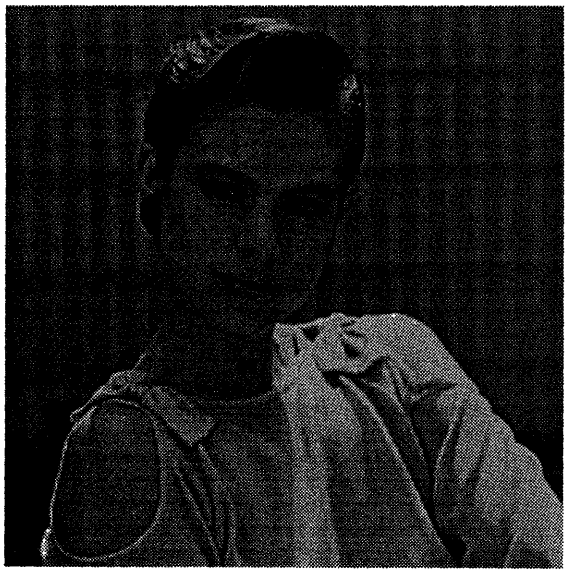

Figure 3.3-b. Image reconstructed from level 1 baseband signal.

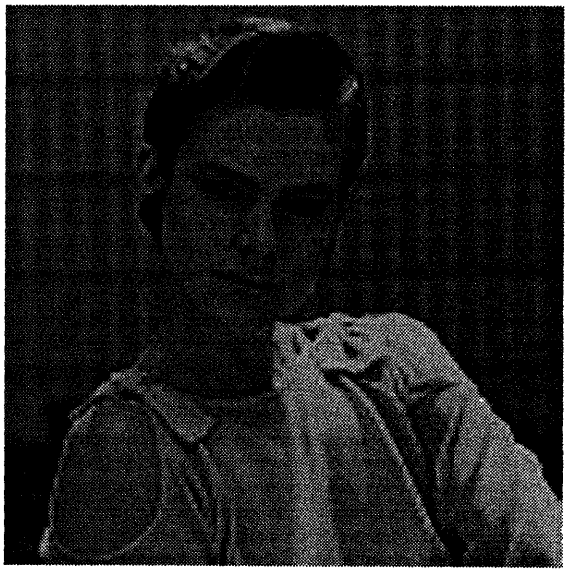

Figure 3.3-c. Image reconstructed from level 2 baseband signal.

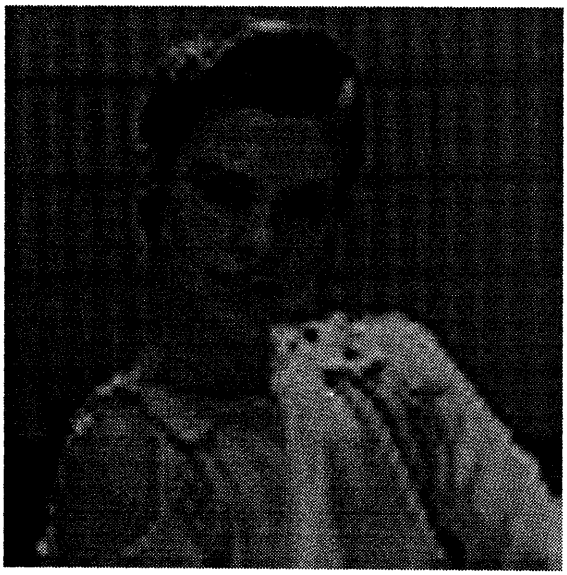

Figure 3.3-d. Image reconstructed from level 3 baseband signal.

3.2.2 Quantization

The wavelet transformed frequency subbands can be coded at multiple resolutions by utilizing multiresolution quantizers [44, 45]. In the wavelet domain, the base-band signal is extremely important because it contains the intensity information of an input image. Quantization noises that occur in the base-band may be amplified during the image reconstruction. Consequently, sufficiently high bit-rate should be allocated to this subband to ensure that average quantization error is kept low. The video encoder implemented here uses a 7 bits/pixel exponential scalar quantizer to encode the base-band signal. To optimize for speed, the quantization table is pre-computed and the quantization process is implemented as a single table look-up.

The horizontal, vertical and diagonal subbands hold the edge information of an input image. Most of the coefficient values in these subbands concentrate around zero, except those near large discontinuities in intensity. Antonini *et al.* [1, 2, 3] have shown that vector quantization provides a very efficient way of encoding the subband coefficients in the wavelet domain.

Vector quantization exploits statistical correlation between all the dimensions of a vector (or image block) to achieve a bit-rate of 1 bit/pixel or less [22]. In vector quantization, all input vectors are mapped onto a set of *representative* vectors. This set of vectors forms the *code-book* for the vector quantization. The vector space in which all vectors are mapped to a single representative vector is called a *cell.* Many schemes have been devised to partition a given vector space into non-overlapping cells and to compute the representative vectors. Most of these algorithms use iterative techniques to converge to the minimum distortion code-book of a given size and have $O(n^2)$ computational complexity. The first and the most commonly used convergence scheme is the Linde, Buzo and Gray algorithm [32]. The LBG algorithm is computationally intensive. Also, it does not bound convergence time or even guarantee that it

will converge at all. For these reasons, other methods have been developed that do not use this iterative convergence approach to construct the code-book and have much lower computational costs. For example, the lattice vector quantization has $O(1)$ computational complexity. Buda [6] has shown that some lattice codes can achieve the performance of the optimal codes proposed by Shannon. Conway and Sloane [14, 15] have determined the best known lattices for several vector dimensions as well as fast quantizing and decoding algorithms for these lattices. Furthermore, Antonini *et al.* [2] have demonstrated that the wavelet coefficients of a transformed image can be coded at very low bit-rate using lattice quantization.

In lattice vector quantization, a lattice is formed by the set of all vectors spanning an m dimensional space. That is, it is defined as the set of vectors

$$\mathbf{x} = u_1\mathbf{a}_1 + \cdots + u_n\mathbf{a}_n$$

where $\mathbf{a}_1, \ldots, \mathbf{a}_n$ are linearly independent vectors in m-dimensional real Euclidean space with $m \geq n$, and $u_1, \ldots, u_n$ are integers. The representative vectors are selected from the center points of sets of polytype cells which make up the uniform lattice structure. The performance of a lattice quantizer is largely determined by the shape of the cells. In this thesis, an orthogonal lattice quantizer is selected for the scalable video codec. Although orthogonal lattice quantizers generally do not give the best performance in the mean-squared error sense, they are comparable in visual performance to other types of lattice quantizers [43]. The quantization algorithms for these quantizers, however, are the fastest of all lattice quantizers.

Because a regular lattice quantizer has an infinite number of representative vectors, the lattice needs to be truncated in practice. This can be achieved by either scaling the input vectors or by selecting the set of representative vectors for cells that are most densely populated. The latter approach is chosen because it exploits the statistics of the input. In

addition, it is important to select a method that has a good balance between performance and computational complexity. Tree-based algorithms are the prime candidates based on the above criteria. Tree-based algorithms can generate a code-book of given size n in $n \log n$ time. It can also encode input vectors in $n \log n$ time.

Here, the tree-based vector quantization algorithm selected for the scalable video encoder is the Kd-Tree introduced by Bentley *et al.* [21]. A Kd-Tree organizes vector space partitions in a binary search tree to allow efficient vector search and encoding, at the expense of certain constraints placed on the subdivision process [40]. Each node in the Kd-Tree has a binary decision operation associated with it. During a search, the tree is traversed based on the outcome of the binary decision. For a Kd-Tree vector quantizer, each node represents a single partition of the vector space. The constraint placed on the Kd-Trees is that only one vector dimension can be split at each node. That is, each node cuts the k dimensional vector region of interest into two partitions with a $k-1$ dimensional hyperplane. The dimension which is subdivided at each node maybe different. Two child nodes are created as a result of the split, each representing "half" of the vector space being partitioned. To build a Kd-Tree vector quantizer, one starts with populating the root of the tree with the entire vector space. This multi-dimensional space is then continuously subdivided by splitting the leaf with the highest distortion along some dimension until a certain bound condition is met. The split hyperplanes are orthogonal to the axis being split because of the constraint of Kd-Trees. An example of a Kd-Tree with $k = 2$ is depicted in Figure 3.4. Methods of selecting the split dimension and split point represent some degrees of freedom in the design that are useful for tuning the quantizer for a specific application. However, the distortion criterion used for leaf selection has the most impact on the performance of the quantizer.

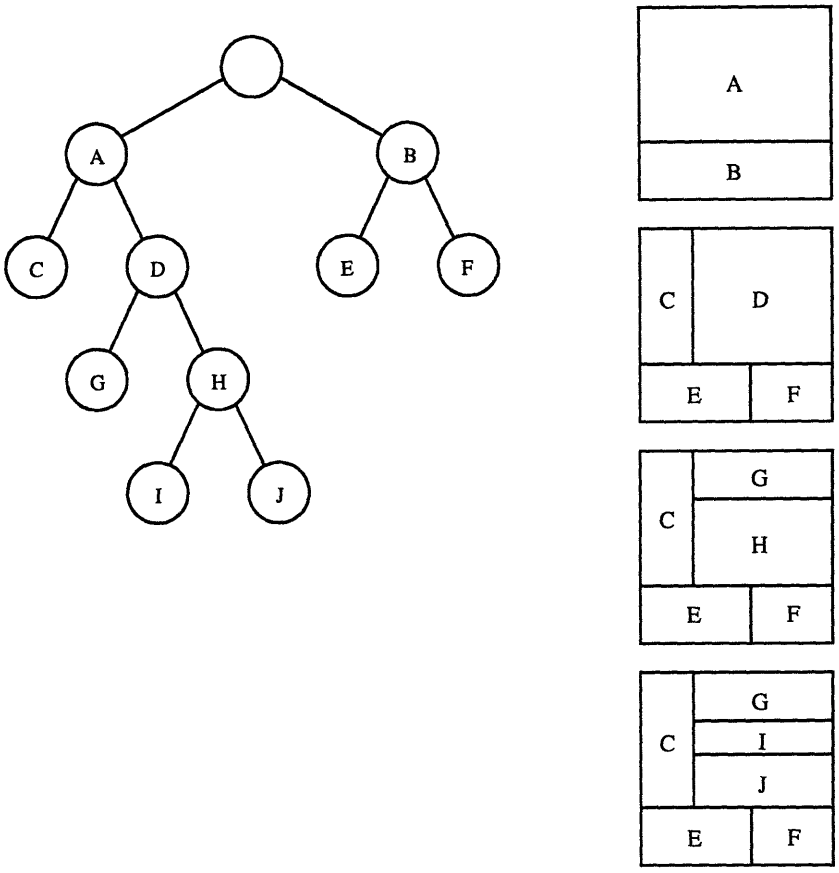

Figure 3.4. A typical Kd-Tree for 2-dimensional vector space.

A Kd-Tree vector quantizer is defined by a set of parameters that dictates the tree splitting process. These parameters are:

- **Node/Dimension Selection:** This parameter controls how the leaves are chosen for subdivision. Candidates for this include selecting the leaf (and the dimension within the leaf) based on the variance, cell extent, mean-squared error (with respect to the representative), absolute error (with respect to the representative) or maximum number of constituents. A distortion function is devised using one of these metrics. It is constrained to return a monotonically increasing number in proportion to the amount of distortion incurred in each cell. It is also required to return a 0 value when the subdivision process should be terminated.

- **Split Point Selection:** This parameter determines at what value is the chosen split axis to be partitioned. Candidates for this are the mean, median, mode, or mid-point of the split dimension.

- **Representative Calculation:** This parameter describes the method for calculating the representative vector for a cell. Candidates for this are the centroid, the mid-point of the cell or the constituent vector with the most occurrences in the cell.

- **Bounding Condition:** The parameter sets the criterion for terminating the cell subdivision process. Candidates for this include the code-book size, total distortion or peak distortion.

Starting with a single node (i.e., the root) whose constituency is the entire vector space, a Kd-Tree can be constructed according to the following steps:

1. Find the leaf with the maximum distortion measure.

2. Make the leaf an intermediate node and create two new leaves. Assign all vectors in the intermediate node to either the left or the right child, depending on whether the vector is less than or equal to or greater than the split point in the split dimension.

3. Calculate the representative vectors, distortion values (in each dimension) and split points for the two new leaves.

4. Repeat step 1-3 until the bound condition is met.

The control parameters for the particular Kd-Tree vector quantizer used in this thesis are the following:

- The distortion measure is given by the mean-squared error in each dimension.
- The split point is chosen to be the mid-point of a cell along the split dimension.
- The representative vector is computed as the centroid of a cell.
- The bound condition is determined by the desired code-book size.

To build a Kd-Tree based lattice vector quantizer, a set of training vectors is first quantized using a regular (i.e., unbounded) lattice structure. The resulting vectors are then used to construct the Kd-Tree for a given code-book size (according to the procedure described above). Taken together, these two steps generate a static lattice quantizer with the specified bit-rate. Since static quantizers use fixed code-books, they have to compute the code-books only once. They do not require the periodic code-book updates that dynamic quantizers do and therefore do not need to transmit and synchronize the code-books between the sender-receiver pairs/groups.

Furthermore, the same Kd-Tree is used to add the multiresolution aspect of the vector quantizer. That is, the same Kd-Tree structure is employed to represent multiple code-books of different sizes. Given that the full (bit-rate) Kd-Tree has been constructed, a subtree can be selected to produce a quantizer for a lower rate. Likewise, multiple quantizers can be represented by the same Kd-Tree structure. If the lower bit-rate quantizers are strictly made of the leaves of the full bit-rate Kd-Tree, the corresponding code-books must be subsets of the full-size code-book. If the quantizers include both the leaves and the intermediate nodes, different code-books must be constructed. The former method consumes less system resources because only one code-book needs to be maintained; the latter gives better performance because

the optimal subtree is selected. Therefore, depending on the requirement of the coding system being designed, trade-offs can be made between quantizer performance and complexity.

For the scalable video encoder implemented in this thesis, the first method is chosen to reduce the software complexity. The optimal bit allocation for each vector quantizer is computed statistically. The overall bit allocation scheme is given in Figure 3.5 and the quantizer performance in Figure 3.6 (using the test image shown in Figure 3.7 and the training images in Figure 3.8). Note that zero bits are allocated for the highest-frequency diagonal subband because the image information it contains is negligible. The wavelet transform coefficients in this subband are either 0's or very close to 0's.

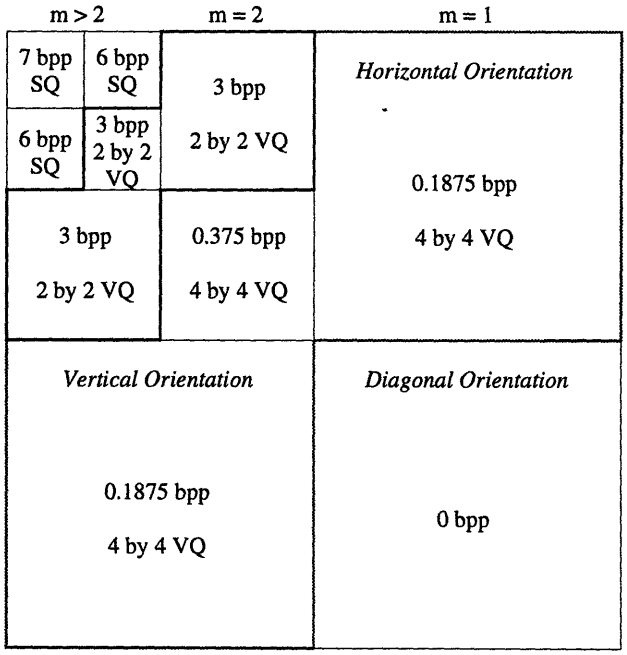

Figure 3.5. Wavelet subband bit allocation scheme.

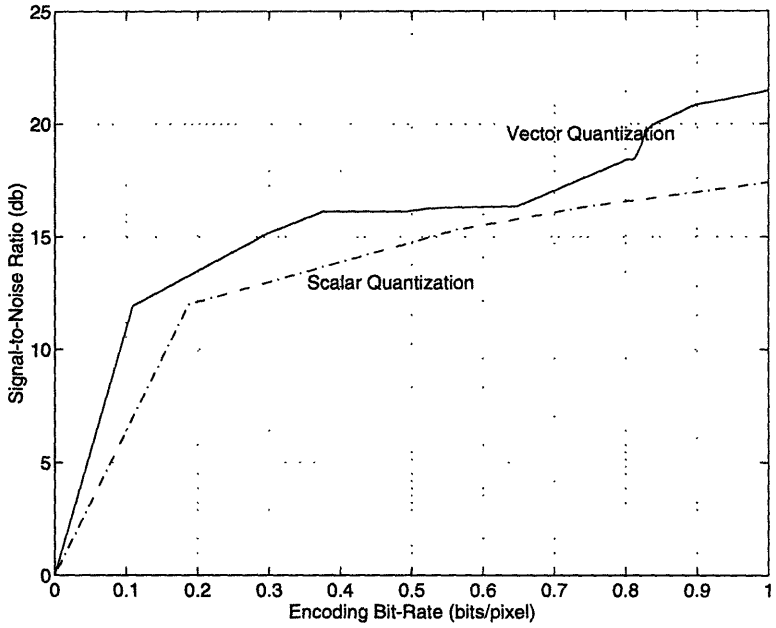

Figure 3.6. Static bit-allocation for the vector quantizer.

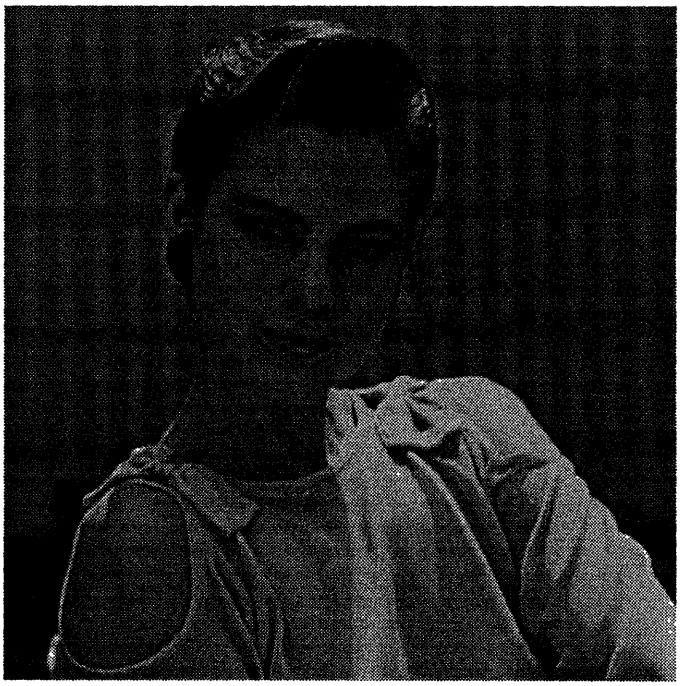

Figure 3.7. Test image used to determine the bit allocation scheme for the vector quantizer.

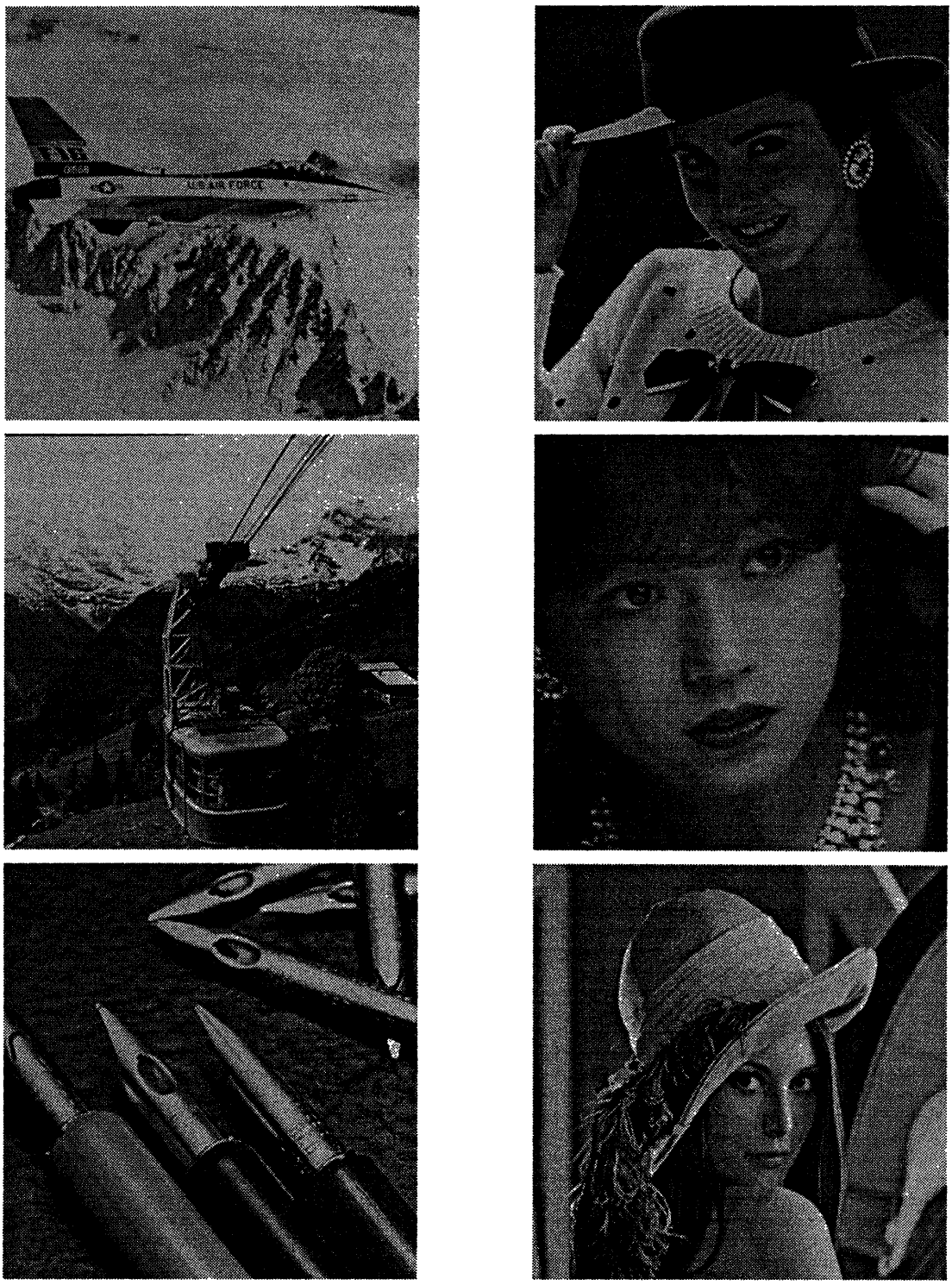

Figure 3.8. Images used to train the Kd-Tree based orthogonal lattice quantizer.

3.3 Video Decoder

The video decoder design follows immediately that of the encoder. It also consists of two main stages (Figure 3.9). The first stage inversely quantizes the ten subbands. Since the quantizer code-books are statistically computed, inverse quantization can be efficiently implemented as a table look-up operation. The second stage performs the 2-dimensional inverse wavelet transform. Currently, the resolution of the decoded video is determined at the time of encoding. In a more sophisticated system, an extra processing stage can be added before the inverse wavelet transform stage to perform the resolution conversion.

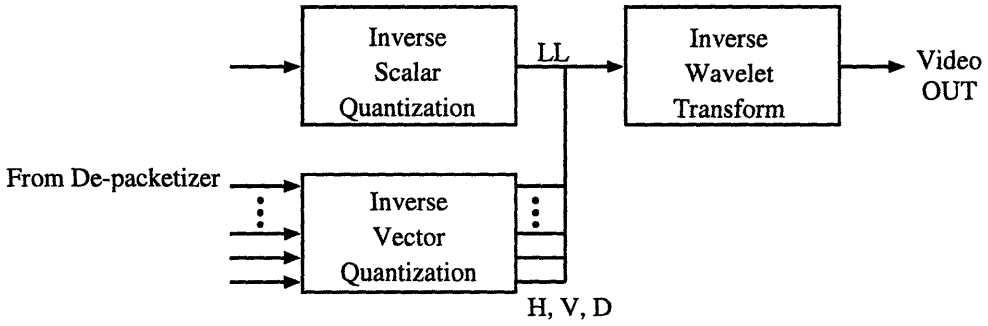

Figure 3.9. Packet video decoder.

A more sophisticated decoder for packet video will also include a mechanism that performs error concealment. Error concealment is carried out in the wavelet domain before the inverse wavelet transform. It has been suggested [27, 28] that, if an error occurs in a high frequency subband due to a packet loss, the *zero-substitution* technique can be a simple but effective way of reducing the perceptual impact of the packet loss. However, our experience indicates that the zero-substitution technique may cause abrupt changes in the image quality of a decoded video sequence when packet losses are random and spread over several consecutive frames. This phenomenon is particularly noticeable when packet losses persist over a long

period of time and/or when the loss rate is moderate. The decoder built in this thesis uses a simple predictive error concealment technique in the temporal dimension. Namely, the missing data in a subband is substituted by the data from the corresponding region of that subband available from the closest frame. This substitution technique does not require any computation. When general processors become fast enough, more exotic interpolation methods (such as the iterative spatial interpolation method proposed in [53]) can be used to achieve better performance in error concealment.

Chapter 4

End-to-End Flow Control for Packet Video

In the absence of explicit transport service classes for real-time traffic from the network, end-to-end flow control of video sources can have significant impacts on the overall performance of the network. Without flow control, network delay and packet loss rate will continue to increase as the network becomes congested. Furthermore, non-flow-controlled traffic can cause large throughput reduction in flow controlled traffic under overload conditions. When taking into account of the bandwidth requirement and duration of typical video sessions, it becomes evident why flow controlling packet video sources is critical. End-to-end flow control mechanisms are investigated in this thesis. An end-to-end flow control mechanism has the advantage that it does not require any modification to the intermediate switches in the network.

4.1 Assumptions

The flow control mechanism proposed here imposes very few requirements on the underlying transport network and the video source. In addition, all these requirements have been or can be met with the existing network and computer technologies.

The transport network is assumed to be packet based. It supports both flow controlled sessions (e.g., TCP) and unicast/multicast datagrams (e.g., UDP). Switches in the network use simple first-in first-out (FIFO) scheduling of outbound packets.

The video source must support scalable video coding. It is required to generate video traffic approximately equal to a specified bit-rate at a frame boundary. The output bit-rate is determined by the flow control mechanism and provided before each frame is encoded.

4.2 End-to-End Overload Control Alternatives

Overload control mechanisms for packet video traffic can be implemented in the end-nodes, or the intermediate switches or both. End-to-end control (if effective) is highly desirable because it involves modifying only the end-nodes. There are many proposed alternatives to implementing overload controls. These [26] include:

1. Receiver-only concealment

   The receiver mitigates the effect of lost packets by exploiting spatial or temporal correlation between available data and missing data. An interpolation method is applied accordingly to recover some information.

2. Forward error recovery

   The video packets sent contain forward error correction codes. While this technique can be used to recover bit errors exactly, the recovery is limited to within the same packet. Obviously, it is useless when an entire packet is dropped. This approach is not considered in this thesis.

3. Prioritized transmissions

   If the network provides multiple transmission priorities, packets can be prioritized by assigning those with essential visual information (e.g., the base-band signal in the wavelet domain) higher priorities than those with enhancement information (e.g.,

high-frequency edge subbands). This approach is not considered either because prioritized transmission is not readily available in most existing networks.

4. Multi-layered coding

   Multi-layered coding describes a class of algorithms that represent a signal as a coarse resolution approximation of the original and a set of detailed signals at various resolutions. Wavelet coding is an example of a multi-layered coding method.

5. Traffic shaping at sender

   By smoothing out the bursty output of a video stream over a time interval, the occurrence of a temporary congestion period can be reduced moderately. However, the buffering of packets at the sender's network output queue increases total end-to-end delay as seen by packets. This approach is not considered here.

6. Feedback flow control

   By analyzing the feedback information from the receiver or the network, the sender estimates the available network bandwidth and its share of it. It then adapts to the changing bandwidth by modulating the output bit-rate of the video source.

Some of the above approaches are orthogonal. One could design a hybrid scheme that uses more than one of these mechanisms. In fact, the overall overload control for the adaptive packet video system proposed and implemented in this thesis utilizes mechanisms 1, 4 and 6. The designs of the first two mechanisms have been given in Chapter 3, and the design of the third one will be described in this chapter.

4.3 Design Objectives

The goal of a flow control mechanism for real-time packet video is to optimize the perceptual quality of the video delivered over the network in the presence of packet losses. Packets are dropped at a switch due to congestion or at the receiver due to late arrival. Packet losses will affect the perceptual quality of the transmitted video in several different ways. The obvious one is degradation in image quality. Packet losses can also lead to annoying visual artifacts such as aliasing or oscillation in "image focus." In general, the perceptual quality of a received video stream is largely affected by such factors as which packets are dropped, what information is contained in these packets, and how close together these packet drops are relative to each other.

From the above discussion, it is clear that flow control mechanisms designed to maximize average throughput or end-to-end delay do not necessarily produce optimal visual quality for a video session. The goals of feedback control for video traffic are very different from those for data traffic. In the case of data transfer, the main objective is to maximize average throughput (or minimize average delay) while preserving the integrity of the data stream. Packet retransmission is used to recover from transmission errors. In the case of video communications, the main objective is to maximize the perceptual quality of the received video. In packet-switching networks, most packet losses are caused by network congestion and can be minimized by an adaptive flow control that reduces output traffic of a packet video source during a congestion period. Consequently, an adaptive packet video stream will suffer less packet losses than a non-adaptive stream and will produce better play-back quality. Furthermore, by removing information according to a loss-preference model derived from the human visual system, a video encoder can generate an optimally coded video stream at a lower bit-rate. As a result, the overall perceptual quality of an adaptive packet video stream will be much higher than that of a non-adaptive stream.

Selective-dropping of information in a video sequence is fairly well understood from a coding stand-point. The quantization step in a video coding algorithm uses the same kind of bit allocation process to optimize the image quality for a given bit-rate. A major challenge of a flow control for an adaptive packet video system is to estimate the series of target rates such that packet drop rates both in the network and at the receiving node are minimized. In addition, it must do this without driving the source to its lowest output rate and without causing oscillation in image quality.

Another important objective for such a flow control mechanism is to provide fair sharing of bandwidth and other network resources among different flow controlled video and data streams. Here, traffic flows are considered as achieving fair sharing if all the flows competing for bandwidth will generate output bit-rates roughly in the same proportion to their maximum output rates over time. That is, fair sharing is measured in the asymptotic sense, rather than any transient sense. Flow controlled data traffic such as TCP traffic behaves extremely well under heavy load, by dynamically reducing its throughput rate. When such traffic is mixed with non-flow controlled traffic such as UDP traffic, it tends to be "squeezed out" when the network becomes congested. This leads to rather poor sharing of the bandwidth. By using a flow control mechanism that behaves similarly to the TCP's congestion control and avoidance mechanism, adaptive packet video sessions can better share network resources with other flow controlled sessions.

In summary, an effective flow control mechanism for packet video must satisfy at least the following criteria under overload conditions:

- Maximize the perceptual quality of the video delivered by minimizing packet drop rates both inside the network and at the receiving nodes; and

- Provide fair sharing of bandwidth and other network resources with other flow controlled video and data traffic.

4.4 Proposed Flow Control Mechanism and Algorithms

The design of the proposed packet video flow control mechanism is closely guided by the design of the BSD TCP's congestion control and avoidance mechanism [24]. In [24], Jacobson introduces the "packet conservation" principle:

> *By 'conservation of packets' we mean that for a connection 'in equilibrium', i.e., running stably with a full window of data in transit, the packet flow is what a physicist would call 'conservative': A new packet isn't put into the network until an old packet leaves.*

To avoid persistent congestion that could lead to total network collapse, a "congestion window" is used in TCP to regulate the amount of data packets in transit. The size of the TCP congestion window is changed dynamically in response to packet losses as triggered by time-outs. In particular, the congestion window size is dropped to 1 packet whenever there is a packet loss (triggered by a sender time-out). Upon the receipt of each positive acknowledgment, the congestion window is increased according to the slow-start algorithm. Specifically, it is opened up exponentially until its size exceeds the slow-start threshold, after which it is expanded linearly.

Similar to the TCP design, the proposed flow control for packet video employs a window mechanism and adaptively adjusts the window size based on estimated delay parameters. However, it obeys the "packet conservation" principle in only a very loose manner. In TCP, conservation of packet is critical to the stability of the network dynamics. It makes congestion collapse "the exception rather than the rule." The reason for needing such a strict

conservation principle is due to the positive feedback used to recover from transmission errors. That is, if a packet is lost or damaged during transmission, a copy of it will be sent repeatedly until it is received correctly. Because congestion causes packet dropping, it triggers data retransmission. Retransmission adds additional traffic to the already congested network and leads to further packet dropping. This creates a "vicious cycle" that may eventually cause the whole network to collapse. The "packet conservation" principle is devised precisely to avoid this cycle. Since retransmission is not used in packet video transmission, there is no positive feedback loop. If the correct flow control is not used to regulate the output of packet video sources, throughput or delay of the network can deteriorate as the offered network traffic increases. Nevertheless, as long as any single packet video source does not overwhelm the capacity of the bottleneck link or switch, total network collapse will not occur. Consequently, it is not necessary to follow the "packet conservation" principle strictly.

The proposed flow control mechanism contains a set of algorithms, most of which can also be found in the TCP flow control and its derivatives:

1. end-to-end and round-trip delay estimation,
2. exponential window reduction on congestion,
3. slow-start,
4. rate stabilizer.

Each of these algorithms will be treated in one of the following sub-sections. First, several system parameters need to be defined.

4.4.1 Important System Parameters

In order to facilitate the flow control mechanism to adjust the output rate of a packet video source in such a way that also meets the requirements of a particular coding system, the proposed flow control mechanism requires the encoder to supply three parameters:

- R_{MAX} - maximum encoding output rate (in bits/second),
- R_{MIN} - minimum encoding output rate (in bits/second),
- Δ_R - minimum change for each rate adjustment (in bits/second).

In return, the flow control mechanism provides the encoder with a single parameter B that specifies the upper-limit of the encoding bit-rate for the next video frame. The encoder is expected to code the next frame at a bit-rate of no more than B bits.

4.4.2 End-to-End and Round-Trip Delay Estimates

The average end-to-end and round-trip delays and delay variances are estimated based on the feedback information. The sender expects to receive periodic updates on packet delays and losses from the receiver. Currently, the receiver sends this information once per video frame, in form of acknowledgment packets. An acknowledgment packet for the current awaited frame is sent upon the reception of a packet that belongs to a subsequent frame. Information contained in the acknowledgment packet includes:

- SEQ_{ACK} - the frame number it is acknowledging,
- T_F - the average end-to-end delay of packets for that frame,
- P_D - whether any packet in that frame is dropped,
- M - the number of missing frames (if any) between the current awaited frame and the most recently received frame.

In addition, the sender computes the end-to-end delay in the reverse direction T_R from the time-stamp in the acknowledgment packet.

As in the TCP protocol specification, the proposed flow control mechanism estimates the mean round-trip time and its deviation using a low-pass filter

$$T_{RTT} = (1-\alpha)T_{RTT} + \alpha(T_F + T_R)$$

$$D_{RTT} = (1-\alpha)D_{RTT} + \alpha\left|(T_F + T_R) - T_{RTT}\right|$$

where T_{RTT} is the average round-trip time, D_{RTT} is the mean deviation for round-trip time, and α is a filter gain with a value of 0.125 as suggested in [24]. In addition, it estimates the mean end-to-end delay and its deviation as

$$T_{ETE} = (1-\beta)T_{ETE} + \beta T_F$$

$$D_{ETE} = (1-\beta)D_{ETE} + \beta\left|T_F - T_{ETE}\right|$$

where T_{ETE} is the average end-to-end delay, D_{ETE} is the mean deviation for end-to-end delay, and β is a filter gain also with a value of 0.125.

Currently, both round-trip and end-to-end delays are computed from the time-stamp attached to each packet. This means that, in order to have meaningful end-to-end delay measurement, the end-nodes participating in a video session need to be time-synchronized. Unfortunately, time-synchronizing end-nodes is not always feasible. For systems that do not support time synchronization, a packet video source can estimate the end-to-end delay as

$$D_{ETE} = D_{RTT}/2$$

This causes the packet video source to under-estimate the amount of available bandwidth to it. Therefore, this is a "safe" alternative in the sense that it will not lead to worse performance than the one using end-to-end delay used for time-synchronized systems.

4.4.3 Window Adjustment on Congestion

The proposed flow control mechanism uses a *window* to limit the amount of data in transit that comes from a particular packet video session. So the two key parameters to the flow control algorithms are the window size and the amount of data currently in transit. The latter parameter can be derived from the number of video frames current traveling in the forward direction. It is estimated from the average end-to-end delay and the frame-rate of the video F as

$$N_{ETE} = T_{ETE} \times F$$

In addition, it is also useful to estimate the number of frames sent in one round-trip time as

$$N_{RT} = (T_{RTT} + 2D_{RTT}) \times F$$

Under the proposed flow control mechanism, network congestion is indicated by the reception of an acknowledgment packet whose round-trip time is longer than or equal to $T_{RTT} + 2D_{RTT}$. This is analogous to the way that the TCP protocol predicts whether the network is congested. Upon the detection of congestion, the flow control window size is multiplicatively decreased (or equivalently, exponentially decreased if the congestion persists)

$$W_i = \rho\, W_{i-1}$$

where ρ (< 1) is set to 0.8. To ensure that the output rate of a video stream does not drop to zero, the window should be at least $R_{MIN} \cdot N_{ETE}$ in size. That is, the window should be adjusted according to

$$W_i = \max(\rho\, W_{i-1}, R_{MIN} N_{ETE})$$

It is important to note the reason for choosing ρ to be 0.8 instead of a much smaller value. In TCP, the congestion window is set to 1 after a time-out. This drastic reduction in throughput rate is necessary to prevent positive feedback from building up and thus to maintain the stability of the network. This is also possible given the fact that data traffic is very elastic

(i.e., its throughput can take on any value). For video traffic, the dynamic range of acceptable throughput rates is much smaller and the instantaneous change in throughput rate must be constrained to avoid large shifts in perceptual quality of the packet video. Since the proposed flow control mechanism does not create any positive feedback loop, it can afford to converge to the right output level for a packet video session slowly. Although assigning ρ to be 0.8 seems somewhat arbitrary, it has been shown to give a good balance between improving delay and loss performance and optimizing perceptual quality performance (see Chapter 5 for simulation results).

4.4.4 Slow-Start

In order to avoid a sudden surge of packet injection during the start-up of a new packet video stream or streams coming out of a congested state, the proposed flow control mechanism employs a slow-start algorithm. This algorithm is different from the one used in TCP. The new slow-start algorithm increases the window size linearly using two different rates. A slow-start threshold ST is used to switch between the two rates. The rate at which the window is opened up will be faster if the current window size is less than ST.

At the beginning of a congestion period, ST is set in the previous congestion period to be

$$ST = \left(\frac{1+\rho}{2}\right) W_{i-1}$$

If the current window size is below ST, the window is increased as

$$W_i = W_{i-1} + \gamma_1 \cdot \Delta_R$$

where $\gamma_1 = N_{RT}$. Otherwise, it is increased as

$$W_i = W_{i-1} + \gamma_2 \cdot \Delta_R$$

where γ_2 has a constant value of 1. In addition, the window should not exceed the maximum allowable number of bytes in transit

$$W_i = \min(W_{i-1} + \gamma_k \cdot \Delta_R, R_{MAX} \cdot N_{ETE})$$

where $k = 1$ or 2.

It follows from the previous assumption that slow convergence of video traffic to its maximum allowable output level does not cause catastrophic network congestion collapse. In fact, slow window resizing for an adaptive packet video session is highly desirable because it leads to gradual changes in video quality. However, it is necessary to expand the window of a packet video session faster initially to keep in line with the rapid window adjustment in other flow controlled sessions such as TCP.

4.4.5 Rate-Control Stabilization

The perceptual quality of a video stream is determined not only by the encoding bit-rate but also by its first derivative. If the output rate of a video source oscillates too quickly, the playback video may appear to be phasing-in and -out. This artifact is particularly noticeable and annoying to human viewers. Therefore, it is important to adjust the output rate gradually so that the changes are subtle to viewers. In the proposed flow control mechanism, this is achieved by a slow window adjustment policy that adjusts the window only once every round-trip time instead of once for each feedback acknowledgment received. The output rate is updated at the beginning of each time interval and is held constant during that interval. The goal is to set the output bit-rate at a level so that the current window is filled in one $T_{RTT} + 2D_{RTT}$ time.

Specifically, the encoding rate B is calculated as

$$B_i = B_{i-1} + \frac{W_{i-1} - L_{ETE}}{N_{RT}}$$

where L_{ETE} is the estimated number of bytes in transit for each video session. L_{ETE} is computed from

$$L_{ETE} = \frac{N_{ETE} \times L_{RT}}{SEQ_{ACK} + M - SEQ_{ACK-1}}$$

where L_{RT} is the total number of bytes that have not been acknowledged and SEQ_{ACK-1} is the previously acknowledged sequence number. The sender is expected to keep track of the value of L_{RT}.

4.5 Implementation Specifics

To carry out the flow control and coding adaptation proposed in this thesis, certain information needs to be exchanged between the sender and the receiver. In addition to the sequence number and time-stamp that are already contained in every packet, several other pieces of application-specific information needed to be sent in each direction. From the sender to the receiver, these include

- *SEQ* (16 bits) - This is the frame sequence number
- B_i (8 bits) - This indicates the coded bit-rate for the current frame
- *SB* (8 bits) - This gives the subband to which the current packet belongs.

From the receiver back to the sender, these include

- SEQ_{ACK} (16 bits) - This is the sequence number of the frame being acknowledged
- P_D (1 bit) - This bit indicates if a frame is partially received (1 = true, 0 = false)
- *M* (7 bits) - This gives the number of missing frames (if any) between the current awaited frame and the most recently received frame

- *U* (32 bits) - This value gives the average end-to-end delay of packets for the frame being acknowledged.

Because the proposed flow control mechanism does not require special support from the underlying transport network, it can be used with UDP, IP multicast or RTP [42]. RTP is a transport for real-time applications. A RTP data packet header has fixed fields that are useful to the flow control. Specifically, it provides a packet sequence number, a time-stamp and a payload type for identifying the content type of a packet. In addition, it allows one header extension to augment the packet header in an application-specific way. For the adaptive packet video system constructed in this thesis, the extra information is sent in header extensions to RTP packets.

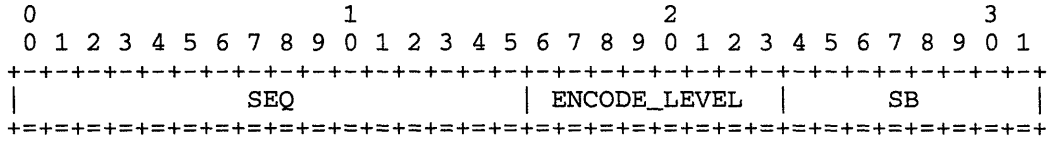

Figure 4.1. Format of header extension for a RTP packet used by senders. SEQ = *SEQ*; ENCODE_LEVEL = *B*; SB = *SB*.

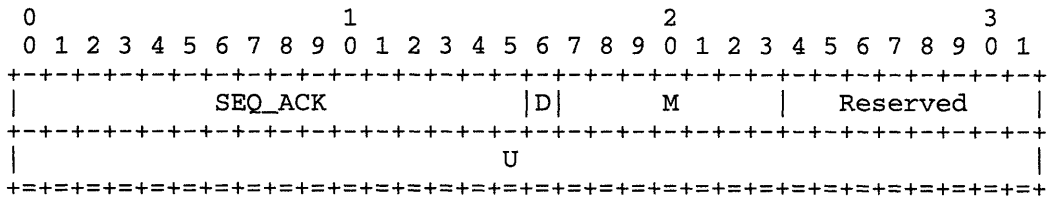

Figure 4.2. Format of header extension for a RTP packet used by receivers. SEQ_ACK = SEQ_{ACK}; D = P_D; M = *M*; U = *U*.

Chapter 5

Simulation Experiments

To validate the design of the proposed flow control mechanism for packet video, a network simulator is used to provide a controlled hosting environment. The simulation results are used to verify the effectiveness of the control algorithms and to fine-tune them. In addition, the impacts of these algorithms on the performance of TCP and packet video sessions are carefully studied.

5.1 Network Topology

The dynamic behavior and the performance of the proposed flow control mechanism is simulated and analyzed using The Network Simulator (or NetSim). The NetSim is a software-based network simulator and can simulate system that can be modeled by a network of components that communicate by sending messages to each other. Currently, a number of network topologies have been built for the simulator. The particular topology used in this thesis is the Dartnet, a sub-network inside the Internet. The Dartnet is an experimental test-bed to support network research in such areas as advanced routing algorithms and advanced network applications. The general topology for the Dartnet is given in Figure 5.1 and the simulation model (completed with end nodes) in Figure 5.2.

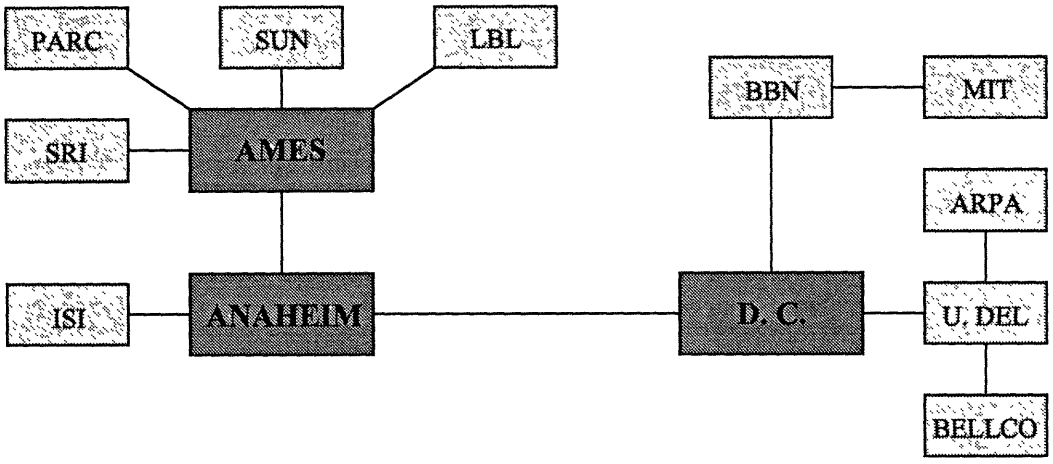

Figure 5.1. Dartnet topology.

Today, the Dartnet is composed of a cross-country T1 backbone and a dozen participating sites connected to the backbone via T1 circuits. Within each site, Ethernet provides the LAN connectivity to the host machines. In the simulator model, the Dartnet has been "upgraded" so that the impacts of newer communications and processing technologies can be tested. In particular, all the T1 circuits (including the backbone) have been replaced with T3 circuits and Ethernet LAN's replaced with FDDI rings. Also, the SPARC-1 stations (at 10 MIPS) have been substituted by Alpha-class machines (at 100 MIPS). The topology of the network remains the same however. The upgraded Dartnet provides a good environment for studying new flow control algorithms operating over links with large delay-bandwidth products. Currently, the Dartnet has a maximum end-to-end delay of 83 milli-seconds (between MIT and Xerox PARC). The bottleneck link in the Dartnet is the backbone link (the D.C.-Anaheim T3 circuit).

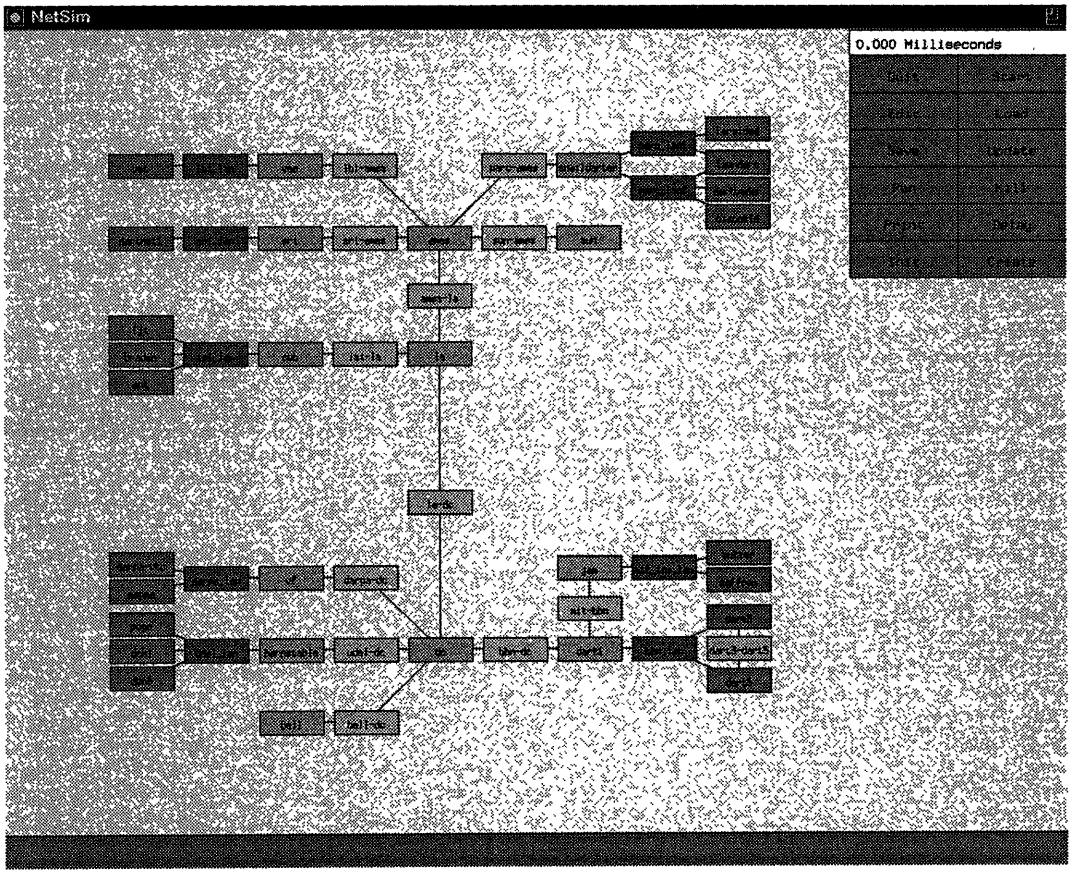

Figure 5.2. Dartnet simulation model for the NetSim.

5.2 Simulation Set-ups

In order to study the effects of the proposed flow control algorithms, two simulation set-ups with the same combination of traffic sources are used, as shown in Figure 5.3. The difference between the two is that there is no flow-controlled video session in first set-up and a mix of flow controlled and non-flow-control video sessions in the second. Video and Poisson traffic flows are uni-directional. Feedback information (if any) for video sessions travels in the reverse direction. TCP traffic flows are bi-directional. There are three main types of traffic in the each of the simulation set-ups:

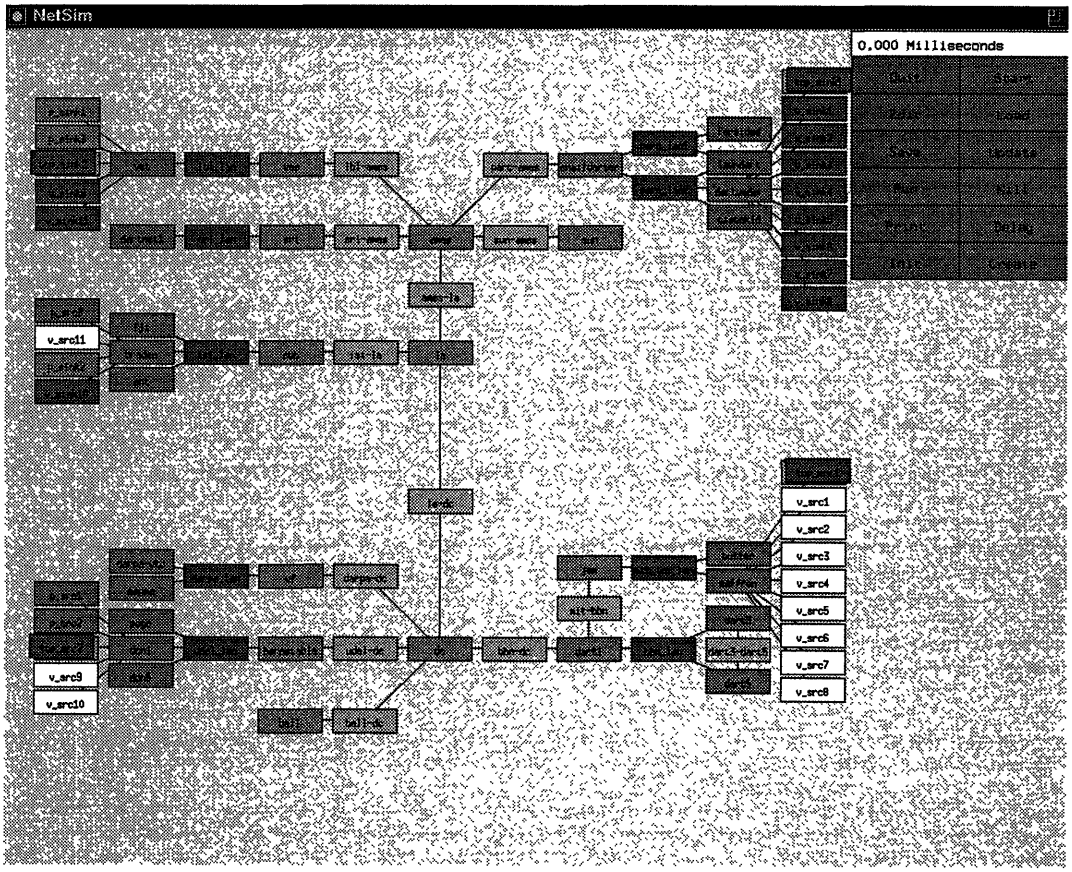

Figure 5.3. Simulation set-up.

- 11 real-time packet video streams (either flow controlled or non-flow-controlled),
- 2 Purdue TCP streams, and
- 3 Poisson UDP streams.

In the current set-ups, there are a total of eight video streams and one Purdue TCP stream going from MIT's `saffron` to Xerox PARC's `dartvader`. The video streams are non-flow-controlled in the first case and flow controlled in the second. In addition, there are three sets of cross traffic that are the same for both set-ups. The first set goes between University of Delaware's `dcn1` and LBL's `vet`. It consists of one non-flow-controlled video

stream, one Poisson stream and one Purdue TCP stream. The second set goes from `dcn1` to ISI's `braden` and consists of one non-flow-controlled video stream and one Poisson stream. The third group goes from `braden` to `vet` and also consists of one non-flow-controlled video stream and one Poisson stream. All stream information is summarized in Table 5.1.

Stream Name	Stream Type	Start Time (seconds)	Duration (seconds)	Bit-Rate (Mbits/s)
v_src1	Video 1 or 2	2.0	100.0	1.926/4.653
v_src2	Video 1 or 2	2.0	100.0	1.926/4.653
v_src3	Video 1 or 2	4.0	100.0	1.926/4.653
v_src4	Video 1 or 2	4.0	100.0	1.926/4.653
v_src5	Video 1 or 2	6.0	100.0	1.926/4.653
v_src6	Video 1 or 2	6.0	100.0	1.926/4.653
v_src7	Video 1 or 2	8.0	100.0	1.926/4.653
v_src8	Video 1 or 2	8.0	100.0	1.926/4.653
v_src9	Video 1	2.0	100.0	4.653
v_src10	Video 2	2.0	100.0	1.926
v_src11	Video 2	2.0	100.0	1.926
tcp_src1	Purdue TCP	10.0	Variable	Variable
tcp_src2	Purdue TCP	10.0	Variable	Variable
p_src1	Poisson	0.0	Infinite	4.000
p_src2	Poisson	0.0	Infinite	4.000
p_src3	Poisson	0.0	Infinite	4.000

Table 5.1. List of streams used in the simulation set-ups.

For the simulation set-up, there are two types of video streams. These are summarized in Table 5.2.

Video Stream Type	1	2
Resolution (pixels)	640×480	320×240
Frame-rate (frames/sec)	30	30
Max Bit-Rate (bits/frame)	155,104	64,200

Table 5.2. Scalable video stream types.

5.3 Simulations of Non-Flow-Controlled Packet Video Sessions

Several simulation runs are carried out with the first experiment set-up. In the case, none of the packet video stream is flow controlled. These simulation runs vary in that they use different combinations of video sources for *v_src1* through *v_src8* (i.e., different mix of video type 1 and type 2). Results from three of these runs (for maximum, medium and minimum traffic load conditions) are presented here.

Under the maximum loading condition, streams *v_src1* through *v_src8* transmit at 4.653 Mbits/s each. The total average traffic from the video and Poisson streams going over the backbone (i.e., D.C.-Anaheim link) is 51.803 Mbits/s. In other words, the backbone link is 115% utilized. This already exceeds the speed of the backbone, which is a T3 circuit at 45 Mbits/s. In addition, there are two Purdue TCP streams going over the same bottleneck link. The backbone capacity is exceeded after the last two video streams (*v_src7* and *v_src8*) are activated. Therefore, significant increases in end-to-end delay and packet loss rate should be expected after the 8.0 second into the simulation. The resulting end-to-end delay for *v_src1* is given in Figure 5.4 and packet loss rate in Figure 5.5. The average throughput rate for *v_src1* (averaged over each video frame) is plotted in Figure 5.6. The delay and loss curves for the other seven packet video streams (*v_src2* through *v_src8*) exhibit similar behaviors and, consequently, are left out to save space.

The TCP streams are `ftp` sessions doing bulk transfer of 50 Mbytes long data files each. Since the TCP streams start at the 10.0 second, their throughput rates will be low because the backbone link is already overloaded. The average TCP throughput rate for *tcp_src1* is shown in Figure 5.7.

Under the medium loading condition, streams *v_src1* through *v_src4* transmit at 1.926 Mbits/s each, and *v_src5* through *v_src8* at 4.653 Mbits/s each. Under this condition, the

backbone link is 91% utilized when the TCP streams start. The corresponding plots for this case are given in Figure 5.8 to Figure 5.11.

Under the minimum loading condition, streams *v_src1* through *v_src8* transmit at 1.926 Mbits/s each. Here, the backbone link is 67% utilized when the TCP streams start. The corresponding plots are given in Figure 5.12 to Figure 5.15.

As the traffic level increases in the network and approaches a congested state at the bottleneck switch/link, both end-to-end delay and packet loss rate multiplies. Furthermore, as we have predicted, non-flow-controlled streams cannot coupe with flow controlled streams in terms of resource sharing. When the network becomes congested, flow controlled traffic yields to the non-flow-controlled traffic and gets squeezed out of the network (see Figure 5.7 for an example).

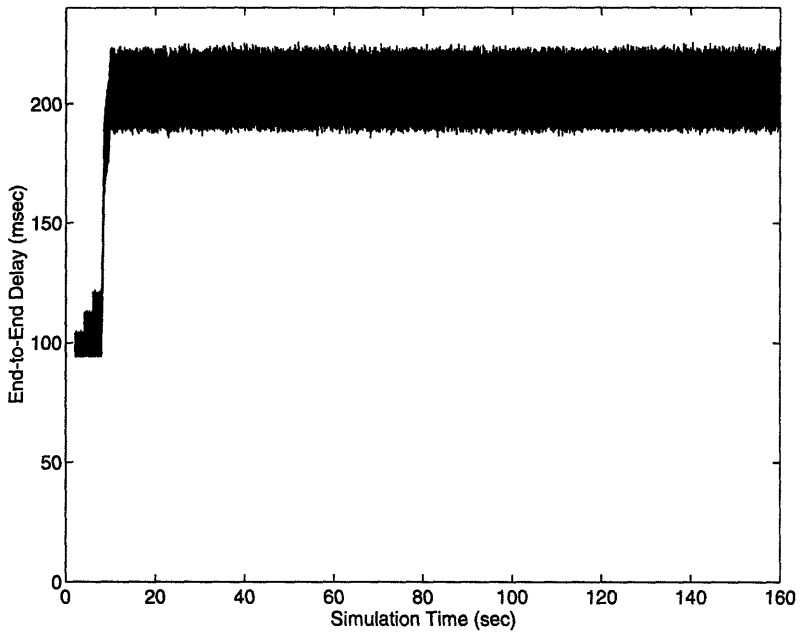

Figure 5.4. End-to-end delay for *v_src1* under maximum loading condition.

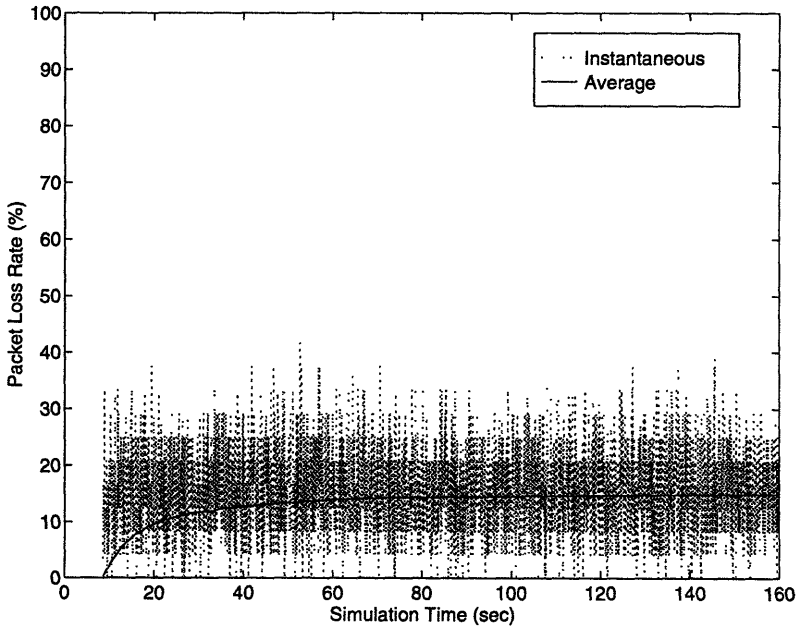

Figure 5.5. Packet loss rate for *v_src1* under maximum loading condition.

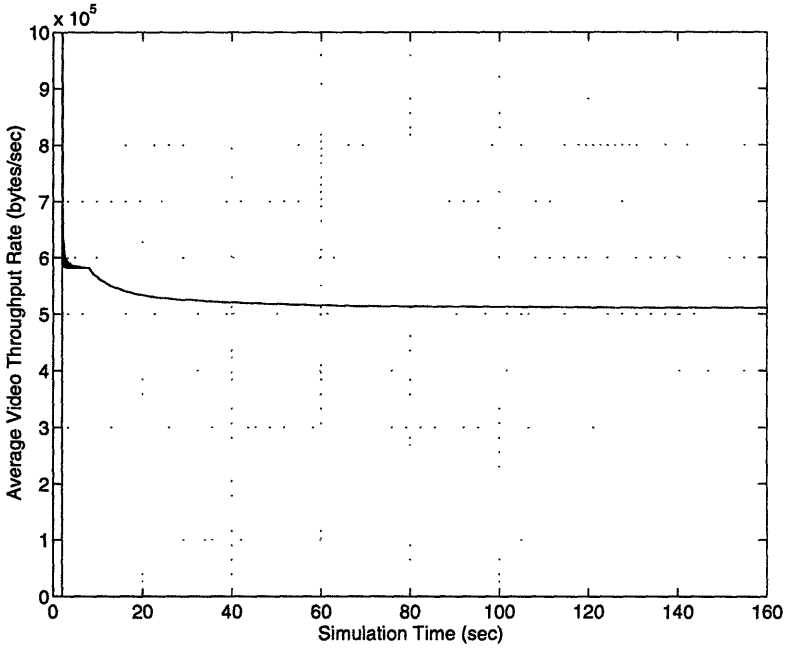

Figure 5.6. Average throughput rate for *v_src1* under maximum loading condition.

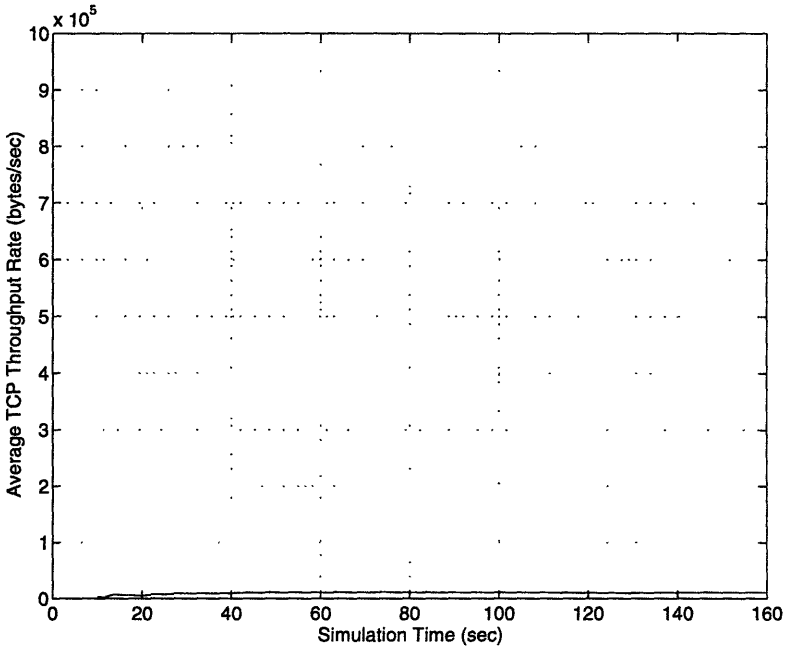

Figure 5.7. Average throughput for *tcp_src1* under maximum loading condition.

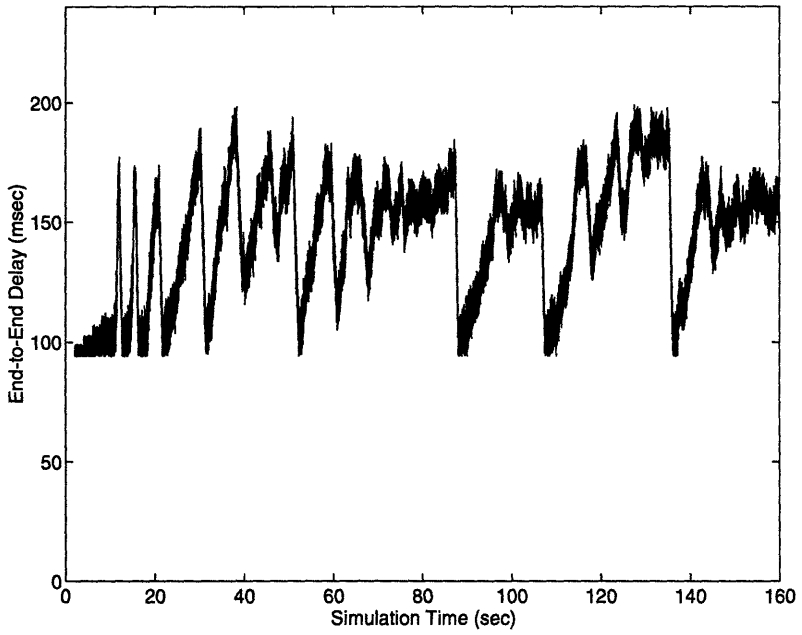

Figure 5.8. End-to-end delay for *v_src1* under medium loading condition.

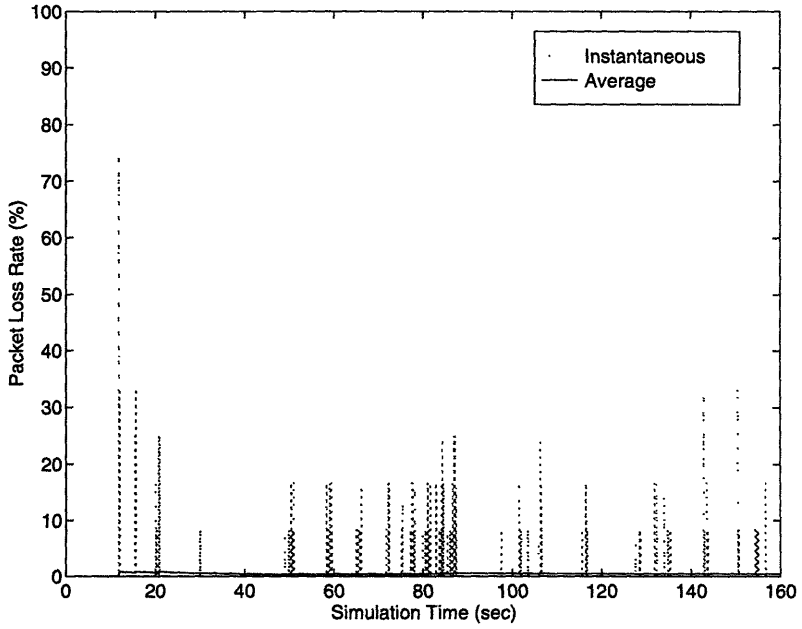

Figure 5.9. Packet loss rate for *v_src1* under medium loading condition.

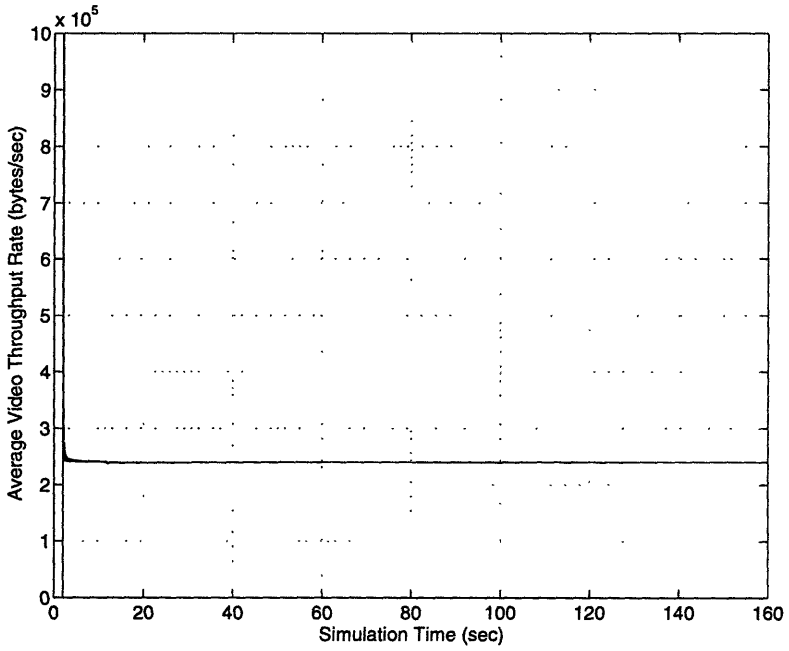

Figure 5.10. Average throughput rate for *v_src1* under medium loading condition.

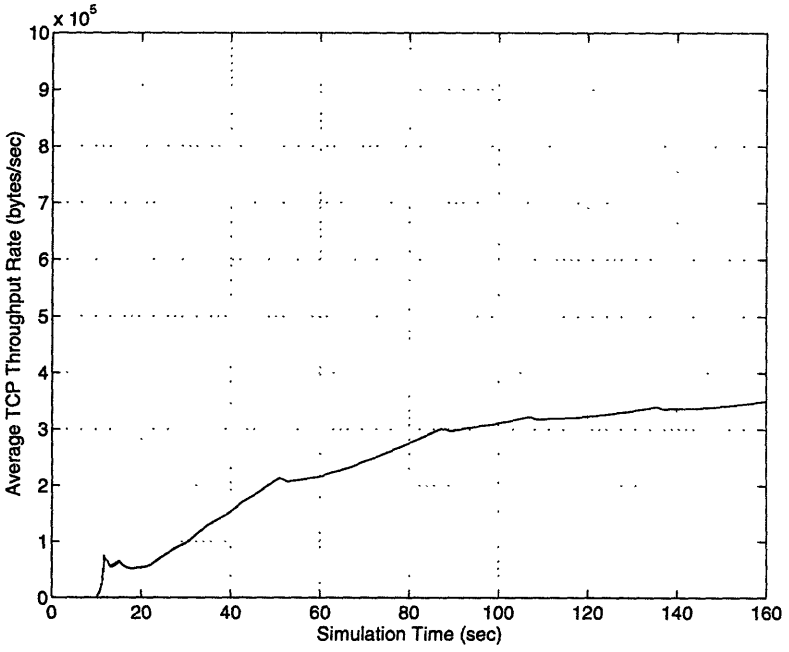

Figure 5.11. Average throughput for *tcp_src1* under medium loading condition.

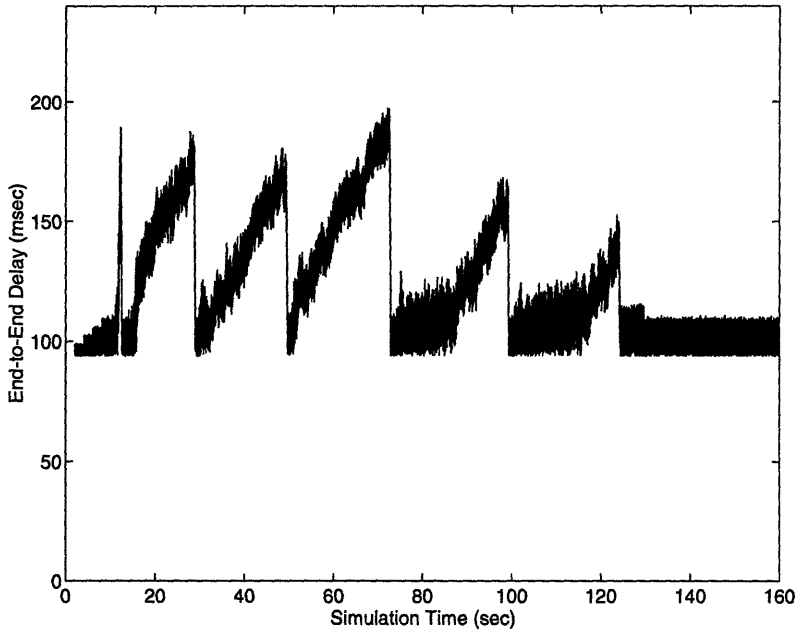

Figure 5.12. End-to-end delay for *v_src1* under minimum loading condition.

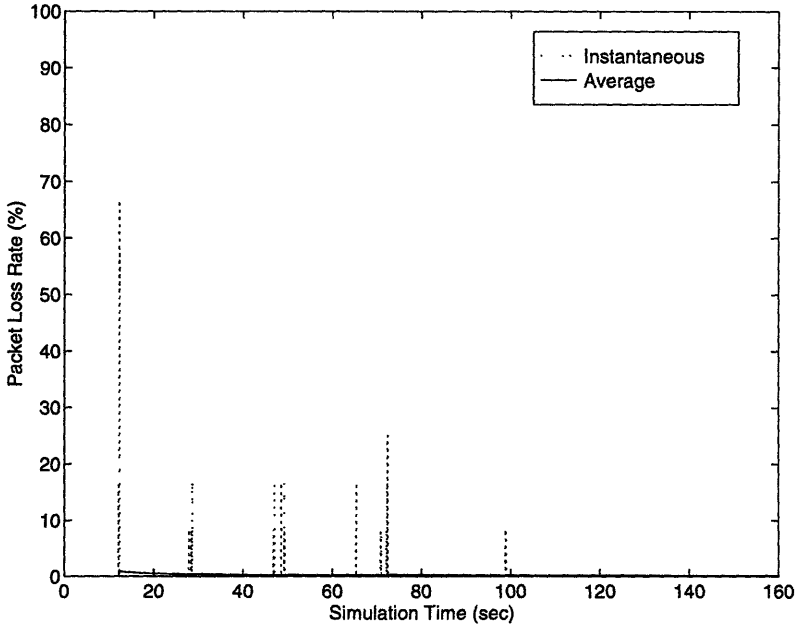

Figure 5.13. Packet loss rate for *v_src1* under minimum loading condition.

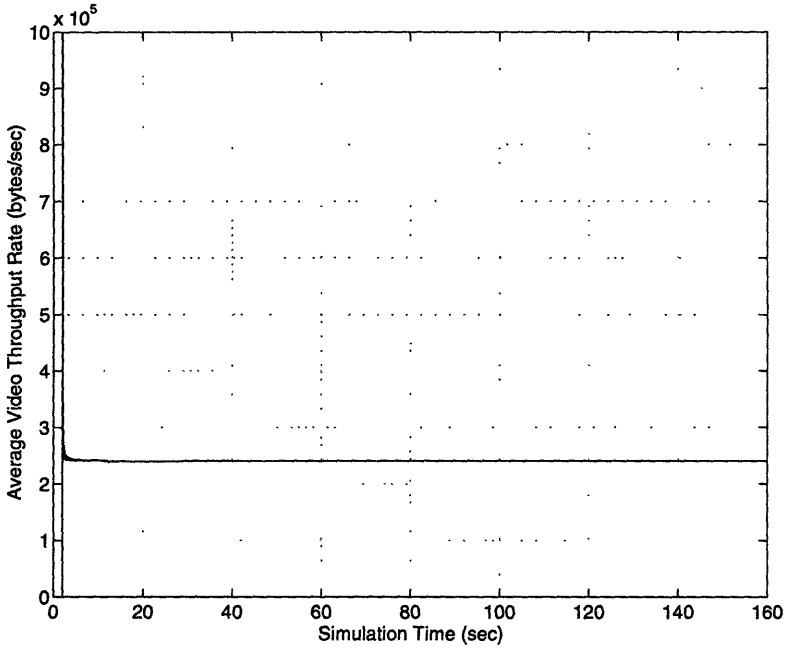

Figure 5.14. Average throughput rate for *v_src1* under minimum loading condition.

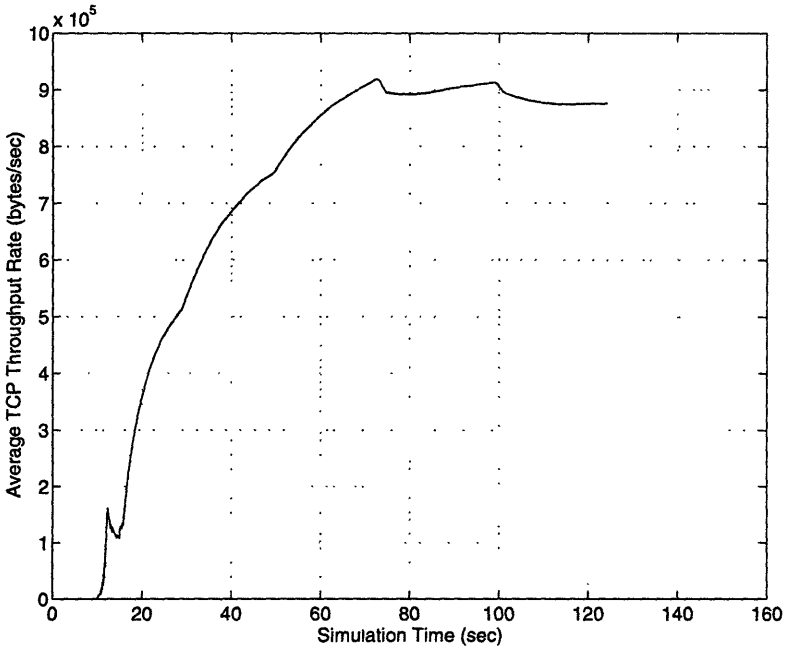

Figure 5.15. Average throughput for *tcp_src1* under minimum loading condition.

5.4 Simulations of Flow Controlled Packet Video Sessions

The identical set of simulation runs are conducted for the second experiment set-up. In this round of simulations, the proposed flow control mechanism (as described in Chapter 4) is activated for streams *v_src1* through *v_src8*. Again, various traffic load conditions are simulated and the performance of the flow control mechanism is examined under the conditions. The results will be presented shortly.

To understand the true impacts of this new flow control mechanism on a packet video system, it is important to gauge the perceptual quality of the received video in addition to the ordinary delay or loss measures. Ideally, a subjective measure of the perceptual quality such as the Mean Opinion Score (MOS) should be used. However, the MOS would involve many human observers and is not practical for this thesis. Instead, two objective measures are used. The first one is the Signal-to-Noise Ratio and the second one is the encoding level for each video frame.

Numerous experiments have shown that the SNR is an imperfect measure of the perceptual quality of a video sequence. Nevertheless, it gives a reasonable first-order approximation of the perceptual quality and can be computed easily. In addition, the encoding level for a video frame gives the absolute upper-bound on its perceptual quality. As mentioned in the previous chapter, the scalable encoder for a flow controlled packet video system is required to supply three parameters — maximum encoding rate, minimum encoding rate and minimum rate change for each adjustment. In return, it receives (from the flow control mechanism) a single parameter specifying the maximum bit-rate for the next coded video frame. This bit-rate can be used to indicate the maximum achievable perceptual quality.

For the current simulation set-ups, two types of multiresolution video streams are used. These are summarized in Table 5.3.

Scalable Video Stream Type	1	2
Resolution (pixels)	640×480	320×240
Frame-rate (frames/sec)	30	30
Max Bit-Rate (bits/frame)	155,104	64,200
Min Bit-Rate (bits/frame)	54,304	40,800
Min Bit-Rate Adjustment (bits)	536	232
Number of Encoding Levels	9	9

Table 5.3. Scalable video stream types.

To illustrate the (limited) correlation between the SNR and the perceptual quality of a video frame, a frame from a Type 2 video stream is scalably encoded at all possible bit-rates. The SNR for the video frame coded at these bit-rates is given in Figure 5.16. The perceptual quality at these bit-rates is shown in Figure 5.17.

The results for the three simulation runs under the maximum, medium and minimum traffic load conditions are shown in Figure 5.18 through Figure 5.35.

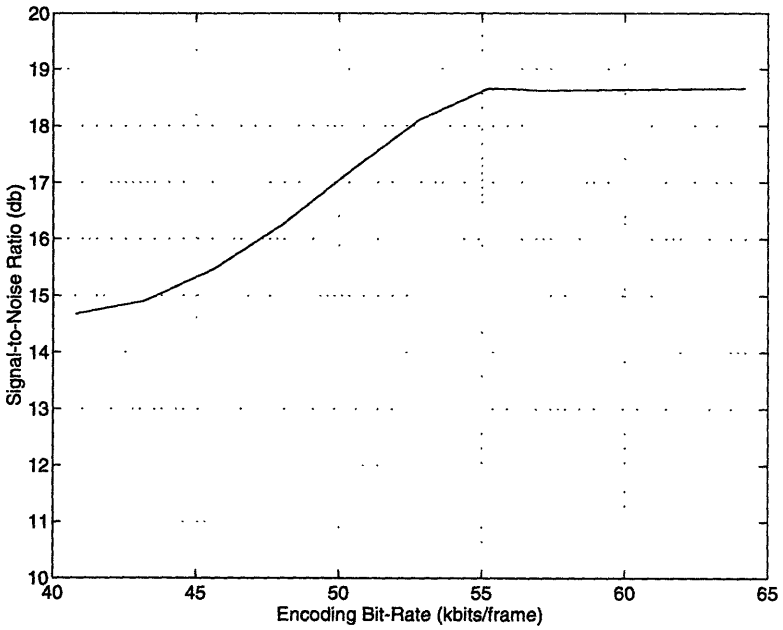

Figure 5.16. The SNR of a video frame encoded at different bit-rates.

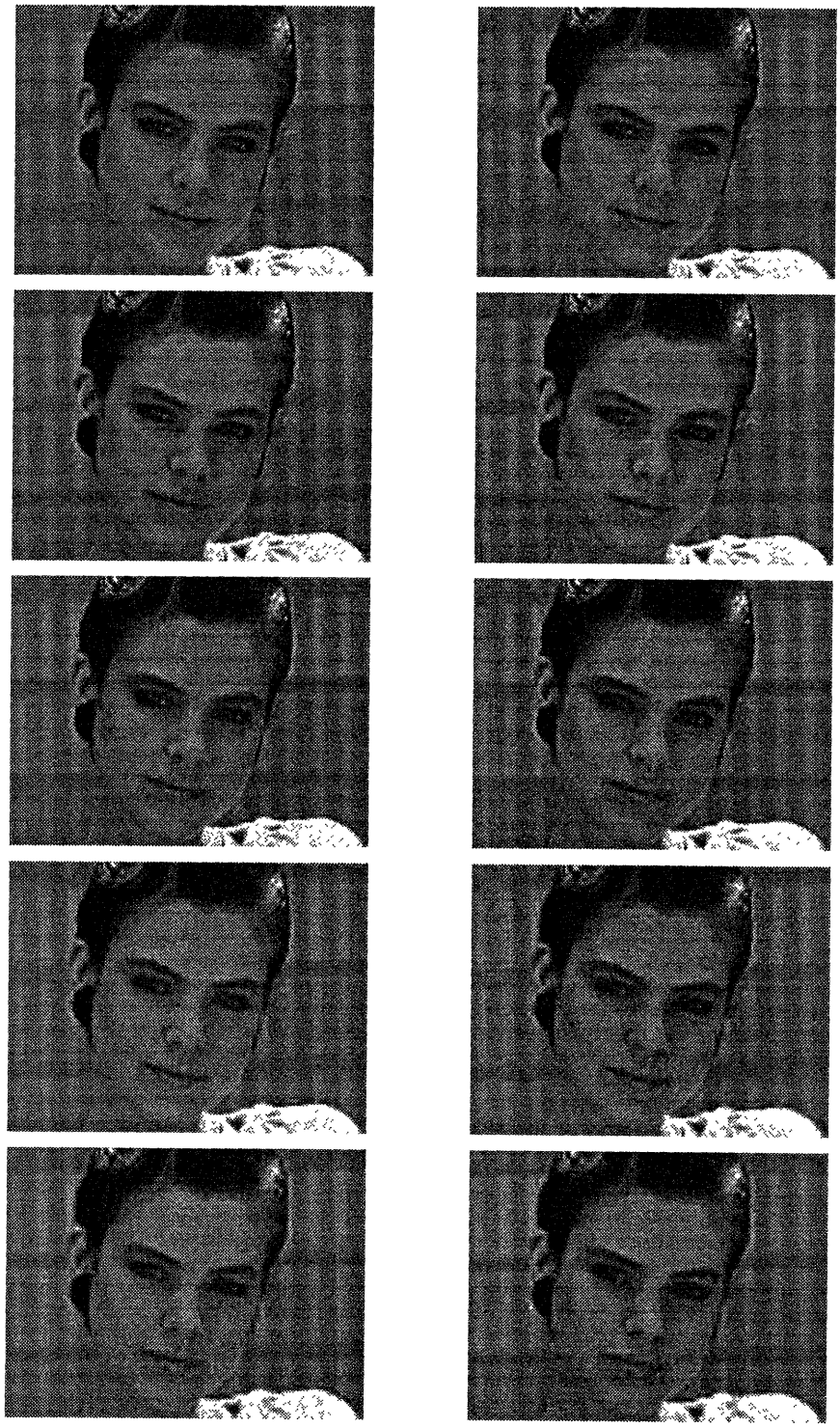

Figure 5.17. The perceptual quality of a video frame encoded at different bit-rates.

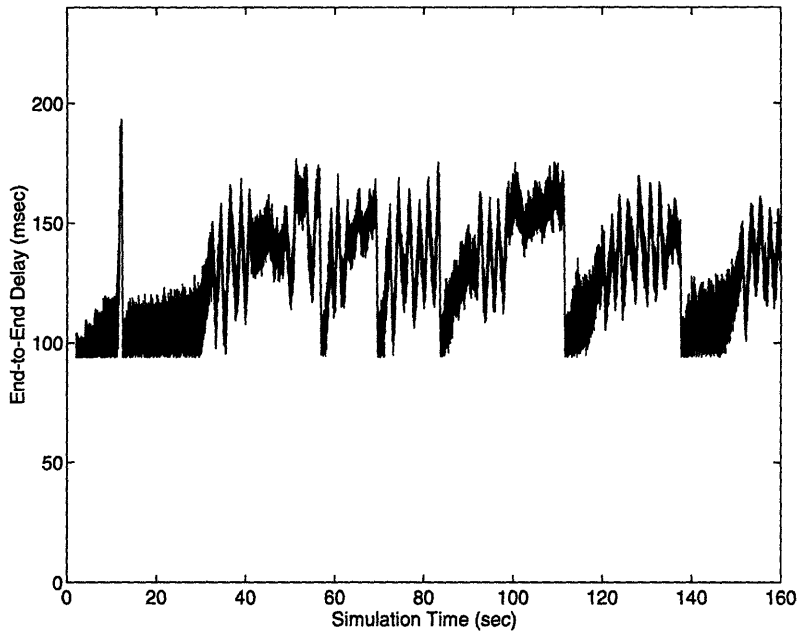

Figure 5.18. End-to-end delay for adaptive *v_src1* under maximum loading condition.

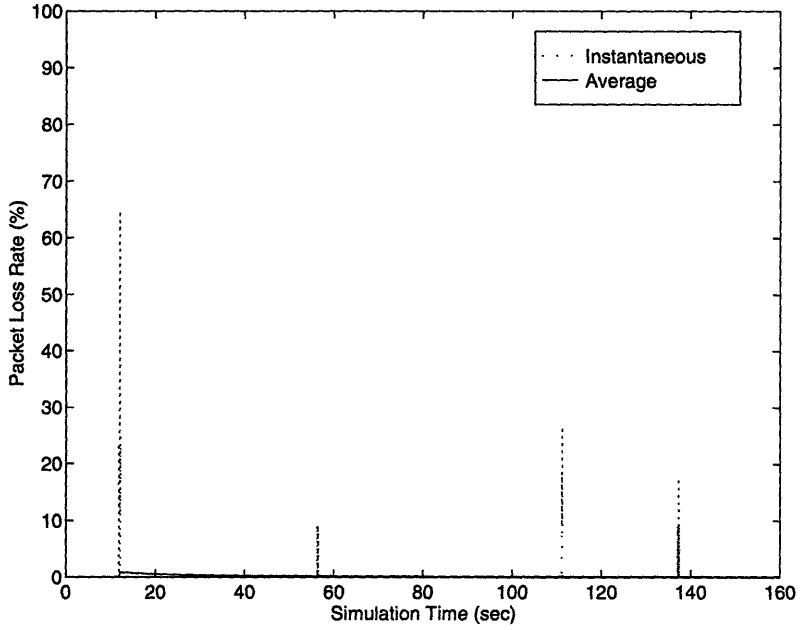

Figure 5.19. Packet loss rate for adaptive *v_src1* under maximum loading condition.

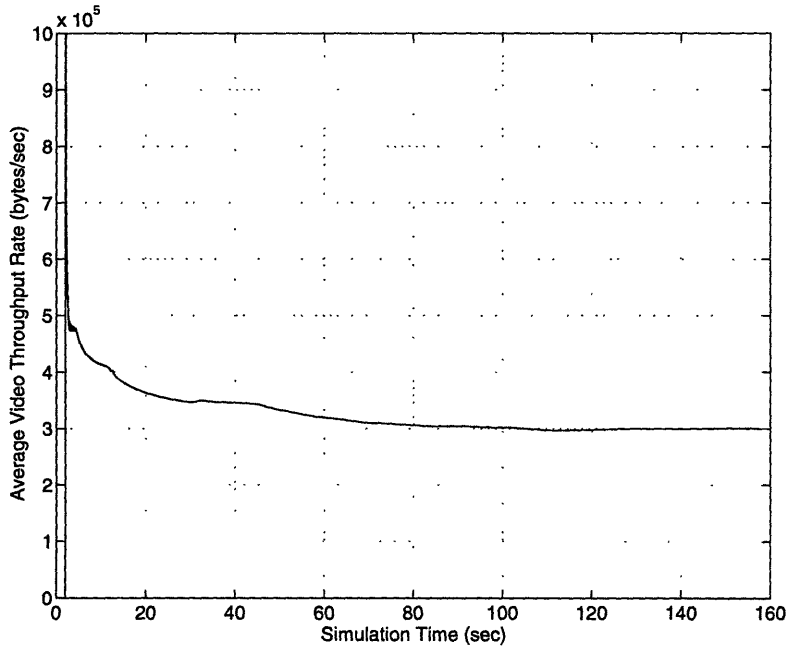

Figure 5.20. Average throughput rate for adaptive *v_src1* under maximum loading condition.

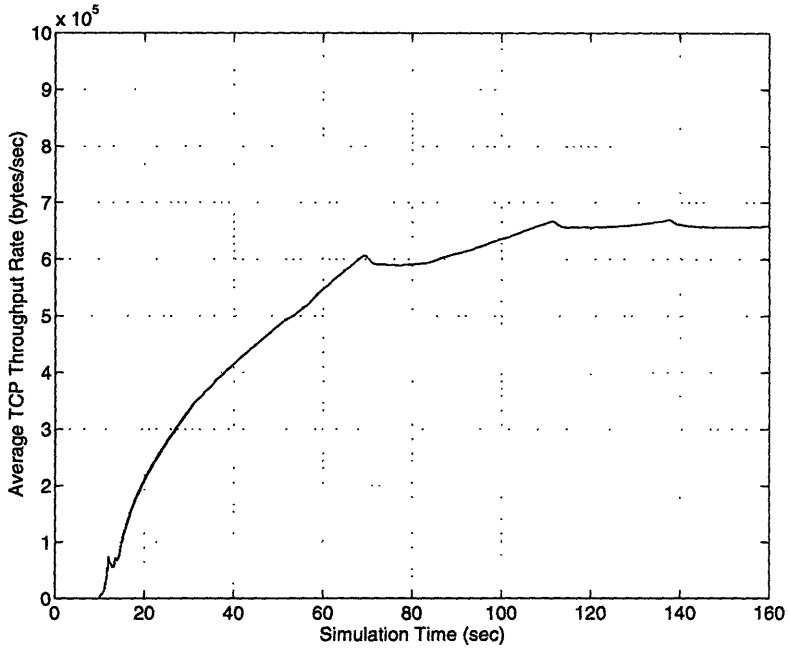

Figure 5.21. Average throughput for adaptive *tcp_src1* under maximum loading condition.

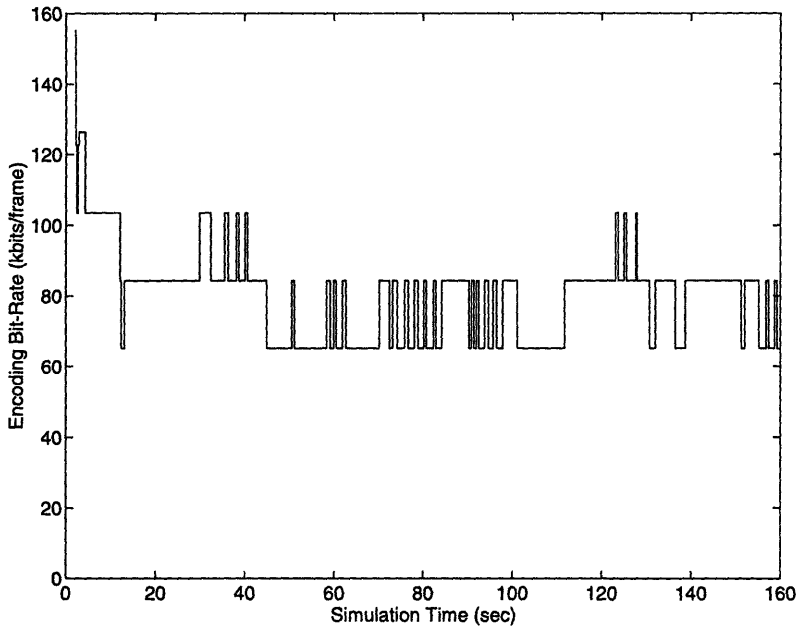

Figure 5.22. Encoding bit-rate for adaptive *v_src1* under maximum loading condition.

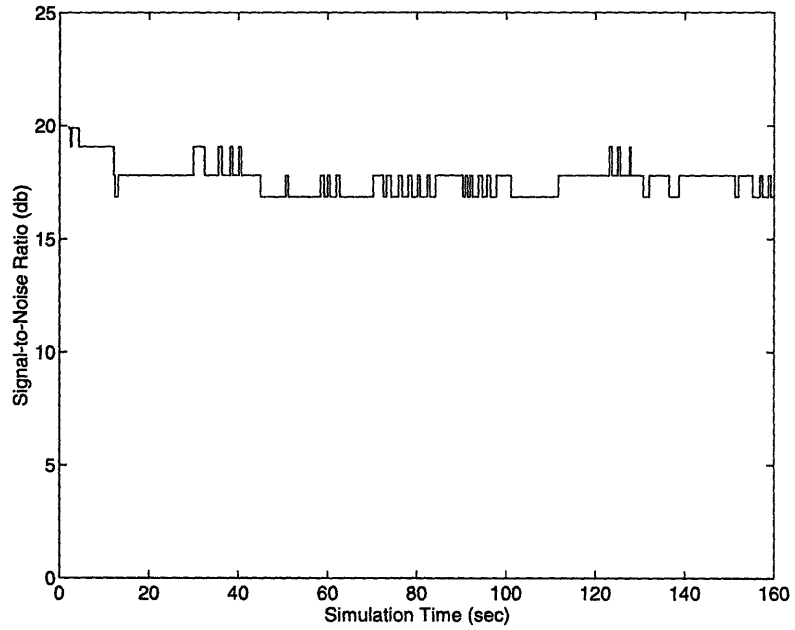

Figure 5.23. SNR for adaptive *v_src1* under maximum loading condition.

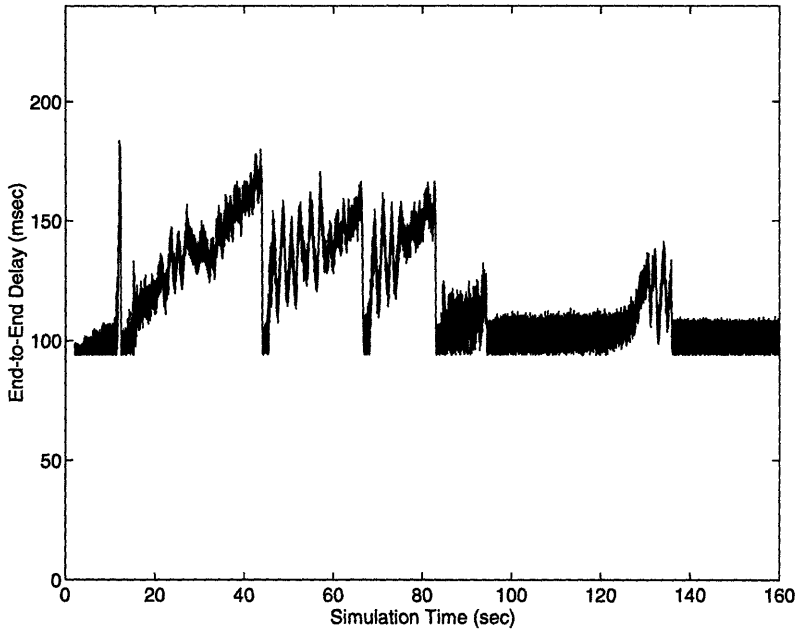

Figure 5.24. End-to-end delay for adaptive *v_src1* under medium loading condition.

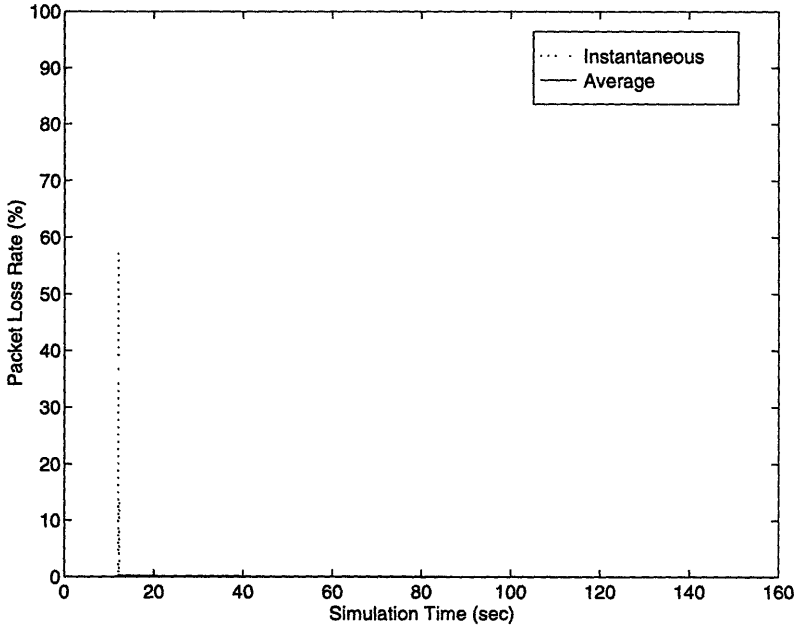

Figure 5.25. Packet loss rate for adaptive *v_src1* under medium loading condition.

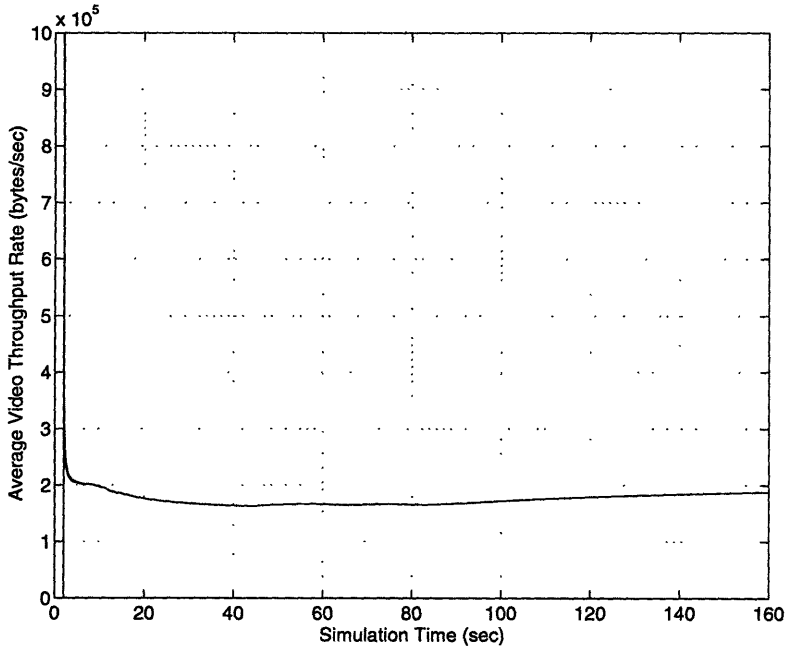

Figure 5.26. Average throughput rate for adaptive *v_src1* under medium loading condition.

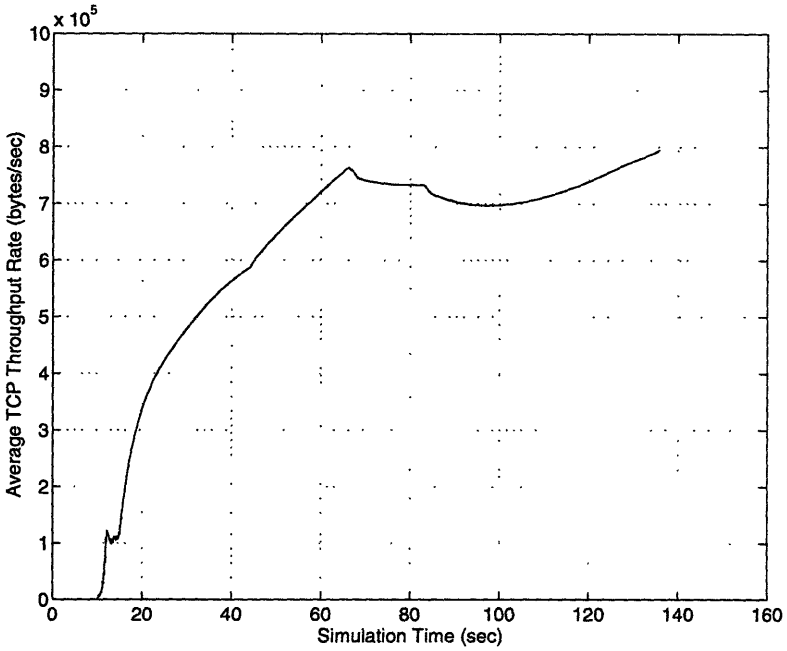

Figure 5.27. Average throughput for adaptive *tcp_src1* under medium loading condition.

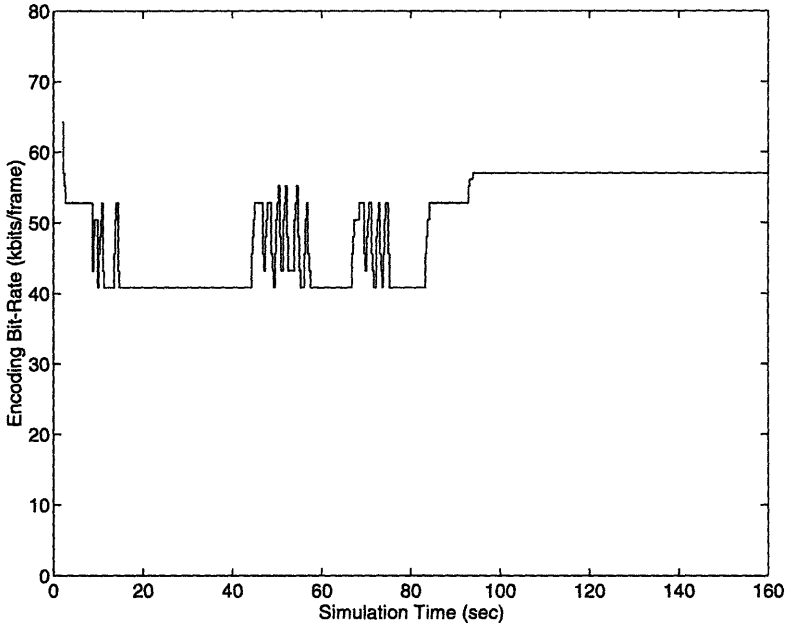

Figure 5.28. Encoding bit-rate for adaptive *v_src1* under medium loading condition.

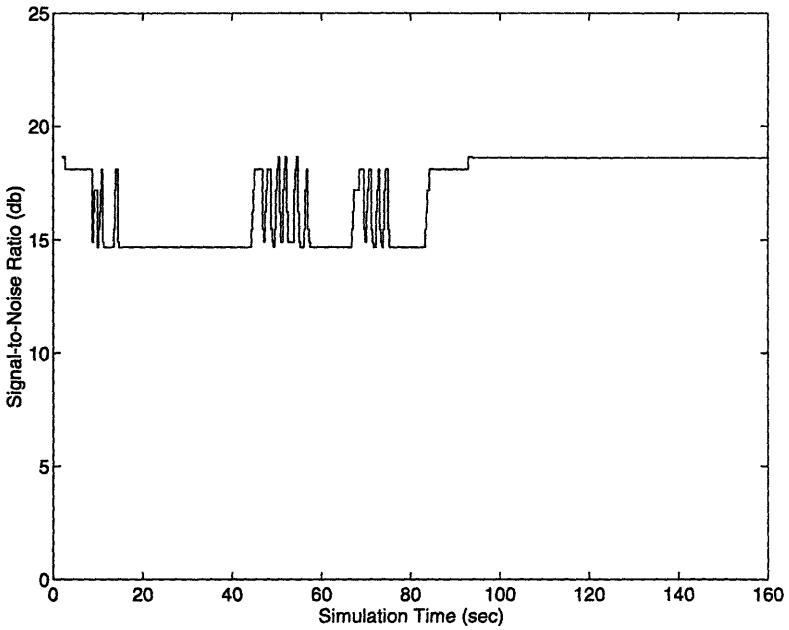

Figure 5.29. SNR for adaptive *v_src1* under medium loading condition.

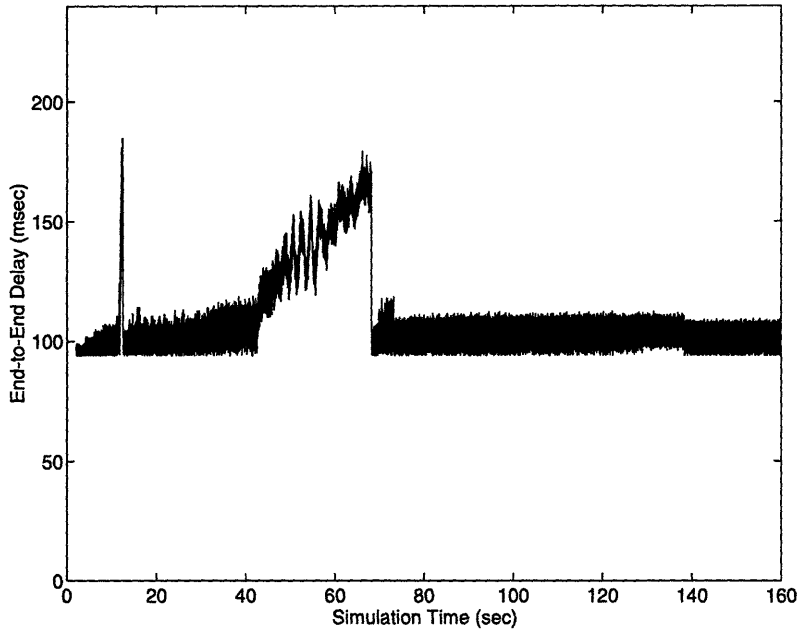

Figure 5.30. End-to-end delay for adaptive *v_src1* under minimum loading condition.

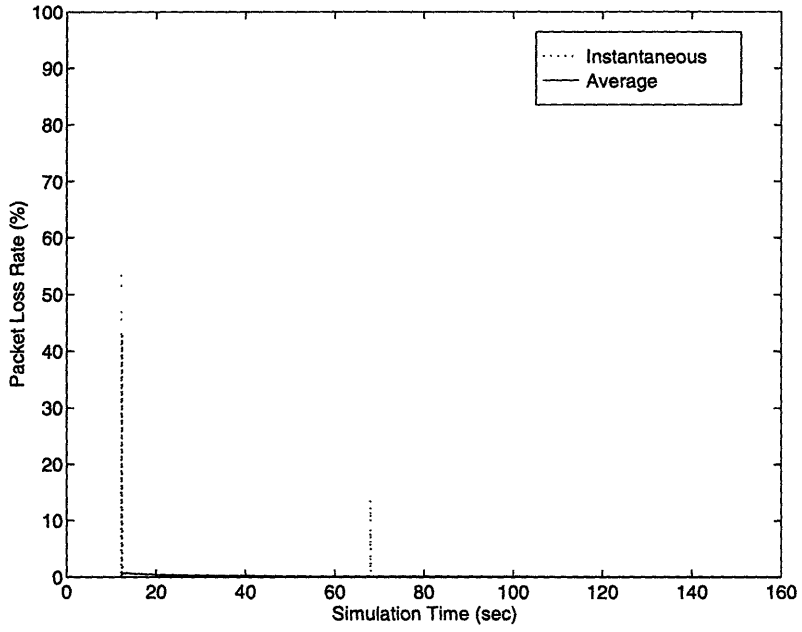

Figure 5.31. Packet loss rate for adaptive *v_src1* under minimum loading condition.

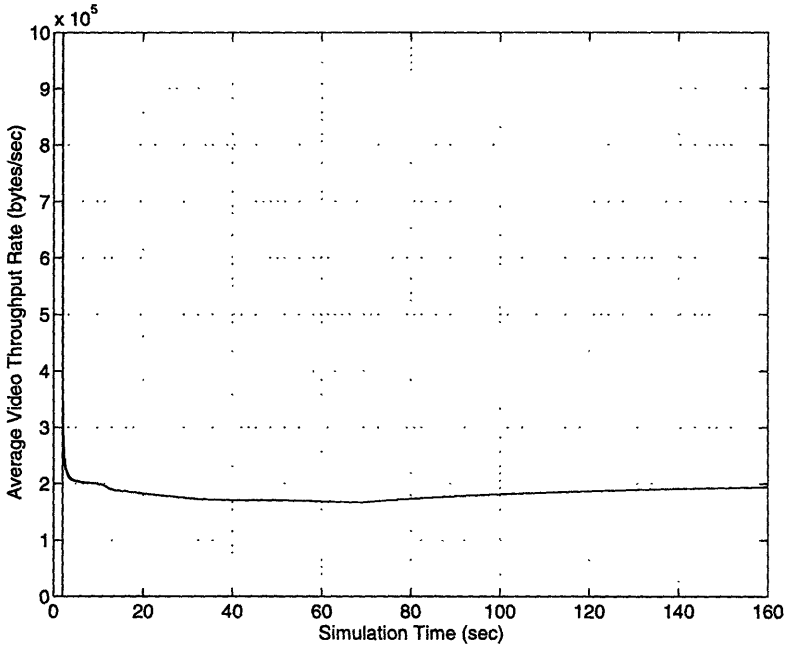

Figure 5.32. Average throughput rate for adaptive *v_src1* under minimum loading condition.

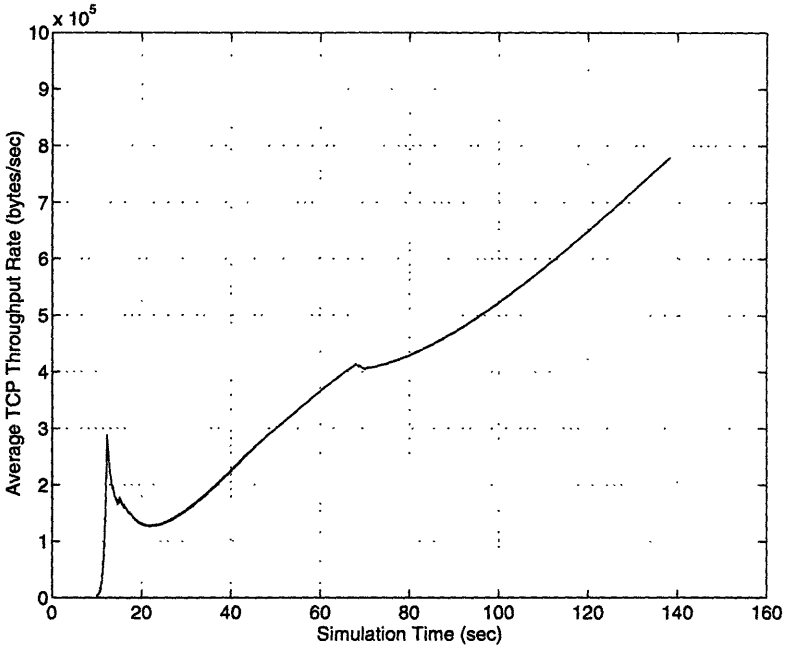

Figure 5.33. Average throughput for adaptive *tcp_src1* under minimum loading condition.

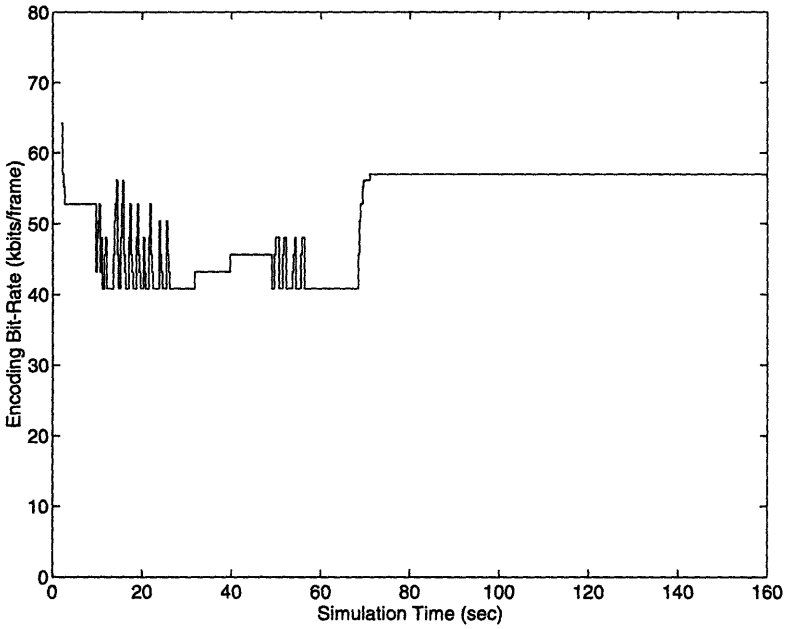

Figure 5.34. Encoding bit-rate for adaptive *v_src1* under minimum loading condition.

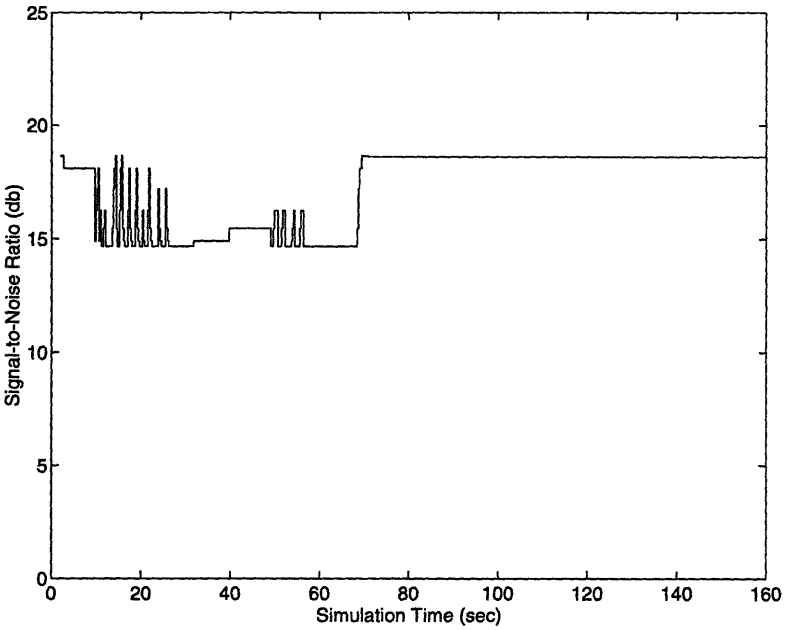

Figure 5.35. SNR for adaptive *v_src1* under minimum loading condition.

For the three simulation cases, the amount of traffic traveling through the bottleneck link (in the absence of any TCP traffic) with respect to its capacity is listed in Table 5.4. The maximum link utilization factor gives the percentage of bottleneck link bandwidth used when all video sources are generating traffic at their respective highest output levels. The minimum link utilization factor gives the percentage of bottleneck link bandwidth used when all adaptive video sources have backed off to their lowest output levels. The TCP traffic is not included here because the peak throughput rate cannot be computed accurately. However, the TCP throughput rate is upper-bounded by the maximum throughput of each host's network interface less the bandwidth occupied by the video and UDP traffic.

Simulation Case	Max Link Utilization	Min Link Utilization
Maximum loading	115%	61%
Medium loading	91%	58%
Minimum loading	67%	54%

Table 5.4. Bottleneck link utilizations.

By comparing the figures in 5.4 through 5.15 with the corresponding figures in 5.18 through 5.35, several findings are immediately obvious. Drastic performance improvements are found in the areas of end-to-end delay, packet loss rate and TCP throughput.

For each simulation case, both the maximum and the average end-to-end delays of a flow controlled video stream are reduced. The simulation results are summarized in Table 5.5 and 5.6. The improvement in the worst-case end-to-end delay ranges from 6.4% for the minimum loading case to 14.3% for the maximum loading case; the improvement in the average delay ranges from 12.3% for the minimum loading case to 36.0% for the maximum loading case.

Simulation Case	Non-Flow-Controlled	Flow Controlled
Maximum loading	226 ms	193 ms
Medium loading	199 ms	184 ms
Minimum loading	197 ms	185 ms

Table 5.5. Maximum end-to-end delay.

Simulation Case	Non-Flow-Controlled	Flow Controlled
Maximum loading	200 ms	128 ms
Medium loading	145 ms	117 ms
Minimum loading	123 ms	108 ms

Table 5.6. Average end-to-end delay.

The time series plots for the end-to-end delay show that the proposed flow control mechanism responds quickly to the changing network traffic load. For example, in every simulation run, two TCP streams (with different source-destination pairing but traveling through the same bottleneck link) start at 10.0 second into the simulation. This sudden surge of traffic load causes a sharp increase in the end-to-end network latency, which in turn triggers the flow control mechanism to reduce the output rate of each affected adaptive packet video source. This adaptation is indicated by the impulse in the end-to-end delay right after 10.0 second (see Figure 5.18, 5.24 or 5.30).

Furthermore, the improvement in packet loss rate inside the network is evident when comparing the corresponding plots for non-flow-controlled and flow controlled cases. The simulation results are summarized in Table 5.7. With the proposed flow control mechanism, packet losses rarely occur. On the other hand, the average packet loss no longer changes exponentially with respect to the offered traffic in the network.

Simulation Case	Non-Flow-Controlled	Flow Controlled
Maximum loading	14.95%	0.10%
Medium loading	0.42%	0.02%
Minimum loading	0.13%	0.05%

Table 5.7. Average packet loss inside the network.

Lastly, the improvement in average TCP throughput is dramatic under heavy loading conditions, as shown in Table 5.8. As expected, if the network is overloaded by non-flow-controlled traffic such as UDP, flow controlled traffic such as TCP will suffer from poor throughput performance. In particular, if a TCP stream is initiated during a period when the network is congested, its throughput will be relatively small if other traffic sources do not yield. This is the case for the simulations with non-flow-controlled packet video sources at maximum and medium loading conditions. Also as expected, the TCP throughput can be much improved if most of the packet video sources are flow controlled in a way that is "compatible" with the TCP. From the simulation results, it is clear that the traffic generated by the proposed flow control mechanism coupes well with the TCP traffic in the sense that it shares network bandwidth fairly with the TCP.

Simulation Case	Non-Flow-Controlled	Flow Controlled
Maximum loading	12 kB/sec	669 kB/sec
Medium loading	350 kB/sec	794 kB/sec
Minimum loading	920 kB/sec	779 kB/sec

Table 5.8. Maximum time-averaged TCP throughput.

Under the current implementation, each flow controlled packet video source can only back off to some minimum output threshold. The implicit assumption here is that the quality of the video delivered will be unacceptable if the output bit-rate falls below that minimum

threshold value. Such a direct consequence of this restriction, congestion collapse can occur if the total traffic overwhelms some bottleneck link even after all adaptive packet video sources have fully backed off. Although simple remedies can be devised to alleviate this problem (e.g., each video session is required to terminate at some probability that increase over time if the congestion persists), no end-to-end solution will produce results that satisfy all traffic sources. Some form(s) of network support (e.g., prioritized transmission services) will be needed. The design of a network service or mechanism that would solve the above problem effectively is outside the scope of this thesis.

In fine tuning the performance of the adaptive system, several parameters have been observed to have strong influence on the system behavior. The dynamic range of acceptable output bit-rates of the encoder $[R_{MIN}, R_{MAX}]$ determines the extent of the adaptation. The rate of the adaptation is affected by Δ_R. In order to facilitate the rate-control stabilization, the value of Δ_R should be small comparing to the allowable output bit-rates, typically on the order of $0.01 \cdot (R_{MAX} - R_{MIN})$. By in large, the effectiveness of the system is dictated by the three flow control parameters, ρ, γ_1 and γ_2. In general, ρ should be fixed to a value between 0.5 and 1.0 to enforce the slow window adjustment policy. This value should be further determined by the desired rate of convergence. On the other hand, γ_1 and γ_2 should be set adaptively. If they are fixed to some large values, the flow control mechanism may exhibit oscillatory behaviors in adjusting the window size. If they are fixed to some small values, the flow control mechanism may not open up the window size fast enough, thus losing throughput to other traffic sessions (e.g., TCP). Only when they are set adaptively can they keep the flow control out of the undesirable operation regions.

Chapter 6

Conclusion

6.1 Concluding Remarks

In this thesis, an adaptive packet video system has been proposed and implemented that consists of a wavelet-based multiresolution video codec and an end-to-end flow control mechanism. The overall goal is to maximize the perceptual quality of the video delivered over packet-switching networks that provide no more than the best-effort service.

Through extensive simulations of various network settings, the proposed adaptive packet video system has been shown to meet its design objectives successfully. The system is very effective in terms of reducing end-to-end network delay and packet loss rate inside the network under heavy loading conditions. This is shown to be the case even for networks with large delay-bandwidth products. The system optimizes that the perceptual quality of a packet video stream at all times by using an encoder that discards information according to a loss-preference model derived from the human visual system. In addition, the traffic generated by this adaptive system that coexists well with other flow-controlled traffic and achieve fair sharing of the network resources.

Due to the time constraint, this thesis can only address some of the most important areas of an adaptive packet video system. There are still a number of networking and coding issues

that need to considered and resolved. In addition, there are several improvements that should be incorporated into the current system. These issues are detailed in the following section.

6.2 Future Work

Currently, the wavelet-based codec uses intra-frame coding. That is, it encodes each video frame individually. This approach has the advantage of localizing a transmission error to a single frame (rather than propagating an error over several frames). However, there is a price associated with this independence. With intra-frame coding, the amount of compression achievable for a given play-back quality is moderate. Much higher compression ratio can be attained by using inter-frame coding. While compression gain is not the most important objective in a packet video codec system design, higher compression should be used if it does not lead to performance degradation. Therefore, inter-frame coding schemes (i.e., ones that exploit inter-frame correlation and remove redundancy among several adjacent frames) may worth investigating.

Furthermore, the present wavelet-based codec only approximates constant quality coding. Because the statistical distribution of wavelet coefficients is roughly the same for many types of images, a set of quantizers can be constructed that yield roughly the same amount of distortion for each video frame. The intra-frame codec implemented in this thesis uses one such set of quantizers. However, for an inter-frame codec, the statistics of wavelet coefficients are very different for video frames depending on whether they are reference frames or interpolated frames. Fast quantization techniques for this type of codec that preserves constant quality of a coded video stream need to be examined.

The proposed flow control mechanism is very effective in reducing packet loss rate inside the network. However, it does not fully control network delay. The current receiver

plays back video according to a rigid time line determined by a frame-rate of the source video. So, due to momentary surge in the network delay, packets arrived at the receiver can still miss their play-back points in time. The quality of the received video can be improved if the play-back points can be set adaptively to reduce packet drop rate at the receiver. This technique should be further considered.

The most difficult challenge ahead is dealing with multicast video distribution. Up until now, almost all flow control mechanisms have been designed to handle pairwise data transfer sessions. Multicast video presents several much harder problems comparing to unicast video. Traffic in the network grows linearly the number of senders. Data packets generated in a multicast session are sent to all receivers. Any effective flow control mechanism for multicast video must consider information produced in both sending and feedback directions.

One of the most important aspects of a flow control mechanism for multicast video is to be able to adapt in such a way that would maintain relatively high quality video (as received by most receivers) under heavy loading conditions. Upon congestion, it is natural for a multicast flow control mechanism to adapt to the worst-performing receiver (i.e., the lowest common denominator) [5]. Such a mechanism cannot be considered as effective because the adaptation process essentially has only two states, one for sending at full rate and one for sending at the lowest rate dictated by the worst-performing receiver. On the other hand, a multicast flow control mechanism should not adapt to the best-performing receiver either without network assistance at some level (e.g., provision of prioritized transmission). The operating point should lie somewhere between the two extremes. Therefore, it is useful for senders to obtain information on both the worst- and the best-performing receivers. They can subsequently use this information to determine the appropriate traffic levels. The worst- and the best-performing receiver candidates should be re-elected periodically on a relative large time-scale (e.g., once

every few minutes). Another advantage of using only those pieces of information on the worst- and the best-performing receivers is that the amount of feedback information remains constant regardless of the size of a multicast group. This is an important feature because any flow control mechanism for video will require periodic updates that are on a time-scale of every several frames. In contrast, feedback information packets could add significant load to the network if they grow at least linearly with the number of receivers in a multicast group.

By utilizing the information on the worst- and the best-performing receivers, each sender can decide what would be the appropriate output traffic level. It may also be able to improve its adaptation algorithms by incorporating the knowledge on the current size of the sender group and that of the receiver group. The design of a multicast flow control mechanism that uses the above approach needs further investigation.

Appendix A

Wavelet Theory

A.1 Notation

Let $\boldsymbol{Z}$ and $\boldsymbol{R}$ denote the set of integers and real numbers respectively. $\boldsymbol{L}^2(\boldsymbol{R})$ denotes the vector space of measurable, square-integrable one-dimensional functions $f(x)$. For $f(x) \in \boldsymbol{L}^2(\boldsymbol{R})$ and $g(x) \in \boldsymbol{L}^2(\boldsymbol{R})$, the inner product of $f(x)$ and $g(x)$ is written as

$$\langle f(u), g(u) \rangle = \int_{-\infty}^{+\infty} f(u)\overline{g(u)}du.$$

The convolution of $f(x)$ and $g(x)$ is written as

$$\begin{aligned} f(x) * g(x) &= \big(f(u) * g(u)\big)(x) \\ &= \int_{-\infty}^{+\infty} f(u)g(x-u)du. \end{aligned}$$

A.2 Wavelet Functions

A wavelet family are the set of functions generated from a single function $\psi \in \boldsymbol{L}^2(\boldsymbol{R})$[A.1], by dilations and translations [7, 19, 31, 36]

$$\psi^{a,b}(x) = |a|^{-1/2}\psi\left(\frac{x-b}{a}\right) \tag{A.1}$$

A.1 For now, we assume that ψ is a function with a one-dimensional variable x.

where a is a dilation coefficient and b a translation coefficient. The *mother wavelet* ψ must satisfy the condition $\int \psi(x)dx = 0$.

To represent a function $f \in \boldsymbol{L}^2(\boldsymbol{R})$ as a superposition of wavelets $\psi^{a,b}$, f is projected onto the wavelet basis functions by taking the inner-products $\langle f, \psi^{a,b} \rangle$ over all a's and b's. The mapping $f \rightarrow \langle f, \psi^{a,b} \rangle$ is called continuous wavelet transform. f can be synthesized from this set of transform coefficients as the integral of wavelets $\psi^{a,b}$ weighted by the coefficients. For DSP applications, a discrete representation of f is introduced to replace the integral operation with summation in the signal synthesis. This representation can be realized by restricting the domains of a and b to $a = a_0^m$ and $b = nb_0a_0^m$ respectively, with $m, n \in \boldsymbol{Z}$ and $a_0 > 1$ and $b_0 > 0$ fixed. The wavelets constructed with the above values of a and b are given by

$$\psi_{m,n}(x) = \psi^{a_0^m, nb_0a_0^m}(x) = a_0^{-m/2}\psi(a_0^{-m}x - nb_0). \tag{A.2}$$

A.3 Orthogonal Wavelets

A particularly interesting class of wavelets are the *orthogonal wavelets*. For $a_0 = 2$ and $b_0 = 1$, there exist special choices of ψ such that the wavelets $\psi_{m,n} = 2^{-m/2}\psi(2^{-m}x - n)$ constitute an orthonormal basis. Various orthogonal wavelets ψ have been constructed by Meyer [35], Lemarié [30], Battle [4], and Daubechies [18]. With these wavelets, an arbitrary function $f \in \boldsymbol{L}^2(\boldsymbol{R})$ can be represented as

$$\begin{aligned} f(x) &= \sum_{m,n} \langle f(u), 2^{-m/2}\psi(2^{-m}u - n) \rangle \cdot 2^{-m/2}\psi(2^{-m}x - n) \\ &= \sum_{m,n} \left(f(u) * 2^{-m/2}\psi(-2^{-m}u) \right)(2^m n) \cdot 2^{-m/2}\psi(2^{-m}x - n). \end{aligned} \tag{A.3}$$

Orthogonal wavelet transforms can be better explained in the context of multiresolution analysis. In multiresolution analysis, *two* functions are used: the mother wavelet ψ and a

scaling function ϕ. The reason for introducing this new function ϕ will become apparent shortly. Like the wavelet basis functions $\psi_{m,n}$, $\phi_{m,n}$ are the dilated and translated versions of the scaling function ϕ. $\phi_{m,n}(x) = 2^{-m/2}\phi(2^{-m}x - n)$ (for each fixed m) forms an orthonormal basis.

To understand the relationship between ϕ and ψ and to the multiresolution analysis, two spaces need to be defined. Let V_m denote the space spanned by the $\phi_{m,n}$ and W_m denote the space spanned by the $\psi_{m,n}$, for each value of m. The spaces V_m over m describe successive approximation spaces at resolution 2^m,

$$\cdots \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \subset \cdots$$

The multiresolution approximation property of the spaces V_m guarantees that $f(x)$ is in V_m if and only if $f(2x)$ is in V_{m-1}. W_m is the orthogonal space complementing V_m in V_{m-1}, i.e., a space in V_{m-1} that satisfies the relations

$$W_m \perp V_m$$

and

$$\begin{aligned} V_{m-1} &= V_m \oplus W_m \\ &= W_m \oplus W_{m+1} \oplus \cdots \oplus W_{+\infty}. \end{aligned}$$

This means that W_m contains the "detail" information lost when going from an approximation of f at resolution 2^{m-1} to the coarser approximation at resolution 2^m. Consequently, $\oplus W_m = \boldsymbol{L}^2(\boldsymbol{R})$.

An orthogonal scaling function is a function ϕ such that the set $\{\phi(x-n) \mid n \in \boldsymbol{Z}\}$ is an orthonormal basis of V_0. The collection of functions $\{\phi_{m,n} \mid n \in \boldsymbol{Z}\}$ forms an orthonormal basis of V_m. Since $\phi \in V_0 \subset V_{-1}$, a sequence (h_k) exists such that the scaling function satisfies

$$\phi(x) = 2\sum_k h_k \phi(2x-k). \tag{A.4}$$

This functional equation is called the *refinement equation*, *dilation equation* or *two-scale difference equation*. The scaling function is uniquely defined by this equation and the normalization,

$$\int_{-\infty}^{+\infty}\phi(x)dx = 1. \tag{A.5}$$

It is interesting to note that, in many cases, an explicit expression for ϕ cannot be solved for. Instead, one works directly with the h_k.

Likewise, an orthogonal wavelet function is a function ψ such that the set $\{\psi(x-n) \mid n \in \mathbf{Z}\}$ is an orthonormal basis of $W_0 \subset V_{-1}$. The collection of functions $\{\psi_{m,n} \mid n \in \mathbf{Z}\}$ forms an orthonormal basis of W_m. Since the wavelet ψ is an element of V_{-1}, a sequence (g_k) exists such that

$$\psi(x) = 2\sum_k g_k \phi(2x-k). \tag{A.6}$$

In addition, the wavelet is required to have a vanishing integral, i.e.,

$$\int_{-\infty}^{+\infty}\psi(x)dx = 0. \tag{A.7}$$

Again, one frequently works directly with the g_k since ψ is often not available in closed form.

Since $V_{m-1} = V_m \oplus W_m$, a function $v_{m-1} \in V_{m-1}$ can be uniquely represented as the sum of a function $v_m \in V_m$ and a function $w_m \in W_m$

$$\begin{aligned}\sum_n \chi_{m-1,n}\phi_{m-1,n}(x) = v_{m-1}(x) &= v_m(x) + w_m(x) \\ &= \sum_l \chi_{m,l}\phi_{m,l}(x) + \sum_l \mu_{m,l}\psi_{m,l}(x).\end{aligned}$$

The decomposition step of the wavelet transform then becomes

$$\chi_{m,l} = \left\langle v_{m-1}, \tilde{\phi}_{m,l} \right\rangle = \sqrt{2} \left\langle v_{m-1}, \sum_k \tilde{h}_{k-2l}\, \tilde{\phi}_{m-1,k} \right\rangle$$
$$= \sqrt{2} \sum_k \tilde{h}_{k-2l}\, \chi_{m-1,k},$$
$$\mu_{m,l} = \sqrt{2} \sum_k g_{k-2l} \chi_{m-1,k}.$$

Similarly, the reconstruction step becomes

$$\chi_{m-1,k} = \sqrt{2} \sum_l h_{k-2l}\, \chi_{m,l} + \sqrt{2} \sum_l g_{k-2l}\, \mu_{m,l}.$$

These formulae define the analysis and synthesis of a transform called the *fast wavelet transform.*

From equations (A.4) through (A.7), one can prove that, for exact reconstruction, the orthogonal wavelet is derived from the orthogonal scaling function by

$$g_k = (-1)^k \overline{h_{1-k}}. \tag{A.8}$$

Equation (A.8) shows that, once the sequence (h_k) is determined, both the scaling function and the wavelet are completely characterized.

Most orthonormal wavelet bases have infinite support, which means that the sequence (g_k) has an infinite number of elements. Daubechies [18] gives the construction rules for finding ψ with finite support. One of the most widely used wavelets with finite support is derived from the scaling function D_4 [18], whose translates can perfectly reproduce any function of the form $ax+b$. Applying equation (A.8) to the scaling function D_4, the sequence (g_k) is shown to take on the values $g_0 = (3+\sqrt{3})/4$, $g_1 = -(1+\sqrt{3})/4$, $g_2 = (1-\sqrt{3})/4$, and $g_3 = -(3-\sqrt{3})/4$. The scaling function D_4 and the wavelet W_4 are plotted in Figure A.1-a and A.1-b respectively.

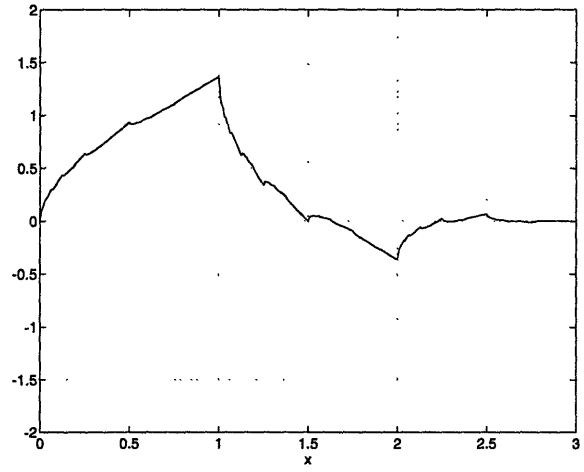

Figure A.1-a. Scaling function D_4

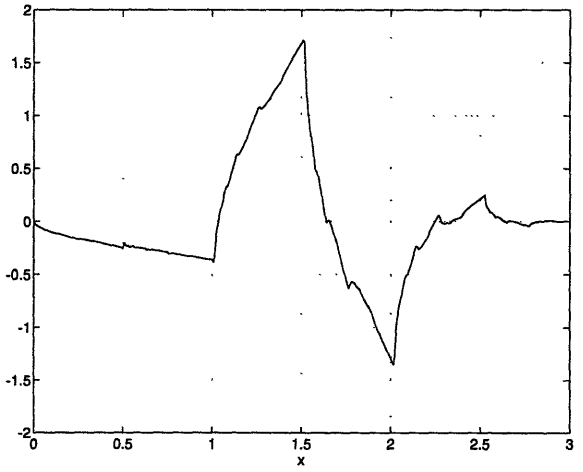

Figure A.1-b. Wavelet W_4

Note that these two functions have finite support, but neither has symmetry. The orthogonality property puts strong constraints on the construction of wavelets. For orthogonal wavelets, it has been shown that there are no nontrivial symmetric wavelets with both finite support and exact reconstruction [46]. The only exception is the Haar basis, which is a piecewise constant function and therefore gives poor approximation to most smooth functions. One can preserve the symmetry of the wavelet by relaxing the orthogonality requirement. This generalization leads to a class of wavelets known as the *biorthogonal wavelets.*

A.4 Biorthogonal Wavelets

Biorthogonal wavelets have recently been constructed independently by Cohen, Daubechies and Feauveau [11] and by Herley and Vetterli [52]. Biorthogonal wavelets no longer require orthogonality of spaces V_m and W_m. Instead, orthogonality is maintained between V_m and a space $\tilde{W}_m$ spanned by a dual wavelet $\tilde{\psi}$ and between W_m and a space $\tilde{V}_m$ spanned by a dual scaling function $\tilde{\phi}$. That is

$$W_m \perp \tilde{V}_m, \quad \tilde{W}_m \perp V_m \quad \text{and} \quad W_m \perp \tilde{W}_{m'} \text{ for } m \neq m'.$$

A dual multiresolution analysis[A.2] can be generated by the two scaling functions ϕ, $\tilde{\phi}$ and the two wavelets ψ, $\tilde{\psi}$. To form a multiresolution analysis, the dual functions must satisfy

$$\tilde{\phi}(x) = 2\sum_k \tilde{h}_k \tilde{\phi}(2x-k) \tag{A.9}$$

and

$$\tilde{\psi}(x) = 2\sum_k \tilde{g}_k \tilde{\phi}(2x-k) \tag{A.10}$$

with similar constraints as given by equations (A.5) and (A.7). Note that the role of the primary (i.e., ϕ and ψ) and the dual functions (i.e., $\tilde{\phi}$ and $\tilde{\psi}$) can be interchanged. Also, as in the case of orthogonal wavelets, one almost always works directly with $\tilde{h}_k$ and $\tilde{g}_k$ in the multiresolution analysis.

To derive the relationship between the coefficient sequences (h_k), (g_k), $(\tilde{h}_k)$, and $(\tilde{g}_k)$, an exact reconstruction condition is assumed. The exact reconstruction requires that

$$\tilde{g}_k = (-1)^k \overline{h_{1-k}}, \tag{A.11}$$

and

$$g_k = (-1)^k \overline{\tilde{h}_{1-k}}. \tag{A.12}$$

An example of a biorthogonal wavelet is a variant of the "spline filter" (i.e., the scaling function is a variant of a B-spline function) [1]. The particular wavelet chosen for this thesis has the "smallest" support in its family, with polynomial degrees of 4. It has been shown to be well suited for image coding applications [2, 3]. The filter coefficients are listed in Table A.1 and the scaling function and wavelet are shown in Figure A.2-a and A.2-b respectively.

A.2 Even though the scaling function and wavelet are not orthogonal, the multiresolution analysis can still be orthogonal. A biorthogonal scaling function and wavelet that generate an orthogonal multiresolution analysis are said to be *semiorthorgonal.*

k	0	±1	±2	±3	±4
$2^{-1/2}h_k$	0.602949	0.266864	-0.078223	-0.016864	0.026749
$2^{-1/2}\tilde{h}_k$	0.557543	0.295636	-0.028772	-0.045636	0

Table A.1. Filter coefficients for the spline variant with polynomial degrees of 4.

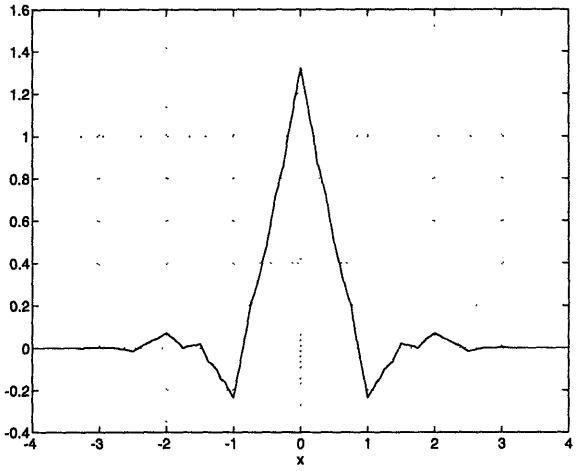

Figure A.2-a. Scaling function ϕ

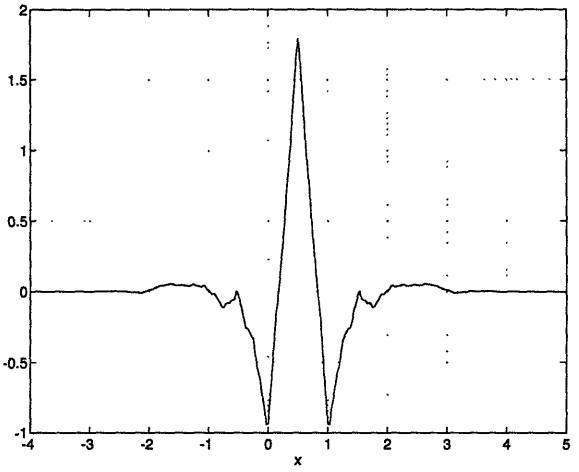

Figure A.2-b. Wavelet ψ

This appendix only provides a brief introduction of the basic theory on wavelets. Detailed studies of the mathematical properties and construction rules of orthogonal and biorthogonal wavelets can be found in such sources as [7, 19, 36, 48, 49, 13, 25]. In addition, wavelet applications are covered in [8, 41].

Appendix B

Source Code

For soft-copies of the source code used for this thesis, please contact the author at ygu@poppy.lcs.mit.edu.

Bibliography

[1] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image Coding Using Vector Quantization in the Wavelet Transform Domain," in *Proceedings of IEEE ICASSP*, Albuquerque, pp. 2297-2300, 1990.

[2] M. Antonini, M. Barlaud and P. Mathieu, "Image Coding Using Lattice Vector Quantization of Wavelet Coefficients," in *Proceedings of IEEE ICASSP*, Toronto, Canada, pp. 2273-2276, 1991.

[3] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image Coding Using Wavelet Transform," *IEEE Transactions on Image Processing*, Vol. 1, No.2, pp. 205-220, April 1992.

[4] G. Battle, "A Block Spin Construction of Wavelets, Part I: Lemarié functions," *Communications on Mathematics and Physics*, Vol. 110, pp. 601-615, 1987.

[5] J.-C. Bolot, T. Turletti and I. Wakeman, "Scalable Feedback Control for Multicast Video Distribution in the Internet," preprint.

[6] R. D. Buda, "Some Optimal Codes Have Structure," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 6, pp. 893-899, August 1989.

[7] C. K. Chui, *An Introcution of Wavelets*, Academic Press, 1992.

[8] C. K. Chui, editor, *Wavelets: A Tutorial in Theory and Applications*, Academic Press, 1992.

[9] D. D. Clark and D. L. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," in *ACM SIGCOMM 1990 Symposium*, 1990.

[10] D. D. Clark, S. Shenker and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism," in *ACM SIGCOMM 1992*, 1992.

[11] A. Cohen, I. Daubechies and J. C. Feauveau, "Biorthogonal Bases of Compactly Supported Wavelets," *AT&T Bell Laboratory Technical Report*, TM 11217-900529-07, 1990.

[12] A. Cohen and I. Daubechies, "Non-Separable Bidimensional Wavelet Bases," preprint, AT&T Bell Laboratories, 1991.

[13] J. M. Combes, A. Grossmann and Ph. Tchamitchian, editors, *Wavelets: Time-Frequency Methods and Phase Space*, Springer-Verlag, 1989.

[14] J. H. Conway and N. J. A. Sloane, "Fast Quantizing and Decoding Algorithms for Lattice Quantizers and Codes," *IEEE Transactions on Information Theory*, Vol. 28, No. 2, pp. 227-232, March 1982.

[15] J. H. Conway and N. J. A. Sloane, "A Fast Encoding Method for Lattice Codes and Quantizers," *IEEE Transactions on Information Theory*, Vol. 29, No. 6, pp. 820-824, November 1983.

[16] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*, Prentice-Hall, 1983.

[17] A. Croisier, D. Esteban and C. Galand, "Perfect Channel Splitting by Use of Interpolation/Decimation/Tree Decomposition Techniques," in *International Conference on Information Sciences and Systems*, Patras, pp. 443-446, August 1976.

[18] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," *Communications on Pure and Applied Mathematics*, Vol. 41, pp. 909-996, 1988.

[19] I. Daubechies, *Ten Lectures on Wavelets*, SIAM Publications, 1992.

[20] D. Esteban and C. Galand, "Application of Quadrature Mirror Filters to Split Band Voice Coding Schemes," in *Proceedings of IEEE ICASSP*, pp. 191-195, May 1977.

[21] J. J. Friedman, J. L. Bentley and R. L. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematics Software*, pp. 209-226, September 1977.

[22] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[23] M. Gilge and R. Gusella, "Motion Video Coding for Packet-Switching Networks — An Integrated Approach," *ICSI Technical Report TR-91-065*, International Computer Science Institute, University of California at Berkeley, 1991.

[24] V. Jacobson, "Congestion Avoidance and Control," in *Proceedings of SIGCOMM '88*, pp. 314-329, Stanford, CA, 1988.

[25] B. Jawerth and W. Sweldens, "An Overview of Wavelet Based Multiresolution Analyses," preprint, University of South Carolina, 1993.

[26] H. Kanakia, P. P. Mishra and A. Reibman, "An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport," in *Proceedings of SIGCOMM '93*, pp. 20-31, Ithaca, NY, 1993.

[27] G. Karlsson and M. Vetterli, "Subband Coding of Video for Packet Networks," *SPIE*, Vol. 27, No.7, pp. 381-393, July 1988.

[28] G. Karlsson and M. Vetterli, "Packet Video and Its Integration into the Network Architecture," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 739-751, June 1989.

[29] J. Kovacevic and M. Vetterli, "Nonseparable Multidimensional Perfect Reconstruction Filter Banks and Wavelet Bases for $\Re^n$," *IEEE Transactions on Information Theory*, Vol. 38, No. 2, pp 533-555, March 1992.

[30] P.-G. Lemarié, "Une Nouvelle Base d'Ondelettes de $\boldsymbol{L^2(R)}$," *Journal de Mathematiques Pures et Appliquées*, Vol. 67, pp. 227-238, 1988.

[31] P.-G. Lemarié, editor, *Les Ondelettes en 1989*, Springer-Verlag, 1990.

[32] Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, Vol. 28, pp. 84-95, January 1980.

[33] S. G. Mallat, "A Theory of Multiresolution Signal Decomposition: the Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 7, pp. 674-693, July 1989.

[34] S. G. Mallat, "Multifrequency Channel Decompositions of Images and Wavelet Models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 37, No. 12, pp. 2091-2110, December 1989.

[35] Y. Meyer, "Principle d'Incertitude, Bases Hilbertiennes et Algèbres d'Opérateurs," *Seminaire Bourbaki*, No. 662, 1985-1986.

[36] Y. Meyer, *Ondelettes et Opérateurs, I: Ondelettes, II: Opérateurs de Calderón-Zygmund, III:* (with R. Coifman), *Opérateurs Multilinéaires*, Hermann, Paris, English translation by Cambridge University Press, 1990.

[37] M. Nomura, T. Fujii and N. Ohta, "Basic Characteristics of Variable Rate Video Coding in ATM Environment," *IEEE Journal of Selected Areas in Communications*, Vol. 7, No. 5, pp. 752-760, June 1989.

[38] A. Parekh, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks," *Technical Report LIDS-TR-2089*, Laboratory for Information and Decision Systems, MIT, 1992.

[39] H. L. Resnikoff, "Wavelets and Adaptive Signal Processing," *Optical Engineering*, Vol. 31, pp. 1229-1234, June 1992.

[40] P. Romano Jr., *Vector Quantization for Spatiotemporal Sub-band Coding*, Master's Thesis, MIT, 1990.

[41] M. B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, editors, *Wavelets and their Applications*, Jones and Bartlett, 1992.

[42] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF Internet Draft, July 18, 1994.

[43] T. Senoo and B. Girod, "Vector Quantization for Entropy Coding of Image Subbands," *Vision and Modeling Group Technical Report 139*, The Media Laboratory, MIT, 1990.

[44] A. Singh, *Variable Resolution Video*, Master's Thesis, MIT, 1991.

[45] A. Singh and V. M. Bove Jr., "Multidimensional Quantizers for Scalable Video Compression," *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 1, pp. 36-45, January 1993.

[46] M. J. Smith and D. P. Barnwell, "Exact Reconstruction for Tree-Structured Subband Coders," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 34, pp. 434-441, 1986.

[47] J. Stöckler, "Multivariate Wavelets," in *Wavelets: A Tutorial in Theory and Applications*, pp 325-356, Academic Press, 1992.

[48] G. Strang, "Wavelets and Dilation Equations: A Brief Introduction," *SIAM Review*, Vol. 31, pp. 614-627, 1989.

[49] G. Strang, "Wavelet Transforms versus Fourier Transforms," preprint, MIT.

[50] P. P. Vaidyanathan, "Quadrature Mirror Filter Banks, M-Band Extensions and Perfect-Reconstruction Techniques," *IEEE ASSP Magazine*, pp. 4-20, July 1987.

[51] M. Vetterli, "A Theory of Multirate Filter Banks," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 35, pp. 356-372, March 1987.

[52] M. Vetterli and C. Herley, "Wavelets and Filter Banks: Relationships and New Results," in *Proceeding of IEEE ICASSP*, Albuquerque, 1990.

[53] Y. Wang and V. Ramamoorthy, "Image Reconstruction from Partial Subband Images and its Application in Packet Video Transmission," *Signal Processing: Image Communication*, Vol. 3, pp. 197-229, 1991.